Evaluating and Predicting Overall Process Risk Using Event Logs

A. Pika[a,*], W.M.P. van der Aalst[b,a], M.T. Wynn[a], C.J. Fidge[a], A.H.M. ter Hofstede[a,b]

[a] *Queensland University of Technology, Brisbane, Australia*
[b] *Eindhoven University of Technology, Eindhoven, The Netherlands*

Abstract

Companies standardise and automate their business processes in order to improve process efficiency and minimise operational risks. However, it is difficult to eliminate all process risks during the process design stage due to the fact that processes often run in complex and changeable environments and rely on human resources. Timely identification of process risks is crucial in order to insure the achievement of process goals. Business processes are often supported by information systems that record information about their executions in event logs. In this article we present an approach and a supporting tool for the evaluation of the *overall process risk* and for the prediction of process outcomes based on the analysis of information recorded in event logs. It can help managers evaluate the overall risk exposure of their business processes, track the evolution of overall process risk, identify changes and predict process outcomes based on the current value of overall process risk. The approach was implemented and validated using synthetic event logs and through a case study with a real event log.

Keywords: `overall process risk, event log, process risk evaluation, mining process risk`

1. Introduction

In order to deliver desirable outcomes in an efficient manner and minimise operational risks companies often standardise and automate their business operations. Business processes are exposed to different risks that can jeopardise the achievement of process goals in terms of cost, timeliness or the quality of outputs [14]. It is not possible to consider all process execution scenarios and to eliminate all operational risks during the process design stage [27, 30] as processes are often executed in complex [35] and changeable [2] environments. Multiple cases (i.e., process instances) are often processed in parallel, e.g., multiple loan applications can be processed in a bank at the same time. The same resources may be involved in execution of these process instances. Human resources tend to make mistakes and their productivity levels can vary. Human factors are considered to be "unequivocally the single most important element that can affect project success" [31]. A case can be affected by events, either happening in the case itself or external to the case [35]. For example, when a resource is busy processing a complex case, he may neglect other cases assigned to him, hence causing delays. Sometimes an event that is not risky on its own can be risky in combination with other

*Corresponding author
Email address: `anastasiia.pika@hdr.qut.edu.au` (A. Pika)

events, e.g., consider the arrival of an urgent case – one such case will typically not be a problem, but when multiple urgent cases arrive within a short period of time, outcomes (e.g., quality and duration) of all cases active during this period may be affected.

Furthermore, it can be very costly to address all known process risks during the process design stage. For example, in order to decrease the likelihood of human mistakes, the result of every completed task instance can be checked by a human resource not involved in its execution ("four-eyes principle" [39]). However, this strategy is very time-consuming and is too costly to be applied in non-critical processes. Although it is not always possible to build mechanisms into a process that address all known process risks during the process design stage, it is important to evaluate the level of overall process risk during the execution of a process.

Effective risk management is crucial for organisations [8]. ISO Guide 73:2009 defines risk as the "effect of uncertainty on objectives" where effect is "a deviation from the expected – positive and/or negative" [13]. Risk identification is a critical task and one of the most challenging in the risk management process [29]. Although many risk management approaches provide high-level guidelines about risk identification strategies, they do not provide any guidelines on how to operationalise them [21, 29]. Recently a few approaches have been proposed that allow us to identify some process-related risks [3, 5, 24, 25]. Existing approaches typically analyse characteristics of individual process instances and predict risks on the process instance level, though "the handling of cases is influenced by a much broader context" [35]. They also do not consider the fact that process risk can change over time. As managing risks in individual process instances can be very costly, for non-critical processes companies may be more interested in managing overall process risk rather than dealing with risks in individual process instances. For example, a manager may wish to be alerted only when overall process risk reaches some predefined threshold rather than receiving "delay likelihood" notifications for individual process instances.

Up to now insufficient attention has been paid to the problem of evaluation of overall process risk [4], i.e., the risk that threatens the achievement of overall process goals (e.g., completing the majority of cases within a given time period or within a given budget) and can be caused by different process-related risk factors in multiple process instances. Hence, our first research question in this paper is: *how can we evaluate overall process risk at a given point in time considering different risk factors across all running process instances?* A method for evaluation of overall process risk at different points in time can help managers identify changes in overall process risk. In the case of a significant increase of overall process risk, managers may wish to look more closely at business operations in order to investigate sources of the risk increase and take corresponding actions. For example, they can discover that the overall risk increased because specific tasks in a process are often repeated or delayed. Consequently, they may decide to provide training to resources that execute these tasks or to hire additional employees.

When multiple events that increase process risks (we refer to them as risky process behaviours) happen in a process within a short period of time, many cases that are active during this period may not achieve their goals (e.g., in terms of time or quality). Let us consider as an example a service desk organisation. Within a few hours many urgent requests may be lodged, a few complex incidents may be re-opened and a special type of expertise may be

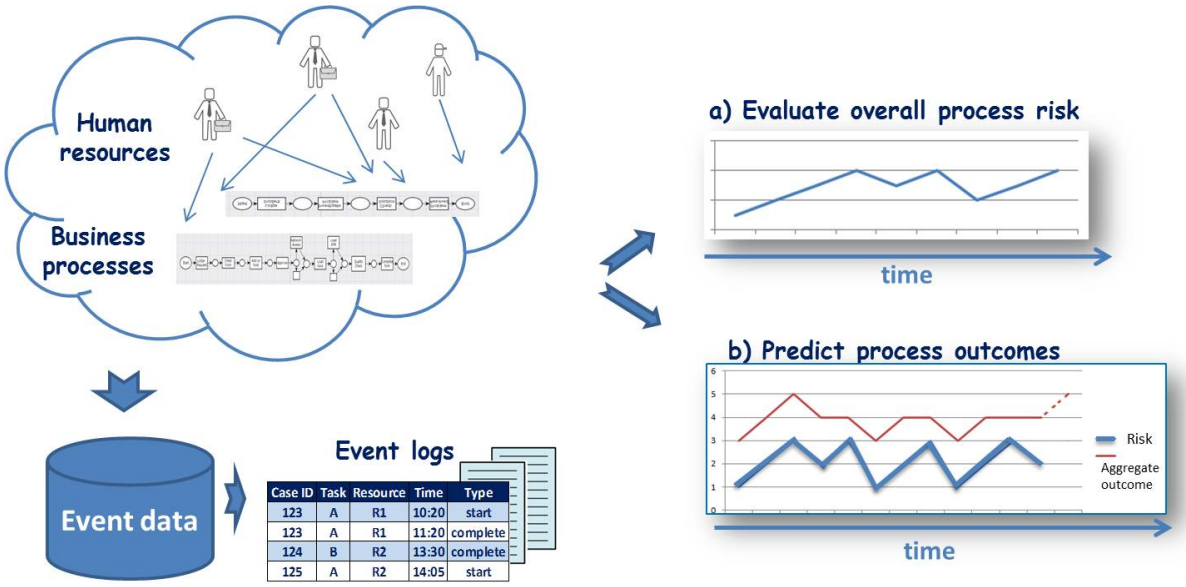


Figure 1: Evaluating overall process risk and predicting process outcomes

required for some urgent cases. During such periods resources may become abnormally busy. Due to the fact that the resources need to solve many urgent or complex issues it is likely that many cases that are active during this period may be delayed or completed with mistakes. Our second research question we tackle in this paper is: *how can we predict aggregate process outcomes based on the current value of overall process risk?* Examples of aggregate process outcomes include: the percentage of active cases that will not produce a process outcome of a high quality or the number of active cases that will be delayed. Such a prediction may help managers to mitigate significant risks, e.g., when they learn that many running process instances are likely to be delayed or completed with mistakes, they may prioritise tasks or decide to outsource some work.

Companies often use information systems to support executions of their business processes. These information systems record information about process executions, e.g., process tasks that were executed and resources involved in the tasks. Such information can be converted to event logs. Information systems may record different data attributes, e.g., case complexity or task urgency, or these may be derived from raw data. In this paper we present an approach and a supporting tool that allows to 1) evaluate overall process risk and 2) predict aggregate process outcomes based on the analysis of information about process executions recorded in event logs (Figure 1). As the first step, we need to be able to capture risky behaviours of a process. An input to our approach is a process model that models desired process behaviour, i.e., behaviour that does not threaten the achievement of process goals. We consider deviations from such process model as risky process behaviours. Then, we need to identify such risky process behaviours in process instances. To tackle this issue we use an existing technique for replaying an event log on a process model [6]. We devise two measures of overall process risk at a given point in time that are based on the identified risky process behaviours. To track the evolution of the overall process risk over time we generate a time series of the overall process

risk values, visualise them and annotate with automatically detected change points to facilitate the analysis for a user. Time series are often used to model the evolution of different phenomena [16]. (An example of the overall process risk time series is depicted in Figure 15.) To tackle our second research question, we generate overall process risk time series for all processes whose behaviours affect outcomes of a given process as described above; we then extract aggregate process outcome time series from an event log, learn a regression model from past process executions (using extracted overall process risk time series and aggregate process outcome time series), and predict aggregate process outcome based on the current value of overall process risk. The approach is implemented as a plugin of the process mining framework ProM[1]. We demonstrate the approach using an illustrative example and evaluate it using synthetic event logs. We also present a case study on the overall process risk evaluation using a real event log.

2. Related work

A few approaches have been proposed that deal with risks in individual process instances. Conforti et al. [5] proposed a sensor-based approach for real-time monitoring of process risks. Their approach provides a language for definition of process risks, allows to monitor running process instances and to alert managers when the likelihood of some risk in a process instance exceeds a given threshold. In other work, Conforti et al. [3] proposed a method for predicting risks in individual process instances by learning predictors from historical data. Wickboldt et al. proposed a framework that uses process execution data for risk assessment [37]. Risk assessment modules of the framework use information about risk events reported during past activity executions. Grigori et al. proposed a method for exception analysis, prediction, and prevention [9]. Maggi et al. proposed a framework for predictive monitoring of business processes that monitors business goals specified by users using Linear Temporal Logic and attempts to predict the likelihood of achieving these goals [20]. In our own previous work we presented an approach for identification of the risk of case delays [24, 25]. We defined Process Risk Indicators for case delays and methods for recognising their presence in process instances. All these methods focus on risks, exceptions or constraints in individual process instances, while in this paper we focus on overall process risk.

Conforti et al. [4] proposed a recommendation system that recommends actions to resources (i.e., recommends tasks and data values) that minimise risk across all running process instances. Their recommendation system first estimates risks for individual process instances and then finds an optimal distribution of tasks to resources which "minimizes the weighted sum of overall execution time and overall risk across all running instances" [4]. Our goal on the other hand was not to provide an optimal distribution of tasks to resources, but to be able to evaluate overall process risk based on given risky process behaviours, track the evolution of overall process risk over time, and predict aggregate process outcomes based on the current value of overall process risk.

Our approach is inspired by quality improvement programs such as Six Sigma and Lean that originated in the manufacturing world. Six Sigma uses statistical metrics to monitor the quality of outputs and Lean is focused on

[1] www.promtools.org

elimination of waste and reduction of variability [1]. Multiple approaches to measuring process performance have been proposed, e.g., Balanced Scorecard or Statistical Process Control (SPC) [17]. They aim to evaluate how well a process performs based on a set of measures. SPC methods typically focus on monitoring of "final product quality variables" or different manufacturing process characteristics, e.g., temperature or pressure [19]. Similar to quality improvement programs and process performance evaluation approaches our method also reasons about an overall process by analysing its characteristics, but unlike these approaches we focus on overall process risk rather than quality or performance and we provide a method that can mine such risks from event logs.

In recent years multiple Business Intelligence tools have been proposed, however these typically focus on data, rather than processes. An exception is the Business Process Intelligence tool suite proposed by Grigori et al. [10] that helps to manage process execution quality. Their tool suite allows users to analyse, predict, monitor, control and optimise different process behaviors, e.g., they can "examine how many process instances per fiscal quarter are initiated by a certain user" or "the variance of the duration" of a service during weekends [10]. The tool also allows to predict risks of "abnormal behaviors" [10] for running process instances. Like them we also analyse event logs and use user-defined process characteristics as inputs, but we focus on overall process risk, rather than different characteristics of process quality or risks in individual process instances.

A framework for detecting process concept drifts was proposed by Bose et al. [2]. In this article we also consider the fact that processes change over time, but we focus on evaluation of overall process risk and prediction of aggregate process outcomes, rather than detection of process changes.

In summary, the approach we present in this paper is novel because it focuses on the overall process risk and aggregate process outcomes (rather than looking at risks in individual process instances, process performance or quality), it is based on the analysis of information recorded in event logs, and it allows to track the evolution of overall process risk over time.

3. Approach

The main input to our approach is an event log that contains information about process executions. An *event log* is a set of events with different attributes. We assume that events in a log have at least the following attributes: case identifier, task name, resource involved in the task, transaction type (e.g., *start* or *complete*) and a time stamp. They can also have different data attributes, e.g., cost or urgency. Figure 2 depicts an example of an event log that contains four events with attributes: *Case ID, Task, Resource, Time, Type,* and *Urgency*. We can see, for example, that in an urgent case with identifier '123' Sara completed task 'Investigate Problem' in less than two hours.

Another input to our approach is a process model that models desired process behaviour, i.e., behaviour that does not lead to process risks. In this paper, we use Petri nets with data [7]. We consider deviations from such process models as risky process behaviours. To identify the presence of risky process behaviours in process instances, we use an existing technique for replaying an event log on a process model [6] that allows us to identify discrepancies

Case ID	Task	Resource	Time	Type	Urgency
123	Lodge Incident	John	10/11/2011 10:00	complete	High
123	Investigate Problem	Sara	10/11/2011 10:15	start	High
123	Investigate Problem	Sara	10/11/2011 11:43	complete	High
124	Lodge Incident	Mike	10/11/2011 11:45	complete	Normal

Figure 2: Example of an event log

between an event log and a process model. Then we evaluate overall process risk based on the identified risky process behaviours and produce a chart that depicts values of the overall process risk at different points in time, e.g., as shown below in Figure 15. In our method for predicting aggregate process outcomes, we extract overall process risk time series as described above, extract aggregate process outcome time series, learn a regression model from past process executions and predict aggregate process outcome by evaluating the current value of overall process risk. To evaluate the prediction method, we learn the regression model from a training data set, apply the model to a test data set and produce a chart depicting real and predicted aggregate process outcomes in the test data set, e.g., as shown in Figure 16.

To facilitate the description of our proposed methods in this section, we first provide an illustrative example (Section 3.1), we then define the research problem (Section 3.2), this is followed by the description of our method for evaluating overall process risk (Section 3.3), and we conclude with the description of our method for predicting an aggregate process outcome based on the current value of overall process risk (Section 3.4).

3.1. Illustrative example

Consider processes in a processing center that provides transaction processing services to banks. Typical operations in such a processing center include: issuing of new credit cards, installation of new ATM[2] and POS[3] terminals, processing card transactions, clearing card transactions with Payment systems, investigating and resolving incidents, monitoring and preventing card transaction fraud, etc. The processes rely on human resources and they are exposed to risks typical for a service desk environment: employees make mistakes leading to cost and quality issues, and cases run over time or are re-opened. The same human resources can be involved in different processes. The number of resources working in the processing center is constant, while the amount and type of work they perform can vary significantly. We will use two processes as examples: process *Add new terminal* (Figure 3a) and process *Investigate incident* (Figure 3b). Figures 3a and 3b depict desirable control flows for both processes (using BPMN notation [36]). In real life deviations from these processes are possible and we will show examples of such deviations.

The first process (Figure 3a) specifies the steps followed by engineers in the processing center to add information about new terminals (ATM or POS) on the host. The process starts when a client bank lodges a request for adding a new terminal on the host (task *Lodge Request*). Then an engineer checks information provided by the bank (task

[2] Automated Teller Machine

[3] Point of Sale

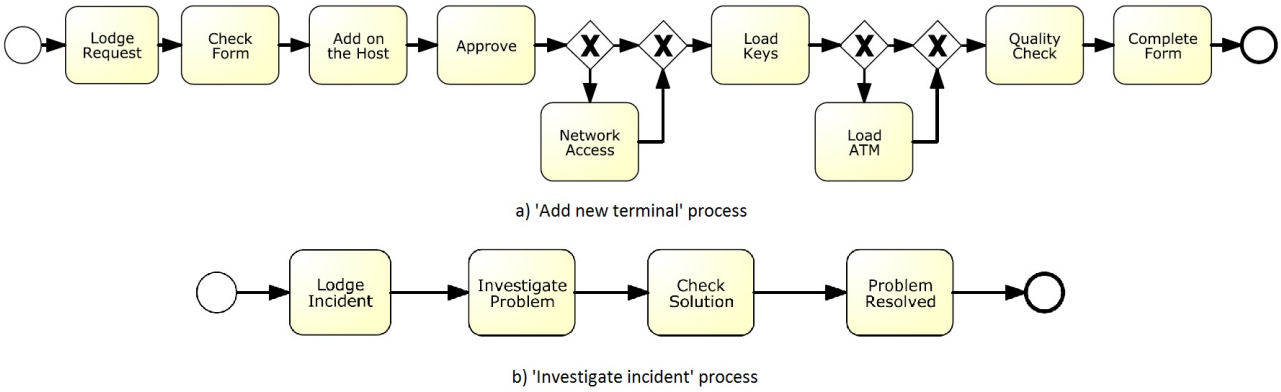


Figure 3: Desirable flows for processes *Add new terminal* (a) and *Investigate incident* (b)

Check Form) and adds information about the terminal on the host (task *Add on the Host*). Then information about the terminal on the host is double-checked by another engineer (task *Approve*). For ATM terminals network access should be allowed (task *Network Access*). Then security keys are loaded on the host to enable safe communication with the terminal (task *Load Keys*). ATM terminals should be started after this (task *Load ATM*). For POS terminals tasks *Network Access* and *Load ATM* are not performed. Then an engineer checks the quality of the terminal installation (task *Quality Check*) and fills in a form with the host specifications of the terminal which is sent to the bank (task *Complete Form*).

The second process (Figure 3b) provides a high-level specification of the steps followed by engineers to resolve an incident. The process starts when a client bank lodges an incident (task *Lodge Incident*), an engineer investigates the incident and provides a solution (task *Investigate Problem*), then the solution is checked (task *Check Solution*) and the incident is closed (task *Problem Resolved*).

Figures 3a and 3b depict desirable control flows for the processes. Deviations from these processes are possible in practice and they can be risky. For example, in process *Add new terminal* task *Approve* can be skipped, or it can be completed by the same engineer that completed task *Add on the Host*. Let us consider two scenarios.

3.1.1. Scenario 1

In our first scenario we look at examples of situations that may happen in process *Add new terminal* and can cause financial losses, i.e., we describe risky process behaviours that increase the risk of cost overrun (depicted in Figure 4).

- Let us consider situations when task *Approve* is skipped or tasks *Add on the Host* and *Approve* are executed by the same engineer. As a result, the likelihood of mistakes in a terminal's specification on the host increases. For example, a wrong currency may be assigned to an ATM cassette or a terminal may be assigned to a wrong terminal profile on the host. Consequently, incorrect amounts of money will be withdrawn which may cause financial loss for a bank or for a client.

- Another example of a risky situation in the process is when an urgent case is not started within three hours after

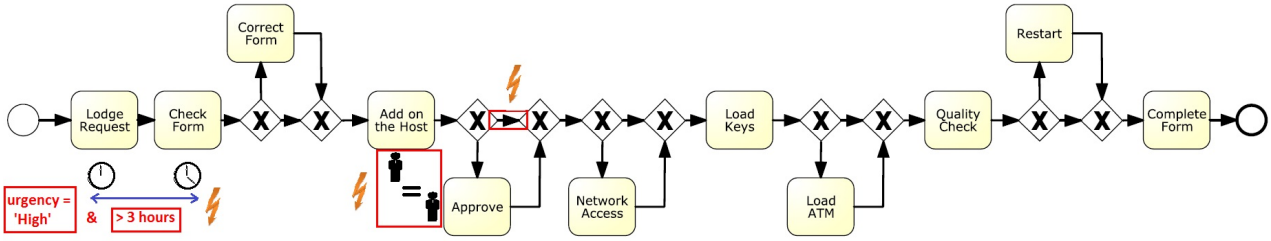


Figure 4: Scenario 1: risky process behaviours in process *Add new terminal*

it was lodged by a bank as this breaks the existing Service Level Agreement (SLA) between the bank and the processing center, which may result in a monetary penalty for the processing center.

Such undesirable situations in processes are often known by domain or process experts and some companies may even keep risk registers [22, 38]. (Note that the process model depicted in Figure 4 allows for execution of tasks *Correct Form* and *Restart* which are not a part of desirable process flow depicted in Figure 3a, but in this scenario we do not consider the execution of these tasks risky.) Let us now assume that a manager would like to know the overall process risk with respect to the risky process behaviors that can cause financial losses and see how the risk evolves over time. We tackle the problem of evaluation of overall process risk in Section 3.3 and demonstrate how to evaluate overall process risk for this scenario in Section 4.1.

3.1.2. Scenario 2

In the second scenario we look at process behaviours that can affect outcomes of process *Add new terminal* in terms of time or quality (depicted in Figure 5). We assume that the same resources can be involved in processes *Add new terminal* and *Investigate incident* at the same time. Moreover, one resource can be involved in multiple instances of the two processes at a given moment of time. Hence, when a resource is very busy working on a case other instances of the two processes in which the resource is involved may get less attention resulting in delays or mistakes. We assume that process analysts or domain experts can provide us with process behaviours that contribute to more risky periods. Let us consider examples of behaviours in process *Add new terminal* that contribute to such risky periods.

- When a higher number of urgent cases arrive within a short period of time employees will focus on the urgent cases, as a result other types of cases may be neglected.
- In process *Add new terminal* two types of cases are possible – adding a new ATM terminal or a new POS terminal. Adding ATM terminals is more time-consuming as the process has more steps, hence, a higher number of requests for adding ATM terminals within a short period of time also contributes to “busy” periods.
- Another example is the execution of task *Correct Form* after task *Check Form*. If these two tasks are performed in this order in a case, this means that the bank provided incomplete or erroneous information about a terminal which is often associated with case delays or problems with the terminal in the future.

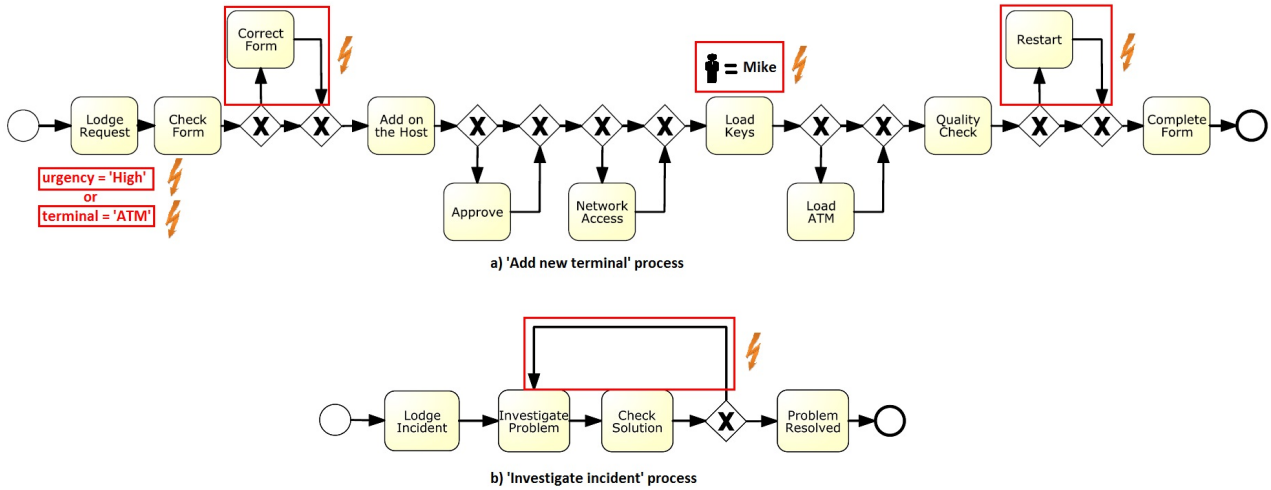


Figure 5: Scenario 2: risky process behaviours in processes *Add new terminal* (a) and *Investigate incident* (b)

- Another risky behaviour is execution of task *Restart* after task *Quality Check* which indicates that there were problems with the terminal's installation.
- It may be known that some resources are often associated with delays or mistakes when executing certain tasks. For example, here we consider the execution of task *Load Keys* by a given resource as a risky behaviour.
- Finally, when tasks *Investigate Problem* and *Check Solution* are repeated in process *Investigate incident* it typically indicates a more complex incident as it was not resolved after the initial investigation. Hence, repetitions of these tasks also contribute to risky periods.

It is important to underline that situations like arrival of an urgent case or repetition of a task in a case are not necessarily risky on their own, but *multiple occurrences of such process behaviours within a short period of time can be risky*. After we have defined risky process behaviours we would like to be able to predict aggregate outcomes of process *Add new terminal* (in terms of the process timeliness and erroneousness) by evaluating the current level of overall process risk. We describe our method for predicting aggregate process outcomes in Section 3.4 and in Section 4.1 we demonstrate how to predict aggregate process outcomes for Scenario 2.

Note that risky process behaviours described in Scenario 1 and Scenario 2 are related to different process perspectives: the control-flow perspective (skipping of task *Approve*, repetition of tasks *Investigate Problem* and *Check Solution*, execution of tasks *Restart* or *Correct Form*); the resource perspective (execution of tasks *Add on the Host* and *Approve* by the same resource, execution of task *Load Keys* by a specific resource); the time perspective (a case is not started within three hours after it was lodged); and the data perspective (a higher number of urgent cases arrive or an ATM has to be added on the host).

3.2. Research problem

In this section we describe the research problem without reference to the particular technical solution (which will be presented in later sections). We start by introducing notations for related concepts.

Risky process behavior, *rpb*, is a process behaviour displayed in a case that may pose a risk on its own or in combination with other risky process behaviours in the same case or in other cases, and which can threaten the achievement of goals by the case, by other cases of the same process or by other processes in a company. An example of a risky process behaviour is the execution of a specific task in a case, e.g., execution of task *Restart* in process *Add new terminal* (Section 3.1). This task is only executed when there is a problem with a terminal and it can cause a delay of the case or other cases processed by the same engineer. It also increases the chance of problems in the terminal's operations in the future. Another example concerns the arrival of urgent cases. While arrival of one urgent case will not usually be considered risky, arrival of multiple urgent cases within a short period of time increases the chance of delays for other cases processed by the same resources, as the resources will give priority to the urgent cases. Other examples of risky process behaviours include: repetitions of certain tasks [12], involvement of certain resources who have been involved in case failures before, and delayed execution of tasks.

3.2.1. Basic notations

Let t, t_1, and t_2 be given points in time, $t_1 \leqslant t_2$, and c be a given case. Then we introduce the following notations.

$C_a(t)$ – the set of all cases active[4] at time t, i.e., cases that were started before or on t and completed after t.

$C_a(t_1, t_2)$ – the set of all cases active during the period $[t_1, t_2)$. Analysts may want to look at cases that were started or completed during the period, or were started before t_1 and completed after t_2.

$RPB(c, t)$ – the set of all risky process behaviours that happened in case c before time t.

$RPB(C, t) \triangleq \bigcup_{c \in C} RPB(c, t)$ – the set of all risky process behaviours that happened in all cases in C before time t.

$CR(c, t)$, or *case risk* at time t, is a function of all risky process behaviours displayed in case c before time t, i.e., $CR(c, t) = g(RPB(c, t))$, $CR(c, t) \in [0, 1]$ (g is a place-holder function; the function definition will be provided in a later section, as here we define the research problem without reference to the particular technical solution).

$OPR(t)$, or *overall process risk* at time t, is a function of all risky process behaviours displayed before time t in the process instances that were active at time t, i.e., $OPR(t) = f(RPB(C_a(t), t))$, $OPR(t) \in \mathbb{R}_{\geqslant 0}$ (f is a place-holder function which will be defined in a later section).

$OPR(t_1, t_2)$, *overall process risk* during the period $[t_1, t_2)$ is a function of all risky process behaviours displayed in the process instances that were active during the period $[t_1, t_2)$, i.e., $OPR(t_1, t_2) = f(RPB(C_a(t_1, t_2)))$, $OPR(t_1, t_2) \in \mathbb{R}_{\geqslant 0}$.

Figure 6 depicts process instances, each consisting of a sequence of task instances A, B, etc., started at different points in time and risky process behaviours (denoted by triangles). Those risky process behaviours that contribute to the overall process risk at time t are highlighted with circles.

[4] We consider the time of the first event in a case as the case start time and the time of the last event in a case as the case completion time.

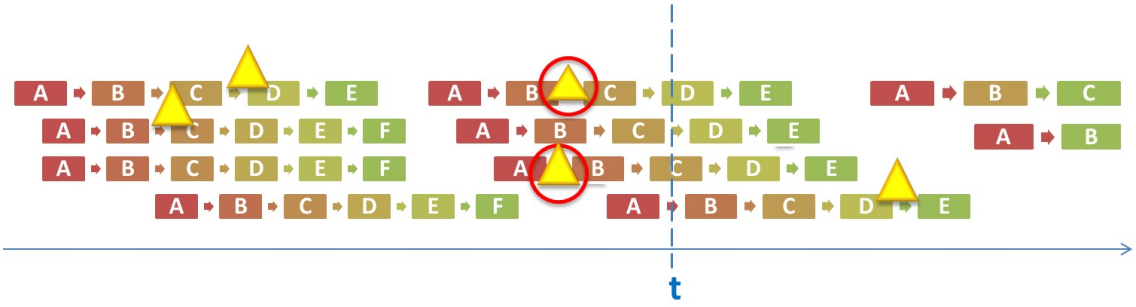


Figure 6: Risky process behaviours contributing to overall process risk at time t

Our first goal can now be re-expressed as: *How to evaluate overall process risk at a given point in time, $OPR(t)$, or during a given period, $OPR(t_1, t_2)$?*

3.2.2. Notations for basic outcomes

Let $Outcome(c)$ denote an outcome of case c, e.g., case duration, cost or the quality of output.

$APO(t)$, or *aggregate process outcome* at a given point in time t, is a function of outcomes of all cases that were active at time t.

Let v denote a given case outcome value, e.g., it can be used to specify the value of normal case duration or an acceptable level of output quality.

$APO(t, v)$ denotes *aggregate process outcome* at time t for a given case outcome value v.

Some examples of aggregate process outcomes include: the number of cases that produced low-quality outputs, the fraction of delayed cases with respect to the total number of active cases, average case cost, or the number of cases that received negative customer feedback.

Notation $APO(t, [v])$ is used as a generalisation of $APO(t)$ and $APO(t, v)$, i.e., we use it in contexts that are applicable to both $APO(t)$ and $APO(t, v)$.

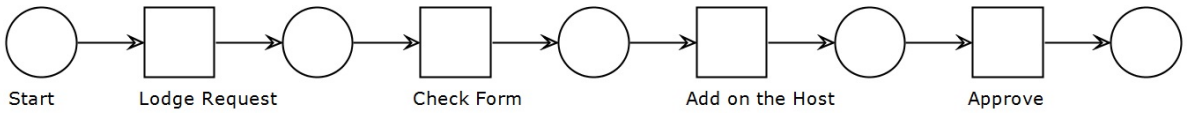
Let $EAPO(t, [v])$ denote the *expected aggregate process outcome* for cases active at time t.

Our second goal can therefore be re-expressed as: *How to predict expected aggregate process outcome at time t, $EAPO(t, [v])$, based on the value of overall process risk at time t, $OPR(t)$?*

3.3. A method for evaluation of overall process risk

An input to our method is information about risky process behaviours. Domain experts often know which factors can cause negative process outcomes. Some companies keep risk registries specifying these risky behaviours. For example, in a hospital emergency department the following process behaviours increase patients' health risks: task *X-ray* is not performed for a patient with chest pain, an urgent patient is not attended to within 15 minutes after admission, a patient record is not completed within an hour after the patient is discharged, etc. Some of these behaviours may be more risky than others, e.g., not performing task *X-ray* is more risky than not completing a patient's record in time. Other examples of risky process behaviours were described in Section 3.1.

Our method uses as an input a process model that represents desirable (i.e., not risky) process behaviour. We consider deviations in cases from the process model as risky process behaviours. We use Petri nets with data [7]. Petri



Transition	Variables written	Guard
Lodge Request	urgency, LodgeRequest_Time	
Check Form	CheckForm_Time	((urgency = "High") ∧ (CheckForm_Time' ≤ (LodgeRequest_Time +3))) V (urgency = "Normal")
Add on the Host	AddOnTheHost_Resource	
Approve	Approve_Resource	Approve_Resource' ≠ AddOnTheHost_Resource

Figure 7: Example of a Petri net with data

nets with data can be translated/created from processes captured in well-known process modelling languages such as BPMN [36] or EPCs [32] if needed.

> A Petri net with data is "a Petri net in which transitions can write variables" [6]. It can be defined as $N = (P, T, F, V, U, W, G)$, where (P, T, F) define a Petri net (P – a set of places, T – a set of transitions, F – the flow relation between transitions and places), V – a set of variables, U – a function that defines the domain of each variable in V, W – a write function "that labels each transition with a set of write operations", G – a guard function that "associates a guard with each transition" [6]. A guard is a formula over the process variables, that can use the logical operators conjunction, disjunction, and negation [6].

Figure 7 depicts an example of a Petri net with data which comprises the first four tasks from the process depicted in Figure 3a (*Lodge Request, Check Form, Add on the Host* and *Approve*). The table depicted in Figure 7 specifies variables written by each transition and transition guards. For example, transition *Approve* writes variable *Approve_Resource*[5] and there is a guard associated with the transition: $Approve\_Resource\prime \neq AddOnTheHost\_Resource$[6]. Modelling desired process behaviour as a Petri net with data allows us to consider different process perspectives, such as the control-flow, time, resource and data perspectives (the resource and time perspectives are modelled as data [6]).

To evaluate overall process risk at a given point in time t our method follows the steps:

1. *Select all cases* $C_a(t)$ *that were active at time* t, i.e., started before t and completed after t.
2. For all $c \in C_a(t)$ *discard all events that happened after time* t, and let $C_{at}(t)$ denote the set of the resulting truncated cases.
3. For all $c \in C_{at}(t)$ *evaluate case risks at time* t, $CR(c, t)$.

   We use an existing algorithm for replaying an event log on a Petri net with data [6]. The algorithm finds optimal alignments of cases in an event log and a process model. Figure 8 depicts an example of alignments for two cases and the process model depicted in Figure 7. In *Case 1* task *Approve* was not executed, hence the case deviates from the process model, in *Case 2* tasks *Add on the Host* and *Approve* were executed by the

[5] We add prefix [Transition name]_ to data variable names to simplify tracking the transitions that write these data variables.

[6] If a prime symbol is used with a variable in a transition guard it refers to the value of the variable after occurrence of the transition [6].

Alignment 1		Alignment 2	
Process model	Case 1	Process model	Case 2
Lodge Request	Lodge Request [urgency = "High", LodgeRequst_Time = 1]	Lodge Request	Lodge Request [urgency = "High", LodgeRequst_Time = 1]
Check Form	Check Form [CheckForm_Time = 2]	Check Form	Check Form [CheckForm_Time = 2]
Add on the Host	Add on the Host [AddOnTheHostResource = Mike]	Add on the Host	Add on the Host [AddOnTheHostResource = Mike]
Approve	-	Approve	Approve [Approve_Resource = Mike]

Figure 8: Example of alignments of two cases with a process model

same resource (Mike) which does not satisfy the guard associated with task *Approve* (Figure 7), hence *Case 2* also deviates from the model. The guard associated with task *Check Form* is not violated in both cases as the time difference between tasks *Check Form* and *Lodge Request* is smaller than 3. The algorithm first finds alignments of cases and a process model considering only the control-flow perspective, and then it applies Integer Linear Programming to find an optimal alignment that also considers other process perspectives, such as data, resource and time. The algorithm uses the costs of deviations as an input and aims to find alignments that minimise the cost. When replaying truncated cases it is important not to penalise improper completion of cases, the algorithm provides for this possibility. When replaying truncated cases we cannot consider skipping of a task as a risky process behaviour as the algorithm cannot distinguish between an incomplete case and a skipped task. This is a limitation of our approach and a direction for future work. The possibility to use other approaches to identify risky process behaviours from event logs could also be explored, e.g., the use of ProM's LTL Checker, an approach based on Linear Temporal Logic, can be considered [34] or checking against a Declare specification [23].

We replay all cases $c \in C_{at}(t)$ using the algorithm which returns an optimal alignment for each such case. Risky process behaviours may have different levels of severity, hence our approach allows users to assign different costs to different risky process behaviours (these costs are not the same as the costs used by the replay algorithm). Recall that we model risky process behaviours as deviations from a process model which represents desirable process behaviour. Such deviations include: execution of a task in a case that is not in the process model, skipping of a task that is in the process model, non-writing of a variable by a task, and execution of a task whose guard is false. If a guard is a conjunction of a few different constraints, our approach allows users to assign different costs to these constraints. Let $Cost(rpb)$ denote the cost of risky process behaviour *rpb*. The case risk of case c at time t is:

$CR(c,t) \triangleq \sum_{rpb \in RPB_{(c,t)}} Cost(rpb)$.

4. *Evaluate overall process risk at time t*, $OPR(t)$.

   We use two measures:

   1) *Mean process risk* at time t: $OPR_M(t) \triangleq \sum_{c \in C_{at}(t)} CR(c,t)/|C_{at}(t)|$

   2) *Total process risk* at time t: $OPR_T(t) \triangleq \sum_{c \in C_{at}(t)} CR(c,t)$

To get an idea of the evolution of overall process risk over time we generate a time series of process risk values at different points in time (e.g., daily or weekly). Time series are often used to model the evolution of different

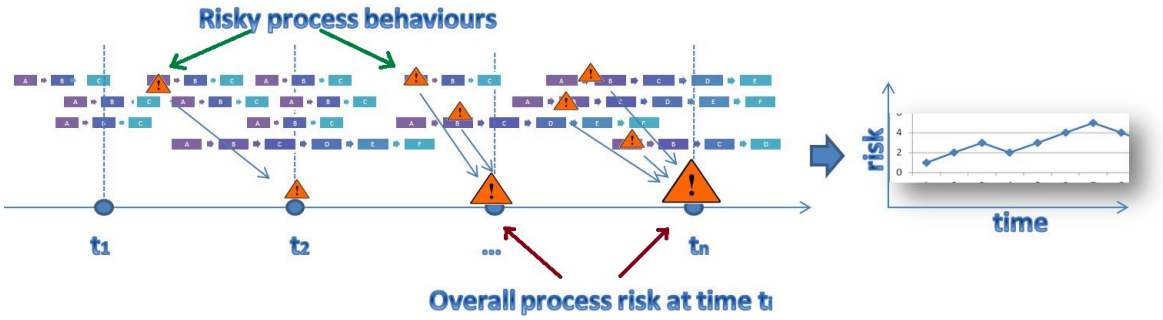


Figure 9: The main idea of our method for evaluating overall process risk

phenomena [16]. As per formal definitions of sequence types in discrete mathematics, we model a time series as a set of ordered pairs, where the first element of each pair determines the position in the series (timestamp in our case) and the second element is the value at that position. The time series sampling rate is an input parameter. Selection of the time series sampling rate is an important step that can affect the analysis results. It is a well-known problem often discussed in the literature [18]. Let TS_{start} be the starting time point, $TS_{slotsize}$ be the sampling rate and TS_{size} be the number of time slots, then an OPR time series can be defined as:

$$TS_{OPR} \triangleq \{(t, OPR(t)) \mid t \in \{TS_{start} + i * TS_{slotsize} \mid i \in \{0, 1, \ldots, TS_{size} - 1\}\}\}$$

Figure 9 depicts the main idea of the method, in which the overall process risk is evaluated at the end of each time slot. We can see, for example, that at time t_2 four cases are active and in one of the cases one risky process behaviour occurred before t_2 (depicted as a triangle on top of a process instance). Let the cost of this risky process behaviour, $Cost(rpb)$, be 1. Then $OPR_T(t_2)$ is 1 and $OPR_M(t_2)$ is 0.25. We use notation $OPR(t)$ when evaluating overall process risk at time t in one process; when considering multiple processes, we denote overall process risk of process P_i at time t as $OPR_{P_i}(t)$.

The procedure for evaluating overall process risk during a given period of time, $OPR(t_1, t_2)$, is similar to the procedure for evaluation of overall process risk at a given point in time $OPR(t)$ described above with the following differences. During the first step we select cases that were active during time slot (t_1, t_2). Analysts may choose to look at cases that were completed during time slot (t_1, t_2), started during time slot (t_1, t_2), or were started before t_1 and completed after t_2. For the cases that were not completed by time t_2 analysts may choose to use complete cases or cases that are truncated at time t_2. This allows for flexibility when selecting which cases and events should be considered. The possibility to evaluate overall process risk considering cases active during a given time slot is introduced for users who are only interested in tracking the changes of overall process risk over time, while we evaluate overall process risk at a given point in time if we want to use it for prediction of aggregate process outcomes.

3.4. A method for predicting an aggregate process outcome by evaluating overall process risk

In this section we describe our method for predicting aggregate process outcomes. As a part of the method we generate overall process risk time series as described in Section 3.3. We then extract aggregate process outcome time

series and learn a prediction model as described below.

Recall that an aggregate process outcome at an earlier time t, $APO(t, [v])$, is a function of outcomes of the cases that started before t and were completed after t, and v denotes a given case outcome value. Examples of case outcomes include: case duration, cost, quality of case output, customer feedback, etc. Our method allows one to look at the following types of aggregate process outcomes: 1) the number of cases with a given case outcome ("*total*"), 2) the fraction of cases with a given case outcome ("*fraction*"), and 3) the mean value of a given case outcome ("*mean*"). The value of a case outcome should be stored in a case attribute or in an attribute of a given task (a user input).

1. Aggregate process outcome (*total*):
   $APO(t, v) \triangleq |\{c \in C_a(t) \mid Outcome(c) = v\}|$, i.e., the number of cases that were active at time t and completed with a given case outcome, e.g., the number of cases completed with quality level "Low".
2. Aggregate process outcome (*fraction*):
   $APO(t, v) \triangleq |\{c \in C_a(t) \mid Outcome(c) = v\}|/|C_a(t)|$, i.e., the fraction of cases that were active at time t and completed with a given case outcome, e.g., the fraction of cases completed with quality level "High".
3. Aggregate process outcome (*mean*):
   $APO(t) \triangleq \sum_{c \in C_a(t)} Outcome(c)/|C_a(t)|$, i.e., the mean value of a given case outcome for the cases that were active at time t, e.g., mean case duration.

Our method for predicting aggregate process outcomes by evaluating overall process risk uses the following steps:

1. *Generate aggregate process outcome time series.*
   For a given time series sampling rate we generate a time series that consists of the values of aggregate process outcomes at each point in time. Let TS_{start} be the starting time point, $TS_{slotsize}$ be the sampling rate, TS_{size} be the number of time slots, then an APO time series can be defined as:
   $TS_{APO} \triangleq \{(t, APO(t, [v])) \mid t \in \{TS_{start} + i * TS_{slotsize} \mid i \in \{0, 1, \ldots, TS_{size} - 1\}\}\}$
2. *Generate overall process risk time series.* For each process $P_1 \ldots P_n$ whose behaviours can affect the outcomes of a given process we generate the overall process risk time series as described in Section 3.3 using the same time series parameters as in Step 1 of this method. Let $TS_{OPR_{P_i}}$ denote the overall process risk time series for process P_i.
3. *Learn a prediction model.* For this purpose we use a method for non-parametric regression proposed by Racine and Li [26]. It is a kernel-based method that works with continuous and categorical data and "do not impose any functional form assumptions" [26]. The method is suitable for our purpose as it allows us to use multiple independent variables and it does not make assumptions about data distribution. Other multi-variate non-parametric methods could have been used. The values of the overall process risk time series $TS_{OPR_{P_i}}$ are used as the values of independent variables, while the values of the aggregate process outcome time series TS_{APO} are used as the values of the dependent variable.

4. *Predict expected aggregate process outcome at time t.* During the last step we use the values of the overall process risk at time t for each process P_i, $OPR_{P_i}(t)$, and the regression model fitted during Step 3 to predict the expected aggregate process outcome at time t, $EAPO(t, [v])$.

Figure 10 depicts the idea of our method for an example when the outcomes of process P_1 are affected by risky process behaviours in processes P_1 and P_2. For example, let $APO(t, v)$ be the number of cases that were active at time t and completed later with quality level "Low" (v = "Low"; such 'failed' cases are marked by shields with crosses in Figure 10). We can see that at time t_2 four cases were active in process P_1 and one of the cases completed later with a "Low" quality, hence $APO(t_2$,"Low") is 1. The expected aggregate process outcome for process P_1 at time t_n is predicted based on the values of the overall process risk for processes P_1 and P_2 at time t_n using the regression model that was learned from past values of the overall process risk for processes P_1 and P_2 (independent variables) and the past values of the aggregate process outcome for process P_1 (the dependent variable). A manager may wish to be alerted when a predicted aggregate process outcome exceeds some threshold, and he may take some mitigation actions, e.g., prioritise tasks assigned to employees or outsource some process operations.

4. Validation

We implemented our approach as a plugin "Process Risk Evaluation"[7] of the process mining framework ProM 6[8]. The plugin takes event logs in the XES format[9] and process models in data-aware PNML format[10] as input. It allows us to evaluate overall process risk at different points in time, produce a chart illustrating the overall process risk time series, and automatically detect change points [28], i.e., points in time when the overall process risk changed. The plugin also allows us to extract overall process risk time series and aggregate process outcome time series from event logs (for both training and test data sets), to learn a prediction model using time series extracted from a training data set and to produce and visualise predictions for a test data set. The plug-in calculates *R-squared* [11], as this measure is commonly used in statistics to evaluate the goodness of predictions. For change point detection and regression analysis we use R[11]. Our plug-in accesses R's functionality using the JRI Java/R Interface[12]. To detect change points we use the CPM framework that allows us to detect changes in location, in scale or arbitrary distributional changes and which is implemented as R package *cpm*[13]. For regression analysis we use R package *np*[14]. To evaluate logical expressions we use the MVEL library[15], and to visualise the results of the analysis we use the JFreeChart library[16].

[7] http://yawlfoundation.org/risk/files/ProcessRiskEvaluation.7z
[8] www.promtools.org/prom6/
[9] http://www.xes-standard.org/
[10] http://www.pnml.org/
[11] http://www.r-project.org/
[12] http://rforge.net/JRI/
[13] http://cran.r-project.org/web/packages/cpm/vignettes/cpm.pdf
[14] http://cran.r-project.org/web/packages/np/index.html
[15] http://mvel.codehaus.org/
[16] http://www.jfree.org/jfreechart/

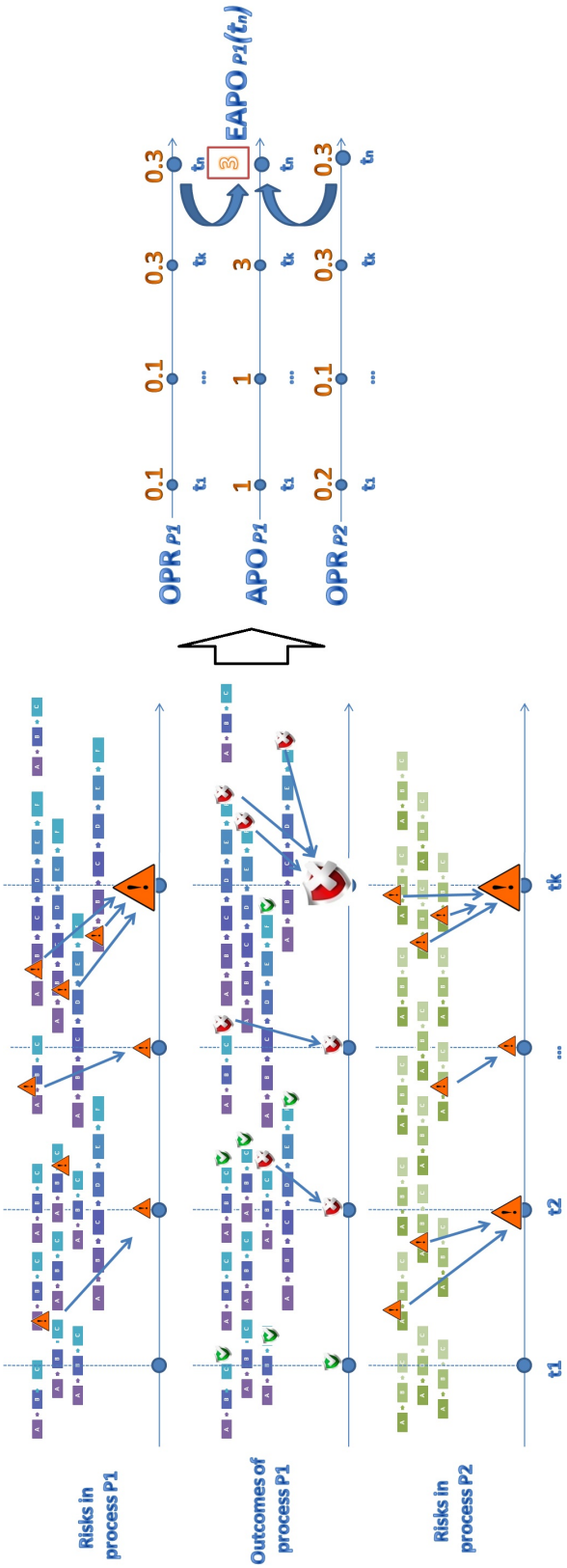


Figure 10: Predicting aggregate process outcome by evaluating overall process risk

We first validate our method for evaluation of the overall process risk using a synthetic event log created for the process from Scenario 1 (Section 3.1). In the synthetic event log we introduce all risky process behaviours from Scenario 1. We vary the probability of these risky process behaviours in different parts of the event log. We then apply our method for evaluation of the overall process risk to the event log and check if the resulting charts reflect the changes we introduced in the log. We then validate our method for prediction of aggregate process outcomes using synthetic event logs created for the two processes from Scenario 2 (Section 3.1). We introduce all risky process behaviours from Scenario 2 varying their probabilities in different parts of the event logs. The probability of negative process outcomes depends on the value of overall process risk at a given time. We created three variants of the event logs: the probability of negative process outcomes during risky periods is high in Variant 1, it is lower in Variant 2 and it does not depend on the overall process risk in Variant 3. For each variant we generated two sets of event logs used as a training and a test data set. We then applied our method for predicting aggregate process outcomes to the event logs. The regression model was learned using a training data set and it was applied to a test data set to generate predictions. To evaluate the goodness of the predictions we check the values of *R-squared* and charts depicting real and predicted values. Below we describe in details the simulation procedure used to create the synthetic event logs, the setup of the experiments and discuss the results.

4.1. Evaluation of the approach using synthetic event logs

We created synthetic event logs for both scenarios described in Section 3.1[17] using CPN Tools[18]. In CPN Tools we can model processes using Coloured Petri Nets which allow us to specify data types for places and attach data values to tokens [15].

4.1.1. Scenario 1: Evaluating overall process risk

We generated an event log for the process depicted in Figure 4. We introduced three risky process behaviours from Scenario 1 described in Section 3.1: task *Approve* is skipped in some cases, task *Check Form* is completed later than three hours after the completion of task *Lodge Request* in some urgent cases, and tasks *Add on the Host* and *Approve* in some cases are performed by the same resource. For the purposes of this evaluation we assume that the occurrence of a risky process behaviour follows the Bernoulli distribution with a given probability. The probability of these risky process behaviours differs in different parts of the event log. It is 10% from time t_0 to t_1 (time points t_0–t_3 are depicted in Figures 12 and 13), 20% from time t_1 to t_2, 50% from time t_2 to t_3, and 80% after time t_3. Time between case arrivals follows an exponential distribution[19] with a mean value of three hours. Only events with transaction type *complete* were recorded in the event log; we filtered out incomplete cases. The resulting event log contains 993 cases with an average case duration of 35 hours.

[17] The synthetic event logs and the CPN models can be downloaded from http://yawlfoundation.org/risk/files/SyntheticData.7z

[18] http://cpntools.org/

[19] Exponential distributions describe the time between events in a Poisson process; Poisson distribution is often used to model arrival rates in queueing theory.

We used the process model depicted in Figure 11a to replay the event log. The process model does not allow skipping task *Approve*, hence in all cases in which the task was skipped the risky process behaviour is detected. In order to identify the other two risky process behaviours we added the following annotations to the process model. Transition (i.e., task) *Add on the Host* writes *String* variable *AddOnTheHost_Resource*, transition *Approve* writes *String* variable *Approve_Resource*, transition *Lodge Request* writes *String* variable *urgency* and *Float* variable *LodgeRequest_Time*, and transition *Check Form* writes *Float* variable *CheckForm_Time*.

The following guard was added to task *Approve*: $Approve\_Resource' \neq AddOnTheHost\_Resource$

Task *Check Form* was annotated with the guard:

$((urgency = \text{“}High\text{”}) \wedge (CheckForm\_Time' \leqslant (LodgeRequest\_Time + 3))) \vee (urgency = \text{“}Normal\text{”})$

The plugin pre-processes a log and adds an attribute ($[TaskName]\_Time$) to events that specifies the elapsed time since the beginning of the case in a given time unit, e.g., in hours or days. We used an hour as the value of the time unit in the experiments reported in this section. One day was used as the time series sampling rate and we evaluated daily values of the overall process risk with respect to cases completed during the day. The total duration was 130 days. When replaying event logs on process models one needs to specify the costs of deviations used by the replay algorithm to create optimal alignments. We used cost 1 for all deviations. We also need to specify the second type of costs that are used to evaluate case risks, different costs can be used for different types of risky process behaviours. We used cost 1 for all types of risky process behaviours as we do not consider that some behaviours are more risky than others in this scenario.

The plugin generated an overall process risk time series with respect to the three risky process behaviours. Consider an example when five cases are completed during a time slot. In one urgent case task *Approve* was not executed and task *Check Form* was completed six hours after the completion of task *Lodge Request*, in two cases the same resource executed tasks *Add on the Host* and *Approve*, and two other cases completed without any risky process behaviours. The cost of each risky process behaviour is 1, hence the total value of overall process risk for this time slot is 4, while the mean value of overall process risk is 0.80.

Figure 12 depicts the mean values ($OPR_M(t)$) of the overall process risk and Figure 13 depicts the total values ($OPR_T(t)$) of the overall process risk. Red vertical lines mark the points in time when the changes in the risky process behaviours were introduced and diamond shapes mark change points automatically detected by the plug-in. We can observe that both charts reflect the changes in the probabilities of the three risky process behaviours we introduced in the log. The plug-in precisely detected the change point at time t_2, and the other two changes at times t_1 and t_3 were pinpointed in the log a few days earlier or later.[20] We demonstrated in this section that our first method presented in this article can successfully evaluate overall process risk. The changes in probabilities of risky process behaviours that were introduced in the synthetic event log can be clearly observed on the charts depicting overall process risk time

[20]Note, that the change point detection method uses complete time series to estimate the change points. As the method is not 100 % precise, the change points can be detected earlier or later than the actual change was introduced. The precision also depends on the test used and the sensitivity threshold (input parameters). We looked at changes in location, but other tests can be used, e.g., a test that detects changes in scale.

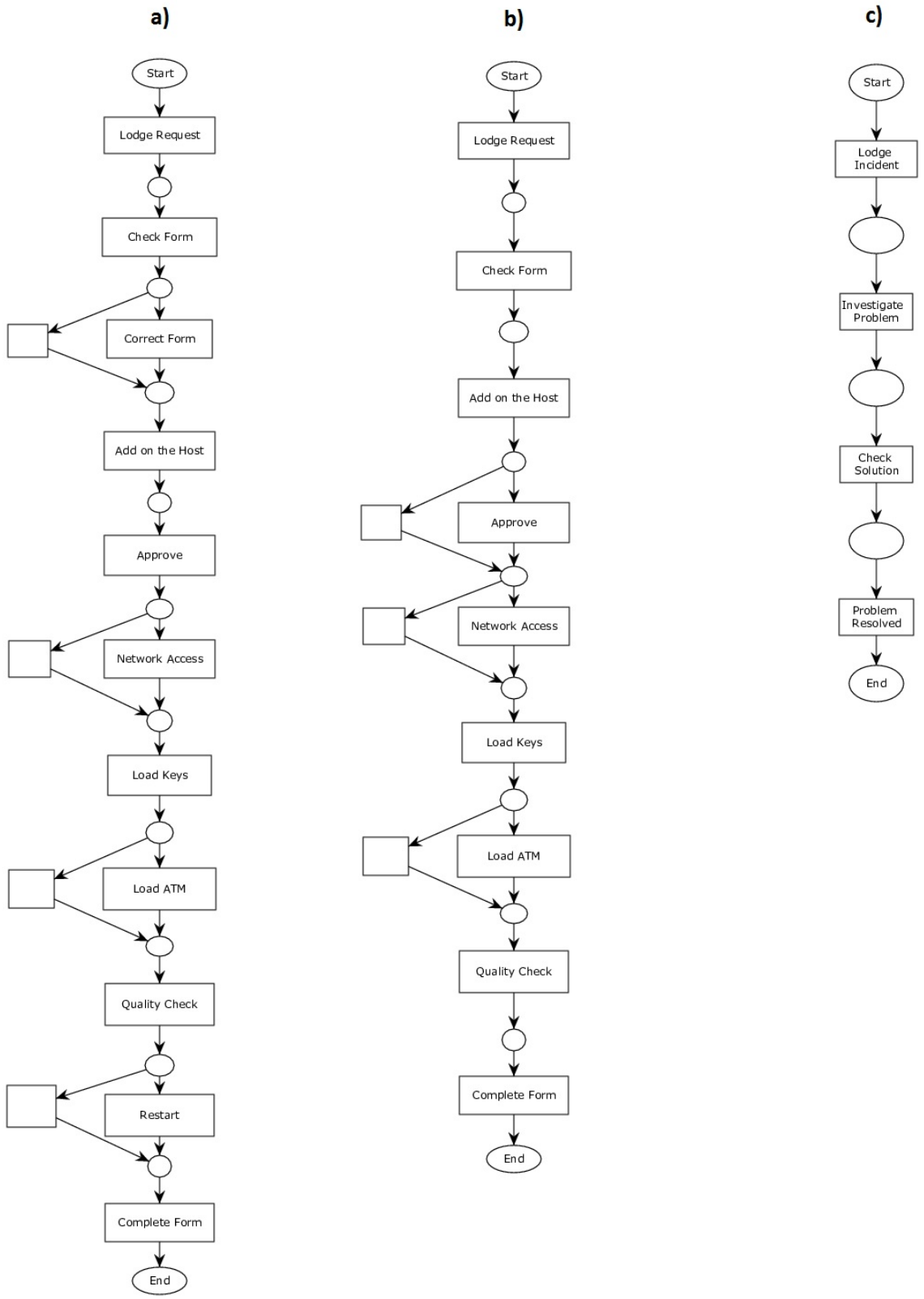


Figure 11: Petri nets used to replay event logs: a) Scenario 1, process *Add new terminal*; b) Scenario 2, process *Add new terminal*; c) Scenario 2, process *Investigate incident*.

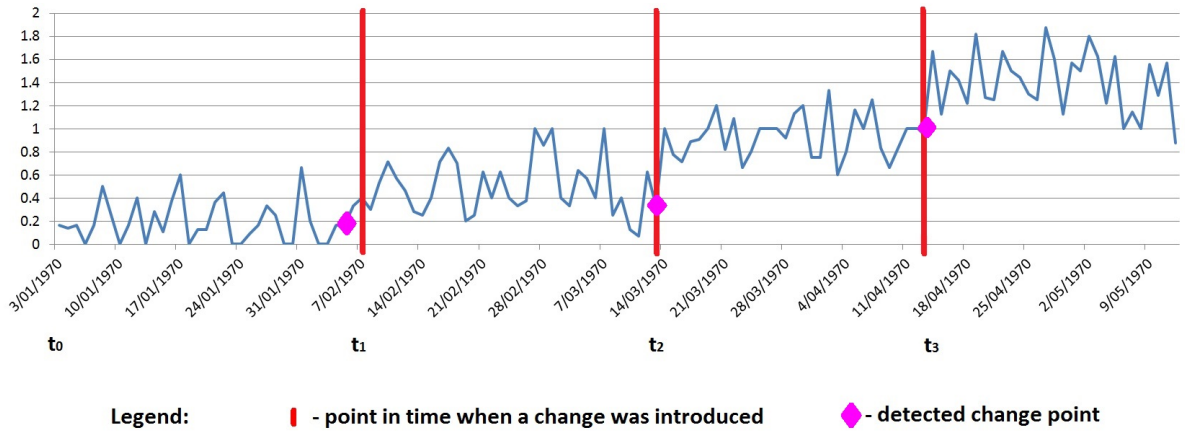


Figure 12: Evolution of overall process risk from Scenario 1 - mean risk ($OPR_M(t)$ on the vertical axis)

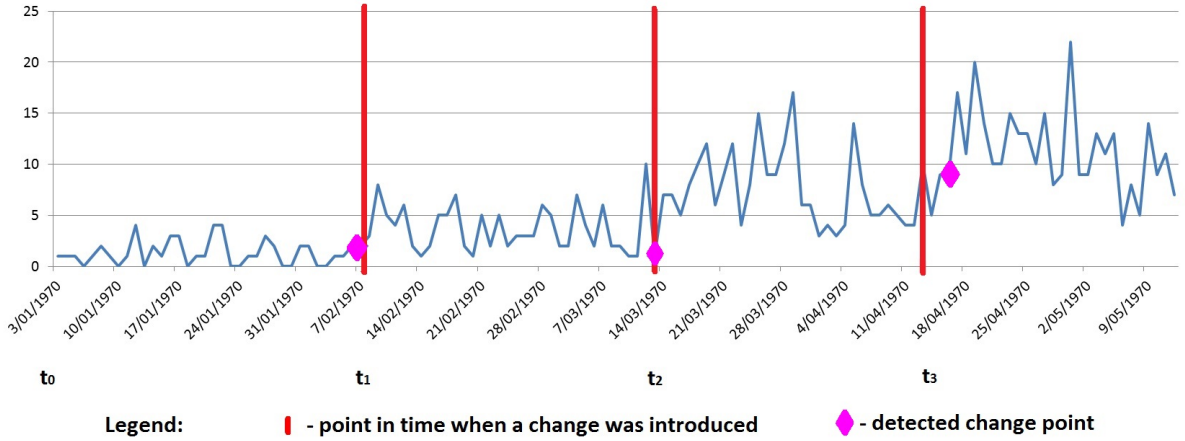


Figure 13: Evolution of overall process risk from Scenario 1 - total risk ($OPR_T(t)$ on the vertical axis)

series produced by the plug-in. Furthermore, the plug-in can annotate such charts with change points to facilitate the analysis for a user.

4.1.2. Scenario 2: Predicting aggregate process outcomes using overall process risk

In this experiment we generated a set of event logs representing the processes depicted in Figures 5a and 5b using CPN Tools and following the procedure described below. To simulate the two processes we used the Petri net depicted in Figure 14[21]. The net represents the two processes (process *Investigate incident* on the left and process *Add new terminal* on the right) that share one dedicated place labeled as *Total Risk*. Place *Total Risk* contains one token (of type *Integer*) that stores the current value of overall process risk. Whenever a risky process behaviour happens in process *Add new terminal* (i.e., an urgent case arrives, an ATM terminal needs to be added on the host, task *Correct Form* is executed, task *Restart* is executed or task *Load Keys* is performed by a "risky" resource) the value of the token in place

[21]The model depicted in Figure 14 is an abstraction (a number of data annotations have been removed), the complete specification of the model used in the simulation can be downloaded from http://yawlfoundation.org/risk/files/SyntheticData.7z.

Total Risk is increased by 1. When tasks *Investigate Problem* and *Check Solution* are repeated in process *Investigate incident* the value of the token in place *Total Risk* is increased by 2. We also created two dedicated places for each process, *Case Risk* and *Incident Case Risk*, that keep track of the risk values contributed by each case to place *Total Risk*. When an instance of any of the processes is completed, we decrease the value of the token in place *Total Risk* by the risk value added by the process instance. Hence, at any given point in time the value of the token in place *Total Risk* represents the current value of the overall risk with respect to all running instances of the two processes. Instances of process *Add new terminal* can be completed without any issues or delays (in which case the value of attribute *Outcome* of task *Complete Form* is set to 0), or they can be completed with a negative outcome, e.g., the terminal is not working or it was added on the host with a delay (in which case the value of attribute *Outcome* is set to 1). The probability of a negative case outcome depends on the value of the token in place *Total Risk* at the moment of the case's completion.

The time between case arrivals in both processes follows an exponential distribution with a mean value of three hours. We set the probability of occurrence of the risky process behaviours as follows:

- It was set to 10% during the first 10000 steps (i.e., transition firings) of the simulation process (during this period the value of the token in place *Total Risk* was below 15);
- It was set to 40% during the next 5000 steps of the simulation (the value of the token in place *Total Risk* varied from 15 to 45 during most of this time); and
- It was set to 60% during the last 5000 steps of the simulation (the value of the token in place *Total Risk* exceeded 45 during most of this time).

We refer to these three different risk levels as *Normal, Medium* and *High*.

We generated three variants of event logs with different probabilities of negative case outcomes for each risk level as described in Table 1. In the first variant during periods with *Normal* risk level (i.e., when the value of the token in place *Total Risk* is less than 15) the probability of a negative case outcome is 5%, while during periods with a *Medium* or *High* risk level it is 40% and 60% correspondingly. In the second variant the probability of negative case outcomes during the period with *Normal* risk level is also 5%, while during periods with *Medium* and *High* risk levels it is 20% and 30% respectively. In the third variant the probability of negative case outcomes does not depend on the risk level and it is always 25%. For each variant we generated two event logs – one used to learn a regression model (a training data set) and the other one used for getting the predictions (a test data set). As a result of the simulation we created 12 event logs (six for process *Add new terminal* and six for process *Investigate incident*). Table 2 specifies characteristics of the event logs.

To evaluate the overall process risk in each process we used the process models depicted in Figures 11b and 11c. Skipping of task *Approve* is not considered as a risky process behaviour in Scenario 2, hence the model in Figure 11b allows to skip the task (a silent transition). In the model of process *Add new terminal* transition (i.e., task) *Lodge*

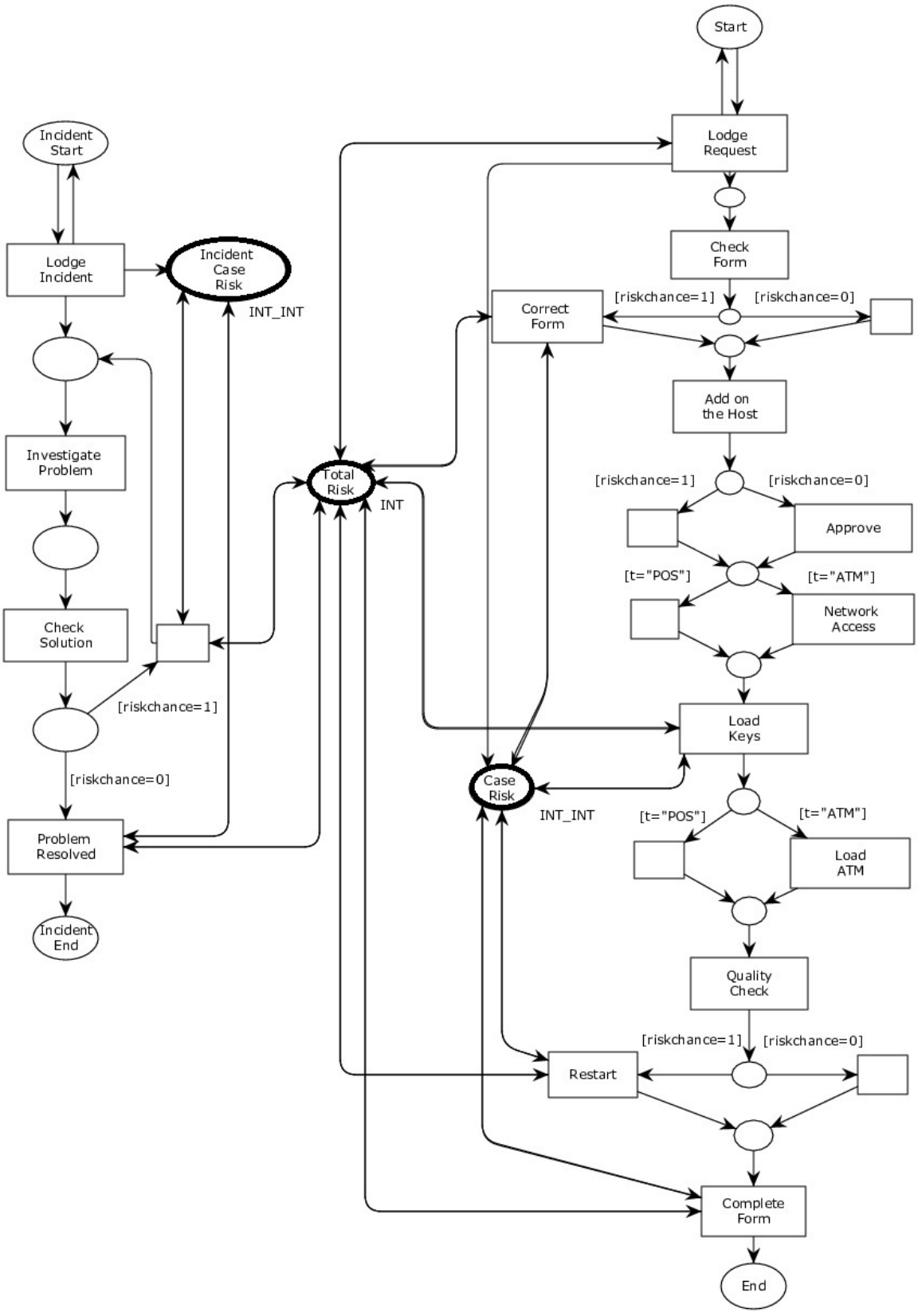


Figure 14: Simulation model for the two processes from Scenario 2

Table 1: Probability of negative case outcomes for different risk levels

	Normal risk	Medium risk	High risk
Variant 1	5%	40%	60%
Variant 2	5%	20%	30%
Variant 3	25%	25%	25%

Table 2: Characteristics of simulated event logs for Scenario 2

Event log	Number of cases	Mean case duration
Add new terminal, Variant 1, training set	1171	49.7 hours
Add new terminal, Variant 1, test set	1172	49.7 hours
Investigate incident, Variant 1, training set	1095	22.6 hours
Investigate incident, Variant 1, test set	1124	22.5 hours
Add new terminal, Variant 2, training set	1162	45.9 hours
Add new terminal, Variant 2, test set	1136	46.3 hours
Investigate incident, Variant 2, training set	1154	22.2 hours
Investigate incident, Variant 2, test set	1203	22.2 hours
Add new terminal, Variant 3, training set	1164	48.8 hours
Add new terminal, Variant 3, test set	1168	48.3 hours
Investigate incident, Variant 3, training set	1106	22.9 hours
Investigate incident, Variant 3, test set	1127	22.4 hours

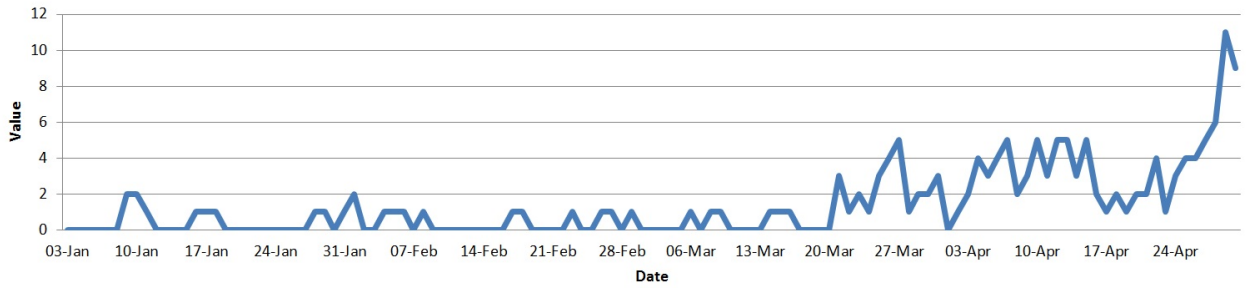


Figure 15: Example of total process risk ($OPR_T(t)$ on the vertical axis) in process *Investigate incident*, variant 1, training set

Request writes *String* variables *urgency* and *terminal*, and transition *Load Keys* writes *String* variable *resource*. To transition *Lodge Request* we added the guard: $urgency' \neq$ "*Urgent*" $\wedge$ $terminal' \neq$ "*ATM*"; to transition *Load Keys* we added the guard: $resource' \neq$ "*Risky*"[22]. When an urgent case arrives or an ATM has to be added on the host the guard of task *Lodge Request* will evaluate to false. When task *Load Keys* is executed by resource *Risky* the task's guard evaluates to false. The model of process *Add new terminal* (Figure 11b) does not specify tasks *Correct Form* and *Restart*, hence executions of these tasks are deviations from the model and are treated as risky process behaviours. The model of process *Investigate incident* (Figure 11c) does not allow repetitions of tasks *Investigate Problem* and *Check Solution*, hence their repetitions in cases will be considered as deviations.

When replaying event logs on process models we need to specify the costs of deviations used by the replay algorithm to create optimal alignments and the costs that are used to evaluate case risks. We used cost 1 for all deviations and for all types of risky process behaviours as we do not consider that some behaviours are more risky than others in this particular scenario. The guard of task *Lodge Request* is a conjunction of two constraints: we check if a terminal is not *ATM* and if urgency is not *High*. We assign cost 1 to each constraint, hence if both constraints are not satisfied the value of that case's risk is increased by 2, if only one constraint is not satisfied, the value of the case's risk is increased by 1.

We looked at the daily values of the overall process risk (total risk measure) during 120 days. The overall process risk at each point in time was evaluated considering only those parts of cases that were completed before this point in time. Hence, when replaying the event logs we did not penalise improper case completions. We used the number of cases completed with a negative outcome (i.e., those cases in which the value of attribute *Outcome* is set to 1) as the measure of aggregate process outcome and we considered only completed cases when creating the aggregate process outcome time series. We used training data sets to learn regression models for each variant of event log and we then used test event logs to generate predictions. The plugin generated charts depicting overall process risk time series and a chart that depicts real and predicted values of aggregate process outcomes. As an example, Figure 15 depicts the evolution of total process risk for process *Investigate incident* in the training set of Variant 1.

Figures 16, 17 and 18 depict real and predicted aggregate process outcomes (i.e., the number of cases completed with a negative case outcome) for Variants 1, 2 and 3 respectively. We can observe that predicted values of aggregate

[22]We used name *Risky* to label the resource who often makes mistakes or causes delays when executing task *Load Keys*.

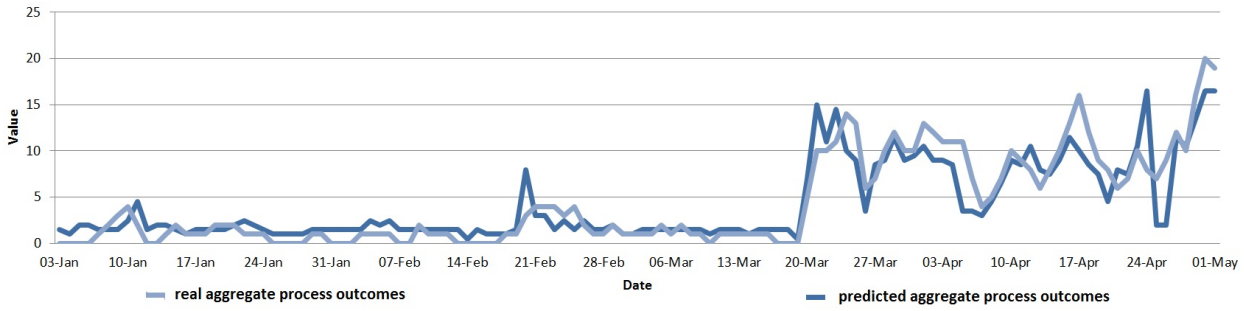


Figure 16: Predicted and real aggregate process outcome (the number of cases completed with a negative outcome, on the vertical axis), Variant 1

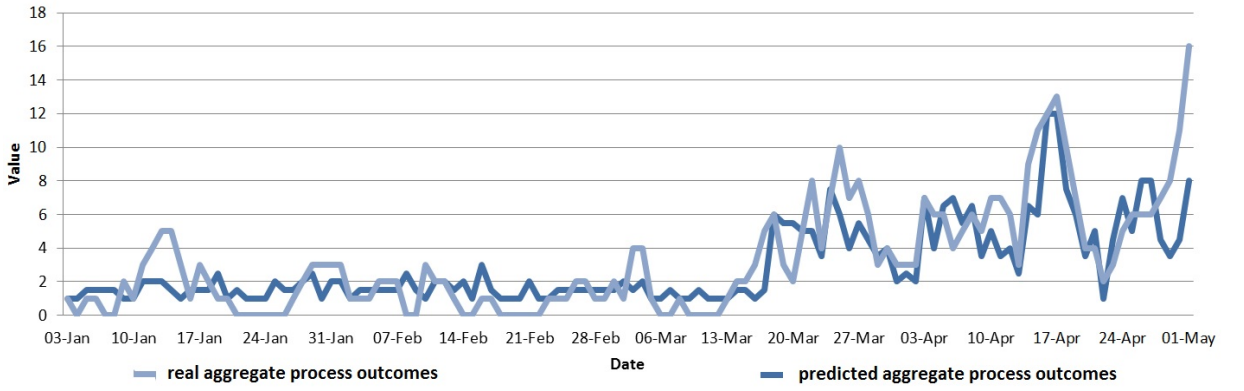


Figure 17: Predicted and real aggregate process outcome (the number of cases completed with a negative outcome, on the vertical axis), Variant 2

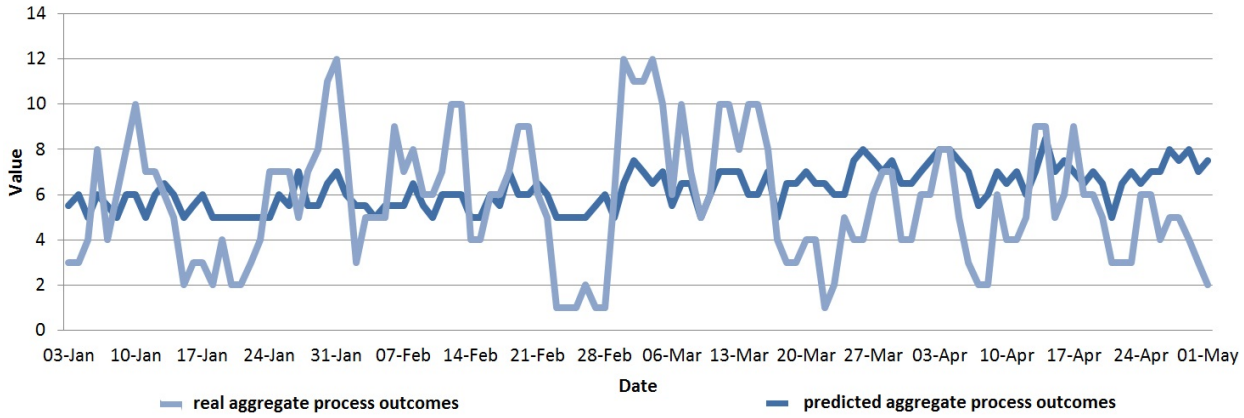


Figure 18: Predicted and real aggregate process outcome (the number of cases completed with a negative outcome, on the vertical axis), Variant 3

Table 3: Coefficient of determination (*R-squared*) for training and test data sets

	Training data set	Test data set
Variant 1	0.89	0.81
Variant 2	0.80	0.66
Variant 3	0.26	0.05

process outcomes are close to the real values for Variant 1, i.e., for the event log with higher probabilities of negative case outcomes (40% and 60%) during risky periods (starting after March 15, e.g., depicted in Figure 15). Predicted and real values of aggregate process outcomes differ more for Variant 2 for which the probability of negative case outcomes during risky periods was 20% and 30%. For Variant 3 we can observe that predictions are not close to the real values which is the expected outcome, as process outcomes and risks do not correlate in this event log. This is also reflected by the *R-squared* statistics provided in Table 3 for different variants for both training and test data sets. As expected, *R-squared* values are the highest for Variant 1 (0.81 for the test set), they are lower for Variant 2 (0.66 for the test set), and they are very low for Variant 3 (0.05 for the test set). The experiment outcomes depicted in Figures 16, 17 and 18 and confirmed by *R-squared* values in Table 3, show that our method can precisely predict aggregate process outcomes if risky process behaviours specified by business analysts affect process outcomes (e.g., in Variants 1 and 2).

Through these experiments we showed how to use our approach to evaluate overall process risk and to predict aggregate process outcomes. We demonstrated different types of risky process behaviours that relate to the control-flow, resource, data and time process perspectives which can be considered by the approach. We showed that the plugin can evaluate overall process risk and identify changes that we introduced in synthetic event logs and we showed that the approach can be used to predict aggregate process outcomes. As expected, we could observe that the quality of predictions depends on the "predictive power" of risky process behaviours specified by process analysts. If the risky process behaviours affect process outcomes, we can get precise predictions (as in Variant 1 in Scenario 2). If they do not actually affect process outcomes (as in Variant 3 in Scenario 2), we learn from data a regression model with a poor ability to predict process outcomes.

4.2. Case study based on a real event log

In addition to the evaluation experiments conducted with synthetic logs, we evaluated our proposed approach using an event log from an Australian company. The work is presented in an anonymised manner in accordance with the confidentially agreement signed between the parties.

The business context of the organisation is as follows. The company provided an event log of a process for one of their core businesses. The process is system-driven but employees are given flexibility to do their work. The log covers all cases for a period of two years. The data was collected from different information systems, anonymised and

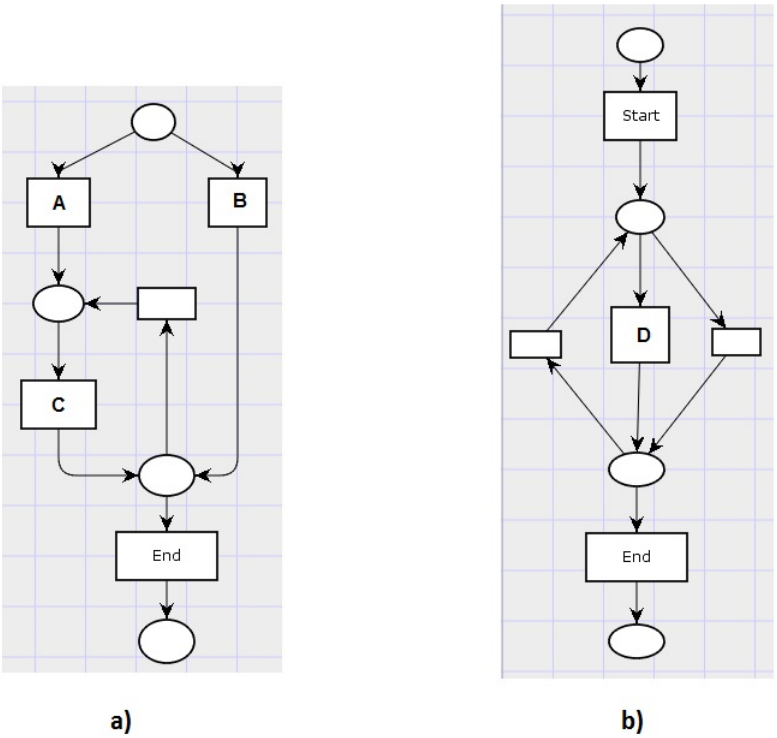


Figure 19: a) A simplified process model created to detect the risky behaviour of skipping task *C* for cases of type *a* (*RPB 1*), b) A simplified process model created to detect the risky behaviour of not completing task *D* within 24 hours (*RPB 2*).

cleaned up based on input from process experts. It was an iterative process that involved a few meetings with company representatives during which we discussed the data requirements and agreed on the data attributes used in the analysis. After this initial data pre-processing the resulting event log contained 17,750 cases and more than 700,000 events. In addition to the information contained in the log, the company also provided its risk register. As expected, some of the risks identified in the risk register are not process-related. The researchers also had discussions with the domain expert to identify a set of process-related risks that could potentially be detected using our approach. Two different types of risky process behaviours captured in the company's risk register were then selected for our experiments. We refer to them here as risky process behaviour 1 (*RPB 1*) and risky process behaviour 2 (*RPB 2*).

We applied our method for overall process risk evaluation described in Section 3.3 to the event log and extracted overall process risk time series, TS_{OPR}, considering these two risky process behaviours separately. For both risky process behaviours we evaluated weekly values of overall process risk (both total risk and mean risk measures) considering cases that were completed during the week. Below we describe the two risky process behaviours, process modeling and further data pre-processing that was conducted, and discuss the evaluation procedure and the results.

RPB 1 manifests itself when a specific task, referred to here as task *C*, is not executed in a case of a specific type, referred to here as type *a* (case type is recorded in the event log as a case attribute, similar to case *urgency* in the example in Section 3.1). We filtered out events that are not required to evaluate overall process risk with respect to *RPB 1* and only used events related to the beginning and the end of a case, and task *C*. We pre-processed the log to create two different case start events – task *A* is recorded as the start event for cases of type *a*, and task *B* is recorded as the start event for cases of other types. We created a process model depicted in Figure 19a that was used to evaluate

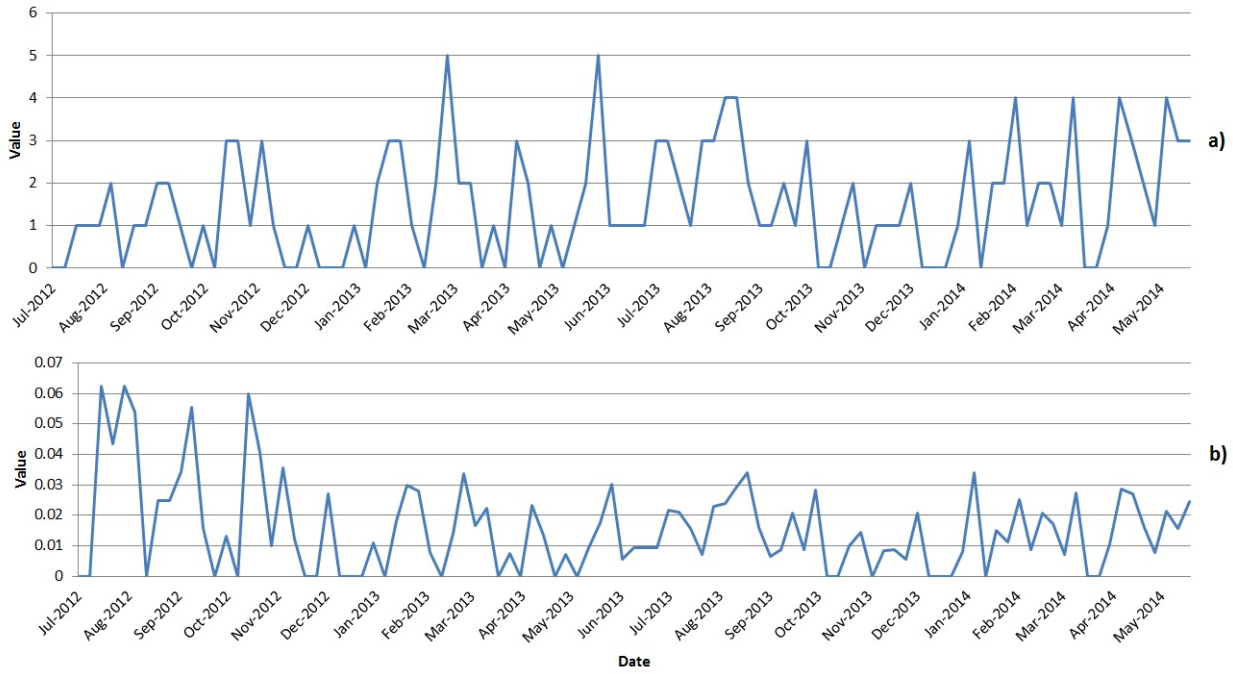


Figure 20: Overall process risk with respect to *RPB 1*: a) total risk, $OPR_T(t)$, on the vertical axis; b) mean risk, $OPR_M(t)$, on the vertical axis.

overall process risk with respect to *RPB 1*. The process model does not allow skipping task *C* in cases that start with task *A*, hence *RPB 1* is detected in all cases in which task *A* was executed but task *C* was skipped. Figure 20 depicts weekly values of overall process risk corresponding to *RPB 1*. We can observe that the values of overall process risk with respect to *RPB 1* are typically very low which means that task *C* is not often skipped in cases of type *a*.

RPB 2 manifests itself when the time difference between *request* and *complete* events related to a specific task, referred to here as task *D*, exceeds 24 hours. We filtered the event log and only used events that correspond to the beginning and the end of a case, and *complete* events for task *D* (it can be performed multiple times in a case). We pre-processed the log, adding to each instance of task *D* data attribute *delayed* whose value is 1 if the time difference between the *request* and *complete* events of task *D* is higher than 24 hours, and 0 otherwise. Figure 19b shows the process model that was used to evaluate overall process risk with respect to *RPB 2*. Task *D* was annotated with a guard (not shown in Figure 19b) that checks whether the value of data attribute *delayed* of task *D* is equal to 0, hence violations are detected in all cases in which the value of the attribute is 1.

Figure 21 depicts weekly values of overall process risk corresponding to *RPB 2*. The values of overall process risk with respect to *RPB 2* (Figure 21) are typically higher than overall process risk values for *RPB 1* (Figure 20) and they also change over time. The increase in overall process risk with respect to *RPB 2* starting from February 2014 (Figure 21a) may be related to changes in the company's process that were implemented at that time, while the causes for the earlier variations of the risk require further investigation.

We also investigated which resources might be contributing to the task delays (*RPB 2*). In order to be able to do this we created five separate event logs each only containing cases in which a given resource completed task *D*. This was done for five resources, referred to here as *R1–R5*, that most frequently executed task *D*. As an example,

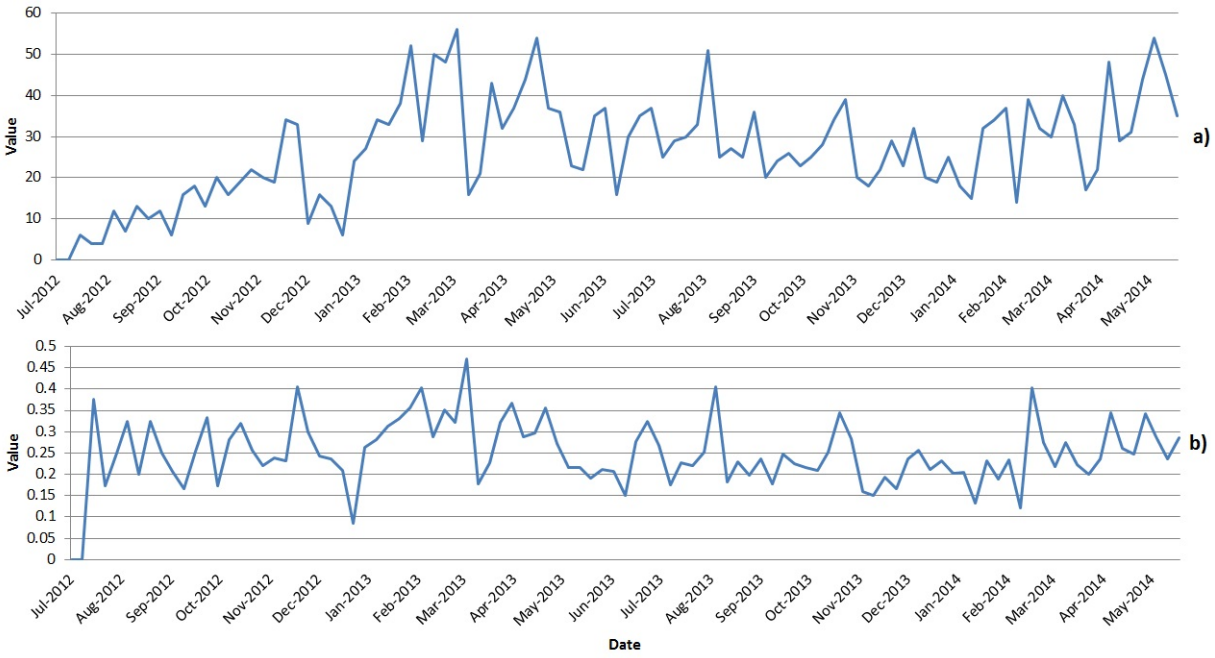


Figure 21: Overall process risk with respect to *RPB 2*: a) total risk, $OPR_T(t)$, on the vertical axis; b) mean risk, $OPR_M(t)$, on the vertical axis.

Figure 22 shows total process risk values associated with resources *R1, R2* and *R3*. We can observe that total risk values increase for resource *R1* till December 2013 and that they decrease afterwards, the values of the overall process risk associated with resource *R3* increase during the first part of 2013 and they decrease afterwards, while the total process risk associated with resource *R2* follows a relatively stable pattern starting from December 2012.

Figure 23a and Figure 23b show total and mean values of the overall process risk associated with resource *R4*. While the total risk values are increasing, the mean risk values are decreasing. This means that the increase in the number of instances of task *D* that are completed by resource *R4* with a delay is related to an increase in the total number of instances of task *D* completed by *R4*. While the total number of delayed tasks completed by resource *R4* increases, the fraction of delayed tasks among those completed by resource *R4* decreases.

As the real-life event log used does not contain information that is necessary to identify other risks from the company's risk register, we did not evaluate overall process risk with respect to all risky process behaviours. The log also does not contain information about organisational outcomes that can be affected by the risks, hence we did not apply our method for predicting aggregate process outcomes based on the current value of overall process risk. Evaluation of our method for predicting aggregate process outcomes with a real data set is a direction for future work.

The case study findings support the practical application of our approach in an organisational setting. By using our approach, we are able to observe trends and detect the differences in the overall process risk. We showed that the approach can help us to track the amount of overall process risk associated with different resources (this is only possible if a risky process behaviour can be linked to a resource, as was the case with *RPB 2*). We demonstrated how considering different measures of overall process risk (total risk and mean risk) can help us to investigate causes of risk variations over time (e.g., the change in overall process risk associated with resource *R4*).

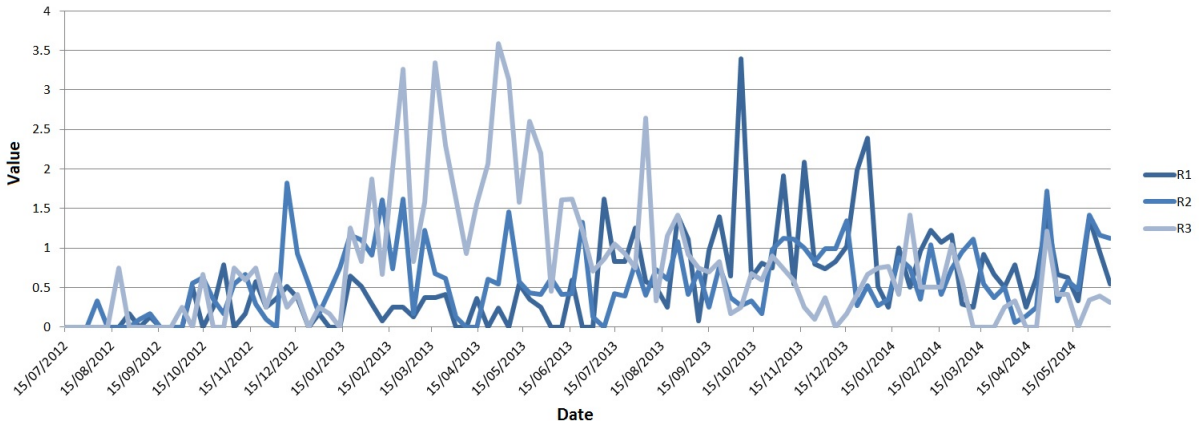


Figure 22: Overall process risk (total risk, $OPR_T(t)$, on the vertical axis) with respect to *RPB 2* associated with resources *R1, R2* and *R3*

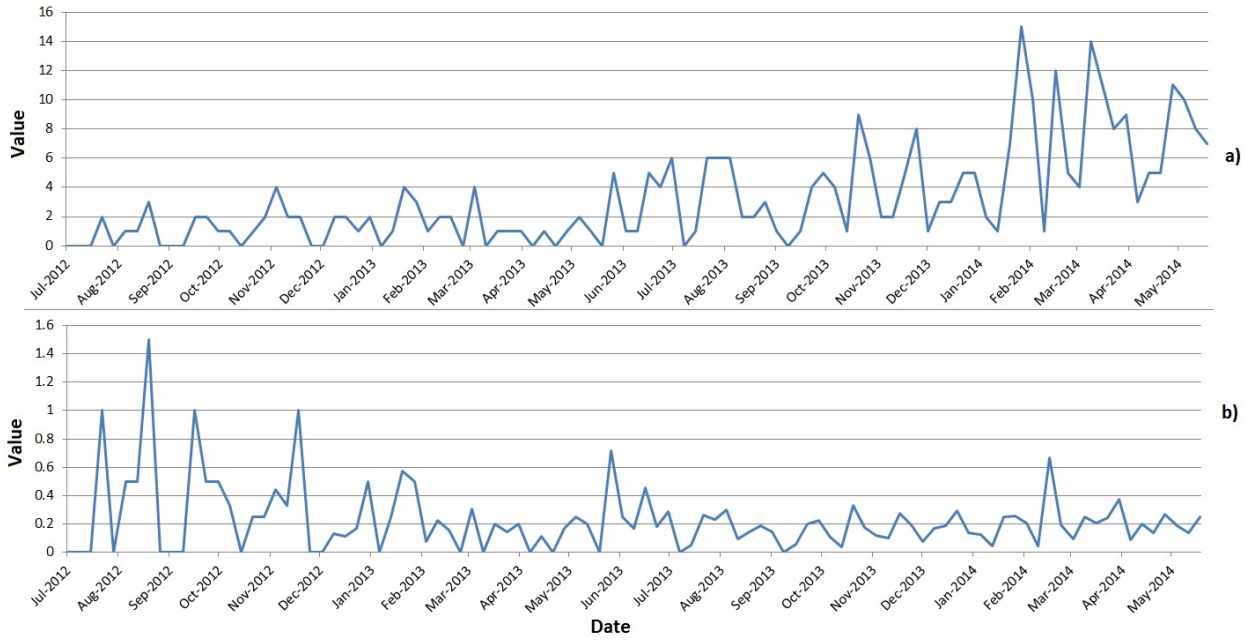


Figure 23: Overall process risk with respect to *RPB 2* associated with resource *R4* a) total risk, $OPR_T(t)$, on the vertical axis b) mean risk, $OPR_M(t)$, on the vertical axis.

5. Assumptions, limitations and future work

We assume that companies know risks which threaten their business processes. Factors that can cause negative process outcomes are often known by domain experts and process analysts. Some companies keep risk registries that specify such risks [22, 38].

Our approach is based on the analysis of process execution data recorded in event logs. Corporate information systems record various information about process executions which can be transformed into event log format[23][33]. We assume that event log data is up-to-date and accurately captures process execution history. Types of information required for the analysis depend on the types of process risks. For example, if we would like to consider as a risky process behaviour an activity delay, we would need to use a log which contains the following event attributes: case identifier, activity name, time stamp and transaction type (both *start* and *complete*). Other risky process behaviours may require richer event logs, e.g., logs that contain information about resources and different data attributes. Our method for the prediction of aggregate process outcomes requires information about case outcomes to be recorded in an event log. We assume that an activity can be completed by one resource[24].

A limitation of our approach is that skipping of an activity cannot be used as a risky process behaviour if the user wishes to evaluate overall process risk in real time, in which case we can only analyse events up to the present, which inevitably means that the algorithm cannot distinguish between an incomplete case and a skipped activity. Compensating for this unavoidable limitation is a direction for future research. Another limitation of our approach is that it can only consider one process outcome. To consider multiple process outcomes using our approach it would be necessary to pre-process an event log and annotate cases with values resulting from a combination of different process outcomes, e.g., a combination of time and quality. The possibility to consider multiple process outcomes is another direction for future work.

We evaluated our method for predicting aggregate process outcomes using synthetic event logs. Evaluation of the method with a real data set is a direction for future work. Another direction for future work is evaluation of the prediction method for processes whose behaviour and outcomes are affected by seasonal fluctuations. We showed examples of risky process behaviours related to different process perspectives which can be identified by our approach, e.g., the execution of a specific activity, repetition of an activity, execution of an activity by a specific resource, completion of an activity with a delay, or execution of an activity with a specific data attribute. A systematic study of different types of risky process behaviours which can and cannot be identified by our approach could be a direction for future research. We devised two measures of overall process risk, total risk and mean risk. Definition and evaluation of other measures of overall process risk, e.g., median risk or risk variation, could be another direction for future work.

[23] We use the XES standard for event log data (http://www.xes-standard.org/).

[24] This limitation is imposed by the XES standard for event log data which only allows one resource per event (current version 2.0).

6. Conclusions

Managing risks in business processes is a key concern for most companies. Existing approaches to process risk identification typically focus on risks in individual process instances. However, processes often run in complex environments and outcomes of one process instance can be affected by events that are external to the process instance. Moreover, managing risks in individual process instances can be too costly. Insufficient attention has been paid to the problem of evaluation of overall process risk based on the analysis of event logs.

In this paper we presented a novel approach that allows us to 1) evaluate overall process risk at a given point in time considering different risk factors across all running process instances; and 2) predict aggregate process outcomes based on the current value of overall process risk. The approach is based on the analysis of information about process executions recorded in event logs. An input to our approach is a process model that models the desired process behaviour (a Petri net with data) and deviations from such a process model are considered as risky process behaviours. To identify such risky process behaviours we use a technique for replaying event logs on a process model [6]. We presented two measures of overall process risk based on the identified risky process behaviours. To track the evolution of overall process risk and facilitate the analysis for a user, we extract overall process risk time series and annotate it with detected change points. To predict aggregate process outcomes, we apply multivariate non-parametric regression to the overall process risk time series (independent variables) and aggregate process outcome time series (the dependent variable) extracted from an event log. We evaluated our approach using synthetic event logs. We showed that the approach can evaluate overall process risk and predict aggregate process outcomes (provided that the risky process behaviours considered affect the outcomes of the process). We also conducted a case study using a real event log from an Australian company. In the case study we showed how our approach can help to track evolution of overall process risk, identify changes, and investigate causes of risk variations by using different measures of overall process risk.

In summary, while it may not be possible to eliminate all process risks during the process design stage, and it is impossible to predict changes to processes imposed by the external environment, it is important to monitor the level of overall process risk and its effect on process outcomes. The approach presented in this paper allows for such an analysis. When a significant change in overall process risk is identified or an adverse aggregate process outcome is predicted, a manager may wish to be alerted and may take mitigation actions, e.g., adding resources from other processes, prioritising tasks assigned to employees or outsourcing some tasks to partners.

Acknowledgments

This research is a part of and funded by the "Risk-Aware Business Process Management" project (ARC DP110100091) for which ethical clearance was obtained (approval number: 1100000288). We used an anonymised data set in our experiments, this research is exempt from ethical review in conformity with the National Statement on Ethical Conduct in Human Research (2007)[25].

[25] https://www.nhmrc.gov.au/guidelines-publications/e72

References

[1] Edward D Arnheiter and John Maleyeff. The integration of lean management and six sigma. *The TQM magazine*, 17(1):5–18, 2005.

[2] J.C.R.P. Bose, W.M.P. van der Aalst, I. Zliobaite, and M. Pechenizkiy. Dealing with concept drifts in process mining. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):154 – 171, 2013.

[3] Raffaele Conforti, Massimiliano de Leoni, Marcello La Rosa, and Wil MP van der Aalst. Supporting risk-informed decisions during business process execution. In Camille Salinesi, Moira C. Norrie, and Oscar Pastor, editors, *Advanced Information Systems Engineering, 25th International Conference, CAiSE 2013, Valencia, Spain, June 17-21, 2013. Proceedings*, volume 7908 of *LNCS*, pages 116–132. Springer, 2013.

[4] Raffaele Conforti, Massimiliano de Leoni, Marcello La Rosa, Wil MP van der Aalst, and Arthur HM ter Hofstede. A recommendation system for predicting risks across multiple business process instances. *Decision Support Systems*, 69:1–19, 2015.

[5] Raffaele Conforti, Giancarlo Fortino, Marcello La Rosa, and Arthur HM ter Hofstede. History-aware, real-time risk detection in business processes. In R. Meersman et al., editor, *On the Move to Meaningful Internet Systems: OTM 2011, LNCS*, volume 7044, pages 100–118. Springer, 2011.

[6] Massimiliano De Leoni and Wil MP van der Aalst. Aligning event logs and process models for multi-perspective conformance checking: An approach based on integer linear programming. In Florian Daniel, Jianmin Wang, and Barbara Weber, editors, *Business Process Management, 11th International Conference, BPM 2013, Beijing, China, August 26-30, 2013*, volume 8094 of *LNCS*, pages 113–129. Springer, 2013.

[7] Massimiliano de Leoni and Wil MP van der Aalst. Data-aware process mining: discovering decisions in processes using alignments. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 1454–1461. ACM, 2013.

[8] Peter Edwards and Paul Bowen. *Risk management in project organisations*. UNSW Press, 2004.

[9] D. Grigori, F. Casati, U. Dayal, and M.C. Shan. Improving business process quality through exception understanding, prediction, and prevention. In *27th International Conference on Very Large Databases (VLDB 2001), Rome, Italy, September 11-14, 2001*, pages 159–168. Morgan Kaufmann Publishers Inc., 2001.

[10] Daniela Grigori, Fabio Casati, Malu Castellanos, Umeshwar Dayal, Mehmet Sayal, and Ming-Chien Shan. Business process intelligence. *Computers in Industry*, 53(3):321–343, 2004.

[11] Damodar Gujarati. *Basic Econometrics*. The McGrawHill Companies, 2004.

[12] Neil Hardie. The prediction and control of project duration: a recursive model. *International Journal of Project Management*, 19(7):401–409, 2001.

[13] International Organization for Standardization. *Risk management: vocabulary = Management du risque: vocabulaire (ISO guide 73)*. Geneva, 2009.

[14] A.K. Jallow, B. Majeed, K. Vergidis, A. Tiwari, and R. Roy. Operational risk analysis in business processes. *BT Technology Journal*, 25(1):168–177, 2007.

[15] Kurt Jensen, Lars Michael Kristensen, and Lisa Wells. Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems. *International Journal on Software Tools for Technology Transfer*, 9(3-4):213–254, 2007.

[16] D. Cryer Jonathan and Kung-sik Chan. *Time series analysis: with applications in R*. Springer, 2008.

[17] Peter Kueng. Process performance measurement system: a tool to support process-based organizations. *Total Quality Management*, 11(1):67–85, 2000.

[18] J. Lijffijt, P. Papapetrou, and K. Puolamaki. Size matters: Finding the most informative set of window lengths. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012*, pages 451–466. Springer, 2012.

[19] JF MacGregor and T Kourti. Statistical process control of multivariate processes. *Control Engineering Practice*, 3(3):403–414, 1995.

[20] Fabrizio Maria Maggi, Chiara Di Francescomarino, Marlon Dumas, and Chiara Ghidini. Predictive monitoring of business processes. In M. Jarke et al., editor, *Advanced Information Systems Engineering, 26th International Conference, CAiSE 2014, Thessaloniki, Greece, June 16-20, 2014. Proceedings*, volume 8484 of *LNCS*, pages 457–472. Springer, 2014.

[21] R. Moeller. *COSO enterprise risk management: understanding the new integrated ERM framework*, chapter 3: Components of COSO ERM, pages 47–93. John Wiley & Sons, Inc., Hoboken, NJ, 2007.

[22] Fiona D Patterson and Kevin Neailey. A risk register database system to aid the management of project risk. *International Journal of Project Management*, 20(5):365–374, 2002.

[23] Maja Pesic, Helen Schonenberg, and Wil MP Van der Aalst. Declare: Full support for loosely-structured processes. In *11th IEEE International Conference on Enterprise Distributed Object Computing, EDOC 2007*, pages 287–287. IEEE, 2007.

[24] A. Pika, W.M.P. van der Aalst, C.J. Fidge, A.H.M. ter Hofstede, and M.T. Wynn. Predicting deadline transgressions using event logs. In M. La Rosa and P. Soffer, editors, *BPM 2012 International Workshops, Tallinn, Estonia, September 3, 2012. Revised Papers*, volume 132 of *LNBIP*, pages 211–216. Springer, 2013.

[25] A. Pika, W.M.P van der Aalst, C.J. Fidge, A.H.M. ter Hofstede, and M.T. Wynn. Profiling event logs to configure risk indicators for process delays. In Camille Salinesi, Moira C. Norrie, and Oscar Pastor, editors, *25th International Conference, CAiSE 2013, Valencia, Spain, June 17-21, 2013. Proceedings*, volume 7908 of *LNCS*, pages 465–481. Springer, 2013.

[26] Jeff Racine and Qi Li. Nonparametric estimation of regression functions with both categorical and continuous data. *Journal of Econometrics*, 119(1):99–130, 2004.

[27] M. Rosemann and M. zur Muehlen. Integrating risks in business process models. In B. Campbell, H. Underwood, and D. Bunker, editors, *Proceedings of the 16th Australasian Conference on Information Systems*, pages 1–10, Australia, New South Wales, Sydney, 2005. Australasian Chapter of the Association for Information Systems.

[28] G.J. Ross and N.M. Adams. Two nonparametric control charts for detecting arbitrary distribution changes. *Journal of Quality Technology*, 44(2):102–116, 2012.

[29] Standards Australia and Standards New Zealand. *Risk management: principles and guidelines (AS/NZS ISO 31000:2009)*. Sydney, NSW, Wellington, NZ, 3rd edition, 2009.

[30] Suriadi Suriadi, Burkhard Weiss, Axel Winkelmann, Arthur ter Hofstede, Michael Adams, Raffaele Conforti, Colin Fidge, Marcello La Rosa, Chun Ouyang, Michael Rosemann, et al. Current research in risk-aware business process management–overview, comparison, and gap analysis. *Communications of the Association for Information Systems*, 34(1), 2014.

[31] Vicknayson Thevendran and MJ Mawdesley. Perception of human risk factors in construction projects: an exploratory study. *International Journal of Project Management*, 22(2):131–137, 2004.

[32] Wil MP van der Aalst. Formalization and verification of event-driven process chains. *Information and Software technology*, 41(10):639–650, 1999.

[33] W.M.P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, Berlin, 2011.

[34] W.M.P. van der Aalst, H.T. de Beer, and B.F. van Dongen. Process mining and verification of properties: An approach based on temporal logic. In Robert Meersman and Zahir Tari, editors, *OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2005, Agia Napa, Cyprus, October 31 - November 4, 2005, Proceedings, Part I*, volume 3760 of *LNCS*, pages 130–147, 2005.

[35] W.M.P. van der Aalst and Schahram Dustdar. Process mining put into context. *Internet Computing, IEEE*, 16(1):82–86, 2012.

[36] Stephen A White. *BPMN modeling and reference guide: understanding and using BPMN*. Future Strategies Inc., 2008.

[37] J.A. Wickboldt, L.A. Bianchin, R.C. Lunardi, L.Z. Granville, L.P. Gaspary, and C. Bartolini. A framework for risk assessment based on analysis of historical information of workflow execution in it systems. *Computer Networks*, 55(13):2954–2975, 2011.

[38] Terry M Willams. Using a risk register to integrate risk management in project definition. *International Journal of Project Management*, 12(1):17–22, 1994.

[39] Christian Wolter, Andreas Schaad, and Christoph Meinel. Task-based entailment constraints for basic workflow patterns. In *Proceedings of the 13th ACM symposium on access control models and technologies*, pages 51–60. ACM, 2008.