University of Groningen
Faculty of Science and Engineering
Computing Science

Change Mining and Logging in Process-Aware Information Systems: A Literature Review

Hande Naz Turgay
August 2022

Abstract

Context: Change mining enables organizations to understand the changes that occurred in their business processes. This allows them to enhance their business processes and adapt to dynamic environments. Therefore, change mining is becoming a topic of interest for researchers, scholars, and practitioners. However, to the best of our knowledge, there has not been a study that summarizes the area so far to act as a guide to the people in the field.
Objective: This paper aims to investigate the literature in change mining and logging in process-aware information systems and provide an overview of the methods that are used in the publications.
Method: A literature review is conducted to identify and define methods to mine, store, and record changes in business processes. From 765 publications, we selected 6 papers related to changes in business process and extended the list to 9 papers by including the relevant articles referenced by the papers that we selected originally.
Results: We have identified two classes of change mining methods, two ways of recording the changes into change logs, five formats for change log representation, and four objectives to be learned from changes.
Conclusion: The literature review provides a summary of change mining and logging methods in process-aware information systems which support researchers to gain an overview of the topic and identify possible research gaps.

Keywords: Change Mining, Business Process Management, Change Logs, Business Processes, Process-Aware Information Systems, Literature Review

Contents

List of Figures

List of Tables

1 Introduction

Change mining helps organizations to examine the details of past changes in their business processes and enables them with information that may be useful to plan and evaluate possible future changes. The visibility of changes in business processes can help organizations to manage and optimize their systems and use of resources, which can be beneficial for them in the long term. Therefore, the area of research started to attract more interest from researchers, scholars, and practitioners. An introductory guide would be helpful to cover the current aspects of the field for such development. We believe that a literature review would serve this purpose well as, to the best of our knowledge, there has not been a literature review or any similar work in the field. Thus, in this paper, we have conducted a literature review on change mining and logging in process-aware information systems to provide a comprehensive overview of the current state of research for researchers, scholars, and practitioners in the field. The remainder of the section is dedicated to the introduction of the concepts that will be presented in the results. The remainder of the paper is organized as follows. In Section 2, the research methodology to conduct the literature review is explained in detail. Section 3 presents the results by synthesizing the information from the literature that we collected to answer the research questions. Section 4 provides a discussion on the results. Finally, Section 5 concludes the review.

1.1 Business Processes

Business processes can be defined as a series of activities that address a business objective in an organization and allow a business activity to be executed efficiently. Business process models are sets of relationships between business activities, which depict the possible sequences of events[1] that may occur on execution time. Business processes can be notated as BPMN (Business Process Model and Notation) diagrams, Petri nets, and UML (Unified Modeling Language) diagrams. Automation of business processes can be achieved with the use of BPMS (Business Process Management System) tools, workflow automation systems, ERP (Enterprise Resource Planning) systems, CRM (Customer Relationship Management) systems, or XML (Extensible Markup Language) business process languages [13]. Business process management (BPM) is a management approach that combines information technology with management sciences to model and enhance business processes [36].

The modification of a business process to meet new business requirements is called process change. Process changes can occur at run-time or build-time of a business process and result in the addition of new features and modifications to the process elements. Process changes must be analyzed and dealt with as the validity of the original business process models is affected by them.

[1] An event is the execution of a business activity.

1.2 Process Mining

Process mining is the extraction of information from event logs, which are stored in information systems, to explore, track, and enhance business processes. Process mining covers perspectives such as [38]:

1. *Control-Flow Perspective:* Focuses on the order of the activities and aims to acquire an accurate characterization of each possible path. The result can be expressed in process notations such as Petri nets, EPCs, BPMN, or UML activity diagrams.

2. *Organizational Perspective:* Focuses on information about resources, involvement of actors, and relationships among them. The aims of this perspective are:

   (a) Providing a structure to the organization through classification of the people in the organization by their roles and organizational units.

   (b) Presenting the social network.

3. *Case Perspective:* Focuses on properties of cases, which can be characterized by the path it follows through the execution of a process, by the actors that operate on it, or by the values of data elements contained in it.

4. *Time Perspective:* Focuses on the timing and frequency of events. Storage of events with timestamps enables discovery of bottlenecks, measurement of service levels, monitorization of resource usage, and estimation of execution times of the active process instances.

The input for process mining methods is event logs. In an event log, the information about how events are ordered should be available and each event should be connected to a process instance and a specific activity. In most event logs, the information about fields such as the performer of the event, the timestamp of the event, or data recorded with the event is present as well. There are three types of process mining which are [11]:

1. *Discovery:* Construction of a process model based on the information on an event log. It is relevant when an a-priori process model does not exist.

2. *Conformance:* Comparison between an a-priori process model and reality. It is used for the detection of deviations and presentation of their data. Outputs a diagnostic of conformance.

3. *Extension:* Enrichment of an a-priori process model through additions of new aspects or perspectives. Outputs a new process model.

1.3 Change Mining

In business processes, management of changes can be enhanced with the help of process mining, which is known as change mining. Changes in traces[2] of a business process can be detected with the utilization of event logs or change logs in change mining.

A change is an unforeseen event in a business process event log. Changes occur by "applying a sequence of change primitives or operations to the respective process graph" [32]. Changes can be applied to:

1. *Business Processes:* This can be the addition, deletion, or modification of an activity, resource, or data of a process [34].

2. *Configurable Business Processes:* This can be the addition, deletion, or modification of a variation point, or its variants.

Changes are recorded on change logs that are created automatically in systems such as process-aware information systems [10]. If a system does not provide a change log, it can be created from event logs by change mining [14] [19].

Change mining focuses on the adaptation of processes in order to fulfill the requirements that change during the process execution. Change mining is performed on event logs to extract data. A single event log or a collection of event logs can be used for the input of change mining. A single event log is used when changes in a business process are to be discovered. When changes in process variants of a configurable process model are to be discovered, a collection of event logs are used. Both of the inputs result in a change log. A change log is a file that consists of parts referred to as change instances. Change logs provide information about the attributes of change which can help the business process designers to optimize a business process. The analysis of change logs can offer important information on changes that happened during the execution phase of a process, as change logs include all the details about changes in a business process. This can help organizations to understand the characteristics of past changes on business processes [11] and evaluate the effect of the future changes before their application [21]. The manual analysis of change logs is a hard task and change mining simplifies the process of finding the required information by allowing rapid and effective detection of changes.

1.4 Configurable Business Process Models

Configurable business processes have appeared with the emergence of business process reuse [8]. A configurable business process model combines variants of a business process in a single model in order to simplify the management of business process variants [9]. Configurable business process models can be represented by modeling languages such as C-EPC [39] and C-BPMN [46], which

[2]Traces are parts of the event log that contain the execution information of a process instance.

allow the generation of process variants from a configurable process model.

Business process variants can be created from the configurable process model by configuring the variation points. Variation points are the places in which the variation happens in the business process model and each one has multiple alternatives called variants. A new process variant is generated through the assignment of variants to variation points [9]. There are three steps to manage variability in a configurable business process [8]:

1. Modeling
2. Configuring
3. Evolving

A business process or a family of business processes can evolve and change to fulfill new requirements [16]. These changes are registered as events on event logs which are used as a source for all process mining [38] and change mining [38] methods. There are two methods to generate a configurable business process model which are [14]:

1. *Creation from Scratch:* A configurable process model can be generated from process models by merging the variants of a business process. If there are no process models to start with, a configurable process model can be created from scratch.
2. *Creation by Process Mining:* A configurable process model can be created through the use of process mining methods on a collection of event logs.

1.5 Process-Aware Information Systems

An information system is a work system that uses information technology to capture, store, or modify information [7]. A process-aware information system (PAIS) is a special kind of information system that handles the management and execution of processes, which contain people, applications, and data sources based on process models and needs to be able to deal with ambiguity, unexpected situations and changes in environment [31] [37]. PAISs are commonly used in workflow management systems, case-handling systems, and enterprise information systems [37].

PAISs can be developed in two ways [10]. One way is to make a process support system where an organization builds its process support system from scratch with the purpose of supporting the processes. The other way is to configure a generic system where an organization needs to configure the system by specifying processes, applications, and organizational entities.

Many works have been conducted in order to make PAISs more flexible, which resulted in the emergence of adaptive process management approaches (e.g. ADEPT, CBRFlow, WASA) that allow users to dynamically change process models to address changed requirements. Adaptive PAISs enable the dynamic application of changes to different process aspects, such as control-flow

and data-flow, at multiple levels, such as process instance and process type. Particularly, the application of ad-hoc changes[3] at the instance level enables the adaptation of single process instances to dynamic conditions [30]. Usually, ad-hoc changes are recorded in change logs in adaptive PAISs [32] which, in turn, provides more meaningful log information than the log information that can be retrieved from traditional PAISs. However, only little work has been conducted to answer fundamental questions such as what can be learned from change logs, how to utilize them, and how to generate enhanced process models by using change logs.

2 Research Methodology

In this paper, a literature review is conducted based on the scientific guidelines specified in [22]. The guideline for the research methodology of this paper is outlined in Figure 1, which will be discussed in detail. The research methodology that will be used to conduct the literature review will assist us to specify the research question, the research protocol, and the selection protocol.

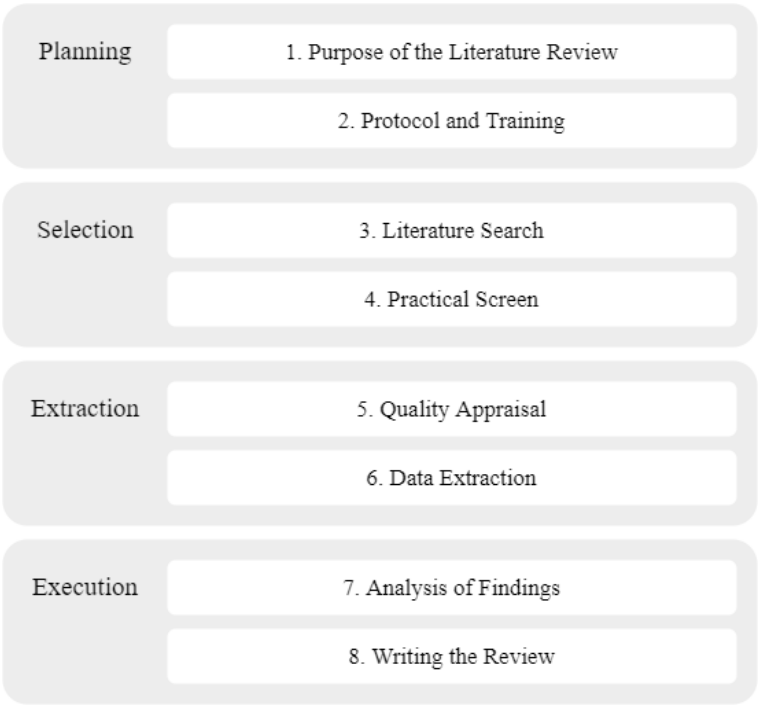

Figure 1: Research Methodology (Following [1])

[3]A process change that is applied to a process instance to deal with exceptions and unexpected events.

We needed to define a clear protocol for research after we decided on the purpose of the review. It is important that the criteria are defined clearly before the review is conducted, as a poorly conducted literature review can result in misconceptions. Then, we conduct an extensive literature search based on the protocol. It should be a comprehensive and transparent search to allow other researchers to replicate and reproduce the results. Multiple search engines should be used in a literature review to ensure that all the relevant data is collected as there are differences between literature engines. Finally, we extract and synthesize the data from the data set to write the review.

The guideline in Figure 1 has eight elements that are required for a literature review as specified in [27]. All of the steps that are mentioned are essential to the review and they will ensure that the review is scientifically rigorous [27].

1. *Purpose of the Literature Review:* The reviewer needs to clearly identify the purpose and goal of the review so that the contents of the review is clear to the readers.

2. *Protocol and Training:* The reviewer is required to have a detailed protocol to ensure consistency in the conduction of the review.

3. *Literature Search:* The reviewer should be clear in the details of the search and be able to justify its comprehensiveness.

4. *Practical Screen:* The reviewer needs to explain which studies were included and excluded in the review and be able to clearly explain the reason behind why they were included and excluded.

5. *Quality Appraisal:* The reviewer needs to define the criteria to determine which publications are of sufficient quality to be included in the review synthesis.

6. *Data Extraction:* The reviewer needs to extract the information from each of the publications that were chosen to be included in the review.

7. *Analysis of Findings:* The reviewer needs to combine the information that is extracted from the publications by using the appropriate techniques.

8. *Writing the Review:* The process of a literature review needs to be reported besides the findings of the publications to ensure that the results can be reproduced independently.

2.1 Research Questions

The aim of this paper is to review and evaluate the research in change mining and logging in process-aware information systems. To achieve this, we formulated research questions by using the PICOC (Population, Intervention, Comparison, Outcome, Context) criteria that are suggested by [22] as can be seen in Table 1. In the criteria, population refers to the subjects that are affected by the intervention. Intervention refers to a tool that is directed at an issue.

Comparison refers to another tool that is compared against the intervention. Outcome refers to factors that emerge as a result of the intervention. Context refers to conditions that the intervention takes place.

Criteria	**Description**
Population	Literature on change mining and logging in process-aware information systems.
Intervention	Extraction of methods and techniques that are used in change mining and logging in process-aware information systems.
Comparison	N/A
Outcome	Describe the approaches used in change mining and logging in process-aware information systems.
Context	Research to synthesize peer-reviewed literature.

Table 1: PICOC Criteria

As a result, we formulated the following research question by following the criteria in Table 1:

RQ1. *What primary studies have been published in the area of process-aware information systems that focus on change mining and logging?*

Since we want to focus on the content of the publications that we will find as a result of RQ1, we formulated four more research questions to help us to focus the research and investigate the technologies and methods that have been proposed in the publications.

RQ2. *How do we mine changes from business processes?*

RQ3. *How are changes that happened in business processes recorded?*

RQ4. *How are changes that happened in business processes stored?*

RQ5. *What do we learn from the recorded change information?*

RQ2 will help us to explore various change mining methods. RQ3 and RQ4 will provide us with information on how changes applied to processes are recognized, organized, and stored. RQ5 will helps us to discover different outputs of change mining approaches.

2.2 Literature Research

Our approach for the literature search was to find as many peer-reviewed publications as possible that can answer the research questions and narrow down the number of publications by using the selection criteria as can be seen in Figure 2. To obtain the scientific publications for the review, we ran a manual horizontal search. The digital libraries of IEEE Xplore, Web of Science, SciVerse Scopus, ACM Digital Library, and Google Scholar were used to perform a horizontal search. The search strings that are used to carry the research are provided in Section 2.2.1.

We started by searching for papers that are in the scope of our research. We narrowed down the initial set of papers by reading the title, abstract, and keywords on each of them. The general idea was to strictly keep the papers that used the keywords that are mentioned in Section 2.2.1 in the title, abstract, or keywords. Then, we proceeded to remove the duplicate entries from the set of papers that we obtained. To remove the duplicate entries, we followed [24] which suggests that if a publication that has the same title and the same authors has been published in more than one digital library, they will be considered as one study and if the same authors published multiple publications on the same topic, the most recent one will be considered. Some of the papers were on specific areas of change in process-aware information systems so they had to be read in full to be categorized as relevant. Hence, we scanned the whole paper in the next step to further narrow down the set of papers and minimize the chance of removing a relevant publication. Publications that included any of the keywords but concentrated on areas that are not business process management and/or computer science were excluded. Furthermore, publications that focused on change propagation, concept drift, and process model discovery were excluded as they were not exactly the focus of our research, although the publications contained the search words. Finally, we were left with 6 publications that focused on change mining and logging in process-aware information systems, which we extended to 9 publications by examining the references of them.

After we determined the final set of papers, we designed a database to store the bibliographic information that we need from the publications. The bibliographic information that we collected consists of the following:

1. Electronic Database: The database the publication was retrieved from.
2. Title: The title of the publication.
3. Authors: The authors of the publication.
4. Publication Year: The year that the publication was published.
5. Publication Source: The journal, conference, or workshop that the publication was published in.
6. Publication Type: The type of the publication.

Figure 2: Literature Search and Selection Process (Following [25] [45])

2.2.1 Search Strings

We start by defining a search string to conduct the literature review. To define the search string, we followed the guidelines suggested by Kitchenham et al. [23] and as a result of this:

(a) "process-aware information systems", "change", "mining" and "logging" are the main search terms that were derived from the research question.

(b) "business processes", "service composition", "service orchestration", "adaptation", and "workflows" are technical synonyms that were derived from the main search terms.

(c) The Boolean AND was used to connect some of the search terms in (a) and (b) to be able to narrow down the search results.

(d) The Boolean OR was used to incorporate some of the search terms in (a) and (b) to be able to broaden the search results.

(e) Some of the keywords in (a) and (b) were put in quotation marks to search for an exact match to be able to narrow down the search results.

Finally, we combined all the terms and the guidelines that are mentioned above which led us to the following search string:

((adaptation OR change) AND (logging OR mining) AND (workflow OR "business process" OR "service composition" OR "service orchestration" OR "process-aware information systems"))

The keywords "information systems" and "processes" were considered in the initial string but they were removed after we observed that they brought irrelevant results. The keyword "ad-hoc change" was removed from the string as it did not change the result of the search.

2.2.2 Search Sources

After we defined the search strings, the search sources had to be determined. It is important that we have a list of all the search sources that we used as this will ensure that other researchers, scholars, and practitioners will get the same search results from the search sources with the search strings that we use for the literature review. Table 2 shows the list of electronic databases that were used for the research of the review. The databases below were chosen as suggested by [22] [29] because they cover most of the scientific publications in the field of computer science and guarantee to provide the confidence level for the inclusion of all the required primary studies. We applied the criteria defined in Table 3 to all the digital libraries that are listed in Table 2.

Database	**Institution**	**Abbreviation**
ACM Digital Library	ACM	ACM
Web of Science	Thomson Reuters	ISI
SciVerse Scopus	Elsevier	ELSV
IEEE Xplore	IEEE	IEEE
Google Scholar	N/A	N/A

Table 2: List of Electronic Databases

The ACM Digital Library provides the ability to extend the search by using the ACM Computing Literature Library which helped us to enrich the publications that we collected. Google Scholar brought up over 15000 results when we used the search string and narrowed down the search to publications that were

published between the years 2000-2022. We decided to only cover the first 100 results that were listed according to their relevance to the search. SpringerLink was initially considered in the list of libraries to conduct the search as suggested by the guideline of Kitchenham et al. [22] but the results that we obtained were either irrelevant to the research or already found in the other digital libraries.

2.3 Selection Criteria

We had to ensure that the set of papers that we collected was relevant to the topic. Therefore, we determined a set of criteria which can be seen in Table 3 to obtain the papers for the literature review. If a publication does not pass any of the criteria listed on Table 3, there will be no further evaluation and the publication will be excluded. Publications that pass all of the criteria will be subjected to further assessment to decide if they propose a solution for the research questions.

All of the digital libraries that are listed in Table 2 offered the option to exclude and include languages, publication types, publication years, and disciplines on the search which we utilized by using the criteria that are defined in Table 3. The use of the criteria helped us to find the key publications that would help us to answer the research questions and it ensured that the results are reliable and reproducible.

Our criteria focus on content and publication. Content-related exclusion was applied in Steps 3 and 5 of the search process and publication-related exclusion was applied in Steps 2 and 6 of the search process which can be seen in Figure 2.

Criteria	Description
Inclusion Criteria	Is the full-text of the study digitally accessible?
	Is the full-text of the study in English?
	Has the study been published between the years 2000 and 2022?
	Does the title, abstract, and keywords of the study comply with the search strings?
	Has the study been published in a scientific peer-reviewed source?
Exclusion Criteria	Is the study in an area other than business process management and/or computer science?
	Is the focus of the study not software systems, workflows, processes, or service compositions?

Table 3: Inclusion/Exclusion Criteria

2.4 Extending the Publication List

We wanted to extend the set of publications that we have collected from our research as they were not enough to answer the research questions that we have defined in Section 2.1. We decided to achieve such an extension by looking at the references of the publications that we have already collected, as we already knew that they contained the information that was relevant to our research.

The procedure that we used to extend the set of publications with the references was similar to the original procedure. Furthermore, the criteria that

we have defined in Table 3 were valid for the extension procedure as well. The difference was that we started by checking the citations on sections that had information about changes in business processes in the publications that were on our final set of papers. Then, we looked at the titles in the entirety of the bibliography of the publications. Initially, we kept all the publications with titles that complied with our search string. Then, we read the abstract, introduction, and conclusion to classify them as relevant. We fully read the ones that we choose as a result of this to further analyze. Finally, we decided on 3 papers from the references in order to present a comprehensive result.

2.5 Data Extraction and Synthesis

The publications that were collected had to be analyzed and structured to obtain the information that we needed to address the research questions. To collect the relevant information from the publications, we read the publications in detail and created lists of attributes related to each research question, and created a data extraction form. The list of identified attributes can be seen in Table 4.

In order to explore *RQ1 - What primary studies have been published in the area of process-aware information systems that focus on change mining and logging?*, we collected a set of common fields such as the title, publication year, publication source, and publication type of the publications.

In order to answer *RQ2 - How do we mine changes from business processes?*, we chose the attributes that are listed below:

1. *Adapted Process Mining Methods:* In this class, we include the research works which adapt process mining methods to mine changes from business processes.

2. *Novel Methods for Change Mining:* In this class, we include research works that present novel methods for change mining. By novel, we mean the methods that are not extensions of process mining methods.

In order to answer *RQ3 - How are changes that happened in business processes recorded?*, we chose the attributes that are listed below:

1. *Recording Individual Change Operations:* In this class, we include the research works that create change logs by recording each change operation.

2. *Process Model Comparison:* In this class, we include research works that compare two versions of a process model and extract the changes that the model has gone through in order to create a change log.

In order to answer *RQ4 - How are changes that happened in business processes stored?*, we chose the attributes that are listed below:

1. *ADEPT Change Log:* In this class, we include research works that use ADEPT change logs for change log storage.
2. *XML:* In this class, we include research works that use XML for change log storage.
3. *MXML:* In this class, we include research works that use MXML for change log storage. This includes works that convert other formats into MXML as a preparation step for different procedures as well.
4. *CSV:* In this class, we include research works that use CSV for change log storage.
5. *XES:* In this class, we include research works that use XES for change log storage, which is the standardized format for the representation of event logs.

In order to answer *RQ5 - What do we learn from the recorded change information?*, we chose the attributes that are listed below:

1. *Change Model:* In this class, we include research that creates a model of the mined changes. These models can be represented in different formats such as Petri nets or BPMN diagrams and show the dependencies between change operations.
2. *Change Trees:* In this class, we include research works that create a change tree from the mined changes in order to visualize the frequencies and dependencies of different change operations.
3. *Change Recommendations:* In this class, we include research works that focus on generating a list of recommended changes to be applied to processes.
4. *Change Rule:* In this class, we include research works that focus on the creation of change rules with the use of process change information and contextual information.

Research Question	**Attributes**
RQ1	Title, Publication Year, Publication Type, Publication Source
RQ2	Adapted Process Mining Methods, Novel Methods for Change Mining
RQ3	Recording Individual Change Operations, Process Model Comparison
RQ4	ADEPT Change Log, XML, MXML, CSV, XES
RQ5	Change Model, Change Trees, Change Recommendations, Change Rule

Table 4: Attributes for Data Extraction

Table 5 depicts the form[4] that we have used to extract data from our final set of publications. We have filled the fields in the form for each publication and read each publication a minimum of 3 times in order to check the validity and relevance of the extracted information.

Data Item	Aim
Title and Reference	Provide full title and publication details of the paper.
Aim of Research	Summarize what researchers aimed to achieve in the paper.
Relevant Research Questions	Identify which of the research questions are answered in the paper.
Relevant Attributes	Identify which of the attributes in Table 4 exist in the paper.
Research Contribution	Summarize contributions of the paper to the field.
Implementation Details (If Applicable)	Explain the use of tools, datasets, and techniques.
Conclusion	Summarize the conclusions that are stated by the authors.

Table 5: Literature Review Form

3 Results

This section will present the results of the literature review that we conducted. First, we will outline the bibliographic information about the publications from the final set. Then, we will display the findings that we have collected from the publications to answer the research questions.

3.1 Overview of Selected Publications

In this section, we provide an overview of the publications by presenting the publication types, publication sources, and publication years.

Table 6 gives an overview of the publication types of the selected publications and their publication years. The table shows that the set of publications consists of 5 conference papers and 4 journal articles that are published between the years 2000 and 2022. Table 7 provides the publication sources of the selected publications and their publication years. The table shows that the publications in the set are from 4 different journals and 5 different conferences.

	2022	**2021**	**2020**	**2015**	**2008**	**2006**	**Total**
Conference	2	0	1	1	0	1	5
Journal	0	2	0	1	1	0	4
Total	2	2	1	2	1	1	9

Table 6: Summary of Publication Years

[4]The form with the information that is extracted from the publications can be accessed at: `https://docs.google.com/spreadsheets/d/1ZFXsqRkkknaQHfENsWkoe0iECXFXJfd4LPCmMRmtLwg/edit?usp=sharing`

	Journals	Conferences
2022		IRASET'2022: 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology
		SIMPDA'2022: 5th International Symposium on Data-Driven Process Discovery and Analysis
2021	International Journal of Advanced Computer Science and Applications	
	Journal of Physics Conference Series	
2020		NISS2020: Proceedings of the 3rd International Conference on Networking, Information Systems & Security
2015	World Wide Web	ICSOC'2015: International Conference on Service-Oriented Computing
2008	International Journal of Business Process Integration and Management	
2006		BPM'06: Proceedings of the 4th International Conference on Business Process Management

Table 7: Summary of Publication Sources

3.2 Findings

In this section, we present the results that we have obtained from the publications with the data extraction form.

In [11], three challenges have been recognized for providing a framework for mining ad-hoc changes in adaptive process management systems which are:

1. Determination of runtime information that needs to be logged and the optimal representation of the selected information in order to achieve better mining results.

2. Development of advanced mining techniques which use change logs together with execution logs.

3. Integration of the developed mining techniques with the already existing adaptive process management technology.

Günther et al. propose two new methods in [11] for ad-hoc change mining in adaptive process management systems. The proposed methods are implemented as plug-ins in ProM and discover change knowledge from ADEPT change logs mapped to MXML [12] [40] format. They acquire an abstract change process as a Petri net model, which shows the changes made on instances of a specific

process type. In order to map ADEPT change logs into MXML, Günther et al. propose the use of ProMimport, which converts data from various systems to MXML [12]. Günther et al. state that change logs can be converted to and represented by MXML with slight modifications. For instance, the attributes that define a change event are not treated in the standard MXML format, which are [11]:

1. The type of change.
2. The subject that mainly got affected.
3. The syntactic context of change which can be defined as the elements in the process model that directly precede or follow the subject of the change.

Instead, they are recorded as attributes in the "Data" field in the MXML change log. The attributes of a change event may include a rationale field as well, which provides the incentive behind the change. The field that indicates the person who has applied the respective change is the originator field. The exact date and time of the change is indicated by the timestamp field. Change events can be seen as atomic operations and for that reason, the event type is "complete" by default. Process elements must be indicated for each change event in order to achieve backward compatibility of MXML change logs with the conventional process model algorithms. However, such information is unavailable as the change process does not adhere to an a-priori process model. Therefore, a combination of the change type and subject is used instead in order to uniquely identify classes of changes. Conversion of change log data to MXML allows the usage of conventional process mining algorithms on change logs, which are evaluated by the authors.

For the evaluation, Günter et al. used an extension of the ADEPT demonstrator and compared algorithms based on quality criteria. They have analyzed the α-Algorithm, the MultiPhase Miner, and the Heuristics Miner and found that these algorithms capture dependencies between change operations successfully for simple processes and a small set of change operations. However, the quality of the change processes decreases when different change operations are used and the complexity of processes is increased. In such a scenario, nonexistent dependencies are generated by the mining algorithms and change processes become less accurate. The main source of this problem is the fact that change logs are not as populated as enactment logs. Therefore, they contain little data to learn from.

The first method they propose is based on MultiPhase mining [41] and uses additional information related to the semantics of change operations. Günther et al. state that they chose to build the method based on the MultiPhase algorithm, which is a process mining technique, as change process mining is similar to process mining from enactment logs. According to the authors, the MultiPhase algorithm stood out from the other available process mining algorithms as it handles complex branchings robustly, but they note that any other process mining approach which detects causalities explicitly could be used. Based

on the casualties derived from the log, the MultiPhase mining algorithm can construct basic workflow graphs, EPC models, and Petri nets. The MultiPhase mining algorithm generates a model for each process instance which only needs to capture causal dependencies as there are no choices in a single instance [11]. Then, the generated models are aggregated to create a general model for the set of change logs. As van Dongen and van der Aalst state, the causal relations are inferred from the change log as follows in the MultiPhase algorithm [41]:

> *If a change operation A is followed by another change B in at least one process instance, and no instance contains B followed by A, the algorithm assumes a possible causal relation from A to B.*

Günther et al. introduce the concept of commutativity-induced concurrency to extend MultiPhase mining by removing unnecessary causal relations that do not reflect dependencies between change operations. This in return ensures that not every two change operations are required to be observed in both orders to be found concurrent, which is specifically important in change logs since they usually contain fewer data compared to execution logs.

The second method they propose is based on the theory of regions [5] and generates a Petri net model by mapping change logs to a labeled state transition system. This method builds upon the fact that a process model can be reconstructed from the original model and a sequence of change operations. A transition system is constructed based on the fact that a sequence of changes defines a state and the assumption that the changes are memoryless, which means that the process model after the change contains the necessary data. Every process model visited in the change log is represented by the states in the transition system. The theory of regions provides the functionality to map the transition system to a change process model, such as a Petri net. In such a context, Petrify [4] can be utilized to construct Petri nets for transition systems with the use of regions.

Günther et al. state that the proposed methods provide an overview of instance changes that have been made at the system level, to the process engineers. They conclude that the methods produce different results as they are different ways of evaluating change processes but add that the first method performs better on small quantities of change log instances. The second method performs better when large quantities of change log instances are available as it has better accuracy in capturing the observed sequences of changes.

In [19], a comparative trace clustering technique is introduced to detect changes in the behavior of a business process by comparing changes in clusters over sub-logs. In [19], Hompes et al. extend the work in [20] to present a new method for the detection of change points. The method presented in [20] combines outlier detection and trace clustering for the detection of deviating and common behavior, and uses the MCL (Markov Cluster) algorithm [42]. The method in [19] introduces the detection of changes in control-flow and data between the different executions of a business process. The method is implemented in ProM and is available as a package called TraceClustering.

In the method proposed in [19], a stochastic similarity matrix between the

different executions of a business process is used as an input in MCL. Expansion and inflation operations are alternated repeatedly on the similarity matrix in order to achieve the separation of the matrix into distinct elements, which results in clusters. In expansion, the matrix is raised to a certain power and in inflation, every element is raised to a certain power and the matrix is normalized so that it remains stochastic. Similarities between different executions of business processes are computed by mapping each execution to a profile vector with the use of perspectives and calculation of the values of pair-wise vector similarities. Perspectives can be the frequency of specific activities, execution data, event data, or such. As new events happen in a specific trace, the similarity of that trace to other traces will be changed, which can be captured in the similarity matrix. The change of values in the similarity matrix is computed over time to find important change points. For each event window defined over time, executions of business processes that contain events either in that window or before that window are used in the computation of the next similarity matrix. Attributes of process executions are assumed to be known from the beginning of a trace of an execution. Therefore, change points will be visible at the beginning of an execution that deviates and will result in a sub-log of all events that occurred before a specific time. For that sub-log, a similarity matrix is constructed and compared with the similarity matrix from before. If the change points are known, the behavior before and after the change points can be compared by using the clusters of process executions based on the time of occurrence relative to the change points, which can reveal the characteristics of the behavior that has changed. There will be a separate cluster for a behavior that became less common and that will not be put in the same cluster as another common behavior.

In [15], a framework is proposed for change mining in business process variability, which can be used as a foundation for a recommendation system to help the business process designers change processes in order to satisfy new requirements. The framework is based on change mining from a collection of event logs to recognize and recommend changes in business process variability. Hmami et al. state that the use of a collection of event logs for change discovery of a configurable process that is related to a family of process variants is difficult due to the fact that such collections require the identification of the variability specification of the collection and discovery of changes of the variable elements when a mining approach is applied to them. The framework that they propose consists of four components:

1. *The First Component* takes a set of similar event logs as input and merges them into one event log. The output contains common parts and variable parts of the logs that are merged.

2. *The Second Component* reduces the volume of the output data from the first component by filtering the variable parts from the common parts, which results in a variability event log. Hmami et al. provide a filtering approach to be used in the second component which relies on the variability specification file. The variability specification file indicates the variable

fragments which are to be mined in order to discover variability changes.

3. *The Third Component* compares the variability event log and variability specification file to highlight the change points and outputs a variability change log that contains the changes that were applied to the variable elements.

4. *The Fourth Component* generates a list of changes from the variability change log, which are recommended for the business process models in the future, for business process designers.

After the filtration of the variability event log, only the variability-related data will remain, which makes change mining more straightforward and efficient.

In [17], an event log of variable fragments is created from a collection of event logs to be used as an input for change mining.

An event log is composed of a set of traces, which is a record of an executed business process. An event log can be represented as a CSV file, MXML (Mining Extensible Markup Language), or XES (Extensible Event Stream). CSV was used by Hmami et al. in [17] to make the implementation of the proposed method more straightforward.

Hmami et al. presents a merging and a filtering algorithm, to be used in the first and second components of the framework that they have proposed in [15]. The steps for the procedure are:

1. Event logs with similarities are selected.

2. Event logs are looped through. An ID is given to each trace of the event log and all of the traces are merged.

3. Events related to variation points are filtered by using the variability specification file. The filtering is done by selecting the activities related directly to the variation points and by selecting the dependencies.

4. An event log of variable fragments is acquired.

In [14], a change mining approach which can be used in the third component of the framework that they have proposed in [15]. The collection of event logs represents a data stream as the approach used is inspired by a machine learning technique called STAGGER, which is used to extract the changed fragments from a set of events. The extracted change fragments are stored in a change log which is formatted as an XML file.

Hmami et al. emphasize the importance of detection of changes before a process mining approach is applied to an event log. Thus, in [14], the focus is on the detection of sudden and recurrent changes. A sudden change is an unforeseen event that occurs unexpectedly. A recurrent change is a seasonal change that can recur multiple times.

The approach that they propose starts with the preparation of the event log of variable fragments. The event log of variable fragments is prepared by

merging and filtering the collection of event logs as described in [15]. The input of the preparation phase is a variability specification file and a collection of event logs. The output is the event log of variable fragments. Change mining is applied to the output of the previous phase and a variability change log is created as a result. Lastly, the variability change log is analyzed to extract the most significant changes to be recommended for the future evolution of the configurable business process model.

The proposed algorithm consists of six steps which are classification, initialization, projection, evaluation, aggregation, and storage. The tasks of each step are:

1. *Classification:* The list of fragments is labeled to choose valid and invalid fragments on the event log of variable fragments. The values for each attribute of variable fragments are assigned and a set of valid fragments are listed.

2. *Initialization:* The fragment pool is initialized with the fragments of the trace. Then, the fragments of the trace are formatted in the form of the attribute that is chosen.

3. *Projection:* Projection of fragments of the selected trace on the list of valid fragments, which have been created on the classification step, is done in order to determine the fragments that are most likely to be selected.

4. *Evaluation:* If a change in a fragment of the trace is detected in projection, the element that is concerned by the detected change will be identified. If there is a change in the previous or the next elements, it is identified as a position change, which is a fragment change. If there is a change in the variation, it is identified as a change in variation points.

5. *Aggregation:* The change is named by the appropriate target which is change on a fragment, change on variants, or change on variation point and a count number is given to the target name so that it can be put on a specific change type group.

6. *Storage:* The fragments that are not changed are deleted from the pool and the fragments that are changed are stored in an XML file.

In order to contain the data necessary to perform future analyzes, the generated change logs should be detailed and accurate. Thus, the data in the logs should be able to provide answers to the following questions [14]:

1. When did the change happen?

2. Which trace is concerned by this change?

3. Which business process variant is concerned by this change?

4. Which variability is concerned by this change?

5. Is it a fragment change or a variant change?

6. What is the name of the element concerned by this change?

7. What is the new element in this change?

In order to answer the questions listed above, each one of the detected changes, which is stored in an XML file, should contain the following attributes [14]:

1. Date of Change

2. ID of Trace

3. ID of Process Variant

4. Variation Point

5. Type of Change (Fragments or Variants)

6. Name of the Changed Element

7. Name of the New Value due to the Change

In order to test the proposed approach, researchers have implemented the method on the toolset "Random Configurable Process Model Generator", where they have previously implemented the merging and filtering steps of the proposed framework. The results have shown that the approach proposed by Hmami et al. [14] has the ability to detect drift on synthetic event logs. However, tests on a real collection of event logs have not been done yet.

In [18], Hmami et al. propose a new change tree method to facilitate the analysis of change logs of a configurable process model, which are obtained through configurable process mining methods such as change mining. The proposal aims at an easier and readable representation of change so that valuable information can be retrieved for planning the future evolution of the model. To achieve readability, the information in change logs should be rearranged in a way that all changes to each variable fragment can be seen clearly. Change logs of variable fragments are proposed to be presented as change trees [14]. The proposed format has been already used in [21] to represent change logs of business processes. However, as stated by [18], the representation in [21] is concerned with the change log of only one business process and [18] focuses on the need to handle change logs of variable fragments which belong to a configurable process model. Therefore, [18] extends the change tree method presented in [21] by forming a relationship between changes and variable fragments in order to simplify the analysis of the change log.

The framework consists of a part that creates the initial change tree, which needs the variability file as input, and another part that adds the changes to the foundation tree, which takes the change log and the initial change tree as the input to generate a change tree that rearranges all changes as output.

A change tree method is a representation of the events on a change log as

a tree [18]. The algorithm goes through each line of the change log and puts the changes selected from the log into the appropriate spots in order to build the change tree, which creates a path for each instance that may have some common parts with the existing paths [18]. As stated by Hmami et al., the tree format of the representation emerges from the commonalities of parts. A relation is needed to be formed between changes and variable fragments in order to acquire an alignment between the change tree method and the configurable process model. Also, the number of repeats for the changes should be determined. To achieve the foretold specifications, a tree that has the configurable process model as a root that has a direct connection to each variable fragment is needed. Each fragment should be connected to the elements of the root which are the activities that form the fragments.

The information that is required to create the change tree can be collected from the configurable process model which can be available as BPMN diagrams (Business Process Management Notation) [3], C-EPCs (Configurable Event-Driven Process Chains) [26], UML AD (UML Activity Diagrams) [6] or in another format [18]. Another option is to use the variability file [33] in which the information related to the variation points in the configurable process model is defined. In [18], the variability file generated from the configurable process model generator [17] is used as it contains only the information related to variation points. In the proposed approach, after the creation of the change tree, the change log is looped over and every change instance is put into an appropriate place on the tree. Therefore, when the same change recurs, there is no need for the creation of a new leaf. Instead, a number that identifies the number of continuous recurrences of a change will be incremented. In the end, the change tree will be composed of collections of paths composed of elements. The order of the elements on the tree follows from the order of the elements on the collection and the connections between consecutive elements form the arcs. On the change tree, the locations of the elements and the fragment must be determined in order to add or extend an arc. The location depends on the relationship between the change and an element of the variable fragment of the configurable process model. There are four cases:

1. *Case One:* An existent change is incremented when the change instance already exists on the tree, the change is related to the same process variant, and no other change instance appears before the recurrence of the change instance. A new arc will not be added, instead, the present one will be incremented in such a case.

2. *Case Two:* A discontinuous change is added which is a change that appears on the same arc on the change tree after a different change, and the change is about the same process variant. A new leaf is added after the previous change in such a case.

3. *Case Three:* Addition of a change related to a previously changed element occurs on a different process variant. A new leaf is put in connection with

the related element of the variable fragments and the new change will be a sibling of the previous one in such a case.

4. *Case Four:* A change is added to the correspondent element that was not changed previously.

When the algorithm completes the loop over the change log, the change tree will be formed and the elements in the fragments that have been affected by the changes, and the duration of changes will be clearly visible to the configurable process model managers.

In [21], two representation methods, change trees and n-gram change trees, for change log data are presented to analyze changes in adaptive process instances. Kaes and Rinderle-Ma state that analysis of similarities between evolutions of process instances can serve as a basis for the prediction of future changes. To visualize which process instances evolved similarly, a representation should contain changes made to all instances chronologically. In change trees, the order of change operations from change instances is preserved and represented along paths that go from the root of the tree to the leaves. The methods are implemented as a plugin in ProM, which uses MXML and XES log files.

Kaes and Rinderle-Ma present an algorithm for mining a change tree from a change log in [21]. The algorithm creates a root node and loops over the change log to check the changes in each instance by starting from the first series of changes that have been performed on the process instances in the change log. If a change is not in any of the other nodes, a new node is created. If the new node represents the last change in the process instance, the counter for the node is incremented and the instance is finished as the leaves are reached and another instance with the same changes is found. If the node does not correspond to the last change in the process instance, the instance is added to the set of instances that will be traversed in the next iterations. When there is a change after the current node in the set of instances, a child node will be created to represent the change. This procedure is repeated recursively until a path that represents each change instance exists on the change tree.

The main advantage of change trees compared to other change mining end products is that they visualize various change instances and change occurrences that are observed in highly adaptive process scenarios. On top of that, n-gram change trees provide information on changes that occurred after a specific change sequence.

To visualize the process instances that have evolved similarly after a specific change sequence, every occurrence of the change sequence should be identified in the representation. Kaes and Rinderle-Ma point out that change instances can be treated as strings and change sequences can be treated as substrings, which transforms the problem at hand into the problem of n-gram models in language processing. Thus, in such a context, the n-gram will represent the change pattern of interest and enable the identification of the change sequences that follow the change pattern for all occurrences in the change log. In n-gram change trees, a change gets appended to the n-gram change tree if it occurs after

the application of the n-gram to the instance compared to regular change trees.

In [32], Rinderle et al. present the set of information related to changes that should be stored in change logs, and a method for the efficient representation of the stored changes, which is implemented in ADEPT2[5]. Rinderle et al. state that the type of logged change information and the representation of logged change information affect the usefulness of the change logs, and present four use cases for change log management to discuss where purged change logs are more useful than noisy change logs [32]:

1. Restoration of the logical structure of a process instance that has gone under ad-hoc changes by providing additional information to be used with event log information.

2. Enablement of change traceability, which is a critical requirement as all changes applied to the standard procedures have to be documented for legal reasons in specific domains.

3. Reuse of change if similar situations happen again.

4. Detection of conflicts between concurrent changes applied to a process from change log information.

Change logs can be purged from noise, which is unnecessary, irrelevant, or wrong information in the change log. This is crucial for change mining, change effect analysis, and comparison of changes, and change log management should allow different views of the log information for such use cases. For instance, only the necessary information should be provided for change mining and conflict detection, as noisy or irrelevant information makes it difficult to check or compare changes in processes. However, the presentation of all change transactions, whether necessary or irrelevant, is required for traceability purposes.

In order to meet the requirements of the aforementioned use cases, a suitable representation for the change log information and methods to process it must be determined. Rinderle et al. recognize two basic approaches for defining the changes in process graphs which are:

1. *Application of Graph Primitives:* Changes are defined on process graphs by application of sequences of graph primitives, such as insertion or deletion of nodes and edges. This approach would allow restoration of process structures, and enable effective conflict checks, but results in loss of information related to the semantics of the changes, which limits change traceability and conflict analyses.

2. *By High-Level Change Operations:* Changes are defined on process graphs with the use of high-level operations which combine change primitives in specific ways. This approach contains more change semantics and is associated with formal preconditions and postconditions. High-level change

[5]ADEPT2 is an adaptive, high-performance process management tool.

operations can be grouped together in change transactions in order to express complex changes, which may be needed if the application of a set of concomitant change operations is required.

Rinderle et al. propose an algorithm for purging change logs from the noise, which can be applied to complex change transactions as well as simple ones. The algorithm for purging determines sets of all added activities and all deleted activities in the change log. Then, the change log is iterated by starting from the end and moving towards the beginning by determining whether the inspected change operations or transactions have an actual effect on the process template. If the change operation at hand has affected the process template, it is added to the purged change log. Auxiliary sets are used to keep track of the activities, control edges, data elements, and data edges that have been treated. The creation of purged change logs based on the algorithm provides a specific view of the occurred changes, which may be presented to the users to provide them with a summary of the actual change effects on the initial process template. An exception to the algorithm is when a deleted activity in a set of data-dependent activities that are concomitantly deleted is re-inserted without the reinsertion of other data-dependent steps. In such cases, there will be a remaining effect and as a consequence, the change operations associated with other steps cannot be entirely purged from the change log.

Rinderle et al. presented the concept of the delta layer as well to reflect the deviations of process instances from the process template. They represent the delta layer by using an object, which implements the same interfaces as the process template object, and therefore, offers the same operations. However, the delta layer object only reflects on the changed parts of the process template in a specific instance. Instance-specific templates of biased instances can be acquired by using the process template object together with the delta layer object. As the full change logs are kept, applied changes are traceable at the type level and instance level.

In [2], an approach is proposed to mine workflow changes of users in WaaS (Workflow Platform as a Service), discover change rules from them, and reuse the change rules to adapt workflow models to different situations automatically in the future. The proposed framework in [2] is composed of three phases:

1. A workflow model will be specified and deployed to the WaaS.
2. Changes applied to the original workflow model will be retrieved and used for the discovery of change rules.
3. Discovered change rules will be automatically applied to the workflow model based on the context information.

Cao et al. state that a process model can be converted to a well-structured model [44], and a well-structured process model can be converted into a tree, which is known as a Process Structure Tree (PST) [43]. Therefore, [2] compares PSTs of two versions of a process model in order to retrieve change operations applied to the original version, instead of registration of all change operations

which is hard to analyze and reuse if there are too many operations. As the retrieved changes are organized compared to records of all changes, it is simpler to reuse them.

Detection of differences between two PSTs requires the largest common subtree to be found. This problem is called the Maximum Common Subtree Isomorphism problem [35]. However, available Maximal Common Subtree (MCST) isomorphism methods cannot be used for PSTs, as PSTs are semi-ordered trees. In [2], the authors revise the algorithm from [28] which computes constraint edit distances between two semi-ordered trees in order to solve the top-down MCST isomorphism problem in PSTs. Based on MCSTs, they provide a change operation retrieval algorithm, which assumes that all deletions occur before all insertions in order to avoid the retrieval of redundant operations. The retrieval algorithm takes the original PST and the revised PST as input. The algorithm proceeds as follows:

1. Create an empty variable that represents the tree edit sequence.
2. Find top-down MCST of the original and the revised PSTs.
3. Delete the disconnected subtrees which exist in the original PST but not in the MCST. Record the deletions into the variable that holds the tree edit sequence.
4. Insert the disconnected subtrees that exist in the revised PST but not in the MCST from top to bottom. Record the insertions into the variable that holds the tree edit sequence.
5. Return the variable that holds the tree edit sequence.

Cao et al. state that the retrieved change information can be utilized to mine rules for process model configuration. For example, a decision tree can be used to generate reuse rules for individual change operations by transformation of a path into a rule. The steps for the generation of rules are as follows:

1. Record context and change information into change logs.
2. Preprocess the change logs and put non-redundant change operations into a set.
3. Create a decision tree for every change operation in the set.

Cao et al. have implemented a prototype to evaluate the aforementioned methods in Java, which has three layers:

1. *User Layer:* Provides functionalities to describe contexts, and design process models.
2. *Persistent Layer:* Stores event logs, change logs, process models, context templates, change rules, and such data.

3. *Engine Layer:* Provides a change management module with four components which are:

   (a) *Change Retrieval:* Compares processes and retrieves change sequences.

   (b) *Change Rule Retrieval:* Mines change rules through analysis of relationships between context parameters and change operations in change logs based on a decision tree model.

   (c) *Change Rule Reuse:* Generates a suggested process model based on the process context.

   (d) *Context Description:* Records or acquires context information.

4 Discussion

This section will present an analysis of metadata of the publications, elaborate on the results and their relevance to our research questions, and discuss the limitations of the review.

4.1 Discussion of Publications

In our final set of publications, 5 of the papers were published in the last 3 consecutive years, with 3 of them being conference papers, which indicates a trend in change mining and more interest from the scientific community. Also, it is noteworthy that none of our selected literature was published in the same journal or presented at the same conference as another one. This can suggest that there is no established journal or conference that focuses on change mining.

4.2 Discussion of Findings

In this paper, we presented a literature review on methods for change mining and logging in process-aware information systems. We presented a review methodology and as a result, we picked 9 publications that were published between the years 2000 to 2022 to answer *RQ1*, which can be seen as a list on Table 8.

Reference	Title
[2]	Mining Change Operations for Workflow Platform as a Service
[11]	Using Process Mining to Learn from Process Changes in Evolutionary Systems
[14]	Handling Sudden and Recurrent Changes in Business Process Variability: Change Mining Based Approach
[15]	A New Framework to Improve Change Mining in Configurable Process
[17]	Enhancing Change Mining from a Collection of Event Logs: Merging and Filtering Approaches
[18]	Applying Change Tree Method in Configurable Process Mining
[19]	Detecting Change in Processes Using Comparative Trace Clustering
[21]	Mining and Querying Process Change Information Based on Change Trees
[32]	On Representing, Purging, and Utilizing Change Logs in Process Management Systems

Table 8: List of Primary Studies

We analyzed and classified the publications to answer the rest of the research questions that we presented in Section 2.1. The findings of the review in regards to the other research questions are as follows:

RQ2: We have recognized two main categories of methods for change mining:

(a) *Adapted Process Mining Methods:* In this class, we have included the methods that utilize the application of process mining methods for change mining purposes. Application of α-Algorithm, MultiPhase Miner, and Heuristics Miner algorithms to MXML change logs has been presented in this category. However, so far only MultiPhase Miner has been optimized amongst them to handle complex processes and various changes [11]. Application of trace clustering based methods for change mining purposes [19] are considered in this category as well.

(b) *Novel Methods for Change Mining:* In this class, we have included the methods for change mining that are not adaptations of process mining methods. The methods in this class are mapping of change logs to labeled state transition systems [11], algorithms based on machine learning [14], and methods based on comparing process structure trees [2].

RQ3: We have recognized that changes in business processes are recorded into change logs either by recording each change operation [2] or by comparing original and revised process models [2]. It is noteworthy that sometimes change information does not get recorded at all and has to be extracted from event logs [15]. Also, we touch upon the problem of noisy change logs that need to be purged when all change operations are recorded [32]. We would like to emphasize that the articles we selected barely focused on how changes are recorded. Recording of changes is a task that must be handled by process-aware information systems, and only [2] and [32] were written from the perspective of a process-aware information system.

RQ4: We have recognized five formats for storage of change logs which are ADEPT change logs [11], XML [14] [18], MXML [11] [21], CSV [17], and XES [21]. However, [2], [19], [15], and [32] did not specify a format for storage of change logs.

RQ5: We have recognized five ways to learn from changes in business processes:

(a) *Change Model:* Change models are created from change logs in order to visualize the changes that have been applied to instances of a process [11].

(b) *Change Trees:* Change trees are used for the presentation of the information in the change logs [21] [18]. The advantage of change trees is that they provide a view of various change instances along with their occurrences, and reveal the similarities between changes applied to different process instances.

(c) *Change Recommendations:* A list of change recommendations to be applied to a process can be generated from change information in the context of business process variability [15], which enables the enhancement of business processes.

(d) *Change Rule:* Change rules can be generated from change information and can be used for auto-configuration of processes based on context information [2].

4.3 Limitations

A possible limitation to this literature review is the excluded publication types. Only peer-reviewed sources have been included in this review. However, there is a possibility that valuable information about change mining can be found in sources such as websites of organizations that work on business process management, and documentations of change mining tools. We can assume that information that can be found in these sources are parts of research that have not yet been completed and will be published in the future. If this is indeed the case, the review can be extended to retrieve the information in the future by repeating the review methodology described in this paper.

Another limitation to this literature review was the search limit that we determined for Google Scholar because of the project duration. As we only looked at the first 100 results, there could be relevant publications that we did not include in the review. Also, in this literature review, the research and the data extraction have been performed by a single researcher, which may be a threat to the validity of the results. It is suggested in [22] that data extraction should be performed independently by two or more researchers. However, this was not possible due to the way the research was organized.

5 Conclusion

This literature review aimed to analyze the methods that are used to mine, store, and record changes in process-aware information systems and explored what can be learned from such practices. To explore the field, we presented a review methodology and examined 765 peer-reviewed publications that were published between the years 2000 and 2022. We presented a set of criteria and applied them to the publications which resulted in the selection of 6 papers. We extended the publication list by examining the references of the papers that we selected. We ended up with 9 publications that had to be analyzed and synthesized to answer our research questions. We extracted the data of each publication by using the form on Table 5 and presented the results of the research.

At the end of the research, we have recognized that process mining is a more mature research field than change mining. Therefore, the identification of commonalities and differences between change mining and process mining may enable the utilization of more process mining techniques in change mining. For

future work, we propose a study to compare and adapt other available process mining methods for change mining. Furthermore, a comparative study that identifies the advantages and disadvantages of different change mining and logging methods may be beneficial for researchers to identify possible improvements and fill the research gaps. Results of such a study would assist practitioners to decide on methods to use for the development of PAISs as well. In addition, we believe that there may be machine learning algorithms, other than STAGGER [14], that can be utilized for change mining purposes. Identification of methods similar to STAGGER and evaluation of whether they can be used in change mining or not might lead to the discovery of more efficient methods.

6 Acknowledgments

I would like to show my appreciation for my supervisors, Prof. Dr. Dimka Karastoyanova and Ph.D. student Arash Yadegari. Specifically, I wish to acknowledge the help provided by Arash Yadegari, who assisted me at every stage of the research project and helped me to finalize the paper.

References

[1] Ifadhila Affia, Luh Putu Eka Yani, and Ammar Aamer. Factors affecting iot adoption in food supply chain management. In *9th International Conference on Operations and Supply Chain Management*, pages 19–24, 2019.

[2] Jian Cao, Yan Yao, and Yi Wang. Mining change operations for workflow platform as a service. *World Wide Web*, 18:1071–1092, 2014.

[3] Michele Chinosi and Alberto Trombetta. Bpmn: An introduction to the standard. *Computer Standards and Interfaces*, 34(1):124–134, 2012.

[4] Jordi Cortadella, Michael Kishinevsky, Alex Kondratyev, Luciano Lavagno, and Alexandre Yakovlev. Petrify: a tool for manipulating concurrent specifications and synthesis of asynchronous controllers. *IEICE Transactions on information and Systems*, 80(3):315–325, 1997.

[5] Jordi Cortadella, Michael Kishinevsky, Luciano Lavagno, and Alexandre Yakovlev. Deriving petri nets for finite transition systems. *IEEE Transactions on Computers*, 47(8):859–882, 1998.

[6] Marlon Dumas and Arthur H.M. ter Hofstede. Uml activity diagrams as a workflow specification language. In *4th International Conference on the Unified Modeling Language, Modeling Languages, Concepts, and Tools*, volume 2185 of *Lecture Notes on Computer Science*, pages 76–90, Canada, 2001.

[7] Marlon Dumas, Wil M.P. van der Aalst, and Arthur H.M. ter Hofstede. *Process-Aware Information Systems: Bridging People and Software Through Process Technology*. John Wiley & Sons, New Jersey, 2005.

[8] Loubna El Faquih, Hanae Sbai, and Mounia Fredj. Semantic variability modeling in business processes: A comparative study. In *9th International Conference for Internet Technology and Secured Transactions*, pages 131–136, London, 2014. IEEE.

[9] Florian Gottschalk. *Configurable process models*. PhD thesis, Technische Universiteit Eindhoven, 2009.

[10] Christian W. Günther, Stefanie Rinderle, Manfred Reichert, and Wil M.P. van der Aalst. Change mining in adaptive process management systems. In *OTM Confederated International Conferences*, pages 309–326. Springer, 2006.

[11] Christian W. Günther, Stefanie Rinderle-Ma, Manfred Reichert, Wil M.P. van der Aalst, and Jan Recker. Using process mining to learn from process changes in evolutionary systems. *International Journal of Business Process Integration and Management*, 3(1):61–78, 2008.

[12] Christian W. Günther and Wil M.P. van der Aalst. A generic import framework for process event logs. In *Business Process Management Workshops*, volume 4103, pages 81–92, Berlin, 2006. Springer.

[13] Paul Harmon. *Business process change: A guide for business managers and BPM and Six Sigma professionals*. Elsevier, 2010.

[14] Asmae Hmami, Mounia Fredj, and Hanae Sbai. Handling sudden and recurrent changes in business process variability: Change mining based approach. *International Journal of Advanced Computer Science and Applications*, 12(4):632–640, 2021.

[15] Asmae Hmami, Hanae Sbai, and Mounia Fredj. A new framework to improve change mining in configurable process. In *3rd International Conference on Networking, Information Systems & Security*, number 34, pages 1–6, USA.

[16] Asmae Hmami, Hanae Sbai, and Mounia Fredj. Change mining in business process variability: A comparative study. In *4th World Conference on Complex Systems*, pages 1–5. IEEE, 2019.

[17] Asmae Hmami, Hanae Sbai, and Mounia Fredj. Enhancing change mining from a collection of event logs: Merging and filtering approaches. *Journal of Physics: Conference Series*, 1743(1), 2021.

[18] Asmae Hmami, Hanae Sbai, and Mounia Fredj. Applying change tree method in configurable process mining. In *2nd International Conference on Innovative Research in Applied Science, Engineering and Technology*, pages 1–6, Morocco, 2022.

[19] Bart F. A. Hompes, Joos C. A. M. Buijs, Wil M.P. van der Aalst, Prabhakar M. Dixit, and Johannes Buurman. Detecting change in processes using comparative trace clustering. In *5th International Symposium on Data-driven Process Discovery and Analysis*, volume 1527, pages 95–108, 2015.

[20] Bart F. A. Hompes, Joos C. A. M. Buijs, Wil M.P. van der Aalst, Prabhakar M. Dixit, and Johannes Buurman. Discovering deviating cases and process variants using trace clustering. In *27th Benelux Conference on Artificial Intelligence*, Belgium, 2015.

[21] Georg Kaes and Stefanie Rinderle-Ma. Mining and querying process change information based on change trees. In *13th International Conference on Service-Oriented Computing*, pages 269–284, Berlin, 2015. Springer.

[22] Barbara Kitchenham and Stuart Charters. Guidelines for performing systematic literature reviews in software engineering. Technical report, Keele University and Durham University, 2007.

[23] Barbara Kitchenham, Emilia Mendes, and Guilherme H. Travassos. Cross versus within-company cost estimation studies: A systematic review. *IEEE Transactions on Software Engineering*, 33(5):316–329, 2007.

[24] Anders Kofod-Petersen. How to do a structured literature review in computer science. Technical report, 2012.

[25] Maria Leitner and Stefanie Rinderle-Ma. A systematic review on security in process-aware information systems – constitution, challenges, and future directions. *Information and Software Technology*, 56(3):273–293, 2014.

[26] Jan Mendling, Jan Recker, Michael Rosemann, and Wil M.P. van der Aalst. Generating correct epcs from configured c-epcs. In *Symposium on Applied Computing*, volume 2, pages 1505–1510, France, 2006. ACM.

[27] Chitu Okoli and Kira Schabram. A guide to conducting a systematic literature review of information systems research. *Research Methods & Methodology in Accounting eJournal*, pages 870–910, 2010.

[28] Aïda Ouangraoua and Pascal Ferraro. A constrained edit distance algorithm between semi-ordered trees. *Theoretical Computer Science*, 410(8–10):837–846, 2009.

[29] Shaya Pourmirza, Sander Peters, Remco Dijkman, and Paul Grefen. A systematic literature review on the architecture of business process management systems. *Information Systems*, 66:43–58, 2017.

[30] Manfred Reichert and Peter Dadam. Adeptflex—supporting dynamic changes of workflows without losing control. *Journal of Intelligent Information Systems*, 10(2):93–129, 1998.

[31] Manfred Reichert, Stefanie Rinderle-Ma, and Peter Dadam. Flexibility in Process-Aware Information Systems. *Data and Knowledge Engineering*, 66(3):115–135, 2009.

[32] Stefanie Rinderle, Manfred Reichert, Martin Jurisch, and Ulrich Kreher. On representing, purging, and utilizing change logs in process management systems. In *International Conference on Business Process Management*, pages 241–256. Springer, 2006.

[33] Rabab Sikal, Hanae Sbaï, and Laila Kjiri. Configurable process mining: Variability discovery approach. In *5th International Congress on Information Science and Technology*, pages 137–142. IEEE, 2018.

[34] Wei Song and Hans-Arno Jacobsen. Static and dynamic process change. *IEEE Transactions on Services Computing*, 11(1):215–231, 2016.

[35] Gabriel Valiente. *Algorithms on Trees and Graphs*, volume 112. Springer, Berlin, 2002.

[36] Wil M.P. van der Aalst. *Process-Aware Information Systems: Design, Enactment, and Analysis*. 2008.

[37] Wil M.P. van der Aalst. Process-Aware Information Systems: Lessons to be Learned from Process Mining. *Transactions on Petri Nets and Other Models of Concurrency*, 2:1–26, 2009.

[38] Wil M.P. van der Aalst, Arya Adriansyah, Ana Karla A. de Medeiros, Franco Arcieri, et al. Process mining manifesto. In *Business Process Management Workshops*, pages 169–194, 2011.

[39] Wil M.P. van der Aalst, Alexander Dreiling, Florian Gottschalk, Michael Rosemann, and Monique H. Jansen-Vullers. Configurable process models as a basis for reference modeling. In *Business Process Management Workshops*, pages 512–518. Springer, 2005.

[40] Boudewijn F. van Dongen, Ana Karla A. de Medeiros, Henricus M. W. Verbeek, A. J. M. M. Weijters, and Wil M.P. van der Aalst. The prom framework: A new era in process mining tool support. In *26th International Conference on Applications and Theory of Petri Nets*, volume 3536 of *Lecture Notes in Computer Science*, pages 444–454, Berlin, 2005. Springer.

[41] Boudewijn F. van Dongen and Wil M.P. van der Aalst. Multi-phase process mining: Building instance graphs. In *International Conference on Conceptual Modeling*, volume 3288 of *Lecture Notes in Computer Science*, pages 362–376, Berlin, 2004. Springer.

[42] Stijn van Dongen. A cluster algorithm for graphs. Technical report, National Research Institute for Mathematics and Computer Science in the Netherlands, 2000.

[43] Jussi Vanhatalo, Hagen Völzer, and Jana Koehler. The refined process structure tree. *Data Knowledge Engineering*, 68(9):793–818, 2008.

[44] Jussi Vanhatalo, Hagen Völzer, Frank Leymann, and Simon Moser. Automatic workflow graph refactoring and completion. In *International Conference on Service Oriented Computing*, volume 5346 of *Lecture Notes on Computer Science*, pages 100–115, 2008.

[45] Roy Wendler. The maturity of maturity model research: A systematic mapping study. *Information and Software Technology*, 54(12):1317–1339, 2012.

[46] Hongyan Zhang, Weilun Han, and Chun Ouyang. Extending bpmn for configurable process modeling. In *ISPE Concurrent Engineering*, Advances in Transdisciplinary Engineering, pages 317–330, 2014.