applied sciences

MDPI

Article

Semi-Automated Approach for Building Event Logs for Process Mining from Relational Database

Jaciel David Hernandez-Resendiz [1,†], Edgar Tello-Leal [2,*,†], Ulises Manuel Ramirez-Alcocer [1] and Bárbara A. Macías-Hernández [2]

[1] Multidisciplinary Academic Unit Reynosa-Rodhe, Autonomous University of Tamaulipas, Reynosa 88779, Mexico
[2] Faculty of Engineering and Science, Autonomous University of Tamaulipas, Victoria 87000, Mexico
* Correspondence: etello@docentes.uat.edu.mx
† These authors contributed equally to this work.

Abstract: Process mining is a novel alternative that uses event logs to discover, monitor, and improve real business processes through knowledge extraction. Event logs are a prerequisite for any process mining technique. The extraction of event data and event log building is a complex and time-intensive process, with human participation at several stages of the procedure. In this paper, we propose a framework to semi-automatically build an event log based on the XES standard from relational databases. The framework comprises the stages of requirements identification, event log construction, and event log evaluation. In the first stage, the data is interpreted to identify the relationship between the columns and business process activities, then the business process entities are defined. In the second stage, the hierarchical structure of the event log is specified. Likewise, a formal rule set is defined to allow mapping the database columns with the attributes specified in the event log structure, enabling the extraction of attributes. This task is implemented through a correlation method at the case, event, and activity levels, to automatic event log generation. We validate the event log through quality metrics, statistical analysis, and business process discovery. The former allows for determining the quality of the event log built using the metrics of accuracy, completeness, consistency, and uniqueness. The latter evaluates the business process models discovered through precision, coverage, and generalization metrics. The proposed approach was evaluated using the autonomous Internet of Things (IoT) air quality monitoring system's database and the patient admission and healthcare service delivery database, reaching acceptable values both in the event log quality and in the quality of the business process models discovered.

Keywords: event log; process mining; relational database; event data; business process; attribute extraction

check for updates

Citation: Hernandez-Resendiz, J.D.; Tello-Leal, E.; Ramirez-Alcocer, U.M.; Macías-Hernández, B.A. Semi-Automated Approach for Building Event Logs for Process Mining from Relational Database. *Appl. Sci.* **2022**, *12*, 10832. https://doi.org/10.3390/app122110832

Academic Editor: Stefan Schönig

Received: 31 August 2022
Accepted: 21 October 2022
Published: 26 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Process mining can discover, monitor, and improve business processes by extracting knowledge from event logs [1,2]. An event log is a collection (historical data) of execution traces of a business process [3]. Process mining techniques can identify bottlenecks, anticipate problems in process execution, record business rule violations, recommend countermeasures, provide information for decision-making, and streamline the process. In this way, the process discovery task through an algorithm can automatically produce a business process model from the behavior observed in the event log without requiring a-priori information [4,5]. The conformance checking task verifies the relationship between the behavior observed in the event log and the behavior modeled in the business process, making it possible to identify possible deviations [1,6]. On the other hand, process enhancement consists of extending or improving the existing process model using the current process information contained in its event log [1,7], the results obtained from process discovery and conformance checking, and analysis of attribute properties of the event log.

When a process-oriented information system is not implemented in the organization, a procedure is required to extract data events from an information system to convert them into an event log [8,9]. These systems may have many database tables from which event data can be extracted that need to be converted into a structure and format that a process mining method can interpret. Furthermore, the event log generation procedure may involve multiple databases from different back-end systems [10]. A high precision achieved by a process mining method is related to the quality of the event log used [11]. Therefore, validating and verifying event data conversion to event logs is essential to guarantee reliable results. The eXtensible Event Stream (XES) is the standard format for event log specification [12], facilitating the exchange of event data between different systems and implementing process mining techniques. The event log is a fundamental element for implementing methods based on process mining. The building of event logs presents several challenges; for example, extracting events from a relational database, deep knowledge of the domain of the executed business processes is required and of the application context of these business processes. Knowledge of the domain is essential to selecting the correct attributes of the database tables [13]. Hence, experts at the technical and business levels are required since knowing the semantics of the event data values is essential.

Several approaches are available in state-of-the-art to support the generation of event logs: in a semi-automatic way from relational databases [14], derive process activities through a method based on natural language inference [15], from a real-world data warehouse of a healthcare system [16], from database redo logs [17,18], from data stored on the blockchain [19], and conversion of event data to event logs using deep learning [11]. An event data source that has provoked great interest in the process mining community is information systems based on Internet of Things (IoT) technology [20]. These systems generate a large amount of data, which can be collected utilizing robots or independent sensors, and processed automatically by web services, which receive the request, process it, and return a response, allowing an operation to continue in the thing. From a business process point of view, these approaches can be classified within the emerging robotic process automation technology [21]. In these systems, things operate in an interconnected ecosystem, with an autonomous but cooperative operation with other things, embedded systems or software components, to coordinate actions that allow compliance with the objectives and policies previously defined or updated in run-time.

Hence, business process execution and monitoring through IoT technologies can represent an important innovation source for organizations [22,23]. In addition, process mining based on IoT event data can enable an even more comprehensive view of business processes and realize the unused potential for process optimization [24]. For example, optimization of business process flows through analyzing big data (built with instances of executing these processes) or optimizing processes using smart objects [25].

Therefore, this paper presents a semi-automated approach to generating event logs from a relational database. This approach is based on a framework that guides the event data extraction, the definition of the event log structure, and automatically converting event data to the event log. In the event log generation, a set of formal rules have been defined that guarantee the precision and efficiency of the method. The built event log quality is assessed using the metrics of accuracy, consistency, completeness, and uniqueness. Finally, the generated event log is evaluated and tested through discovering the business process by implementing the directly-follows graph (DFG) and split-miner algorithms. The quality of the discovered process model is evaluated by employing precision, recall, and generalization metrics, obtaining very acceptable values. A real case of a relational database of an IoT air quality monitoring system and a case of the flow of the process of admission and patient care with symptoms of pneumonia in a hospital is used to evaluate the framework.

2. Background

2.1. Preliminaries and Definitions

An event log is a hierarchically structured file with historical information about the execution of a business process generated by a process-oriented information system. Cases organize attribute values within an event log. A case contains a group of events that belong to the same business process execution. An event is made up of properties or attributes. An event's typical attributes are the activity's name, timestamp, and resource. Depending on the granularity required in the event log and the type of process mining in which this event log will be used, other attributes can be included. In this direction, the sequence of events related to a case is known as a trace. In summary, the following should be considered when generating an event log:

- A process consists of cases.
- A case consists of events such that each event is related to a case.
- The events within a case must be ordered.
- Events can have a set of attributes.

The proposed approach is formalized by the following definitions, considering the elements that comprise the framework explained in Section 3.

Definition 1 (**Event log**). *An event log L is composed of a set of traces T (see Definition 3), where T is all the business process execution instances. Each trace T_i is formed of a finite set of events E (see Definition 2), which are the events executed in the business process.*

Definition 2 (**Event**). *An event $e \in E$ is a tuple $\varepsilon = (c, e, et)$, where $e \in E$ is a process event, $c \in C$ is a case identifier that allows identifying the events executed within the case identifier c. et is a set of attributes describing each event e. The main attributes are activity identifier ($a \in et$), timestamp ($time : stamp \in et$), activity name ($concept : name \in et$), and resource that executed the activity ($resource \in et$), among others.*

Definition 3 (**Trace**). *A trace is a sequence $T_c =< e_1, e_2, ..., e_{|T_c|} > \in E$ of events belonging to the same case (process instance), identified with the same case identifier c. The events $< e_1, e_2, ..., e_{|T_c|} > \in E$ are ordered by their timestamps ($time : stamp$).*

Definition 4 (**BPMN process model**). *This statement is based on the definition proposed by [26]. A business process model and notation (BPMN) process model is a graphical component $M = (i, o, T, G, E_m)$, where i is the initial event, o is the final event, and T is a non-empty set of tasks. $G + \cup Gx \cup G*$ is the union of the AND gateway set (G+), the XOR gateway set (Gx), and the OR gateway set (G*); and $E_m \subseteq (T \cup G \cup i)x(T \cup G \cup o)$ is the set of edges. Given $g \in G$, g is a division gateway if it has more than one output edge or a union gateway if it has more than one input edge.*

Definition 5 (**Directly-Follows Graph (DFG)**). *It is a component $\varrho = (N, Ed)$ where N is the nodes that represent the events in the event log; and the edges Ed represent the direct relationship between two nodes (A, B), if at least one trace T in the event log L, the source node A is followed by the destination node B ($A \rightarrow B$) [26,27]. These graphs allow you to add or display additional model information, for example, the frequency of a relationship between node A and node B within the event log and the time elapsed between the transition from one node to another.*

2.2. Evaluation Metrics

The following metrics are used to calculate the time consumed in executing the activities at the case and event levels.

Average time of cases (ATC): calculates the average time of the duration of the traces T in the event log L, where $\frac{1}{|T|} \sum_{e_j \in T}^{|T|} (e_{j+1}.timestamp - e_j.timestamp)$ is the average time

of the duration of the trace T, while $\sum_{e_j \in T}^{|T|}(e_{j+1}.timestamp - e_j.timestamp)$ is the total time of the duration of the traces T, which is obtained by adding the duration of each of the events within the trace T. We calculate the ATC as follows (Equation (1)).

$$ATC = \frac{1}{\mid L \mid} \sum_{T \in L}^{|L|} \frac{1}{\mid T \mid} \sum_{e_j \in T}^{|T|} (e_{j+1}.timestamp - e_j.timestamp) \quad (1)$$

Average time number of events (ATE): calculates the average time of the duration of each event $e \in E$ in the entire event log L. The time it takes for each of the e events in each of the traces is added ($successor(e).timestamp - e.timestamp$); where $successor(e)$ refers to the next event of event $e \in T$, and lg is the total of times event e appears in the event log L.

$$ATE = e \in E(\frac{1}{\mid lg \mid} \sum_{T \in L}^{|L|} \{\ successor(e).timestamp - e.timestamp,\ (lg = lg + 1) \quad if \quad e \in T\ \}) \quad (2)$$

The quality of the process models discovered from event logs built using the proposed framework (see Section 3) is evaluated using metrics defined below.

Precision: this metric indicates how much additional behavior the process model allows that was not seen in the event log; the discovered model should not allow a behavior other than what is found in the event log [28,29]. We calculate the precision as follows (Equation (3)):

$$Qp = 1 - \frac{\sum_{nodes_visited} \#visits * (\#outgoing_edges - \#used_edges_by_replay)}{\sum_{nodes_visited} \#visits * \#outgoing_edges} \quad (3)$$

where $\sum_{nodes_visited}$ refers to a sum of each node which counts the total number of times visited ($\#visits$) by subtracting the total of its output paths ($\#outgoing_edges$) minus the total of those output paths visited in the replay of the model ($\#used_edges_by_replay$), divided by the product of the multiplication of the sum of the nodes visited $\sum_{nodes_visited}$-$\#visits$ by the total of the exit paths ($\#outgoing_edges$).

Recall: it measures the degree to which all cases within the event log can be reproduced by the discovered process model [28,29]. We calculate the recall as follows (Equation (4)):

$$rp = \frac{\sum_{\sigma \in L}(1 - \frac{replay(\sigma)}{length(\sigma)})}{|L|} \quad (4)$$

where it is calculated from a division of the sum of the length of the case reproducible by the discovered model between the length of the case, divided by the total number of cases in the event log.

Generalization: estimates how well the process model describes the system's behavior and not just the relationship between the event log and the observed system behavior. The idea of generalization is that if all parts of the process model are used frequently, the model is likely to be generic [28,30]. To calculate this measure following equation is used (Equation (5)):

$$Qg = 1 - \frac{\sum_{nodes}(\sqrt{\#executions})^{-1}}{\#nodes_in_model} \quad (5)$$

where $\sum_{nodes}(\sqrt{\#executions})^{-1}$ indicates the sum of the square root of the total number of repetitions of each node to the power of -1 (to normalize it to a value between 0 and 1), divided by the total number of nodes of the process model ($\#nodes_in_model$).

2.3. Related Work

In [31], the relational XES (RXES) framework is proposed based on a scheme to store events in a database. This framework is derived from the UML class diagram used to

define the XES standard, separates the content of the event logs, traces and events in different tables, assigns unique identifiers, and maintains the correlation between these elements. In the testing stage, it is shown that RXES generally uses less memory compared to file-based implementations of XES. In [32], the authors present a declarative process mining approach by implementing SQL query patterns for extracting the constraints of the declarative process model from the database. The authors propose using a RXES schema, assuming the data will be stored in this database type. They insert several event logs in a database based on the RXES schema to validate the approach. Defining a mapping between the attributes of the tables through a template supported by predefined SQL queries. The preceding allows for discovering constraints involving different perspectives of the business process and some control flow elements. In our approach, the event data stored in a relational database are identified and extracted to build an event log based on the XES standard.

The work in [33] presents another approach to discovering multi-perspective process models from event logs that allows the user to discover declarative process models taking into account all the information that an event log can provide. Similarly, a set of SQL-based patterns is used for retrieving the event data. In [14] the authors propose a quality-based semi-automated approach for generating event logs from a relational database, developing an assessment of their quality based on common data quality dimensions. From the quality dimensions, support is assumed to decide the data's suitability for implementing a process mining approach. This approach allows the mapping of database columns to event log attributes, making it possible to generate the event log. The main difference with our experiment is the granularity level in data events extraction. Our approach proposes using the attribute values that represent an activity within the business process, focusing on the fine granularity level in extracting the event data. Additionally, the built event log is evaluated using quality metrics to guarantee the quality of the discovered business process.

3. Materials and Methods

The proposed framework consists of three stages: identification of requirements considered at the time of design and construction of the database, construction of the event log, and evaluation of the event log. Figure 1 shows the activities and tasks that form the proposed framework to build event logs semi-automatically.

3.1. Requirements Identification Stage

The requirements identification stage is performed manually and requires the participation of system and business users, business process specialists, database designer, and database administrator. These stakeholders will support in identifying the elements (tables and attributes) related to the execution of each business process (both at a macro level and a micro-level) within the conceptual (based on system documentation), logical and physical designs of the database in operation.

3.1.1. Data Interpretation Analysis

In this activity, the stakeholders will know and identify the concepts at the business level and at the technical level of the business process and how the relational database (DB) supports the operation of the business process by storing the data generated in each process transaction. In addition, the database schema $schema(DB)$ elements are analyzed: the set of tables $Tb \in schema.tables(DB)$, the name of the columns $TbColumns \in schema.columns(Tb_i)$, primary keys $PK \in TbColumns$, foreign keys $FK \in TbColumns$, and the values stored in each row $rows \in Tb_i$ for each attribute $TbColumns$. Moreover, the activity consists of studying the entity-relationship diagram of the database $ER(DB)$ and its link with the elements of the business process model. The links between the tables and attributes with the business process activities are identified through reverse engineering. Furthermore, the relationships between the database tables are used in the specification of data extraction rules.

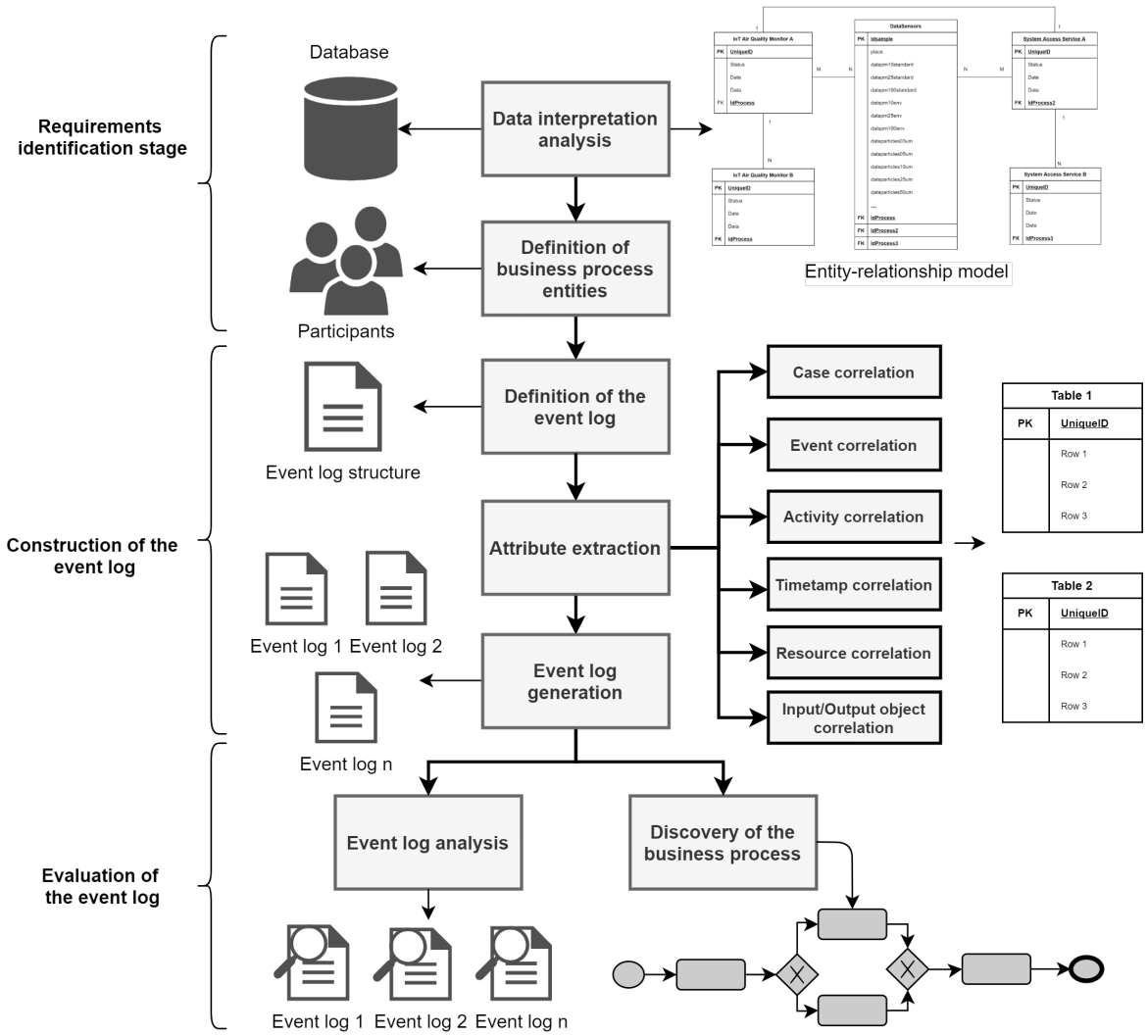


Figure 1. Sequence of activities that make up the framework to generate event logs.

3.1.2. Definition of Business Process Entities

The activity consists of identifying the possible entities of the business process within the attributes that make up each database table. In the search for the entities of the business process, the type of process must be considered: (1) in collaborative or inter-organizational business processes, the participating entity (*Ps*) will be searched, and it must meet the condition of at least two participants identified; (2) in the private business processes, the entity name of Lane (*Ls*) will be searched for, if this entity is not available, the process (*P*) name entity will be searched. Then, the tables related to each participant (lane or process) $Tb \subset schema.tables(DB)$ will be selected from analyzing the previously carried out of the tables, attributes, and relationships. For example, each $Ps_i \leftarrow Tb \subset schema.tables(DB)$ will be related to a set of database (*DB*) tables that store the values of the business process execution corresponding to the participant.

3.2. Construction of the Event Log

3.2.1. Definition of the Event Log

This activity specifies a structure for the event log L that will be built (see Definition 1), considering a design based on the XES schema. For each *event e* within the *traces* $T \in L$, the following attributes (*et*) will be considered: case identifier (mandatory), activity name (mandatory), timestamp, participant, resource, and in the case of the process of collaborative business, at least one attribute related to the exchange of messages between the participants

must be included (for example business documents or messages). The event log structure will allow the values of the actions carried out during the execution of the business process (private or collaborative) to be stored.

3.2.2. Attribute Extraction

The activity's objective is to create a mapping between the columns of the relational database with the attributes defined for the event log, using the knowledge generated in the previous activities. This activity is performed of several tasks that allow the definition of correlation between the parts for constructing the event log.

Case correlation. It refers to searching and selecting an identifier for the cases that will be extracted from the database. This identifier will be linked to the *IdCase* from the event log. An *IdCase* refers to a single instance of the process. The case correlation task is defined by specifying the table $TbCase \subset schema.\ tables(DB)$ and the columns $TbNCase \in schema.columns(TbCase)$ that store data to extract the case identifier attribute ($IdCase \leftarrow TbNCase$). In the structure of an event log, all *cases* are derived from the *process* (IdProcess), which represents the root of the event log.

Event correlation. In this task will define the unique identifier for each event generated by the business process's execution. It is necessary to identify the table $TbIdEvent \in schema.tables(DB)$ and the column $TbNIdEvent \in schema.columns(TbIdEvent)$ that allows us to obtain the identifier of the event e ($IdEvent \leftarrow TbNIdEvent$). Each event in the event log must refer to a case. Furthermore, the events are related to some activity.

Activity correlation. In this task, the columns that store the name of activities (for example, register a sales order, evaluate a discount request, or reject a purchase order) executed within the business process flow will be selected. It is necessary to identify the tables $TbActivityC \subset schema.tables(DB)$, relations $ER(DB)$ and the columns $TbNActivityC \in schema.columns(TbActivityC)$ that allow obtaining the name of the activities carried out by a resource within a valid behavior for a business process participant, which will be rendered by the activity (a) name attribute in the event e ($concept:name \leftarrow TbNActivityC$). The value of the activity name column will be mapped to the *concept:name* attribute of the event log.

TimeStamp correlation. This task is composed of a rule to build the *timestamp* attribute, where the table $TbTimeS \in schema.tables(DB)$ and the column $TbNTimeS \in schema.columns(TbTimeS) \mid type(TbNTimeS) = time:stamp$, allowing to know the date and time of the execution of the *event e* ($time:stamp \leftarrow TbNTimeS$), allowing to calculate the time consumed in the business process execution at the case level.

Resource correlation. This task links the column that contains information on users, services, or systems with the capacity to execute user, service, manual, business rule, or script type tasks (defined as atomic activities in the BPMN language) with the resource attribute of the event log. Then, we must identify the table $TbResource \subset schema.tables(DB)$ and its columns $TbNResource \subset schema.columns(TbResource)$ that contain the name of the user, system, or web service that executed each *activity a* ($resource \leftarrow TbNResource$).

Input/Output object correlation. The tables and their columns containing data from input or output objects (for example, business documents) exchanged by the business process participants through tasks or intermediate events of sending or receiving messages (in the case of inter-organizational business processes) are specified. Then, the tables $TbDataOI \subset schema.tables(DB)$ and their columns $TbNDataOI \subset schema.columns(TbDataOI)$ are specified with data from the message exchange containing input/output objects from the event set $e \subset E$ ($input/output_data \leftarrow TbNDataOI$).

3.2.3. Event Log Generation

The traces that make up the event log are generated based on the following rules. Each different value of the $rows \in IdCase \leftarrow TbNCase$ of the column related to the $IdCase$ attribute will be considered a T_i trace of the event log $L \mid |L| = |rows|$. Each trace T_i will have an identifier $IdCase \in TbNCase$, and will be composed of a set of events E, where the identifier of each event $e \in E$ will be represented by each different value of the $rows \in IdEvent \leftarrow TbNIdEvent$. The activity name will correspond to the value of the $rows \in Concept : name \leftarrow TbNActivityC$. The timestamp will correspond to the different values of $rows \in Time : stamp \leftarrow TbNTimeS$. The resource attribute will agree to the values of $rows \in Resource \leftarrow TbNResource$, and the input/output data attribute will belong to the values of the rows of $rows \in input/output_data \leftarrow tbnDataOI$.

3.3. Evaluation of the Event Log

3.3.1. Event Log Quality Assessment

The built event log must be accurate, available, and complete to ensure a high-quality event log, enabling success in the discovered business process model. This evaluation must be carried out before applying a process mining algorithm. The measurement of the quality of the event log will use the metrics of accuracy, consistency, completeness, and uniqueness [34,35], implemented through the event log quality plug-in proposed in [34] using the ProM tool.

3.3.2. Event Log Analysis

An analysis of the generated event logs will be performed considering statistics such as their complexity (for example, number of cases, number of events, and number of unique events), as well as the average time of cases (ATC) and average time number of events (ATE).

3.3.3. Discovery of the Business Process

The task is to discover the business process model from each generated event log. The discovered model will be represented by BPMN private process and DFG diagrams. Furthermore, the business process models discovered will be evaluated using the metrics of precision, recall, and generalization. These metrics allow to measure the quality of the discovered model based on the behavior observed within the event logs and to validate the structure of the built event log.

4. Results

The experimentation of the proposed approach is evaluated through two scenarios of relational databases. On the one hand, the construction of event records is presented from the relational database that supports an air quality information system with IoT technology, which operates autonomously in the cloud. In this scenario, three participants (systems, applications, or software components) are identified as collaborating through a collaborative business process. On the other hand, the proposed approach is validated using a relational database that stores information on the patient journey map with pneumonia symptoms who requests medical attention in a hospital. This scenario is derived from the case study presented in [36], following the database design displayed in [37]. The scenario represents the flow of the business process of patient care in different hospital areas, ending with the patient's discharge.

4.1. Requirements Identification Stage

4.1.1. Data Interpretation Analysis

The analysis of the schema database schema (DB) of the air quality monitoring system allowed us to identify and interpret the data stored in its tables. The database is made up of 12 tables distributed as follows (see Figure 2):

- Five tables store values of the data collected by the sensors that make up the air quality monitoring system.
- Three tables support the operation of various web services that manage data transfer between the system that collects them and their storage in the cloud.
- Four tables that support embedded functionalities of various software components that work collaboratively within the air quality monitoring system.

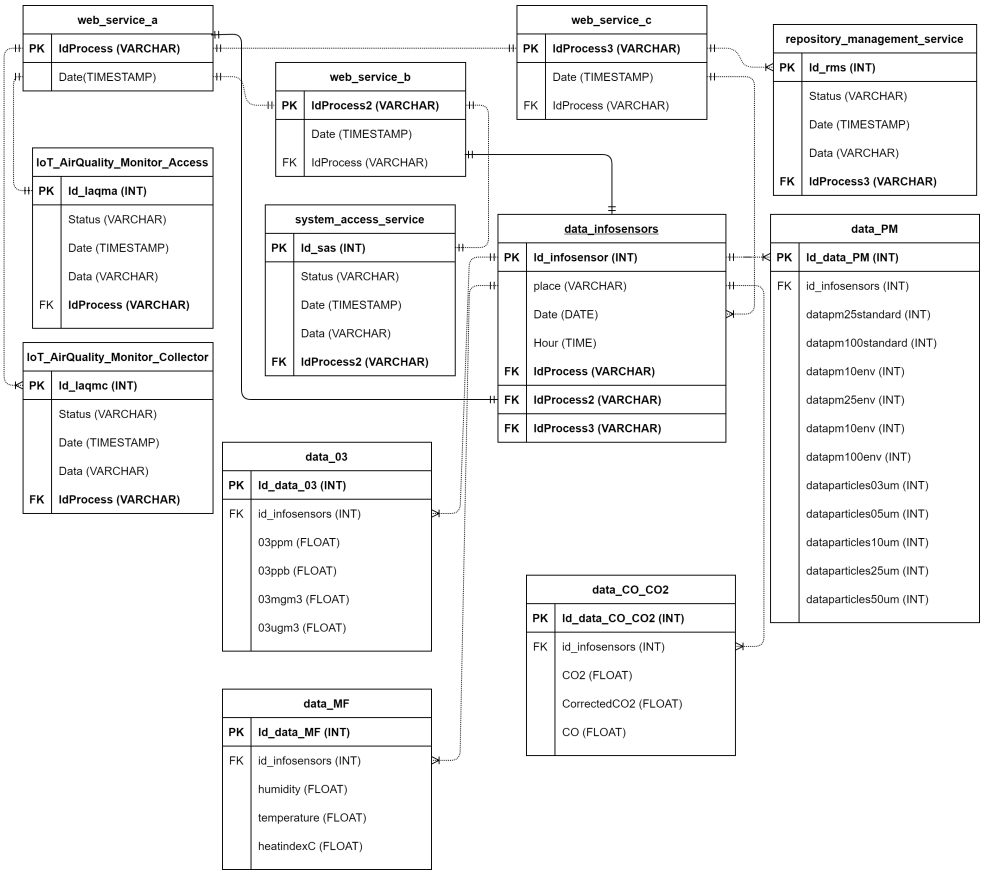


Figure 2. The entity-relationship diagram of the IoT air quality information system database.

The main table *data_infosensors* stores data on the place, date, and time of values collection. In addition, it is related to four reference tables (*data_O3*, *data_CO_CO2*, *data_PM*, and *data_MF*) that store the collected values of pollution by gases and particles, as well as meteorological factors. These tables are referenced by the foreign key (FK) *id_infosensors* with the *data_infosensors* table. This table has a relationship with the *web_service_c* table since it acts as the main table. *Web service C* has the functionality to receive, transmit, and insert the data collected by the sensors. It coordinates with a *repository management service* to validate and process the data received. These activities are deployed in the business process as service and script tasks, supplemented with exclusive gateways and returns to previous activities. Subsequently, the processed data are returned to the *Web service C* to perform the insertion in the corresponding tables. Secondary tables are related to the main table using the FK *IdProcess3*.

The *IoT_AirQuality_Monitor_Access* and *IoT_AirQuality_Monitor_Collector* tables store data from the business process flow of a single participant. At the design time of the information system, it was decided to store them in different tables due to the context of the data. The table *IoT_AirQuality_Monitor_Access* stores information on the validation of the operation of the sensors, request for IP address assignment, and access requests to the cloud storage system. These activities are done through various service tasks that process, execute, and validate data to make decisions about the business process flow through exclusive gateways. The tasks' status, date and time of execution, and the data exchanged

are stored in the *IoT_AirQuality_Monitor_Access* table. The software system access requests are coordinated by the *Web service A*, with which there is a relationship through the FK *IdProcess*. On the other hand, in the *IoT_AirQuality_Monitor_Collector* table, information about data collection, validation, and sending of the collected values to *web_service_a* is stored through a relationship between tables with the FK *IdProcess*. This part of the business process is executed by automatic service and business rule tasks that enable data collection, validation, and sending. Moreover, the actions carried out during the business process execution, the date was executed, and the resources sent or received in each completed task are saved.

Finally, in the *system_access_service* table, information is stored on the access requests to the cloud storage system accepted or rejected and on the assignment of IP to each monitoring station equipment. These actions were modeled in the business process as automatic service tasks that process and evaluate requests, generate system access agreements or rejections, and assign valid IP addresses to monitoring stations. Values of the unique identifier (primary key, PK), the status of the executed task, the date of execution, and the data exchanged with *Web service B* are stored in this table. This service supports management with the *system_access_service* table through a relationship defined in the FK *IdProcess2*.

4.1.2. Definition of Business Process Entities

From the analysis of the entity-relationship diagram of the database and the business process model, it is identified that a relationship with the *data_infosensors* table was defined in three tables (see Figure 2), through the FKs *IdProcess*, *IdProcess2*, and *IdProcess3*. The preceding supports the functionalities of accessing the cloud storage system, collecting values of pollutants and meteorological factors, transmitting the collected data, inserting the data in the corresponding columns, and validating the data insertion process. Allowing us to conclude that there are three identifiable participants in the database, coinciding with the business process model, where the *IdCase* of each participant will be the FKs *IdProcess*, *IdProcess2*, and *IdProcess3*, respectively; allowing us to form the cases for each participant.

The IoT Air Quality Monitor participant involves the *IoT_AirQuality_Monitor_Access* and *IoT_AirQuality_Monitor_Collector* tables. The Repository Management Service participant is linked to the *repository_management_service* table. The System Access Service participant involves the *system_access_service* table. In addition, it was identified that the first part of the IoT Air Quality Monitor participant and the System Access Service participant business processes are only executed when the air quality monitoring system is started or when the system reboot is activated. On the other hand, the second part of the IoT Air Quality Monitor participant (related to the *IoT_AirQuality_Monitor_Collector* table) and the Repository Management Service participant business processes execute a cyclic behavior, since they are continuously sensing data and sending them for insertion in the database.

4.2. Construction of the Event Log

4.2.1. Definition of the Event Log

The code presented in Listing 1 shows the hierarchical structure defined for the event log, derived from the analysis carried out in the previous activities of the framework. In the structure of observing the correctly nesting elements. Based on the links between the table columns and the event log attributes, the following elements are specified in the structure for the event log document.

- The *IdCase*, which allows to contain unique elements, identify and track the events executed within each case.
- The *IdEvent* of each executed event.
- The activity (*concept:name*) performed on each event.
- The date and time (*time:stamp*) that an event was executed. This attribute allows for maintaining an ordering of events.
- The *resource* (user, system, web service, or software component) that executed the activity referred in the event.

- The *participant* represents the business process executed.
- The *dataIO* allows the exchange of business documents between the participants of a collaborative business process.

Listing 1: Event log structure based on the XES standard.

```
<?xml version="1.0" encoding="UTF-8"?>
<log>
        <trace>
                <string key="IdCase" value=""/>
                <event>
                        <string key="IdEvent" value=""/>
                        <string key="concept:name" value=""/>
                        <date   key="time:stamp" value=""/>
                        <string key="dataIO" value=""/>
                        <string key="resource" value=""/>
                        <string key="participant" value=""/>
                </event>
                <event>
                    .......
                </event>
                <event>
                    .......
                </event>
        </trace>
</log>
```

4.2.2. Attribute Extraction

Case correlation. According to the analysis of the entity-relationship diagram and the values contained in the columns of the database tables, the following columns were selected to generate the *IdCase* for each participant. The *data_infosensors* table (see Figure 2) has the foreign-key *IdProcess* specified, which links to the referenced tables *IoT_AirQuality_Monitor_Access* and *IoT_AirQuality_Monitor_Collector*. Furthermore, the *data_infosensors* reference table is related to the *System_Access_Service* reference table via the *IdProcess2* foreign key. Moreover, the referenced table *Repository_Management_Service* contains is associated with the *data_infosensors* table through the foreign key *IdProcess3*. The value contained in each record of these foreign keys can be transferred to the *IdCase* attribute of the event log, allowing the creation of traces of the different business process instances for each of the three participants identified in the previous stage. The *IdCase* for the participant *IoT Air Quality Monitor* can be defined as $IdCase \in row(data_infosensors) \mid TbNCase = IdProcess$. For the *Repository Management Service* participant its *IdCase* is specified as $IdCase \in row(data_infosensors) \mid TbNCase = IdProcess3$, and for the participant *System Access Service* its *IdCase* is defined as $IdCase \in row(data_infosensors) \mid TbNCase = IdProcess2$.

Event correlation. In the database, it was identified that there is a table associated with each of the business process participants. The primary key of each table can be used as a unique event identifier (*IdEvent*), allowing access to all database records related to the participant. Furthermore, linking the activities executed by the participant through the relationship inside the table through the primary key and the relationship with other tables through foreign keys. The above will allow maintaining an alignment between the events that correspond to a case since an event can only be linked to one case. For example, in the table *IoT_AirQuality_Monitor_Access* there is the primary key *Id_Iaqma*. Then, the event mapping can be defined as $(IdEvent \leftarrow rows \in TableName \mid schema.columns(TableName) = type(PK))$.

Activity correlation. The selection of the activity name and its mapping to the *concept:name* attribute of the event log is derived from the event constructor proposal presented in [14]. In this way, a column X is selected from a table Y ($X \subset schema.columns(Y)$), and its unique values are assigned as activity name ($concept : name \leftarrow rows \in Y \mid schema.columns(Y) = X$). In the database, there is a *Status* column in each table linked to a participant that stores the task executed within the flow of the participant's business process. Unique values are extracted from these columns using the following rule ($concept : name \leftarrow unique(rows \in TableName \mid schema.columns(TableName) = "Status")$).

Time: Stamp correlation. A timestamp is stored in various database columns when a transaction is performed. In this case, the column that has a relationship with the *IdEvent* and the executed activity was chosen, since this timestamp determines the time in which an event occurred. These columns named *Date* (time:stamp datatype) were mapped to the *time:stamp* attribute in the event log, following the rule ($time : stamp \leftarrow rows \in TableName \mid schema.columns(TableName) = Date\ and\ type(Date) = timestamp$).

Resource correlation. The air quality information system was developed using an IoT paradigm, whereby the system operates independently with the ability to transfer data over the network without requiring human-to-human or human-to-computer interaction. Therefore, the resources that execute the business process tasks are web services, embedded systems in the monitoring stations, or the message delivery system contained in the communication protocol (invoked by a web service or by an embedded system). Several rules have been defined that only differ in the software component that executes the activities of that participant in the business process. For example, ($resource \leftarrow web_service_a \in schema.tables(DB)$).

Input/Output correlation. From the definition of the correlations at the cases, events, and timestamps levels, the columns containing information related to input/output data are filtered. These data are linked to the events of receiving or sending, that is, the exchange of messages between the participants of the business process; which are stored in a column called *Data*, formalized as follows: ($dataIO \leftarrow rows \in TableName \mid schema.columns(TableName) = Data$)

Table 1 shows the correlation between the selected database fields with the event log attributes, according to what was defined in the attribute extraction activity.

Table 1. Mapping between the columns associated with the IoT_AirQuality_Monitor_Access table and the event log attributes.

Field	Data Type	Event Log Attribute	Key
IdProcess	varchar	IdCase	FK
Id_Iaqma	Int	IdEvent	PK
Status	varchar	concept:name	
Date	time:stamp	time:stamp	
Data	varchar	dataIO	

4.2.3. Event Log Generated

The event log is built automatically by a software application developed for this purpose. The application implements the rules defined in the previous stage of the framework, displaying per participant the columns and tables linked to it (the application only requires read permissions on the database), as well as the attributes that were mapped to these columns. In addition, it allows you to add tables and columns available in the database to create new mappings between the columns and the attributes of the event log, following the different mappings of the previous task. The application user can confirm the predefined mappings and, where appropriate, the mappings added by the user. The queries required to extract the data and transfer these values to the corresponding attributes within the

event log are generated automatically (based on the SQL language) from the mappings accepted by the application user, according to the event log structure shown in Listing 1.

The *IoT Air Quality Monitor* participant event log is generated from obtaining all case identifiers from the *data_infosensors* table $IdCases \leftarrow unique(rows \in data_infosensors \mid schema.columns(IdProcess)$. To build each trace T, for each $IdCases_i$ the following rules are applied.

- For each $IdCases_i$, in the IoT_AirQuality_Monitor_Access (IAQMA) and IoT_AirQuality_Monitor_Collector (IAQMC) tables, all the records R that meet the following criteria are selected:$R = rows \in IAQMA \mid schema.columns(IAQMA) = IdProcess\ and\ rows(j, IdProcess) = IdCases_i) \cup rows \in IAQMC \mid schema.columns(IAQMC) = IdProcess1\ and\ rows(j, IdProcess) = IdCases_i)$.
- Since $R(j, ColumnName)$ is a record, where j indicates the position of the row and *ColumnName* indicates the column's name. Then, we can extract the value of record j from column *ColumnName*.
- The records in R represent the set of events E within the trace $E \in T_i$, with $IdCase = IdCases_i$. Then, each event $e \in E$ is given the following attributes: $et \leftarrow IdCase = IdCases_i$, $et \leftarrow IdEvent = R(j, id_Iaqma)$, $et \leftarrow concept : name = R(j, Status)$, $et \leftarrow time : stamp = R(j, Date)$, $et \leftarrow resource =$ "Web service A" , and $et \leftarrow dataIO = R(j, Data)$.

A similar procedure is performed for the Repository Management Service participant using the Repository_Management_Service Table and for the System Access Service participant using the System_Access_Service table, mapping the values contained in *IdProcess2* and *IdProcess3* to the *IdCase* attribute, respectively. It is essential to mention that the tables of these participants have the same fields in common, which allowed using the same data correlation shown in Table 1, modifying the event identifier.

Listing 2 shows the elements generated for *IdCase* = "62576aab70c34", and the events built from the execution data of the business process of the *Repository Management Service* participant. This code has been generated under the XES standard and represents an extract from the event log of the *Repository Management Service* participant. In the first <event> element, an event corresponding to *IdEvent* = "11009" is shown, which has the "Insert Data Package" activity executed on 13 April 2022 at 7:28:27 p.m. as part of the properties of this event.

Listing 2: Event log extract built for the *Repository Management Service* participant.

```
<?xml version="1.0" encoding="UTF-8" ?>
<log>
        <trace>
        <string key="IdCase" value="62576aab70c34"/>
        <event>
                <string key="IdEvent" value="11009"/>
                <string key="concept:name" value="Insert Data Package"/>
                <date   key="time:stamp" value="13-04-2022 19:28:27"/>
                <string key="dataIO" value="Insert Data Package"/>
                <string key="resource" value="Web Service C"/>
                <string key="participant" value="Repository Management Service"/>
        </event>
        <event>
                <string key="IdEvent" value="11010"/>
                <string key="concept:name" value="Process Data Package"/>
                <date   key="time:stamp" value="13-04-2022 19:28:27"/>
                <string key="dataIO" value=""/>
                <string key="resource" value="Web Service C"/>
                <string key="participant" value="Repository Management Service"/>
        </event>
        ....
        ....
        <event>
                <string key="IdEvent" value="11014"/>
                <string key="concept:name" value="Confirm Correct Insertion"/>
                <date   key="time:stamp" value="13-04-2022 19:28:27"/>
                <string key="dataIO" value="Confirm Correct Insertion"/>
                <string key="resource" value="Web Service C"/>
                <string key="participant" value="Repository Management Service"/>
        </event>
```

```
        <event>
                <string key="IdEvent" value="11015"/>
                <string key="concept:name" value="Inform␣Shutdown␣AQ␣System"/>
                <date   key="time:stamp" value="13-04-2022␣19:28:27"/>
                <string key="dataIO" value="Inform␣Shutdown␣AQ␣System"/>
                <string key="resource" value="Web␣Service␣C"/>
                <string key="participant" value="Repository␣Management␣Service"/>
        </event>
        </trace>
<log>
```

4.2.4. Evaluation

Table 2 presents the quality metrics calculated for each event log. The scores are also visualized with grades: low under 0.45, medium between 0.45 to 0.75, and high above 0.75, respectively. The accuracy measure is calculated by considering four elements: infrequent event ordering, overlapping events per resource at the trace level, future entry, and precision at the event level. All three event logs achieve a 0.92 value for the accuracy metric, which is a very acceptable performance for an event log. The completeness metric as a quality parameter is calculated as follows: at the event level through the identification if there is any time-stamp missing in the events; at the activity level, it is verified if there are missing events; at the trace level, missing activities are detected and counted, and at the log level, missing traces are detected in the event log. The value obtained in evaluating the three event logs in the completeness metric is 0.74, with a very close high quality value.

Consistency measure is examined at the activity, trace, and log levels to confirm that you have the same granularity at each level. A mixed granularity can lead to an incorrect order of activities. The evaluation of the three event records confirmed that the granularity is consistent at each level, with a very high value close to 1 (see Table 2). Uniqueness is evaluated at the activity, trace, and log levels, identifying the existence of unwanted duplicates at each level of abstraction. In the case of the IoT Air Quality Monitor event log, duplicate traces and activities were identified, causing the quality of this event log to be close to low, with a value of 0.47. Duplicate traces are identified in the Repository Management Service and System Access Service event logs, causing a medium quality grade with a value of 0.66, which is an acceptable value for the event log quality using the uniqueness metric.

Table 2. Quality metrics calculated for each of the event logs built.

Event Log	Accuracy	Completeness	Consistency	Uniqueness
IoT Air Quality Monitor	0.922 (high)	0.749 (medium)	0.984 (high)	0.472 (medium)
Repository Management Service	0.920 (high)	0.740 (medium)	0.980 (high)	0.666 (medium)
System Access Service	0.922 (high)	0.749 (medium)	0.990 (high)	0.666 (medium)

Table 3 shows the metrics of the three event logs generated using the proposed approach. The IoT Air Quality Monitor participant event log is the largest, containing 511,049 events, with 24 unique activities within 9180 cases. The Repository Management Service participant event log achieved 215,651 events with 8 unique activities. This event log contains a high event total because many interactions are performed between these two participants in the business process behavior. The System Access Service participant event log recorded 64,260 events with 7 unique activities. The lower number of events with the same total cases than the other event logs is since the business goal defined in the business process executed by the participant is focused on managing access to the cloud storage system from a request from the IoT Air Quality Monitor participant. The "ATC" column of Table 3 shows the average time (seconds) required to execute one case for each event log, e.g., in the System Access Service participant event log, the time consumed was approximately 1 s. The "ATE" column shows the average time consumed between the event execution; in the Repository Management Service event log is observed that more

time is required for the event execution (15 s). In this case, it can be considered correct due to the business goal defined for the business process that generates this event log.

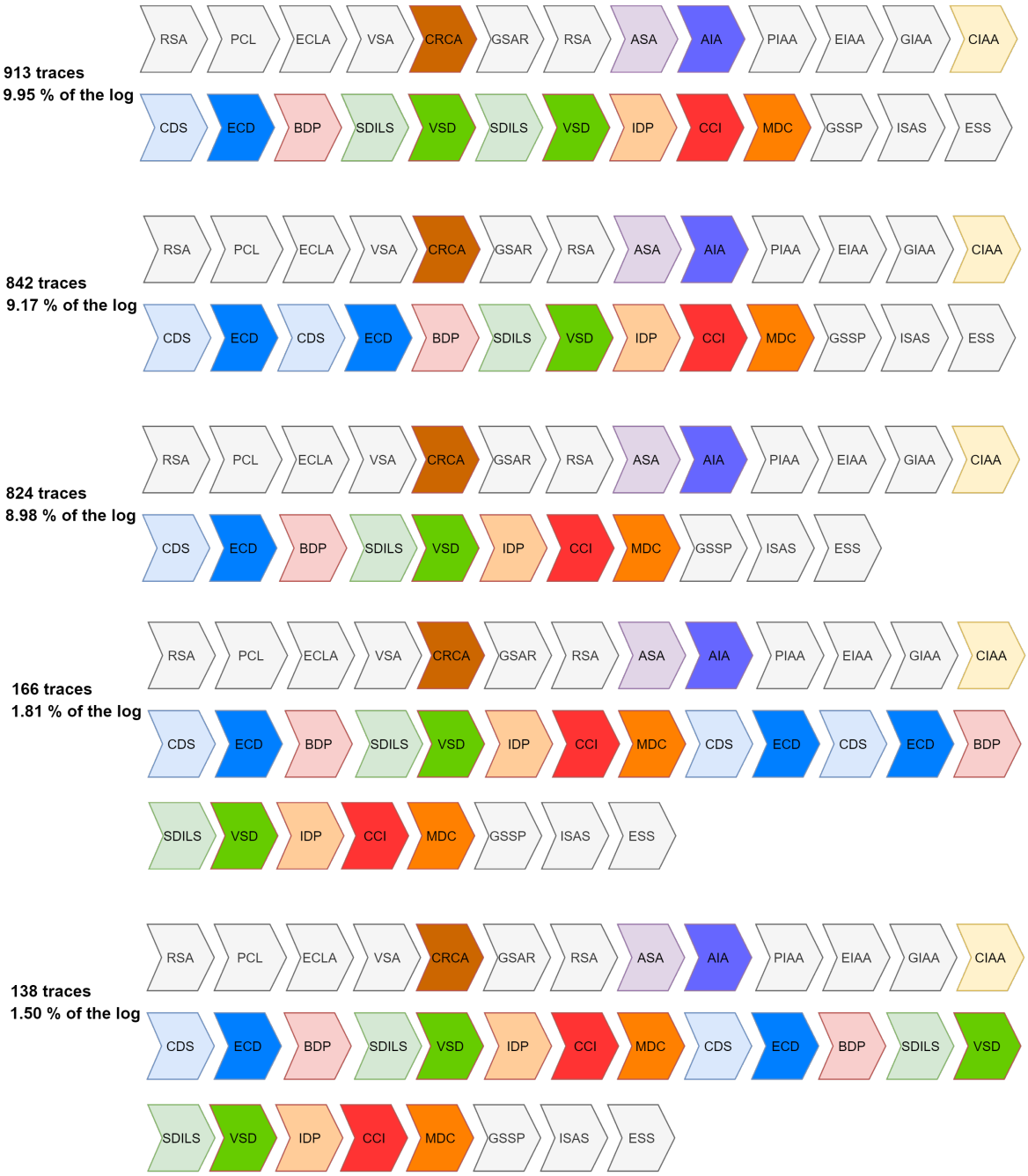
Table 3. Metrics of each event log generated.

Event Log	Cases	Events	Unique Activities	ATC	ATE
IoT Air Quality Monitor	9180	511,049	24	75	4
Repository Management Service	9180	215,651	8	75	15
System Access Service	9180	64,260	7	1	0.4

Figure 3 shows an analysis of the main process variants retrieved from the event log of the IoT Air Quality Monitor participant. Through the process variants, characteristics found in the traces of an event log can be identified [38], allowing one to know the different directly-follows relations, i.e., discovering a sequence of specific activities through the flow of the business process. In the first map of the process variant presented in Figure 3 (from top to bottom), it is observed that a sequence of SDILS-VSD activities, executed twice in a row, may correspond to a loop within the flow of the process. Compared with the second map of process variants, it is observed that the first 15 activities are similar to the two process variants. However, in the second variant, the CDS-ECD sequence activities were executed on two continuous occasions (see that there is a loop in this part of the process). The SDILS-VSD sequence of activities was only performed once, with respect to the first variant of the process. The differences between the process variants are presented in the second part of the process map, showing various loops at different stages of the business process flow. Therefore, it allows an execution pattern to be identified from the process variables.

Figure 4 shows the execution frequency of each activity per participant. In the IoT Air Quality Monitor event log, the activities "Collect data sensors", "Evaluate collected data", "Save data in local storage", and "Verify saved data" present a higher frequency percentage. On the other hand, in the Repository Management Service participant event log, the activities "Confirm IP address assignment", and "Inform shutdown AQ system" have the lowest frequency percentage of 4%. Finally, in the event log of the system access service participant, the seven activities have the same appearance rate within the record.

The discovery of the business process model is one of the core tasks in process mining. Therefore, in our proposed approach, the Split-miner [26] and DFG [39] algorithms were applied to discover the business process model for each participant, allowing us to compare and verify the completeness of the event logs generated. The DFG algorithm was implemented using the PM4PY tool. In Figure 5, the graph discovered from the *IoT Air Quality Monitor* event log using the DFG algorithm is shown (see Definition 5). This graph is composed of 24 activities. The highlighted activities represent the behavior in the business process of collecting data through the sensors. Moreover, several cycles are observed between the "collect data sensors" and "evaluate collected data" activities, as well as in the "save data in local storage" and "verify saved data" activities. In addition, there is a return immediately from the "manage data collection" to "collect data sensors" tasks. In Figure 6, the discovered business process model based on the BPMN language (see Definition 4) using the Split-miner algorithm [26] is shown. In the first part of the discovered process, a flat behavior is observed, only a sequence of activities. This sequence is because no gateways were detected in the event log that determines a path division or a loop in the process's behavior. The business process models discovered by the two algorithms coincide in their flow and behavior. The above guarantees that the built event log can be interpreted by process mining tools, validating that the structure of the event log is correct.



Activities

Request System Activation (**RSA**), Process Component List (**PCL**), Execute Component List Activation (**ECLA**), Validate Sensor Activation (**VSA**), Confirm Request Component Activation (**CRCA**), Generate System Access Request (**GSAR**), Request System Access (**RSA**), Agree System Access (**ASA**), Assign IP Address (**AIA**), Process IP Address Assignment (**PIAA**), Evaluate IP Address Assignment (**EIAA**), Generate IP Address Assignment (**GIAA**), Confirm IP Address Assignment (**CIAA**), Collect Data Sensors (**CDS**), Evaluate Collected Data (**ECD**), Build Data Package (**BDP**), Save Data into Local Storage (**SDILS**), Verify Saved Data (**VSD**), Insert Data Package (**IDP**), Confirm Correct Insertion (**CCI**), Manage Data Collection (**MDC**), Generate Shutdown System Procedure (**GSSP**), Inform Shutdown AQ System (**ISAS**), Execute Shutdown System (**ESS**).

Figure 3. Top five process variants contained in the *IoT Air Quality Monitor* event log.

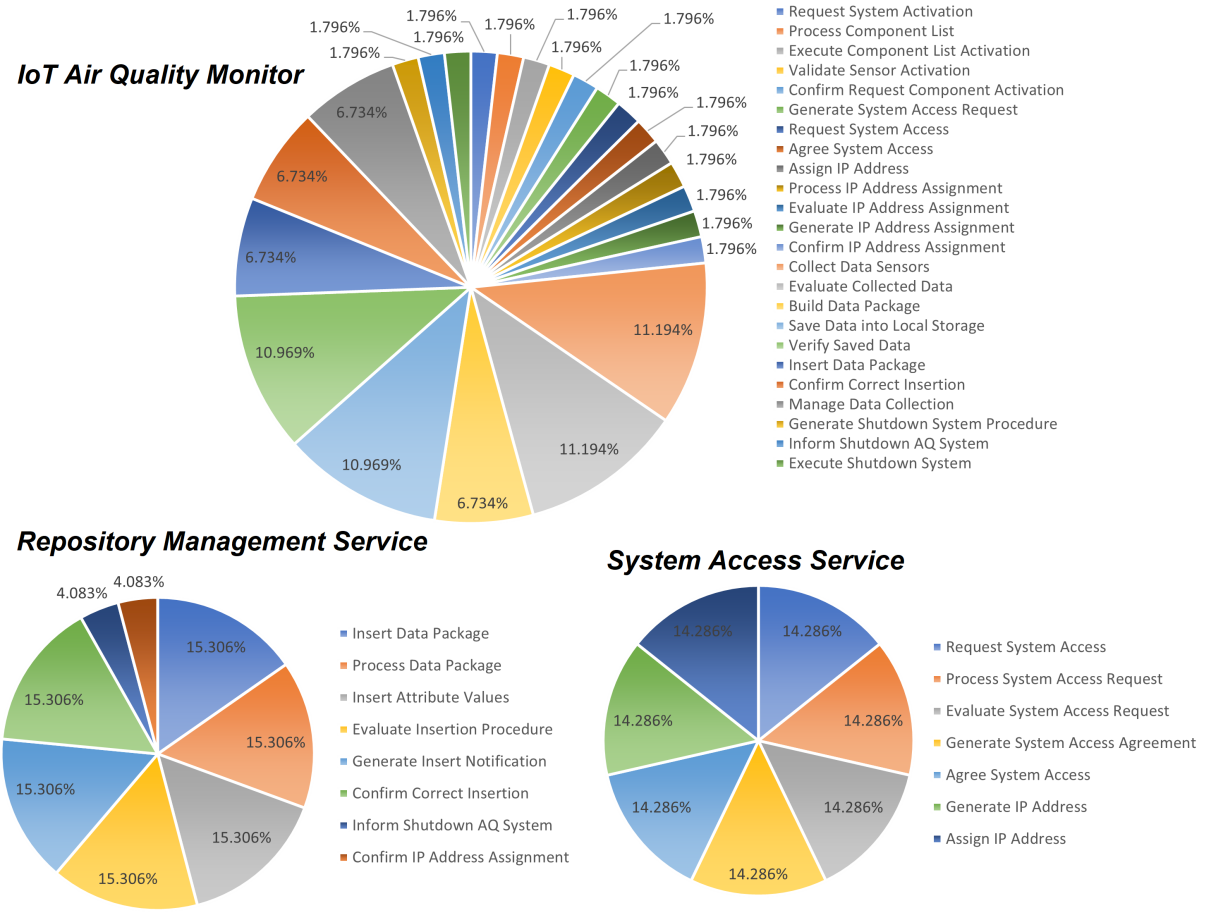


Figure 4. Frequency of each activity contained in the event logs.

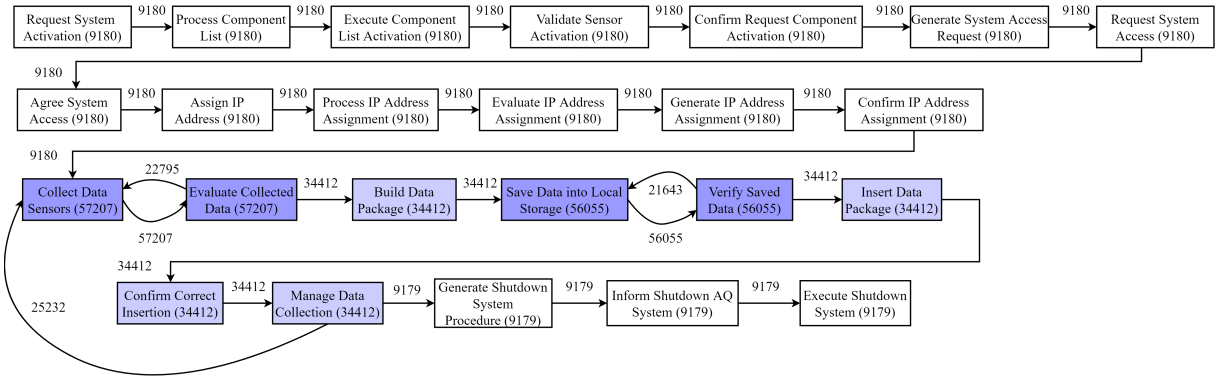


Figure 5. Process model discovered using DFG algorithm for *IoT Air Quality Monitor* participant event log.

Figure 7 shows the process model discovered from the event log built for the *Repository Management Service* participant using the DFG algorithm. In the discovered business process flow, some tasks are identified that relate to the tasks contained in the discovered process model for the *IoT Air Quality Monitor* participant, which is correct because these participants exchange messages to execute the collaborative business process. This behavior defines a cycle of execution of the activities with a path division element that determines a return to the "Insert Data Package", allowing execution to be restarted for the storage management of the values collected by the sensors of the air quality monitoring system. Figure 8 shows the BPMN process model discovered for the same participant using the

split-miner algorithm. The models discovered by the two algorithms agree on the flow and behavior of the business process.

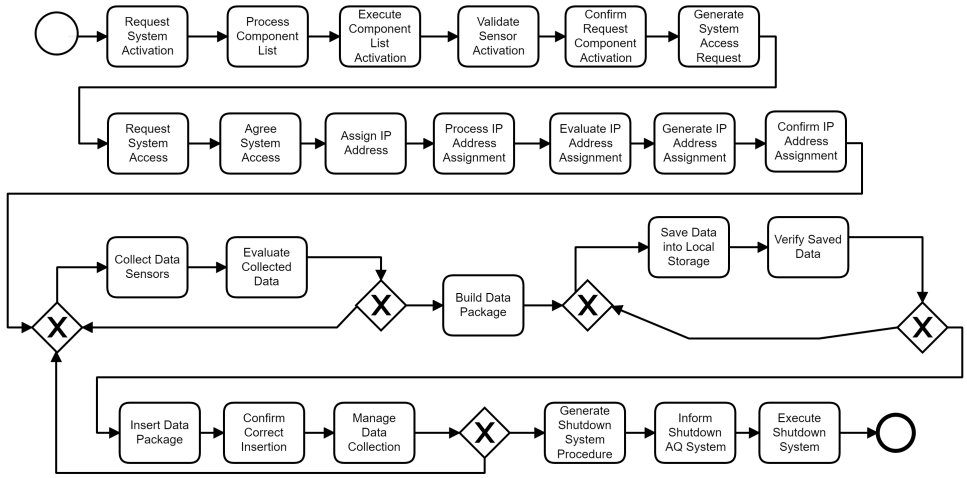


Figure 6. BPMN process model discovered using Split-miner for *IoT Air Quality Monitor* participant event log.

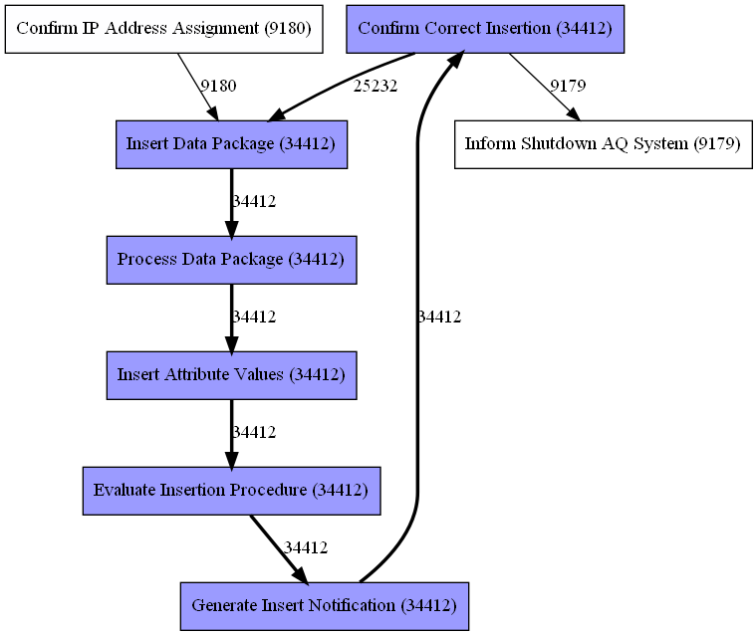


Figure 7. Process model discovered by DFG (*Repository Management Service* participant).

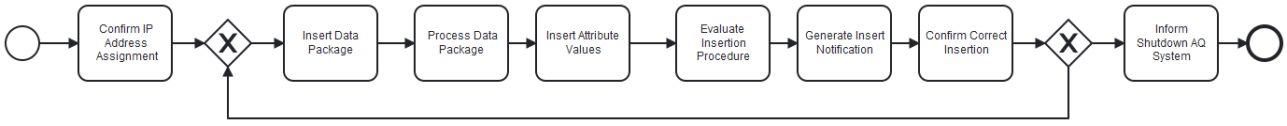


Figure 8. BPMN process model discovered using *Repository Management Service* event log.

Finally, the values achieved in the quality metrics of the business process models discovered are shown in Table 4. In the precision (Equation (3)) and recall (Equation (4)) metrics, the maximum value of 1.0 is obtained in the business process models of the three participants. On the one hand, the precision metric value indicates that each discovered model can reproduce all the behaviors within the event log. On the other hand, the value reached in the recall metric indicates that the models do not contain behaviors that are not included in the event log. In the case of the generalization (Equation (5)) metric,

the evaluated process models reached the values of 0.991, 0.993, and 0.980 for the *IoT Air Quality Monitor*, *Repository Management Service*, and *System Access Service* participants, respectively. The value of the generalization metric indicates that the process flow and the behaviors it can reproduce are frequently used.

Table 4. Results of the quality metrics applied in the business process models discovered.

Process Model	Precision	Recall	Generalization
IoT Air Quality Monitor	1.0	1.0	0.991
Repository Management Service	1.0	1.0	0.993
System Access Service	1.0	1.0	0.980

4.3. Patient Treatment Process Scenario

In this section, we present only the evaluation of the built event log since the previous stages of the framework have the same foundation as described in the earlier case study. Table 5 presents the quality metrics of the event log built for the patient treatment journey map. The event log's consistency obtains a value in the lower range of the medium degree (0.450) due to the fact that a mixed granularity was identified in the level of activity abstraction, affecting the evaluation of the event log. Likewise, a mixed granularity at the log abstraction level was determined, caused by different types of formats in the event log instances. The value obtained in the uniqueness metric is since, at the trace level, several traces show the same timestamp, which may be an error in the insertion of the tuples in the relational database. The traces probably occurred in sequence but were stored with the same timestamp. On the other hand, the values achieved in the accuracy and completeness metrics are very acceptable, confirming the quality of the event log generated.

Table 5. Quality metrics calculated for patient admission event log built.

Event Log	Accuracy	Completeness	Consistency	Uniqueness
Patient Treatment Process	0.834 (high)	0.747 (medium)	0.450 (medium)	0.555 (medium)

Figure 9 shows the main variants retrieved from the event log of the patient's treatment process. The analysis of process variants tries to identify the differences in the traces representing the execution of the process derived from the same business process [40,41]. In the third map of the process variants, it is observed that the sequence of HO-LE activities is only executed once. However, in the discovered process diagrams, it is identified that the LE activity is of the loop type; because this behavior is infrequent, it is not represented within the main variants of the process. In the fourth variant (from top to bottom) presented in Figure 9, the sequence of LE-MI activities was executed with a return to the LE activity on continuous occasions, before and after the execution of the MI activity, which allows identifying a loop in this part of the process flow.

Figure 10 shows the graph discovered using the DFG algorithm from the event log built from the process identified for the patient treatment with pneumonia symptoms. At the beginning of the business process, three paths are distinguished to start the treatment and hospitalization of the patient. After admission to the hospital, several cycles related to the laboratory exam activity are recognized; for example, between the activities laboratory exam and medical images, as well as between the activities laboratory exam and procedure. Likewise, a low-frequency sequence is distinguished in the process flow between laboratory exams, medical images, medical ward activities, and returning to the laboratory exam activity before the patient's possible discharge. In addition, several tasks have loop-like behavior in the generated event log but are discarded by the DFG algorithm due to their low frequency in the event log traces.

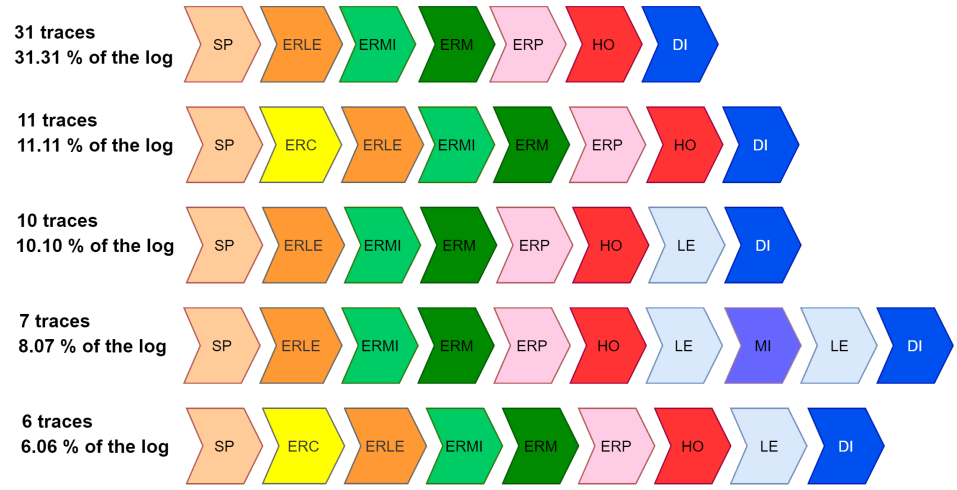


Figure 9. Top 5 process variants contained in the *patient treatment* event log.

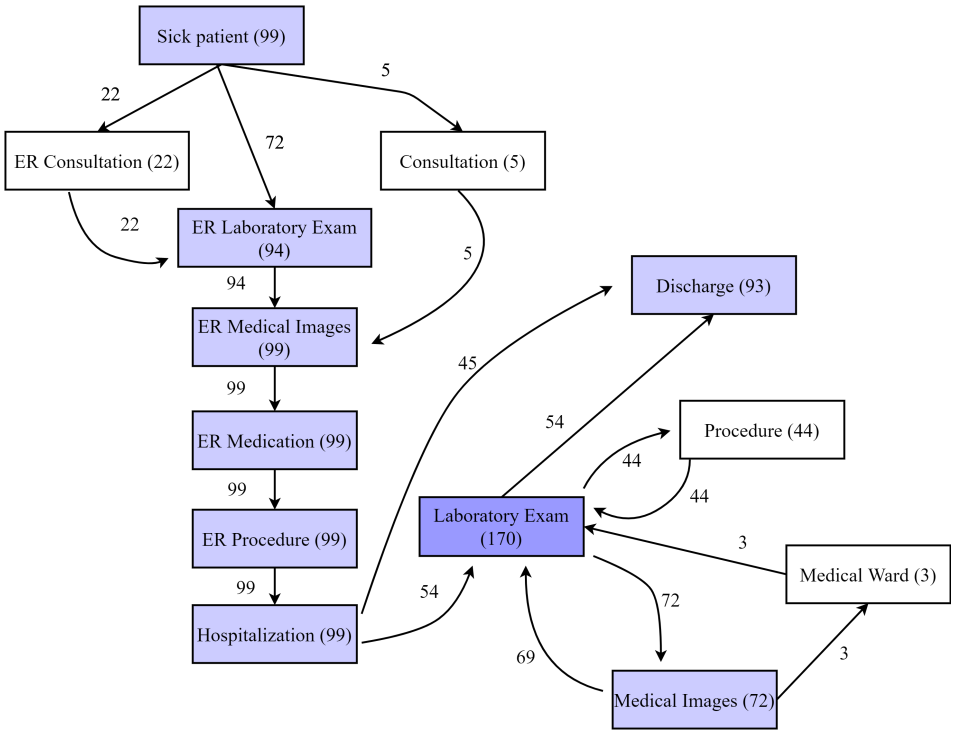


Figure 10. Process model discovered using DFG algorithm for *patient treatment* event log.

Figure 11 shows the business process model based on the BPMN language, which was discovered using the split-miner algorithm. The business process flow discovered by the two algorithms is consistent in its behavior, tasks, and identified decision points. However, the split-miner algorithm distinguishes several loop-type tasks (Laboratory Exam, Medical Imagesm and Procedure) in the event log traces.

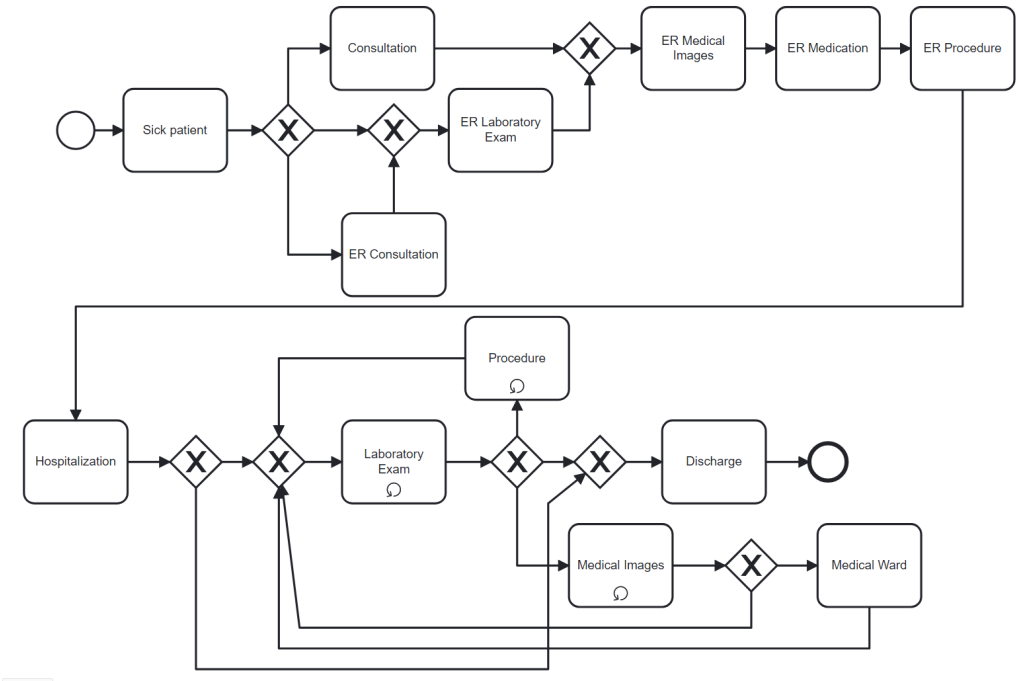


Figure 11. BPMN process model discovered using Split-miner for *patient treatment* participant.

Table 6 shows the calculated values for the quality metrics of the uncovered patient treatment process model. The value of 0.771 in the generalization metric indicates to what extent the discovered model can reproduce a future process behavior. Even when the accuracy of the process model is high (1.0), the generalization metric tells us that there may be difficulties when deciding on the path to follow at a decision point. On the other hand, the value 0.825 in the recall metric guarantees that the discovered model can accurately reproduce the cases recorded in the event log.

Table 6. Results of the quality metrics for the patient treatment model discovered.

Process Model	Precision	Recall	Generalization
Patient Treatment Process	1.0	0.825	0.771

5. Conclusions

In this study, we proposed a framework to guide the construction of an event log from a relational database. The event log generated is based on the XES standard for use with process mining algorithms. The event log is constructed semi-automatically because, in the first stage, an analysis of the relational database tables and their link with the business process activities is carried out. In this analysis, experts are required at a technical and business level, with knowledge of the database structure and semantics of the data stored and its relationship with the business process. This part is fundamental so that a record of quality events is automatically built in the next stage.

Correlation methods were defined at different levels of abstraction to extract the attribute values from the database tables, that is, at the case level, event, and event properties. For this, a set of rules were defined that allowed data extraction and converting them into the attribute values of the event log. The time consumed at the case and event level was analyzed from the event logs. In addition, process variants were retrieved for each event log, identifying activity sequence patterns throughout the business process flow.

Finally, the quality of the constructed event logs was evaluated using accuracy, completeness, consistency, and uniqueness metrics, obtaining acceptable values in the two case studies presented. In addition, from the generated event logs, business process discovery techniques were implemented using DFG and split-miner algorithms. The discovered

process models were measured to determine their quality, obtaining very acceptable values in the accuracy, recall, and generalization metrics.

Author Contributions: Conceptualization, J.D.H.-R. and E.T.-L.; methodology, J.D.H.-R. and E.T.-L.; software, J.D.H.-R.; validation, J.D.H.-R., E.T.-L., U.M.R.-A. and B.A.M.-H.; formal analysis, J.D.H.-R.; investigation, J.D.H.-R., E.T.-L., U.M.R.-A. and B.A.M.-H.; resources, E.T.-L.; data curation, J.D.H.-R. and U.M.R.-A.; writing—original draft preparation, J.D.H.-R. and E.T.-L.; writing—review and editing, J.D.H.-R., E.T.-L., U.M.R.-A. and B.A.M.-H.; visualization, J.D.H.-R.; funding acquisition, E.T.-L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by the Consejo Nacional de Ciencia y Tecnología (CONACYT) under grant number 719519.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors are grateful to the Autonomous University of Tamaulipas, Mexico for supporting this work.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
XES	eXtensible Event Stream
DFG	directly-follows graph
BPMN	business process model and notation
ATC	Average time of cases
ATE	Average time of events
DB	relational database
FK	foreign key
PK	primary key

References

1. van der Aalst, W.M.P. Process Mining: A 360 Degree Overview. In *Process Mining Handbook*; van der Aalst, W.M.P., Carmona, J., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 3–34. [CrossRef]
2. Ghasemi, M.; Amyot, D. From event logs to goals: A systematic literature review of goal-oriented process mining. *Requir. Eng.* **2020**, *25*, 67–93. [CrossRef]
3. Camargo, M.; Dumas, M.; González-Rojas, O. Discovering generative models from event logs: Data-driven simulation vs. deep learning. *PeerJ Comput. Sci.* **2021**, *7*, e577. [CrossRef]
4. van der Aalst, W.M.P. *Process Mining*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2016. [CrossRef]
5. Ponce de León, H.; Nardelli, L.; Carmona, J.; vanden Broucke, S.K. Incorporating negative information to process discovery of complex systems. *Inf. Sci.* **2018**, *422*, 480–496. [CrossRef]
6. Lee, W.L.J.; Verbeek, H.; Munoz-Gama, J.; van der Aalst, W.M.; Sepúlveda, M. Recomposing conformance: Closing the circle on decomposed alignment-based conformance checking in process mining. *Inf. Sci.* **2018**, *466*, 55–91. [CrossRef]
7. Munoz-Gama, J.; Martin, N.; Fernandez-Llatas, C.; Johnson, O.A.; Sepúlveda, M.; Helm, E.; Galvez-Yanjari, V.; Rojas, E.; Martinez-Millana, A.; Aloini, D.; et al. Process mining for healthcare: Characteristics and challenges. *J. Biomed. Inform.* **2022**, *127*, 103994. [CrossRef] [PubMed]
8. Diba, K.; Batoulis, K.; Weidlich, M.; Weske, M. Extraction, correlation, and abstraction of event data for process mining. *WIREs Data Min. Knowl. Discov.* **2020**, *10*, e1346. [CrossRef]
9. Jans, M.; Soffer, P.; Jouck, T. Building a valuable event log for process mining: An experimental exploration of a guided process. *Enterp. Inf. Syst.* **2019**, *13*, 601–630. [CrossRef]
10. Kampik, T.; Weske, M. Event Log Generation: An Industry Perspective. In *Enterprise , Business-Process and Information Systems Modeling*; Augusto, A., Gill, A., Bork, D., Nurcan, S., Reinhartz-Berger, I., Schmidt, R., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 123–136. [CrossRef]
11. Sim, S.; Sutrisnowati, R.A.; Won, S.; Lee, S.; Bae, H. Automatic Conversion of Event Data to Event Logs Using CNN and Event Density Embedding. *IEEE Access* **2022**, *10*, 15994–16009. [CrossRef]

12. IEEE Task Force on Process Mining. IEEE 1849-2016 XES Standard Definition. 2016. Available online: https://www.xes-standard.org/ (accessed on 17 March 2022).
13. Stein Dani, V.; Leopold, H.; van der Werf, J.M.E.; Lu, X.; Beerepoot, I.; Koorn, J.J.; Reijers, H.A. Towards Understanding the Role of the Human in Event Log Extraction. In Proceedings of the International Conference on Business Process Management, Rome, Italy, 6–10 September 2021; Springer: New York, NY, USA, 2021; pp. 86–98. [CrossRef]
14. Andrews, R.; van Dun, C.; Wynn, M.; Kratsch, W.; Röglinger, M.; ter Hofstede, A. Quality-informed semi-automated event log generation for process mining. *Decis. Support Syst.* **2020**, *132*, 113265. [CrossRef]
15. Kecht, C.; Egger, A.; Kratsch, W.; Röglinger, M. Event Log Construction from Customer Service Conversations Using Natural Language Inference. In Proceedings of the 2021 3rd International Conference on Process Mining (ICPM), Eindhoven, The Netherlands, 31 October–4 November 2021; pp. 144–151. [CrossRef]
16. Remy, S.; Pufahl, L.; Sachs, J.P.; Böttinger, E.; Weske, M. Event Log Generation in a Health System: A Case Study. In *Business Process Management*; Fahland, D., Ghidini, C., Becker, J., Dumas, M., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 505–522. [CrossRef]
17. de Murillas, E.G.L.; van der Aalst, W.M.P.; Reijers, H.A. Process Mining on Databases: Unearthing Historical Data from Redo Logs. In *Business Process Management*; Motahari-Nezhad, H.R., Recker, J., Weidlich, M., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 367–385.
18. Bano, D.; Lichtenstein, T.; Klessascheck, F.; Weske, M. Database-Less Extraction of Event Logs from Redo Logs. *Bus. Inf. Syst.* **2021**, *1*, 73–82. [CrossRef]
19. Mühlberger, R.; Bachhofner, S.; Di Ciccio, C.; García-Bañuelos, L.; López-Pintado, O. Extracting Event Logs for Process Mining from Data Stored on the Blockchain. In *Business Process Management Workshops*; Di Francescomarino, C., Dijkman, R., Zdun, U., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 690–703. [CrossRef]
20. Bertrand, Y.; De Weerdt, J.; Serral, E. A Bridging Model for Process Mining and IoT. In *Process Mining Workshops*; Munoz-Gama, J., Lu, X., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 98–110. [CrossRef]
21. Leno, V.; Polyvyanyy, A.; Dumas, M.; Rosa, M.L.; Maggi, F.M. Robotic Process Mining: Vision and Challenges. *Bus. Inf. Syst. Eng.* **2021**, *63*, 301–314. [CrossRef]
22. Ammirato, S.; Sofo, F.; Felicetti, A.M.; Raso, C. A methodology to support the adoption of IoT innovation and its application to the Italian bank branch security context. *Eur. J. Innov. Manag.* **2019**, *22*, 146–174. [CrossRef]
23. Zancul, E.d.S.; Takey, S.M.; Bezerra Barquet, A.P.; Kuwabara, L.H.; Cauchick Miguel, P.A.; Rozenfeld, H. Business process support for IoT based product-service systems (PSS). *Bus. Process Manag. J.* **2016**, *22*, 305–323. [CrossRef]
24. Janiesch, C.; Koschmider, A.; Mecella, M.; Weber, B.; Burattin, A.; Di Ciccio, C.; Fortino, G.; Gal, A.; Kannengiesser, U.; Leotta, F.; et al. The Internet of Things Meets Business Process Management: A Manifesto. *IEEE Syst. Man Cybern. Mag.* **2020**, *6*, 34–44. [CrossRef]
25. Del Giudice, M. Discovering the Internet of Things (IoT) within the business process management: A literature review on technological revitalization. *Bus. Process Manag. J.* **2016**, *22*, 263–270. [CrossRef]
26. Augusto, A.; Conforti, R.; Dumas, M.; Rosa, M.L.; Polyvyanyy, A. Split miner: Automated discovery of accurate and simple business process models from event logs. *Knowl. Inf. Syst.* **2019**, *59*, 251–284. [CrossRef]
27. Leemans, S.J.; Poppe, E.; Wynn, M.T. Directly Follows-Based Process Mining: Exploration & a Case Study. In Proceedings of the 2019 International Conference on Process Mining (ICPM), Aachen, Germany, 24–26 June 2019; pp. 25–32. [CrossRef]
28. Buijs, J.C.A.M. Flexible Evolutionary Algorithms for Mining Structured Process Models. Ph.D. Thesis, Technische Universiteit Eindhoven, Eindhoven, The Netherlands, 2014. [CrossRef]
29. Janssenswillen, G.; Donders, N.; Jouck, T.; Depaire, B. A comparative study of existing quality measures for process discovery. *Inf. Syst.* **2017**, *71*, 1–15. [CrossRef]
30. Syring, A.F.; Tax, N.; van der Aalst, W.M.P., Evaluating Conformance Measures in Process Mining Using Conformance Propositions. In *Transactions on Petri Nets and Other Models of Concurrency XIV*; Koutny, M., Pomello, L., Kristensen, L.M., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; pp. 192–221. [CrossRef]
31. van Dongen, B.F.; Shabani, S. Relational XES: Data Management for Process Mining. In Proceedings of the CAiSE Forum, Stockholm, Sweden, 8–9 June 2015; Volume 2015, pp. 169–176.
32. Schönig, S.; Rogge-Solti, A.; Cabanillas, C.; Jablonski, S.; Mendling, J. Efficient and Customisable Declarative Process Mining with SQL. In *Advanced Information Systems Engineering*; Nurcan, S., Soffer, P., Bajec, M., Eder, J., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 290–305.
33. Schönig, S.; Di Ciccio, C.; Maggi, F.M.; Mendling, J. Discovery of Multi-perspective Declarative Process Models. In *Service-Oriented Computing*; Sheng, Q.Z., Stroulia, E., Tata, S., Bhiri, S., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 87–103.
34. Fischer, D.A.; Goel, K.; Andrews, R.; van Dun, C.G.J.; Wynn, M.T.; Röglinger, M. Enhancing Event Log Quality: Detecting and Quantifying Timestamp Imperfections. In *Business Process Management*; Fahland, D., Ghidini, C., Becker, J., Dumas, M., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 309–326.
35. Kherbouche, M.O.; Laga, N.; Masse, P.A. Towards a better assessment of event logs quality. In Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece, 6–9 December 2016; pp. 1–8. [CrossRef]

36. Arias, M.; Rojas, E.; Aguirre, S.; Cornejo, F.; Munoz-Gama, J.; Sepúlveda, M.; Capurro, D. Mapping the Patient's Journey in Healthcare through Process Mining. *Int. J. Environ. Res. Public Health* **2020**, *17*, 6586. [CrossRef]
37. Johnson, A.E.; Pollard, T.J.; Shen, L.; Lehman, L.w.H.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Celi, L.A.; Mark, R.G. MIMIC-III, a freely accessible critical care database. *Sci. Data* **2016**, *3*, 1–9. [CrossRef]
38. Nguyen, H.; Dumas, M.; La Rosa, M.; ter Hofstede, A.H.M. Multi-perspective Comparison of Business Process Variants Based on Event Logs. In *Conceptual Modeling*; Trujillo, J.C., Davis, K.C., Du, X., Li, Z., Ling, T.W., Li, G., Lee, M.L., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 449–459. [CrossRef]
39. van der Aalst, W.M. A practitioner's guide to process mining: Limitations of the directly-follows graph. *Procedia Comput. Sci.* **2019**, *164*, 321–328. [CrossRef]
40. Taymouri, F.; Rosa, M.L.; Dumas, M.; Maggi, F.M. Business process variant analysis: Survey and classification. *Knowl.-Based Syst.* **2021**, *211*, 106557. [CrossRef]
41. Cremerius, J.; Weske, M. Data-Enhanced Process Models in Process Mining. *arXiv* **2021**, arXiv:2107.00565.