JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 29 No. 6, pp. 101-112, June 2024
https://doi.org/10.9708/jksci.2024.29.06.101

EDF: An Interactive Tool for Event Log Generation for Enabling Process Mining in Small and Medium-sized Enterprises

Frans Prathama*, Seokrae Won*, Iq Reviessay Pulshashi*, Riska Asriana Sutrisnowati*

*Research Engineer, IOChord, Inc., Busan, Korea
*Director of R&D Center, IOChord, Inc., Busan, Korea
*Research Engineer, IOChord, Inc., Busan, Korea
*Research Engineer, IOChord, Inc., Busan, Korea

[Abstract]

In this paper, we present EDF (Event Data Factory), an interactive tool designed to assist event log generation for process mining. EDF integrates various data connectors to improve its capability to assist users in connecting to diverse data sources. Our tool employs low-code/no-code technology, along with graph-based visualization, to help non-expert users understand process flow and enhance the user experience. By utilizing metadata information, EDF allows users to efficiently generate an event log containing case, activity, and timestamp attributes. Through log quality metrics, our tool enables users to assess the generated event log quality. We implement EDF under a cloud-based architecture and run a performance evaluation. Our case study and results demonstrate the usability and applicability of EDF. Finally, an observational study confirms that EDF is easy to use and beneficial, expanding small and medium-sized enterprises' (SMEs) access to process mining applications.

▸ **Key words**: Event Log, Log Generation, Process Mining, Cloud Computing, Distributed Computing

[요 약]

본 논문에서는 프로세스 마이닝을 위한 이벤트 로그 생성을 지원하도록 설계된 대화형 도구인 EDF(Event Data Factory)를 소개한다. EDF는 다양한 데이터 커넥터를 통합하여 사용자가 다양한 데이터 소스에 연결할 수 있도록 지원한다. 이 도구는 그래프 기반 시각화와 함께 로우 코드/노코드 기술을 사용하여 비전문가 사용자가 프로세스 흐름을 이해하도록 돕고, 사용자 경험을 향상시킨다. EDF는 메타데이터 정보를 활용하여 사용자가 case, activity 및 timestamp 속성을 포함하는 이벤트 로그를 효율적으로 생성할 수 있도록 한다. 로그 품질 메트릭을 통해 사용자는 생성된 이벤트 로그의 품질을 평가할 수 있다. 우리는 클라우드 기반 아키텍처에서 EDF를 구현하고 성능 평가를 실행했으며, 본 연구와 결과는 EDF의 사용성과 적용 가능성을 보여주었다. 마지막으로 관찰 연구를 통해 EDF가 사용하기 쉽고 유용하여 프로세스 마이닝 애플리케이션에 대한 중소기업(SME)의 접근을 확장한다는 사실을 확인했다.

▸ **주제어**: 이벤트 로그, 로그 생성, 프로세스 마이닝, 클라우드 컴퓨팅, 분산 컴퓨팅

• First Author: Frans Prathama, Corresponding Author: Seokrae Won
*Frans Prathama (frans@iochord.com), IOChord, Inc.
*Seokrae Won (srwon@iochord.com), IOChord, Inc.
*Iq Reviessay Pulshashi (pulshashi@iochord.com), IOChord, Inc.
*Riska Asriana Sutrisnowati (riska@iochord.com), IOChord, Inc.
• Received: 2024. 04. 25, Revised: 2024. 05. 27, Accepted: 2024. 05. 28.

I. Introduction

In the rapid disruption of the business landscape, organizations must transform to remain competitive. Digital transformation (DT) drives organizations to consistently optimize operations, improve efficiency, and reduce costs. Through DT initiatives, organizations can drive business transformation and enhance their capabilities. A critical aspect of successful DT implementation within organizations lies in business processes understanding to realize value from digital initiatives and investments. Process mining (PM) is a technique that utilizes transactional data on running processes to visualize and analyze business processes. In this regard, PM offers a potent capability to facilitate organizations transform their business processes through data-driven insight and can pave the way to accelerate DT within the organization.

The utilization of PM has proven to be advantageous for several major corporations like Airbus, BMW, Uber, Bosch, and SAP, as it helps to enhance their business processes [1], [2]. The successful integration of PM in large enterprises has motivated small and medium-sized enterprises (SMEs) to make efforts to extract data from their information systems and apply process analysis using PM [3]. However, applying PM in SMEs poses several issues, such as limited resources, immaturity of process, inadequate documentation, scarcity of resources, and poor management [4], [5]. Consequently, the stage of implementing PM in SMEs is becoming more difficult, and overcoming these issues is essential for unleashing the potential of PM across various businesses.

From a technical perspective, PM employs algorithms and advanced analytics to extract insights from event logs and other process data. It typically consists of multiple steps, including data extraction, data cleaning, process discovery, conformance checking, and process enhancement [4]. The core aspect of PM is acquiring data from diverse data sources, cleaning it, and transforming it into a meaningful event log, which is carried out during data extraction and data cleaning steps. Several research studies [6-8] have pointed out that generating an event log is the most challenging aspect of PM and is often viewed as complex, laborious, and time-consuming work. Thus, it is crucial and far from a trivial step [9].

When it comes to SMEs, the adoption of PM in SMEs faces several issues, as previously mentioned. Among those issues, limited resources, particularly the absence of data experts, may become a primary issue and lead SMEs to rely on external services for PM implementation. That condition leads to an inevitable rise in both development costs and time. To address those issues, we propose the development of an automated tool to enable PM in SMEs. Our objectives are twofold: first, we aim to assist organizations in the data preparation process by offering an automated tool as an alternative technology, even if data experts are not available, while maintaining the quality of the results. Second, we aim to reduce processing time by automating manual and repetitive tasks typically performed by individuals.

Lately, a wide range of commercial PM tools have been established and are available on the market, such as Fluxicon, Celonis, Apromore, UIPath, and many others [9]. Based on our research, out of these tools, only three offer users the capability to extract and generate an event log. These tools are Celonis, Apromore, and UIPath. However, it is important to note that these products provide limited features when it comes to generating an event log, specifically from SAP. This means that they may be suitable for organizations that rely on SAP for their system, but they may not be suitable for organizations that utilize a variety of systems beyond SAP, such as local ERP, spreadsheets, databases, or documents. Therefore, the design of a tool that can generate an event log from various data sources could offer advantages in terms of flexibility and productivity gains for organizations.

This work describes the development of the Event Data Factory (EDF). EDF is an easy-to-use tool for the integrated generation of PM event logs. The specific contributions of this work are as follows:

- We introduce EDF, an interactive and easy-to-use tool for PM event log generation that leverages low-code/no-code technology. This permits the generation of event logs without the need for hard coding.
- We present a metadata analysis-based method for automatically discovering event log attributes, namely case, activity, and timestamp, and assessing the quality of the resulting event logs.
- We perform a use case using real-world data to demonstrate the usability of the tool.

The rest of the paper is structured as follows: Section 2 discusses related works. Section 3 explains the detailed system architecture. Section 4 contains our implementation results. Section 5 concludes the paper with some future work directions.

II. Related Works

2.1 Process mining overview

The core principles of process mining include process discovery, conformance, and enhancement [10]. PM begins with an event log, which is essential for gaining insights into a process. Central to PM are the notions of case and trace in such event logs, where a case is the set of events in a single process instance and a trace is a sequence of the events in a case, respectively [11]. Hence, to perform PM analysis, an event log with minimal data such as case, activity, and timestamps is required. Additionally, an event log may also include attributes like resources and cost [12], [13], which show information about the business unit and cost associated with the process execution.

The topic related to event log generation has been widely discussed due to its complexity. The data for PM can be stored in various sources. Depending on the information systems, these data might have different characteristics and formats. Some research has proposed techniques for generating event logs from various data sources, like redo logs [14] or relational databases [15-17]. In addition to the data extraction, event labeling is an important aspect, as it involves finding mandatory attributes from the event log as minimum requirements, such as case, activity, and timestamp [6], [18], [19].

2.2 PM Tool and Low-code/no-code technology

The importance of low-code/no-code (LCNC) development for PM arises from the end-to-end involvement of business users and domain experts throughout the development of applications [20]. With LCNC technology, PM could be more efficient and accurate, and it could help reduce time spent on unnecessary manual tasks. The LCNC technology promises the opportunity of making PM more accessible to a broader range of people, enabling users with limited technical expertise to perform PM effectively, and thus increasing trust in PM results [21]. Consequently, it is no wonder that many PM vendors are trying to apply LCNC technology to their products, such as Celonis, Apromore, and UIPath [9].

Plenty of commercial tools are already available for users. Specifically, Celonis, Apromore, and UIPath offer a generic feature for generating an event log. However, most of the event logs are generated exclusively from the SAP system, limiting the range of data sources available for process exploration and selection. To the best of our knowledge, EDF is the first integrated and universal tool that facilitates automatic event log generation without having to write any code. It supports various data connectors that allow users to develop data pipelines from diverse data sources in a user-friendly manner. Moreover, the data flow is visualized in a graph that helps non-domain expert users have a better understanding of the process.

III. Architecture

3.1 Architecture

This section describes the architecture of the EDF tool. EDF is a tool designed for event log creation for PM analysis. It takes databases and/or files as input, transforms the data, extracts event log attributes, and finally generates an event log. The EDF tool is built on a cloud infrastructure, offering greater scalability and flexibility compared to the on-premises infrastructure. The salient features of EDF are:

- design for automatic data pipelines and easy setup through an intuitive user interface;
- support various data connectors to connect and extract data from many source systems; and
- metadata analysis-based method to automatically discover event attributes, reducing processing time.

The architecture of the EDF tool is shown in Fig. 1. EDF offers advantages due to its architecture, which consists of six functional layers: input, presentation, orchestration, analytical, infrastructure, and output. The **input layer (1)** manages and stores the data, making it available for application. The **presentation layer (2)** serves as the user interface, facilitating user interaction. The **analytical layer (3)** focuses on data extraction, processing, transformation, and analysis from various sources, preparing for event log generation. The **orchestration layer (4)** provides various data connectors, coordinates task execution, and enables scheduled automation. The **infrastructure layer (5)** virtualizes and provides essential services for the application, while the **output layer (6)** delivers the final event logs.

3.2 Data Workflow

This following section describes the data workflow of the EDF tool. Fig. 1 also illustrates the end-to-end data workflow. The EDF is comprised of seven components divided into three layers: presentation, analytics, and orchestration (see Fig. 1, part 2-4). The components include a workflow editor, a data catalog extractor, a data-to-event analyzer, a data source connector, a task scheduler, a data orchestrator, and a PM tool connector. Below is a detailed description of each component.

- **Orchestration model editor**. The orchestration model editor acts as the main user interface, and it receives connection settings as input,

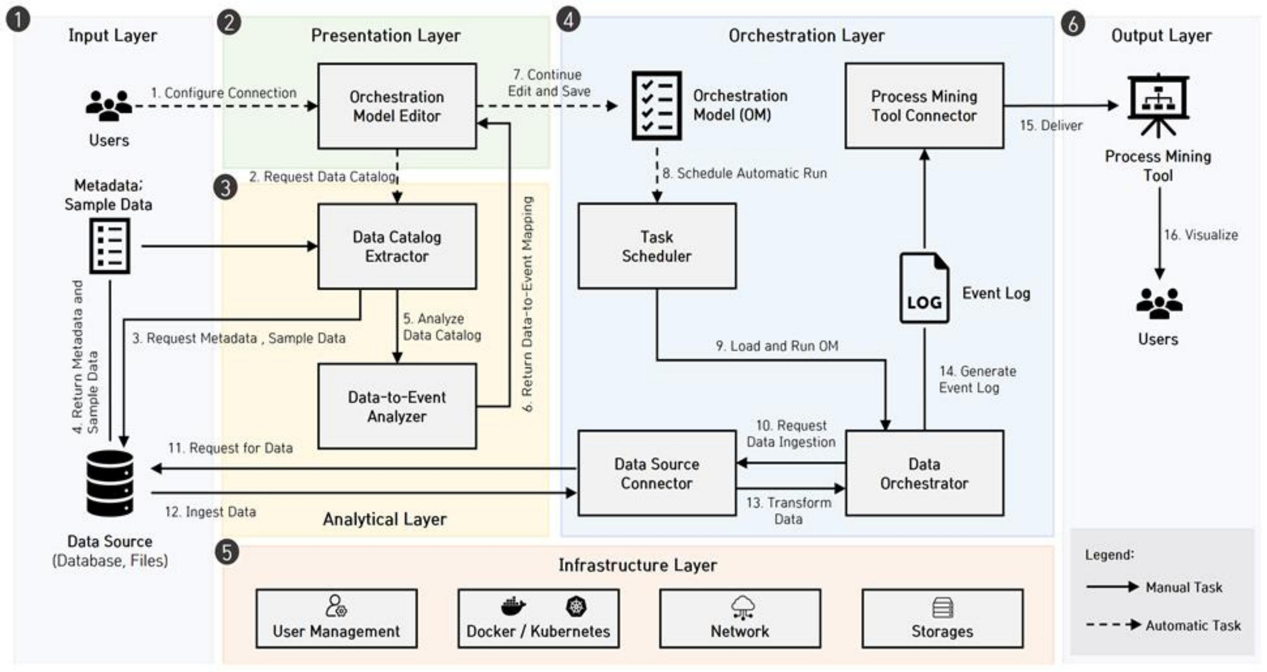


Fig. 1. The system architecture and components of EDF tool

such as database credentials or file information. Once the connection is verified, the orchestration model editor sends the input into the data catalog extractor.

- **Data catalog extractor.** The data catalog extractor receives connection settings from the orchestration model editor. It then extracts metadata from each data source and converts it into data models. Metadata describes detailed information about the structure, attributes, and types of the data at the same data source. The data models represent the logical structures of the data, including entity relationships. After generating data models, a data catalog is created. This data catalog comprises a collection of data models, defined as the primary knowledge for event log generation.
- **Data source connector.** The data source connector enables users to connect to external systems and extract data from them into the EDF platform. It consists of data connectors that enable dynamic integration with external data sources, including databases (i.e., MSSQL, MySQL, MariaDB, Oracle, and PostgreSQL) and flat files (i.e., XLS and CSV).
- **Data-to-event analyzer.** The data-to-event analyzer acts as a component for identifying event attributes by using metadata as an input and transforming the raw data into an event log. To perform PM analysis, an event log should contain at least three attributes, including case, activity, and timestamp related to the process execution [6], [12].
- **Task scheduler.** The task scheduler manages the execution sequence and enables users to perform routine tasks on the system by using orchestration models. With this component, users can schedule tasks to run at specified time intervals. It ensures tasks are performed consistently and saves time.
- **Data orchestrator.** The data orchestrator allows the system to efficiently manage ingested data and process it. It helps users manage the data pipeline in the system.
- **PM tool connector.** The final output of EDF is an event log with attributes of case, activity, and timestamp. To connect an event log to a PM tool, an external connector is needed. This connector enables users to easily connect the generated event log into a specified PM tool for analysis.

3.3 Automated Event Attribute Discovery

This section describes the process of event attribute discovery. To achieve this, we implement the metadata analysis-based method we introduced in our prior work [22] within the EDF tool. The detailed workflow for event log attribute discovery is shown in Fig. 2. The method consists of three main steps: (1) data preparation, (2) attribute discovery, and (3) log quality evaluation, which are described below.

1. **Data Preparation.** The very first step is data preparation. Our tool provides a data preparation function to extract metadata and sample data from various data sources. Metadata are transformed into data models. After data models are generated, the system creates a data catalog that consists of a set of data models as output. This process is performed within the data catalog extractor component. The data preparation process is briefly summarized in the pseudocode shown in Fig. 3.
2. **Event Attribute Discovery.** Given a data catalog, the objective of this step is to discover the event attribute necessary for the basic components of an event log, such as case, timestamp, and activity [23], [24]. This task can be achieved by performing these steps: (1) identifying timestamps and activities through pattern matching analysis; (2) calculating in-degree centrality; and (3) identifying cases through reference key analysis. The result will be a set of event mappings (d2e), each containing potential event attribute candidates. The event attribute discovery process is briefly

summarized in the pseudocode shown in Fig. 4.

3. **Log quality evaluation.** To achieve cost efficiency, sample event logs are produced. Initially, the system extracts sample data from various data sources. Next, sample event logs are created according to the event mappings. After generating the sample event logs, the system evaluates their quality. The output is a collection of event mappings along with their corresponding quality scores. Lastly, the user selects the preferred event mapping candidate, leading to the generation of a complete event log based on the selected event mapping. The complete process is detailed in the prior work [22].

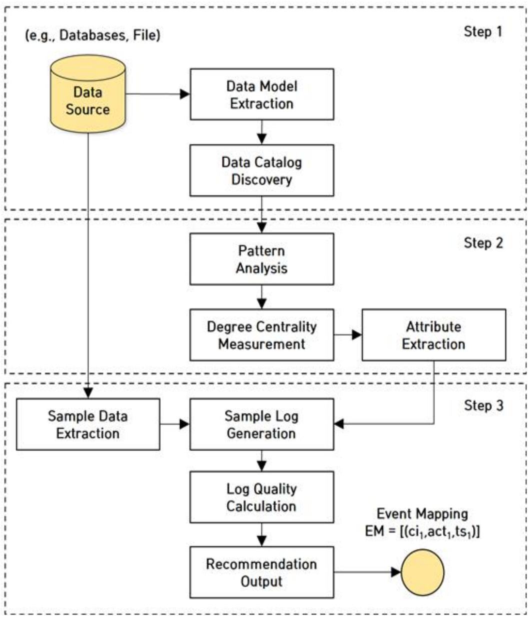


Fig. 2. Event attribute discovery workflow

```
Algorithm 1. Data Catalog Extraction
    Input:
    - List<string> conninfos = [] // List of Data Source Connection Info
    - int samplelimit = 25000 // Sample Row Count Limit
    Output:
    - Tuple<List<DataEntity>, List<DataEntityRelation>> data_catalog // Data Catalog
    - Dict<DataEntity, DataFrame> sampledata // Sample Data for each Data Entity
    Procedure: ExtractDataCatalog(conninfos, samplelimit)
        List<Tuple<List<DataEntity>, List<DataEntityRelation>> data_catalog = []
        Dict<DataEntity, DataFrame> sampledata = {}
        Foreach conninfo in conninfos, do
            DataSourceConnector conn = GetDataSourceConnector(conninfo)
            Tuple<List<DataEntity>, List<DataEntityRelation>> md = ExtractMetadata(conn)
            data_catalog.appendAll(md)
            Foreach de in data_catalog[0], do
                sampledata[de] = GetSampleData(conn, de, samplelimit)
        return data_catalog, sampledata
```

Fig. 3. Pseudocode for data catalog extraction

```
Algorithm 2. Data to Event Mapping
    Input:
    - Tuple<List<DataEntity>, List<DataEntityRelation>> data_catalog // Data Catalog
    - Dict<DataEntity, DataFrame> sampledata // Sample Data for each Data Entity
    Output:
    - Data2EventMapping d2e // Data Entity to Event Log Mapping
    Procedure: GetDataToEventMapping(data_catalog, sampledata)
        List<Tuple<Data2EventMapping, double>> d2ecandidates =
        AnalyzeMetadata(data_catalog, sampledata)
        If not d2ecandidates, do
            d2ecandidates = GenerateCandidates(data_catalog, sampledata)
            Data2EventMapping d2e = None
        If d2ecandidates, do
            d2e = d2ecandidates[0]
        return d2e
```

Fig. 4. Pseudocode for data catalog extraction

To evaluate the quality of event logs, we present a global function called the event log *interestingness* score [22]. This score aims to provide an indication of how interesting an event log is to be analyzed for analysis given a set of parameters. To do so, we have defined several statistical metrics inspired by previous works [6], [15]. These metrics help us evaluate the necessary structural properties that an event log should possess to be considered an interesting candidate. We calculated statistical metrics including *case identifier ratio, trace variant ratio, average ratio of unique activity per case, endpoint activity ratio*, and *start-end activity independence ratio* (see our prior work [22] for further explanation). The interestingness score is determined by combining the score values for each of the aforementioned metrics. When analyzing a process, it is desirable to have an event log with as many cases as possible. Therefore, a higher *interestingness* score could indicate that the event log exhibits complex processes or patterns that are worth further analysis [6].

IV. Results

4.1 Dataset and Environmental Settings

For our use case, we used a non-public dataset received from a company in South Korea. The dataset is stored in a relational database, and it consists of six tables: balanceaccount, marketplace, ordermaster, orderdetail, salesinfo, and qualitymaster. The dataset consists of around 4 months of logistic records. Due to confidential issues, we only provided the Entity Relationship (ER) diagram of the original data. The ER diagram of the logistic dataset is shown in Fig. 5, while the experimental environment is described in Table 1.

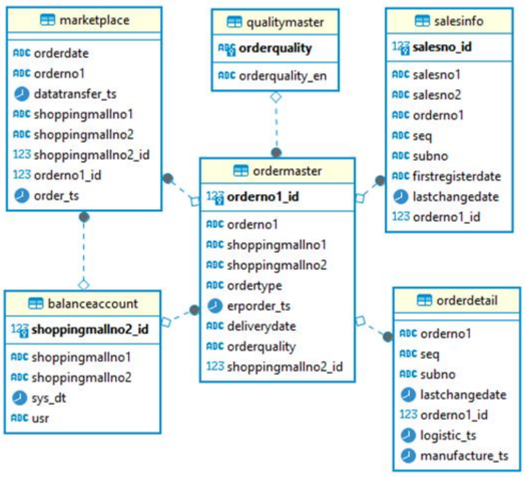


Fig. 5. ER Diagram of the logistic dataset

Table 1. Experimental Environment

Item	Description
OS	Ubuntu 22.04.3 LTS
Memory	32 GB
Storage	120 GB SATA SSD (OS) 64 GB SATA SSD (Data)
CPU	8 Core of Intel of Xeon Silver 4114 CPU @ 2.2GHz

4.2 The User Interface

EDF is a web application designed to simplify the process of generating event logs. It offers enhanced user-friendly experiences through low-code/no-code technology, making it simple to use for various user types. In Fig. 6, EDF consists of four menus: home, model, schedule, and task. To generate an event log, users can access the model menu, which is further divided into four main parts:

1. **Main menu.** This part displays all the menus in the EDF including, home, model, schedule, task, and logout button.
2. **Execution menu.** Buttons with essential functions such as delete, automatic event mapping discovery, test, save, and exit.
3. **Operation panel.** The operation panel lists all the components classified according to its functionality, such as import, export, convert, and operation.
4. **Model viewer.** The model viewer is the visualizer of the data pipeline. This part is prepared to define the data pipeline and all the components. The EDF model viewer is divided into three sections:
   - **Import.** The import section allows the user to connect various data sources at once. This is helpful when users need to extract from various sources. For example, users can load relational databases and files simultaneously.
   - **Operation.** The operation section shows the list of data entity components found in data sources. This section allows for all the operations tasks, such as data transformation, attribute discovery, and event log generation. Moreover, users can display detailed information for each component by clicking on the node.
   - **Export.** The export section allows users to export and connect the generated event log to a specific PM tool.

To illustrate the generation process in EDF, in Fig. 6, we provide an example of event log generation from various data sources and connect it into a PM tool, namely the IPR (Intelligent Process Re-Engineering) tool [25].

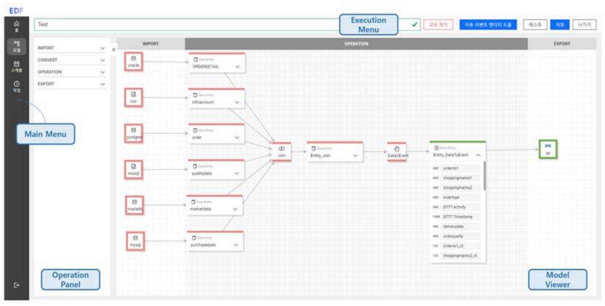


Fig. 6. The user interface of EDF tool

4.3 The Use Case

To show the usability of our tool, we conducted a use case that focused on the automatic generation of event logs using the proposed metadata analysis method [22] for a logistic process. In order to maintain consistency with the original setting, we initially designed a database server to store the original data. The process started by establishing a connection to the data source and extracting the required information. Once the connection was established, six tables (also referred to as entities) were extracted: *balanceaccount, orderdata, orderdetail, salesinfo, qualitymaster,* and *marketplace.* The rest of the process is automatically executed after the user clicks the “auto event mapping” button. As there were six entities, the join task was automatically carried out by the system to produce a unified table (referred to as Entity_join). Subsequently, a metadata analysis and timestamp-to-activity transformation are executed to identify event mappings and convert data into event log format with associated attributes (referred to as Data2Event), respectively. Finally, an event log (referred to as Event_Entity) with an interestingness score of 0.9369 was generated and linked to IPR (referred to as IPR). The entire generation process is illustrated in Fig. 7.

The fragment of the event log generated from EDF is shown in Fig. 8. Later on, the process mining model for the logistic process is processed with the IPR tool and visualized in Fig. 9. In Fig. 9, there are two process models: all paths and the most frequent path. The process model visualized the actual flow of the logistic process as it was executed. The start of the process is illustrated by the triangle at the top. Similarly, the end of the process is illustrated by the stop symbol at the bottom of the model. By default, the total frequencies of activity are displayed at the nodes, and the average lead time between activities are displayed at the arcs. As a result, there were six activities discovered in the event log, such as *order_ts_act, datatransfer_ts_act, erporder_ts_act, manufacture_ts_act, sys_dt_act,* and *logistic_ts_act.* In both process models, the start activity varies, but they all end with the activity *sys_dt_act.*

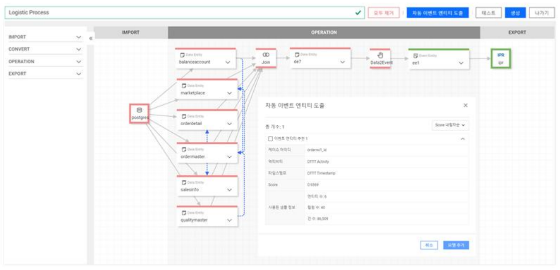
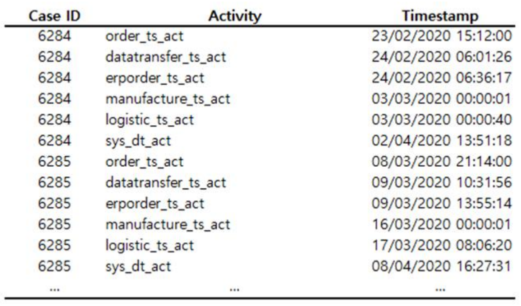


Fig. 7. Event log generation process on logistic data



Case ID	Activity	Timestamp
6284	order_ts_act	23/02/2020 15:12:00
6284	datatransfer_ts_act	24/02/2020 06:01:26
6284	erporder_ts_act	24/02/2020 06:36:17
6284	manufacture_ts_act	03/03/2020 00:00:01
6284	logistic_ts_act	03/03/2020 00:00:40
6284	sys_dt_act	02/04/2020 13:51:18
6285	order_ts_act	08/03/2020 21:14:00
6285	datatransfer_ts_act	09/03/2020 10:31:56
6285	erporder_ts_act	09/03/2020 13:55:14
6285	manufacture_ts_act	16/03/2020 00:00:01
6285	logistic_ts_act	17/03/2020 08:06:20
6285	sys_dt_act	08/04/2020 16:27:31
...	...	...

Fig. 8. Event log generation process on logistic data

The process model results uncovered an unnatural order of activity, particularly with the last two activities identified as manufacture_ts and sys_dt (see Fig. 7: most frequent path). To verify our result, we discussed it with domain experts, who confirmed that our system generated an unnatural order of activity. Upon data examination, we identified several issues associated with the original dataset, which contributed to these unnatural results. We found that a large number of null values were present, certain column names did not accurately represent their corresponding values, and some process data was missing. Consequently, we conclude that incomplete data has been shared. Our findings revealed the complexity of generating an event log from enterprise data, emphasizing the necessity for meticulous data preparation to ensure accuracy and reliability. Therefore, it is imperative to not completely disregard domain experts, as the event log must be reviewed in order to establish its validity.

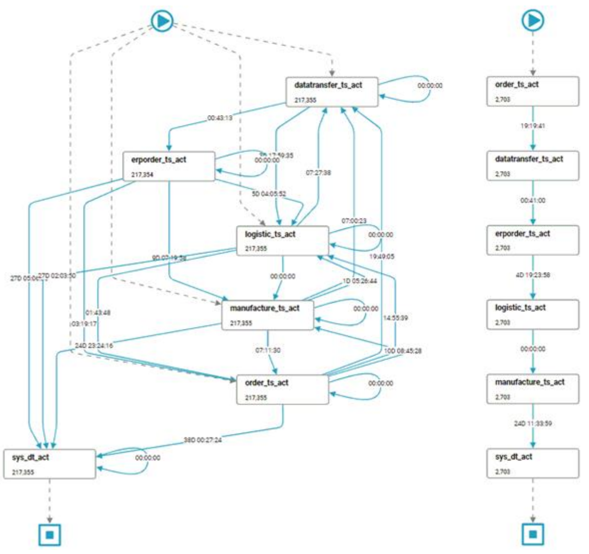


Fig. 9. Process model result
(Left) All path; (Right) The most frequent path

4.4 Performance Results

We have conducted an internal performance evaluation focusing on the performance and scalability of the EDF tool. However, due to the nonexistence of similar functionality and/or the unavailability of such functionality in the demo version of the other products, we are unable to conduct numerical performance tests to compare EDF with other commercial products such as Celonis, Apromore, and UIPath.

We presented the results, focusing on evaluating the internal performance and scalability of the tool. The cloud deployment allows us to evaluate the scalability and robustness of the tool by analyzing the processing time under the environment described in Table 1. In order to achieve that objective, we created data sets of varying sizes and measured the time it took to generate event logs. We employed both the original data and manipulated versions, where we divided the data into various scenarios based on the data size (i.e., the number of transactions). Specifically, we conducted experiments using scenarios 1, 2, 3, and 4, where each scenario corresponds to a data size of 100%, 50%, 25%, and 15%, respectively. For instance, scenario 2 indicates that 50% of the original data was used. The performance result of the EDF tool is presented in Table 2. The data size column represents the amount of data stored in the database, while the event log size represents the size of the generated event log derived from the respective data.

Table 2. Performance Result

Scenario	Data size (MB)	Event Log Size (MB)	Processing Time (s)
1	82	1,276	43
2	44	636	24
3	20	310	15
4	15	157	13

Performance is measured by measuring the time it takes to process the data sets of varying sizes. The experiment results showed that for smaller dataset sizes of 25% and 15%, loading and processing times are around 15 seconds. Meanwhile, for larger datasets, the processing time consistently remains below 60 seconds. Our experiment result demonstrates that our tool effectively tackles the challenge of large datasets, providing a robust foundation for big data computation. Additionally, microservice architecture development allows for independent scaling of services and ensures scalability and cost-effectiveness.

4.5 Observational Study Results

An observational study was carried out to assess how EDF's target users would use this tool for event log generation and to evaluate the usefulness of the tool. A total of 14 participants (7 male and 7 female) were recruited for the study with a various range of ages (10 participants were in the range of 28-35 and 4 participants were above 35). All participants were employees working in SMEs, and none of them had known about the EDF tool before. An online exit questionnaire consisting of 10 questions in total was utilized to gather feedback on the EDF tool's usability, learnability, and usefulness. Prior to the survey, participants were asked to explore EDF on their computers. Additionally, participants were also provided with a brief guideline and encouraged to test as many features as possible.

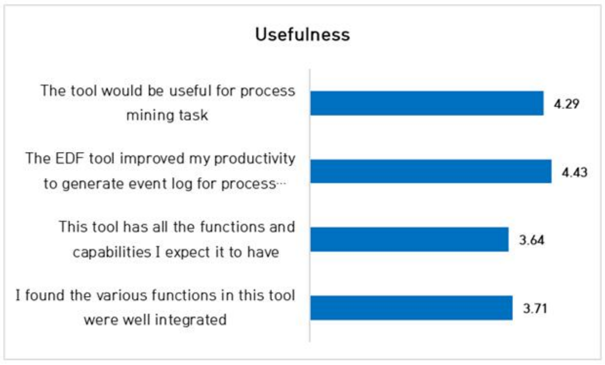


Fig. 10. Usefulness survey results

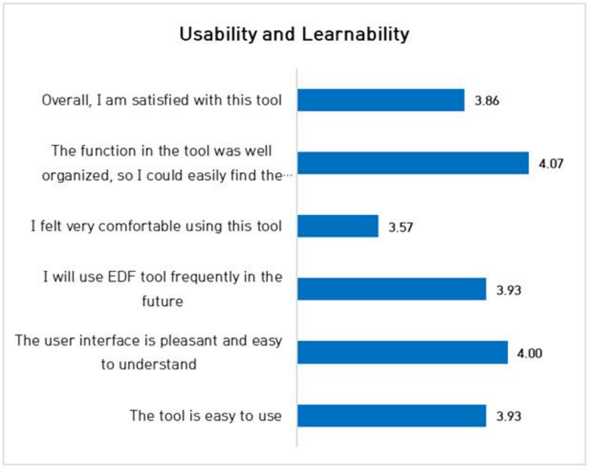


Fig. 11. Usefulness survey results

The exit survey included a series of 5-point Likert-scale questions (1 being “strongly disagree” and 5 being “strongly agree”). All the average Likert ratings for usefulness (Fig. 10), usability and learnability (Fig. 11) are all above 3.5. The results indicated that participants found our tool easy to use and beneficial for their purpose. However, the results also revealed that the lowest rating was given to the question “I felt comfortable using this tool.” highlighting user experience as a major concern.

4.6 Limitations and Future Works

In this work, we did not empirically compare EDF with other products. As stated in Section 4.4, the comparison between EDF and other products is limited due to the nonexistence of similar functionality and/or the unavailability of such functionality in the demo version of the other products. Additionally, it was observed that scalability and robustness evaluation could be expanded by considering different settings, such as the number of node clusters and system specifications. Lastly, future research should focus on obtaining more datasets that reflect real-world scenarios.

The tool is currently in its initial development phase and presents opportunities for potential improvement. Through evaluation, we have identified several key aspects where EDF can be improved and future development plans.

- There is a need to enhance the menu structure, and participants with limited knowledge of PM were interested in understanding how the system makes outputs. Thus, we believe that enhancing the user interface with best practices and adding a step-by-step tutorial model could assist users and enhance their experience.
- The current EDF tool conducts analysis based on column names and their type, which results in the inability to process data with abbreviated column names. Therefore, it is worth noting that assessing the column value is important when dealing with abbreviated column names. Additionally, employing machine learning approaches, such as transformers or attention mechanisms, captures the semantics of column contents, helps in understanding context across tables.
- The current tools can only process batch data, and given the continuous influx of new data generated by application systems, expanding our environment into a streaming setting could be a feasible strategy to manage data synchronization.
- Our EDF relies solely on the evaluation of the quality metrics; however, in practice, the creation of event logs is a subjective task where the quality and correctness of the generated event logs highly depend on the expertise and experience of domain experts. Hence, we aim to improve our quality metrics by integrating domain experts’ feedback through transfer learning.

V. Conclusion

In this paper, we introduced EDF, an interactive tool for assisting event log generation and enabling process mining analysis in SMEs. EDF leverages low-code/no-code technology, thus allowing users to easily see the data flow through graph visualization. It offers various data connectors, ranging from flat files to databases, and incorporates a metadata-based algorithm to generate an event log automatically. As an output, an event log is generated, along with its quality (i.e., interestingness score).

We have presented instantiation as a software prototype and tested its functionality on enterprise data. The cloud-based deployment ensures scalability, allowing dynamic resource scaling to meet users' demands. We have demonstrated that our tool is robust and can cope with varying volumes of data. Moreover, the results of the observational study also highlighted that our tool is easy to use and beneficial during the preparation stage of PM analysis. We hope our work can stimulate further research and development of process mining tools that lower the barrier to understanding and appropriately applying PM in SMEs.

ACKNOWLEDGEMENT

This work is supported by the Busan Innovation Institute of Industry, Science & Technology Planning (BISTEP) grant funded by the Ministry of Science and ICT (MSIT) (No. 2023-연구산업진흥단지육성-부산-1세부-스타트업2). We would also like to express our gratitude to our colleagues, Yujeong Jang, Yejin Seo, Sinwoo Kim, Naeun Hyun, and Junchan Lee for their participation in this project and the development of the prototype implementation.

REFERENCES

[1] T. Grisold, J. Mendling, M. Otto, and J. vom Brocke, “Adoption, use and management of process mining in practice,” Bus. Process Manag. J., vol. 27, no. 2, pp. 369–387, Mar. 2021, doi: 10.1108/BPMJ-03-2020-0112/FULL/PDF.

[2] L. Reinkemeyer, Process Mining in Action - Principles, Use Cases and Outlook. Springer Cham, 2020. doi: 10.1007/978-3-030-40172-6.

[3] S. J. Urrea-Contreras et al., “Applying Process Mining: The Reality of a Software Development SME,” Appl. Sci. 2024, Vol. 14, Page 1402, vol. 14, no. 4, p. 1402, Feb. 2024, doi: 10.3390/APP14041402.

[4] F. ; B. ; Zafrah et al., “Eliminating Non-Value-Added Activities and Optimizing Manufacturing Processes Using Process Mining: A Stock of Challenges for Family SMEs,” Sustain. 2024, Vol. 16, Page 1694, vol. 16, no. 4, p. 1694, Feb. 2024, doi: 10.3390/SU16041694.

[5] M. Eggert and J. Dyong, “Applying Process Mining in Small and Medium Sized IT Enterprises – Challenges and Guidelines,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 13420 LNCS, pp. 125–142, 2022, doi: 10.1007/978-3-031-16103-2_11/COVER.

[6] E. G. L. de Murillas, H. A. Reijers, and W. M. P. van der Aalst, “Case notion discovery and recommendation: automated event log building on databases,” Knowl. Inf. Syst., vol. 62, no. 7, pp. 2539–2575, Jul. 2020, doi: 10.1007/S10115-019-01430-6/FIGURES/14.

[7] D. Dakic, D. Stefanovic, T. Lolic, D. Narandzic, and N. Simeunovic, “Event Log Extraction for the Purpose of Process Mining: A Systematic Literature Review,” Springer Proc. Bus. Econ., pp. 299–312, 2020, doi: 10.1007/978-3-030-44711-3_22/COVER.

[8] T. Kampik and M. Weske, “Event Log Generation: An Industry Perspective,” Lect. Notes Bus. Inf. Process., vol. 450, pp. 123–136, 2022, doi: 10.1007/978-3-031-07475-2_9/COVER.

[9] “Market Guide for Process Mining.” Accessed: Feb. 01, 2024. [Online]. Available: https://www.gartner.com/en/documents/3939836

[10] L. Zimmermann, F. Zerbato, and B. Weber, “What makes life for process mining analysts difficult? A reflection of challenges,” Softw. Syst. Model., pp. 1–29, Nov. 2023, doi: 10.1007/S10270-023-01134-0/TABLES/11.

[11] D. A. Fischer, K. Goel, R. Andrews, C. G. J. van Dun, M. T. Wynn, and M. Röglinger, “Towards interactive event log forensics: Detecting and quantifying timestamp imperfections,” Inf. Syst., vol. 109, p. 102039, Nov. 2022, doi: 10.1016/J.IS.2022.102039.

[12] W. Van Der Aalst et al., “Process mining manifesto,” Lect. Notes Bus. Inf. Process., vol. 99 LNBIP, no. PART 1, pp. 169–194, 2012, doi: 10.1007/978-3-642-28108-2_19/COVER.

[13] R. P. J. C. Bose, R. S. Mans, and W. M. P. Van Der Aalst, "Wanna improve process mining results?," Proc. 2013 IEEE Symp. Comput. Intell. Data Mining, CIDM 2013 - 2013 IEEE Symp. Ser. Comput. Intell. SSCI 2013, pp. 127-134, 2013, doi: 10.1109/CIDM.2013.6597227.

[14] D. Bano, T. Lichtenstein, F. Klessascheck, and M. Weske, "Database-Less Extraction of Event Logs from Redo Logs," Bus. Inf. Syst., vol. 1, pp. 73-82, Jul. 2021, doi: 10.52825/BIS.V1I.66.

[15] R. Andrews, C. G. J. van Dun, M. T. Wynn, W. Kratsch, M. K. E. Röglinger, and A. H. M. ter Hofstede, "Quality-informed semi-automated event log generation for process mining," Decis. Support Syst., vol. 132, p. 113265, May 2020, doi: 10.1016/J.DSS.2020.113265.

[16] J. D. Hernandez-Resendiz, E. Tello-Leal, U. M. Ramirez-Alcocer, and B. A. Macías-Hernández, "Semi-Automated Approach for Building Event Logs for Process Mining from Relational Database," Appl. Sci. 2022, Vol. 12, Page 10832, vol. 12, no. 21, p. 10832, Oct. 2022, doi: 10.3390/APP122110832.

[17] A. Berti, G. Park, M. Rafiei, and W. M. P. van der Aalst, "An Event Data Extraction Approach from SAP ERP for Process Mining," Lect. Notes Bus. Inf. Process., vol. 433 LNBIP, pp. 255-267, 2022, doi: 10.1007/978-3-030-98581-3_19/FIGURES/4.

[18] S. Bala, J. Mendling, M. Schimak, and P. Queteschiner, "Case and activity identification for mining process models from middleware," Lect. Notes Bus. Inf. Process., vol. 335, pp. 86-102, 2018, doi: 10.1007/978-3-030-02302-7_6/FIGURES/5.

[19] D. A. Fischer, K. Goel, R. Andrews, C. G. J. van Dun, M. T. Wynn, and M. Röglinger, "Enhancing event log quality: Detecting and quantifying timestamp imperfections," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 12168 LNCS, pp. 309-326, 2020, doi: 10.1007/978-3-030-58666-9_18/COVER.

[20] A. Van Looy, "On the synergies between business process management and digital innovation," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 11080 LNCS, pp. 359-375, 2018, doi: 10.1007/978-3-319-98648-7_21/TABLES/4.

[21] C. Ullrich, T. Lata, and J. Geyer-Klingeberg, "Celonis Studio - A Low-Code Development Platform for Citizen Developers," in International Conference on Business Process Management, 2021.

[22] Lee. S, Won. S, Sutrisnowati. R. A., Pulshashi. I. R., Prathama. F., and Kim. U, "System and Method For Generating Multi-Perspective Event Logs," KR. Patent No. 1026551980000, 2024.

[23] W. M. P. van der Aalst, "Process Mining: A 360 Degree Overview," Lect. Notes Bus. Inf. Process., vol. 448, pp. 3-34, 2022, doi: 10.1007/978-3-031-08848-3_1/TABLES/3.

[24] J. De Weerdt and M. T. Wynn, "Foundations of Process Event Data," Lect. Notes Bus. Inf. Process., vol. 448, pp. 193-211, 2022, doi: 10.1007/978-3-031-08848-3_6/FIGURES/6.

[25] "IPR." [Online]. Available: https://ipr.iochord.io/process-mining

Authors

Frans Prathama received the B.Cs., degree in Informatics Engineering from Universitas Ma Chung, Indonesia in 2016, and M.Eng degree in data analytics from Hankuk University of Foreign Studies, South Korea, in 2021. He is currently working as a research engineer at IOChord R&D Center. He is mainly working on data analytics, machine learning, and recommender system.

Seokrae Won received B.S. degree in Material Science and Engineering from Pukyong National University in 2002 and M.S. degree in Management and Technology from Pukyong National University in 2018, South Korea. Won joined IOChord in 2017 and is currently working as Director of R&D Center. He is working closely on cloud computing, big data systems, process mining and data mining.

Iq Reviessay Pulshashi received B.S degree in Informatics Engineering from Sepuluh Nopember Institute of Technology (ITS), Indonesia, in 2011, and Ph.D degree in Industrial Engineering from Pusan National University, South Korea, in 2020. He is currently working as a research engineer at IOChord R&D Center. He is mainly working on process mining, database, cloud computing, and big data systems.

Riska Asriana Sutrisnowati received B.S. degree in Information Systems from Sepuluh Nopember Institute of Technology (ITS), Indonesia, in 2010, and Ph.D. degrees in Big Data from Pusan National University, South Korea, in 2017. She is currently working as a research engineer at IOChord R&D Center. She is mainly working on data mining and process mining research area.