Decision Support Systems 69 (2015) 1–19

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

A recommendation system for predicting risks across multiple business process instances

Raffaele Conforti [a,*], Massimiliano de Leoni [b,c], Marcello La Rosa [a,d], Wil M.P. van der Aalst [a,b], Arthur H.M. ter Hofstede [a,b]

[a] *Queensland University of Technology, Australia*
[b] *Eindhoven University of Technology, The Netherlands*
[c] *University of Padua, Italy*
[d] *NICTA Queensland Lab, Brisbane, Australia*

ARTICLE INFO

Article history:
Received 12 February 2014
Received in revised form 3 October 2014
Accepted 22 October 2014
Available online 6 November 2014

Keywords:
Business process management
Risk management
Risk prediction
Job scheduling
Work distribution
YAWL

ABSTRACT

This paper proposes a recommendation system that supports process participants in taking risk-informed decisions, with the goal of reducing risks that may arise during process execution. Risk reduction involves decreasing the likelihood and severity of a process fault from occurring. Given a business process exposed to risks, e.g. a financial process exposed to a risk of reputation loss, we enact this process and whenever a process participant needs to provide input to the process, e.g. by selecting the next task to execute or by filling out a form, we suggest to the participant the action to perform which minimizes the predicted process risk. Risks are predicted by traversing decision trees generated from the logs of past process executions, which consider process data, involved resources, task durations and other information elements like task frequencies. When applied in the context of multiple process instances running concurrently, a second technique is employed that uses integer linear programming to compute the optimal assignment of resources to tasks to be performed, in order to deal with the interplay between risks relative to different instances. The recommendation system has been implemented as a set of components on top of the YAWL BPM system and its effectiveness has been evaluated using a real-life scenario, in collaboration with risk analysts of a large insurance company. The results, based on a simulation of the real-life scenario and its comparison with the event data provided by the company, show that the process instances executed concurrently complete with significantly fewer faults and with lower fault severities, when the recommendations provided by our recommendation system are taken into account.

1. Introduction

A *process-related risk* measures the likelihood and the severity that a negative outcome, also called *fault*, will impact on the process objectives [1]. Failing to address process-related risks can result in substantial financial and reputational consequences, potentially threatening an organization's existence. Take for example the case of Société Générale, which went bankrupt after a €4.9B loss due to fraud.

Legislative initiatives like Basel II [2] and the Sarbanes–Oxley Act[1] reflect the need to better manage business process risks. In line with these initiatives, organizations have started to incorporate process risks as a distinct view in their operational management, with the aim to effectively *control* such risks. However, to date there is little guidance as to how this can be concretely achieved.

As part of an end-to-end approach for risk-aware Business Process Management (BPM), in [3–5] we proposed several techniques to model risks in executable business process models, detect them as early as possible during process execution, and support process administrators in mitigating these risks by applying changes to the running process instances. However, the limitation of these efforts is that risks are not *prevented*, but rather *acted upon* when their likelihood exceeds a tolerance threshold. For example, a mitigation action may entail skipping some tasks when the process instance is very likely to exceed the defined maximum cycle time. While effective, mitigation comes at the cost of modifying the process instance, often by skipping tasks or rolling back previously-executed tasks, which may not always be acceptable. Moreover, we have shown that it is not always possible to mitigate all process risks [4]. For example, rolling back a task for the sake of mitigating

* Corresponding author at: Science and Engineering Faculty, Queensland University of Technology, 2 George St, Brisbane, QLD, 4000, Australia.
E-mail addresses: raffaele.conforti@qut.edu.au (R. Conforti), m.d.leoni@tue.nl (M. de Leoni), m.larosa@qut.edu.au (M. La Rosa), w.m.p.v.d.aalst@tue.nl (W.M.P. van der Aalst), a.terhofstede@qut.edu.au (A.H.M. ter Hofstede).
[1] www.gpo.gov/fdsys/pkg/PLAW-107publ204.

http://dx.doi.org/10.1016/j.dss.2014.10.006
0167-9236/

a risk of cost overrun, may not allow the full recovery of the costs incurred in the execution of that task.

To address these limitations we propose a recommendation system that supports process participants in taking risk-informed decisions, with the aim to *reduce* process risks preemptively. A process participant takes a decision whenever they have to choose the next task to execute out of those assigned to them at a given process state, or via the data they enter in a user form. This input from the participant may influence the risk of a process fault to occur. For each such input, the technique returns a risk prediction in terms of the likelihood and severity that a fault will occur if the process instance is carried out using that input. This prediction is obtained via decision trees which are trained using historical process data such as process variables, resources, task durations and frequencies. The historical data of a process is observed using decision trees which are built from the execution logs of the process, as recorded by the IT systems of an organization.

This way, the participant can take a risk-informed decision as to task to execute next, or can learn the predicted risk of submitting a form with particular data. If the instance is subjected to multiple potential faults, the predictor can return the weighted sum of all fault likelihoods and severities, as well as the individual figures for each fault. The weight of each fault can be determined based on the severity of the fault's impact on the process objectives.

The above technique only provides “local” risk predictions, i.e. predictions relative to a specific process instance. In reality, however, multiple instances of (different) business processes may be executed at any time. Thus, we need to find a risk prediction for a specific process instance that does not affect the prediction for other instances. The interplay between risks relative to different instances can be caused by the sharing of the same pool of process participants: two instances may require the same scarce resource. In this setting, a sub-optimal distribution of process participants to the set of tasks to be executed may result in a risk increase (e.g. overtime or cost overrun risk). To solve this problem, we equipped our recommendation system with a second technique, based on integer linear programming, which takes input from the risk prediction technique, to find an *optimal distribution* of process participants to tasks. By optimal distribution we mean one that minimizes the overall execution time (i.e. the time taken to complete *all* running instances) while minimizing the overall level of risk. This distribution is used by the recommendation system to suggest to process participants the next task to perform.

We operationalized our recommendation system on top of the YAWL BPM system by extending an existing YAWL plug-in and by implementing two new custom YAWL services. This implementation prompts process participants with risk predictions upon filling out a form or for each task that can be executed. We then evaluated the effectiveness of our recommendation system by conducting experiments using a claim handling process in use at a large insurance company. With input from a team of risk analysts from the company, this process has been extensively simulated on the basis of a log recording one year of completed instances of this process. The recommendations provided by our recommendation system significantly reduced the number and severity of faults in a simulation of a real life scenario, compared to the process executed by the company as reflected by the event data. Further, the results show that it is feasible to predict risks across multiple process instances without impacting on the execution performance of the BPM system.

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 contextualizes the recommendation system within our approach for managing process-related risks, while Section 4 presents the YAWL language as part of a running example. Next, Section 5 defines the notions of event logs and faults which are required to explain our techniques. Section 6 describes the technique for predicting risks in a single process instance while Section 7 extends this technique to the realm of multiple process instances running concurrently. Section 8 and Section 9 discuss the implementation and evaluation of the recommendation system, respectively. Finally, Section 10 concludes the paper. A provides the formal definition of a YAWL specification, the algorithms to generate a prediction function, and technical proofs of two lemmas presented in Section 7.

2. Related work

The approach presented in this paper is related to work on risk prediction, job scheduling, operational support and work-item distribution for business processes. In this section we review the state of the art in these fields to motivate the need for our approach.

2.1. Risk prediction

Various risk analysis methods such as OCTAVE [6], CRAMM [7] and CORAS [8] have been defined which provide elements of risk-aware process management. Meantime, academics have recognized the importance of managing process-related risks. However, risk analysis methods only provide guidelines for the identification of risks and their mitigation, while academic efforts mostly focus on risk-aware BPM methodologies in general, rather than on concrete approaches for risk prediction [9].

An exception is made by the works of Pika et al. [10] and Suriadi et al. [11]. Pika et al. propose an approach for predicting overtime risks based on statistical analysis. They identify five process risk indicators whereby the occurrence of these indicators in a trace indicates the possibility of a delay. Suriadi et al. propose an approach for Root Cause Analysis based on classification algorithms. After enriching a log with information like workload, occurrence of delay, and involvement of resources, they use decision trees to identify the causes of overtime faults. The cause of a fault is obtained as a disjunction of conjunctions of the enriching information. Despite looking at the same problem from different perspectives, these two approaches have quite similar results. These two approaches suffer from the limitation of not considering the data perspective. Further, they limit their scope to the identification of indicators of risks or of causes of faults to support overtime risks only.

In previous work, we presented a wider approach which aims to bridge the gap between risk and process management. This approach consists of two techniques. The first one [3,5] allows process modelers to specify process-related faults and related risks on top of (executable) process models, and to detect them at run-time when their risk likelihood exceeds a tolerance threshold. Risks are specified as conditions over control-flow, resources and data aspects of the process model. The second technique [4] builds on top of the first one to cover risk mitigation. As soon as one or more risks are detected which are no longer tolerable, the technique proposes a set of alternative mitigation actions that can be applied by process administrators. A mitigation action is a sequence of controlled changes on a process instance affected by risks, which takes into account a snapshot of the process resources and data, and the current status of the system in which the process is executed.

For a comprehensive review and comparative analysis of work at the intersection of risk management and BPM, we refer to [9].

2.2. Job scheduling

The problem of distributing work items to resources in business process execution shares several similarities with the job-shop scheduling [12–15]. Job-shop scheduling concerns M jobs that need to be assigned to N machines, with $N < M$, while trying to minimize the make-span, i.e. the total length of the schedule. Jobs may have constraints, e.g. job i needs to finish before job j can be started, certain jobs can only be performed by given machines.

Unfortunately, these approaches are intended for different settings and cannot be specialized for risk-informed work-item assignment. To our knowledge, the techniques of job-shop scheduling are unaware of