TRANSACTIONS ON DATA PRIVACY 16 (2023) 123–151

Identifying the Context of Data Usage to Diagnose Privacy Issues through Process Mining

Azadeh Sadat Mozafari Mehr, Renata M. de Carvalho, Boudewijn van Dongen

Department of Mathematics and Computer Science, Eindhoven University of Technology,

Eindhoven, The Netherlands.

E-mail: `a.s.mozafari.mehr@tue.nl`, `r.carvalho@tue.nl`, `b.f.v.dongen@tue.nl`

Received 18 January 2022; received in revised form 28 March 2023; accepted 14 April 2023

Abstract. In recent years, data privacy issues are increasingly concerned by organisations and governments. Organisations often define a set of rules as privacy policies for protecting sensitive data of their business. Regulations like the European General Data Protection Regulation (GDPR) added another layer of importance to data security emphasizing personal data protection, making it not only a business requirement but also a legal requirement. Existing access control mechanisms are not sufficient for data protection. They are only preventive and cannot guarantee that data is accessed for the intended purposes. This paper presents the underlying theory of a novel approach for multi-perspective conformance checking which considers the process control-flow, data and privacy perspectives simultaneously. In addition to detecting deviations in each perspective, the approach is able to detect hidden deviations where non-conformity relates to either a combination of two or all three aspects of a business process. Moreover, by reconciling the process, data and privacy aspects, it can detect spurious data access and identify privacy infringements where data have been processed for unclear or secondary purposes by an authorised role. The approach has been implemented in the open source ProM framework and was evaluated through controlled experiments using synthetic and real logs.

Keywords. Privacy, Data privacy, Multi-perspective Analysis, Conformance Checking, Process Mining

1 Introduction

Regulations like GDPR[1] and HIPAA[2] brought significant changes to the privacy landscape, imposing challenges to organisations on how to handle personal data. Translating the concept of the regulations to a practical implementation is not easy as it relates to different aspects of a business process. Take as an example a new privacy rule that denotes "who

[1]https://gdpr-info.eu/
[2]https://www.hhs.gov/hipaa/

can access data for which purpose". Such privacy rule is closely related to three different perspectives: i) the control-flow perspective, or the tasks being executed; ii) the data perspective, or the flow and processing of information; and iii) the resource or privacy perspective, or the legitimate role allocation. However, the use of access control as it has been adopted so far is no longer enough. Such access control systems regulate only who may carry out which data, and do not check for which purpose data are processed after the access to data is granted [1].

Furthermore, an additional threat is introduced, as it is well documented in the literature that real process behavior often deviates from the expected process which can open the way to fraudulent behaviour or performance issues [2, 3]. In this context, organisations must audit their process execution considering both the conformance of their business rules and data protection rules. In particular, they should investigate multi-perspective process constraints that need to be satisfied for the process to be considered healthy and aligned with the business goals.

So far, various conformance checking techniques have been proposed to pinpoint discrepancies between modeled and observed behavior [4, 5, 6, 7, 8]. Nonetheless, such techniques either focus on the control-flow perspective and completely abstract from the data, resource and other perspectives, or they consider other perspectives as adjacent [9, 10, 11, 3]. As discussed, this is no longer sufficient as it does not provide comprehensive diagnostics about the context where data is being used. Hence, the aforementioned techniques miss some important violations related mainly to data or privacy aspects.

In this extended paper, the underlying theory of the proposed multi-perspective conformance checking in [12] is throughout examined and a complete derivation of its inputs, method to generate the synchronous product, to compute alignment, and how to identify the deviations are provided. This approach considers all perspectives together, without prioritizing any of them. Therefore, this approach is able to identify intra-layer violations – violations within only one layer –, and inter-layer violations – either between two out of the three layers, or involving all of them. More specifically, the proposed approach advances the state-of-the-art by being able to: i) detect spurious data access and identify privacy infringements where data have been processed for unclear or secondary purposes by an authorised role; and ii) enable the identification of a larger range of hidden deviations between different perspectives, providing an accurate diagnostics of deviations.

As a proof of concept, we implemented and tested our approach over synthetic logs generated from simulation of a real-life healthcare process. We also applied the developed tool to a real data set for analyzing a lead management process in car dealership industry in order to gain insights from different aspects of the process. Accordingly, this paper also provides an extended analysis of the design and functionalities of the developed tool, along with a extended discussion of the new violations and insights that might be gathered when applying this approach to real-life data sets.

This article is organised as follows. Section 2 introduces a running example along with some scenarios to discuss the motivation of this work. Preliminaries are presented in section 3. Section 4 illustrates our approach. Here we describe how the input data is represented and linked, how deviations are detected and what such deviations mean. It is followed by section 5, which describes the tool support. Section 6 presents the results obtained by a set of experiments, and section 7 discusses related work in more details. Finally, section 8 concludes the paper and provides directions for future works.

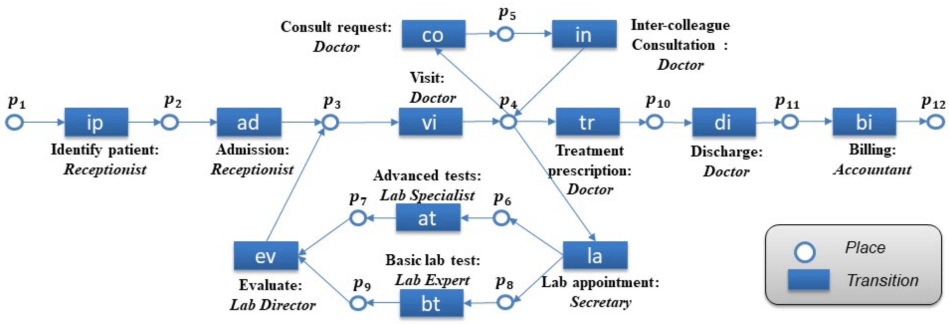


Figure 1: An example of healthcare treatment process in Petri net notation (adapted from [11])

2 Motivating Example

Consider as a running example, a healthcare treatment process derived from Alizadeh *et.al.* [11]. Fig. 1 shows the process in Petri net notation. The process starts with the identification of the patient (ip) and admission (ad) by the receptionist. Next, the patient is visited by a doctor (vi). The doctor might request a basic lab test (bt) and advanced tests such as MRI scans (at), for which the secretary makes an appointment for the patient (la). After a lab expert and a lab specialist performs the tests, a lab director evaluates the results (ev). Based on the evaluation, the treating doctor may request inter-colleague consultation ((co) followed by (in)), request more lab tests, or prescribe a treatment plan (tr). Finally, the patient is discharged by the doctor (di) and a bill is created and sent to the patient's insurance company by an accountant (bi). In this process, certain data operations on specific data fields are required to be performed during each activity. Table 1, presents these data operations. For instance, during performing activity "Admission" patient information such as identity, name and patient ID are checked by executing data operation Read(ID, PatientID, Name) and an id will be created by executing data operation Create(AdmissionID).

An execution example of this process is depicted in Fig. 2. This figure shows observed behavior from three perspectives which can be extracted from the recorded behavior in the process and data logs. For each activity, a start event and a complete event are expected. Whenever they both occur and are performed by the same resource, they are linked as a yellow rectangle as shown in Figure 2(a). The sequence of yellow triangles in Figure 2(b) shows a data trace consisting of nineteen data events. The events in the process trace and data trace record information regarding the process instance or case, the corresponding activity and data operation, the time of the execution, and the actor who executed the activity or data operation, separately. Each hexagon presents the role of the actor under whose name the event is registered in the system.

Below, we present some scenarios to motivate the need for investigating data and/or privacy compliance in addition to control-flow conformance to detect hidden deviations:

Scenario 1: According to the process model in Fig. 1 and the data model in Table 1, different roles like doctors, lab experts and director are allowed to access sensitive data of the patients, such as prescriptions, medical histories and test results. A curious actor may exploit this privilege in order to use the information of patients for personal or financial gain. This scenario shows a violation of the patient's privacy. Standard access control is not sufficient for data protection. In this mechanism access is independent of context. It is only preventive and more critically, it does not monitor for which purpose data are processed

Table 1: Data model of treatment process. R:Read, C:Create [12]

Activity	Data Operations
Identify patient (ip)	R(ID)

Admission (ad)	R(ID,PatientID,Name)
	C(AdmissionID)

Visit (vi)	R(AdmissionID,PatientID,MedicalHistoryID)
	C(VisitID,PrescriptionID)

Lab appointment (la)	R(AdmissionID,PatientID C(LabAppointment)

Basic lab test (bt)	R(AdmissionID,PatientID,PrescriptionID)
	C(BLabPID)

Advanced tests (at)	R(AdmissionID,PatientID,PrescriptionID)
	C(ALabPID)

Evaluate (ev)	R(AdmissionID,PrescriptionID, BLabPID,ALabPID) C(TestResultID)

Consult request (co)	R(AdmissionID,PatientID)
	C(ConsAppointment)

Inter-colleague consultation (in)	R(AdmissionID,PatientID,VisitID, PrescriptionID,TestResultID,MedicalHistoryID) C(VisitID,CPrescriptionID)

Treatment prescription (tr)	R(AdmissionID,VisitID,MedicalHistoryID)
	C(TreatmentPlan)

Discharge (di)	R(AdmissionID,PatientID)
	C(Confirmation)

Billing (bi)	R(AdmissionID,PatientID,PaymentID)
	C(PaymentReceipt)

after access to data has been granted [1]. This example demonstrates that privacy rules for data protection should be considered and investigated in both process and data layers together to find whether data is accessed for the intended purposes.

Scenario 2: Instead of a lab specialist, a nurse takes a blood sample from the patient. The occurrence of this activity along with its data operations are permitted based on both data model and the control-flow of the process, however from the privacy perspective, the nurse is supposed to undertake this activity. Since the nurse is not expert in performing the task, any mistake in taking the sample could impact the test results. Consequently, this may lead to repeating the test or a wrong treatment plan which affects the care quality. In this scenario, all three perspectives of process, data, and privacy must be evaluated concurrently in order to detect the hidden deviation.

Scenario 3[11]: Doctors are supposed to add a prescription or treatment plan to the patient's medical history during each visit. A doctor may negligently forget to update patient's medical history. As a result of such missing data operation, other doctors may prescribe an incompatible drugs to the patient. This may have negative impact on care quality or result in major health consequences for the patient. This scenario implies that it is important to check executed behaviour in both process and data layer. In this scenario, from control-flow perspective there is no violation while in the data perspective there is a missing data operation violation.

Scenario 4[11]: A doctor accesses patient information for providing medical treatment, but later uses this information to conduct a clinical trial (ct). In this case of secondary usage of data, the doctor performs a clinical trial activity that does not contribute to the completion of the treatment process as defined in Fig 1. During performing the mentioned activity, the doctor accesses medical history of the patient for the purpose other than the primary purpose of the process.

Scenario 5: The lab director may evaluate the test results without receiving advanced test results. This may result in increasing patient referral to the lab as well as additional costs to the patient's insurance company.

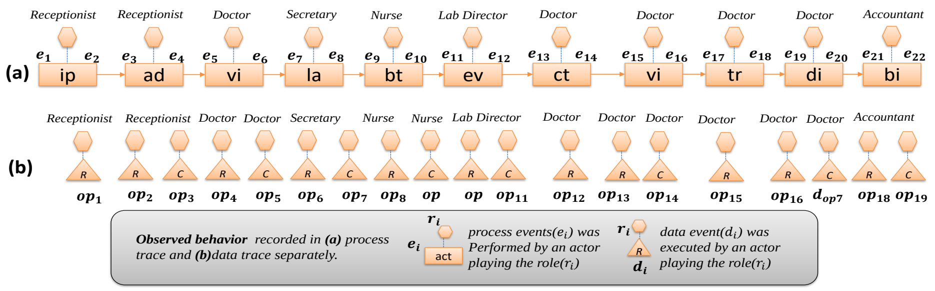

Receptionist
Receptionist
Doctor
Secretary
Nurse
Lab Director
Doctor
Doctor
Doctor
Doctor
Accountant
e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8 e_9 e_{10} e_{11} e_{12} e_{13} e_{14} e_{15} e_{16} e_{17} e_{18} e_{19} e_{20} e_{21} e_{22}
(a)
ip
ad
vi
la
bt
ev
ct
vi
tr
di
bi
Receptionist
Receptionist
Doctor
Doctor
Secretary
Nurse
Nurse
Lab Director
Doctor
Doctor
Doctor
Doctor
Doctor
Doctor
Accountant
(b)
R C R C R C R C R C R C R R C R C
op_1 op_2 op_3 op_4 op_5 op_6 op_7 op_8 op op op_{11} op_{12} op_{13} op_{14} op_{15} op_{16} d_{op7} op_{18} op_{19}
Observed behavior recorded in **(a)** process trace and **(b)**data trace separately.
r_i
e_i
act
process events(e_i) was Performed by an actor playing the role(r_i)
r_i
R
d_i
data event(d_i) was executed by an actor playing the role(r_i)

Figure 2: Observed behavior. Rectangles depict process events, triangles show data operations and hexagons indicate the role of resource.(a) process trace. (b) data trace [12]

3 Preliminaries

This section introduces a set of concepts that will be used through the paper.

The normative behavior of a business process can be described using process models. A process model illustrates the activities to be performed in a special order to reach a certain business goal. In this paper, we represent process models using a subclass of Petri nets known as workflow nets [13]. We define an extension of workflow nets which incorporates both the process perspective and the resource perspective as follows:

Definition 1 (WFR-net)**.** Let $\mathcal{U}_a$ be the universe of activities and $A \subseteq \mathcal{U}_a$ be a set of activities. Let $\mathcal{U}_r$ be the universe of roles and $R \subseteq \mathcal{U}_r$ be a set of roles. $N = ((P, T, F, \lambda), \rho, m_i, m_f)$ is a a workflow net with roles (WFR-net) such that:

- P and T are finite sets of places and transitions fulfilling $P \cap T = \emptyset$;
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation between places and transitions (and between transitions and places);
- $\lambda : T \rightarrow A^\tau$ is a function mapping transitions to activities. (We reserve $\tau \notin A$ as the label of all invisible transitions. For convenience, for any set $A \subseteq \mathcal{U}_a$, $A^\tau = A \cup \{\tau\}$ denotes the union set of A and $\{\tau\}$);
- $\rho : T \rightarrow R \cup \{\perp\}$ is a function assigning a role to each transition. We use the special symbol $\perp$ as the role of all invisible transitions.
- $m_i, m_f \in P$ are places satisfying $((T \times \{i\}) \cap F = \emptyset)$ and $((\{f\} \times T) \cap F = \emptyset)$
- for any node $n \in P \cup T$ there exists a path from m_i to n and from n to m_f.

A process model shows prescribed behavior whereas a process log records the executed behavior. A process model (here a WFR-Net) describes which role is allowed to perform which activities (tasks) and in which order. Each executed task results in start and complete events, that record information regarding the case, the corresponding activity, the time of the execution, the actor who executed the activity etc. To capture all mentioned information, we formalize process events and their attributes as follows:

Definition 2 (Process Event, Attribute)**.** Let $\mathcal{U}_\mathcal{E}$ be the process event universe, i.e., the set of all possible event identifiers. A process event may have various attributes. Let AT be the attribute universe. For each attribute name $atr \in AT$, we define the attribute function $\#_{atr} : \mathcal{U}_\mathcal{E} \rightarrow Val$ which maps each event $e \in \mathcal{U}_\mathcal{E}$ to the value assigned to event e for this attribute name atr. In other words, $\#_{atr}(e)$ returns the value of attribute atr for event e.

Let $\mathcal{U}_i$ be the identifier universe, $\mathcal{U}_c$ be the case universe, $\mathcal{U}_a$ be the activity universe, $\mathcal{U}_u$ be the actor universe, $\mathcal{U}_{Ai}$ be the aim universe, and $\mathcal{U}_t$ be the time universe. We define the following attributes for any event $e \in \mathcal{U}_\mathcal{E}$:

- $\#_{caseId}(e) \in \mathcal{U}_c$, which denotes the case associated to e;
- $\#_{eventId}(e) \in \mathcal{U}_i$, which denotes the ID associated to e;
- $\#_{id}(e) \in \mathcal{U}_i$, which denotes the ID associated to the activity instance of e;
- $\#_{act}(e) \in \mathcal{U}_a$, which denotes the activity associated to e;
- $\#_{actor}(e) \in \mathcal{U}_u$, which denotes the actor associated to e;
- $\#_{type}(e) \in \{Start, Complete\}$, which indicates the type of the event e;
- $\#_{time}(e) \in \mathcal{U}_t$, which indicates the timestamp of event e.

Events that denote activity executions as part of the same case are grouped together to form a process trace. Therefore, a process trace represents the behaviour recorded for one specific instance of the process. Based on [14], we formalize process log and process-trace as follows:

Definition 3 (Process Log)**.** Let $\mathcal{U}_\mathcal{E}$ be the process event universe. A process log $L_P \subseteq \mathcal{U}_\mathcal{E}$ is a finite set of process events.

Definition 4 (Process Trace)**.** Let L_p be a process log and $c \in \mathcal{U}_c$ be a case identifier. We define $\sigma_c = \langle e_1, e_2, \ldots, e_n \rangle \in {L_p}^*$ to be a trace such that:

- For any $e_i, e_j \in \sigma_c$ where $1 \leq i < j \leq n$ it holds that $\#_{case}(e_i) = \#_{case}(e_j) = c$, i.e. each event in a trace refers to the same case identifier.
- For any $e_i, e_j \in \sigma_c$ where $1 \leq i < j \leq n$ it holds that $e_i \neq e_j$, i.e. each event occurs only once in the trace.
- For any $e_i, e_j \in \sigma_c$ where $1 \leq i < j \leq n$ it holds that $\#_{time}(e_i) < \#_{time}(e_j)$, i.e. events are totally ordered.
- There is no $e \in L_p$ such that $\#_{case}(e) = c$ and $e \notin \sigma_c$, i.e. in each case all events of a case are in the trace.
- For all $\sigma_c, \sigma_{c'} \in {L_p}^*$ holds that if $c \neq c'$ then $\sigma_c \cap \sigma_{c'} = \emptyset$.

The execution of an activity may require performing certain data operations on several data fields. In this research, we assume that operations on data fields are always executed in the context of process activities. This relation is often represented using a data model. The data model relates the process logic to the data layer by indicating which data operations on given data fields must be executed in order to complete a given activity;

Definition 5 (Data model)**.** Let $\mathcal{U}_a$ be the universe of activities. Let $\mathcal{U}_o$ be the universe of data operations and $\vdash \in \mathcal{U}_o$ be the no operation. We define the Data model as $\delta : \mathcal{U}_a \rightarrow \mathcal{P}(\mathcal{U}_o) \backslash \emptyset$ such that for $a \in \mathcal{U}_a$ if $\vdash \in \delta(a)$ then $\delta = \{\vdash\}$

Data operations are typically recorded by the database management systems. We refer to a recorded data operation as a data-event (Definition6). A data log is a finite set of data events (Definition7). A sequence of data events related to the same case is defined as data trace (Definition8).

Definition 6 (Data Event, Attribute)**.** Let $\mathcal{U}_d$ be the universe of all identifiable data events also called data operations. A data event (op) may have various attributes. Let AT be the attribute universe. For any $op \in \mathcal{U}_d$ and attribute name $atr \in AT$,we use the notation $\#_{atr}(op)$ for the value of attribute atr for event op.
Let $\mathcal{U}_i$ be the identifier universe, $\mathcal{U}c$ be the case universe, $\mathcal{U}_o$ be the universe of data operations, $\mathcal{U}_u$ be the actor universe and $\mathcal{U}_t$ be the time universe. We define the following attributes for any data event $op \in \mathcal{U}_d$:

- $\#_{id}(op) \in \mathcal{U}_i$, which denotes the id associated to op;
- $\#_{case}(op) \in \mathcal{U}_c$, which denotes the case associated to op;
- $\#_{opr}(op) \in \mathcal{U}_o$, which denotes the data operation associated to op;
- $\#_{actor}(op) \in \mathcal{U}_u$, which denotes the actor associated to op;
- $\#_{time}(op) \in \mathcal{U}_t$, which indicates the timestamp of op;

Definition 7 (Data Log)**.** Let $\mathcal{U}_d$ be the data event universe. The data event log $L_D \subseteq \mathcal{U}_d$ is a finite set of data events.

Definition 8 (Data Trace)**.** Let L_D be a data log and $c \in \mathcal{U}_c$ be a case identifier. We define $\beta_c = \langle op_1, op_2, \ldots, op_n \rangle \in {L_D}^*$ to be a data trace such that:

- For any $op_i, op_j \in \beta_c$ where $1 \leq i < j \leq n$ it holds that $\#_{case}(op_i) = \#_{case}(op_j) = c$, i.e. each data event in a trace refers to the same case identifier.
- For any $op_i, op_j \in \beta_c$ where $1 \leq i < j \leq n$ it holds that $op_i \neq op_j$, i.e. each data event occurs only once in the trace.
- For any $op_i, op_j \in \beta_c$ where $1 \leq i < j \leq n$ it holds that $\#_{time}(op_i) < \#_{time}(op_j)$, i.e. events are totally ordered.
- There is no $op \in L_D$ such that $\#_{case}(op) = c$ and $op \notin \beta_c$, i.e. in each case all data events of a case are in the trace.
- For all $\beta_c, \beta_{c'} \in {L_D}^*$ holds that if $c \neq c'$ then $\beta_c \cap \beta_{c'} = \emptyset$.

An organisational model (Definition 9) represents the interaction between roles and actors. Each role can have multiple actors and each actor can have multiple roles.

Definition 9 (Organisational Model)**.** Let R be the set of roles in a WFR-net. Let U be a set of actors. An organisational model $\mathcal{O}$ consists of a set of entries; each entry $x \in \mathcal{O}$ is a pair $(role, actor) \in R \times U$.

Definition 10 (minimal/maximal element of a set)**.** With $\mathcal{U}$ we define universes. For sake of argument, we assume all universes to be totally ordered and for any $u \in \mathcal{U}, u \neq \emptyset$ we use $\downarrow u$ and $\uparrow u$ to denote the minimal and maximal element of u.

4 Proposed Multi-Layer Alignment Approach

In this section, we propose our approach for multi-perspective conformance checking. The main goal of this approach is to align process, data and privacy policy layers to find hidden deviations between these three perspectives of a business process in addition to detecting the deviations in each layer.

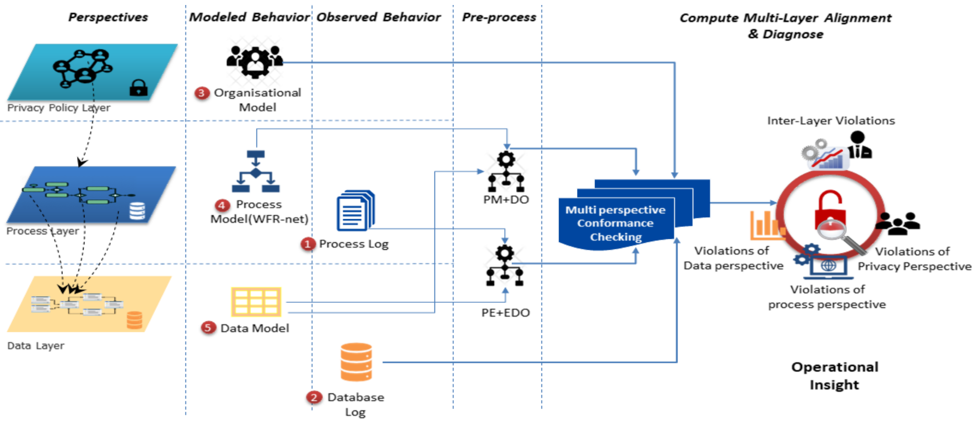


Figure 3: An overview of the proposed approach [12]

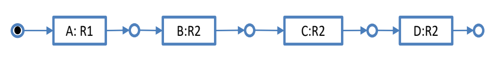
Figure 3 shows an overview of our approach together with its inputs and outputs [12]. A Process log (❶) records process executions and a data log (❷) contains data operations showing which user accessed which data. These two inputs indicate observed behaviors.

To represent the modeled behaviors the approach considers a process model (❹), a data model (❺) and an organisational model (❸). A process model describes the activities to be performed in a specific order to reach a certain business goal. The data model relates the process logic to the data layer by indicating which data operations must be executed in order to complete a given activity. The organisational model links users to their roles. The role of actors in the process log and data log can be retrieved from this model. As discussed before, using only an organisational model for access control is not sufficient to check data privacy. Therefore, in order to find the context of data access, first we integrate the activities with their corresponding roles in the process model to unify the two perspectives of process and privacy into a single model. Second, using the data model, we enrich the aforementioned process model with expected data operations in a pre-processing step, shown as “PM+DO” in Fig. 3. In another pre-processing step, we enrich the events of the process log with the expected data operations using the data model (“PE+EDO” in Fig. 3).

The combination of the process model with role information, the process event log showing the start and complete of activities by specific resources and the data log showing who accessed what at which time is translated into a large so-called synchronous product model. Such a synchronous product is the foundational model for conformance checking and techniques exist to find the optimal execution given a cost function that penalizes specific deviations.

4.1 Construction of Synchronous Product Model

To clarify the pre-processing steps and constructing the synchronous product in our approach, let us consider the inputs shown in Fig. 4. For the rest of the paper, we are going to refer this example as running example 2 while the running example 1 is the healthcare process.



(a). Process Model

Activity	Data Operations
A	d1: Read(x) , d2: Update(y)
B	d3: Update(z)
C	d4: Update(K)
D	d5: Read(x, y)

(b). Data Model

Role	Actor (Resource)
R1	u1
R2	u2

(c). Organizational Model

CaseID	EventID	ID	Activity	Actor	Type	Time
100	e1	1	A	u1	Start	2021-02-01 12:38:44
100	e2	1	A	u1	Complete	2021-02-01 12:58:44
100	e3	2	B	u1	Start	2021-02-01 13:19:37
100	e4	2	B	u1	Complete	2021-02-01 13:39:38
100	e5	3	C	u1	Start	2021-02-01 13:42:38
100	e6	3	C	u1	Complete	2021-02-01 13:48:38
100	e7	4	F	u1	Start	2021-02-01 14:20:38
100	e8	4	F	u1	Complete	2021-02-01 14:36:38

(d). Process Trace (a fragment of the process log)

CaseID	EventID	Operation	Actor	Type	Time
100	op1	Read(x)	u1	Complete	2021-02-01 12:40:44
100	op2	Update(z)	u1	Complete	2021-02-01 13:20:37
100	op3	Update(m)	u1	Complete	2021-02-01 13:32:38

(e). Data Trace (a fragment of the data log)

Figure 4: The inputs of the proposed approach in the running example 2

Figure 4(a) shows a workflow-net as the process model. This process model starts with activity A by role R1 and continues with activities B, C, and D by role R2. According to the data model depicted in Fig. 4(b), two data operations Read(x) and Update(y) should be executed while performing activity A, and each of activities B, C, and D is expected to execute one data operation. Figure 4(c) shows the organisational model in our example. There are two roles in the organisational model. Actor (resource) u1 has the role R1 and the actor u2 has the role R2.

Figure 4(d) shows one trace of the process log. This trace contains eight process events that correspond to a single case. The start and complete events with the same activity name and id indicate the occurrence of an instance of a specific activity. For example, e_3 and e_4 both with the same id equal to 2 indicate the execution of one instance of activity B. The events are sorted by their occurrence time. The process trace in Fig. 4(d) can be presented as the sequence $\langle A_s, A_c, B_s, B_c, C_s, C_c, F_s, F_c \rangle$. Figure 4(d) together with Fig. 4(e) shape the observed behavior for Case (process instance) 100.

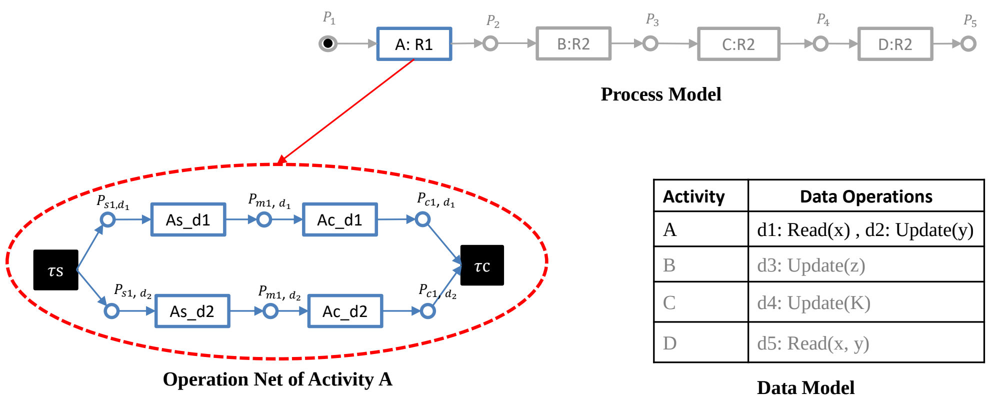
Figure 4(e) presents a data trace with three data operations op1, op2, and op3 which were executed on the data fields x, z, and m while performing the process instance (case) 100.

As the first step of pre-processing, we define the operation net to enrich the process model with the expected data operations specified in the data model.

Definition 11 (Operation Net)**.** Let δ be a data model and $\mathcal{U}_a$ be the universe of activities and $\mathcal{U}_r$ the universe of roles . Let $N = ((P, T, F, \lambda), \rho, m_i, m_f)$ be a WFR-Net. Let $t \in T$ be a transition with $\lambda(t) = a$, $a \in \mathcal{U}_a$ and $\rho(t) = r$, $r \in \mathcal{U}_r$. We define operation net, $ON_t = (P_t, T_t, F_t, \lambda_t, \rho_t)$ where:

- $P_t = \{p_s, p_m, p_c\} \times \{t\} \times \delta(a)$.
- $T_t = \{\tau_s^t, \tau_c^t\} \cup (\{t\} \times \{Start, Complete\} \times \delta(a))$
- F_t=$\{(\tau_s^t, p_{s,t,d}) | d \in \delta(a)\} \cup \{(p_{s,t,d}, (t, Start, d)) | d \in \delta(a)\} \cup \{((t, Start, d), p_{m,t,d}) | d \in \delta(a)\} \cup \{(p_{m,t,d}, (t, Complete, d)) | d \in \delta(a)\} \cup \{((t, Complete, d), p_{c,t,d}) | d \in \delta(a)\} \cup \{(p_{c,t,d}, \tau_c^t) | d \in \delta(a)\}$
- $\lambda_t : T_t \rightarrow (\mathcal{U}_a \times \{Start, Complete\} \times \delta(a)) \cup \{\tau\}$ such that:
  - $\lambda_t(\tau_s^t) = \lambda_t(\tau_c^t) = \tau$,
  - $\forall (t, et, d) \in (\{t\} \times \{Start, Complete\} \times \delta(a))\ \lambda_t((t, et, d)) = (\lambda(t), et, d)$
- $\rho_t : T_t \rightarrow r \cup \{\perp\}$ such that:
  - $\rho_t(\tau_s^t) = \rho_t(\tau_c^t) = \perp$,
  - $\forall t \in T_t, t \notin \{\tau_s^t, \tau_c^t\}\ \rho_t(t) = r$

Each transition in the process model is represented by an operation net corresponding to the data operations of the activity that the transition is labeled with. For instance, Fig. 5 shows the operation net for activity A of example 2.



Activity	Data Operations
A	d1: Read(x) , d2: Update(y)
B	d3: Update(z)
C	d4: Update(K)
D	d5: Read(x, y)

Figure 5: Operation Net of activity A in the running example2

The model net, which is one of the foundations of the synchronous product is constructed by replacing each activity in the process model with corresponding operation net. Definition 12 formalizes the model net.

Definition 12 (Model Net)**.** Let $N = ((P, T, F, \lambda), \rho_M, m_i, m_f)$ be a WFR-Net, $\mathcal{U}_a$ be the universe of activities, $\mathcal{U}_r$ be the universe of roles. We define $N_M = ((P_M, T_M, F_M, \lambda_M), \rho_M, m_{i_M}, m_{f_M})$ as a model net, where:

- $P_M = P \cup \bigcup_{t \in T} \{P_t \mid ON_t = (P_t, T_t, f_t, \lambda_t, \rho_t)\}$
- $T_M = \bigcup_{t \in T} \{T_t \mid ON_t = (P_t, T_t, f_t, \lambda_t, \rho_t)\}$
- $F_M = \bigcup_{t \in T} \{F_t \mid ON_t = (P_t, T_t, f_t, \lambda_t, \rho_t)\} \cup \{(p, t') \in P \times T_M | \exists (p, t) \in F, t' = \tau_s^t\} \cup \{(t', p) \in T_M \times P | \exists (t, p) \in F, t' = \tau_c^t\}$
- $\lambda_M : T_M \to (\mathcal{U}_a \times \{Start, Complete\} \times \delta(a)) \cup \{\tau\}$ such that: $\forall t \in T\ \forall t' \in T_t\ \lambda_M(t) = \lambda_t(t')$
- $\rho_M : T_M \to \mathcal{U}_r$ such that: $\forall t \in T\ \forall t' \in T_t\ \rho_M(t) = \rho_t(t')$.
- $m_{i_M} = m_i$, and $m_{f_M} = m_f$.

Fig. 6 shows the model net for our running example 2. In essence, this model is the output of enriching the process model (Fig. 4(a)) with the expected data operations shown in Fig. 4(b) in the pre-processing step (see "PM+DO" in Fig. 3).

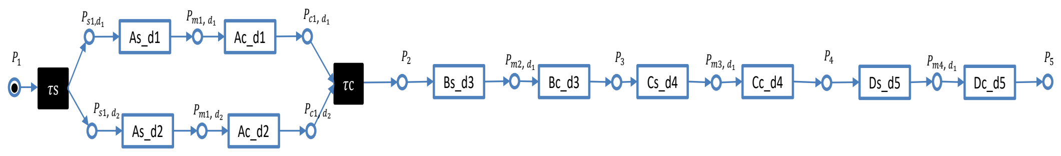


Figure 6: Model net of the running example 2

As the second foundation of our synchronous product, now we define the process net.

Definition 13 (Process Net)**.** Let δ be a data model $\mathcal{U}_a$ be the universe of activities. Let e be a process event and $\#_{act}(e) = a$ and $a \in \mathcal{U}_a$. Let L_P be a process log and $\sigma = \langle e_1, e_2, \ldots, e_n \rangle \in L_P{}^*$ be a process trace of length n. $\delta(\#_{act}(e))$ is the set of expected data operations for $\#_{act}(e)$. $\downarrow \delta(\#_{act}(e))$ denotes the first element of the set and $\uparrow \delta(\#_{act}(e))$ denotes the last element of the set. We define $N_P = ((P_P, T_P, F_P, \lambda_P), m_{i_P}, m_{f_P})$ as a process net, where:

- P_P=$\{p_{e,d} | e \in \sigma, d \in \delta(\#_{act}(e))\}$ $\cup$$\{p_f\}$
- $T_P = \bigcup_{e \in \sigma} \{e\} \times \delta(\#_{act}(e))$
- F_P=$\{(p_{e,d}, (e, d)) | e \in \sigma, d \in \delta(\#_{act}(e))\} \cup \{((e, d), p_{e',d'}) | (e, d) \in T_P, \langle \ldots, e, e', \ldots \rangle = \sigma, d = \uparrow \delta(\#_{act}(e)), d' = \downarrow \delta(\#_{act}(e'))\} \cup \{((e_n, d), p_f) | (e_n, d) \in T_P, d = \uparrow \delta(\#_{act}(e_n))\}$
- $\lambda_P : T_P \to \mathcal{U}_a \times \{Start, Complete\} \times \delta(\#_{act}(e))$ such that: $\forall (e, d) \in T_P\ \lambda_P((e, d)) = (\#_{act}(e), \#_{type}(e), d)$
- $m_{i_P} = \{p_{e_1,d} | d = \downarrow \delta(\#_{act}(e_1))\}$
- $m_{f_P} = \{p_f\}$

The process net represents a process trace. The process net defined above is a sequence of the transitions labelled with activities and their life cycle as they appeared in the process trace. As discussed earlier, in the pre-processing step we also enrich events in the process trace with expected data operations. Consequently, each event will be replaced by some artificial events if its corresponding activity exists in the data model. Fig. 7 shows the process net for our process trace example $\langle A_s, A_c, B_s, B_c, C_s, C_c, F_s, F_c \rangle$, where we chose new identifiers for the places, different from the places in Fig. 6.

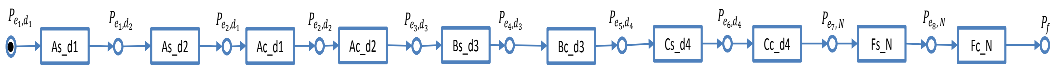


Figure 7: Process net of the running example 2

To match start and complete events related to one instance of an activity we define a matching function as follows:

Definition 14 (Matching Function)**.** Let $N_P = ((P_P, T_P, F_P, \lambda_P), m_{i_P}, m_{f_P})$ be a process net. Let t_P and t'_P be two different transitions in the process net. $\pi_1(t)$ denotes the first element of t which is the corresponding event identifier. $S : T_P \to T_P$ is a function that for a given start transition provides the complete transition and vice-versa. $S(t) = t'$ if and only if:

- $\#_{type}(\pi_1(t)) \neq \#_{type}(\pi_1(t'))$, and
- $\#_{act}(\pi_1(t)) = \#_{act}(\pi_1(t'))$, and
- $\#_{actor}(\pi_1(t)) = \#_{actor}(\pi_1(t'))$, and
- $\#_{id}(\pi_1(t)) = \#_{id}(\pi_1(t'))$.

This function matches two events, one of type start and one of type complete, but both carry the same activity with the same actor and id.

The third foundation of the synchronous product in our approach is the data net. We define data net as follows:

Definition 15 (Data Net)**.** Let $\mathcal{U}_o$ be the universe of data operations. Let L_D be a data log and $\beta_c = \langle op_1, op_2, \ldots, op_n \rangle \in {L_D}^*$ be a data trace. $N_D = ((P_D, T_D, F_D, \lambda), m_iD, m_fD)$ is a data net such that:

- P_D=$\{p_{i,op} | 1 \leq i \leq 2, op \in \beta_c\}$,
- T_D=$\{op | op \in \beta_c, op \neq \vdash\}$,
- F_D=$\{(p_{i,op}, op) | i = 1, op \in \beta_c\} \cup \{(op, p_{i,op}) | i = 2, op \in \beta_c\}$,
- $\lambda_D : T_D \to \mathcal{U}_o$ such that $\forall t \in T_D \; \lambda_D(t) = (\#_{opr}(op))$,
- m_{iD}=$\{p_{1,op} | op \in \beta_c\}$,

- m_{fD}=$\{p_{2,op} | op \in \beta_c\}$.

Figure 8 shows the data net for our data trace example $\langle op1, op2, op3\rangle$, where we chose new identifiers for the places, different from the places in figures 6 and 7.

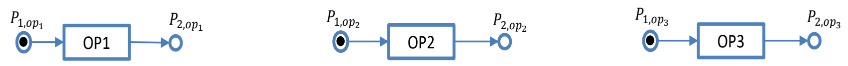


Figure 8: Data net of the running example 2

Using the model net, process net and data net, we define the synchronous product model as the combination of the three nets with two additional set of totally and partially synchronous transitions.

Definition 16 (Synchronous Product). Let $N_M = ((P_M, T_M, F_M, \lambda_M), m_{i_M}, m_{f_M})$, $N_P = ((P_P, T_P, F_P, \lambda_P), m_{i_P}, m_{f_P})$, and $N_D = ((P_D, T_D, F_D, \lambda_D), m_{i_D}, m_{f_D})$ be model net, process net and data net respectively. The synchronous product of N_M, N_P, and N_D is the Petri net $N = N_M \otimes N_P \otimes N_D = ((P, T, F, \lambda), m_i, m_f)$ where:

(a) P=$P_M \cup P_P \cup P_D$ is a finite set of places including all places in the model, process and data nets.

(b) $T \subseteq (T_M^{\gg} \times T_P^{\gg} \times T_D^{\gg})$ is a finite set of transitions, such that:
T=$\{(t_M, \gg, \gg) | t_M \in T_M\} \cup$
$\{(\gg, t_P, \gg) | t_P \in T_P\} \cup \{(\gg, \gg, t_D) | t_D \in T_D\} \cup$
$\{(t_M, t_P, \gg) \in (T_M \times T_P \times \{\gg\}) | \lambda_M(t_M) = \lambda_P(t_P) \neq \tau\} \cup$
$\{(t_M, t_P, t_D) \in (T_M \times T_P \times T_D) | \lambda_M(t_M) = \lambda_P(t_P) \neq \tau, \lambda_D(t_D) = \pi_3(\lambda_M(t_M)),$
$\#_{actor}(\pi_1(\lambda_P(t_P))) = \#_{actor}(t_D),$
$\#_{time}(\pi_1(\lambda_P(t_P))) \leq \#_{time}(t_D) \leq \#_{time}(\pi_1(\lambda_P(S(t_P))))\}$

(c) $F \subset (P \times T) \cup (T \times P)$ is the flow relation between places and transitions (and between transitions and places), such that:

F=$F_M \cup F_P \cup F_D \cup$

$\{(p_M, (t_M, t_P, \gg)) | (t_M, t_P, \gg) \in T, (p_M, t_M) \in F_M\} \cup$
$\{(p_P, (t_M, t_P, \gg)) | (t_M, t_P, \gg) \in T, (p_P, t_tP) \in F_P\} \cup$
$\{((t_M, t_P, \gg), p_M) | (t_M, t_P, \gg) \in T, (t_M, p_M) \in F_M\} \cup$
$\{((t_M, t_P, \gg), p_P) | (t_M, t_P, \gg) \in T, (t_P, p_P) \in F_P\} \cup$
$\{(p_M, (t_M, t_P, t_D)) | (t_M, t_P, t_D) \in T, (p_M, t_M) \in F_M\} \cup$
$\{(p_P, (t_M, t_P, t_D)) | (t_M, t_P, t_D) \in T, (p_P, t_P) \in F_P\} \cup$
$\{(p_D, (t_M, t_P, t_D)) | (t_M, t_P, t_D) \in T, \pi_2(\lambda_P(t_P)) = Start, (p_D, t_D) \in F_D\} \cup$
$\{((t_M, t_P, t_D), p_M) | (t_M, t_P, t_D) \in T, (t_M, p_M) \in F_M\} \cup$
$\{((t_M, t_P, t_D), p_P) | (t_M, t_P, t_D) \in T, (t_P, p_P) \in F_P\} \cup$
$\{((t_M, t_P, t_D), p_D) | (t_M, t_P, t_D) \in T, \pi_2(\lambda_P(t_P)) = Complete, (t_D, p_D) \in F_D\}$

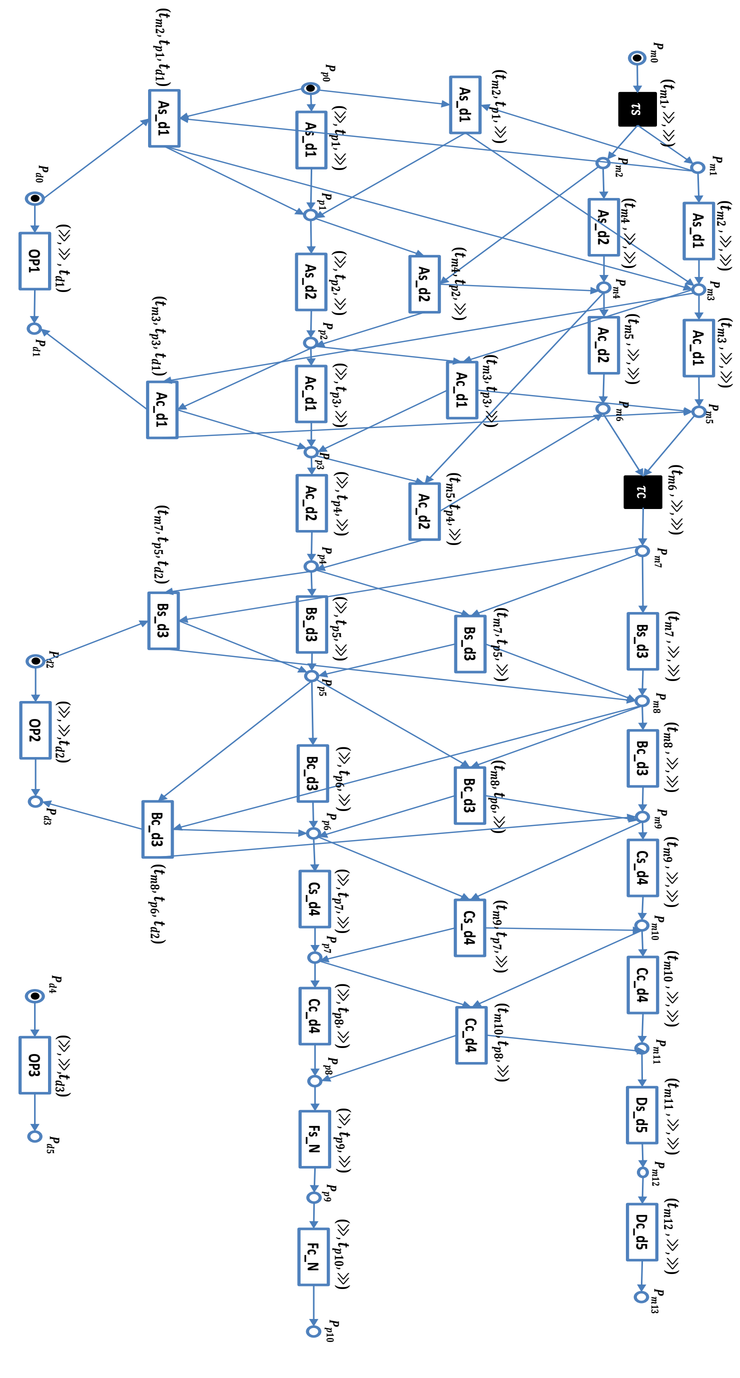


Figure 9: Synchronous product model based on the inputs of the running example 2

(d) λ is a labeling function that maps each transition to activity/data operation, such that for all $(t_M, t_P, t_D) \in T, \lambda((t_M, t_P, t_D)) = \lambda_M(t_M)$ if $t_P, t_D = \gg, \lambda((t_M, t_P, t_D)) = \lambda_P(t_P)$ if $t_M, t_D = \gg, \lambda((t_M, t_P, t_D)) = \lambda_D(t_D)$ if $t_M, t_P = \gg, \lambda((t_M, t_P, t_D)) = \lambda_M(t_M)$ otherwise.

(e) $m_i = m_{i_M} \cup m_{i_P} \cup m_{i_D}$ is the set of initial places including all initial places of model, process and data nets.

(f) $m_f = m_{f_M} \cup m_{f_P} \cup m_{f_D}$ is the set of final places including all final places of model, process and data nets.

Figure 9 displays the synchronous product for our running example 2, which is a combination of three nets: model net, process net and data net. These individual nets are presented in figures 6, 7 and 8, respectively. Part (a) of definition 16 ensures that the synchronous product model contains all places in the mentioned nets. Part (e) of definition 16 emphasise that the initial places of the synchronous product model comprise all initial places of the model, process, and data nets. Moreover, part (f) of this definition indicates the final places of the synchronous product model encompass all final places in these nets.
Part (b) of definition 16 brings together all transitions of the model, process, and data nets into the synchronous product model. Additionally, it defines two sets of synchronous transitions in the synchronous product model, namely, totally synchronous transitions and partially synchronous transitions. Totally synchronous transitions only exist when an expected activity and data operation appear in the process net and data net. Additionally, the timestamp of the data event should be between the start and completion time of the expected activity and the actors (resources) of both data and process events should be the same. Partially synchronous transitions pair transitions in the model net and the process net that have the same label, but no data operation was found in the data net that would also match.
Part (d) of definition 16 labels all transitions in the synchronous product model. Finally part (c) of definition 16 shows how original model, process, data nets and added synchronous transitions in the synchronous product model are connected.

4.2 Multi-layer Alignment

An alignment is a firing sequence of transitions from initial marking m_i to final marking m_f in the synchronous product model. We need to relate "moves" in the logs to "moves" in the model in order to establish an alignment between the model, process trace and data trace. However, it may be that some of the moves in the logs cannot be mimicked by the model and vice-versa. We explicitly denote such "no moves" by "$\gg$". Formally, we set t_M to be a transition of the activities in the model net (process model), t_P to be a transition of the events in process net (process trace), and t_D to be a transition of the events in data net (data trace). We define types of moves in our approach as follows:

Definition 17 (Move Types). Let $N_M = ((P_M, T_M, F_M, \lambda_M), \rho_M, m_{i_M}, m_{f_M})$ be a model net,$N_P = ((P_P, T_P, F_P, \lambda_P), m_{i_P}, m_{f_P})$ be a process net and $N_D = ((P_D, T_D, F_D, \lambda), m_iD, m_fD)$ be a data net and $N = N_M \otimes N_P \otimes N_D$ be a synchronous product model. Let R be a set of roles. Let $\mathcal{U}_u$ be the universe of actors, U a set of actors and $\mathcal{O} \subseteq R \times U$ be an organizational model. We use $\#_{actor}(t) : T \to \mathcal{U}_u$ with $\#_{actor}(t) = u$ to identify the actor of transition t. A legal move in a multi-layer alignment is represented by a tuple $(t_M, t_P, t_D) \in T$ such that:

(a) (t_M, t_P, t_D) is move on model if $t_M \neq \gg$ and $t_P = \gg, t_D = \gg$,

(b) (t_M, t_P, t_D) is move on process log if $t_P \neq \gg$ and $t_D = \gg, t_M = \gg$,

(c) (t_M, t_P, t_D) is move on data log if $t_D \neq \gg$ and $t_P = \gg, t_M = \gg$,

(d) (t_M, t_P, t_D) is partially synchronous move with legitimate role if $t_P \neq \gg, t_M \neq \gg, t_D = \gg$, and $(\rho_M(t_M), \#_{actor}(t_P)) \in \mathcal{O}$,

(e) (t_M, t_P, t_D) is partially synchronous move with illegitimate role if $t_P \neq \gg, t_M \neq \gg, t_D = \gg$, and $(\rho_M(t_M), \#_{actor}(t_P)) \notin \mathcal{O}$,

(f) (t_M, t_P, t_D) is totally synchronous move with illegitimate role if $t_M \neq \gg$, and $t_P \neq \gg, t_D \neq \gg$, and $\#_{actor}(t_P) = \#_{actor}(t_D)$, and $(\rho_M(t_M), \#_{actor}(t_P)) \notin \mathcal{O}$,

(g) (t_M, t_P, t_D) is totally synchronous move with legitimate role if $t_M \neq \gg$ and $t_P \neq \gg, t_D \neq \gg$, and $\#_{actor}(t_P) = \#_{actor}(t_D)$ and $(\rho_M(t_M), \#_{actor}(t_P)) \in \mathcal{O}$.

All other moves are considered as illegal moves.

In this synchronous product, totally synchronous move with legitimate role represent expected behavior (definition 17(g)). We further distinguish six kinds of deviations:

- A *move on model* happens when there are unobserved activity and data operation (definition 17(a)).
- A *move on process log* happens when an unexpected activity was performed (definition 17(b)).
- A *move on data log* happens when a not-allowed data operation was executed (definition 17(c)).
- A *partially synchronous move with legitimate role* happens when there is a missing data operation in the data log. In this case, the expected activity was performed by a legitimate role (definition 17(d)).
- A *partially synchronous move with illegitimate role*, as the previous, but performed by a not-allowed role (definition 17(e)).
- A *totally synchronous move with illegitimate role* happens when an expected activity and data operation were done by a not-allowed role (definition 17(f)).

The computation of an optimal alignment relies on the definition of a proper cost function for the possible kinds of moves. We extend the standard cost function to include data and privacy costs. We define our default multi-layer alignment cost function as follows:

Definition 18 (Multi-Layer Alignment Cost function)**.** Let (t_M, t_P, t_D) be a move in alignment between a model, process trace and a data trace. The cost $K(t_M, t_P, t_D)$ is defined as:

$$K(t_M, t_P, t_D) = \begin{cases} 4 & \text{if } (t_M, t_P, t_D) \text{ is a move on process log} \\ & \text{or a move on data log, or a move on model} \\ 2 & \text{if } (t_M, t_P, t_D) \text{ is a partially synchronous move} \\ & \text{with legitimate role} \\ 2 + penaltyCost & \text{if } (t_M, t_P, t_D) \text{ is a partially synchronous move} \\ & \text{with illegitimate role} \\ 0 & \text{if } (t_M, t_P, t_D) \text{ is a totally synchronous move} \\ & \text{with legitimate role} \\ penaltyCost & \text{if } (t_M, t_P, t_D) \text{is a totally synchronous move} \\ & \text{with illegitimate role} \end{cases}$$

Note that, in order to include the cost for deviations related to the privacy layer, we considered a penalty cost in our cost function. The penalty cost can get a value more than 0 and equal to or less than 1. If the actor of observed behavior was not allowed to perform the activity and/or the data operation we add a penalty cost. If we assign the value 1 to the penalty cost, the cost for partially synchronous move with an illegitimate role is equal to 3, and the cost for totally synchronous move with an illegitimate role is equal to 1.

The alignment with the lowest cost is called an optimal alignment. We define Optimal Multi-Layer Alignment as follows:

Definition 19 (Optimal Multi-Layer Alignment)**.** Let N be a WFR-net, $\sigma_c \in L_P$ and $\beta_c \in L_D$ be a process trace and data trace, respectively. Assuming $\mathcal{A}_\mathcal{N}$ as the set of all legal alignment moves, a cost function K assigns a non-negative cost to each legal move: $\mathcal{A}_\mathcal{N} \rightarrow \mathbb{R}_0^+$. The cost of an alignment γ between σ_c, β_c and N is computed as the sum of the cost of all constituent moves $\mathcal{K}(\gamma) = \sum_{(t_M, t_P, t_D) \in \gamma} K(t_M, t_P, t_D)$. Alignment γ is an optimal alignment if for any alignment γ' of σ_c, β_c and N, $\mathcal{K}(\gamma) \leq \mathcal{K}(\gamma')$.

For finding the optimal alignments we employed A* algorithm. Figure 10 illustrates an optimal alignment for running example 2, depicted on top of the synchronous product shown in Fig. 9. It shows that there are six kinds of deviations between observed behavior and modeled behavior, namely partially synchronous moves on transitions As_d2 and Ac_d2 in dark blue color, totally synchronous moves with illegitimate role on transitions Bs_d3 and Bc_d3 in orange color, partially synchronous moves with illegitimate role on transitions Cs_d4 and Cc_d4 in light blue color, model moves on transitions Ds_d5 and Dc_d5 in purple color, process log moves on transitions Fs_N and Fc_N in yellow color, and a data log move on transition $OP3$ in red color.

5 Tool Support

We implemented the approach illustrated in Fig. 3 as a package named "Multi-LayerAlignment" in the ProM[3] framework [15]. It is available for the end users in ProM 6.11 release[4]. The tool takes as inputs a process model, a data model, an organisational model, a process log, and a data log then computes multi-layer alignments between the process, data, and privacy policy perspectives.

As the output of the tool, we developed two different visualisations that illustrate the abstractions of multi-perspective alignment results with projection to process log and data log, respectively.

"Projection to process log" visualisation facilitates observing the violations of each perspective in an overall view as well as providing detailed information on each process event/activity. Figure 11 shows a screenshot of the projection to process log visualisation. In this visualisation privacy, process and data perspectives are depicted by circles, chevron arrows and triangles respectively. Expected behaviors in each layer are shown in green color. Red color indicates unexpected behavior and highlighted white/red color marks missing behavior. Note that, in control-flow based conformance checking, purple and yellow colors are known as move on model and move on process log in the process mining research community. We used these colors to show how our approach can detect hidden deviations related to other perspectives of a business process in addition to control-flow violations. Figure 11, clearly shows that control-flow based conformance checking

[3]https://www.promtools.org/

[4]https://svn.win.tue.nl/trac/prom/wiki/ProM611

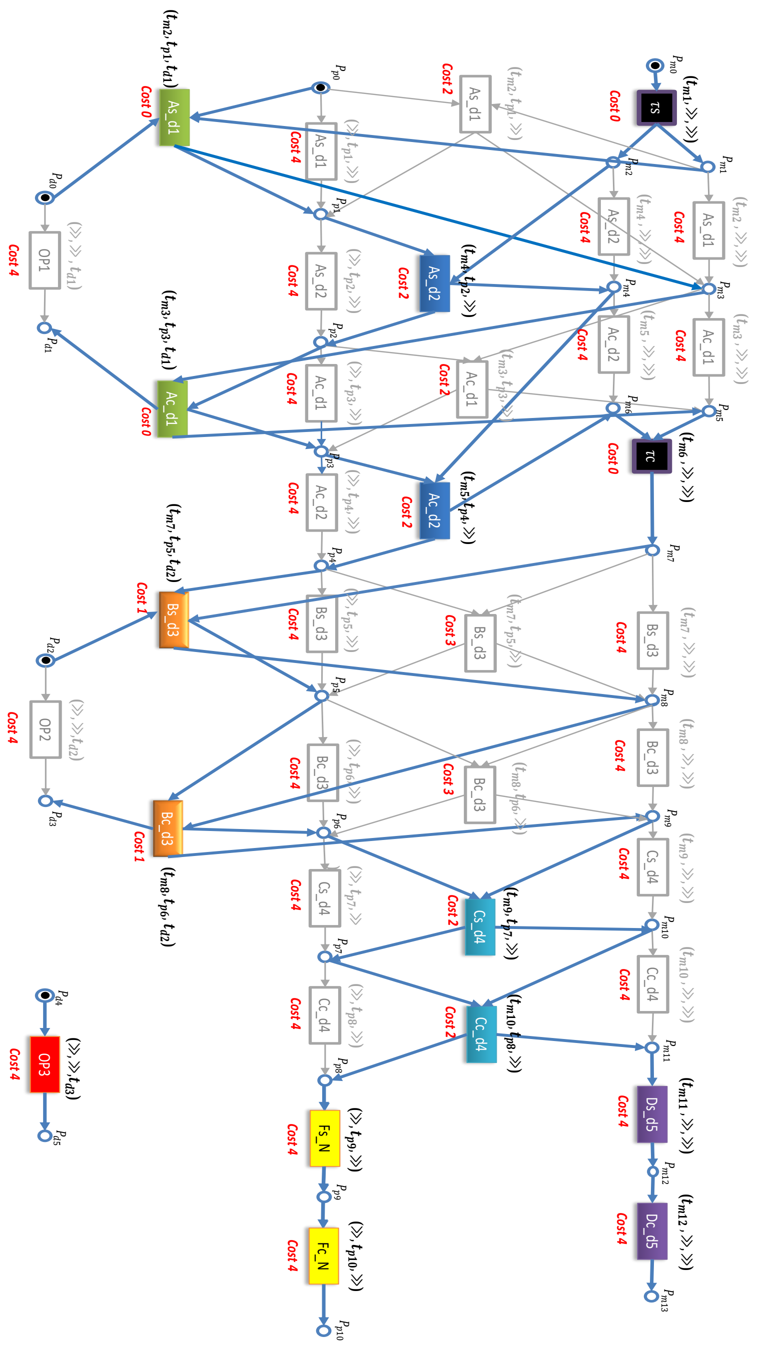


Figure 10: Full run of the synchronous product corresponding to an optimal alignment, assuming a multi-layer cost function for the running example 2

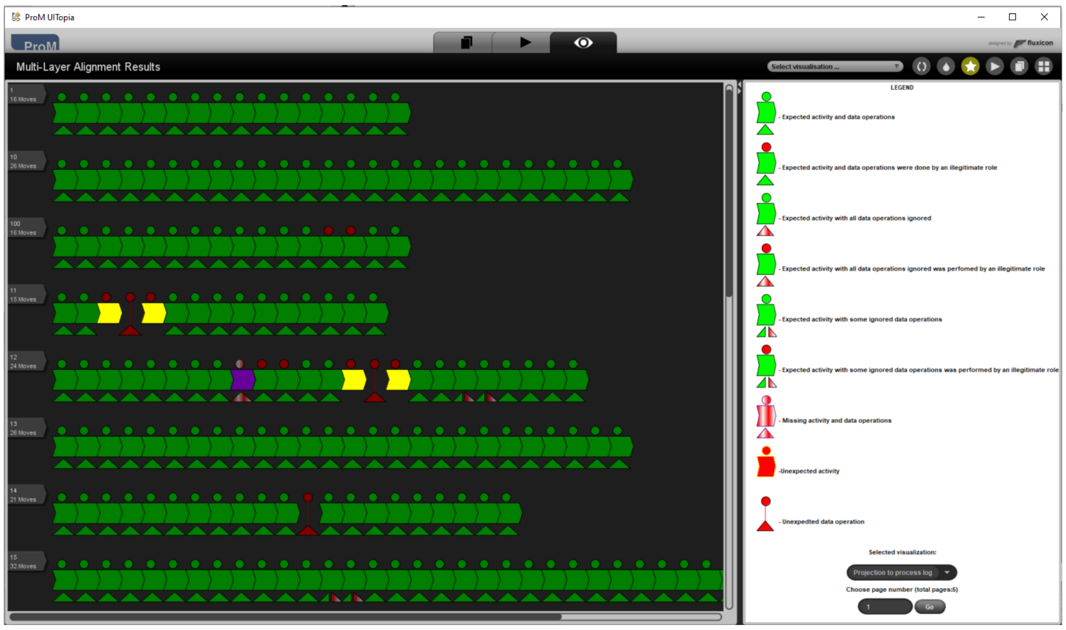

ProM UITopia
ProM
designed by fluxicon
Multi-Layer Alignment Results
Select visualisation ...
1 16 Moves
10 26 Moves
100 16 Moves
11 15 Moves
12 24 Moves
13 26 Moves
14 21 Moves
15 32 Moves
LEGEND
- Expected activity and data operations
- Expected activity and data operations were done by an illegitimate role
- Expected activity with all data operations ignored
- Expected activity with all data operations ignored was perfomed by an illegitimate role
- Expected activity with some ignored data operations
- Expected activity with some ignored data operations was performed by an illegitimate role
- Missing activity and data operations
-Unexpected activity
- Unexpedted data operation
Selected visualization:
Projection to process log
Choose page number (total pages:5)
1
Go

Figure 11: Screenshot of our MLA Tool - Projection to process log visualisation

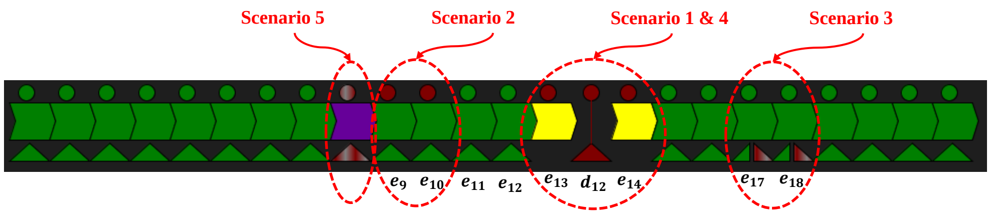


Figure 12: Projection to process log visualisation for running example 1

approaches can only detect the violations shown by purple and yellow colors while our approach is able to identify a larger range of hidden deviations. In the developed tool, by clicking on model moves, purple color is changed to red/white color to indicate missing behavior in all three perspectives of the process. By clicking on the moves on process log, the yellow color is changed to red to mark unexpected behavior from privacy and control-flow points of view.

Returning to our running example 1 in Section 2, the optimal alignment we get from the synchronous product is translated back into a multi-perspective alignment as shown in Figure 12.

Our technique for multi-perspective alignment is able to detect all of the violations discussed in the scenarios of Section 2 as explained next.

Scenario 2 can be recognized in Fig. 12 by observing that basic lab test (started in e_9 and completed in e_{10}) happened in right order and all of its related data operations are expected according to the data model represented in Table 1. However, this activity along with its data operations were done by the illegitimate role “nurse”.

The visualisation points that there is an ignored data operation when the doctor per-

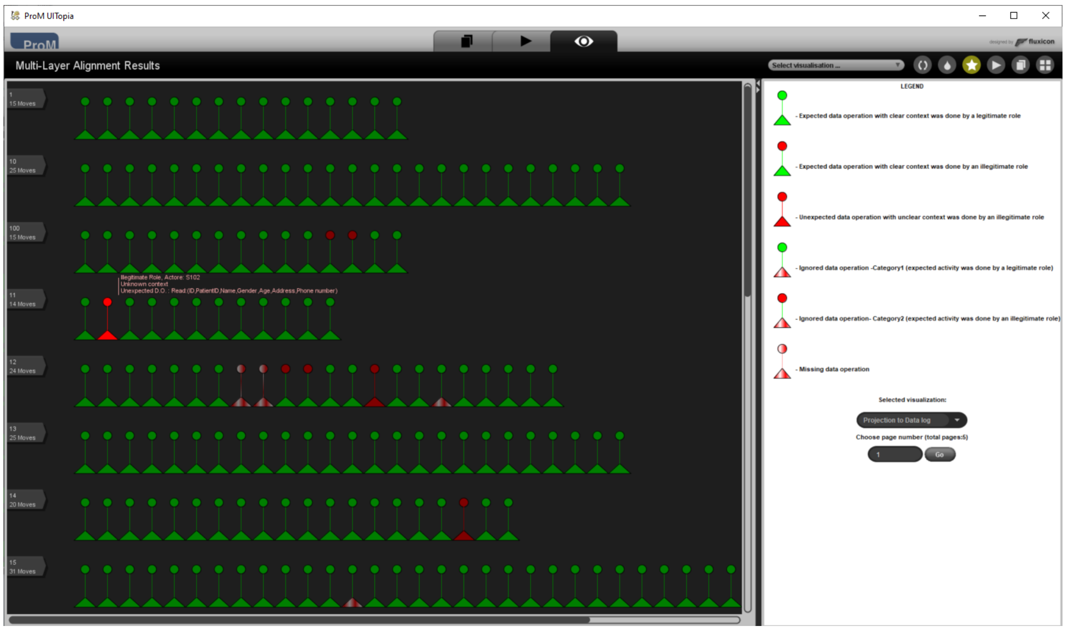

ProM UITopia
ProM
designed by fluxicon
Multi-Layer Alignment Results
Select visualisation ...
1
15 Moves
10
25 Moves
100
15 Moves
11
14 Moves
12
24 Moves
13
25 Moves
14
20 Moves
15
31 Moves
Illegitimate Role, Actore: S102
Unknown context
Unexpected D.O. : Read:(ID,PatientID,Name,Gender,Age,Address,Phone number)
LEGEND
- Expected data operation with clear context was done by a legitimate role
- Expected data operation with clear context was done by an illegitimate role
- Unexpected data operation with unclear context was done by an illegitimate role
- Ignored data operation -Category1 (expected activity was done by a legitimate role)
- Ignored data operation- Category2 (expected activity was done by an illegitimate role)
- Missing data operation
Selected visualization:
Projection to Data log
Choose page number (total pages:5)
1
Go

Figure 13: Screenshot of our MLA Tool- Projection to Data log visualisation

formed the activity treatment prescription (tr) recorded by start and complete events e_{17} and e_{18} (scenario 3 in Section 2).

Secondary usage of data in scenario 4 can be recognized in Fig. 12 by observing that after some unjustified access to patient medical history, recorded by op_{12}, the doctor performed a clinical trial (recorded in e_{13} and e_{14}), which based on the process model (Fig. 1) does not contribute to the fulfillment of the treatment process. The visualization illustrates these two violations as unexpected activity (e_{13} and e_{14}) and unexpected data operation (op_{12}).

Finally, missing activity and data operations in scenario 5 can be detected in Fig. 12 by observing that evaluation of test results (recorded by e_{11} and e_{12}) happened without previously executing advanced test (at) by lab specialist.

As another view of the multi-layer alignment results, we developed "Projection to data log" visualization. Fig 13 shows a screenshot of the projection to data log visualisation for the running example 1.

The goal of this visualisation is to indicate violations related to data layer. Expected data operations are shown in green color. Red color indicates unexpected data operations and highlighted white/red color marks missing or ignored data operations. Furthermore, "Projection to Data Log" visualisation provides an answer to the important privacy rule,"who performed which data operation for which purpose?" by indicting the context of each data event. All data operations performed with unclear or secondary purposes are marked by red color in this visualisation.

Fig 14 shows "Projection to Data log" visualisation for our running example 1. The approach distinguished two missing data operations that should be executed by the role lab expert in the context of taking advanced test (depicted in highlighted red/white color between op_7 and op_8) and one ignored data operation (Update(TreatmentPlan)) during performing treatment prescription (tr) by the role doctor in the context of visit activity (depicted as highlighted red/white triangle and green circle between op_{14} and op_{15}). The re-

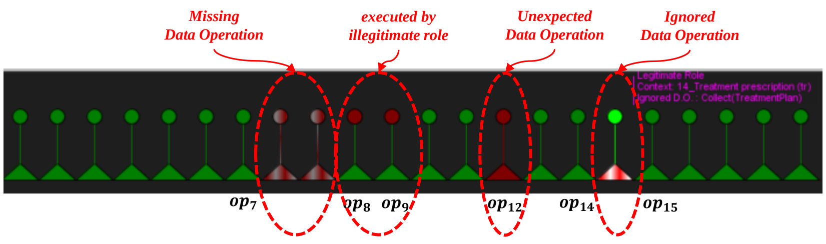


Figure 14: Projection to Data log visualisation for running example 1

sults indicate that data event op_{12} in red color was executed with unclear context by the role doctor. Two data events op_8 and op_9 show that corresponding data operations were executed by the illegitimate role nurse. All other data operations were expected by the data model and executed by legitimate roles in a clear context.

6 Evaluation

We evaluated the proposed approach using the developed MLA tool and synthetic logs. The aim of the evaluation with the synthetic event logs was to perform controlled experiments with known ground truth and assess the accuracy of the obtained diagnostics. We used a real-life process to show that the approach provides useful insights into deviations and is robust to logs and models with real-life complexity. For instance, we show that our implementation can deal with loops in the process model and can easily handle event logs containing considerable trace length. The experiments were performed using a machine with 2.8 GHz Intel Core i7 processor and 16 GB of memory.

6.1 Experimental Setup

For the experiments with synthetic data, we designed the process model in Fig. 1 using CPN tools[5]. The CPN tool provides a graphical user interface that allows constructing, simulating, and analyzing process models in a Petri net format. CPN is a graphical modeling language used for describing and analyzing the behavior of discrete event systems. It is widely used in software engineering and workflow management, and other areas. The tool allows users to construct models using a graphical editor, simulate the behavior of the system under various conditions, and analyze the results using various metrics. By applying this tool, we designed the healthcare treatment process and, via the simulation generated 10,000 process traces and 10,000 data traces consisting of 186,798 process events and 176,798 data operations, respectively. We started the experiments with these logs, in which no random noise was introduced in the process and data traces (experiment 0 with clean process and data logs in Table 4). Generated process and data event logs contain traces of considerable length (between 10 and 32 events per trace). A fragment of the process log is presented in Table 2. This table shows all activities that were performed by different roles in the healthcare treatment process for the case 101. Table 3 presents a fragment of the database log showing all data accessed in the healthcare treatment process for the same case in table 2.

[5]http://cpntools.org

Table 2: A fragment of the process log showing all activities performed in the healthcare treatment process for case 101.

CaseID	EventID	ID	Activity	Actor	Type	Time
101	e1	1	Identify patient (ip)	R101	START	2020-01-02T08:44:23
101	e2	1	Identify patient (ip)	R101	COMPLETE	2020-01-02T09:34:14
101	e3	2	Admission (ad)	R101	START	2020-01-02T10:34:08
101	e4	2	Admission (ad)	R101	COMPLETE	2020-01-02T11:26:55
101	e5	3	Visit (vi)	D111	START	2020-01-02T11:52:24
101	e6	3	Visit (vi)	D111	COMPLETE	2020-01-02T12:45:43
101	e7	4	Consult request (co)	D121	START	2020-01-02T13:03:48
101	e8	4	Consult request (co)	D121	COMPLETE	2020-01-02T13:54:16
101	e9	5	Inter-colleague consultation (in)	D119	START	2020-01-02T14:20:10
101	e10	5	Inter-colleague consultation (in)	D119	COMPLETE	2020-01-02T15:15:05
101	e11	6	Lab appointment (la)	S101	START	2020-01-02T15:46:19
101	e12	6	Lab appointment (la)	S101	COMPLETE	2020-01-02T16:38:05
101	e13	7	Basic lab test (bt)	LE104	START	2020-01-02T16:56:24
101	e14	8	Advanced tests (at)	LE205	START	2020-01-02T17:10:39
101	e15	7	Basic lab test (bt)	LE104	COMPLETE	2020-01-02T17:10:39
101	e16	8	Advanced tests (at)	LE205	COMPLETE	2020-01-02T18:10:01
101	e17	9	Evaluate (ev)	LD102	START	2020-01-02T18:39:14
101	e18	9	Evaluate (ev)	LD102	COMPLETE	2020-01-02T19:38:31
101	e19	10	Visit (vi)	D110	START	2020-01-02T19:57:47
101	e20	10	Visit (vi)	D110	COMPLETE	2020-01-02T20:50:25
101	e21	11	Treatment prescription (tr)	D116	START	2020-01-02T21:17:18
101	e22	11	Treatment prescription (tr)	D116	COMPLETE	2020-01-02T22:16:30
101	e23	12	Discharge (di)	D103	START	2020-01-02T22:48:10
101	e24	12	Discharge (di)	D103	COMPLETE	2020-01-02T23:36:14
101	e25	13	Billing (bi)	A105	START	2022-01-02T23:36:14
101	e26	13	Billing (bi)	A105	COMPLETE	2022-01-03T00:34:05

In order to assess the capability of detecting different kinds of deviations, we manipulated the generated process traces and data traces. In particular, in the first series of experiments, we conducted six experiments in which different kinds of noise were produced randomly in the process and data logs in order to simulate the different violations. By adding or removing some process events and/or data operations, changing the resource attribute (actor) of events in the process trace and data trace with an actor playing a not-allowed role, we simulated several scenarios including the scenarios discussed in Section 2. Experiments 1-6 add noise to the clean process and data logs. In experiment 1, we introduced noise by randomly selecting 5% of events in the process log and related data operations in the data log and changing their actors to users playing different roles. This helped simulate a scenario where an expected activity and data operation were performed by an actor with the wrong role. In experiment 2, we inserted noise by randomly selecting 5% of executed data operations in the data log and removing them. This aimed to simulate a scenario where certain data operations were ignored (Scenario 3). In experiment 3, we introduced noise by randomly adding 500 events to different traces in the process log and inserting 500 data operations with the same actors in the related data traces in the data log. This aimed to simulate unexpected activities and data operations (Scenario 4). In experiment 4, we introduced noise by randomly selecting 5% of events in the process log and related data operations in the data log and removing them. This aimed to simulate skipped tasks (activity and data operations) (Scenario 5). In experiment 5, by adding unauthorized data operations, we inserted 5% of random noise to the clean data log to simulate one of the situations of secondary usage of data other than the case discussed in scenario 4 of Section 2.

Table 3: A fragment of the database log showing all data accessed in the healthcare treatment process for case 101.

CaseID	EventID	Operation	Actor	Type	Time
101	op1	Read:(ID)	R101	COMPLETE	2020-01-02T09:02:12
101	op2	Collect:(AddmissionID)	R101	COMPLETE	2020-01-02T11:03:28
101	op3	Read:(ID,PatientID,Name,Gender,Age,Address, Phone number)	R101	COMPLETE	2020-01-02T11:06:52
101	op4	Read:(AddmissionID,PatientID,MedicalHistoryID)	D11	COMPLETE	2020-01-02T12:09:10
101	op5	Collect:(VisitID,PrescriptionID)	D111	COMPLETE	2020-01-02T12:22:06
101	op6	Collect:(ConsAppointment)	D121	COMPLETE	2020-01-02T13:24:01
101	op7	Read:(AddmissionID,PatientID)	D121	COMPLETE	2020-01-02T13:30:47
101	op8	Read:(AddmissionID,PatientID,VisitID, PrescriptionID,TestResultID,MedicalHistoryID)	D119	COMPLETE	2020-01-02T14:38:30
101	op9	Collect:(VisitID,CPrescriptionID)	D119	COMPLETE	2020-01-02T14:43:50
101	op10	Collect:(LabAppointment)	S101	COMPLETE	2020-01-02T16:16:36
101	op11	Read:(AddmissionID,PatientID)	S101	COMPLETE	2020-01-02T16:16:44
101	op12	Collect:(BLabPID)	LE104	COMPLETE	2020-01-02T17:23:30
101	op13	Read:(AddmissionID,PatientID,PrescriptionID)	LE104	COMPLETE	2020-01-02T17:27:56
101	op14	Read:(AddmissionID,PatientID,PrescriptionID)	LE205	COMPLETE	2020-01-02T17:37:03
101	op15	Collect:(ALabPID)	LE205	COMPLETE	2020-01-02T17:41:52
101	op16	Read:(AddmissionID,PatientID,PrescriptionID, BLabPID,ALabPID)	LD102	COMPLETE	2020-01-02T19:04:41
101	op17	Collect:(TestResultID)	LD102	COMPLETE	2020-01-02T19:12:28
101	op18	Read:(AddmissionID,PatientID, MedicalHistoryID)	D110	COMPLETE	2020-01-02T20:24:56
101	op19	Collect:(VisitID,PrescriptionID)	D110	COMPLETE	2020-01-02T20:30:18
101	op20	Read:(AddmissionID,PatientID,VisitID, MedicalHistoryID)	D116	COMPLETE	2020-01-02T21:43:54
101	op21	Collect:(TreatmentPlan)	D116	COMPLETE	2020-01-02T21:44:07
101	op22	Read:(AddmissionID,PatientID)	D103	COMPLETE	2020-01-02T23:07:18
101	op23	Collect:(Confirmation)	D103	COMPLETE	2020-01-02T23:08:49
101	op24	Read:(AddmissionID,PatientID,PaymentID)	A105	COMPLETE	2020-01-02T23:53:04
101	op25	Collect:(PaymentReceipt)	A105	COMPLETE	2020-01-03T00:05:07

This scenario shows the data was accessed directly from the database. In experiment 6, we aimed to simulate the scenario which represents the unsuccessful attempt of a not-allowed role to access data by performing an activity. In this scenario, the executed activity can be observed in the process log without successful links to the data operations in the data log. We inserted 5% of noise of this kind of violation in both process and data logs, by updating the actor of random process events with an actor playing a not-allowed role and removing related data operations from the data log.

In the second series of experiments, we conducted the experiments in which all kinds of noise happened in the trace level (experiment 7) or in the log level (experiment 8). In experiment 7, we produced 500 trace ids randomly. Then we inserted all deviations related to experiments 1 to 6 in the random process and data traces. As a result, in this setting, we produced 5% of noise where process instances contain all kinds of intra and inter-layer deviations in the level of the trace. In experiment 8, in order to create different kinds of random noise in the level of the log, we inserted the same kind of noise for experiments 1 to 6 through six iterations starting with the process and data log with no noise. In each iteration, we inserted 5% of noise into the logs randomly. In this process to create the logs, 4% of noise was randomly added to traces that already contained other kinds of noise. As a result, in this setting, we produced 26% of noise in the level of the log, where process instances contain one or more kinds of intra- and inter-layer deviations. Table 4 summarizes the differences of conducted experiments in terms of the type of deviation and the perspectives that the deviation happened.

6.2 Experimental Results

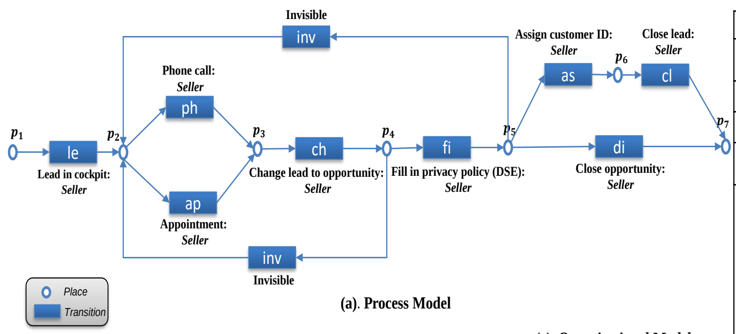
To assess the approach's capability of detecting different kinds of deviations and the accuracy of obtained results, we computed precision, recall, and F_1-measure [16]. Precision is computed as the fraction of detected deviations that are actual deviations, whereas recall is computed as a fraction of the inserted deviations that are detected. The F_1-measure is the harmonic mean of precision and recall. In each experiment, the recall was known since deviations were introduced artificially.

Table 4: The Result of Experiments. MA indicates missing activity; UA indicates unexpected activity; NAO indicates not allowed data operation; MO_AR indicates expected activity with missing data operation done by an allowed role; MO_NAR indicates an activity with missing data operation done by not allowed role; EO_NAR indicates expected activity and data operation done by not allowed role; OK indicates expected activity and data operation done by an allowed role.

	MA	**UA**	**NAO**	**MO_AR**	**MO_NAR**	**EO_NAR**	**OK**	**Comp.T(S)**
deviation happened in	All Three layers	Process layer	data layer	data layer	data layer & privacy layer	privacy layer	-	
Legal Move	Move on Model	Move on Process Log	Move on Data Log	Partially Sync. Move	Partially Sync. Move with Penalty cost	Totally Sync. Move with Penalty cost	Totally Sync. Move	
	P-R-F1	**P- R- F1**	**P- R- F1**	**P- R- F1**	**P- R- F1**	**P- R- F1**	**P- R- F1**	
Experiment0 (0% noise)							1.00-1.00-1.00	46
Experiment1 (5% noise)						1.00-1.00-1.00		48
Experiment2 (5% noise)				1.00-1.00-1.00				49
Experiment3 (5% noise)		1.00-1.00-1.00	1.00-1.00-1.00					79
Experiment4 (5% noise)	1.00-1.00-1.00							74
Experiment5 (5% noise)			1.00-1.00-1.00					58
Experiment6 (5% noise)					1.00-1.00-1.00			75
Experiment7 (5% noise)	1.00-1.00-1.00	1.00-1.00-1.00	1.00-1.00-1.00	1.00-1.00-1.00	1.00-1.00-1.00	1.00-1.00-1.00		65
Experiment8 (26% noise)	1.00-1.00-1.00	1.00-1.00-1.00	1.00-1.00-1.00	1.00-1.00-1.00	1.00-1.00-1.00	1.00-1.00-1.00		346

Table 4 reports the results of the experiments for different levels of noise. Overall, the results show high precision and recall. Considering all experiments, we conclude that the approach is able to detect all deviations that happened in one, two, or all three combined process perspectives (control-flow, data and privacy policy). Moves on model, which were detected by the implemented approach in 500 process instances of experiment 2, showed that the deviation happened in all three perspectives. Whereas totally synchronous moves in the experiments represented that the executed behavior complied with modeled behavior in all three perspectives of the process (i.e. experiment 0). Experiment 5 (which detected 500 moves on data log) clearly indicated that the approach computes the data layer deviations independent from the control-flow perspective. Experiments 0, 1, 2 and 6 indicate that the approach considers privacy compliance in addition to the control-flow and data perspectives in the conformance analysis. In particular, experiment 1 (which detected 500 totally synchronous moves with penalty cost) proved that the approach is able to detect the hidden deviations where executed behavior and modeled behavior completely complied from the process and data perspectives, but the non-conformity was related to the privacy perspective.

In addition to detecting multi-layer deviations, the experiments remark that the approach is capable to reconstruct and provide the link between executed activities in the process



(a). Process Model

(b). Data Model

Activity	Data Operations
Lead in cockpit	d1: Read(CNAME, EMAIL, PHONE, FAX, ADDRESS)
Phone call	d2: Read(PHONE)
Appointment	d3:Read(EMAIL)
Change lead to opportunity	d4:Read(LEADTYPE)
Fill in privacy policy (DSE)	d5: Update(NO_DAIMLER_DSE) d6: Update(ADDRESS_DSE) d7: Update(PHONE_DSE) d8: Update(EMAILE_DSE) d9: Update(SMS_DSE) d10: Update(FAX_DSE)
Assign customer ID	d11: Create(CUSTID)
Close opportunity	d12: Read(OPPID)
Close lead	d13: Read(LEADID)

(c). Organizational Model

Role	Actor (Resource)
Admin	user1
Seller	user2

Figure 15: Lead management process, data and organisational model

layer and performed data operations in the data layer.

We conclude this section by briefly reporting on the execution time. For each experiment, we measured the time needed to construct and compute the multi-layer alignment. Finding optimal alignment for the data and process logs, each containing 10,000 recorded traces, with inserted 26% of all kinds of noise took almost 6 minutes. The average computation time for the experiments with the logs containing 5% of one kind of noise was 77 seconds.

6.3 Further Evaluation

To assess the practical feasibility of the approach, we also performed an evaluation with an automotive car dealership. The lead management process model (Fig. 15(a)) along with corresponding data model (Fig. 15(b)) were provided by a process analyst of the company. The process model consists of 10 transitions: 8 transitions with unique labels, plus 2 invisible transitions. The data model contains 13 data operations, which are read, create or update by the activities when being executed. In the organisational model (Fig. 15(c)) there are two roles: admin and seller. All activities in the lead management process model were supposed to be performed by the role seller.

We were also provided with an event log that recorded the execution of 78 real instances of such process. By splitting activities and data operations from the original data set, we obtained a process event log and a data log. The resulting process event log and data log contain 1,014 process events and 892 data events, respectively. We applied our approach to compute multi-perspective alignment to check conformance of the recorded executed behavior in these logs with modeled behaviors designed in process, data and organisational models. On average, the construction of optimal multi-perspective alignment from the data and process logs and the prescribed behavior required 0.44 s per case.

In total, we identified 27 missing data operations, 11 ignored data operations and 42 data operations without clear context. The results also showed 27 activities were skipped during the whole process instances. 50 unexpected process events (start and complete events related to 25 activities) were performed by the actors and while performing 16 activities the actors ignored executing expected data operations. In all identified deviations, the tool provided detailed information about the actors and the legitimization of their roles. In one case the admin of the system has accessed the phone number of the customer to call the customer while this activity was supposed to be performed by the role seller.

It is worth mentioning that, in the design of our approach, we placed a strong emphasis on its generality with respect to the process and data models. This implies that our technique is not dependent on any particular set of features that may be present in the data model. It is designed to be broadly applicable across a wide range of models. Moreover, as illustrated in Table 3, our technique is effective even in cases where the data has been anonymized and it can still identify valuable insights such as spurious data access, ignored data operations, skipped tasks, and unexpected activities.

In summary, our approach is designed to be flexible and widely applicable, enabling valuable insights to be extracted from data models that may differ in their features and levels of anonymization.

7 Related Work

Process mining is a set of techniques that aim at analyzing business process execution data recorded in event logs. We limit related work to the research approaches most related to our contribution to the field of conformance checking.

In Adriansyah *et.al.* [8], alignment algorithms are proposed as a robust approach to conformance checking based on the use of a cost function. This work focuses only on process perspective relating observed events in the event log to runs of the model. Besides the control-flow, there are also other perspectives like data or resources that are often crucial for conformance analysis. Few approaches have investigated how to include these perspectives in the conformance analysis. De Leoni *et.al.* [9] extend the alignment approach to bring other perspectives' impact in the identification of non-conformity. This approach considers data, resource, and time as data attributes of process events. Thus, control-flow is aligned first, and then data are considered. Mannhardt *et.al.* [10] extend the work in [9] to propose a more balanced approach using data-aware Petri net as the prescribed model and check executed behaviors in the process log with respect to the values of the variables in the guards in addition to control-flow conformance. Both approaches are unable to consider the three perspectives separately since these methods give priority to the control-flow. Accordingly, some important violations such as missing data operations or not allowed data access can be missed in the alignment results.

Alizadeh *et.al.* [11] proposed an approach for linking data and process perspectives for conformance analysis. Similarly to [9] and [10], they extend the alignment approach to handle the data perspective in which control-flow is aligned first and then data are considered. In contrast to the proposed approaches in [9] and [10], Alizadeh *et.al.* [11] aligned data and process perspectives independently. They applied a CRUD matrix that relates process activities to data operations and defined two criteria functions to link data operations in the data traces and events in the process traces.

We have extended the work in [11] and added privacy perspective in addition to process and data perspectives. To this end, we integrate the activities with corresponding roles in the process model in addition to using organisational model. Therefore, our approach can provide more comprehensive diagnostics than [11]. Similar to [11], we use a data model that relates process activities to data operations. However, we employed the data model in a completely different way to bring data perspective into conformance analysis. In [11], the approach applies a data model along with two criteria functions to link data operations in data traces with events in process traces. They performed this step in post-processing (after alignment computation) locally for each event in the alignment trace to find the deviations related to the data layer. This is the reason why their approach is not able to identify all the deviations correctly. For instance, in the presence of concurrent process events, a data operation can be linked to different process events with the same activity name. We solved this problem globally by allowing the alignment algorithm to find the best match. In contrast to [11], we use the data model in the pre-processing step to enrich the process model with related data operations in order to model prescribed behavior from all three perspectives. By constructing it, our approach is able to link data and process layer in a more robust way.

A large body of literature is related to privacy-preserving process/data mining, e.g. [17, 18, 19, 20, 21]. They are not compared here since they consider privacy issue at design time to minimise privacy risks while maximising data utility for analysis. Furthermore, they do not consider the run-time perspective of business process management.

To the best of our knowledge, the work in this paper is the first work that proposes a novel technique for computing alignment by considering all control-flow, data, and privacy perspectives of a business process at the same time without giving priority to one perspective.

8 Conclusion

As regulations like GDPR impose purpose control over data processing, in this work we presented a new method for multi-perspective conformance checking. In particular, we described our approach based on a new multi-layer alignment and cost function. We discussed that, by considering all perspectives without prioritizing any, our approach is able to find the context of data access in addition to detect hidden deviations between control-flow, data, and privacy perspectives of business processes. Moreover, through some scenarios we showed that by reconciling the process, data and privacy aspects, our technique can detect spurious data access and identify privacy infringements where data have been processed for unclear or secondary purposes by an authorised role. With the aforementioned contributions, we believe we extend the state-of-the-art techniques, not only in the Process Mining (or Conformance Checking) field, but also in the Data Privacy field, as we showed how the context of data process can be audited.

As proof of concept, we implemented the approach in the open source ProM framework. An evaluation of the proposed approach has been carried out using real logs of a car dealer lead management process and synthetic logs generated from the simulation of a healthcare process. The evaluation showed the applicability of our implementation to real-life complexity. Current approaches for multi-perspective conformance checking have several limitations. All of them give the priority to the control-flow perspective and do not consider all perspectives together to find the deviations. The experiments confirmed that our approach is able to provide more accurate diagnostics of deviations than control-flow based conformance checking approaches. The evaluation results also implied that the proposed approach allows the user to identify violations that cannot be detected by taking into consideration only one or two perspectives.

In future work, we plan to provide some measures to rank non-conforming behaviors according to their criticality to guide users towards an in depth identification of problems and risky behaviors in the execution of business processes. Extending the application of the approach and making it suitable for online process mining and monitoring would be another direction of future work.

Reproducibility

The inputs required to reproduce the experiments can be found at
https://doi.org/10.4121/8683fc1a-aca1-447b-aba4-8d7806a9977f

Acknowledgements

The author has received funding within the BPR4GDPR project from the European Union's Horizon 2020 research and innovation programme under grant agreement No 787149.

References

[1] M. Petković, D. Prandi, N. Zannone, Purpose control: Did you process the data for the intended purpose?, in: W. Jonker, M. Petković (Eds.), Secure Data Management, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 145–168.

[2] M. de Leoni, W. Van der Aalst, B. van Dongen, Data- and resource-aware conformance checking of business processes, in: W. Abramowicz, D. Kriksciuniene, V. Sakalauskas (Eds.), Business Information Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 48–59.

[3] S. Zhang, L. Genga, H. Yan, X. Lu, U. Kaymak, Towards multi-perspective conformance checking with fuzzy sets, in: Workshop on Data Fusion for Artificial Intelligence (DAFUSAI 2020), 2020.

[4] W. Van der Aalst, Process Mining: Discovery, Conformance and Enhancement of Business Processes., Springer, Verlag Berlin Heidelberg, 2011.
URL https://doi.org/10.1007/978-3-662-49851-4

[5] A. Rozinat, W. Van der Aalst, Conformance checking of processes based on monitoring real behavior, Information Systems 33 (1) (2008) 64–95.

[6] M. Weidlich, A. Polyvyanyy, N. Desai, J. Mendling, Process compliance measurement based on behavioural profiles, in: B. Pernici (Ed.), Advanced Information Systems Engineering, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 499–514.

[7] J. E. Cook, A. L. Wolf, Software process validation: Quantitatively measuring the correspondence of a process to a model, ACM Trans. Softw. Eng. Methodol. 8 (2) (1999) 147–176.
URL https://doi.org/10.1145/304399.304401

[8] A. Adriansyah, B. van Dongen, W. Van der Aalst, Towards robust conformance checking, in: M. zur Muehlen, J. Su (Eds.), Business Process Management Workshops, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 122–133.

[9] M. de Leoni, W. Van der Aalst, Aligning event logs and process models for multi-perspective conformance checking: An approach based on integer linear programming, in: F. Daniel, J. Wang, B. Weber (Eds.), Business Process Management, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 113–129.

[10] F. Mannhardt, M. de Leoni, H. Reijers, W. Van der Aalst, Balanced multi-perspective checking of process conformance, BPMcenter. org, 2014.

[11] M. Alizadeh, X. Lu, D. Fahland, N. Zannone, W. Van der Aalst, Linking data and process perspectives for conformance analysis, Computers and Security 73 (2018) 172–193.

[12] A. S. Mozafari Mehr, R. M. de Carvalho, B. van Dongen, Detecting privacy, data and control-flow deviations in business processes, in: S. Nurcan, A. Korthaus (Eds.), Intelligent Information

Systems, Springer International Publishing, Cham, 2021, pp. 82–91.
URL https://doi.org/10.1007/978-3-030-79108-7_10

[13] W. Van der Aalst, K. van Hee, Workflow Management: Models, Methods, and Systems, MIT Press, 2002.

[14] J. Carmona, B. van Dongen, A. Solti, M. Weidlich, Conformance Checking Relating Processes and Models, Springer, Cham, Springer Nature Switzerland AG 2018, 2018.

[15] A. S. Mozafari Mehr, R. M. de Carvalho, B. van Dongen, Mla : A tool for multi-perspective conformance checking of business processes, in: Proceedings of the International Conference on Process Mining - Demo Session, 2021.

[16] J. W. Perry, A. Kent, M. M. Berry, Machine literature searching x. machine language; factors underlying its design and development, American Documentation 6 (4) (1955) 242–254.

[17] F. Mannhardt, A. Koschmider, N. Baracaldo, M. Weidlich, J. Michael, Privacy-preserving process mining differential privacy for event logs, Business & Information Systems Engineering 61 (2019) 1–20.

[18] A. Pika, M. T. Wynn, S. udiono, A. H. M. Ter Hofstede, W. Van der Aalst, H. A. Reijers, Privacy-preserving process mining in healthcare, International journal of environmental research and public health 17 (2020).

[19] C. C. Aggarwal, Data Mining: The Textbook, Springer, Cham, 2015.

[20] J. Michael, A. Koschmider, F. Mannhardt, N. Baracaldo, B. Rumpe, User-centered and privacy-driven process mining system design for iot, in: C. Cappiello, M. Ruiz (Eds.), Information Systems Engineering in Responsible Information Systems, Springer International Publishing, Cham, 2019, pp. 194–206.

[21] M. Rafiei, W. Van der Aalst, Privacy-preserving data publishing in process mining, CoRR abs/2101.02627 (2021). URL arXiv:2101.02627.