Comparing and Combining Predictive Business Process Monitoring Techniques

Andreas Metzger, Philipp Leitner, Dragan Ivanović, Eric Schmieders, Rod Franklin, Manuel Carro, Schahram Dustdar, and Klaus Pohl

Abstract—**Predictive business process monitoring aims at forecasting potential problems during process execution before they occur so that these problems can be handled proactively. Several predictive monitoring techniques have been proposed in the past. However, so far those prediction techniques have been assessed only independently from each other, making it hard to reliably compare their applicability and accuracy. We empirically analyze and compare three main classes of predictive monitoring techniques, which are based on machine learning, constraint satisfaction, and Quality-of-Service (QoS) aggregation. Based on empirical evidence from an industrial case study in the area of transport and logistics, we assess those techniques with respect to five accuracy indicators. We further determine the dependency of accuracy on the point in time during process execution when a prediction is made in order to determine lead-times for accurate predictions. Our evidence suggests that, given a lead-time of half of the process duration, all predictive monitoring techniques consistently provide an accuracy of at least 70%. Yet, it also becomes evident that the techniques differ in terms of how accurately they may predict violations and nonviolations. To improve the prediction process, we thus exploit the characteristics of the individual techniques and propose their combination. Based on our case study data, evidence indicates that certain combinations of techniques may outperform individual techniques with respect to specific accuracy indicators. Combining constraint satisfaction with QoS aggregation, for instance, improves precision by 14%; combining machine learning with constraint satisfaction shows an improvement in recall by 23%.**

Manuscript received March 29, 2013; revised August 9, 2013 and January 16, 2014; accepted June 16, 2014. Date of publication August 28, 2014; date of current version January 13, 2015. This work was supported in part by the European Union's Seventh Framework Programme FP7/2007-2013 under Grant 604123 (FIspace), Grant 285598 (FInest), and Grant 215483 (S-Cube), in part by the Austrian Science Fund under Project P23313-N23 (Audit 4 SOAs), in part by the Spanish Ministry of Economy under Project 2008-05624/TIN (Doves), Project TIN2012-39391-C04-03, and Project TIN2012-39391-C04-01 (StrongSoft), and in part by the Madrid Regional Government under Project CAM S2009TIC-1465 (Prometidos-CM). This paper was recommended by Associate Editor J. Lu.

A. Metzger, E. Schmieders, and K. Pohl are with paluno (The Ruhr Institute for Software Technology), University of Duisburg-Essen, 45127 Essen, Germany (e-mail: andreas.metzger@paluno.uni-due.de; eric.schmieders@paluno.uni-due.de; klaus.pohl@paluno.uni-due.de).

P. Leitner is with the University of Zurich, Zurich 8050, Switzerland (e-mail: leitner@ifi.uzh.ch).

D. Ivanović and M. Carro are with the IMDEA Software Institute and the Universidad Politécnica de Madrid, Madrid 28223, Spain (e-mail: dragan.ivanovic@imdea.org; manuel.carro@imdea.org).

R. Franklin is with the Kühne Logistics University, Hamburg 20457, Germany, and also with the Kühne + Nagel Management AG, Schindellegi, Switzerland (e-mail: rod.franklin@the-klu.org).

S. Dustdar is with the Vienna University of Technology, Vienna 1040, Austria (e-mail: dustdar@infosys.tuwien.ac.at).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2014.2347265

Index Terms—**Business data processing, failure analysis, forecasting, neural network applications, transportation.**

I. Introduction

MODERN information technology offers unprecedented means for real-time monitoring and control of cross-organizational business processes. Cloud computing [1], the internet of things [2], and service-oriented computing [3] are increasingly adopted by industry to increase visibility, agility and efficiency of business processes. The application domains in which this is happening include agriculture [4], food [5], manufacturing [6], energy [7], transport and logistics [8], as well as supply chain management [9].

One emerging opportunity facilitated by increased visibility of business processes is predictive business process monitoring, which aims at detecting potential problems during process execution ahead of time so that these problems can be anticipated and thus proactively managed and mitigated [10]. Predictive business process monitoring makes it possible to apply countermeasures to prevent the occurrence of problems and it can help prepare mitigation and replanning mechanisms for an upcoming problem in order to increase the ability to respond to the problem [11]. As an example, if a delay in delivery time is predicted during the execution of a freight transport process, faster means of transport or alternative transport routes could be scheduled proactively and before the delay actually occurs, if such rescheduling brings an advantage to the overall execution time of the process.

Traditionally, business process management solutions offered limited capabilities for predictive monitoring of individual business process instances. While techniques for real-time business process monitoring, such as business activity monitoring [12], exploit real-time data to support dynamic decision making and optimization of running business processes, those solutions lack predictive capabilities. On the other hand, predictive approaches for business processes, such as business (process) intelligence [13], focus on longer-term predictions based on historic data or on predictions of aggregate indicators, such as key performance indicators.

Recently, techniques for predictive monitoring of individual process instances have been proposed. These predictive business process monitoring techniques leverage, for instance, data mining [14], machine learning [15], Quality-of-Service (QoS) aggregation [16], and predictive event processing [17]. Initial empirical evidence has been presented on the applicability and usefulness of these prediction techniques. Yet, empirical evaluation so far focused on the assessment of each of the prediction techniques in isolation. As a consequence, different benchmarks were used and different assumptions were made in each of these evaluations. This makes it hard to reliably compare their applicability and accuracy.

To address the aforementioned gap, we perform an empirical comparison of three main classes of predictive monitoring techniques: machine learning [18], constraint satisfaction [19], and QoS aggregation [20]. We analyze their applicability and prediction accuracy in an industrial case study in the area of transport and logistics. The case study covers a global supply chain involving actual transport and logistics services managed by a large, international forwarding company. The data set underlying our case study contains monitoring data of 3942 business process instances collected over a period of five months, comprising a total of 56 802 executions of business activities.

Our evidence suggests that the prediction techniques may deliver good prediction accuracy with feasible lead-times; e.g., given a lead-time of half of the process duration, all predictive monitoring techniques consistently provide an accuracy of at least 70%. Yet, our evidence also indicates that the techniques differ in terms of how accurately they may predict violations and non-violations. We exploit these differences by combining individual techniques with the aim of improving the prediction process. Our empirical findings indicate that combinations of techniques may outperform individual techniques with respect to specific accuracy indicators; e.g., combining constraint satisfaction with QoS aggregation improves precision by 14%, combining machine learning with constraint satisfaction improves recall by 23%.

The case study in this paper focuses on violations of process performance. Specifically, we aim at predicting, during process execution, whether the effective time of delivery will exceed the planned time of delivery. It should be noted that the predictive monitoring techniques we employ are not limited to performance but might work also for other numeric and monotonic quality attributes. For instance, such quality attributes include time of booking cancellations (to predict late cancellations leading to underutilization of transport assets [8]) or cargo volumes (to predict volume discrepancies which may lead to over- or under-loading situations [17]). Further, the prediction techniques might be employed to predict multiple quality attributes. In the case of fresh produce, as an example, such multiple quality attributes could be time of delivery and quality of delivered product.

The remainder of the paper is structured as follows. Section II discusses how our contribution advances the state of the art in predictive business process monitoring. Section III provides background and technical details on the three predictive monitoring techniques that we compare in this paper. Section IV describes the design and execution of our case study. Section V presents and discusses our findings for the individual techniques. Section VI depicts the results for combining the prediction techniques.

II. Related Work

This section summarizes work related to monitoring and prediction of business processes and discusses how our contributions differ from that work.

Business activity monitoring (BAM) [12] aims to provide continuous situation awareness and access to current business process performance indicators [21]. BAM facilitates immediate operational decisions, based on real-time data, i.e., data which has arrived in the past seconds or minutes and thus focuses on capturing and processing business events with minimum latency [22]. In contrast to more traditional real-time monitoring approaches, BAM draws information from multiple systems and across organizations [23]. Although BAM solutions exploit real-time data to support dynamic decision making and optimization of business processes, BAM solutions lack predictive monitoring capabilities.

Business (process) intelligence [13], which belongs to the broader class of business analytics, aims to determine future trends based on past observations of business process execution and to forecast the impact of business process changes [22]. By exploiting simulation, data mining or optimization techniques, business intelligence focuses on providing longer-term predictions. In order to reduce the latency between a business event and the reaction to the event (see [24]), real-time business intelligence, also known as operational intelligence, has been proposed [25]. Operational intelligence advocates real-time decision making, and thus introduces capabilities for real-time, continuous processes analysis. Typically, operational intelligence techniques focus on the real-time prediction of aggregate indicators computed for a set of process instances, such as key performance indicators that abstract from individual process performance measures. As an example, Castellanos *et al.* [14] introduce a data-mining-based technique for predicting aggregated metric values of process performance indicators, such as average processing time. This means, that operational intelligence techniques typically fall short in predicting performance of individual process instances, and thus only provide limited support to proactively address problems related to the execution of a single process instance.

For what concerns predictive monitoring of individual business process instances, several solutions have been presented in the literature, each focusing on one specific prediction technique. Castellanos *et al.* [14] and Grigori *et al.* [26] propose employing data mining to learn prediction models for process instances. The authors provide evidence for the usefulness of their technique and a high-level analysis of the technique's prediction accuracy. They observe that accuracy increases as process instance execution proceeds, and that "critical" nodes in process execution affect accuracy. In fact, those findings are corroborated by our empirical results. However, the authors do not provide a detailed analysis of accuracy in terms of false positive and negative predictions, an important aspect that can differ significantly between prediction techniques as we observe in Section V-D.

Van der Aalst *et al.* [16] propose predicting the remaining execution time of running process instances by considering process models that are mined from process logs. Process mining allows addressing situations in which the actual process model is not fixed or where it is unknown. Predictions are performed using QoS aggregation rules (see Section III-C) employing descriptive statistics over past observations, such as average, variance, or mean. Bevacqua *et al.* [27] suggest to automatically cluster mined process instances based on context factors (such as workload or calendar time) and to derive prediction models for each of those clusters. The empirical evidence gathered from a real-world example indicates that this clustering-based prediction approach leads to an improvement in prediction accuracy when compared, for instance, with the approach of van der Aalst *et al.* [16] and Rogge-Solti and Weske [28] provide yet another angle to improve the prediction of the remaining execution time of running process instances. In contrast to other prediction techniques, the technique of Rogge-Solti and Weske [28] explicitly considers expected events that have not yet occurred. Business processes are modeled as stochastic Petri nets. Monte Carlo simulations are employed to perform the actual prediction. Rogge-Solti and Weske [28] provide empirical evidence that indicates significant accuracy improvements over the prediction technique of van der Aalst *et al*. [16]. All three aforementioned techniques aim at predicting the remaining time until process completion. Empirical evaluation resorts to prediction error metrics (such as mean absolute error and root mean squared error) to assess prediction accuracy. Despite indicating the promising potential of the proposed prediction techniques, prediction error metrics do not directly unveil how well a technique may predict the actual need for proactively managing and mitigating problems. This, however, is the aim of the metrics that we introduce in Section IV-B, which take into account planned process performance and thus can be used to determine true and false predictions (see [29]).

Kang *et al.* [15] employ machine learning to augment process monitoring with predictive capabilities. To this end, they introduce an extension of support vector machines (see Section III-A) for the purpose of prediction. Based on a data set of 1030 instances, Kang *et al*. [15] provide initial evidence for the applicability of machine learning techniques for predictive business process monitoring. Our results can thus be considered to reinforce their initial findings in a real industry setting.

Feldmann *et al.* [17] provide empirical evidence on the applicability and usefulness of predictive complex event processing (CEP). Generally speaking, CEP is a framework whose purpose is the detection of complex events by analyzing and aggregating incoming raw events [30]. CEP is widely used in business activity monitoring solutions (see [12], [22]) However, as pointed out above, existing business activity monitoring solutions are reactive, i.e., only respond to actual deviations observed during business process executions. Feldmann *et al*. [17] leverage initial proposals to extend CEP with predictive capabilities (such as [31], [32]) and develop a concrete combination of a predictive model with probabilistic rules that fire alerts based on predictions. Feldmann *et al.*'s [17] findings are based on the same raw Cargo 2000 data that is also the basis for our case study. However, in contrast to our case study, the authors address prediction of volume discrepancies, thereby supporting the proactive management of over- and under-load situations of individual air transports.

Summarizing the discussion of related work, this paper provides three main novel contributions. First, using a common case study, we provide an in-depth comparison of predictive monitoring techniques, fostering a better understanding of each of the technique's applicability and limitations in practice. Second, we complement existing proposals for predictive process monitoring (so far advocating data mining, machine learning, QoS aggregation, and predictive CEP) by proposing constraint satisfaction as a further promising technique. Third, we present different combinations of prediction techniques and provide evidence that suggests that such a combination may outperform individual techniques with respect to specific accuracy indicators.

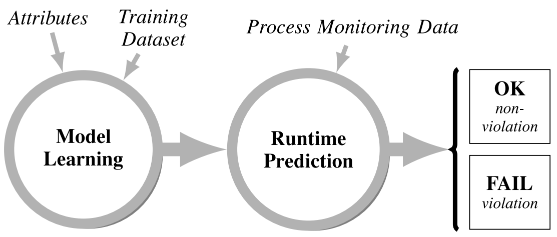

Fig. 1. Main steps of machine-learning technique.

III. Predictive Monitoring Techniques

This section introduces the predictive monitoring techniques we experimentally compare in this paper. We provide background information about the class of techniques, as well as details about each of the concrete techniques used. We have purposefully selected one concrete, nonspecialized technique from each class, avoiding special-purpose or ad-hoc techniques for the sake of generalizability of the results. In some cases, it may have been possible to achieve better results with less general, more case-study-specific approaches or parameterizations, but then the conclusions drawn might not be valid for similar cases where that specific technique or parameterization is not applicable.

A. Machine Learning

Machine learning deals with (semi-)automated learning of relationships or behavior from data. Fig. 1 outlines the overall procedure of the machine learning-based predictive monitoring technique. Essentially, there are two separate steps. First, in the model learning step, a predictor is trained from an existing set of training data, e.g., a data set containing historical process execution traces. In this first step, we also need to decide which data is most important for prediction. This can be done manually by a human or semi-automatically using techniques such as principal component analysis [33]. Second, in the runtime prediction step, data collected from the running process instance is used as input to the trained machine learning model to generate a concrete prediction for this process instance.

Many different machine learning approaches could be used to implement the aforementioned machine learning-based prediction technique. Among the better-known and most relevant approaches, we can cite artificial neural networks (ANNs) [34], [35], decision trees [36], support vector machines (SVMs) [37], Bayes networks [38], and cluster analysis [39]. In what follows, we describe an implementation using ANNs, as we have used this approach to good success in earlier research [18], [40]. ANNs are a machine learning approach inspired by the inner workings of the human brain and, therefore, belong to the larger group of bio-inspired computing techniques [41]. Basically, ANNs consist of one or more layers of nodes (neurons) and directed connections between neurons. There are many possible interconnection schemes (topologies), but ANNs often consist of an input layer, an output layer, and one or several intermediate layers. In each layer, every node is connected with all nodes in the next layer. This architecture is called feed-forward multilayer perceptron [34]. Central to the correct functioning of a neural network is the importance (weight) of each connection between nodes. These weights are learned via training from historical data. Typically, training starts with random weights and is continued in several iterations until a set of weights is found that maximizes the correlation between the outcome of the network and the actual outcome of the examples. Different algorithms can be used to adapt the weights, such as back-propagation [35].

In order to employ ANNs for predictive monitoring of the transport process in our case study, we firstly need to define one or more concrete points in the process structure where we want to carry out predictions. We refer to these points as checkpoints. As an example, such checkpoints could be established before each activity in a business process.

For each checkpoint, we select the process data that is already available and which is most important for prediction (as mentioned above), i.e., we define the attributes that are most relevant and use them to train the ANN. As an example, we use the execution times of the services that are executed up to the checkpoint. At runtime, the technique triggers a prediction whenever the execution of a business process instance passes the checkpoint. It generates a prediction using the runtime process monitoring data collected from this specific process instance. Then, the technique compares the generated prediction with the planned process performance and decides whether a violation might occur.

As explained above, machine learning requires data about the execution of past business process instances in order to train the prediction models. A general concern in machine learning is that only with enough of such past observations, accurate predictions can be expected, and, oftentimes, more data is better than improved machine learning algorithms [42].

B. Constraint Satisfaction

Constraint satisfaction [43] is a problem-solving approach that expresses the conditions that a solution has to meet as constraints, i.e., equations involving problem variables over some (e.g., numeric) domains. This set of constraints is termed the constraint satisfaction problem (CSP). A solution to a CSP is an assignment of values to variables that satisfies all constraints. This assignment is usually automatically generated by a constraint solver. The availability of powerful constraint solvers led to the wide-spread use of constraint modeling in many application domains ranging from logistics to industrial optimization.

Constraint solvers are designed to respect some correctness criteria. When a solver determines that there is no solution to a given problem, then we know for sure that it is not solvable. However, in some cases, a solver may not be able to decide whether there is a solution or not. Therefore, the absence of a solution always means that the system is unsatisfiable, while producing a candidate solution means that the system may be satisfiable, but not neccessarily so.[1]

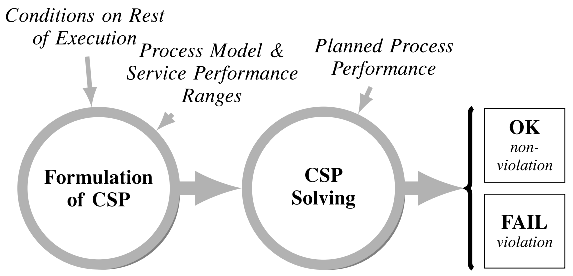
Conditions on Rest of Execution
Process Model & Service Performance Ranges
Planned Process Performance
Formulation of CSP
CSP Solving
OK
non-violation
FAIL
violation

Fig. 2. Main steps of constraint satisfaction technique.

For the application of the constraint satisfaction technique (and also for QoS aggregation, see Section III-C), business process models are seen as structured "programs" whose constructs include both the execution of basic services as well as complex constructs: sequences of service executions, conditional executions, loops, and synchronization points (known also as and-joins).

The CSP we formulate in this context is concerned with quality attributes we wish to analyze—in particular, execution time. The execution time of complex constructs is structurally derived from that of their components: the duration of a sequence of service executions s_1 and s_2 with respective execution times T_1 and T_2 is $T_{\text{seq}} = T_1 + T_2$, while the duration of their and-join (parallel execution) is $T_{\text{and}} = \max(T_1, T_2)$.[2] By using the structure of the business process, a set of equations is progressively built. These equations relate the execution times of the basic services with that of the progressively higher constructs, up to the level of the entire process. For the individual services of a business process, whose internal details are unknown, we need to rely on collected evidence about their behavior. In particular, for the execution times of individual services, we use empirically determined bounds, which give ranges of the form $\ell \leq T \leq u$.

The equations derived from the structure of the business process, together with the ranges for the individual services, form a CSP whose formulation is the initial step in the constraint-based predictive monitoring technique sketched in Fig. 2. All the steps involved in formulating the CSP are purely mechanical and thus do not require involvement of technical experts. Those steps may even be automated to generate the CSP from the process model and statistical data about past service executions. A more detailed description is available in [19].

In the second step, we attempt to solve this CSP taking into account the expected process performance, i.e., the "end-to-end" quality target (a time limit, in our case). We construct

[1]This is an oversimplification for conciseness: in practice, it is often possible to tell apart a true solution, which guarantees satisfiability, from a candidate solution.

[2]Assuming a parallel construct involving s_1 and s_2 always executes the services really in parallel. If they can be executed one after the other, the constraint should be $\max(T_1, T_2) \leq T_{\text{and}} \leq T_1 + T_2$.

two different cases to check: one where we require that the overall time limit is respected (corresponding to the OK case), and another one where we force the time limit to be exceeded (the FAIL case). If the CSP where the time limit is exceeded is unsatisfiable, then a violation is not possible and therefore a nonviolation is predicted. Inversely, if the CSP where the time limit is respected is unsatisfiable, a violation is predicted. If no case is unsatisfiable, no definite prediction can be produced at that time.

The CSP can be created at any point in the execution of a process, and it will capture the conditions imposed by the rest of the execution as determined by process monitoring (e.g., if a service completed slower than planned, less time remains for the rest of the process to complete on time). This makes it possible to perform continuous prediction in several points until the prediction outcome allows us to make a decision. In this paper, for simplicity, we stop the prediction process once the first definite prediction has been obtained.

Our implementation of the constraint-based predictor uses the $ECL^{i}PS^{e}$ constraint logic programming system [44] featuring nonlinear integer and real constraints, and a background MySQL database for storing and analyzing data-dependent ranges of the execution times for the basic services.

Unlike machine learning, constraint satisfaction does not need to be trained and only requires statistical information about the past process executions.

C. QoS Aggregation

QoS aggregation refers to the rule-based reduction of process models to determine end-to-end QoS values. QoS aggregation rules are defined for the control constructs in a process, and include sequence ($T_{\text{seq}} = T_1 + T_2$), parallel execution ($T_{\text{and}} = \max(T_1, T_2)$) and loop ($T_{\text{loop}} = T_1 \cdot k$, with k being the number of loop iterations). QoS aggregation was initially conceived as a design-time technique, but has been extended into a run-time technique for service compositions [45]. During run-time, QoS aggregation rules are evaluated using process monitoring data and planned QoS values for the remaining service executions as input. It should be noted that, despite using similar rules, the constraint satisfaction technique introduced in Section III-B differs from QoS aggregation in so far as constraint satisfaction uses these rules to formulate a system of equations that is then solved during run-time.

Fig. 3 outlines the basic steps performed during QoS aggregation at run-time. The first step is performed after the execution of each individual service, whereas the second step is only performed if the first check indicates a service QoS violation.

To perform the service QoS violation check (first step), the monitored (i.e., effective) QoS values T_i of an executed service s_i are compared with the planned QoS values $\hat{T}_i$ for that service. To this end, the technique checks at runtime whether the service executes slower than planned, i.e., whether $T_i > \hat{T}_i$.

If a service violates its planned QoS, the actual prediction (second step) is initiated. To this end, the technique applies QoS aggregation rules to the process model C, which is augmented with QoS data for all service executions in the model. More specifically, for all services that have been executed up to and including service s_j, which exhibited a QoS violation,

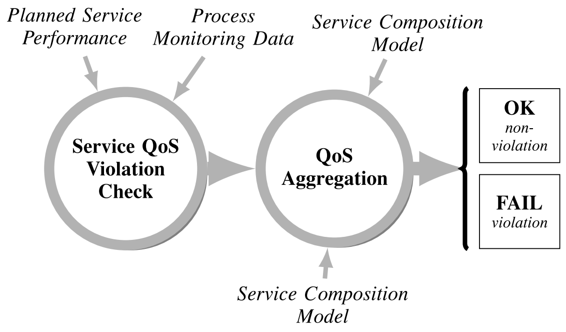
Planned Service Performance
Process Monitoring Data
Service Composition Model
Service QoS Violation Check
QoS Aggregation
OK non-violation
FAIL violation
Service Composition Model

Fig. 3. Main steps of QoS aggregation technique.

actual monitoring data $M = (T_1, \ldots, T_j)$ is used. For all remaining services, the planned QoS values $A' = \hat{T}_{j+1}, \ldots, \hat{T}_n$ are employed as estimates. Hierarchically applying the QoS aggregation rules to this augmented model C, it is computed whether $T(C, M, A') \leq \hat{T}$, i.e., whether the planned process performance, $\hat{T}$, may still be achieved despite the observed constituent QoS violation.

Similarly to constraint satisfaction, the computation of the QoS aggregation model involves purely mechanical steps and thus can be automated. Our implementation of the QoS aggregation technique uses the BOGOR model-checker system to compute expressions specified in the ALBERT specification language [46]. ALBERT allows expressing logical and temporal dependencies of effective and planned service response times along an executed path.

One benefit of QoS aggregation is that it is agnostic to past process executions, because service QoS values are derived from the planned QoS values for each business process. This means that QoS aggregation can be applied immediately once new, modified or extended process models have been deployed (e.g., if additional activities are introduced in the business process).

IV. Case Study Design and Execution

This section introduces the design of our case study, including the research questions we address, the metrics used to assess prediction accuracy, as well as details about the transport and logistics processes and industry data set. It further describes the case study execution and measurement.

A. Research Questions

As motivated in Section I, we aim to analyze the applicability and accuracy of three classes of prediction techniques when applied for predictive monitoring of individual business processes. This led us to define the following two research questions.

RQ1: What is the accuracy that can be achieved when applying machine learning, constraint satisfaction and QoS aggregation for run-time predictions of business process instances?

RQ2: In how far does the accuracy of the individual techniques depend on the point in time during process execution when a prediction is made? The rationale behind this question is that it always takes some time to execute the actions required to respond to violations or mitigate their effects in

TABLE I
CONTINGENCY TABLE

		Predicted	
		Violation	Nonviolation
Actual	Violation	True Positive (TP)	False Negative (FN)
	Nonviolation	False Positive (FP)	True Negative (TN)

the execution of a process. Thus, the longer the lead-time for producing predictions, the smaller the range of opportunities for the proactive management of those violations becomes. In extreme cases, predictive monitoring is not useful if the lead-time exceeds the time needed to react [11]. In business terms, an increase of delays between detection and response to a business event reduces the business value of responding to such events [24], [31].

Further, we aim to investigate into the combination of techniques, leading us to formulate a third research question:

RQ3: Can prediction techniques be combined in order to improve prediction accuracy?

B. Metrics

Informally, accuracy characterizes the ability of a predictive monitoring technique to forecast as many true "problems" as possible, while—at the same time—generating as few false "alarms" as possible [11]. In our case study, we analyze how accurately a technique is able to predict performance violations of business process instances. To this end, we employ well-known metrics from the literature to compute indicators to assess the accuracy of binary (i.e., violation/nonviolation) predictions. Those metrics are derived from the four cases that can result from binary predictions, which are depicted in the contingency table shown in Table I.

Precision $p = TP/(TP+FP)$ indicates how many predicted violations were actual violations. Higher precision means fewer false "alarms." Recall $r = TP/(TP + FN)$ measures how many actual violations were correctly predicted as violations. Higher recall means that more true "problems" are identified. To achieve accurate predictions, a technique should achieve both high precision and recall. However, an intrinsic relationship between precision and recall exists: increasing one of them may decrease the other. For example, always predicting a violation makes $FN = 0$, and therefore recall is one. However, that in turn increases the number of false positivespt (FP) and therefore decreases precision. To combine precision and recall in a single value, literature thus recommends using metrics such as the F-metric [11], which is defined as $F = 2 \cdot p \cdot r/(p + r)$.

The above metrics do not reflect a prediction technique's ability in predicting true negatives [29]. To complement our accuracy measurements, we also include specificity and accuracy. Specificity $s = TN/(TN + FP)$ indicates how many actual nonviolations were correctly predicted as nonviolations. Accuracy $a = (TP + TN)/(TP + TN + FP + FN)$ measures how many predictions (either positive or negative) were correct. However, using accuracy (a) as sole indicator of how well a prediction technique performs may be misleading. Since violations can be expected to occur rather seldom, a technique that always predicts nonviolation could achieve high accuracy and be right in most cases, but it would never flag any violation [11].

C. Transport and Logistics Processes

Fig. 4 shows a model of the business processes covered in the case study as a UML 2.0 activity diagram.[3] The model represents the structure of the business processes of a freight forwarding company, in which up to three smaller shipments from suppliers are consolidated and in turn shipped together to customers in order to benefit from better freight rates or increased cargo security. The business process is structured into incoming and outgoing transport legs, which jointly aim at ensuring that freight is timely delivered to customers. Each transport leg involves the following physical transport services, which are modeled as activities in Fig. 4.

1) *RCS (Freight Reception):* Freight is received by airline. It is delivered and checked in at departure warehouse.
2) *DEP (Freight Departure):* Goods are delivered to aircraft and, once confirmed on board, aircraft departs.
3) *RCF (Freight Arrival):* Freight is transported by air and arrives at destination airport. Upon arrival freight is checked in and stored at arrival warehouse.
4) *DLV (Freight Delivery):* Freight is delivered from destination airport warehouse.

A transport leg may involve multiple flight segments (e.g., in case cargo is transferred to other flights or airlines at stopover airports). In these cases, RCF loops back to DEP. In our case study, the number of segments per leg ranges from one to four.

D. Industry Data Set

The transport services introduced above are denoted by three-letter acronyms following the Cargo 2000 industry standard. Cargo 2000 is an initiative of the International Air Transport Association (IATA). It aims at delivering a new quality management system for the air cargo industry [48]. Cargo 2000 allows for unprecedented transparency in the supply chain. Stakeholders involved in the transport process can share agreed Cargo 2000 messages, comprising transport planning, replanning and service completion events. Cargo 2000 is based on the following key principles.

1) Every shipment gets a plan (called a route map) describing predefined monitoring events.
2) Every service used during shipment is assigned a predefined milestone that defines the planned time for service completion.
3) Stakeholders receive notifications upon successful completion and alerts when a milestone has failed, each including the effective time the milestone has been reached. If an alert is received, stakeholders can update the transport plans by issuing route map updates.

Our case study rests on an industry data set of actual operational data from an international forwarding company. Data has been collected through the company's Cargo 2000 monitoring system and covers a period of five months of business operations. From this Cargo 2000 data, we reconstructed

[3]UML 2.0 activity diagrams provide Petri net semantics. Petri nets are widely used in business process management [47].

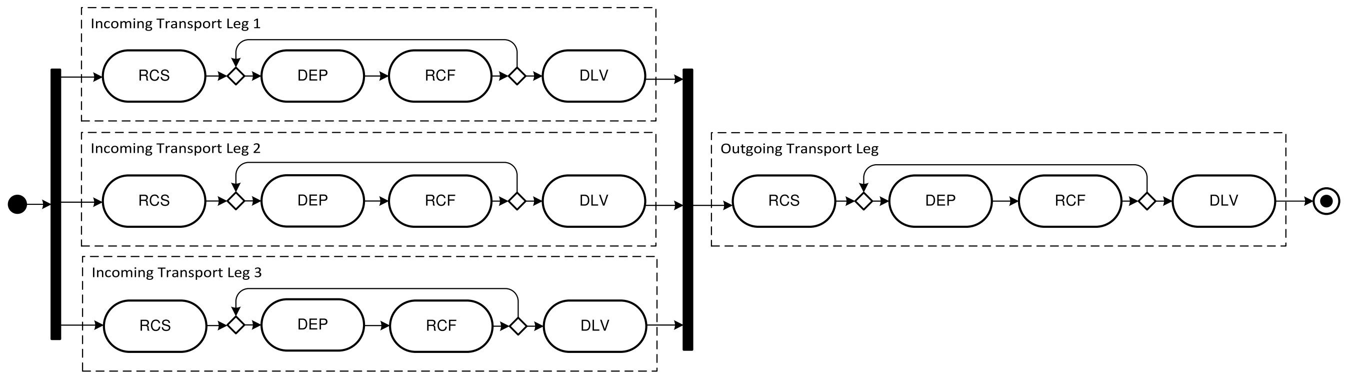

Fig. 4. Structure of transport and logistics process covered by the case study.

TABLE II
ACTUAL PERFORMANCE VIOLATION RATES OF TRANSPORT SERVICES AND PROCESSES OF CASE STUDY

	Instances in Data Set	Rate Actual Violation [%]	Rate Actual Non-Viol. [%]
Transport Service:	**Σ 56,082**		
RCS *(Freight Reception)*	11,874	5.8	94.2
DEP *(Freight Departure)*	16,167	84.0	16.0
RCF *(Freight Arrival)*	16,167	19.5	80.5
DLV *(Freight Delivery)*	11,874	24.0	76.0
End-to-end Process:	**3,942**	**26.6**	**73.4**

execution traces of 3942 actual business process instances, comprising 7932 transport legs and 56 082 service executions. Each execution trace includes the planned and effective duration for each of the activities of the business process. We use this data set for two purposes (see Section IV-E): 1) to predict violations of process performance, i.e., situations in which the time of delivery of the end-to-end process is predicted to be later than the planned time of delivery and 2) to analyze the accuracy of those predictions.

The reconstruction of process execution traces involved data sanitation and anonymization. We filtered overlapping and incomplete Cargo 2000 messages, removed canceled transports (i.e., deleted route maps), cleaned up exceptions from the Cargo 2000 system (such as events occurring before route map creation) and homogenized information representation in different message types. Due to confidentiality reasons, message fields that might exhibit business critical or customer-related data (such as airway bill numbers, flight numbers, and airport codes) were eliminated or masked. Finally, it should be noted that handling of transport documents along the business process differs based on whether the documents are paper-based or electronic. As our data set did not allow us to discern the different document types, we only consider services related to the flow of physical goods.

Table II shows the rates of actual performance violations and nonviolations for the individual transport services as well as for the end-to-end business process.[4] The complete data set, including planned and effective delivery times of the transport services for all the reconstructed business process instances is available online.[5] As can be seen from this data, 26.6% of all process executions have problems with respect to timely delivery and thus provide an opportunity to anticipate and thus proactively manage and mitigate those problems.

E. Execution and Measurement

In order to analyze how accurately our prediction techniques may work in practice (see research question RQ1), we follow the well-known approach of cross-validation and partition our data set into two complementary subsets.[6] The first $2/3$ of the data set are considered historic data, which is used as training data set. How this training data set is employed differs for the three techniques and is thus detailed in Section V. The remaining $1/3$ of the data set are used as testing data set, which means that it is used to validate whether the techniques are in fact able to predict the future.

Using the testing data set, we determine the accuracy of the predictive monitoring techniques as follows. Each of our prediction techniques either predicts violation (true) or nonviolation (false). We compare each prediction that is produced by a technique for a given business process instance, i.e., the predicted violation or nonviolation, with the actual business process performance. Note that we know the actual business process performance at the time of prediction as we work on a recorded data set. If both prediction and actual agree, we note a "true" prediction, if not, we note a "false" prediction. As an example, if we predict a violation but observe an actual nonviolation, we have a "false positive." We count all four contingency cases for all business process instances in the testing data set. From those four cases we then compute the contingency table metrics introduced in (Section IV-B), which are used as indicators for prediction accuracy.

In order to gather insights into the lead-times required to produce accurate predictions (see research question RQ2), we empirically assess the accuracy of the prediction techniques for each of the checkpoints of our business process by following the measurements described above. To visualize the impact of

[4]As can be observed, DEP has a very high rate of violations when compared to the other transport services. This is due to the fact that on-time departure strongly depends on the availability of flights going from an origin to a destination airport. On-time departure, except for environmentally controlled shipments, such as produce and animals, is not as significant from a supply chain perspective as timely arrival.

[5]See the data available at http://www.s-cube-network.eu/c2k

[6]It should be noted that due to the relatively large size of the case study data set (see Section IV-D), we considered one round of cross-validation sufficient to reduce variability of the results.

prediction lead-time on prediction accuracy we employ a normalized scale, which compensates for the fact that the number of service executions varies between the individual process instances. As an example, one transport leg may consist of three segments, while another one may consist of only one segment. To make it possible to compare the results, we chose to present the results as follows. The "positions" provided in the results tables and diagrams in Section V denote logical positions in the business process instances (ranging from 0% to 100%). The and-join (where incoming transports are consolidated) is defined to be at the 50% position.

To assess accuracy improvements when combining the individual techniques (and thus addressing research question (RQ3), we repeat the above measurement process and determine the contingency table metrics for different combinations of techniques (see Section VI).

V. Assessment of Individual Techniques

This section presents empirical results for the three prediction techniques and concludes with a general discussion.

A. Results for Machine Learning

For evaluating the machine learning based prediction technique, we use the implementation of ANNs provided by the WEKA machine learning toolkit in version 3.4.13.[7] WEKA provides a collection of well-tested, popular, and well-known machine learning algorithms, of which we chose the feed-forward multilayer perceptron implementation.

We have established checkpoints for each service execution in the transport and logistics process. More specifically, we have defined one checkpoint after each service invocation, as well as one checkpoint at the beginning of the process, which totals to 21 checkpoints.[8] Based on these checkpoints and the case study design as described in Section IV, we have numerically evaluated the prediction quality of the machine learning approach. To this end, we have trained ANNs for each checkpoint using the training data set, and using tenfold cross-validation. Afterwards, we have used the testing data set (the remaining 1/3 of the process instances) to evaluate the quality of the predictors in the different checkpoints, and calculated the contingency table metrics presented in Section IV-B.

Table III contains a complete list of parameters used to train the ANNs that led to the results presented below. For reasons of brevity, we omit a full discussion of the relevance of each parameter in this paper. More details can be found in the online documentation of WEKA.[9] Note that the concrete network topology depends on the checkpoint that we are training the network for. Each ANN has a number of source (input) nodes equal to the number of available previous service results in this checkpoint, e.g., a checkpoint in which the results of six service executions are already available has six source nodes. Further, each ANN always has exactly one hidden layer of sigmoid nodes. This single hidden layer consists of $\lfloor (sn+1)/2 \rfloor$ nodes, where sn refers to the number of source nodes. Note that all those parameter values are the defaults proposed by WEKA out of the box. Informal parameter optimization, most importantly with regard to the network topology, has not led to significant improvements over those default values. Arguably, this means that results similar to the ones we present here could also be achieved by a user who is not an expert in machine learning or neural network theory.

TABLE III
MULTILAYER PERCEPTRON PARAMETERIZATION

Parameter	Informal Description	Range	Value
Learning Rate	Used to steer how strongly node weights in the ANN can change between each learning iteration; this base learning rate is adapted during training based on the momentum rate	[0;1]	0.3
Momentum Rate	Used to adapt the learning rate during training	[0;1]	0.2
Number of Epochs	Number of training iterations through the entire data set	$\mathbb{N}$	500
Network Topology			
	Number of source nodes	$\mathbb{N}$	Varies (sn)
	Number of hidden layers	$\mathbb{N}$	1
	Number of sigmoid nodes per hidden layer	$\mathbb{N}$	$\lfloor \frac{sn+1}{2} \rfloor$

TABLE IV
CONTINGENCY TABLE RESULTS FOR MACHINE LEARNING

Pos. [%]	TP	FP	TN	FN	TP+TN	FP+FN
0	0.245	0.521	0.170	0.064	0.415	0.585
10	0.250	0.524	0.167	0.059	0.417	0.583
20	0.245	0.528	0.163	0.064	0.408	0.592
30	0.249	0.526	0.165	0.060	0.414	0.586
40	0.246	0.526	0.165	0.063	0.411	0.589
50	0.205	0.339	0.352	0.104	0.557	0.443
60	0.164	0.097	0.594	0.145	0.758	0.242
70	0.161	0.080	0.611	0.148	0.772	0.228
80	0.165	0.072	0.619	0.144	0.784	0.216
90	0.152	0.091	0.600	0.157	0.752	0.248
100	0.256	0.002	0.689	0.053	0.945	0.055

Table IV contains the contingency table results of the evaluation of the machine learning prediction. For each checkpoint, we report the rate of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN).

Analyzing these results, we can come to the observation that prediction based on ANNs is less accurate up to the 50% mark (i.e., in the parallelized part of the process), with about two fifths of predictions being correct (i.e., $TP + TN \approx 2/5$). After the 50% mark, prediction accuracy is visibly improving. However, it is evident that even with almost complete information (i.e., shortly before the 100% mark) some amount of inaccuracy remains using the machine learning approach.

In order to allow for more in-depth interpretation of this data, Fig. 6 visualizes the prediction accuracy in terms of contingency table metrics. From these plots we can observe that there are a small number of points in the execution of the process that impact on prediction accuracy. Firstly, accuracy (in terms of the F-metric) improves visibly after the synchronization point (50% mark) and again after the last RCF service

[7] http://www.cs.waikato.ac.nz/ml/weka/

[8] Each transport leg consists of the services RCS, DEP, RCF and DLV, where DEP and RCF can be repeated up to four times. This means that we reach a total of $2 \cdot (1 + 4 \cdot (1 + 1) + 1) = 20$ checkpoints. Together with the checkpoint before process execution this leads to 21 checkpoints altogether.

[9] http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/MultilayerPerceptron.html

of the outgoing transport leg. In between these points, prediction accuracy does not change significantly. Specificity is very low initially, but improves toward the end of the process execution. This indicates that the machine learning approach is able to indeed predict violations, but has a problem with accurately predicting nonviolations if the lead-time for prediction is longer, thus diminishing prediction accuracy at the beginning of process execution.

In order to avoid impractically short lead-times between the moment at which a violation is predicted and its occurrence, and at the same time maintain accurate predictions, it appears beneficial to place the predictor somewhere around the 60% mark. This would leave more than one third of the time limit for potential corrective actions, while still achieving good accuracy (with an F-metric of about 0.75). In our concrete case, for instance, a broad range of mitigation actions would still be possible, as freight would not yet have been confirmed on board (i.e., process execution would be before the DEP service).

B. Results for Constraint Satisfaction

The results of the constraint satisfaction technique have been determined using the data set as follows. The training data set (i.e., the first $2/3$ of the data) was used to determine the lower and upper bounds of the effective execution time for the different services in the transport process from Fig. 4. The testing data set (i.e., the remaining $1/3$ of the data) has been subjected to the constraint-based continuous prediction. At each prediction point, the approach looks at the remaining services and tries to infer whether a violation of their (local) time limits can be ruled out, therefore leaving the nonviolation as the only possibility, and vice-versa. In that way, the approach is typically able to infer violation or nonviolation of the planned duration for the checkpoints closer to the point of prediction, while remaining undecided for checkpoints that lay farther in the future, for which neither violation nor nonviolation can be ruled out at the moment.

The continuous prediction stops and returns a prediction as soon as it is able to determine violation or nonviolation for the overall process or for some checkpoint before the end but which is close enough to the end. The choice of the position of the earliest service that is used as the predictor for the overall process (i.e., 100% for the final one or a smaller value otherwise) affects the accuracy and the timing of prediction, and is discussed below. If a prediction cannot be done before reaching the end-to-end duration or before the completion of the predicted process instance, we count this as an effective negative prediction.

Table V lists the contingency table results for the constraint satisfaction approach, and Fig. 7 shows the contingency table metrics.

The "%" labels represent the position of the earliest predictor as introduced above. Fig. 5 shows the distribution of the points in time where a prediction of an overall nonviolation (the upper box plot) or a violation (the lower box plot) was made as a percentage of the time when a prediction was made with respect to the total running time of the process instance. Basing the overall prediction strictly on the

TABLE V
CONTINGENCY TABLE RESULTS FOR CONSTRAINT SATISFACTION

Pos. [%]	**TP**	**FP**	**TN**	**FN**	**TP+TN**	**FP+FN**
0	0.122	0.151	0.541	0.186	0.663	0.337
10	0.122	0.151	0.541	0.186	0.663	0.337
20	0.122	0.145	0.547	0.186	0.669	0.331
30	0.120	0.137	0.554	0.189	0.674	0.326
40	0.172	0.124	0.567	0.137	0.739	0.261
50	0.223	0.129	0.562	0.086	0.785	0.215
60	0.210	0.067	0.624	0.099	0.834	0.166
70	0.210	0.063	0.628	0.099	0.838	0.162
80	0.217	0.064	0.627	0.092	0.844	0.156
90	0.253	0.045	0.646	0.056	0.899	0.101
100	0.297	0.010	0.681	0.012	0.978	0.022

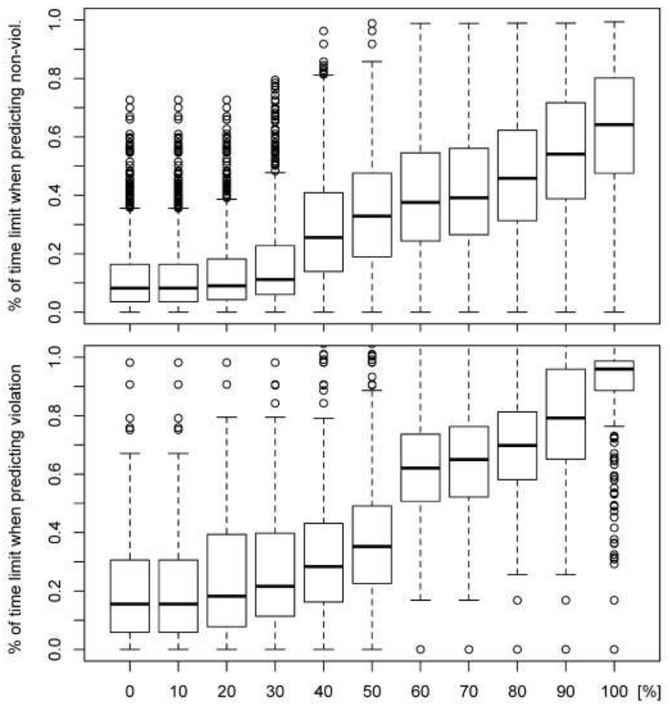

Fig. 5. Prediction timing for constraint satisfaction.

final service (earliest predictor at 100%) gives the highest level of precision, recall, accuracy, specificity, and the F-metric, all very close to 1. However, this may lead to impractically short lead-times between the moment at which a violation is predicted and its occurrence. Fig. 5 shows that in this case the approach is able, on the average, to infer nonviolation after two thirds of the limit, but the prediction of violation comes at about only 2% of the time limit ahead of the violation, which may be too short a time for any kind of corrective action to avoid or mitigate the imminent violation. Therefore, it seems beneficial to choose the position of the earliest predictor somewhere close to, but not as late as, the final service. Fig. 5 suggests that choosing the position of the earliest service somewhere between 60% and 85% of the process dramatically improves the prediction timing, with, on the average, about one third of the time limit available for potential corrective actions. The improvement in timing comes at a price of a decrease in precision, recall, and F-metric (at about 0.72), but with still high specificity (at about 0.90) and accuracy (at about 0.84). In our concrete case, this later position—compared with

TABLE VI
CONTINGENCY TABLE RESULTS FOR QOS AGGREGATION

Pos. [%]	TP	FP	TN	FN	TP+TN	FP+FN
10	0.026	0.030	0.614	0.330	0.640	0.360
20	0.014	0.023	0.635	0.328	0.649	0.351
30	0.017	0.027	0.630	0.326	0.647	0.353
40	0.015	0.018	0.640	0.327	0.655	0.345
50	0.104	0.033	0.620	0.243	0.724	0.276
60	0.012	0.006	0.663	0.319	0.675	0.325
70	0.128	0.046	0.617	0.209	0.745	0.255
80	0.088	0.033	0.630	0.249	0.718	0.282
90	0.135	0.030	0.633	0.202	0.768	0.232
100	0.154	0.003	0.663	0.180	0.817	0.183

the preferred prediction position of machine learning—would reduce the range of mitigation actions, as freight would already have been confirmed on board (at least for the first segment of the outgoing transport leg).

Fig. 7 also shows that a choice of the position of the earliest predictor between 10% and 50% leads to lower prediction accuracy, with the F-metric at around 0.43, specificity at around 0.79, and overall accuracy at around 0.67. This is due to the fact that these early services belong to the incoming parallel transport legs of which only the effectively longest one affects the overall execution time.

C. Results for QoS Aggregation

While machine learning and constraint satisfaction require data about past process instances, QoS aggregation can be employed without such historic data. The technique uses the planned delivery time for each of the services that have not yet been executed as an estimate of service QoS values (also see Section III-C). In order to provide a fair comparison of the techniques, we thus only use the last 1/3 of the data set to perform the QoS aggregation predictions, i.e., we used the same data employed as testing data set for evaluating the other two techniques.

Table VI shows the contingency table results for QoS aggregation, Fig. 8 visualizes the contingency table metrics. Note that, as explained in Section III-B, QoS aggregation only starts prediction once a QoS violation for an individual service has been observed. As a consequence the technique does not deliver predictions for the 0% position. In addition, depending on the actual number of loop iterations, the result for the 100% mark might also include the outcomes for the last-but-one service execution, and thus includes false negatives of predicting the RCF service. As a consequence, recall does not reach 1 at the 100% mark (see Fig. 8).

While precision steadily increases toward the end of process execution, recall experiences a sudden drop at the 60% mark. This correlates with the execution of the DEP service, which—as Table II shows—exhibits a violation rate of 84%. As QoS aggregation is agnostic to historic data and thus cannot learn from past service and process executions, this high violation rate strongly impacts on recall.

For what concerns specificity, QoS aggregation outperforms the other two techniques. Specificity remains within the interval [0.931, 0.996]. The reason is that (as visible from Table II) the duration of the planned services is often chosen too optimistically (e.g., in 24% of the cases for the DLV service). As QoS aggregation exploits those planned durations as estimators for QoS values, this means that QoS aggregation is more pessimistic in predicting violations and thus delivers less false positives (see Table VI).

Due to the "instability" of recall, finding an early yet still accurate moment for prediction is more difficult in the case of QoS aggregation than it is for the other two prediction techniques. It appears beneficial to place the predictor somewhere around the 90% mark, as this would deliver an F-metric of at least 0.54. However, this would leave only little time for potential corrective actions.

D. Findings and Discussion

Considering our research questions defined in Section IV-A, empirical evidence of the case study suggests that all three types of predictive monitoring techniques can be applied to business process instances and in turn lead to good results in prediction accuracy (e.g., research question RQ1). If a prediction is made after 50% of the workflow execution, all three techniques exhibit a prediction accuracy which is greater than 0.7 (see Figs. 6–8), meaning they deliver more true predictions than false predictions (see Section IV-B). Consistently across all the techniques, the ability to predict increases as the point of prediction gets closer to the end of process execution (e.g., research question RQ2).

However, the techniques show visible differences in the increase of accuracy, as well as in terms of false positive predictions and false negative predictions (see Tables IV–VI). These observed differences between techniques are important, as the costs of not taking required actions in case of false negative predictions can differ from the costs of taking unnecessary actions in case of false positive predictions [29].

1) *False Negative Predictions:* In terms of our transport and logistics case study, current situation in industry is that outcomes of individual business processes are not predicted. A possible action in case of negative predictions (nonviolations) could thus be to do nothing. Then, if an actual nonviolation is observed, all is well. If, however, a violation occurs, we are in the same situation as industry is today and need to face the penalties for violating performance promises.
2) *False Positive Predictions:* Reacting to positive predictions (violations) would depend on the cost of proactively addressing them. In case a violation actually happens, cost savings could be determined on a per-case basis (i.e., avoided penalty less cost of addressing the violation; see [18]). However, if no violation happens, i.e., if process execution would have proceeded as planned, any resources spent on addressing this false positive violation would be wasted.

To summarize, whether to proactively address a problem during process execution depends on: 1) the likelihood that the prediction turns out correct, which, as our data suggests, strongly depends on the technique used to predict and the moment in time when the prediction occurs and 2) the expected overall gain, taking into account costs due to a false negative prediction, costs due to a false positive prediction and the savings in case of true positive predictions.

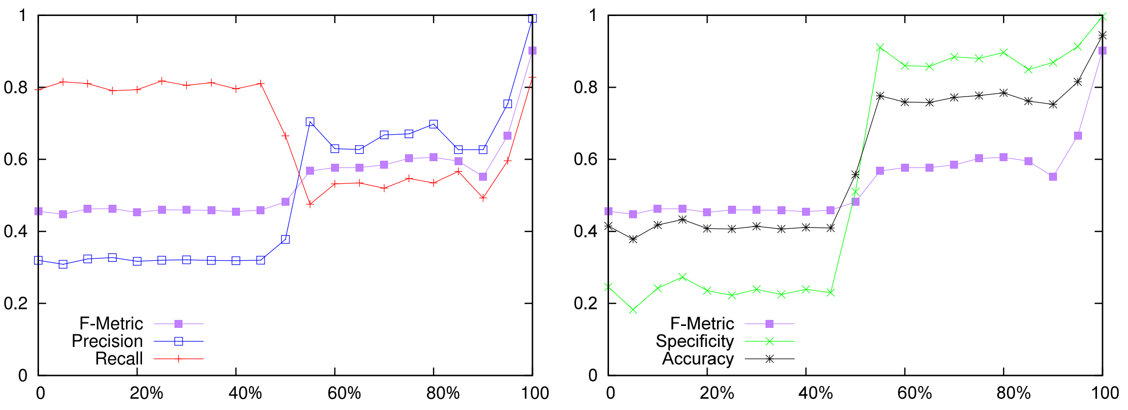

Fig. 6. Accuracy indicators for machine learning.

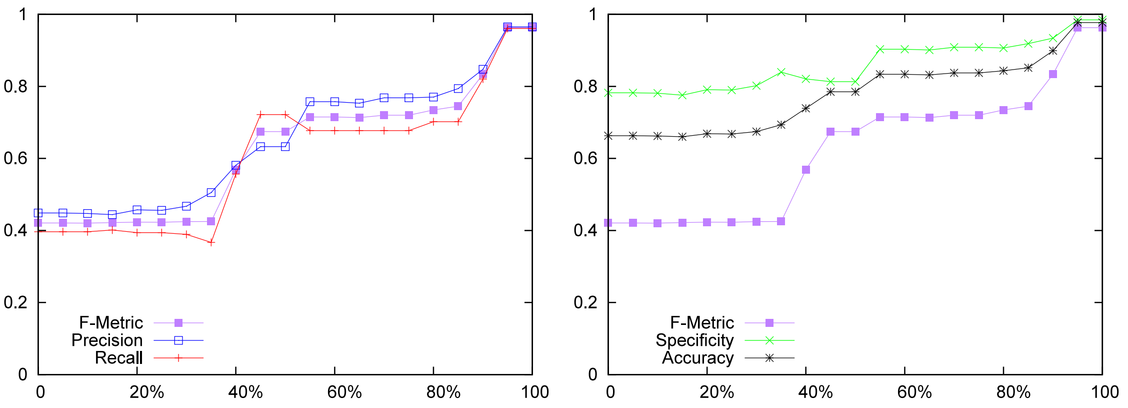

Fig. 7. Accuracy indicators for constraint satisfaction.

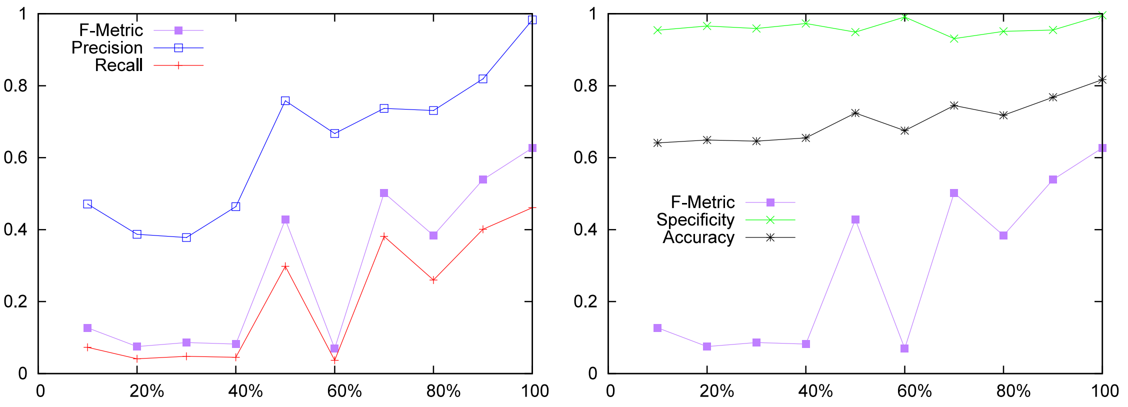

Fig. 8. Accuracy indicators for QoS aggregation.

VI. Assessment of Combined Techniques

As the results in Section V indicate, prediction techniques differ in terms of how accurately they may predict violations and nonviolations. In this section, we thus exploit the characteristics of the individual techniques and propose their combination with the aim of improving the prediction process.

In the area of machine learning, a combination of different prediction models is used to increase prediction accuracy. Known as ensemble learning, different strategies and techniques for combining prediction models exist [49]. We draw inspiration from ensemble learning and aim to analyze whether combinations of the individual techniques may lead to accuracy improvements compared to when using the prediction techniques separately (thereby addressing research question RQ3). To this end, we assess different voting strategies for combining the predictions of the individual techniques.

In order to define the combined predictions, let $q_{\mathrm{ML}}, q_{\mathrm{CS}}, q_{\mathrm{QA}} \in \{\text{true}, \text{false}\}$ be the prediction that is delivered by machine learning (ML), constraint satisfaction (CS), and QoS aggregation (QA) for a given logical position in the business process instance, where true indicates a predicted violation and false indicates a predicted nonviolation.

TABLE VII
COMPARISON OF ACCURACY INDICATORS (ABSOLUTE VALUES IN BOLD, DIFFERENCES IN NORMAL FONT; "+" INDICATES IMPROVEMENT)

	Individual Technique	**Majority**	**Recall-oriented**	**Prec.-/Spec.-oriented**
Precision		**0.683**	**0.448**	**0.776**
ML	**0.429**	+0.254	+0.019	+0.347
CS	**0.675**	+0.009	−0.227	+0.101
QA	**0.683**	0.000	−0.235	+0.093
Recall		**0.547**	**0.830**	**0.250**
ML	**0.678**	−0.131	+0.152	−0.427
CS	**0.630**	−0.083	+0.200	−0.379
QA	**0.272**	+0.275	+0.558	−0.021
F-Metric		**0.607**	**0.582**	**0.378**
ML	**0.526**	+0.082	+0.056	−0.147
CS	**0.651**	−0.044	−0.070	−0.273
QA	**0.389**	+0.219	+0.193	−0.010
Specificity		**0.887**	**0.543**	**0.968**
ML	**0.597**	+0.290	−0.055	+0.370
CS	**0.864**	+0.022	−0.322	+0.103
QA	**0.944**	−0.057	−0.401	+0.024
Accuracy		**0.782**	**0.631**	**0.746**
ML	**0.622**	+0.160	+0.009	+0.124
CS	**0.792**	−0.010	−0.160	−0.046
QA	**0.736**	+0.046	−0.105	+0.010

A. Majority

The first voting strategy we evaluate is well-known from ensemble learning. We use majority voting as we expect it to contribute to an overall improvement in terms of the F-metric and accuracy metric. It predicts a violation if at least two out of the three techniques predict a violation: $q_{\text{majority}} = (q_{\text{ML}} \wedge q_{\text{CS}}) \vee (q_{\text{CS}} \wedge q_{\text{QA}}) \vee (q_{\text{ML}} \wedge q_{\text{QA}})$.

As discussed above, the costs of not taking required actions in case of false negative predictions can differ from the costs of taking unnecessary actions in case of false positive predictions. This means that depending on the actual setting faced in practice, it may be beneficial to improve the accuracy in predicting either positives or negatives. This observation leads us to define three specific voting strategies.

B. Recall-Oriented

We aim to define a voting strategy that may increase recall by combining the predictions of the two techniques which deliver the best recall amongst our three techniques. For our case study (see Figs. 6–8) this means that this strategy combines machine learning and constraint satisfaction. To increase recall, we need to predict as many true positives as possible. To this end, we define the voting strategy such that it predicts a violation if at least one of the two individual techniques predicts a violation, i.e., we are more "optimistic" in selecting violations and thus may catch more true violations: $q_{\text{recall-oriented}} = q_{\text{ML}} \vee q_{\text{CS}}$.

C. Precision-Oriented

This strategy aims to increase precision by combining the two techniques with best precision. In our case study these are constraint satisfaction and QoS aggregation. However, to increase precision, we need to make sure that we avoid producing false positives. We thus define the voting strategy such that it predicts a violation only if both techniques predict a violation, i.e., we are more "skeptical" in predicting violations and thus may deliver less false violations: $q_{\text{precision-oriented}} = q_{\text{CS}} \wedge q_{\text{QA}}$.

D. Specificity-Oriented

This strategy aims to increase specificity by combining the two techniques which delivered the best specificity, viz. constraint satisfaction and QoS aggregation. To increase specificity, we need to make sure that we predict as many true negatives as possible. The voting strategy thus predicts a nonviolation if at least one of the two individual techniques predicts a nonviolation; i.e., similar to the recall-oriented strategy, we are "optimistic" in selecting nonviolations and thus may catch more true nonviolations. In our case study, this strategy is identical to the precision-oriented one, as the same prediction techniques are chosen: $q_{\text{specificity-oriented}} = \neg(\neg q_{\text{CS}} \vee \neg q_{\text{QA}}) = q_{\text{CS}} \wedge q_{\text{QA}}$.

The accuracy indicators for the combined techniques are shown in Table VII. They are compared with the indicators of the three individual techniques, where differences larger than zero indicate improvements over the individual techniques. All indicators are computed based on totaling the predictions for all logical position in the business process instances.

Interestingly, the majority voting strategy does not contribute to a visible improvement in prediction accuracy. For each accuracy indicator, there is at least one technique that provides the same or better results than majority voting. Specifically, the constraint satisfaction technique consistently provides better results in terms of F and accuracy metrics.

In contrast, the voting strategies oriented toward specific contingency table metrics lead to more promising results. For what concerns the recall-oriented strategy, an improvement of at least 0.152 (i.e., an improvement by 23%) can be observed when compared to the recall of the individual techniques. The precision-/specificity-oriented strategy delivers a precision improvement of at least 0.093 (14%) and a specificity improvement of at least 0.024 (3%). As one would expect, though, the increase along one metric is paid for with a decrease along other metrics. In particular, the F-metric, which combines precision and recall, is not improved by any of the combined strategies over the best of the scores given by the techniques taken in isolation.

Based on this initial evidence, we can conclude that it appears feasible to combine prediction techniques in such a way that their accuracy can be targeted toward specific contingency table metrics. Referring to our observations in Section V-D, this means that we may choose one combination if we want to reduce the risk of taking unnecessary actions by decreasing the rate of false positive predictions, and a different strategy if we want to reduce the risk of not taking required actions in case of false negative predictions.

VII. CONCLUSION

This paper provided a comparison of three main types of predictive monitoring techniques, namely machine learning, constraint satisfaction, and QoS aggregation. These techniques may be used to complement existing business process management solutions (such as business activity monitoring, business analytics, and business intelligence) with capabilities for predictive monitoring of individual business process instances.

Empirical evidence gathered from an industrial case study in the domain of transport and logistics suggests that, given a lead-time of half of the process duration, all three prediction techniques consistently provide an accuracy of at least 70%. However, our comparison of the techniques unveiled that they exhibit important differences concerning the rate of positive and negative predictions they deliver. Based on these findings, we combined predictive monitoring techniques in such a way as to reduce the risk of taking unnecessary actions in case of false positive predictions, or the risk of not taking required actions in case of false negative predictions. Our empirical findings indicate that combinations of techniques may indeed outperform individual techniques. Specifically, combining constraint satisfaction with QoS aggregation improves precision by 14%; combining machine learning with constraint satisfaction improves recall by 23%. One further direction for improving predictions is offered by probabilistic prediction techniques. Such probabilistic prediction techniques may provide, for instance, confidence levels for predictions and thereby foster making statistically informed decisions on whether to respond to a predicted violation or not.

We mainly focused on predicting violations of process performance, specifically cases when planned freight delivery times might not be met. However, the predictive monitoring techniques we employed are not limited to process performance. They can similarly be applied to other, numeric and monotonic quality attributes, as well as for predicting multiple quality attributes. As an example, in the case of fresh produce, the time of delivery could be predicted together with the quality of the delivered product.

Our industry case study in transport and logistics demonstrated the benefits and applicability of three major types of prediction techniques for a very relevant application domain. To foster potential generalization of our findings to other application domains, we took two major design decisions for the case study. First, we selected nonspecialized prediction techniques, thereby avoiding special-purpose or ad-hoc techniques for the sake of generalizability of the results. Second, all three techniques have already been successfully employed in the domain of service-oriented software systems (see [19], [20], [40]). This means that, together with the results provided in this paper, those techniques have shown their success and applicability in two quite distinct application domains (logistics processes and software services). This gives us reason to believe that these techniques may also be applicable outside of those two application domains. Of course, to fully understand how reasonable such a generalization may be, further empirical evidence is required. To this end, we provide all relevant parameters and data[10] used in this paper to enable fellow researchers to replicate and extend our experiments and analyses.

Acknowledgment

The authors would like thank the anonymous reviewers for their valuable comments that have greatly benefited this paper and they would also like to thank A. Beulens, Y. Engel, F. Fournier, G. Gröner, C. C. Marquezan, N. Gol Mohammadi, Ø. Olsen, M. Weber, and B. Wetzstein for helpful pointers as well as valuable comments on earlier drafts of the paper.

[10]http://www.s-cube-network.eu/c2k

References

[1] J. Erbes, H. R. M. Nezhad, and S. Graupner, "The future of enterprise IT in the cloud," *IEEE Comput.*, vol. 45, no. 5, pp. 66–72, May 2012.

[2] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] E. Di Nitto, C. Ghezzi, A. Metzger, M. P. Papazoglou, and K. Pohl, "A journey to highly dynamic, self-adaptive service-based applications," *Autom. Softw. Eng.*, vol. 15, nos. 3–4, pp. 313–341, 2008.

[4] D. Folinas, D. Bochtis, and C. Sørensen, "In-field logistics processes management based on business activities monitoring systems paradigm," *Int. J. Logist. Syst. Manage.*, vol. 8, no. 1, pp. 1–18, 2011.

[5] Y. Li, M. R. Kramer, A. J. M. Beulens, and J. G. A. J. van der Vorst, "A framework for early warning and proactive control systems in food supply chain networks," *Comput. Ind.*, vol. 61, no. 9, pp. 852–862, 2010.

[6] A. Estruch and J. A. H. Álvaro, "Event-driven manufacturing process management approach," in *Proc. 10th Int. Conf. Bus. Process Manage. (BPM)* (Lecture Notes in Computer Science), vol. 7481. Tallinn, Estonia, 2012, pp. 120–133.

[7] M. D. Ilic, L. Xie, U. A. Khan, and J. M. F. Moura, "Modeling of future cyber-physical energy systems for distributed sensing and control," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 4, pp. 825–838, Jul. 2010.

[8] A. Metzger, R. Franklin, and Y. Engel, "Predictive monitoring of heterogeneous service-oriented business networks: The transport and logistics case," in *Proc. Serv. Res. Innov. Inst. Global Conf. (SRII)*, San Jose, CA, USA, 2012, pp. 313–322.

[9] N. S. Caswell *et al.*, "Estimating value in service systems: A case study of a repair service system," *IBM Syst. J.*, vol. 47, no. 1, pp. 87–100, 2008.

[10] H. Jalonen and A. Lönnqvist, "Predictive business—Fresh initiative or old wine in a new bottle," *Manage. Decis.*, vol. 47, no. 10, pp. 1595–1609, 2009.

[11] F. Salfner, M. Lenk, and M. Malek, "A survey of online failure prediction methods," *ACM Comput. Surv.*, vol. 42, no. 3, pp. 10.1–10.42, 2010.

[12] W. Schmidt, "Business activity monitoring (BAM)," in *Business Intelligence and Performance Management* (Advanced Information and Knowledge Processing), P. Rausch, A. F. Sheta, and A. Ayesh, Eds. London, U.K.: Springer, 2013, pp. 229–242.

[13] M. Mühlen and R. Shapiro, "Business process analytics," in *Handbook on Business Process Management 2* (International Handbooks on Information Systems), J. vom Brocke and M. Rosemann, Eds. Berlin, Germany: Springer, 2010, pp. 137–157.

[14] M. Castellanos, N. Salazar, F. Casati, U. Dayal, and M.-C. Shan, "Predictive business operations management," *Int. J. Comput. Sci. Eng.*, vol. 2, nos. 5–6, pp. 292–301, Aug. 2006.

[15] B. Kang, D. Kim, and S. Kang, "Periodic performance prediction for real-time business process monitoring," *Ind. Manage. Data Syst.*, vol. 112, no. 1, pp. 4–23, 2011.

[16] W. M. P. van der Aalst, M. H. Schonenberg, and M. Song, "Time prediction based on process mining," *Inf. Syst.*, vol. 36, no. 2, pp. 450–475, 2011.

[17] Z. Feldmann, F. Fournier, R. Franklin, and A. Metzger, "Proactive event processing in action: A case study on the proactive management of transport processes," in *Proc. 7th ACM Int. Conf. Distrib. Event-Based Syst.*, Arlington, TX, USA, 2013, pp. 97–106.

[18] P. Leitner, W. Hummer, and S. Dustdar, "Cost-based optimization of service compositions," *IEEE Trans. Serv. Comput.*, vol. 6, no. 2, pp. 239–251, Apr./Jun. 2013.

[19] D. Ivanović, M. Carro, and M. Hermenegildo, "Constraint-based runtime prediction of SLA violations in service orchestrations," in *Proc. 9th Int. Conf. Serv.-Oriented Comput. (ICSOC)* (Lecture Notes in Computer Science), vol. 7084, Paphos, Cyprus, 2011, pp. 62–76.

[20] E. Schmieders and A. Metzger, "Preventing performance violations of service compositions using assumption-based run-time verification," in *Proc. ServiceWave Conf.* (Lecture Notes in Computer Science), vol. 6994. Poznan, Poland, 2011, pp. 194–205.

[21] K. Chandy and W. Schulte, *Event Processing: Designing IT Systems for Agile Companies*. New York, NY, USA: McGraw-Hill, 2009.

[22] C. Janiesch, M. Matzner, and O. Müller, "A blueprint for event-driven business activity management," in *Proc. 9th Int. Conf. Bus. Process Manage. (BPM)* (Lecture Notes in Computer Science), Clermont-Ferrand, France, vol. 6896. 2011, pp. 17–28.

[23] C. Costello and O. Molloy, "Building a process performance model for business activity monitoring," in *Information Systems Development: Challenges in Practice, Theory, and Education*, W. Wojtkowski, G. Wojtkowski, M. Lang, K. Conboy, and C. Barry, Eds. New York, NY, USA: Springer, 2009, pp. 237–248.

[24] R. Hackathorn, "The BI watch: Real-time to real-value," *DM Rev.*, vol. 14, no. 1, Jan. 2004 [Online]. Available: http://www.information-management.com/issues/20040101/7913-1.html

[25] S. Chaudhuri, U. Dayal, and V. Narasayya, "An overview of business intelligence technology," *Commun. ACM*, vol. 54, no. 8, pp. 88–98, Aug. 2011.

[26] D. Grigori *et al.*, "Business process intelligence," *Comput. Ind.*, vol. 53, no. 3, pp. 321–343, 2004.

[27] A. Bevacqua, M. Carnuccio, F. Folino, M. Guarascio, and L. Pontieri, "A data-adaptive trace abstraction approach to the prediction of business process performances," in *Proc. 15th Int. Conf. Enterprise Inf. Syst. (ICEIS)*, Angers, France, 2013, pp. 56–65.

[28] A. Rogge-Solti and M. Weske, "Prediction of remaining service execution time using stochastic Petri nets with arbitrary firing delays," in *Proc. 11th Int. Conf. Serv.-Oriented Comput. (ICSOC)* (Lecture Notes in Computer Science), vol. 8274. Berlin, Germany, 2013, pp. 389–403.

[29] A. Metzger, O. Sammodi, and K. Pohl, "Accurate proactive adaptation of service-oriented systems," in *Assurances for Self-Adaptive Systems*, J. Camara, R. de Lemos, C. Ghezzi, and A. Lopes, Eds. Berlin, Germany: Springer, 2012, pp. 240–265.

[30] O. Etzion and P. Niblett, *Event Processing in Action*. Greenwich, CT, USA: Manning, 2010.

[31] B. Schwegmann, M. Matzner, and C. Janiesch, "A method and tool for predictive event-driven process analytics," in *Proc. 11th Int. Conf. Wirtschaftsinformatik (WI)*, Leipzig, Germany, 2013, pp. 721–736.

[32] D. Cfarku, Y. Taher, R. Haque, W.-J. Heuvel, and M. P. Papazoglou, "PAEAN: A risk-mitigation framework for business transaction at runtime," in *Proc. 13th Int. Conf. E-Commerce Web Technol. (EC-Web)*, Vienna, Austria, vol. 123. 2012, pp. 25–37.

[33] I. Jolliffe, *Principal Component Analysis*. New York, NY, USA: Wiley, 2005.

[34] S. Haykin, *Neural Networks and Learning Machines: A Comprehensive Foundation*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2008.

[35] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *IEEE Comput.*, vol. 29, no. 3, pp. 31–44, Mar. 1996.

[36] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA, USA: Morgan-Kaufmann, 1993.

[37] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, Sep. 1995.

[38] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Mach. Learn.*, vol. 29, pp. 131–163, Nov. 1997.

[39] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. New York, NY, USA: Wiley, 1990.

[40] P. Leitner, W. Hummer, and S. Dustdar, "Data-driven and automated prediction of service level agreement violations in service compositions," *Distrib. Parallel Databases*, vol. 31, no. 3, pp. 447–470, 2013.

[41] N. Forbes, *Imitation of Life—How Biology is Inspiring Computing*. Cambridge, MA, USA: MIT Press, 2004.

[42] P. Domingos, "A few useful things to know about machine learning," *Commun. ACM*, vol. 55, no. 10, pp. 78–87, 2012.

[43] R. Dechter, *Constraint Processing*. San Francisco, CA, USA: Morgan Kauffman, 2003.

[44] K. R. Apt and M. G. Wallace, *Constraint Logic Programming Using ECLiPSe*. Cambridge, MA, USA: Cambridge Univ. Press, 2007.

[45] G. Canfora, M. D. Penta, R. Esposito, and M. L. Villani, "A framework for QoS-aware binding and re-binding of composite web services," *J. Syst. Softw.*, vol. 81, no. 10, pp. 1754–1769, 2008.

[46] D. Bianculli, C. Ghezzi, P. Spoletini, L. Baresi, and S. Guinea, "A guided tour through SAVVY-WS: A methodology for specifying and validating web service compositions," in *Advances in Software Engineering* (Lecture Notes in Computer Science), vol. 5316, E. Borger and A. Cisternino, Eds. Lipari Island, Italy, 2008.

[47] C. A. L. Oliveira, R. M. F. Lima, H. A. Reijers, and J. T. S. Ribeiro, "Quantitative analysis of resource-constrained business processes," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 42, no. 3, pp. 669–684, May 2012.

[48] R. Cesana. (2004). *Cargo 2000 Phase 3 Specification*. IATA/Cargo 2000 [Online]. Available: www.cargo2000.com

[49] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–45, Sep. 2006.

Andreas Metzger received the Diploma and the Ph.D. degrees in computer science from the Technical University of Kaiserslautern, Kaiserslautern, Germany, in 1998 and 2004, respectively.

He is currently a Senior Academic Councilor and the Head of the Adaptive Systems and Future Internet Applications at paluno, the Ruhr Institute for Software Technology at the University of Duisburg-Essen, Essen, Germany. He was a Research Assistant at the Technical University of Kaiserslautern, where he is involved in the model-driven software development. His current research interests include monitoring and adaptation of service-based applications, cloud systems, as well as operational business processes. He has co-authored over 60 papers, articles, and book chapters, including *Automated Software Engineering*, *Communications of the ACM*, and the International Conference on Software Engineering.

Dr. Metzger was a member of the Steering Committee and the Management Board of the European Network of Excellence on Software, Services, and Systems, and is currently a Chief Technical Architect of the European Future Internet Public Private Partnership Project on transport, logistics, and agri-food. He was a Program Committee Member for numerous international conferences and workshops, including the European Software Engineering Conference and the International Software Product Line Conference, and has co-organized 15 international workshops and conference tracks. He is a member of the IFIP Working Group Services-Oriented Systems and the European Future Internet Public Private Partnerships Architecture Board.

Philipp Leitner received the B.Sc., M.Sc., and the Ph.D. degrees, all in business informatics from the Vienna University of Technology, Vienna, Austria, in 2005, 2007, and 2011, respectively.

After a short research stay in Singapore, he is currently a Senior Research Associate with the University of Zurich, Zurich, Switzerland. He is currently a Group Leader with the European COST Action on Autonomous Control for a Reliable Internet of Services, and a Senior Researcher with the FP7-funded Integrated Project CloudWave. His current research interests include software engineering for service-oriented and cloud-based systems, methods, middleware, and tools that support developers building better and more cost-efficient distributed systems. He has co-authored over 60 research papers, which have appeared (among other venues) in the IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE INTERNET COMPUTING, Springer's Distributed and Parallel Databases, the International Conference on Service-Oriented Computing, and the International Conference on Distributed Event-Based Systems.

Dragan Ivanović received the B.Sc. and M.Sc. degrees in computer science and electrical engineering from the University of Sarajevo, Bosnia and Hercegovina, and the Ph.D. degree in computer science from the Technical University of Madrid, Madrid, Spain, where his main focus was on the use of logic and constraint modeling and programming, as well as the corresponding program analysis methods, to study the properties of complex and adaptive service-oriented computing systems.

He is currently a Post-Doctoral Student with the Madrid Institute for Advanced Studies (IMDEA) Software Institute, Madrid. His current research interests include using computational logic and constraint programming techniques to model and analyze the properties of complex adaptive software systems, such as service compositions provided via cloud, dynamic modeling of cloud provision systems, and the study of probabilistic behavior of service compositions, in terms of their performance and other nonfunctional properties.

Dr. Ivanović was the recipient of the Best Paper Award at the International Conference on Service Oriented Computing (ICSOC 2011). He has taken part in the European Union Network of Excellence on Software, Services, and Systems as well as in several other national and regional projects.

Eric Schmieders received the master's degree in communication and computer science from the University of Duisburg-Essen, Essen, Germany, in 2007. He is currently pursuing the Ph.D. degree from paluno, the Ruhr Institute for Software Technology at the University of Duisburg-Essen, Essen, Germany.

He was a Software Architect and an Auditor in the automotive industry, and is currently a Research Assistant with paluno. His current research interests include runtime verification of response times and privacy compliances of services with special focus on updating and checking runtime models.

Mr. Schmieders has published at several workshops and conferences such as ServiceWave, and co-organized workshops such as Evolution and Maintenance of Long-Living Systems. He was involved in national and international research projects, including the European Network of Excellence on Software, Services, and Systems and is currently researcher in the iObserve project funded by the German Research Foundation (DFG) under the Priority Programme SPP1593.

Rod Franklin received the B.S. (Distinction) and M.S. degrees, both in mechanical engineering from Purdue University, West Lafayette, IN, USA, and from Stanford University, Palo Alto, CA, USA, in 1974 and 1975, respectively, the M.B.A. degree from the Harvard Graduate School of Business, Boston, MA, USA, in 1975, and the Ph.D. degree in management from Case Western Reserve University, Cleveland, OH, USA, in 2001.

He is currently an Adjunct Professor with the Department of Logistics and an Academic Director of Executive Education with the Kühne Logistics University, Hamburg, Germany. He has held management positions at Cameron Iron Works, Houston, TX, USA, Digital Equipment Corporation, Maynard, MA, USA, Entex Information Services, New York, NY, USA, Viacore, Irvine, CA, USA, Usco Logistics and Kuehne + Nagel, Schindellegi, Switzerland. Apart from this, he was a Consultant with Theodore Barry & Associates, Los Angeles, CA, USA, Booz-Allen and Hamilton, McLean, VA, USA, and Arthur Young & Company, Chicago, IL, USA. His current research interests include data analytics, decision science, mathematical modeling, and sustainable logistics.

Dr. Franklin is the Vice Chairman of the Alliance for Logistics Innovation through Collaboration in Europe, a European Technology Platform, and an Advisory Board Member of numerous European Union research projects including Moduluschca, Optet, Cirrus, and Cassandra.

Manuel Carro received the B.S. and the Ph.D. degrees in computer science from the Technical University of Madrid (UPM), Madrid, Spain, in 1994 and 2003, respectively.

He is currently an Associate Research Professor and a Deputy Director with the IMDEA Software Institute, Madrid, and an Associate Professor at UPM. He has previously been a Representative of UPM at Networked European Software and Services Initiative and the Spanish Initiative for Software and Services technology platforms, and is currently a representative of UPM at the Spanish Research Consortium for Informatics and Mathematics and a Deputy Representative of IMDEA Software at the European Research Consortium for Informatics and Mathematics and Informatics Europe. His current research interests include the design and implementation of high-level (logic- and constraint-based) programming languages to improve the quality of software, the analysis of service-based systems, and the effective usage of formal specifications in the process of teaching programming, parallel programming, and parallel implementations of declarative languages, and visualization of program execution. He was a Principal Investigator for the S-Cube European Network of Excellence of UPM, and is currently a Principal Investigator of two Spanish projects and one European project. He has published over 70 papers in international conferences and journals.

Dr. Carro was the recipient of the best conference paper award for some of the international conferences and journals. He has been a Conference Chair and a PC member for several international conferences and workshops and has participated in research projects at the regional, national, and European level.

Schahram Dustdar received the M.Sc. and Ph.D. degrees in business informatics (Wirtschaftsinformatik) from the University of Linz, Linz, Austria, in 1990 and 1992, respectively, and the Habilitation degree (Venia docendi) in process-aware collaboration systems—architectures and coordination models for Virtual Teams, in 2003.

He is a Full Professor of Computer Science (Informatics) with a focus on internet technologies and is the Head of the Distributed Systems Group at the Vienna University of Technology, Vienna, Austria. From 2004 to 2010, he was an Honorary Professor of Information Systems at the Department of Computing Science, the University of Groningen, Groningen, The Netherlands. His current research interests include internet technologies in particular service-oriented architectures and computing, mobile and ubiquitous computing, complex, autonomic, and adaptive systems, and context-aware computing including all aspects related to collaborative systems such as workflow technologies.

Prof. Dustdar was the recipient of the ACM Distinguished Scientist Award in 2009, and the IBM Faculty Award in 2012. He is an Associate Editor of the IEEE TRANSACTIONS ON SERVICES COMPUTING, the *ACM Transactions on the Web*, and the *ACM Transactions on Internet Technology*, and is an Editorial Board Member of the IEEE INTERNET COMPUTING. He is an Editor-in-Chief of *Computing* (an SCI-ranked journal of Springer). He is a member of the Academia Europaea: The Academy of Europe, Informatics Section since 2013.

Klaus Pohl received the Diploma degrees in computer science and information science from the Karlsruhe University of Applied Sciences, Karlsruhe, Germany, and the University of Konstanz, Konstanz, Germany, in 1988 and 1989, respectively, the Ph.D. (Dr. rer. nat.) and the Habilitation degrees in computer science from the RWTH Aachen University, Aachen, Germany, in 1995 and 2000, respectively.

He is a Full Professor of Software Systems Engineering at the University of Duisburg-Essen, Essen, Germany. From 2005 to 2007, he was a Founding Director of the Irish Software Engineering Research Centre (Lero), Ireland. He is a coordinator of several research projects, including the FP7 network of Excellence on Software, Services, and Systems Network. He has (co-)authored over 200 peer-reviewed publications and several text books. As a Consultant, Assessor, and an Expert, he supports reputable companies, research institutes, and programs.

Prof. Pohl is a Program Chair and a General Chair of over ten international conferences, including the 35th International Conference on Software Engineering in 2013, the 9th and 12th International Software Product Line Conferences in 2005/2008, the 18th International Conference on Advanced Information Systems Engineering in 2006, and the International Requirements Engineering Conference in 2002. He is a member of the Steering Committee of the European Technology Platform Networked European Software and Services Initiative and the German Innovation Alliance Software Platform for Embedded Systems.