CHAPTER 1

INTRODUCTION

Process mining is a relatively young and developing research area with the roots in computational intelligence, data mining; and process modeling and analysis [34]. Main idea in this research area is to discover, monitor and improve processes by extracting information from event logs. With this idea, process mining creates a bridge between data mining and business process modeling and analysis. Interest in this research area has two origins; firstly, events are recorded and easily available in the modern information systems. Secondly, highly competitive and rapidly changing business life requires improvement and support for business processes [34]. Traditional process mining approaches work on a single organization; however, with the increase of cloud computing and shared infrastructures, event logs of multiple organizations are currently available for analysis. In principle, there are two main environments where cross-organizational process mining stands out. Firstly, when organizations work together to execute the same process, it is insufficient to analyze only logs of one organization and gathered information from all stakeholders should be merged prior to analysis. Secondly, organizations essentially execute the same processes with different needs and configurations on a common infrastructure, where cross-organizational process mining can help the organizations to learn from each other's experience, knowledge and processes.

Process mining spectrum is extensive and highly inter-related with a different sets of activities grouped under categories of *Cartography*, *Auditing* and *Navigation* [33]. In this spectrum, *Cartography* activities aim to create maps of the real world tasks by using process models whereas *Auditing* activities confront the models and reality;

and the last activity group of *Navigation* activities use process mining methods like a navigation application. This thesis study proposes a hybrid approach and exploits approaches from different categories to create a new point of view. In other words, this study aims to create maps using the discovery techniques from *Cartography* and promotes the better operating locations in the process models from *Auditing* to create recommendations for users like a *Navigation* application.

In cross-organizational process mining area, recent studies focus on commonality and collaboration between organizations; however, they only present results based on how similar the process models and behaviors of organizations under cross comparison [10]. In addition, challenges based on partitioning of tasks and process models between organizations are presented in the literature [32]. This thesis study is based on the environment where processes are executed on several organizations and cross-organizational process mining is applied with the idea of unsupervised learning where predictor variables related to performances of organizations are used.

Recent studies in process mining and similarity area include various different approaches to define relationships between process models. These studies include creating similarity metrics for node matching, structural and behavioral similarities [14] and alignment matrices between models [9]. In addition, differences between process models are tried to be explained by mismatch patterns with the help of comprehensive case studies [13]. In this study, mismatch patterns are mathematically defined and implemented; as it is known to be the first implementation of mismatch patterns. Applicability and performance of using mismatch patterns is also analyzed by comparing to the similarity metrics that are defined in the literature.

In the light of these motivations, approach proposed in this thesis study is a four-stage solution and it starts by mining the process models of organizations with a user defined noise threshold. With the mined models and event logs, second stage calculates the performance indicators for each organization and then clusters organizations based on how well they are operating. Third stage aims to find differences between process models of each organization. Final stage combines the information from all stages and provides a set of recommendations. With this approach it is aimed to help business process management users to focus on the potentially important parts of their business maps. In addition, this approach includes implementation of mismatch patterns and performance indicator based clustering of organizations. As an extensible framework, approach stages are designed with minimum inter-dependency and they are open to include new process mining approaches, performance indicators, clustering approaches and mismatch patterns. Moreover, every stage of the methodology is intended to be configurable for user needs and business environment requirements.

Within this thesis study, proposed methodology is implemented in ProM framework [42] as a set of plugins corresponding for each stage and packaged under the name of *CrossOrgProcMin*. Since ProM is the most popular open-source environment for academia and industry at the time of this thesis published, developed set of plugins are built over this framework. With the help of this implementation, approach proposed in this thesis is tested on a synthetic and real-life event logs. Performance of methodology is assessed with a defined evaluation metrics for each stage and resulting recommendations are presented to show how this approach helps users to focus on learning opportunities between organizations with a performance improvement potential.

Rest of the thesis is constructed as following:

- In Chapter 2, related studies in process mining area are presented.
- In Chapter 3, background information for the relevant topics to this thesis study is explained.
- In Chapter 4, methodology proposed in this thesis is presented with detail.
- In Chapter 5, methodology of this thesis is applied on datasets and results are discussed.
- In Chapter 6, summary of this study is presented with the final remarks and pointers for future work.

CHAPTER 2

RELATED WORK

In this chapter, studies related to the work presented in this thesis are summarized. Firstly, latest trends and studies in the process mining area are explained. Then studies from cross-organizational process mining, which is the main topic of this research, is introduced. Following these, studies related to similarity in process mining is mentioned. Finally, performance and conformance analysis approaches in process mining area are presented.

2.1 State of the Art in Process Mining

In this section, important milestones in process mining and current research trends will be presented. Studies in process mining area have roots based on process discovery techniques for software engineering. These original techniques are studied to discover workflows using neural networks, finite state machines and Markovian approaches [40]. However, using process mining in the workflow management area is introduced by Agrawal et al. and this study aims to find workflow graphs given event logs and identify edge conditions between nodes [4]. In the following efforts, various different approaches based on hierarchical structuring, dependency and frequency graphs; and heuristic methods are suggested to address the same problem. While some of these algorithms are focused on creating partial solutions, some of the algorithms are used as a foundation for further expansions such as "Alpha Algorithm" of van der Aalst et al.[41]. Spread of process mining is not only in kept in academia, but also many vendors presented tools and software to discover processes and help

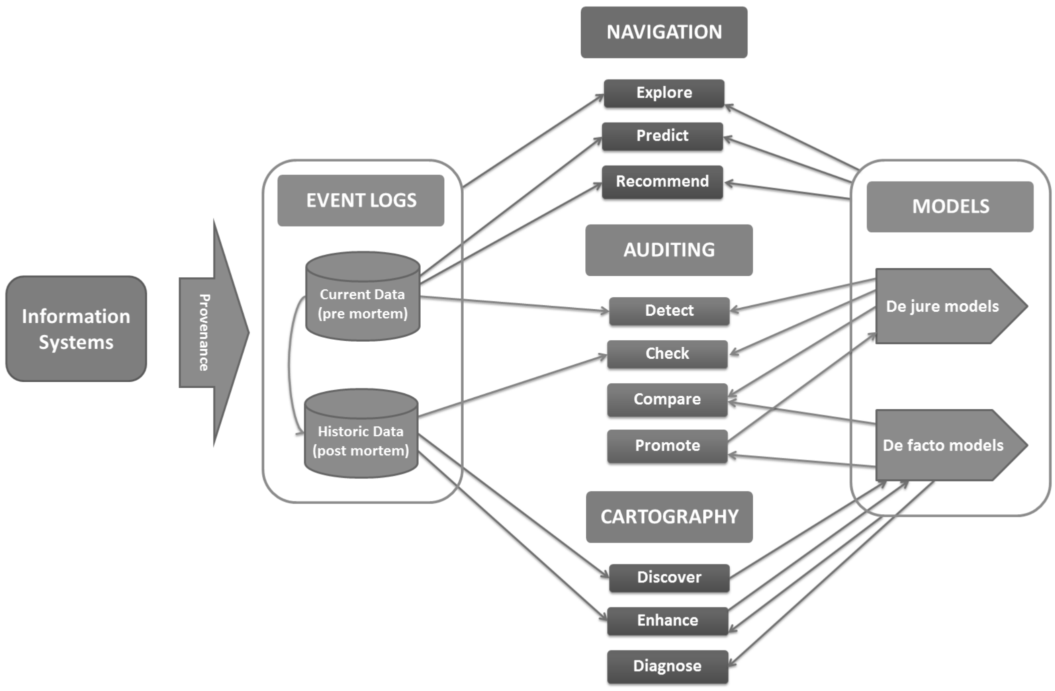


Figure 2.1: Process Mining Framework

organizations manage their workflows [1].

Current challenges in the process mining area can be presented best through mentioning spectrum and its relations. As diagrammed in Figure 2.1 and presented in the research of van der Aalst [33], process mining spectrum is extensive and highly inter-related. Starting with the provenance, process mining deals with gathering and keeping event logs as *current data* and *historic data*. In this level, *post mortem data* means the logs and information gathered from completed events whereas *pre mortem data* means information related to ongoing cases. This separation helps us to use historical knowledge to exploit and gain advantage over current business operations. At the right end of the diagram, models are presented as *de facto models* and *de jure models*. *De jure models* are created with the aim of describing reality as it should be whereas *de facto models* aims to describe reality as it is. Within each model category a mixture of perspectives, in other words point of views from different information levels, can be mixed. In order to fill the gap between event logs and models, ten different process mining activities are listed under three categories [33] as follows:

Cartography: In this category the main aim is to create maps of the real world activities by using process models.

Discovery: Discovery is focused on extracting process models from event logs.

Enhancing: Enhancing activities are based on improving *de facto models* with the information hidden in the event logs.

Diagnose: Diagnose activities stand for identifying the problems caused by directly process models.

Auditing: In this category, activities related to confronting the model and the reality are collected.

Detect: Detection is based on comparing the *de jure models* with the ongoing processes to generate alerts if shifts occur in the reality.

Check: Checking is defined to identify deviations occurred in the past.

Compare: Comparison of *de facto models* and *de jure models* shows the difference between planned and the reality.

Promote: Promotion includes partially updating *de jure models* from the information gathered from *de facto models*.

Navigation: In this category, process mining methods are used like a navigation application.

Explore: Event data and models are used in combination to explore business processes.

Predict: Prediction activities combine the information from past events and make predictions about ongoing processes.

Recommend: Suitable actions can be suggested by using the prediction and contextual information.

Within this framework, the study presented in this thesis is a combination of discovery from cartography, promoting from auditing and recommendation from navigation. In other words and within the metaphor of spectrum, this study aims to discover maps using the discovery techniques and promotes the partially best locations in the map to create recommendations for travelers in their navigation applications.

2.2 Process Discovery in Process Mining

In this section, process discovery studies will be presented briefly. Within the process mining framework, there are various different process mining algorithms proposed which have the same aim of discovering underlying processes. Considering the approaches undertaken with the proposed algorithms, the following grouping is provided [24].

α-algorithm: This algorithm is based on the causality relationship that can be discovered from the event logs and proposed by van der Aalst et al. [41]. However, this approach cannot handle loops and therefore extensions like α++ algorithm [12] is proposed to overcome this problem. Outputs of this approach and its extensions are workflow nets with XOR, AND splits and joins to represent underlying process models.

Inductive Approach: Methods proposed in inductive approach [22, 21] are based on finding the best Hidden Markov Models (HMMs) that represents the underlying process model. Workflow nets are constructed by using HMM nodes and inductive learning.

Hierarchical Clustering: This method [19] firstly splits the event logs into clusters and creates a dependency graph. Using this dependency graph, clusters are defined with a hierarchy tree which is then merged into a single workflow model.

Genetic Approach: The first method provided in this group [36] constructs models based on causal matrix of activities and tries to handle noise, incompleteness, concurrency, hidden and duplicate activities. However, there are many configuration parameters and fine-tuning to handle noise and irrelevant data to use this method. Second method in this group [17] applies genetic algorithms to discover process models using from-to charts and this method finds optimal or sub-optimal solutions for a relatively complex processes within a reasonable processing time.

Heuristic Approach: Heuristic approach for process discovery is an extension of α-algorithm which is based on likelihood calculation of activities and constructs dependency and frequency graphs while handling the noise in the event log. In addition, heuristic approach is used to locate the immediate successors with the help of evaluation metrics generated over raw from-to charts [18].

Proposed methods in process discovery are based on solving different challenges in the process mining area. Considering the scope of this thesis study; process discovery operations are undertaken with inductive methods which is a robust, repeatable and mature set of approaches.

2.3 Cross-organizational Process Mining

In this section, related studies and trends in cross-organizational process mining will be presented. Cross-organizational mining is based on cross-correlation of workflows and the realized activities in different organizations. The main challenge of this topic in process mining is that comparing processes and their performances of different organizational units in an objective approach. This objective approach is open to be enhanced with the process context, namely the environment of the process that is executed. Most of the studies in this area are currently studying to reveal the possible opportunities and some initial approaches to address main challenges.

In the study of Bujis et al. [10], the authors indicated the importance of the increase of Software-as-a-Service (SaaS) and cloud computing infrastructure usages. As a result of this increase, more and more organizations will use a Shared Business Process Management Infrastructure (SBPMI) and it is an opportunity for different organizational units to learn from each other in such infrastructure. The approach presented in this study is based on three questions and three metric groups to answer these questions:

1. Which organizations support my behavior with better process models?
2. Which organizations have better behavior which my process model supports? and
3. Which set of organizations can I support with my process model?

While answering these questions, they used *simplification based metrics* to indicate better process models; and *throughput time metrics* to indicate better behaviors. In this study, it is shown how a generic framework can be used to highlight the main idea behind cross-organizational process mining.

In the study of van der Aalst [31], using configurable process models is proposed for the organizations sharing the same infrastructure and doing the similar work. Configurable process models are defined as a family of process models where each organization can use this family with their configurations according to their business needs. This approach not only creates behaviors needed by each organization but also creates a basis to compare and learn within process mining framework. The configurable process models are formalized by "Causal Nets", which is a notational language based on input and output bindings of each node. Although this study does not provide answer to all challenges related to cross-organizational-process-mining, a formalism of configurable processes models is presented to address learning and conformance checking.

Cross-organizational process mining is divided into intra-organizational and inter-organizational process mining subcategories with two basic ideas: *collaboration* and *exploiting commonality* [32]. In his study, van der Aalst defined *collaboration* for distributed work between multiple organizations and *commonality* as ability of using the same process models and infrastructure between the organizations. In order to exploit these two ideas, two partitioning dimensions are suggested in [32]:

Vertical Partitioning: Process instances, namely cases, distributed over several organizations which collaborate to complete a complex activity.

Horizontal Partitioning: Process parts, namely tasks or activities, are shared within organizations like jigsaw puzzle parts.

For these orthogonal dimensions, a number of questions and challenges are presented in the study [32]. For the vertical partitioning, where organizations are aiming to share infrastructure and knowledge to learn from each other, it is mentioned that supervised learning methods like classification can be used. Notion of this thesis is based on vertical partitioning of cross-organizational process mining with the idea of unsupervised learning where predictor variables related to performances of organizations are used.

2.4 Process Similarity in Process Mining

In this section, prominent studies related to similarity in process mining will be presented with their results. With the emerging attention in business processes, organizations become aware of the fact that they can exploit the business processes and their similarities [9]. In addition, most of the large organizations have repository of process models of similar business operations [14]. There are four main different approaches proposed to this solution in the literature currently.

The first approach, proposed by Dijkman et al. [14], is based on similarity metrics as *a*) node matching similarity; *b*) structural similarity; and *c*) behavioral similarity. Result of the study [14] indicates that using these three similarities can differentiate comparable process models and within these metrics structural similarity is the most prominent one.

The second approach by Bujis and Reijis is an analytical approach [9] to compare the process models of different organizations that does similar works. Proposed algorithm is based on creating an alignment matrix between observed and realized models. This inter-relation is also used to compare different variants of the same process by different organizations. In addition, they suggested their method as a framework to further standardize a process of common interest.

The third approach proposed by Esgin and Senkul focuses on delta analysis between predefined models and discovered models. In [16], a hybrid model which considers the dependencies of process activities and process structure is presented to measure the similarity between process models. In addition, *sequences alignment* notion is applied on delta analysis in [15], which aims to arrange structures to identify similar regions such as protein sequences in bioinformatics.

The fourth and final approach is based on frequently occurring mismatches between similar business processes [13]. In this study [13], a set of mismatch patterns are derived from the different departments of the same organization. Although this research does not present a complete set of possible mismatch patterns, the provided set is comprehensive to identify similarities and differences of the processes of same operations. In this thesis, combination of metric and mismatch pattern approaches are used to identify variations between process models of different organizations.

2.5 Performance and Conformance Analysis in Process Mining

In this section summary of the studies related to performance and conformance analysis will be presented. Event logs do not only contain task sequences but also contain time, resource and contextual information. Analysis of process models with these additional information is used within performance analysis framework to highlight bottlenecks or make predictions. However, in order to undertake a performance analysis there is a need of replaying reality (event logs) on the expectation (process models) and check the conformance of reality to plan [35].

In the study conducted by Rozinat and van der Aalst [29], a conformance framework is proposed by two metrics *fitness metrics* and *appropriateness metrics*. In this study [29], fitness metric is based on replaying the event log on the process model and counting the number of missing or remaining tokens. In other words, replaying and conformance of event logs over process models is modelled as a token passing formalism. On the other hand, appropriateness metrics are based on how accurate the process model in describing reality within a degree of clarity. Simplicity, precision and generalization attributes of the process models are taken into account while calculating appropriateness metrics.

Token passing approach to conformance has a major drawback when the process model and reality of event logs do not fit completely. In this case, there are overestimated process models that are too general, in other words with too much behavior other than the reality. In order to overcome this drawback, heuristic and optimization based methods are proposed by other researchers. In this thesis study, an extended version of optimization based approach presented in [2] is used to replay logs on the process models. Since the main goal of this study is not to evaluate conformance of logs and process models, the method presented by Adriansyah et al. is extended to calculate performance indicators while replaying the logs.

CHAPTER 3

BACKGROUND

In this chapter background information is presented for the relevant topics to this thesis study. Topics are mentioned starting with the building blocks and then approaches with the order of their usage in the methodology presented in this thesis study. Firstly, event logs and process models are mentioned with their mathematical formalization and background. Following these, process discovery approaches are explained and then performance analysis methodologies are presented. Then, machine learning and clustering topics are mentioned. Finally, mismatch patterns in process models are presented. All topics in this chapter are limited to the scope of this thesis study with the aim of constructing a necessary background.

3.1 Event Log

Event logs are the main inputs to any process mining methodology including this thesis study and they include information related to real life activities. Event logs which are the outputs of the software systems like Enterprise Resource Planning (ERP) or Business Process Management (BPM) have common properties that are also assumed in the literature as the properties of event logs. General structure of event logs includes multiple layers as diagrammed in Figure 3.1. Processes have cases which are simply single process instances. Within each case, there are events that are generally represented as a sequence of activities performed. Each event is enhanced with the attributes such as timestamps, resource assignments and other contextual data. A fragment of this structure is presented in Table 3.1 for the Loan Application Process

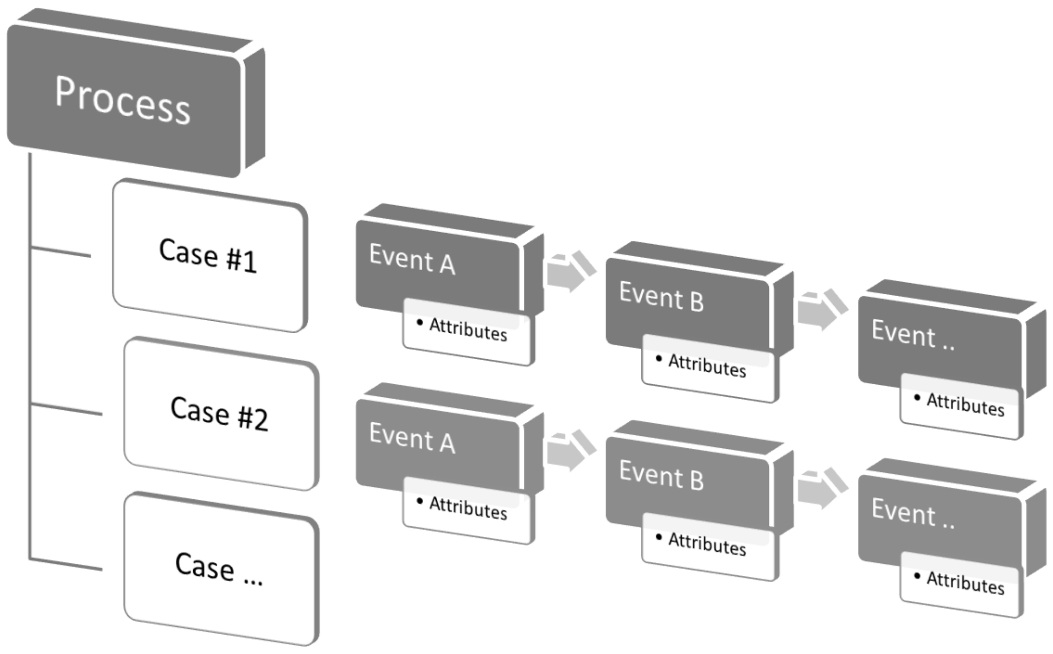


Figure 3.1: Structure of Event Logs

[6], which shows the footprints of a financial organization that provides consumer credits [8]. In the table, each line represents an event with its attributes which are collected under cases to form a complete event log.

Event log structure and related notions are formalized in [33] as following:

Definition 3.1.1. (Event and Event Attributes) Let E be the universe of events which include all possible event identifiers and in this universe any event $e \in E$. Events are enhanced with contextual information, namely attributes. For any event $e \in E$ and attribute A, $\#_A(e)$ is the value of the attribute A for event e. Possible attributes for events include timestamps, people and resource assignments, transaction types and other contextual data.

Definition 3.1.2. (Case and Case Attributes) Let C be the universe of cases which include all possible case identifiers and in this universe any case $c \in C$. Like events, cases are also enhanced with contextual information, namely attributes. For any case $c \in C$ and attribute A, $\#_A(c)$ is the value of the attribute A for case c. Each case has at least one attribute which is trace.

Definition 3.1.3. (Trace) Trace is a sequence of events, $t \in E^*$ such that each event is restricted to occur only once.

Table 3.1: Fragment of event log from Loan Application Process (Variation #1)

Event Log				
		Attributes		
	Event	**Date**	**Time**	**Transition**
Case #1	Register Application	16.04.2013	14:37:27	Complete
	Check Credit	16.04.2013	14:41:19	Complete
	Check System	16.04.2013	14:47:35	Complete
	Calculate Capacity	16.04.2013	14:50:21	Complete
	Accept	16.04.2013	14:53:22	Complete
	Send decision e-mail	16.04.2013	14:55:11	Complete
Case #2	Register Application	16.04.2013	16:28:19	Complete
	Check Credit	16.04.2013	16:36:22	Complete
	Check System	16.04.2013	16:43:10	Complete
	Calculate Capacity	16.04.2013	16:52:40	Complete
	Reject	16.04.2013	16:53:53	Complete
	Send decision e-mail	16.04.2013	17:01:32	Complete
...	...	...	...	...

Definition 3.1.4. (Event Log) Event log is set of cases $L \subseteq C$ such that each event occurs at most once in the event log.

In this study, event logs from different organizations are exploited, thus organization related attributes are used for cases. Within these event logs, traces of cases for each organization are used to discover underlying process models. In addition, timestamps and resource related attributes of events are used to collect performance related data for further analysis.

3.2 Process Modeling

Process modeling is the foundation of process management applications and main tools of people in this profession. Although process modeling can be defined with informal process workflows to document procedures, there is a number of formalized notations which are more suitable to cross-applicability and mathematical analysis. In the control-flow view of process modeling, a process model is aimed to give decisions on which activities to take place with their orders. In this study, control-flow of process models are used to find mismatch patterns between different organizations that execute the same activities. Considering the scope of this thesis, only Petri nets, Workflow Nets and Business Process Modeling Notation (BPMN) will be presented in this section.

3.2.1 Petri Nets

Petri net is a mathematical modeling language that is aimed to describe concurrent systems. Graphical notation of Petri nets seems intuitive and simple; however, it is powerful in terms of being executable and applicability of analysis techniques [37]. Petri nets are directed bipartite graphs where *nodes* represent transitions and *places* represent conditions. Structure represented by Petri nets is static and the state of the net is described by placing *tokens*, namely the process of *marking*. Formalization of Petri nets are explained in [28] as follows:

Definition 3.2.1. (Petri Nets) A *Petri net* is a triplet $N = (P, T, F)$ where P is finite set of *places*, T is finite set of *transitions* and F is set of *flow relations* where:

1. (Separation) $P \cap T = \varnothing$
2. (Flow relation) $F \subseteq (P \times T) \cup (T \times P)$

3.2.2 Workflow Nets

Process models in the real life have additional properties to be executable and they are defined in the Workflow net formalization, which is simply a subset of Petri nets. These additional properties can be formalized in [39] as follows:

Definition 3.2.2. (Workflow Nets) Let $N = (P, T, F)$ be a Petri net and t is a new identifier not in $P \cup T$. N is a workflow net (WF-net) if and only if:

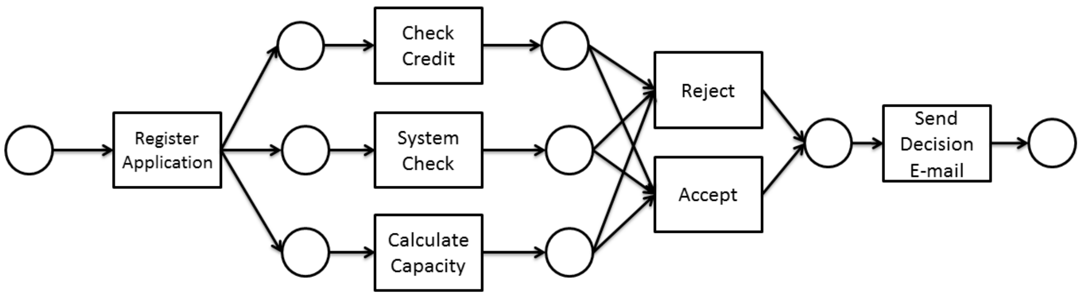


Figure 3.2: Workflow net of Loan Application Process (Variation #1)

1. (Start Node) P contains a *source place i* where no token can be fired to.
2. (End Node) P contains a *sink place o* where no token can be fired from.
3. (Connectedness) $\bar{N} = (P, T \cup \{t\}, F \cup \{(o,t),(t,i)\})$ is strongly connected; in other words, there is a directed path between any pair of nodes in $\bar{N}$.

In simple terms, a Workflow net is a Petri net with a source place to start the process and a sink place to end; furthermore, all nodes are on a path from source place to sink place [30]. In order to illustrate this formalization, the Workflow net for the event log mentioned in Section 3.1 is presented in Figure 3.2. In the figure, places are indicated by circles, transitions are indicated by rectangles, and flow relations are represented by arcs.

Workflow nets are representatives of real life processes; however, they can result with processes including deadlocks, live-locks and never-reached activities. In order to avoid process models from these problems, soundness definition is suggested in [30] and it is simplified as follows with the context of this thesis:

Definition 3.2.3. (Soundness) Let N be a Workflow net and it is sound if and only if:

1. (Safeness) Places cannot hold more than one tokens at the same time.
2. (Proper completion) Any marking of net can reach to sink place.
3. (Absence of dead tasks) Net does not contain any dead transitions.

In this thesis, Workflow nets are used to discover and present underlying process models of different organizations. Considering the applicability of well-known pro-

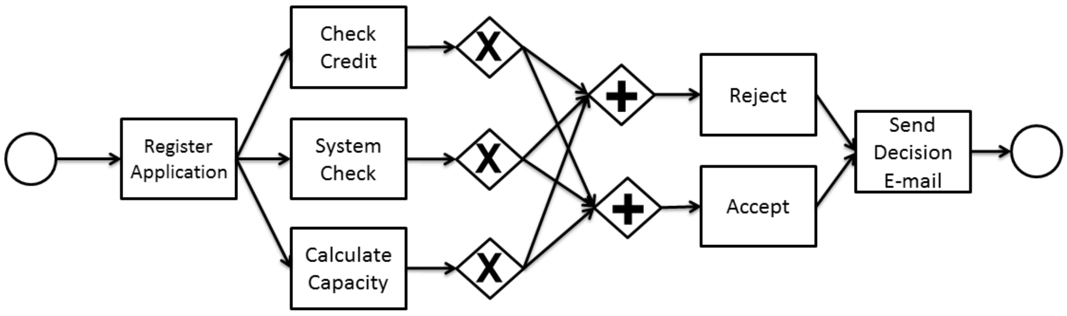


Figure 3.3: Process Model of Loan Application Process (Variation #1) using BPMN

cess mining algorithms on Workflow nets and implementations in ProM Framework [42], Workflow nets are used as the notation for discovery and analysis.

3.2.3 Business Process Modeling Notation (BPMN)

Business Process Modeling Notation (BPMN) is one of the most popular and widely used modeling language implemented by many vendors. In addition to its popularity, this notation is standardized by the Object Management Group (OMG) since 2004. In this notation, atomic activities are named as *tasks* and routing decision logic is implemented by *gateways*. These gateways include split and join gateways with AND, OR and XOR logic operations. In addition, deferred choice pattern is implemented by *event-based XOR gateway* in BPMN to handle race conditions between tasks that are running parallel [38]. Since the primary goal of BPMN is to provide a standardized notation that is easy to understand by business stakeholders, in this study resulting nets are converted to BPMN diagrams for visual analysis by the plugin implemented in ProM [23]. BPMN diagram of the Workflow net from Figure 3.2 is presented in Figure 3.3. As can be seen from the diagram, gateways help to understand the relations and dependencies of the tasks.

3.3 Process Discovery

In process mining field, one of the most challenging tasks is to construct a process model based on the behavior in the event logs, namely process discovery. Many process discovery algorithms are proposed to address different challenges in process discovery and using different notations. However, in this thesis focus of the study is learning from the cross-organizational process mining rather than addressing all process discovery challenges. With this reasoning, Inductive Process Mining [25] is selected as appropriate since it is simple, highly applicable and configurable. In the literature, its derivatives which handles infrequent behaviors [26]; incomplete logs [27]; and model optimization [43] are also available.

Inductive Miner (IM), which is proposed as an extensible framework in [25], aims to discover block-structured process models that are sound and well-fitting to the behavior represented in event log. In addition, this approach focuses on creating rediscoverable models that is a significant attribute for this thesis study. Formalization of the key points in the study [25] is as follows:

Definition 3.3.1. (Block-structured Workflow Nets) Block-structured WF-nets are subset of WF-nets where the workflow can be divided recursively into parts with single entry and exit points.

Definition 3.3.2. (Rediscoverability) Let a process is expressible by a model M which is unknown priori. Given a log L of M, L is a subset of language used to describe model M. M is isomorphic-rediscoverable from L by mining algorithm B if and only if $M \in B(L)$.

Framework developed in the study [25] uses a divide-and-conquer approach to discover subprocesses of sublogs obtained by splitting the event log and its main steps can be listed as follows:

1. Activity Sets: Split the *activities* in log to disjoint sets.
2. Sublogs: Split the *log* by using *activity sets*.
3. Recursive Mining: Mine sublogs with these steps until a sublog contains only single activity.

In the study [25], an algorithm based on this framework is presented that guarantees returning a sound and fitting process model in finite time. Therefore, this framework is selected as appropriate and its extension that can handle infrequent behavior is used within this thesis study to address challenges of different event logs. In real life, most of the cases in the events are samples of frequent behavior; however, there are also infrequent behaviors according to the nature of process execution environment. In real life, these different paths are used infrequently; however, their effect in discovery is still significant. *Inductive Miner Infrequent (IMi)* is proposed in [26] as an extension to *Inductive Miner* to handle noise in the event logs. By filtering the infrequent behavior, it is aimed that *IMi* succeeds with improved models discovered. After each recursive step of *IM*, filtering is applied if the discovered model is a flower model which represents infrequent behavior. Basic idea behind filtering is setting a user-defined threshold between 0 to 1 and with the help of this threshold, both log splittings and mining operations are done on a cleaner subset of logs. When the discovered models compared to *IM*, *IMi* results with lower fitness, higher precision and equal generalization. In this thesis study, infrequent behavior capable implementation is used for experimenting the cross-organizational mining of process models with their implementation of ProM framework [42].

3.4 Process Performance Analysis

In the main and traditional tasks of process mining spectrum, event logs are used to discover and enhance process models. In addition to these main tasks, process mining enables to discover relationships between event logs and process models for conformance and performance analysis. Within conformance, any deviations from modeled behavior can be discovered; moreover when the logs are replayed on the models, bottleneck analysis can be undertaken with the help of timestamps on the event logs. However, in order to replay event logs on the process models there is a need for *alignment* which is formalized in [35]. Formalization and notions presented in the study [35] are based on the assumption that process models and event logs use the same set of activity labels and therefore they can be related.

The first notion in alignment of process model and event log is defining the relationship between *moves in the model and log*. The necessity of this notion is based on the fact that some moves in the event log cannot be operated with the process model and vice versa. In the study [35], move and alignment are defined as follows:

Definition 3.4.1. (Move) For the event log L, $A_L^\perp = A_L \cup \{\perp\}$ is defined where $x \in A_L$ refers to *"move x in log"* and $\perp$ refers to *"no move in log"*. Similarly, $A_M^\perp = A_M \cup \{\perp\}$ is defined where $y \in A_M$ refers to *"move y in model"* and $\perp$ refers to *"no move in model"*. One step in alignment is represented as $(x, y) \in A_L^\perp \times A_M^\perp$ such that:

1. (x, y) is a *move in log* if $x \in A_L$ and $y = \perp$,
2. (x, y) is a *move in model* if $x = \perp$ and $y \in A_M$,
3. (x, y) is a *move in both* if $x \in A_L$ and $y \in A_M$,
4. (x, y) is an *illegal move* if $x = \perp$ and $y = \perp$.

In this environment, *legal moves* are defined as $A_{LM} = \{(x, y) \in A_L^\perp \times A_M^\perp | x \in A_L \vee y \in A_M\}$

Definition 3.4.2. (Alignment) Let $\sigma_L \in L$ a trace in event log L and let $\sigma_M \in \beta(M)$ a full execution sequence of process model M. An alignment of σ_L and σ_M can be defined as a sequence $\gamma \in {A_{LM}}^*$ where the projection of first element yields σ_L and the projection of second element yields σ_M.

In order to qualify the alignment operations, distance function on legal moves is defined in [35] as follows:

Definition 3.4.3. (Distance Function) Distance function is defined on legal moves as $\delta \in A_{LM} \to \mathbb{N}$ to associate costs to moves in alignment:

1. If $x \in A_L$ and $y = \perp$, then $\delta(x, y)$ is the cost of *move x in log*.
2. If $x = \perp$ and $y \in A_M$, then $\delta(x, y)$ is the cost of *move y in model.*
3. If $x \in A_L$ and $y \in A_M$, then $\delta(x, y)$ is the cost of *move x in log and move y in model* and this cost is generally $\delta(x, y) = 0$ if $x = y$.

According to this distance function definition, various different functions can be defined using costs. For instance, in [35] a *standard distance function* is defined as no cost when log and model agree and cost of 1 otherwise. In addition, optional alignment between process model and event logs is defined as follows:

Definition 3.4.4. (Optimal Alignment) Let $\sigma_L \in L$ be a trace in event log L, M a process model and $\Gamma_{\sigma_L,M} = \{\gamma \in {A_{LM}}^* \mid \exists_{\sigma_M \in \beta(M)}\, \gamma\, is\ an\ alignment\ of\ \sigma_L\ and\ \sigma_M\}$. An alignment $\gamma \in \Gamma_{\sigma_L,M}$ is *optimal* for event log trace σ_L and model M if for any $\gamma' \in \Gamma_{\sigma_L,M} : \delta(\gamma') \geq \delta(\gamma)$.

This optimal alignment can be found with the help of different approaches and within process mining field, proposed methods [2, 3] are based on A^* algorithm which is a path-finding algorithm based on graphs. In principle, any optimization methodology can be used to find the optimal alignment and in this thesis study an A^* based implementation in ProM Framework [42] is used to find optimal alignments. Although this thesis study has no direct focus on conformance of process models by event logs, these steps were necessary to replay the event logs over process models. With the help of replay, any performance indicator can be calculated to compare performances of cross-organizational processes. Using the timestamp information or resource utilization in the event logs, performance indicators can be discovered while replaying the log after alignment. These performance indicators can include lead time, service time, waiting time in *time dimension*; and utilization or activity costs in *cost dimension* [33].

3.5 Machine Learning and Clustering

Machine learning is a study area of computer science which have roots in pattern recognition and computational learning theory of artificial intelligence. Machine learning approaches work on construction and learning from data to make further predictions. There are various approaches proposed in this area and clustering approach will be presented in this section. Cluster analysis is based on assigning the set of observations into subsets (*clusters*) so that observations within the same cluster are similar whereas the observations from different clusters are dissimilar, where the similarity criteria is predefined. Clustering is a method in unsupervised learning, where the main problem is learning a hidden structure in unlabeled data. Since the provided data is unlabeled there is no error or reward assignment to the potential solutions provided by these approaches; however, quality metrics on clusters are used to evaluate results. With these characteristics, clustering is a common technique in exploratory data mining and statistical data analysis and it is used in many fields like image analysis, information retrieval and bioinformatics.

In cluster analysis, various algorithms are proposed with different approaches on defining clusters and how to efficiently find them. Popular approaches are based on the idea of decreasing the distance among the members of same cluster, space density of data space, intervals or particular statistical distributions. Within this thesis study, centroid-based clustering is used in which the clusters are defined by a central vector, which might not be a member of data set. When number of clusters are fixed to *k*, the approach is named as *k-means clustering* and the problem is finding k cluster centers and assigning data members to nearest cluster center while minimizing the squared distances of data members to the assigned cluster centers. Although seems easy, this optimization problem is NP-hard and most of the implementations include approximate solutions. A well-known algorithm in k-means clustering is Lloyd's algorithm and it is referred as *k-means algorithm* and its variation based on random initialization *k-means++* are formalized in the study of Arthur and Vassilvitskii [5]:

Definition 3.5.1. k-means Algorithm

1. Arbitrarily choose initial k centers: $C = c_1, c_2, ...c_k$
2. For each $i \in 1, ...k$; set the cluster C_i to be the set of points that are closer to c_i than they are to c_j for all $i \neq j$.
3. For each $i \in 1, ...k$, set c_i to be the center of mass of all points in C_i where $c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$
4. Repeat Step 2 and 3 until C no longer changes.

Definition 3.5.2. k-means++ Algorithm

1. Take one center c_1, chosen uniformly from the set of data points X.
2. Take a new center c_i, chosen from the set of data points X with probability $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$ where $D(x)$ is the shortest distance from a data point to the closest cluster center.
3. Repeat Step 2 until k cluster centers are selected.

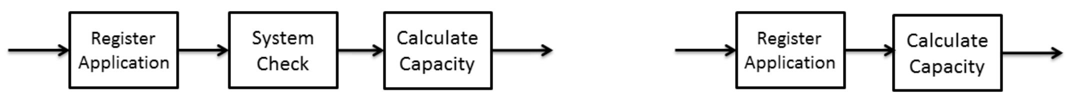


Figure 3.4: Example of Skipped Activity Pattern

4. Proceed with the 2-4 steps of k-means Algorithm.

In this thesis study, clustering of performance analysis results is undertaken and since the number of data instances low, an approach focused on initialization is selected. Implementation of *k-means++* in WEKA (Waikato Environment for Knowledge Analysis) [20] is used as Java API to call from ProM Framework [42].

3.6 Mismatch Patterns in Process Models

In cross-organizational process mining environment, there is a need to align processes of different organizations and in this scope both the organizations and their processes are similar but have significant differences. In order to align these processes and organizations, there is a need for spotting differences between process models. In the study of Dijkman [13], a collection of patterns to describe frequent mismatches between the similar process models are presented. Mismatch patterns are grouped into three as authorization, activity and control flow mismatch patterns. Authorization mismatch patterns are based on assignment of the same tasks to different roles in different processes and left outside the scope of this thesis. Activity mismatch patterns are based on representing the tasks of a process by a different collection of activities in a different process, or not representing at all. Within the scope of this study, the related activity mismatch patterns are defined in study [13] as follows:

Skipped Activity An activity exists in one process but no equivalent activity is found in the other process as illustrated in Figure 3.4.

Refined Activity An activity exists in one process but, as an equivalent, a collection of activities are existing in the other process to achieve the same task. An illustration is provided in Figure 3.5.

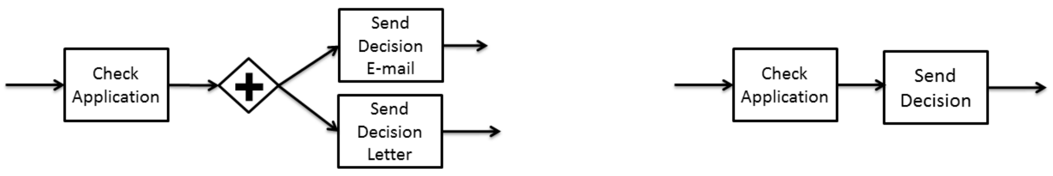


Figure 3.5: Example of Refined Activity Pattern

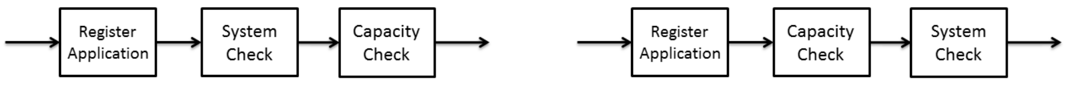


Figure 3.6: Example of Activities at Different Moments in Process Pattern

Control flow mismatch patterns are based on using different control-flow relations and dependencies for the same activities in different processes. Within the scope of this thesis study, the following related control flow mismatch patterns are defined as given in [13]:

Activities at Different Moments in Processes Set of activities are undertaken with different orders in different processes as shown in Figure 3.6.

Different Conditions for Occurrence Set of dependencies are same for two processes; however, occurrence condition is different. An example is provided in Figure 3.7.

Different Dependencies Dependency set of activities differ in different organizations. An example of this pattern is shown in Figure 3.8.

Additional Dependencies This pattern is a special case of different dependencies where one set of activities includes the other and results with additional dependencies as illustrated in Figure 3.9.

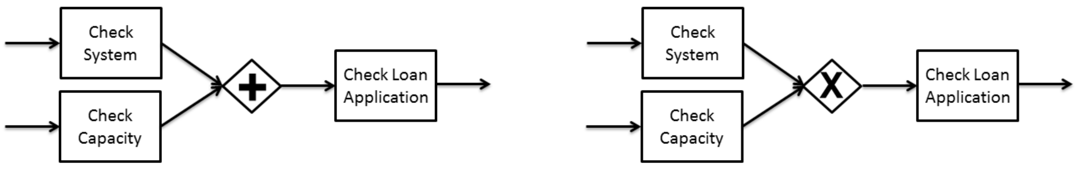


Figure 3.7: Example of Different Conditions for Occurrence Pattern

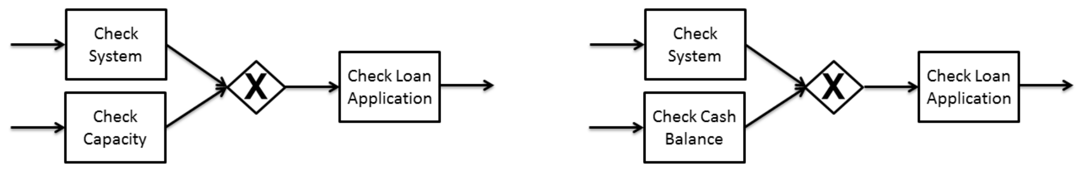


Figure 3.8: Example of Different Dependencies Pattern

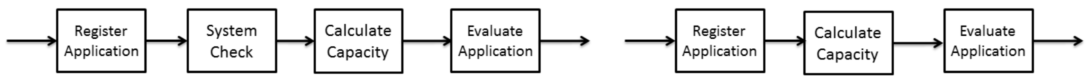


Figure 3.9: Example of Additional Dependencies Pattern

As mentioned in the study [13], their approach does not create a comprehensive list to resolve all mismatches but the most common ones spotted during case studies. In addition, from their definitions and examples it can be easily seen that these patterns are not orthogonal. Moreover, there are no algorithms provided to spot these mismatches in [13] or consequent studies, and thus implementation of spotting mismatch patterns are kept within the scope of this thesis study.

CHAPTER 4

METHODOLOGY

In this chapter, methodology proposed in this thesis study is presented. Firstly, approach overview is described from a high-level perspective. Then, each stage in the methodology is presented together with their importance in the study, mathematical representations and definitions; and black-box diagrams. In the last section of this chapter, implementation details of this methodology in ProM framework is explained in detail with a software architecture overview.

4.1 Approach Overview

Approach proposed in this study consists of four main stages and general information about these stages can be summarized as follows:

Process Model Mining: Process models are extracted from event logs for each organization with a user specified noise threshold.

Performance Indicator Analysis: Event logs are replayed on process models and performance indicators are calculated for each organization. Using these indicators, organizations are clustered based on how well they are operating.

Mismatch Pattern Analysis: Differences between process models of organizations are extracted with a well-established mismatch patterns.

Recommendation Generation: Using the performance indicator clusterings and differences between process models, a set of recommendation for each organization is generated.

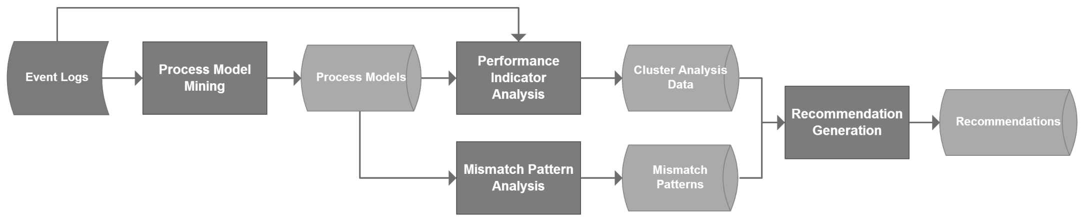


Figure 4.1: Overview of Methodology

Flow of these stages with the important input and outputs can be visualized in Figure 4.1. In the following sections, each stage will be explained in detail with their mathematical backgrounds.

4.2 Process Model Mining

Process model mining stage in the proposed approach has the aim of creating reproducible and generalized process models from event logs. In order to achieve this aim, implementation of *Inductive Miner Infrequent (IMi)*, which is proposed in [26] as an extension to *Inductive Miner* to handle noise in the event logs, is used in this study.

The selected implementation has the ability of pruning data to handle noise in the event logs. Like the other data mining approaches, event logs include data related to the infrequent behaviors occurred in real life. Although these infrequent behaviors might be caused by important structural or case related issues that should be analyzed; they make the process of mining and results more complex. Within the scope of this study, without cleaning the data, most of the process model mining approaches result with *spaghetti-like models* [33] which are difficult to further analyze. Since this thesis focuses on learning from the cross-organizational process mining rather than creating the best-fitting process models, data cleaning and noise handling is a necessary step. Therefore preprocessing steps are undertaken with a compatible approach of *Inductive Miner Infrequent (IMi)*, where data is cleaned in every inductive step.

Considering the scope of this study, instead of computational details of *Inductive Miner Infrequent (IMi)*; black-box representation is used to explain its usage in the methodology. In order to set a filtering threshold, a user-provided value between 0 to 1 is added as input to method in addition to event logs. As a result, workflow net

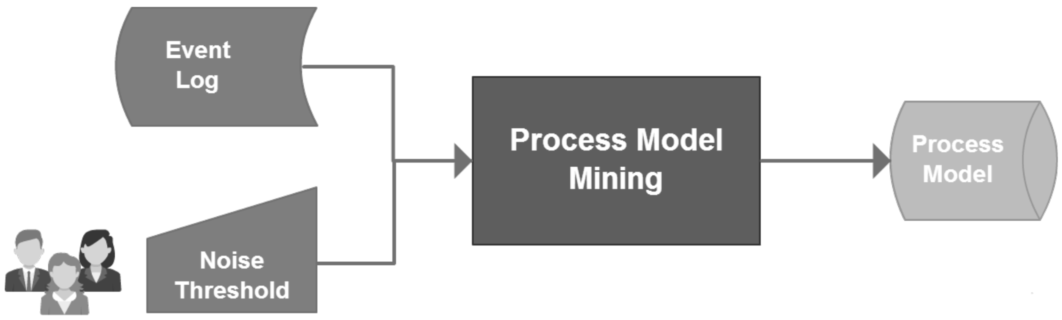


Figure 4.2: Process Model Mining Stage as Black-box

is produced which is a sound and properly completed Petri net without deadlocks. Black-box of this stage is illustrated in Figure 4.2 and this stage is called for every organization's event logs to create their own process models in Workflow net formalization.

4.3 Performance Indicator Analysis

Performance indicator analysis stage focuses on calculating and analyzing the performance values using the event logs and mined process models. This stage consists of mainly two concepts as *a*) alignment and calculation of performance indicators; and *b*) clustering of organizations based on their performance values. In order to evaluate the performance of an organization based on their process models and past activities; there is a number of indicators in time dimension, cost dimension and utilization [33]. However, in this study, process related performance values are considered since differences in the process models are studied in the next stages. With this reasoning, the following performance indicators are calculated and analyzed in this stage of methodology:

Average Time Between Activities For each activity in the process model, average time to reach other activity is calculated. This is a simple but powerful performance metric for organizations since it can yield the average time to complete one task based on a starting point. From the performance perspective, organizations want to minimize average time between activities to increase their throughput [35]. This notion can be defined as follows:

Definition 4.3.1. Average time between activity A and B in organization i is $AvgTime^{i}_{A \to B} = \frac{\sum_{Case\ c \in EventLog_i} TimeBetween_c(A,B)}{|Occurences_{Event\ Log_i}(A,B)|}$ where

1. $TimeBetween_c(A, B) = EndTime_c(B) - StartTime_c(A)$
2. $StartTime_c(A)$ is start time of activity A in case c,
3. $EndTime_c(B)$ is end time of activity B in case c,
4. $|Occurences_{EventLog_i}(A, B)|$ is number of occurrences of activity A followed by B in $Event\ Log_i$.

Standard Deviation of Time Between Activities Time between activities in real life is not stable and they deviate due to various reasons such as people responsible of tasks, size and the content of tasks or seasonality [33]. On the other hand, organizations want to be confident about their processes and therefore they want to minimize the deviation in the time between activities. Minimized deviation in time helps organizations to plan, act and re-organize the activities in the processes with high accuracy [35]. With the same approach above, the following formulation can be defined:

Definition 4.3.2. Standard deviation time between activity A and B in organization i is $StdDevTime^{i}_{A \to B} = \sqrt{\frac{\sum_{Case\ c \in EventLog_i} [TimeBetween_c(A,B) - AvgTime^{i}_{A \to B}]^2}{Occurences_{Event\ Log_i}(A,B)}}$

In addition to average and standard deviation, minimum and maximum times between activities can also be analyzed. However, these performance values are mostly result of rare cases in the event logs and these rare occurrences have the probability of being eliminated as noise in process mining stage. Therefore, the minimum and maximum values have the risk of not representing the actual performances. Thus, only average and standard deviation of the time between activities are selected in the study.

4.3.1 Replay and Performance Indicator Calculation

Replay of event logs on process models is based on the idea of *alignment* which is formalized in [35] and the basic assumption in this concept is that process models and event logs have the same activity labels. Alignment is based on *moves in the model* *and log* and in order to have a successful replay where optimal alignment should be achieved. As proposed in [2, 3], A^* algorithm which is a path-finding algorithm based on graphs is used to find optimal alignment of event logs on the process models.

In order to appropriately apply A^* algorithm, there are number of manual and computational steps that should be undertaken. The following prerequisite steps are implemented in [3] to apply A^* algorithm:

Set Label Patterns between Process Model and Event Log: In the event logs, there are various different transitions of the same activity which are generally not represented in process models. For instance, there can be *"Activity A, Start"* and *"Activity A, End"* in the event log; however they are reflected as *"Activity A"* in the process model. Therefore, a list of all transitions and events are asked to the user to match to the ones in the process model in terms of patterns or regular expressions.

Create Initial and Final Markings: In case of there is no definite starting or ending point in the process model, user input is necessary to define these activities.

Set Cost Values: Since A^* algorithm is based on alignments which are basically moves along process models and event logs, the following cost values for each move is necessary to be set: *a*) move on process model, *b*) move on event log; and *c*) synchronous move.

In order to use replay as an intermediate stage in this thesis, some of the mentioned steps are automatized with the help of the assumptions in the prior and post stages of methodology. With this reasoning, initial and final markings are created with the first and last activities in the process models. In addition, since there is no explicit priority of process model over event log or vice versa, cost values are set to 1 for both *move on process model* and *move on event log*. Since there is no penalty for *synchronous moves*, its cost value is set to 0. However, since each event log has different set of transition labels, manual user input is still necessary to map label patterns between process model and event log. With the generated and user-specified inputs, replay and performance indicator calculation in the methodology can be visualized in Figure 4.3. For each organization, the steps in the diagram are followed with the corresponding

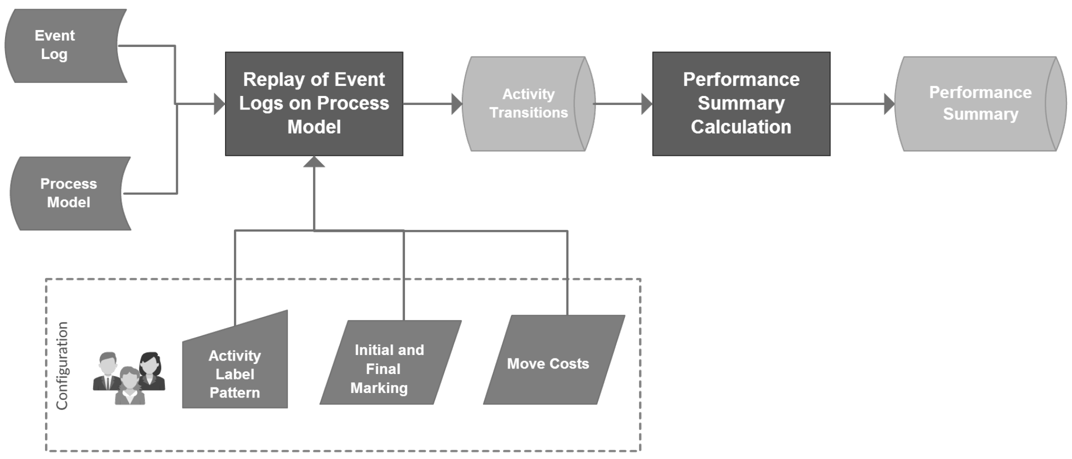


Figure 4.3: Replay and Performance Indicator Calculation Stage as Black-box

event logs and process models; and the resulting process summaries are used for further analysis.

Performance summary calculation step at the end of this stage is used to create a summary of data consists of average and standard deviation of time between activities. Resulting data can be defined as follows:

Definition 4.3.3. Performance Summary data for any organization i is $PerfSum_i = \{AvgTimeSum_i \cup StdDevTimeSum_i\}$ where

1. $AvgTimeSum_i = \{AvgTime^i_{A \to B} | A, B \in Event\ Log_i\}$
2. $StdDevTimeSum_i = \{StdDevTime^i_{A \to B} | A, B \in Event\ Log_i\}$

4.3.2 Performance Indicator Clustering

Clustering is based on the idea of collecting the set of observations into clusters so that observations within the same cluster are similar whereas the observations from different clusters are dissimilar. Being an unsupervised learning method, it aims to find hidden structures in unlabeled data. In this thesis study, clustering is used to gather organizations based on their performance indicator data. In other words, organizations will be set into groups based on how much time in average and with variation takes to complete an activity given a starting point.

Within clustering analysis, various algorithms are proposed to find clusters based on the idea of decreasing in-cluster distances, increasing space density or fitting to particular statistical distributions. However, in this study, a generic approach based on centroid-based clustering is exploited. For the fixed number of *k* clusters, the approach is well-known as *k-means clustering*. Considering the popularity of this approach and its extensions, random initialization based *k-means*++ approach from the study of Arthur and Vassilvitskii [5] is used to cluster organizations.

Clustering methods can be evaluated by the internal and external methods. Internal methods are based on the data that is clustered itself whereas external methods use the additional information such as labels or benchmarks. Although there are various well-established metrics in external methods such as Rand Measure, F-measure, Jaccard index or Confusion Matrices; in this thesis study there is no pre-labeled organizations based on their performance indicators. Therefore, internal evaluation methods are used to assess the quality of clusters. Considering the fact that actual data to cluster is *time interval* and centroid-based clustering is applied in this study, a metric based on decreasing the in-cluster distances is selected. With this consideration, Sum of Squared Error (SSE) is calculated with the following definition. When the clustering results are compared, it is reasonable to choose the one with the least SSE; however, it is a fact that as *k* converges to the number of original sample size SSE value decreases. Therefore, SSE values and number of clusters are plotted and presented to the end user of the methodology to select optimal number of clusters.

Definition 4.3.4. Sum of Squared Error (SSE) is $SSE = \sum_{i=1}^{k} \sum_{x \in c_i} EuclideanDistance(mean_i, x)^2$ where

1. x is a data point in cluster c_i.
2. $mean_i$ is the mean vector of the cluster c_i.

For each organization, *Performance Summary* data is used as the data source in clustering. It is aimed that the organizations with the similar average or standard deviation time between tasks will be assigned to same sets to further reveal the dissimilarities that the other clusters can learn from. Since not every activity exists in every organization, performance summary data is cleaned and merged to include only all shared

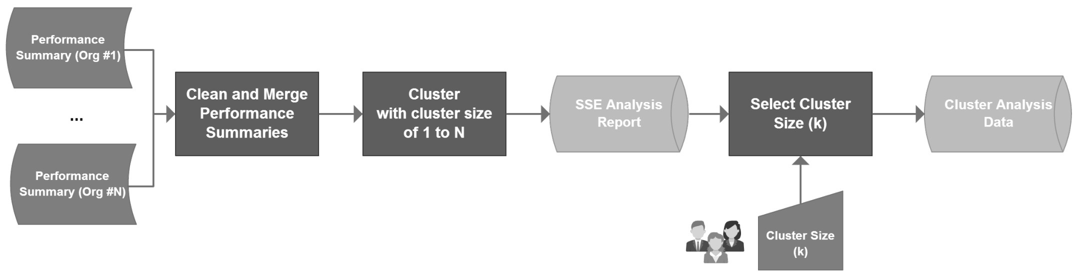

Performance Summary (Org #1)
...
Performance Summary (Org #N)
Clean and Merge Performance Summaries
Cluster with cluster size of 1 to N
SSE Analysis Report
Select Cluster Size (k)
Cluster Analysis Data
Cluster Size (k)

Figure 4.4: Performance Indicator Clustering Stage as Black-box

activities. This cleaning step removes the non-existing activity related performance indicators from all the organizations so that clustering is applied on the data with no process model related information. Since number of clusters are not known priori, k-means clustering is applied starting *k* from 1 to the number of organizations. For each clustering with different number of clusters, SSE values are plotted and user is made to select the appropriate cluster size. For the selected cluster size, clustering related information is used to generate recommendations in the further steps. This stage of methodology can be visualized as an input-output system in Figure 4.4. Resulting cluster analysis data is formulated as follows:

Definition 4.3.5. Cluster Analysis Data is a tuple $(k, Assignments, Cluster\ Centroids)$ where

1. k is the number of clusters,
2. $Assignments$ is a set of tuple $(Organization_i, Cluster_j)$ where i is identifier for organization and $j \leq k$ is identifier for cluster,
3. $Cluster\ Centroids$ is a set of tuple $(Cluster_j, Type, A_{start}, A_{end}, Value)$ where
   (a) $Type$ is performance indicator type which is $Average$ or $StandardDev$,
   (b) A_{start} and A_{end} are starting and ending points of performance indicator,
   (c) $Value$ is the actual value of performance indicator,
   (d) $Cluster\ Centroids_j$ is a function that returns a set of $Centroid$ which is a tuple $(Type, A_{start}, A_{end}, Value)$ for $Cluster_j$.

4.4 Mismatch Pattern Analysis

In order to learn from other organizations, it is necessary to spot the differences between process models of different organizations. In this phase of methodology, differences between process models will be revealed by the mismatch patterns which are defined by Dijkman [13]. In the process mining stage, process models are mined to create Workflow nets; however, mismatch pattern definitions are closer to business environment and it is more suitable spot patterns in BPMN notation. Thus, mined process models are converted to BPMN models and then mismatch pattern analysis is applied. After spotting differences, mismatch patterns for activities and control flows are revealed for the organizations.

Business Process Modeling Notation (BPMN) is one of the widely accepted modeling language in the industry and its a standardized notation by the Object Management Group (OMG) since 2004. Main aim of BPMN is to provide a notation that is easily understood by business stakeholders. In the preceding stages, Petri nets and their subset as Workflow nets are used for mining since they have stronger mathematical background. However, in this stage, BPMN formulation is used since mismatch patterns have roots in real-life business environment [13]. The following definition and conversion are defined in the study [23] and they are used in the mismatch pattern analysis.

Definition 4.4.1. A BPMN model is a tuple $BPMN_{model} = (N, A, G_{XOR}, G_{AND}, e_{start}, E_{end}, SF, \lambda)$, where

1. N is a set of flow nodes,
2. $A \subseteq N$ is a set of activities,
3. $G_{XOR} \subseteq N$, $G_{AND} \subseteq N$ are sets of exclusive and parallel gateways,
4. $e_{start} \in N$ is a start event,
5. $E_{end} \in N$ is a set of end events,
6. Sets $A, G_{XOR}, G_{AND}, \{e_{start}\}, E_{end}$ form a partition of N,
7. $SF \subseteq N \times N$ is a set of sequence flows,

8. $\lambda : N \to U_A$ is a labeling function, where U_A is some universe of activity labels,
9. Start event e_{start} does not have incoming sequence flows and more than one outgoing sequence flows,
10. End events E_{end} do not have outgoing sequence flows.

Definition 4.4.2. Constructing BPMN Model from Petri nets, which are $N = (P, T, F)$ where P is finite set of *places*, T is finite set of *transitions* and F is set of *flow relations* to BPMN models consists of the following steps:

1. Initialize BPMN model with a start node e_{start}.
2. Convert all transitions, T, in the Petri net by creating an activity in BPMN model for each transition in Petri net.
3. Convert all places, P, from Petri net to BPMN routing constructs by finding the corresponding sequence flows.

For each organization, mined process models are converted to BPMN models and mismatch patterns analysis is undertaken on them, in order to locate differences between two process models or cluster of process models. In addition, since performance indicators are calculated based on a starting and ending points in the process model, same approach is applied to locate mismatch patterns. In other words, differences of process models are located through a starting activity to an ending activity. When the complete set of mismatches between process models are required, this approach is used with starting and ending points as source and sink activities. Mismatch patterns are formulated and their importance in the methodology are listed as following:

Skipped Activity Skipped activities are the operations that are not undertaken with some of the organizations. In business environment there could be various different reasons for an organization to exclude any activity in their process flows which are followed by other organizations. Therefore, this mismatch pattern should not be utilized solely but taken in to care with the other patterns. In this study, for each organization a list of activities that they do not include are listed as *Skipped Activity* with the following definition. In addition, these differences are checked for the activities with a particular start and end point.

Definition 4.4.3. Skipped Activity pattern is a tuple $Skipped\,Activity = (O, SA, A_{start}, A_{end})$ where

1. O is the identifier for organization,
2. BPMN model of organization is $BPMN_O$ and $BPMN_*$ is the set of all BPMN models in analysis,
3. SA is a set of skipped activities and $SA = BPMN_*(A) \setminus BPMN_O(A)$,
4. A_{start} and A_{end} are starting and ending points to check mismatch patterns and $A_{start}, A_{end} \in BPMN_O(A)$.

Definition 4.4.4. Skipped Activity Analyzer is a function $SkippedActivityAnalyzer(O, A_{start}, A_{end})$ and it returns a *Skipped Activity* for the organization O and the activities between A_{start} and A_{end}.

Refined Activity Refined activities exist in one process; however, as equivalent a collection of activities are undertaken in another organization's process to complete the same task. Since there is no activity ontology information is kept in event logs and process models, there is no direct way to define whether the tasks can be classified as other tasks' refined state. In this study, assuming the labels of activities are correct and explanatory about their enclosures, similarity between labels are utilized for this aim. Using *Levenshtein distance*, which is very popular in information retrieval area and also known as *edit distance*, similarity of activity labels are calculated for each activity in respect to activities of other organizations' process models. This mismatch pattern presents information about the similar but not same activities in different process models which can be used to make organizations learn from each other. With a user-defined threshold for similarity based on the nature of business environment, most similar tasks can be listed for being refined.

Definition 4.4.5. Refined Activity pattern is a tuple $Refined\,Activity = (O_1, O_2, RA, SimilarityMap, A_{start}, A_{end})$ where

1. O_1 is the first organization where the refined activity is undertaken with the BPMN model of $BPMN_{O_1}$,
2. O_2 is the second organization where a collection of activities checked for similarity with the BPMN model of $BPMN_{O_2}$,
3. RA is the refined activity and $RA \in BPMN_{O_1}(A)$ and $RA \notin BPMN_{O_2}(A)$,
4. $SimilarityMap$ is a set of tuples $(SimilarityValue, B)$ where $SimilarityValue$ indicates the similarity between RA and $B \in BPMN_{O_2}(A)$,
5. A_{start} and A_{end} are starting and ending points to check mismatch patterns and $A_{start}, A_{end} \in BPMN_{O_{1,2}}(A)$.

Definition 4.4.6. Refined Activity Analyzer is a function $RefinedActivityAnalyzer(O_1, O_2, A_{start}, A_{end})$ and it returns a set of *Refined Activity* for the organization O_1 compared to O_2 for the activities between A_{start} and A_{end}.

Activities at Different Moments in Processes Activities at different moments means that organizations are undertaking activities at different orders in their process flows. This mismatch pattern indicates that the organizations could achieve the same task when they change the order of activities and this information can be used by others to enhance their process models. Therefore, formulation of this pattern includes an activity; and its previous and next tasks at different organizations.

Definition 4.4.7. Activities at Different Moments in Processes pattern is a tuple $Different\,Moments = (O_1, O_2, A, Prev_1, Next_1, Prev_2, Next_2, A_{start}, A_{end})$ where

1. O_1 is the first organization with the BPMN model of $BPMN_{O_1}$,
2. O_2 is the second organization with the BPMN model of $BPMN_{O_2}$,
3. A is the activity that is undertaken in different order and $A \in BPMN_{O_{1,2}}(A)$,
4. $Prev_1, Next_1,$ are previous and next tasks of A in O_1 and $Prev_1, Next_1 \in BPMN_{O_1}(A)$,

5. $Prev_2, Next_2,$ are previous and next tasks of A in O_2 and $Prev_2, Next_2 \in BPMN_{O_2}(A)$,
6. In order to ensure different moments $Prev_1 \neq Prev_2$ or $Next_1 \neq Next_2$,
7. A_{start} and A_{end} are starting and ending points to check mismatch patterns and $A_{start}, A_{end} \in BPMN_{O_{1,2}}(A)$.

Definition 4.4.8. Different Moments Analyzer is a function $DifferentMomentsAnalyzer(O_1, O_2, A_{start}, A_{end})$ and it returns a set of *Different Moments* for the organization O_1 compared to O_2 for the activities between A_{start} and A_{end}.

Different Conditions for Occurrence Different conditions for occurrence take place where an activity has the same dependencies in different organizations but with different conditions. For instance, an organization needs to complete all dependency tasks before starting the next one; though another organization checks for the completion of only one of these dependencies. This difference between organizations can lead to loosening or tightening of the conditions based on the application of other organizations. Considering the representation of different conditions for occurrence and other control flow mismatch patterns, a generic definition is developed firstly to use and further customize. Different conditions for occurrence pattern definition is developed based on this generic definition as following.

Definition 4.4.9. Control Flow Mismatch pattern is a tuple $Control\ Flow\ Mismatch = (O_1, O_2, A, Prev_1, G_1, Prev_2, G_2, A_{start}, A_{end})$ where

1. O_1 is the first organization with the BPMN model of $BPMN_{O_1}$,
2. O_2 is the second organization with the BPMN model of $BPMN_{O_2}$,
3. A is the activity that is in front of control flow and $A \in BPMN_{O_{1,2}}(A)$,
4. $Prev_1$ and $Prev_2$ are set of tasks that are connected to gateway G_1 and G_2 in organizations O_1 and O_2 respectively with the following requirements:
   (a) $Prev_1 \subset BPMN_{O_1}(A)$,
   (b) $Prev_2 \subset BPMN_{O_2}(A)$,

(c) $G_1 \in BPMN_{O_1}(G_{XOR}, G_{AND})$,

(d) $G_2 \in BPMN_{O_2}(G_{XOR}, G_{AND})$.

5. A_{start} and A_{end} are starting and ending points to check mismatch patterns and $A_{start}, A_{end} \in BPMN_{O_{1,2}}(A)$.

Definition 4.4.10. Different Conditions for Occurrence pattern is a *Control Flow Mismatch* where

1. $Prev_1 = Prev_2$,
2. $G_1 \neq G_2$.

Definition 4.4.11. Different Conditions Analyzer is a function $DifferentConditionsAnalyzer(O_1, O_2, A_{start}, A_{end})$ and it returns a set of *Different Conditions for Occurrence* for the organization O_1 compared to O_2 for the activities between A_{start} and A_{end}.

Different Dependencies When the dependency set of a specific activity is different in organizations, it can be concluded that this activity and the relation between its dependency sets should be reviewed since it can be achieved in both environments. Organizations can learn from each other to eliminate or enhance their process flows by changing these dependency sets. This can include reorganization of activities as well as removing and adding new activities. The idea is structured in the following simplified definition.

Definition 4.4.12. Different Dependencies pattern is a *Control Flow Mismatch* where

1. $Prev_1 \neq Prev_2$,
2. $G_1 = G_2$.

Definition 4.4.13. Different Dependency Analyzer is a function $DifferentDependencyAnalyzer(O_1, O_2, A_{start}, A_{end})$ and it returns a set of *Different Dependency* for the organization O_1 compared to O_2 for the activities between A_{start} and A_{end}.

Additional Dependencies Additional dependency pattern is a special case of *Different Dependencies* where additional tasks are required in one organization

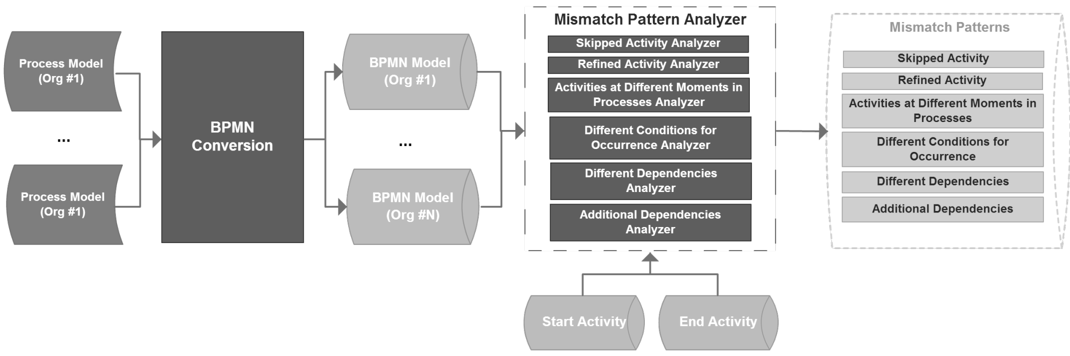

Process Model (Org #1)
...
Process Model (Org #1)
BPMN Conversion
BPMN Model (Org #1)
...
BPMN Model (Org #N)
Mismatch Pattern Analyzer
Skipped Activity Analyzer
Refined Activity Analyzer
Activities at Different Moments in Processes Analyzer
Different Conditions for Occurrence Analyzer
Different Dependencies Analyzer
Additional Dependencies Analyzer
Start Activity
End Activity
Mismatch Patterns
Skipped Activity
Refined Activity
Activities at Different Moments in Processes
Different Conditions for Occurrence
Different Dependencies
Additional Dependencies

Figure 4.5: Mismatch Pattern Analysis Stage as Black-box

in order to start an activity. These additional dependencies can include provisional tasks that are related to the original activity such as *checking system* before *starting application*.

Definition 4.4.14. Additional Dependencies pattern is a *Different Dependencies Mismatch* where $Prev_1 \subset Prev_2$ or $Prev_2 \subset Prev_1$.

Definition 4.4.15. Additional Dependency Analyzer is a function $AdditionalDependencysAnalyzer(O_1, O_2, A_{start}, A_{end})$ and it returns a set of *Additional Dependency* for the organization O_1 compared to O_2 for the activities between A_{start} and A_{end}.

Mismatch pattern analysis stage in methodology can be visualized as a black-box diagram in Figure 4.5. Each organization's mined process models are converted to BPMN models and these models are sent to mismatch pattern analyzers with the starting and ending activities. For each organization, mismatch patterns are gathered and stored for the further analysis as presented in Algorithm 1.

4.5 Recommendation Generation

Recommendation generation stage in the methodology is the final and core stage where all information retrieved from event logs until now are utilized. Until this point, all event logs are used to mine process models of each organization and then these event logs are replayed on the process models to calculate performance indicators. Following these steps, the organizations are clustered based on their performance in-

Algorithm 1: Mismatch Pattern Analysis

```
Input: $O_1$ first organization, $O_2$ second organization, $A_{start}$ starting activity,
        $A_{end}$ ending activity
Output: $MismatchPatterns$ a set of mismatch patterns
1 $MismatchPatterns \leftarrow \{\}$
2 $MismatchPatterns \leftarrow SkippedActivityAnalyzer(\mathrm{O}_1, A_{start}, A_{end})$
3 $MismatchPatterns \leftarrow RefinedActivityAnalyzer(\mathrm{O}_1, O_2, A_{start}, A_{end})$
4 $MismatchPatterns \leftarrow DifferentMomentsAnalyzer(\mathrm{O}_1, O_2, A_{start}, A_{end})$
5 $MismatchPatterns \leftarrow DifferentConditionsAnalyzer(\mathrm{O}_1, O_2, A_{start}, A_{end})$

6 $MismatchPatterns$
  $\leftarrow DifferentDependencysAnalyzer(\mathrm{O}_1, O_2, A_{start}, A_{end})$
7 $MismatchPatterns$
  $\leftarrow AdditionalDependencysAnalyzer(\mathrm{O}_1, O_2, A_{start}, A_{end})$
8 return $MismatchPatterns$
```

dicators and mismatches between their process models are listed. In this stage, clustering results, which are performance indicator vectors for each cluster will be used to generate recommendations for each cluster of organizations based on the differences between their mismatch patterns.

In this study, idea of recommendation is based on providing a set of mismatch patterns for each organization so that they can enhance their processes. These mismatch patterns are generated by comparing the process models of other organizations, particularly which are performing better in terms of their performance indicator values. In addition, clustering of organizations is undertaken to generalize the way of identifying which organizations perform better in this environment. Recommendation idea and recommendation generation function is defined as following:

Definition 4.5.1. Recommendation is a tuple $Recommendation = (O, A_{start}, A_{end}, Mismatch\ Patterns)$ where

1. O is identifier for organization,

2. A_{start} and A_{end} are starting and ending activities in between the recommenda-

tions are checked,

3. $Mismatch\ Patterns$ is collection of mismatch patterns.

Definition 4.5.2. Recommendation generation is a function that is $RecGen(O, C, P)$ and it returns a set of $Recommendation$ where

1. O is identifier for organization,

2. C is *Cluster Analysis Data* which is result of cluster analysis stage,

3. P is *Performance Threshold* which is a real number larger than or equal to 0 and it is calculated over the same type of performance indicators of different organizations in *Cluster Analysis Data*.

Algorithm of recommendation generation function is based on the idea of checking other clusters for a significant change in performance indicators, where significance is defined by the threshold provided by user. After finding significant difference, all organizations in other clusters are checked against mismatch patterns with the starting and ending activities defined in performance indicators. With this constraining, only mismatches which are located between the activities that causes high level of difference in performance indicators are analyzed. This approach is formalized in Algorithm 2.

This stage of methodology can be visualized as gathering inputs of mismatch patterns data and cluster analysis data in Figure 4.6. In addition, a performance difference threshold is gathered from the user to specify how much difference between clusters is necessary to check for mismatch patterns and generate recommendations. This stage generates recommendation data for each organization which contains a set of mismatch patterns.

Algorithm 2: Recommendation Generation

```
Input: $O$ organization, $C$ Cluster Analysis Data, $P$ performance difference threshold
Output: $Recommendations$ a set of recommendations
1  $Recommendations \leftarrow \{\}$
2  $i \leftarrow C(Assignments(O))$
3  for $Centroid \in C(ClusterCentroids_i)$ do
4      for $Centroid' \in C(ClusterCentroids_j)\ i \neq j$ do
5          if $Centroid(A_{start}) = Centroid'(A_{start})$ & $Centroid(A_{end}) = Centroid'(A_{end})$ then
6              if $(|Centroid(Value) - Centroid'(Value)| \div Centroid(Value)) \geq P$ then
7                  $A_{start} \leftarrow Centroid(A_{start})$
8                  $A_{end} \leftarrow Centroid(A_{end})$
9                  $MismatchPatterns \leftarrow \{\}$
10                 for $O' \in C(Assignments(j))$ do
11                     $MismatchPatterns \leftarrow MismatchPatternAnalysis($O,O',A$_{start}$,$A_{end}$)
12                 $Recommendations \leftarrow Recommendation($O,A$_{start}$,$A_{end}$, $MismatchPatterns$)
13 return $Recommendations$
```

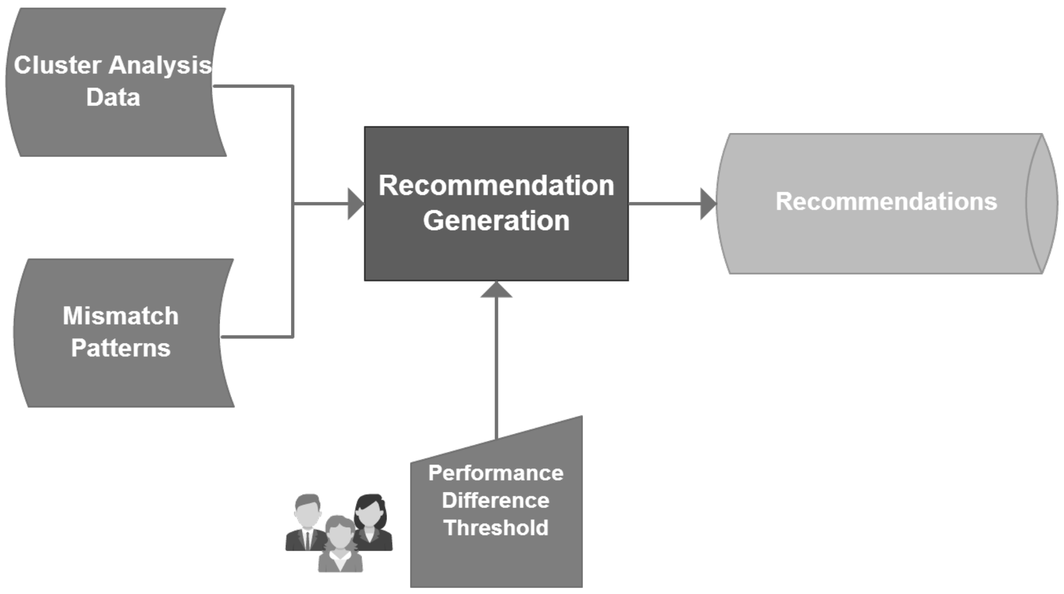


Figure 4.6: Recommendation Generation Stage as Black-box

4.6 Implementation in ProM Framework

Methodology of this thesis study is implemented in ProM [42], which is an extensible framework that supports a wide variety of process mining techniques in form of plugins. ProM is an open source framework for process mining algorithms which provides a platform to users and developers. Aim of this framework is to create a standard process mining platform that is accepted in industry and academia with an active community. Approach of this thesis study is implemented with its each stage as a standalone plugin that enables extensions for further studies. Developed set of plugins are packaged with the name of *CrossOrgProcMin*[1] and published open-source[2] being available in the latest version of ProM release.

Implementation details are presented from two perspectives in this section. Firstly, software architecture of implementation will be explained with the relationships and dependencies with the other plug-ins and ProM framework. Then, user experience perspective will be presented to show how the end user of this methodology can provide inputs, make analysis and gather results.

[1] http://www.promtools.org/prom6/packages/CrossOrgProcMin
[2] http://github.com/onuryilmaz/cross-org-proc-min

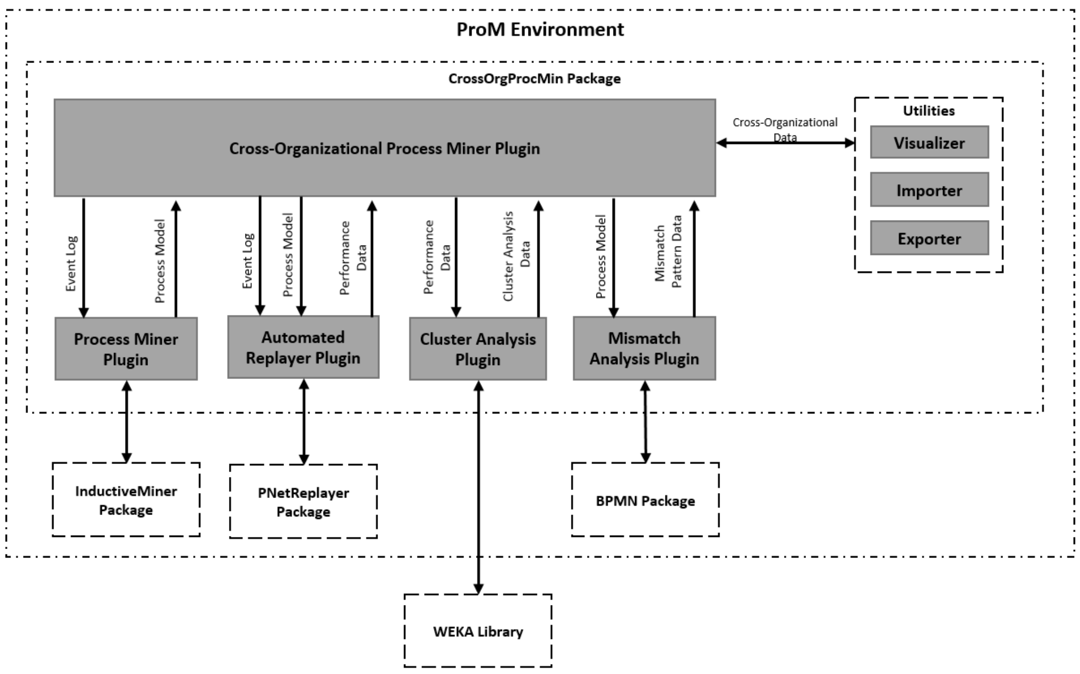


Figure 4.7: Software Architecture for Cross-Organizational Process Miner

4.6.1 Software Architecture

Approach of this study is divided into stand-alone stages where inputs and outputs between them are defined strictly. With the help of this understanding, each stage is developed as a stand-alone plug-in in ProM framework which can be called separately. For this aim, five plug-ins are developed which are visualized in Figure 4.7 to provide a high-level perspective. In addition, a set of utilities are developed to visualize and persistence of data in ProM environment.

Cross-Organizational Process Miner Plugin is the core plugin which handles management and data flow between other plugins. It basically calls each other plugin and gathers outputs from them to proceed to next stages.

Process Miner Plugin is the implementation of Section 4.2 where *Inductive Miner* [26] package in the ProM environment is utilized.

Automated Replayer Plugin is developed to execute the approach in Section 4.3.1 and it replays the event logs over process models using the libraries in *PNReplayer* package [3].

Cluster Analysis Plugin aims to cluster the organizations based on their performance indicators as presented in Section 4.3.2 and this plugin uses WEKA libraries [20] which are external to the ProM environment.

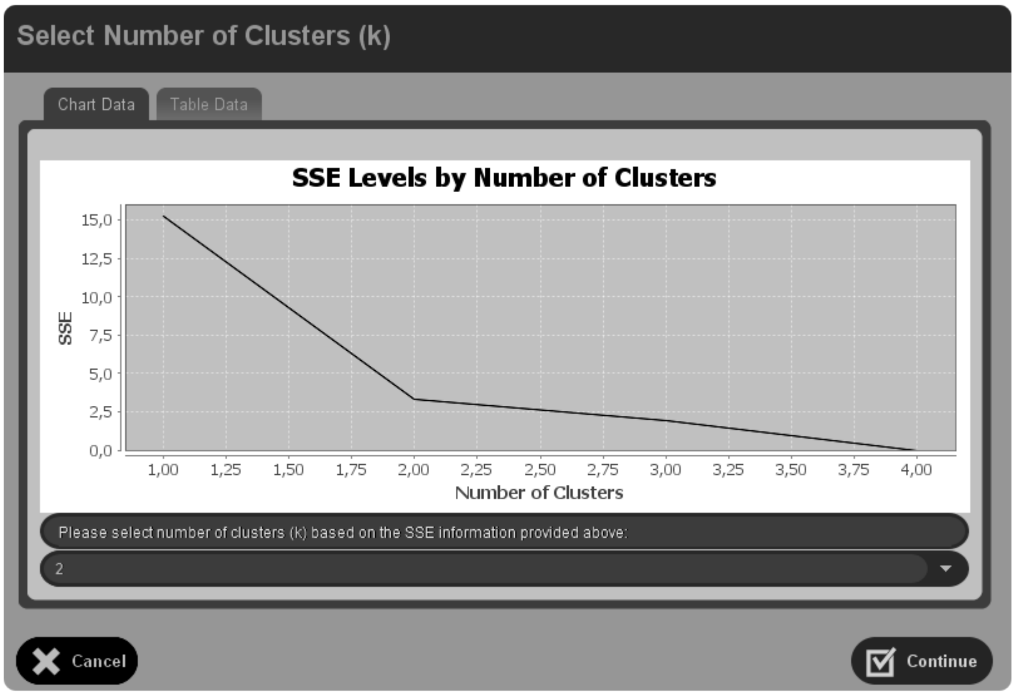
Mismatch Analysis Plugin is developed to discover differences between the process models as explained in Section 4.4. This plugin is developed from scratch since there is no implementation of mismatch patterns [13] in ProM framework.

Utilities are developed to handle data operations of the plugins in *CrossOrgProcMin* package. Import and export libraries are used to enable data specific persistence in ProM framework for these plugins. Visualization libraries are developed to represent the recommendation output to the end user in ProM screens.

4.6.2 User Experience

Approach proposed in this study is a composition of different stages and it is very difficult for an end user to be capable of each step's details. Therefore, the plug-ins are developed with a high level of automation by minimizing the manual inputs required. In addition, each reporting step is well-documented so that user can easily understand and provide inputs without help of another guide. In this section, example screens will be presented to explain the story of end user for the developed plug-ins. Assuming the user is familiar with the ProM framework, only the steps specific to the developed plug-ins are illustrated. Intermediate steps are included in Appendix A.1 and result screens are presented in this section.

When the user selects a number of event logs and starts *Cross-Organizational Process Miner*, an informative screen about the plugin and the following steps are presented to the user as shown in Figure A.1. Following this screen, user is required to provide names for organizations of the event logs in a very simple form as can be seen in Figure A.2. This step is added to provide user-friendly reports in the following stages with the names provided. In the stage of process mining, for each organization, user is expected to provide a noise threshold and naming convention which is illustrated in Figure A.3 and this screen is an extension of *Inductive Miner* [26] screens. Following this, in the replay stage, user input is required for matching activity names in



Please select number of clusters (k) based on the SSE information provided above:

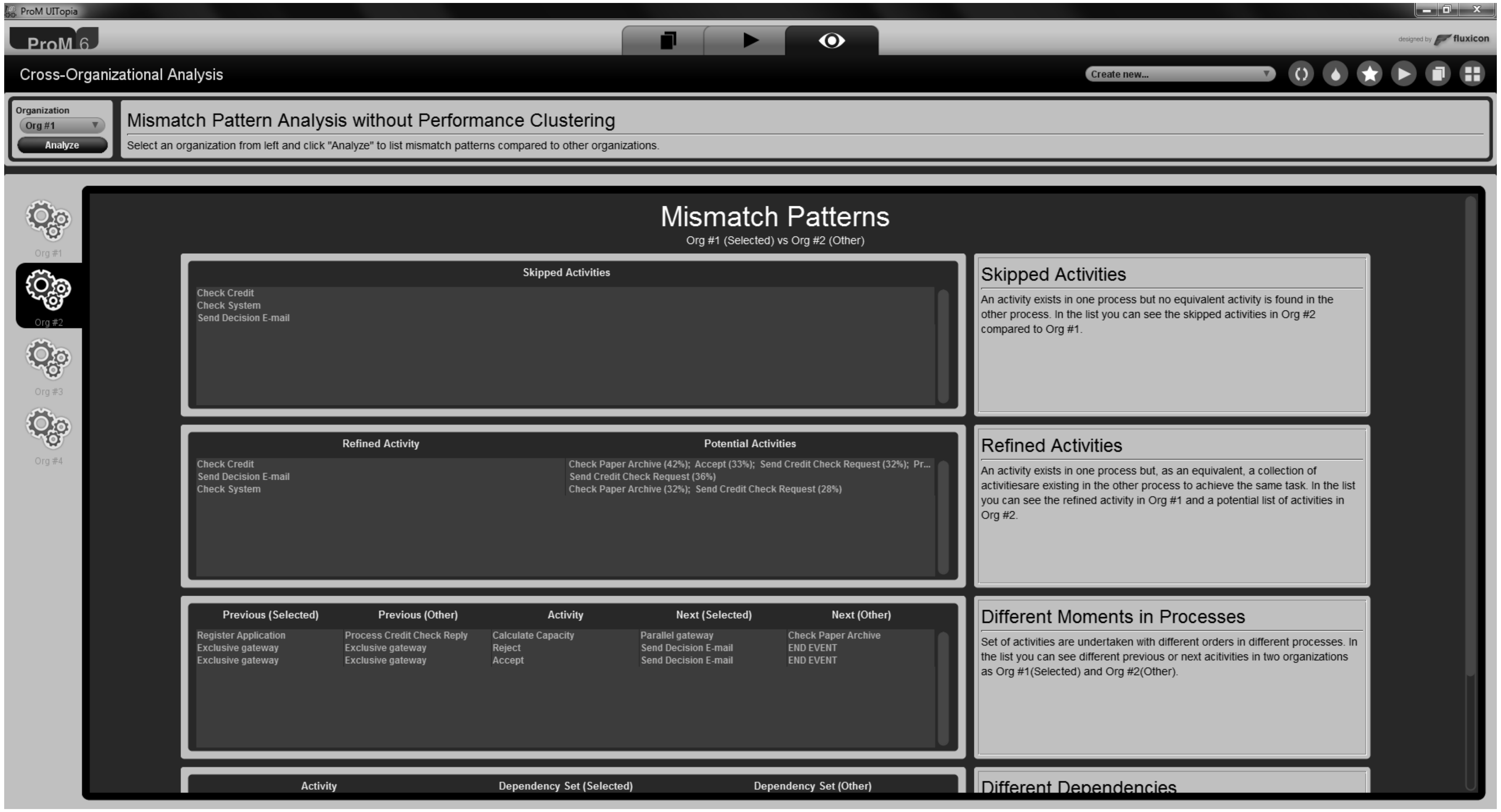
2

Figure 4.8: Select Cluster Size Screen of Cross-Organizational Process Miner Plugin

the event logs to process models as shown in Figure A.4 and this screen is derived from *PNReplayer* package [3] screens. When replay stage is completed, clustering of organizations based on performance indicators starts and user input is required for selecting the appropriate cluster size *(k)* with the SSE values plotted as shown in Figure 4.8.

With the successful execution of plugin, recommendation outputs are presented to the end user. Recommendations are visualized in two ways to create comparable outputs with and without performance clustering. When the user selects to check mismatch patterns without performance clustering, an organization should be selected at the left top of the screen in Figure 4.9. For the selected organization, all other organizations are compared and mismatch patterns are listed in separate tabs as shown in the main panel of Figure 4.9. Main idea behind adding this visualization is to provide an insight to user that how much mismatch patterns can be discovered between organizations when no performance clustering is applied.

When the user selects to check mismatch patterns with performance clustering, in addition to organization, *performance difference threshold* should be selected. For



Cross-Organizational Analysis

Organization: Org #1

Analyze

Mismatch Pattern Analysis without Performance Clustering

Select an organization from left and click "Analyze" to list mismatch patterns compared to other organizations.

Mismatch Patterns

Org #1 (Selected) vs Org #2 (Other)

Skipped Activities
Check Credit
Check System
Send Decision E-mail

Refined Activity	Potential Activities
Check Credit	Check Paper Archive (42%); Accept (33%); Send Credit Check Request (32%); Pr...
Send Decision E-mail	Send Credit Check Request (36%)
Check System	Check Paper Archive (32%); Send Credit Check Request (28%)

Previous (Selected)	Previous (Other)	Activity	Next (Selected)	Next (Other)
Register Application	Process Credit Check Reply	Calculate Capacity	Parallel gateway	Check Paper Archive
Exclusive gateway	Exclusive gateway	Reject	Send Decision E-mail	END EVENT
Exclusive gateway	Exclusive gateway	Accept	Send Decision E-mail	END EVENT

Activity	Dependency Set (Selected)	Dependency Set (Other)

Skipped Activities

An activity exists in one process but no equivalent activity is found in the other process. In the list you can see the skipped activities in Org #2 compared to Org #1.

Refined Activities

An activity exists in one process but, as an equivalent, a collection of activitiesare existing in the other process to achieve the same task. In the list you can see the refined activity in Org #1 and a potential list of activities in Org #2.

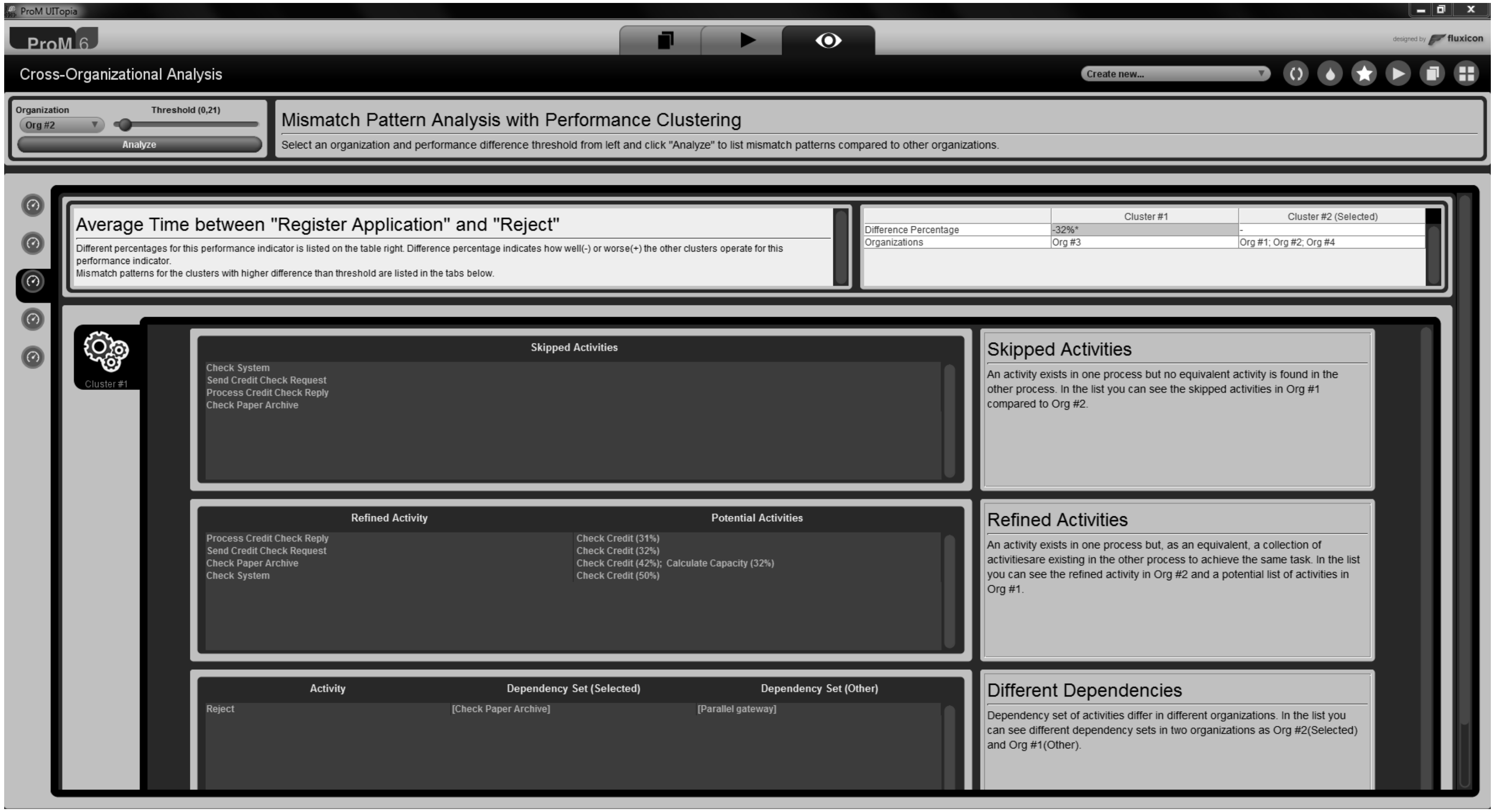
Different Moments in Processes

Set of activities are undertaken with different orders in different processes. In the list you can see different previous or next acitivities in two organizations as Org #1(Selected) and Org #2(Other).

Different Dependencies

Figure 4.9: Mismatch Patterns without Performance Clustering Screen of Cross-Organizational Process Miner Plugin

the selected organization's cluster, all performance indicators are listed as tabs in the main panel as visualized in Figure 4.10. For each performance indicator, difference percentages between clusters are tabulated and highlighted if there is any other cluster that performs better than the threshold. For the clusters performing better than threshold, mismatch patterns between the organizations of these clusters are listed in the second level tabs. This visualization aims to list mismatch patterns for an organization, compared to other clusters which perform better than the difference threshold.



Mismatch Pattern Analysis with Performance Clustering

Select an organization and performance difference threshold from left and click "Analyze" to list mismatch patterns compared to other organizations.

Average Time between "Register Application" and "Reject"

Different percentages for this performance indicator is listed on the table right. Difference percentage indicates how well(-) or worse(+) the other clusters operate for this performance indicator.
Mismatch patterns for the clusters with higher difference than threshold are listed in the tabs below.

	Cluster #1	Cluster #2 (Selected)
Difference Percentage	-32%*	-
Organizations	Org #3	Org #1; Org #2; Org #4

Skipped Activities
Check System
Send Credit Check Request
Process Credit Check Reply
Check Paper Archive

Skipped Activities

An activity exists in one process but no equivalent activity is found in the other process. In the list you can see the skipped activities in Org #1 compared to Org #2.

Refined Activity	Potential Activities
Process Credit Check Reply	Check Credit (31%)
Send Credit Check Request	Check Credit (32%)
Check Paper Archive	Check Credit (42%); Calculate Capacity (32%)
Check System	Check Credit (50%)

Refined Activities

An activity exists in one process but, as an equivalent, a collection of activitiesare existing in the other process to achieve the same task. In the list you can see the refined activity in Org #2 and a potential list of activities in Org #1.

Activity	Dependency Set (Selected)	Dependency Set (Other)
Reject	[Check Paper Archive]	[Parallel gateway]

Different Dependencies

Dependency set of activities differ in different organizations. In the list you can see different dependency sets in two organizations as Org #2(Selected) and Org #1(Other).

Figure 4.10: Mismatch Patterns with Performance Clustering Screen of Cross-Organizational Process Miner Plugin

CHAPTER 5

RESULTS AND DISCUSSIONS

In this chapter, methodology presented in this thesis study is applied on datasets and results are presented. Firstly, evaluation metrics are defined for each stage of methodology to assess the performance of approach. Following this, methodology is applied on two datasets and results are explained with discussions.

5.1 Evaluation Metrics

Approach in this study is an aggregation of various methods and they are significantly different from each other in their mathematical background. Therefore, instead of a global evaluation metric for the complete methodology, each stage will be evaluated within its evaluation metrics. Since these stages are executed sequentially, it is important for each stage to perform well enough to yield a successful outcome. In this section, evaluation metrics for each stage are presented and they will be used to determine the success of methodology on experiments.

5.1.1 Process Model Mining

Process model mining stage takes input as event logs of organizations and creates process models using process mining algorithms, which is *Inductive Miner* [26] in this thesis. Performance of this stage can be measured by the conformance of the process models to the event logs. Four competing quality criteria are defined in [33] as *fitness*, *simplicity*, *precision* and *generalization* for process model mining. Fitness is based on the idea of being able to replay event log over process model whereas precision focuses not underfitting the event log. Simplicity idea is based on the fact that the simplest model is the best model, on the other hand generalization supports not overfitting the event logs. These four competing criteria are mathematically defined in [29] and grouped into two dimensions for analysis purposes:

Fitness Event log and the process model should *fit* to each other, in other words process model should be able to parse the event log.

Appropriateness Since *fitness* does not provide information about the meaningfulness of process models, this dimension is defined. Two notions comprise this idea; *Structural Appropriateness* considers the simplicity whereas *Behavioral Appropriateness* analyzes balance between overfitting and underfitting.

In this thesis study, process model mining stage will be evaluated by the *Fitness* and *Appropriateness* of the mined process models for each organization. It is expected to have higher *fitness*, closer to 1, and a high level of appropriateness to continue to the next stages with the high quality process models. While evaluating, *fitness* will be more dominant since *appropriateness* without *fitness* results with irrelevant process models.

5.1.2 Performance Indicator Analysis

Performance indicator analysis stage consists of two parts as replaying event logs over process models and clustering organizations based on the performance indicators. In the replay phase, main operation is to *align* [35] event logs and process models. This alignment is the based on the idea of finding the optimal alignment where total cost of *move on process model* and *move on event log* are minimized. Since there is no baseline information for alignment costs of organizational logs, total cost information will be used together with the process model mining evaluation metrics. In other words, for the event logs and process models with known conformance metrics, total cost of alignments will create a secondary check if there are any problems related to replaying events over process models. It is expected to have less total cost of alignment for the organizations with higher conformance since it is easy to align event logs and processes with higher fitness values.

For the second stage of performance indicator analysis, there is a need to evaluate performance of clustering organizations. In this stage, organizations are clustered based on their performance indicator values that are calculated over replaying. Since there is no labeled data in the context of this thesis study, any cross-validation techniques could not be applied and thus internal evaluation metrics are exploited. As mentioned in the Section 4.3.2, within-SSE values for clusters are plotted to the end user to select an appropriate number of clusters. It is expected that within-SSE values decreases as number of clusters increases; however, number of clusters should be selected without causing overfitting.

5.1.3 Mismatch Pattern Analysis

Mismatch pattern analysis stage aims to find differences between the process models of different organizations. In this thesis study mismatch patterns presented in [13] are mathematically defined and analyzers are developed to locate these patterns. At the time of this study, it is known to be first to use mismatch patterns in a generic method, therefore evaluation is based on comparing with well-defined prior similarity metrics.

Structural similarity between process models is presented in [14] by the *graph-edit distance* notion. In graph theory, *graph-edit distance* is the minimum number of *graph-edit operations* necessary to get one graph from another. In process mining field, *graph-edit operations* are simply node addition, deletion or substitution. In the study [14], both *graph-edit distance* and *graph-edit similarity* definitions are provided with their mathematical background. In this study, mismatch pattern analysis stage is evaluated by comparing the number of mismatch patterns and the *graph-edit similarity* of process models. Without any performance indicator clustering, it is expected to have larger number of mismatch patterns when the similarity between process models is low. This will ensure the performance and suitability of using mismatch pattern analysis to spot differences of organizations' process models in the methodology.

5.1.4 Recommendation Generation

In the recommendation generation stage, set of mismatch patterns are presented to end user based on the selected organization and performance difference threshold. This stage aims to list whole mismatch patterns that can cause the other organizations perform better than a difference threshold. Idea of using performance threshold should be evaluated in this stage with its responsiveness. In other words, it is expected that this stage will help the end user to focus on the most important performance improvements for the organization analyzed. Therefore, different threshold values will be tried to check how many mismatch patterns are generated for organizations and how they could be used for focused analysis. In addition, quality and applicability of the recommendations should be analyzed and this requires a high level of specialization. Knowledge required to analyze recommendations should include know-how about process changes, domain knowledge about the field of organizations' activity and structural attributions of the organizations.

5.2 Dataset Selection

Cross-organizational mining aims to find cross-correlation of workflows and activities in different organizations and this yields the necessity of organizations that do the same main activity with comparable process flows. From the business environment point of view, this field needs alignment of the tasks from different organizations with different business needs, priorities and organizational structures and culture. Considering these characteristics, there are few dataset available in the literature that are well-structured, documented and valuable. In this thesis study, one synthetic and one real-life event log datasets are presented and used in the following sections to evaluate the performance of the proposed methodology.

Loan Application Process [6] This synthetically created dataset consists of event log variants of a simple loan application in a financial institute. In this application process, a customer fills a form and starts a request over website and it triggers the different approaches of variants in different organizations that ends with notifying the customer about the acceptance of application. This dataset includes artificial event logs of 4 variants where each variant includes different sets of approaches such as parallelism, choices and sequential tasks. These event logs are used to test different approaches of discovering a configurable process model from a collection of event logs [7].

Environmental Permit Application Process [11] This dataset originates from the "Configurable Services for Local Governments (CoSeLoG)" project [31] which investigates the similarities and dissimilarities between several processes of different municipalities in Netherlands. Dataset contains records of receiving phase for the building permit application process in 5 municipalities, which are comparable since activity labels in the different event logs refer to the same activities performed in five municipalities. This dataset is also mentioned in the literature as *"Processing applications for building and/or environmental permits (Wet Algemene Bepalingen omgevingsrecht (WABO) in Dutch)"*.

When the organizations that have similar processes are considered, municipalities are one of the prominent candidates. In Netherlands there are more than 400 municipalities and they offer between 400 and 500 different products and services with their own processes. Unlike corporations, municipalities have the advantage that they can seek for collaboration since they are not direct competitions [10] and this advantage makes them valuable for cross-organizational analysis. CoSeLoG research project aims to develop a business process management system within a shared Software-as-a-Service environment using the commonalities between the processes of municipalities [7]. In the scope of CoSeLoG research, five different processes of municipalities are analyzed and at the time of this thesis study only *Environmental Permit Application Process* dataset is publicly shared.

5.3 Loan Application Process

5.3.1 General Information

In this section, methodology proposed in this thesis study will be applied on the *Loan Application Process* dataset [6] and evaluation results will be presented. Statistical

Table 5.1: Statistical summary of Loan Application Process dataset

	Cases	**Events**	**Percentage**
Variant #1	100	590	24 %
Variant #2	70	420	17 %
Variant #3	200	800	33 %
Variant #4	105	630	26 %
Total	**475**	**2440**	

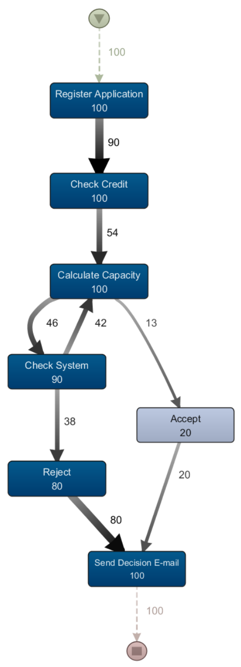
information about this dataset is presented in Table 5.1 to provide an insight about this dataset.

As shown in Table 5.1, total of 475 cases and 2440 events included in this dataset with a fairly even distribution between variants. In the following section, these variants will be used as organizational logs and the methodology presented in this thesis study will be applied.

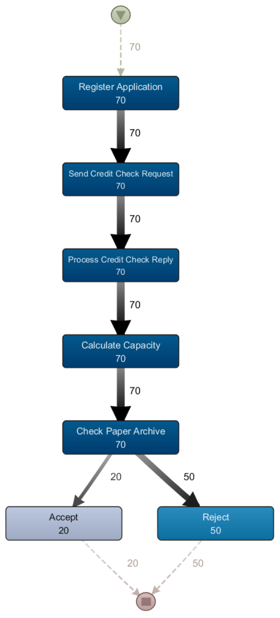
5.3.2 Methodology Stages

In *Process Model Mining* stage, variants of the dataset are used as organizational event logs and they are used to mine process models. Since dataset is synthetically generated, noise threshold in *Inductive Miner* is set to 0 and perfect log fitness is ensured. Appropriateness and fitness evaluation metrics are summarized in Table 5.2 and it can be seen that each event log is successful in terms of representing reality and being appropriate as a process model. Especially for the variants other than Variant #1, there is a perfect fitness and appropriateness as it is expected from a synthetically generated dataset without noise. In addition, process models for each event log is visualized in Figure 5.1.

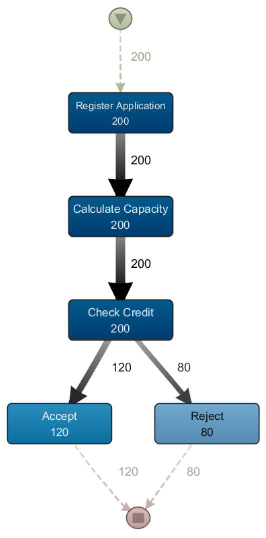
In *Performance Indicator Analysis* stage, firstly event logs are replayed over process models and performance indicators are calculated. For replaying step, alignment costs should be checked with respect to process model mining evaluation metrics; however, since all fitness values of variants are 100 % in Table 5.2, corresponding alignment costs are 0 for this dataset. This ensures that event logs are successfully



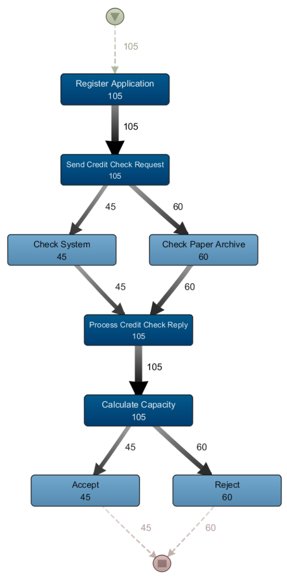
(a) Variant #1



(b) Variant #2



(c) Variant #3



(d) Variant #4

Figure 5.1: Process models of Loan Application Process dataset

Table 5.2: Process Model Mining Evaluation of Loan Application Process Dataset

	Fitness	Structural Appropriateness	Behavioral Appropriateness	Average Appropriateness
Variant #1	100 %	70 %	98.5 %	84.2 %
Variant #2	100 %	100 %	100 %	100 %
Variant #3	100 %	100 %	100 %	100 %
Variant #4	100 %	100 %	98.2 %	99.1 %
Average	**100 %**	**92.5 %**	**99.7 %**	**96.06 %**

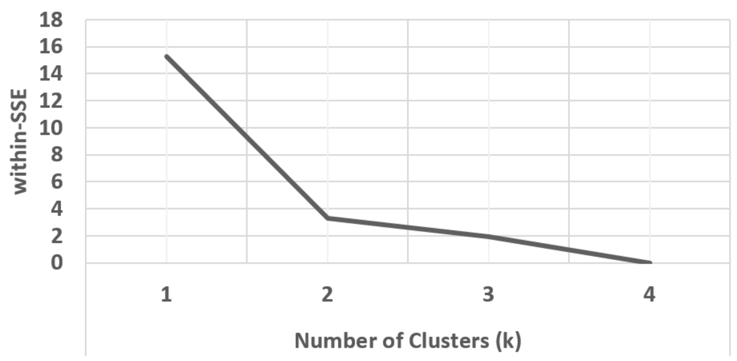


Figure 5.2: Number of Clusters vs. within-SSE for Loan Application Process dataset

replayed over process models and performance indicators are calculated. With these performance indicators, organizations are clustered and internal evaluation of clusters are presented with different number of clusters in Figure 5.2. As expected, within-SSE value decreases when number of clusters, *k*, increases. Considering the effects of overfitting after the *elbow* point in the plot, number of clusters is selected to be 2 for this dataset. For two clusters, Variant #1, #2, and #4 are grouped into one cluster where only Variant #3 is left to other cluster.

In *Mismatch Pattern Analysis* stage, number of mismatch patterns are analyzed with the *graph-edit similarity* between each two organization. For each two organization, their *graph-edit similarity* values are calculated and then our mismatch pattern analyzers are executed to spot differences. As the similarity between process models decreases our method spots more mismatch patterns for most of the variants and it ensures that the developed mismatch pattern analyzers work as expected for this dataset. Correlation between *graph-edit similarity* and number of mismatch patterns are plotted in Figure 5.3.

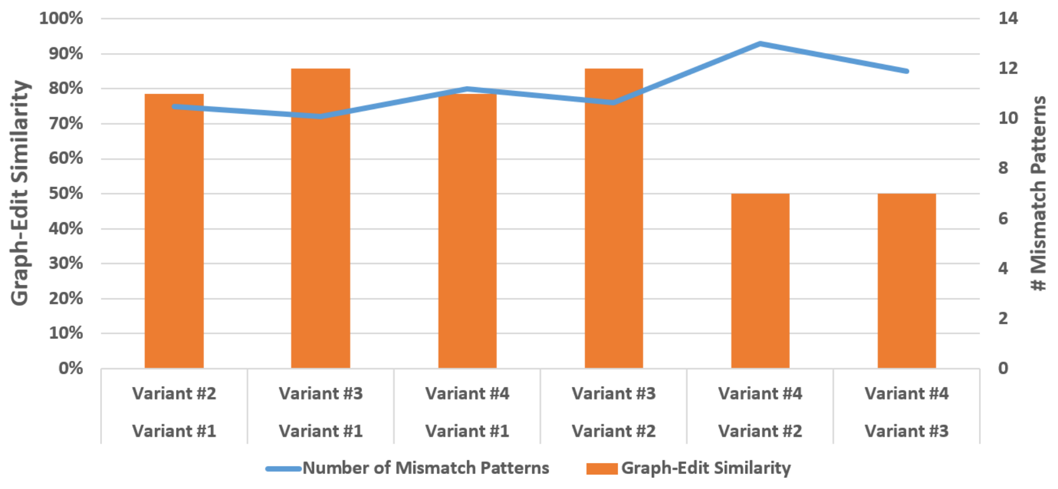


Figure 5.3: Mismatch Patterns vs. Graph-Edit Similarity for Loan Application Process variants

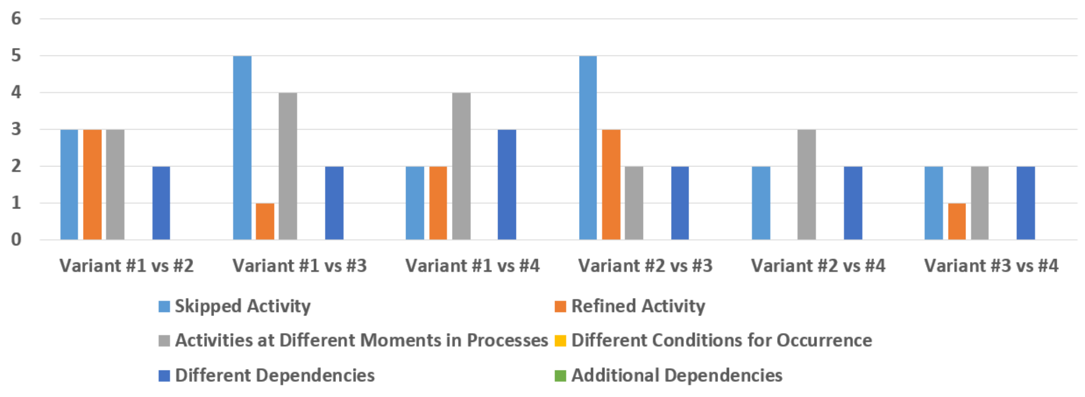


Figure 5.4: Mismatch pattern types for Loan Application Process variants

When the mismatch patterns are analyzed according to their types as diagrammed in Figure 5.4, *"Skipped Activity"* and *"Activities at Different Moments"* patterns are spotted mostly and no *"Different Conditions for Occurrence"* or *"Additional Dependencies"* patterns are discovered. Considering the small amount of this dataset, these numbers and distribution can be counted as acceptable in this stage of methodology.

In *Recommendation Generation* stage, an organization and performance difference threshold is selected as analysis input. For the selected organization's cluster, all other clusters are checked for performing better than the specified threshold. Instead of all mismatch patterns between organizations, only the mismatch patterns that are potential causes of other organizations to perform better are listed. For different threshold

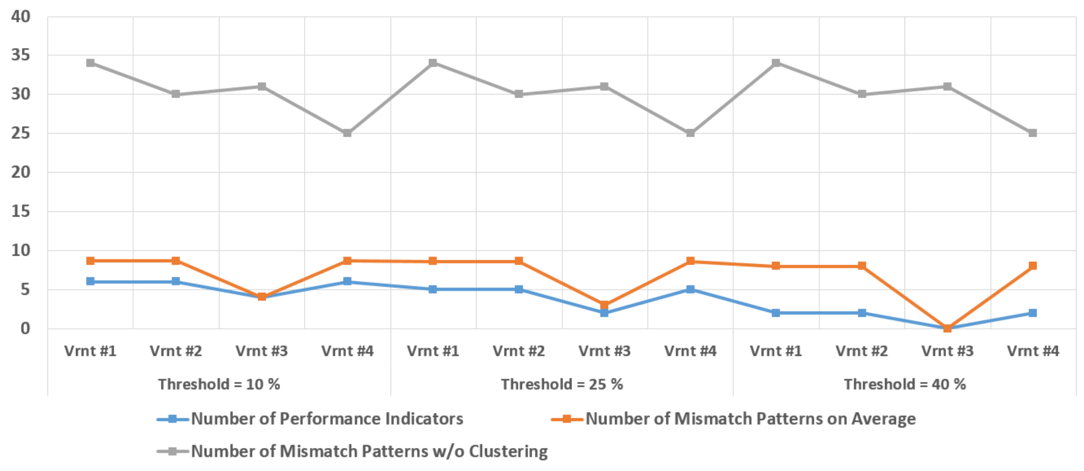


Figure 5.5: Recommendation Generation analysis for Loan Application Process dataset

values, number of performance indicators that are performing better for the selected organization and spotted mismatch patterns are plotted in Figure 5.5.

In order to construct the data in Figure 5.5, every organization is selected one-by-one with different threshold values. For every analysis, number of performance indicators and average number of mismatch patterns causing them are plotted. In addition, total number of mismatch patterns without clustering for each organization is added to the plot as an upper bound. With the help of this upper bound, responsiveness and degree of helping the user to focus on the performance improvement can be analyzed. As can be seen, for each threshold value, average number of mismatch patterns *with performance indicator clustering* are very low compared to *without clustering*. In other words, when user wants to improve its performance with any threshold, there is significantly less number of mismatch patterns on average to check. This shows the methodology proposed in this thesis can help users to focus on differences between organizations given this dataset.

Table 5.3: Statistical summary of Environmental Permit Application Process dataset

	Cases	**Events**	**Percentage**
Municipality #1	54	131	6.1 %
Municipality #2	302	586	27.3 %
Municipality #3	37	73	3.4 %
Municipality #4	340	507	23.7 %
Municipality #5	481	845	39.4 %
Total	**1214**	**2142**	

5.4 Environmental Permit Application Process

5.4.1 General Information

In this section, methodology proposed in this thesis study will be applied on the *Environmental Permit Application Process* dataset [11] and evaluation results will be presented with a similar previous approach. Since this dataset consists of real-life event logs, preprocessing is undertaken prior to start analysis. Incomplete traces, which started but not ended in the time frame of log collection, and exceptional cases are removed from event logs using ProM log visualization tools with a similar approach in [7]. Statistical information about the dataset that is used in this section can be summarized in Table 5.3 after preprocessing.

As shown in Table 5.3, total of 1214 cases and 2142 events included in this dataset with a variable distribution between event logs of municipalities. In the following section, these municipalities will be used as organizational logs and the methodology presented in this thesis study will be applied.

5.4.2 Methodology Stages

In *Process Model Mining* stage, event logs of each municipality in the dataset are used as organizational event logs and they are used to mine process models. Considering preprocessing is undertaken on the event logs, noise threshold in *Inductive Miner* is set to a low value of 10% to achieve a higher fitness.

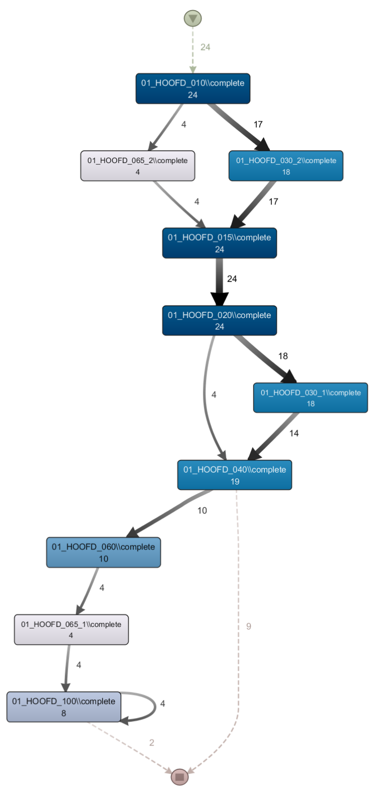
Table 5.4: Process Model Mining Evaluation of Environmental Permit Application Process dataset

	Fitness	**Structural Appropriateness**	**Behavioral Appropriateness**	**Average Appropriateness**
Mun. #1	86 %	97.5 %	54.4 %	76 %
Mun. #2	100 %	100 %	100 %	100 %
Mun. #3	92.3 %	71.1 %	67.2 %	69.1 %
Mun. #4	96.8 %	65.7 %	64 %	64.9 %
Mun. #5	94.5 %	58.8 %	39.7 %	49.3 %
Average	**93.9 %**	**78.6 %**	**65.1 %**	**71.9 %**

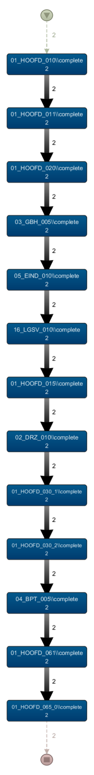
Appropriateness and fitness evaluation metrics are summarized in Table 5.4 and it can be seen that each event log is successful in terms of representing reality with high fitness values. However, some of the process models like Municipality #5 and #4 resulted with low appropriateness values. Process models for each event log are visualized with a detail simplification based on number of activities and paths in Figure 5.6. Detail simplification is only used for visualization and it draws the mainstream process flows instead of whole set of paths and activities. When process model diagrams are checked it can be seen that low appropriateness values resulted with complicated process models that are difficult to analyze visually. It should be kept in mind that actual process models are 10 to 20 times more complicated in terms of number of activities and paths than the ones presented in Figure 5.6.

In *Performance Indicator Analysis* stage, alignment costs are calculated over replay of event logs on process models. As presented in the Table 5.5, as appropriateness and fitness decrease; alignment costs increase for the municipalities. This is an expected behavior since decrease in fitness indicates that process model cannot parse event log well and low appropriateness means the process models are not structured in terms of quality metrics. These results indicates the fact that both process model mining stage and replay stage is in conformity and performance indicators calculated over replay are acceptable.

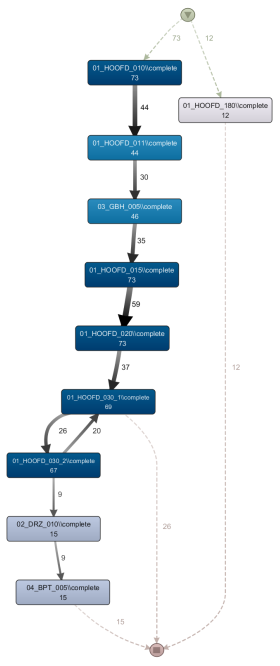
After calculating the performance indicators, municipalities are clustered based on these values and this stage is evaluated by the within-SSE values for different number of clusters as plotted in Figure 5.7. In order to avoid overfitting of clusters, number of clusters, *k*, is selected to be 2 for this dataset for further analysis. For two clusters,



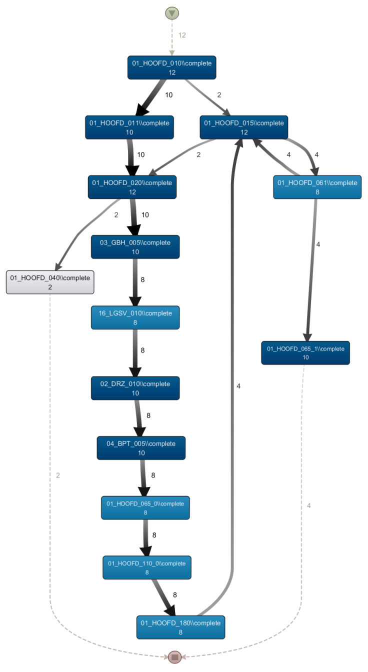
(a) Municipality #1



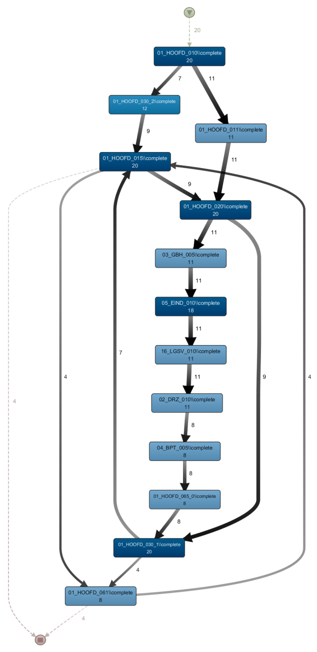
(b) Municipality #2



(c) Municipality #3



(d) Municipality #4



(e) Municipality #5

Figure 5.6: Process models of Environmental Permit Application Process Dataset

Table 5.5: Replay Evaluation of Environmental Permit Application Process Dataset

	Fitness	Average Appropriateness	Alignment Cost
Municipality #1	86 %	76 %	173.2
Municipality #2	100 %	100 %	0
Municipality #3	92,3 %	69,1 %	332.3
Municipality #4	96,8 %	64,9 %	9,1
Municipality #5	94,5 %	49,3 %	35.8

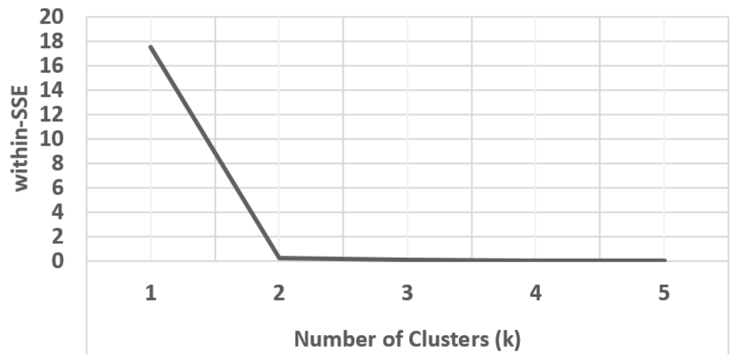


Figure 5.7: Number of Clusters vs. within-SSE for Environmental Permit Application Process dataset

Municipality #1 is located in one cluster where municipalities #2, #3, #4 and #5 are grouped into to other cluster.

In *Mismatch Pattern Analysis* stage, number of mismatch patterns are analyzed with the *graph-edit similarity* between each two municipality. In order to check correlation between *graph-edit similarity* and number of mismatch patterns, data is plotted in Figure 5.8. When the plot is checked, it can be seen that as the similarity between process models of municipalities increases, number of mismatch patterns decreases on most of the cases. When further analyzed, it can be seen that Municipalities #4 and #5, which have significantly more complex process model compared to others, fails in spotting mismatch patterns according to *graph-edit similarity*.

When the mismatch patterns are analyzed according to their types as diagrammed in Figure 5.9, *Skipped Activity* pattern is spotted most frequently. Following this, *Different Dependencies* and *Additional Dependencies* are spotted widely and there are on average of 78 mismatch patterns for process models of municipalities. Unfortunately, *Refined Activity* pattern analyzers could not be applied on this dataset since activity

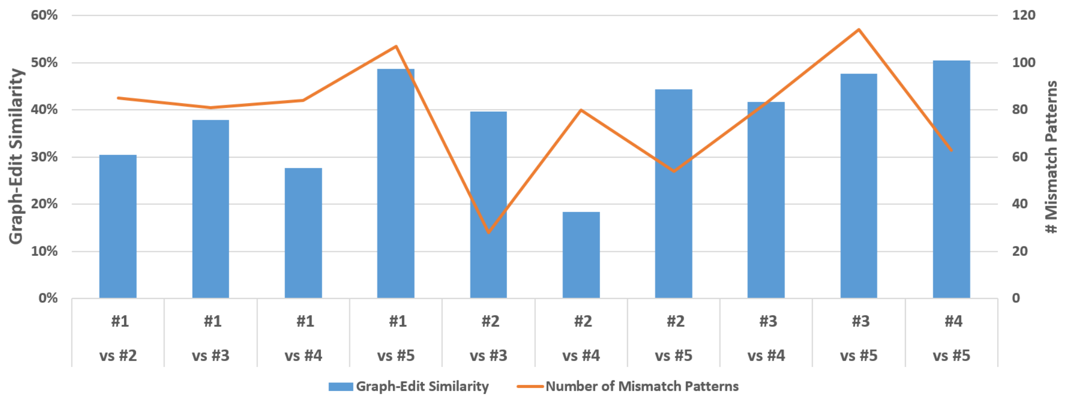


Figure 5.8: Mismatch Patterns vs. Graph-Edit Similarity for Environmental Permit Application Process Dataset

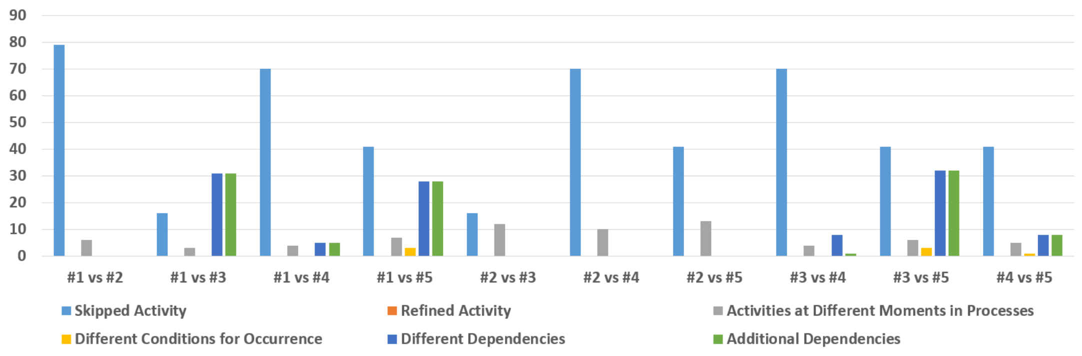
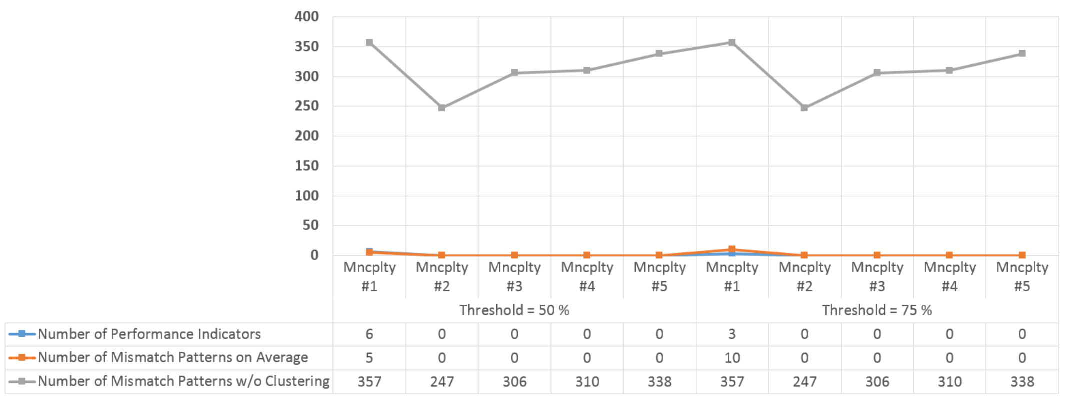


Figure 5.9: Mismatch pattern types for Environmental Permit Application Process dataset

names are not provided and activity codes without any semantic meaning are used instead. Considering the implementation of *Refined Activity* pattern in this study, which is based on the similarity between labels, they are eliminated in the analysis of this dataset.

In *Recommendation Generation* stage, an organization and performance difference threshold is selected as analysis input likewise the previous dataset. For different threshold values, number of performance indicators that are performing better for the selected organization and spotted mismatch patterns are plotted in Figure 5.10.

In the analysis of Figure 5.10, it can be seen that only the cluster that the Municipality #1 has the potential of learning from other cluster since it performs worse in all performance indicators. For the threshold of 50%, cluster of Municipality #1 performs

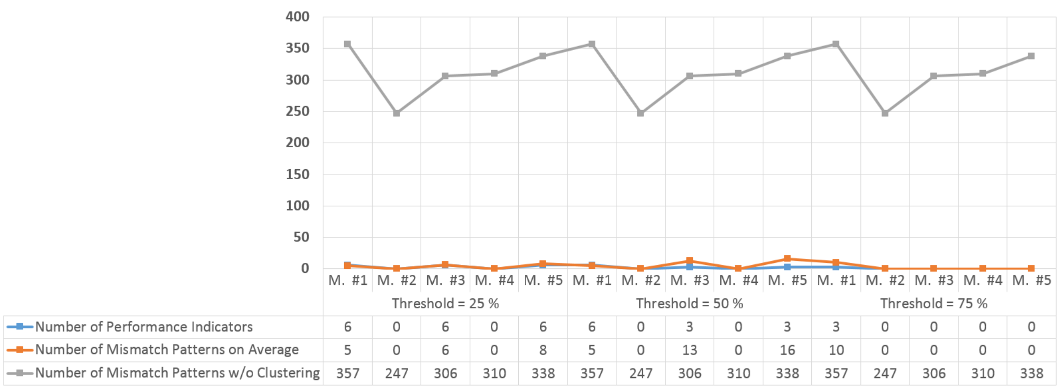


	Threshold = 50 %					Threshold = 75 %				
	Mncplty #1	Mncplty #2	Mncplty #3	Mncplty #4	Mncplty #5	Mncplty #1	Mncplty #2	Mncplty #3	Mncplty #4	Mncplty #5
Number of Performance Indicators	6	0	0	0	0	3	0	0	0	0
Number of Mismatch Patterns on Average	5	0	0	0	0	10	0	0	0	0
Number of Mismatch Patterns w/o Clustering	357	247	306	310	338	357	247	306	310	338

Figure 5.10: Recommendation Generation analysis for Environmental Permit Application Process dataset (2 Clusters)

worse in 6 performance indicators and proposed approach lists total of 30 mismatch patterns. On the average it show 5 patterns for each performance indicator to the user as the potential causes of performance improvement. With the same approach, cluster of Municipality #1 performs worse in 3 indicators with the difference of 75 % and on average 10 mismatch patterns are listed for each performance indicator. When it is compared to the total mismatch patterns of Municipality 1, which is 357, proposed approach helps significantly to the user for focusing performance improvement.

Since selecting number of clusters as 2 did not yield high learning potential for performance improvement, analysis is repeated with selecting number of clusters as 3. When three clusters are created, Municipality #1 is located in the first cluster; Municipality #2 and #4 are located in the second cluster; and Municipality #3 and #5 are grouped in to the last cluster. For these three clusters analysis is repeated for the thresholds of 25 %, 50 % and 75 % since these are the breaking points for performance indicator changes. As plotted in Figure 5.11, in addition to Municipality #1, now Municipality #3 and #5 have the learning potential from other clusters. However, Municipality #2 and #4 performs better in all performance indicators which yielded no mismatch pattern analysis data for them. Increasing cluster sizes in this analysis shows that organizations can learn more from each other but it has the potential danger of overfitting to other organizations process model.



	M. #1	M. #2	M. #3	M. #4	M. #5	M. #1	M. #2	M. #3	M. #4	M. #5	M. #1	M. #2	M. #3	M. #4	M. #5
	Threshold = 25 %					Threshold = 50 %					Threshold = 75 %				
Number of Performance Indicators	6	0	6	0	6	6	0	3	0	3	3	0	0	0	0
Number of Mismatch Patterns on Average	5	0	6	0	8	5	0	13	0	16	10	0	0	0	0
Number of Mismatch Patterns w/o Clustering	357	247	306	310	338	357	247	306	310	338	357	247	306	310	338

Figure 5.11: Recommendation Generation analysis for Environmental Permit Application Process dataset (3 Clusters)

5.5 Discussions

When the evaluation of the stages for *Loan Application Process* and *Environmental Permit Application Process* datasets are gathered together, the following results can be expressed:

- Process mining stage of the proposed methodology can mine the process models with perfect fitness and high appropriateness from event logs which have no noise. When the noise level increases in the dataset, fitness values of mined models decreases to 90 % levels with a decreasing appropriateness of models.

- For the successfully mined models with high fitness values, replay and performance indicator calculation stage works seamlessly as expected. With this step, average and standard deviation time between each activity can be measured for each organization. Number of these metrics are quadratic to the number of activities in each organization's process model and difficult to analyze with a cross comparison.

- Internal measure of clusters indicates that the organizations can be clustered according to their performance indicators which yields a collective approach of organizations for their subprocesses. In other words, within-SSE values decrease significantly when the organizations are divided into clusters which shows that they can be grouped based on how well they are executing in their performance indicators.

- Mismatch analysis spots the differences between process models in coherence with structural similarity of them. This indicates that the idea of using mismatch patterns to reveal differences between process models is a feasible approach since its results are comparable to the similarity metrics of process models in the literature. However, only the total number of mismatch patterns are taken into consideration in this study where their importance and occurrence is variable. For instance, it can be a very useful information for a *Skipped Activity* in a small dataset like *Loan Application Example*; however it is very likely to see huge number of *Skipped Activity* patterns in an immense dataset like *Environmental Permit Application Process*. With this consideration, distributions of mismatch patterns are presented for each dataset to reveal any tendencies for occurrence.

- Recommendation generation aims to gather all generated information in this thesis study to help focusing on the potentially important mismatch patterns for performance improvement. For this aim, different thresholds and different cluster sizes are analyzed to check responsiveness of this stage. When the number of mismatch patterns with and without performance clusterings are checked, it shows that in a small dataset, where even the mismatch patterns can be spotted by visual analysis, performance clustering lists 3 times less number of differences in *Loan Application Example* dataset. When it is impossible to locate mismatch patterns manually like in *Environmental Permit Application Process*, performance clustering spots 100 times less number of differences. This difference helps user to focus on the differences with a potential performance improvement which is one of the aims in this thesis study.

- Although each step of methodology can be counted as successful based on their evaluation metrics, mismatch patterns recommended at the end of methodology can yield important observations as well as being irrelevant and infeasible. Since this decision is based on the business environment of organizations, evaluation of the quality of recommendations for business usefulness requires domain expertise. However, some example recommendations can be presented to provide an insight:

  – In the analysis of *Loan Application Process*, for Variant #3 performance clustering results indicate that other cluster of variants perform 27 % better on average time and 12 % better on standard deviation time between activities "Calculate Capacity" and "Accept". In other words, cluster of other variants complete the path between these activities in a short amount time with less variance. When the mismatch patterns for these performance indicators are checked the following ones can be mentioned:

  * "Check Credit" is a *Skipped Activity* in the other cluster.
  * "Check Credit" is a *Refined Activity* of with "Check System (50 %)"; "Check Paper Archive (42 %)"; "Send Credit Check Request (32 %)"; "Process Credit Check Reply (31 %)" where the corresponding similarity values provided in parentheses.
  * "Calculate Capacity" is a *Different Moments in Processes* which have different previous activities in clusters.

  When these example mismatch patterns are checked, removing "Check Credit" activity and putting other activities instead of it might be the cause of performance improvement. With the same approach, putting "Calculate Capacity" on different orders in processes can effect the average and variance of time between activities. These mismatch patterns are also visualized on process model of Variant #3 and a variant from other cluster in Figure 5.12. In the process models, refined activities of "Check Credit" and different positions of "Calculate Capacity" are indicated.

- In the analysis of *Environmental Permit Application Process* with 3 clusters, Cluster #3 performs 40 % better on average time and 53 % better on standard deviation time between the activities "01_HOOFD_010" and "01_HOOFD_015". When the mismatch patterns between these clusters for the performance indicator is listed the following ones can be mentioned:

  * Activity "03_GBH_005" and "16_LGSV_010" are *Different Moments in Processes* which have different next activities in clusters.
  * Activity "01_HOOFD_015" and "01_HOODF_065" are *Different Dependencies* patterns and they have a different set of dependencies in clusters.

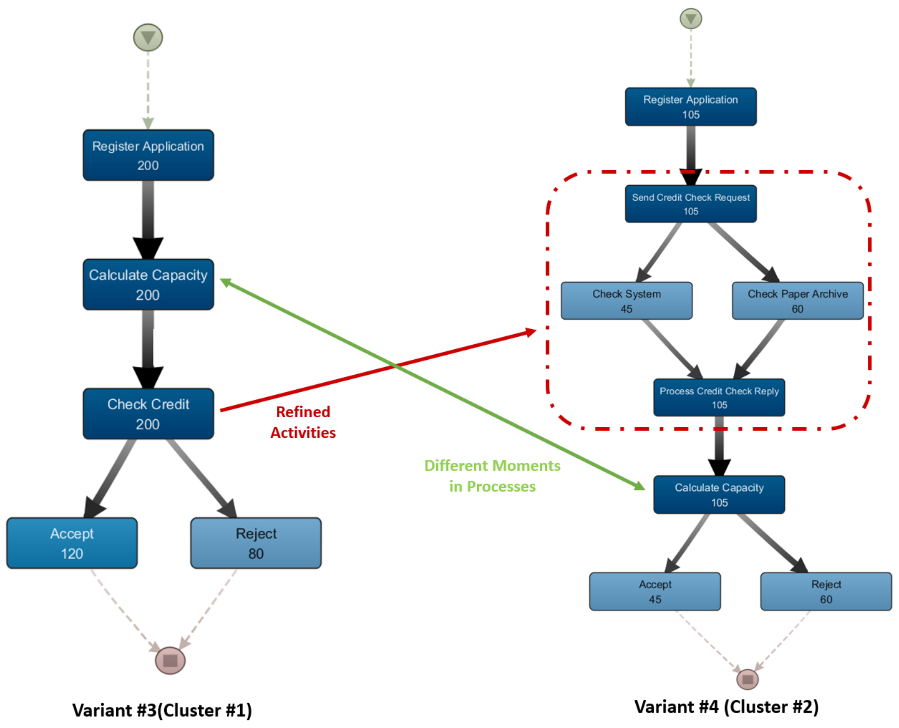


Figure 5.12: Visualization of example recommendation for Loan Application Process dataset

Although the activity codes in mismatch patterns do not reveal any information about their context, they are listed as potential cause of performance improvement. For the municipalities in Cluster #3, the mentioned mismatch patterns are visualized on the fragments of process models in Figure 5.13 and Figure 5.14 since it is difficult to visualize the complete process models. In the process models, each mismatch pattern is marked and this shows how the proposed approach helps to focus on differences between process models.

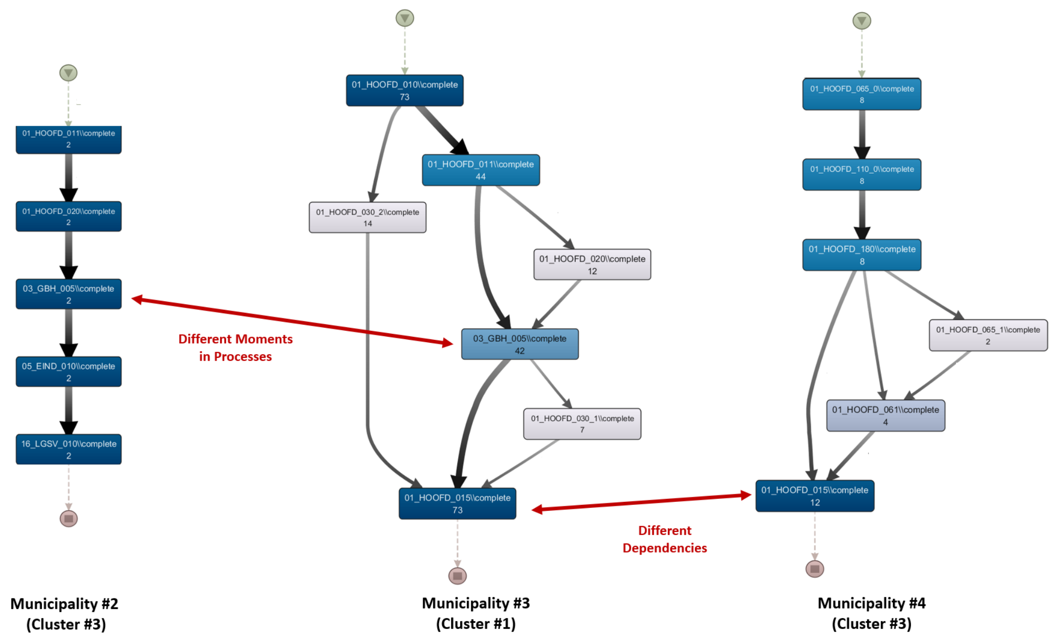

01_HOOFD_011\\complete 2
01_HOOFD_020\\complete 2
03_GBH_005\\complete 2
05_EIND_010\\complete 2
16_LGSV_010\\complete 2
Different Moments in Processes
01_HOOFD_010\\complete 73
01_HOOFD_011\\complete 44
01_HOOFD_030_2\\complete 14
01_HOOFD_020\\complete 12
03_GBH_005\\complete 42
01_HOOFD_030_1\\complete 7
01_HOOFD_015\\complete 73
Different Dependencies
01_HOOFD_065_0\\complete 8
01_HOOFD_110_0\\complete 8
01_HOOFD_180\\complete 8
01_HOOFD_065_1\\complete 2
01_HOOFD_061\\complete 4
01_HOOFD_015\\complete 12
Municipality #2 (Cluster #3)
Municipality #3 (Cluster #1)
Municipality #4 (Cluster #3)

Figure 5.13: Visualization of example recommendation for Environmental Permit Application Process dataset (Municipality #3)

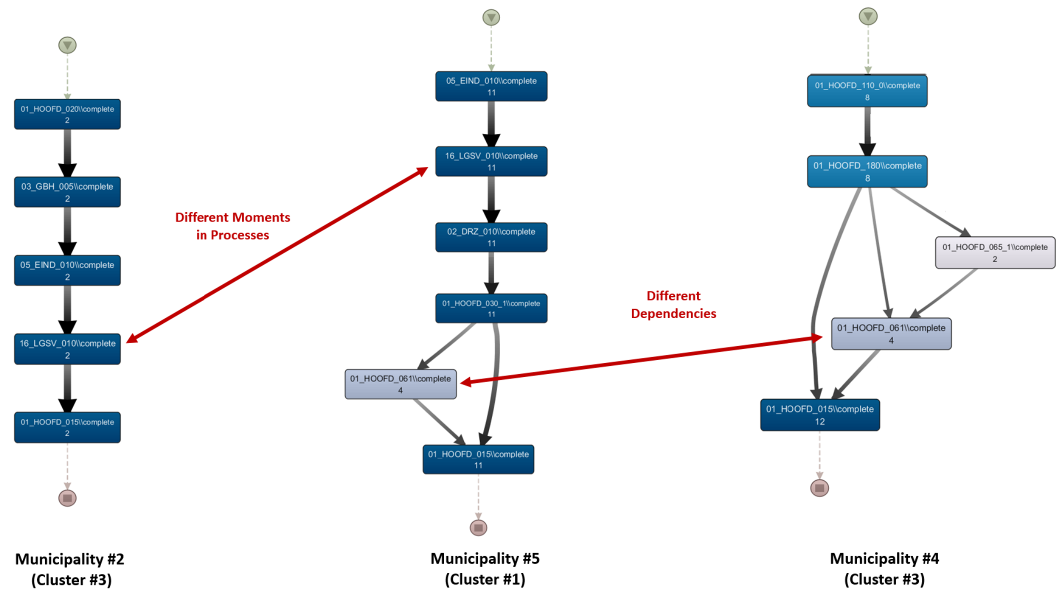


Figure 5.14: Visualization of example recommendation for Environmental Permit Application Process dataset (Municipality #5)

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this thesis study, a new approach is proposed and tested for generating recommendations using cross-organizational process mining for process performance improvement. Cross-organizational process mining is applied with the idea of unsupervised learning where predictor variables related to performances of organizations are used in an environment where processes are executed on several organizations. Results show that it is possible to use cross-organizational process mining and mismatch patterns for performance improvement recommendations. Process mining is a large-spectrum field where different set of activities and approaches are gathered together to discover, monitor and improve processes. In this thesis study, a four-stage solution is presented and their performances are explained.

Process mining stage mines the process models of different organizations and it is shown that a generic, noise-capable process mining method can create process models with high fitness and appropriateness values. This indicates that mining process models of different organizations under a generic method can yield comparable and appropriate process models. Success of this stage directly affects the quality of calculated performance indicators since they are collected through the replay of event logs over process models.

Performance indicator analysis stage in the methodology clusters the organizations based on their performance indicators and internal evaluation metrics show that it is suitable to cluster organizations based on how well they are operating. Although there are studies that clusters organizations based on their process models for structural analysis [19], this approach showed that organizations can be clustered based on their performance indicators.

Mismatch analysis stage in this thesis has the aim of spotting differences between processes of organizations and it is known to be the first implementation of mismatch patterns [13]. When the results of this stage is checked against well-established similarity metrics in the literature, it can be concluded that mismatch pattern finding can be used when there is a need for spotting differences in similar processes.

Recommendation generation stage collects the generated and extracted information in all prior stages to list what the organizations can learn from other organizations which perform better. In order to define performing better, different thresholds are checked for each organization and resulting recommendations show that clustering organizations based on performance indicators and then checking mismatch patterns significantly helps user to focus on the differences with a potential performance improvement. In addition, quality of recommendations in business usefulness is tried to be explained with example outputs and these examples show that the approach yields recommendations difficult to spot manually and visually, which are also potential causes of performance improvement.

In addition, proposed methodology is developed as extensible and configurable set of plugins in ProM framework [42] and published as open-source. This makes the methodology open to include new process mining methods, mismatch patterns and clustering approaches as well as testing with different datasets.

6.2 Future Work

For the approach proposed in this thesis study, the following issues can be listed as pointers to future work:

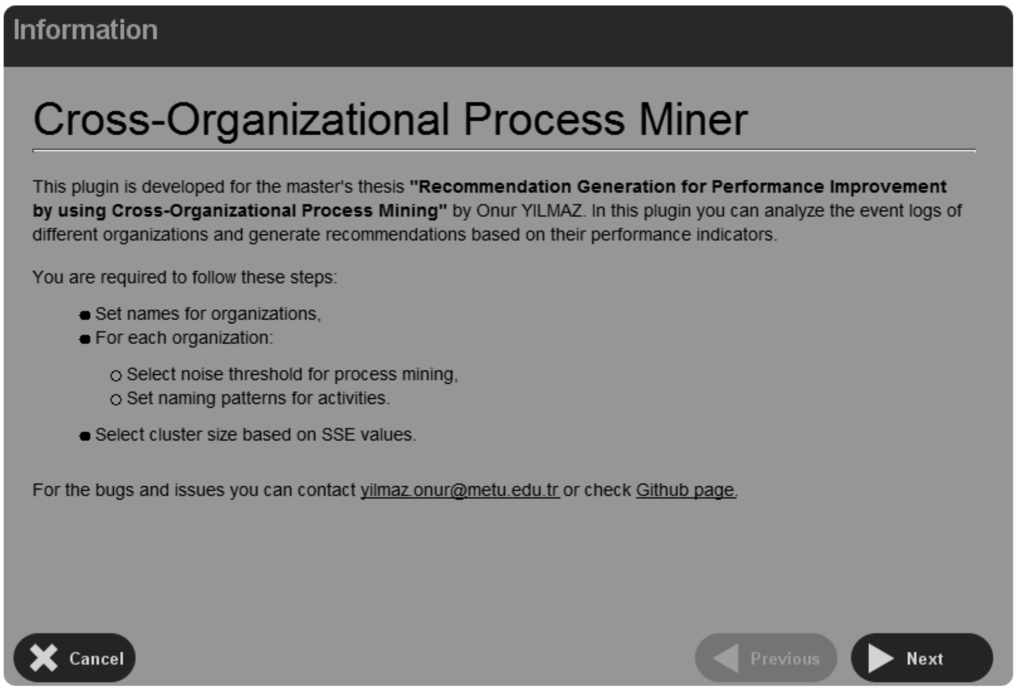
- In the process mining stage, instead of *Inductive Miner*, different techniques can be used which can mine complex process models with higher appropriateness levels while keeping the current high fitness values.

- In the performance indicator analysis stage, new indicators can be defined based on the business environment, event log attributes and user needs. For instance, personnel and resource allocation indicators can be included as well as cost dimension.
- For mismatch pattern analysis, new and business oriented mismatch patterns can be included in the analysis. In addition analyzers can fail when there are loops in the process models in current implementations, therefore more robust implementations for process models with loops can be developed in the future.
- For the generated recommendation, their quality for business environment are not assessed within the scope of this thesis. However, when any feedback from a domain expert or BPM people is provided, the learning approach can be converted to semi-supervised learning from unsupervised learning.
- For ProM implementation of this study, currently user selects an organization to list recommendations; in the future user might be able to select area of interest as well as starting and ending points visually on a process model.

APPENDIX A

APPENDIX

A.1 Methodology Steps


Information

Cross-Organizational Process Miner

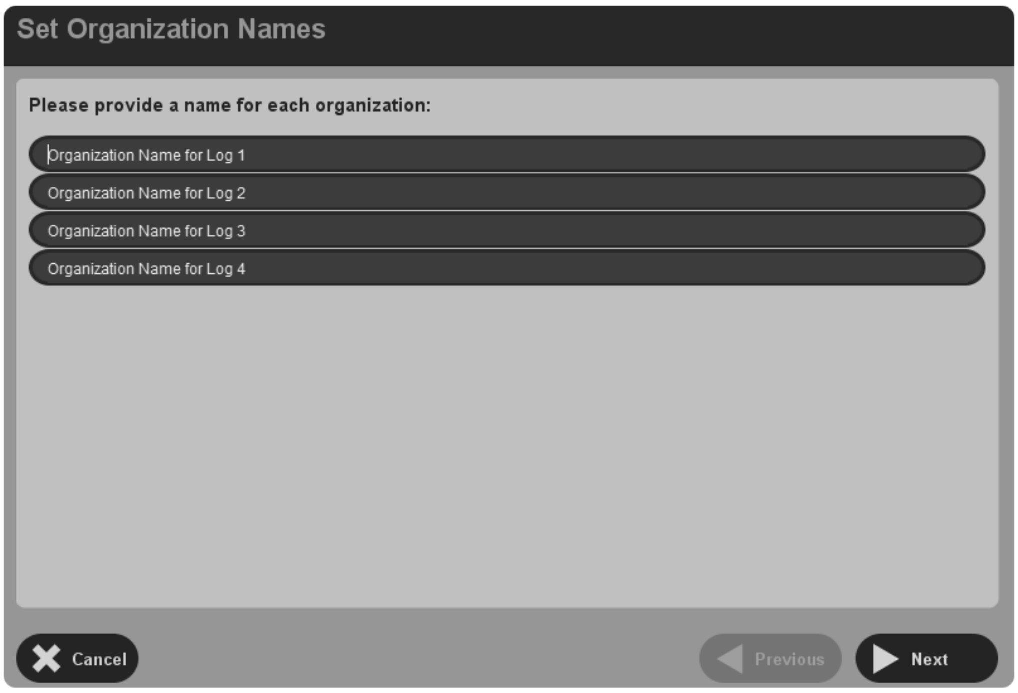
This plugin is developed for the master's thesis **"Recommendation Generation for Performance Improvement by using Cross-Organizational Process Mining"** by Onur YILMAZ. In this plugin you can analyze the event logs of different organizations and generate recommendations based on their performance indicators.

You are required to follow these steps:

- Set names for organizations,
- For each organization:
  - Select noise threshold for process mining,
  - Set naming patterns for activities.
- Select cluster size based on SSE values.

For the bugs and issues you can contact yilmaz.onur@metu.edu.tr or check Github page.

Figure A.1: Information Screen of Cross-Organizational Process Miner Plugin


Set Organization Names

Please provide a name for each organization:

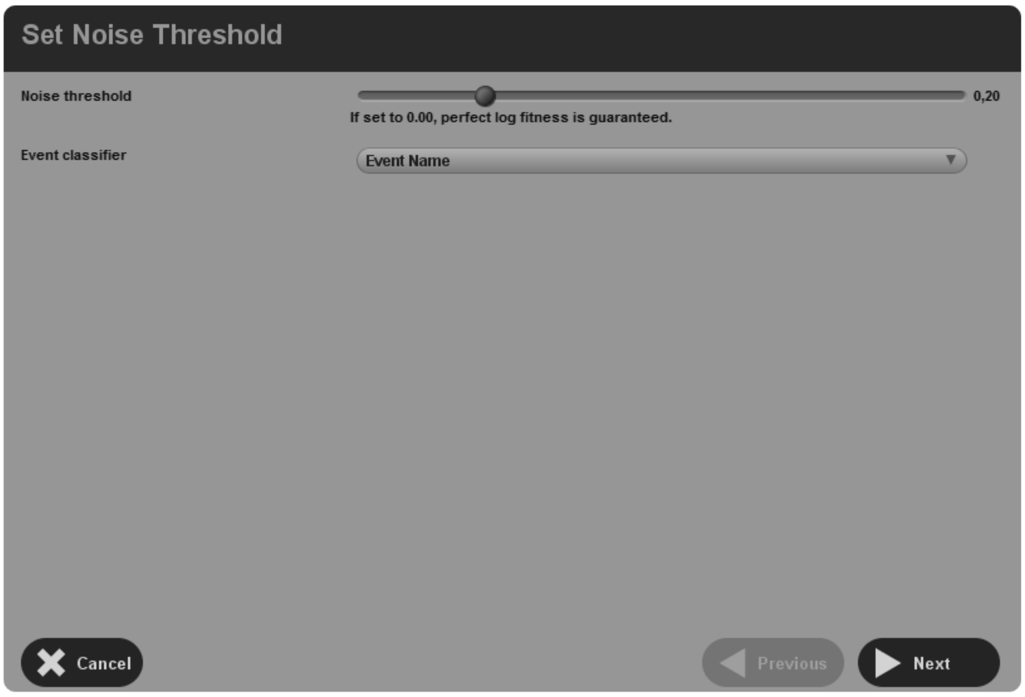
Organization Name for Log 1

Organization Name for Log 2

Organization Name for Log 3

Organization Name for Log 4

Figure A.2: Organization Naming Screen of Cross-Organizational Process Miner Plugin

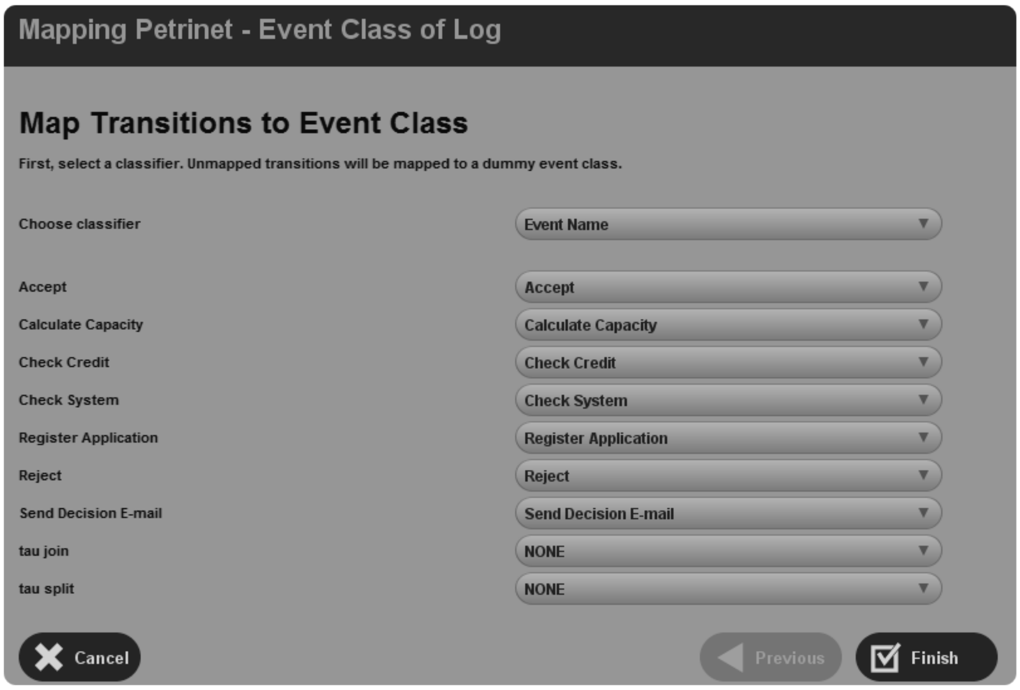

Set Noise Threshold

Noise threshold 0,20

If set to 0.00, perfect log fitness is guaranteed.

Event classifier: Event Name

Figure A.3: Process Miner Screen of Cross-Organizational Process Miner Plugin



Mapping Petrinet - Event Class of Log

Map Transitions to Event Class

First, select a classifier. Unmapped transitions will be mapped to a dummy event class.

Choose classifier	Event Name
Accept	Accept
Calculate Capacity	Calculate Capacity
Check Credit	Check Credit
Check System	Check System
Register Application	Register Application
Reject	Reject
Send Decision E-mail	Send Decision E-mail
tau join	NONE
tau split	NONE

Figure A.4: Automated Replayer Screen of Cross-Organizational Process Miner Plugin