ICIC Express Letters
Volume **19**, Number **3**, March **2025**

 ISSN 1881-803X
pp. **291–299**

TRANSFORMING EVENT LOG WITH DOMAIN KNOWLEDGE: A CASE STUDY IN PORT CONTAINER TERMINALS

Yutika Amelia Effendi[1,2] and Minsoo Kim[1,*]

[1]Department of Industrial and Data Engineering
College of Engineering
Pukyong National University
45 Yongso-ro, Namgu, Busan 48513, Korea
*Corresponding author: minsky@pknu.ac.kr

[2]Robotics and Artificial Intelligence Engineering
Faculty of Advanced Technology and Multidiscipline
Airlangga University
Kampus C UNAIR, Jl. Dr. Ir. H. Soekarno, Mulyorejo, Surabaya, Jawa Timur 60115, Indonesia
yutika.effendi@ftmm.unair.ac.id

Received May 2024; accepted July 2024

Abstract. *Process mining involves a range of techniques for analyzing the actual execution of organizational processes, with event data serving as the foundational component for analysis. While most process mining methods assume event logs as input, event data is typically not originally formatted in this way, requiring extraction from raw or traditional database sources. Consequently, preprocessing event logs from various data formats or databases becomes an important task. Despite its significance, many research endeavors have overlooked this aspect, concentrating on other process mining tasks such as discovery, conformance, or enhancement. The preprocessing of event logs encompasses two main phases: transforming the database into an event log and filtering event log activities from the messages. This study specifically focuses on transforming CSV-formatted data from the Terminal Operating System (TOS) database of a port container terminal into an event log for process mining that captures the goods importation processes. The goal is to obtain essential event log contents, including case ID, activity, time, originator, input/output, cost, and detailed attachments. Experimental results demonstrate that our approach successfully transforms the company's database into a standardized event log suitable for process mining.*

Keywords: Company database, Data preprocessing, Event log, Port container terminal, Process mining

1. **Introduction.** Each company operates with its own tailored information systems and databases, which can vary significantly depending on factors like industry, size, and organizational needs [1,2]. Regardless of these variations, the overarching aim is to establish a strong and cohesive Information Technology (IT) infrastructure that aligns with the company's business goals and operations. Process mining plays a vital role in supporting daily operations by enabling companies to optimize processes, improve efficiency, ensure compliance, and drive continuous improvement [3]. It offers actionable insights derived from real-world process data, empowering organizations to make informed decisions and maintain competitiveness in a dynamic business landscape [4].

Sometimes, there is a discrepancy between the data stored by a company and that utilized in process mining techniques, particularly in their format. Companies typically store data in various databases like relational databases (RDBMS), NoSQL databases, and

DOI: 10.24507/icicel.19.03.291

others [5-7], selecting databases based on specific application needs and business requirements. Some companies even employ multiple database types within their architecture to fulfill diverse system demands [8,9]. In contrast, process mining relies on standardized event logs for activities such as model discovery, conformance checking, and process enhancement [10], highlighting the importance of data preparation in the early stages of process mining [11]. In process mining, an event log is crucial, serving as a structured record of events tied to business process execution [12]. It forms the basis for process mining tools to analyze process dynamics, efficiency, and deviations, capturing data like time, activities, and resources as per company needs [13]. While various event log models exist, the single time event log is widely used [14]. With a few adjustments, it can be transformed into a dual time event log, featuring start and end times of activities, along with waiting times between successive activities [15,16].

This study focuses on a port container terminal, where daily inspections of numerous containers are meticulously recorded in the Terminal Operating System (TOS) database. Despite comprehensively storing all company activities, the database's format hinders direct use for business process management analysis, such as deriving process models or optimizing time and cost. Therefore, the study aims to automate data preprocessing by converting the database into an event log conforming to process mining standards. This conversion, a crucial process in process mining, allows for the seamless transition of comprehensive business process data into an event log. Consequently, relevant divisions can conduct research and analysis, enhancing business process management through process mining techniques.

The rest of this paper is organized as follows. Chapter 2 describes related researches. Chapter 3 elaborates on the research methodology, followed by implementation details discussed in Chapter 4. The paper concludes with a summary in Chapter 5.

2. **Literature Review.**

In process mining, event logs capture process details such as tasks performed, their sequences, and additional contextual information [10,17]. While traditionally featuring a single activity time, contemporary event logs now include both single and dual times, covering the start and end times of each activity [16,18].

Researchers in process mining have extensively explored event log preprocessing. [19] examined event log extraction and preprocessing, presenting foundational concepts and addressing design considerations in extracting event logs from raw event tables. Similarly, [20] conducted a comprehensive literature review, surveying relevant approaches to event data preprocessing for business process mining tasks, aiming to classify techniques and identify significant challenges.

In event log processing for process mining, various techniques aim to extract timestamps for each activity from a database and convert them into event logs. These methods may involve representing timestamps as single time or using the inverse of the normal cumulative distribution to derive activity times from grouped events, as discussed by [21]. Additionally, [15] suggested converting a single time event log into a dual time event log by consulting company experts or incorporating sojourn time. Furthermore, [16] presented a methodology using the PERT-Beta distribution to define dual time event logs, particularly useful when the organization's database provides only single time for grouped activities.

The research gap identified in literature review concerns the preprocessing of event logs, particularly in converting traditional databases into structured event logs for process mining. While existing literature explores various methodologies for preprocessing event logs, such as transforming timestamps and applying statistical distributions, there is a lack of studies specifically addressing direct preprocessing from traditional databases. This gap offers opportunities for further research into developing efficient methodologies or tools for converting traditional database formats into structured event logs. Therefore, this

study focuses on automating the transformation processes to handle traditional databases, ensuring alignment with process mining standards without extensive manual intervention. Such advancements would facilitate easier and more accurate analysis of business processes based on legacy data systems, which are common in many organizations.

3. **Methodology.** This research introduces a systematic approach to converting a company's database into the standard format of an event log for process mining. The study involves several stages, including dataset collection and the definition of algorithms for the transformation and filtering processes, as illustrated in Figure 1.

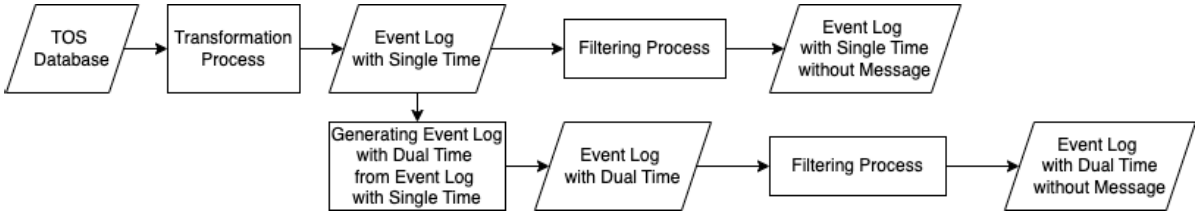

FIGURE 1. Proposed methodology

3.1. **Experimental setup.** Experiments were conducted using actual data sourced from the Terminal Operating System (TOS) database of the port container terminal. This database acts as an information system, generating a CSV file containing records of all processes executed within the terminal. The data utilized for our investigation spanned from January to March 2020 and included 12 traces, totaling 157,965 cases. The TOS database comprises various attributes, such as Container Key (CONTAINER_KEY), Container Type (CTR_TYPE), Vessel Berthing Process (VESSEL_ATB), Discard Date (DISC_DATE), Stack Date (STACK_DATE), Quarantine Process (HAS_QUARANTINE_FLAG), Job Delivery Date (JOB_DEL_DATE), Truck In Date (TRUCK_IN_DATE), and Truck Out Date (TRUCK_OUT_DATE), as shown in Figure 2.

CONTAINER_KEY	CTR_TYPE	VESSEL_ATB	DISC_DATE	STACK_DATE	HAS_QUARANTINE_FLAG	JOB_DEL_DATE	TRUCK_IN_DATE	TRUCK_OUT_DATE
4458248	DRY	1/25/20 16:40	1/25/20 22:47	1/25/20 23:13	YES	3/29/20 12:11	3/29/20 21:47	3/29/20 22:37
4458254	DRY	1/25/20 16:40	1/25/20 22:36	1/25/20 22:57	YES	3/29/20 12:11	3/29/20 17:36	3/29/20 19:32
4466620	DRY	1/27/20 8:38	1/27/20 12:48	1/27/20 13:04	YES	1/29/20 14:47	1/29/20 21:02	1/29/20 21:50
4466621	DRY	1/27/20 8:38	1/27/20 12:43	1/27/20 13:05	YES	1/29/20 14:47	1/29/20 20:55	1/29/20 21:54
4458404	DRY	1/25/20 16:40	1/25/20 23:08	1/25/20 23:22	YES	3/29/20 12:12	3/29/20 21:25	3/29/20 22:24
4466622	DRY	1/27/20 8:38	1/27/20 12:33	1/27/20 12:49	YES	1/29/20 14:47	1/29/20 17:50	1/29/20 19:41
4458412	DRY	1/25/20 16:40	1/25/20 23:36	1/26/20 0:39	YES	3/29/20 12:12	3/29/20 17:41	3/29/20 19:32
4458542	DRY	1/25/20 16:40	1/26/20 0:55	1/26/20 1:56	YES	3/29/20 12:12	3/29/20 22:48	3/30/20 1:42
4466627	DRY	1/27/20 8:38	1/27/20 12:51	1/27/20 13:21	YES	1/29/20 14:47	1/29/20 18:13	1/29/20 19:38

FIGURE 2. Terminal Operating System (TOS) database

The TOS database stored activity groups, and we obtained comprehensive activity lists from experts at the port container terminal. To convert the original CSV format of the TOS database into an event log, we included the following attributes: 1) case ID, a unique identifier assigned to each process instance [10,22]; 2) activity, representing a specific task, action, or operation performed as part of a business process [10,17]; 3) time, providing temporal information associated with each event [13,17]; 4) originator, denoting the entity or participant that initiated each activity [12,23]; 5) input and output, indicating the names of documents that support each activity; 6) cost, encompassing information associated with the resource consumption, expenses, or effort involved in executing each activity [10,23]; and 7) detailed attachments, additional details documented in the database.

3.2. **Transformation and filtering processes.** To preprocess event logs, we analyze each column in the TOS database CSV files, detailing the transformation process outlined in Table 1. This involves defining algorithms to convert TOS database CSV files

into event logs for process mining and conducting a filtering process to differentiate between activities and messages before displaying only activities in the event log. Distinguishing between activities and messages is crucial, as they represent different aspects of recorded events within a business process: activities signify process steps, while messages offer communication or data exchange context. Additionally, the transformation process for detailed attachments, which enriches activity information, considers different criteria: if CTR_TYPE equals DRY, it signifies a dry container; RFR indicates a reefer container. Conversely, if neither DRY nor RFR applies to CTR_TYPE, it is categorized as Uncontainer. Furthermore, CUSTOM_BEHANDLE_COUNT equals 0 and falls under the Green Line classification, while a value of 1 for CUSTOM_BEHANDLE_COUNT categorizes it as part of the Red Line. Finally, if HAS_QUARANTINE is affirmative (YES), it is designated as Quarantine.

TABLE 1. Transformation process of TOS database to an event log

Event log attributes	**Transformation process**
CASE ID	Direct retrieval of data from the CONTAINER_KEY column
TIME	- Generating event log with single time: 1) Expert's opinion; 2) Inverse of the normal cumulative distribution - Converting event log with single time to dual time: 1) Predetermined execution time; 2) Sojourn time; 3) PERT-Beta distribution
ACTIVITY/MESSAGE	Domain knowledge of experts in port container terminals Notes: 1 TOS database column corresponds to more than 2 executed activities in the event log *Example:* **TOS Database**: Truck In Date; Truck Out Date → **List of Activities**: Truck in; Unplug Refeer Cable/ Prepare Tools; Dispatch WQ Delivery to CHE; Lift on Container Truck; Determine Container Type; Truck Go To Gate Out; Determining Refeer/ Determining Dry/ Determining Uncontainer; Check Container before Truck Out; Decide Task Before Lift Container; Truck Out
ORIGINATOR	Customer, SKP (Employee), Pejabat Bea Cukai (Customs Officer), Port Container Terminal
COST	National Container Loading and Unloading Service Rates
INPUT/OUTPUT	- Input: NPWP, SIUP, BC 2.0, LHP, SPPB - Output: BC 2.0, SPPB, CEIR
DETAILED ATTACHMENTS	- CTR_TYPE ➔ DRY, REEFER, UNCONTAINER - HAS_QUARANTINE_FLAG ➔ QUARANTINE - CUSTOM_BEHANDLE_COUNT ➔ GREEN LINE, RED LINE

The preprocessing is conducted using the Python programming language, with the assistance of pandas and pm4py libraries. Algorithm 1 delineates the essential steps for converting company's database files into the event log. The algorithm begins with reading the CSV file and concludes with filtering between activity and message in the event log. Line 1 of Algorithm 1 clarifies its input and output. Line 2 is dedicated to reading a CSV file and converting it into an event log. Two techniques, introduced by [15,21], are available for generating a single time. One method, proposed by [21], utilizes the inverse of the normal cumulative distribution, while the alternative, advocated by [15], incorporates the estimated execution duration provided by experts. Algorithm 1, Line 3, presents pseudocode for obtaining the single time using the inverse of the normal cumulative distribution. Alternatively, experts' opinions can directly add the execution duration time to each activity's time. Line 4 focuses on the filtering process to distinguish between activities and messages in the resulting event log. Identifying the contents of the originator columns in the event log is crucial for this separation. Empty originator columns indicate activity messages in the activity column, and vice versa for activities.

Line	**Algorithm 1: Pseudocode for transforming company's database into process mining's event log**
1	**Input**: Files in tos.csv format **Output**: Event log in tos.csv format

2	**# Function to read CSV file and transform into event log**
	def transformCSVtoEventLog():
	eventLog = [ ]
	csvData = readCSV()
	for row in csvData:
	event = {'CaseID': row['CaseID'], 'Originator': row['Originator'], 'Input': row['Input'], 'Activity': row['Activity'], 'Time': row['Time'], 'Output': row['Output'], 'Cost': row['Cost'], 'Detailed Attachments': row['Detailed Attachments']}
	eventLog.append(event)
	return eventLog
	def readCSV():
	with open(filePath, 'r') as csvFile:
	csvReader = CSVReader(csvFile)
	headers = next(csvReader)
	data = [dict(zip(headers, row)) for row in csvReader]
	return data

3	**# Function to obtain event log with single time using inverse normal cumulative distribution**
	def generate_single_time_event_log(grouped_activities, mean_duration, std_dev_duration):
	event_log = [ ]
	for activity_group in grouped_activities:
	z_score = norm.ppf()
	single_time = np.random.normal(loc=mean_duration, scale=std_dev_duration) ∗ z_score
	for activity in activity_group:
	event_log.append({'activity': activity, 'time': single_time})
	return event_log

4	**# Function for filtering between activity and message in the event log**
	activity_or_message = data_in_column_activity_of_Log
	for x in activity_or_message:
	if originator(x) != NULL:
	Log_message.add(message)
	Log_data.erase(message)

We have the option to either conclude at this stage or proceed seamlessly with the transformation process, accommodating changes in the event log timeframe, referred to as the event log with dual time. This can be accomplished by following the procedures outlined in [15,16], allowing us to convert the existing event log from single time to dual time. To execute these steps formally, Algorithm 1 needs additional procedures incorporated into it. The supplementary pseudocode for generating an event log with dual time is provided in Algorithm 2, where Line 1 explains a function to convert single time to dual time using sojourn time, while Line 2 demonstrates the conversion using the PERT-Beta distribution.

Line	Algorithm 2: Additional pseudocode for generating event log with dual time using steps in [21,22]
1	**# Function to convert single time to dual time using sojourn time**
	def convert_to_dual_time(single_time, sojourn_time):
	end_time = single_time + sojourn_time
	return single_time, end_time
2	**# Function to convert single time to dual time using PERT-Beta distribution**
	def convert_to_dual_time(single_time, mode, optimistic, pessimistic):
	alpha = (4 ∗ mode – optimistic – pessimistic)/(pessimistic – optimistic)
	beta_val = alpha ∗ (1/alpha – 1)
	samples = np.random.beta(alpha, beta_val, size = 1)
	end_time = single_time + samples ∗ (pessimistic – optimistic) + optimistic
	return single_time, end_time[0]

4. **Results and Discussion.** Algorithm 1 was applied to the dataset extracted from the TOS database, spanning from January to March 2020. Managing a dataset of 157,965 entries, our methodology efficiently executed the transformation and filtering processes. A snapshot of the results is depicted in Figure 3(a), providing a standardized event log suitable for process mining, referred to as the event log with single time.

Furthermore, Algorithm 2's implementation yields an alternative output for the event log, incorporating both start and end times for each activity to provide detailed duration

Case ID	Originator	Input	Activity	Output	Time	Cost	Detailed Attachments
4691694	Customer	NPWP, SIUP	Document_Entry_via_PDE	BC 2.0	24/03/20 19.45	0	Dry;Green Line
4691694	TPS		Vessel_Berthing_Process		24/03/20 22.00	48379,74	
4691694	TPS		Discharge_Container		24/03/20 22.49	690,9082	
4691694	TPS		Bring_Container_to_Yard		25/03/20 00.57	74,99	
4691694	TPS		Stack_Container_in_Yard		25/03/20 22.59	28,99	
4691694	SKP	BC 2.0	Verification_Document_Behandle	BC 2.0	27/03/20 08.39	8,604293	Dry;Green Line
4691694	Pejabat Bea Cukai	LHP, BC 2.0	Create_document_SPPB	SPPB	27/03/20 22.48	2055,586	
4691694	Customer	SPPB	Create_Job_Order_Document_Delivery	CEIR	28/03/20 14.45	20,41807	
4691694	Customer		Truck_in		28/03/20 18.30	8,74889	
4691694	TPS		Dispatch_WQ_Delivery_to_CHE		28/03/20 18.35	0,514016	
4691694	TPS		Determine_Container_Type		28/03/20 18.40	1,313631	
4691694	TPS		Determining_Dry		28/03/20 18.50	23,37	
4691694	TPS		Decide_Task_Before_Lift_Container		28/03/20 18.58	1,544251	
4691694	TPS		Lift_on_Container_Truck		28/03/20 19.07	23,97	
4691694	Customer		Truck_Go_To_Gate_Out		28/03/20 19.34	1,095533	
4691694	TPS		Check_Container_before_Truck_out		28/03/20 19.40	1,034407	
4691694	Customer		Truck_Out		28/03/20 19.45	6,305861	

(a)

Case ID	Originator	Input	Activity	Output	Start Time	End Time	Cost	Detailed Attachments
4691694	Customer	NPWP, SIUP	Document_Entry_via_PDE	BC 2.0	24/03/20 19.45	24/03/20 20.06	0	Dry;Green Line
4691694	TPS		Vessel_Berthing_Process		24/03/20 22.00	24/03/20 22.49	48379,74	
4691694	TPS		Discharge_Container		24/03/20 22.49	25/03/20 00.57	690,9082	
4691694	TPS		Bring_Container_to_Yard		25/03/20 00.57	25/03/20 02.59	74,99	
4691694	TPS		Stack_Container_in_Yard		25/03/20 22.59	26/03/20 04.16	28,99	
4691694	SKP	BC 2.0	Verification_Document_Behandle	BC 2.0	27/03/20 08.39	27/03/20 22.48	8,604293	Dry;Green Line
4691694	Pejabat Bea Cukai	LHP, BC 2.0	Create_document_SPPB	SPPB	27/03/20 22.48	27/03/20 23.00	2055,586	
4691694	Customer	SPPB	Create_Job_Order_Document_Delivery	CEIR	28/03/20 14.45	28/03/20 15.00	20,41807	
4691694	Customer		Truck_in		28/03/20 18.30	28/03/20 18.35	8,74889	
4691694	TPS		Dispatch_WQ_Delivery_to_CHE		28/03/20 18.35	28/03/20 18.40	0,514016	
4691694	TPS		Determine_Container_Type		28/03/20 18.40	28/03/20 18.50	1,313631	
4691694	TPS		Determining_Dry		28/03/20 18.50	28/03/20 18.58	23,37	
4691694	TPS		Decide_Task_Before_Lift_Container		28/03/20 18.58	28/03/20 19.07	1,544251	
4691694	TPS		Lift_on_Container_Truck		28/03/20 19.07	28/03/20 19.34	23,97	
4691694	Customer		Truck_Go_To_Gate_Out		28/03/20 19.34	28/03/20 19.40	1,095533	
4691694	TPS		Check_Container_before_Truck_out		28/03/20 19.40	28/03/20 19.45	1,034407	
4691694	Customer		Truck_Out		28/03/20 19.45	28/03/20 19.50	6,305861	

(b)

FIGURE 3. The event log results for March 2020: (a) Single time; (b) dual time

information. The choice between single and dual time usage depends on the intended process mining method; some algorithms utilize single time, while others require dual time. With this flexibility, companies exploring their business processes can seamlessly apply methods without data reformatting. The results of the event log with dual time are shown in Figure 3(b).

To assess the proposed preprocessing method's effectiveness, we evaluate it using two criteria: 1) splitting the dataset into training and testing sets, and 2) comparing the algorithm's prediction of the process model with the company's Standard Operating Procedure (SOP). Common process mining performance metrics like fitness, precision, generalization, and simplicity are utilized for this evaluation [10,24].

For dataset splitting, we divided it into 70% for training and 30% for testing. Figure 4 shows the discovered process model from the preprocessed event log, aligning with the company's SOP. The performance metrics indicate a fitness value of 0.844, precision of 0.830, generalization of 0.973, and simplicity of 0.878. A metric value of 1.000 is optimal. Hence, based on our evaluation, we confidently assert that our preprocessing method effectively transforms the company's database into an event log for process mining, enabling its utilization with other process mining techniques.

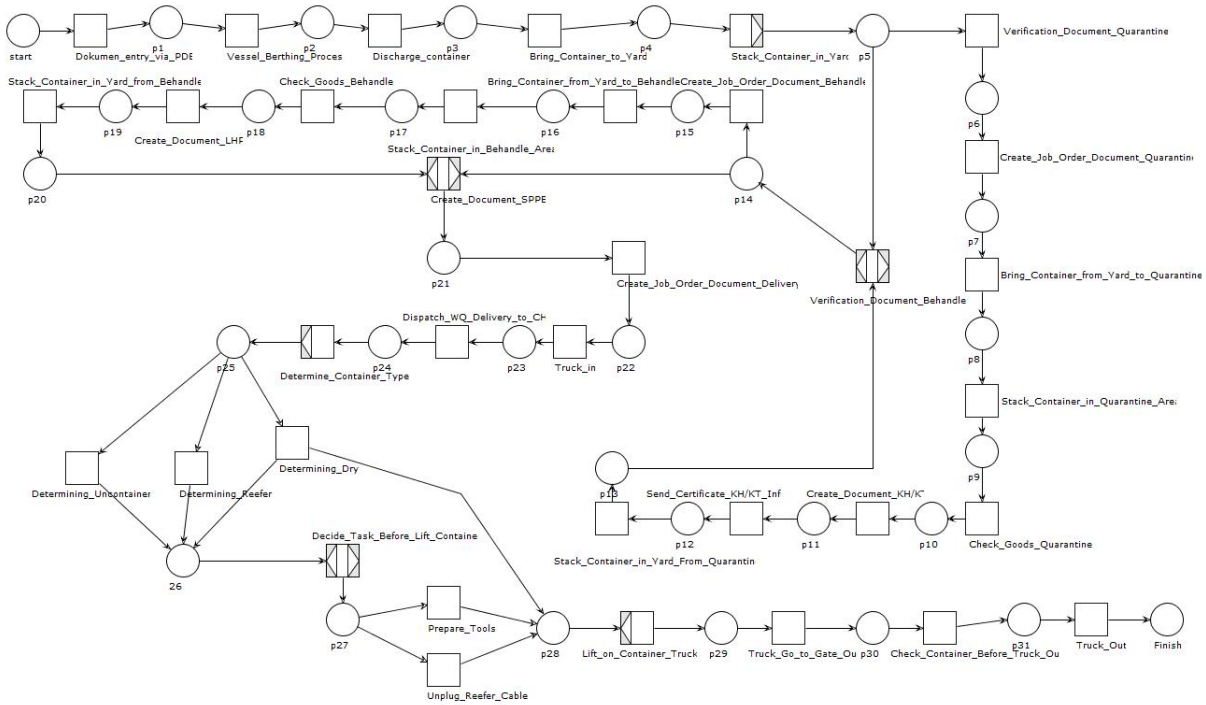

FIGURE 4. Process model discovered by preprocessed event log

5. **Conclusions.** The experimental results demonstrate the effectiveness of the proposed approach in converting the TOS database into a standardized event log for process mining. This initial preprocessing step paves the way for subsequent tasks such as business process modeling, conformance checking, and enhancement. Implemented at the Port Container Terminal, the transformation process automates the conversion of data from the company's database into an event log that adheres to process mining standards. This enables auditors, managers, and experts to optimize and evaluate business processes seamlessly while preserving the integrity of the original database. With validation at the Port Container Terminal, future research could explore the scalability of the transformation process across different terminals or operational databases within the same company. This would assess the adaptability and reliability of the approach in diverse contexts.

Acknowledgment. This work was supported by a Research Grant of Pukyong National University (2023). The authors also express their gratitude for the valuable comments and suggestions provided by the reviewers, which have enhanced the quality of the presentation.

REFERENCES

[1] A. P. Monteiro, J. Vale, E. Leite, M. Lis and J. Kurowska-Pysz, The impact of information systems and non-financial information on company success, *International Journal of Accounting Information Systems*, vol.45, 100557, DOI: 10.1016/j.accinf.2022.100557, 2022.

[2] T. Knauer, N. Nikiforow and S. Wagener, Determinants of information system quality and data quality in management accounting, *J. Manag. Control*, vol.31, pp.97-121, DOI: 10.1007/s00187-020-00296-y, 2020.

[3] I. Beerepoot, C. D. Ciccio, H. A. Reijers, S. Rinderle-Ma, W. Bandara, A. Burattin and F. Zerbato, The biggest business process management problems to solve before we die, *Computers in Industry*, vol.146, DOI: 10.1016/j.compind.2022.103837, 2023.

[4] P. Badakhshan, B. Wurm, T. Grisold, J. Geyer-Klingeberg, J. Mendling and J. vom Brocke, Creating business value with process mining, *The Journal of Strategic Information Systems*, vol.31, DOI: 10.1016/j.jsis.2022.101745, 2022.

[5] G. A. Schreiner, D. Duarte and R. Mello, When relational-based applications go to NoSQL databases: A survey, *Information*, vol.10, no.7, DOI: 10.3390/info10070241, 2019.

[6] F. Ravat, J. Song, O. Teste and C. Trojahn, Efficient querying of multidimensional RDF data with aggregates: Comparing NoSQL, RDF, and relational data stores, *International Journal of Information Management*, vol.54, DOI: 10.1016/j.ijinfomgt.2020.102089, 2020.

[7] T. Taipalus, Database management system performance comparisons: A systematic literature review, *Journal of Systems and Software*, vol.208, 111872, DOI: 10.1016/j.jss.2023.111872, 2024.

[8] J. M. Hellerstein, M. Stonebraker and J. Hamilton, Architecture of a database system, *Foundations and Trends in Databases*, vol.1, no.2, pp.141-259, DOI: 10.1561/1900000002, 2007.

[9] C. Chakroun, L. Bellatreche, Y. Aït-Ameur, N. Berkani and S. Jean, Be careful when designing semantic databases: Data and concepts redundancy, *IEEE 7th International Conference on Research Challenges in Information Science (RCIS)*, Paris, France, pp.1-12, DOI: 10.1109/RCIS.2013.6577692, 2013.

[10] W. M. P. van der Aalst, *Process Mining: Data Science in Action*, Springer, DOI: 10.1007/978-3-662-49851-4, 2016.

[11] A. Khannat, H. Sbai and L. Kjiri, Event logs pre-processing for configurable process discovery: Ontology-based approach, *2020 6th IEEE Congress on Information Science and Technology (CiSt)*, Agadir-Essaouira, Morocco, pp.139-144, DOI: 10.1109/CiSt49399.2021.9357303, 2020.

[12] W. Grossman and S. Rinderle-Ma, Process mining, in *Fundamentals of Business Intelligence*, Springer, 2015.

[13] W. M. P. van der Aalst, *Process Mining in Practice – Process Mining Book 3.0*, Fluxicon, 2022.

[14] R. Sarno, F. Haryadita, Kartini, Sarwosri and A. Solichah A, Business process optimization from single timestamp event log, *2015 International Conference on Computer, Control, Informatics and Its Applications (IC3INA)*, Bandung, Indonesia, pp.50-55, DOI: 10.1109/IC3INA.2015.7377745, 2015.

[15] Y. A. Effendi and R. Sarno, Modeling parallel business process using modified time-based alpha miner, *International Journal of Innovative Computing, Information and Control*, vol.14, no.5, pp.1565-1582, DOI: 10.24507/ijicic.14.05.1565, 2018.

[16] Y. A. Effendi and M. Kim, Refining process mining in port container terminals through clarification of activity boundaries with double-point timestamps, *ICIC Express Letters, Part B: Applications*, vol.15, no.1, pp.61-70, DOI: 10.24507/icicelb.15.01.61, 2024.

[17] G. Acampora, A. Vitiello, B. D. Stefano, W. M. P. van der Aalst, C. W. Gunther and H. M. W. Verbeek, IEEE 1849: The XES standard: The second IEEE standard sponsored by IEEE computational intelligence society, *IEEE Computational Intelligence Magazine*, vol.12, no.2, pp.4-8, DOI: 10.1109/MCI.2017.2670420, 2017.

[18] H. D. Oliveira, V. Augusto, B. Jouaneton, L. Lamarsalle, M. Prodel and X. Xie, Optimal process mining of timed event logs, *Information Sciences*, vol.528, pp.58-78, DOI: 10.1016/j.ins.2020.04.020, 2020.

[19] D. Fahland, Extracting and pre-processing event logs, *Data Structures and Algorithms*, arXiv: 2211.04338, DOI: 10.48550/arXiv.2211.04338, 2022.

[20] H. M. Marin-Castro and E. Tello-Leal, Event log preprocessing for process mining: A review, *Applied Sciences*, vol.11, no.22, DOI: 10.3390/app112210556, 2021.

[21] Y. A. Effendi and F. Retrialisca, Transforming timestamp of a CSV database to an event log, *2021 8th International Conference on Computer and Communication Engineering (ICCCE)*, Kuala Lumpur, Malaysia, pp.367-372, DOI: 10.1109/ICCCE50029.2021.9467226, 2021.

[22] K. Tajima, B. Du, Y. Narusue, S. Saito, Y. Iimura and H. Morikawa, Step-by-step case ID identification based on activity connection for cross-organizational process mining, *IEEE Access*, vol.11, pp.60578-60589, DOI: 10.1109/ACCESS.2023.3284053, 2023.

[23] K. Diba, K. Batoulis, M. Weidlich and M. Weske, Extraction, correlation, and abstraction of event data for process mining, *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol.10, no.3, DOI: 10.1002 /widm.1346, 2020.

[24] A. Rozinat and W. M. P. van der Aalst, Conformance checking of processes based on monitoring real behavior, *Information Systems*, vol.33, no.1, pp.64-95, DOI: 10.1016/j.is.2007.07.001, 2008.