Proactive Learning from SLA Violation in Cloud Service based Application

Ameni Meskini[1], Yehia Taher[2], Amal El gammal[3], Béatrice Finance[2] and Yahya Slimani[1]
[1]*University of Carthage, INSAT, LISI Research Laboratory, Tunis, Tunisia*
[2]*Laboratoire PRiSM, Universite de Versailles/Saint-Quentin-en-Yvelines, Versailles, France*
[3]*Faculty of Computers and Information, Cairo University, Cairo, Egypt*

Keywords: Cloud Service based Application, SLA Violations Prevention, Cloud Environments, Decision Tree.

Abstract: In recent years, business process management and Service-based applications have been an active area of research from both the academic and industrial communities. The emergence of revolutionary ICT technologies such as Internet-of-Things (IoT) and cloud computing has led to a paradigm shift that opens new opportunities for consumers, businesses, cities and governments; however, this significantly increases the complexity of such systems and in particular the engineering of Cloud Service-Based Application (CSBA). A crucial dimension in industrial practice is the non-functional service aspects, which are related to Quality-of-Service (QoS) aspects. Service Level Agreements (SLAs) define quantitative QoS objectivesandis a part of a contract between the service provider and the service consumer. Although significant work exists on how SLA may be specified, monitored and enforced, few efforts have considered the problem of SLA monitoring in the context of Cloud Service-Based Application (CSBA), which caters for tailoring of services using a mixture of Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service (IaaS) solutions. With a preventive focus, the main contribution of this paper is a novel learning and prediction approach for SLA violations, which generates models that are capable of proactively predicting upcoming SLAs violations, and suggesting recovery actions to react to such SLA violations before their occurrence. A prototype has been developed as a Proof-Of-Concept (POC) to ascertain the feasibility and applicability of the proposed approach.

1 INTRODUCTION

In CSBA (Cloud Service Based Application), a client can rent a cloud service or a set of cloud services from a single or multiple providers to create his/her application. The provisioning of these services relies on a Service Level Agreement (SLA) (Peter et al., 2011). In cloud computing, (Brandic et al., 2009) during the application execution, various events are produced by several layers (i.e., Cloud and SOA specific), leading to potential Service Level Objective (SLO) violations.This crucial dimension in industrial practice concerns the non-functional service aspects, which are related to Quality-of-Service (QoS). Service Level Agreements (SLAs) (Boniface et al., 2007) define quantitative QoS objectives, which represents a service contract between the service provider and the service consumer. The service provider promises to deliver the requested service complying with some measurable QoS objectives/constraints. Typically, a SLA comprises of a set of Service Level Objectives (SLOs), each of which represents a quality constraint on the system. SLAs usually define penalties, in monetary terms that the provider has to grant to the customer if a QoS is violated. Furthermore, service-based business processes have to comply with an ever-growing number of laws, regulations and standards, such as Sarbanes-Oxley, Basel-III and ISO 9001. Maintaining the promised QoS level (George et al., 2003) further increases the complexity of such systems. The high dynamism of such systems, and the unprecedented complexity that arises from the mass of information that is associated with runtime, puts an emphasis on their *adaptive capabilities*. In practice, the rapid evolving nature of the business and the compliance domains requires these systems to be equipped with self-adaptive capabilities to ensure the proper execution of Cloud Services-Based Applications. These systems have to *autonomously adapt* themselves to changes on service provisioning, availability of things and content, computing

Meskini, A., Taher, Y., gammal, A., Finance, B. and Slimani, Y.
Proactive Learning from SLA Violation in Cloud Service based Application.
In *Proceedings of the 6th International Conference on Cloud Computing and Services Science (CLOSER 2016) - Volume 1*, pages 186-193
ISBN: 978-989-758-182-3

resources, and network connectivity, while continually meeting QoS and regulatory constraints. Current solutions for adaptive Web services and adaptive service-based business processes fall short to support essential characteristics of such applications. The highly rapid and unforeseen adaptive nature of these applications with their complex and distributed nature may indirectly affect various layers and components of the cloud stack (i.e., SaaS, PaaS and IaaS). This puts a significant emphasis on the monitoring of SLAs, as well as detecting, predicting and resisting to violations, which appear to be more challenging in the case of CSBAs as various providers are dynamically involved. In the literature, there are two main approaches for systems monitoring, i.e., reactive and proactive. The reactive approach mainly reacts after an occurrence of a violation has occurred. While the proactive approach is more preventive, by possibly predicting potential future violations before their occurrence, and by reacting to avoid their occurrence.

The contributions of this paper are twofold: (i) we present a novel monitoring framework for CSBA, namely Proactive learning from SLA violation, based on MAPE-K[1] adaptation loop, and (ii) we concretely address the 'Analysis' component of the proposed framework. This novel proactive learning approach takes advantage of the massive amount of past process execution data in order to predict potential violations. It identifies the best counter measures that need to be applied. As a proof-of-concept of the proposed approach, a prototype has been developed that ascertains the applicability and feasibility of the proposed solution.

The rest of this paper is structured as follows. Section 2 discusses related work. Section 3 introduces a running scenario that will be used throughout the paper. Section 4 lays the background needed to understand the work proposed in this paper. The proposed predictive monitoring framework is presented in Section 5. Section 6 presents our proposed SLA violations learning approach. Section 7discusses the implementation of the proposed approach on a real-life log. Finally, Section 8 draws conclusions and perspectives.

2 LITERATURE REVIEW

There is a massive amount of work in the literature related to cloud-based environment, covering various aspects of this multi-disciplinary domain. In the next discussion, we are focusing on prominent efforts in the area of SLA monitoring and management in CBSA, which is appraised against the work proposed in this paper.and.

In principle, approaches for monitoring and detecting SLA violations with respect to QoS constraints are mainly based on techniques and strategies to adapt QoS settings according to changes and violations detected during execution of CSBA. In this case QoS parameters are generally used to repair and optimize a web service. Generally, these adaptive approaches are based on the ability to select and replace the failed services dynamically at runtime or during deployment. The selection is governed not only by the need to substitute services but to optimize the requirements of QoS of the system. Accordingly, the system must autonomously adapt itself in order to improve the quality of service of the process. In (Tao et al., 2014) proposed a novel hybrid and adaptive multi learners approach for online QoS modeling in the cloud; they described an adaptive solution that dynamically selects the best learning algorithm for prediction (Leitner et al., 2011). The proposals in (Fugini et al. 2010) and (Schmieders et al., 2011) address the problem of violation detection and adaptation of SLA contracts between several layers. For example, (Fugini et al., 2010) proposed a methodology to create, monitor and adapt the inter-layer SLA contracts. The SLA model includes parameters such as KPI (key performance indicators), KGI (indicators key objectives), and metrics infrastructure. (Schmieders et al., 2011) proposed a solution to avoid SLA violations by applying inter-layer techniques. The proposed approach uses three layers for the prediction of SLA violations. The identification of adaptation needs is based on the prediction of QoS, which uses assumptions about the characteristics of the execution context. In (Vincent et al., 2015), the authors introduced a Cloud Application and SLA monitoring architecture, and proposed two methods for determining the frequency these applications need to monitor, they also identified the challenges in regard with application provisioning and SLA enforcement, especially in multi-tenant Cloud services.

Discussion: The main limitation of the aforementioned approaches is that they only consider certain services regions (execution points) of the composition and do not consider all process tasks. Most of the works targeting SLA violations prediction is addressing grid environments or service-oriented infrastructure that differs from cloud infrastructure, therefore the applicability of

[1] MAPE-K (Monitoring, Analysis, Planning Execution-Knowledge).

these approaches on CSBA is limited. To the best of our knowledge, the approach proposed in this paper is the first that uses data mining techniques (Han et al., 2011) to learn from SLA violations (Vaitheki et al., 2014) in order to correlate between multiple violated SLAs. It recommends actions for automatically reconfiguring the CSBA to avoid the predicted violation before its occurrence.

3 MOTIVATION SCENARIO

To illustrate the ideas presented in this paper we will use a simple travel agency scenario, which is composed of three services: (i) Reserve Flight, (ii) Payment Service, and (iii) Reserve Hotel Service.

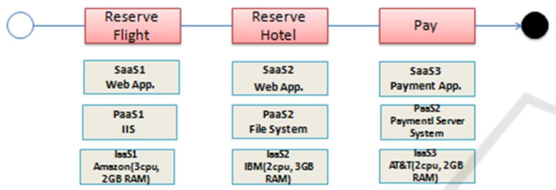


Figure 1: A simple process describing the travel agency scenario.

Table 1: QoS constraints of SLA relevant to the scenario.

SLA Parameter	Value
Response time	≤ 20 Sec
Storage (St)	≥ 2GB
CPU number	≥ 3

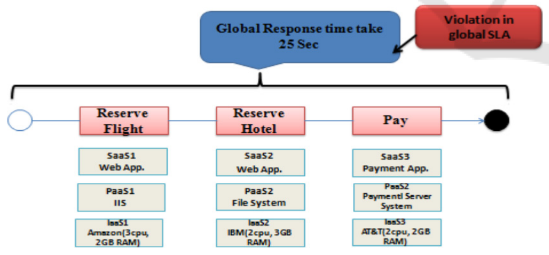


Figure 2: Travel agency SLA Violations.

We summarize in Table 1 the Service Level Objectives (SLOs) for our CSBA. It corresponds to the SLA specifications of all the QoS constraints for the whole application. Each cloud service provider involved in the Travel-Agency-CSBA configuration promises to satisfy the stipulated Qualities of Services (QoS) through a Service Level Agreement (SLA) with his consumer. Each of these services is made up of a mixture of rented Cloud Services (SaaS, PaaS and IaaS).This work aims at locating the failure event and determine adaptation actions in order to prevent its spread at the others layers, as soon as possible. The central focus of Travel-Agency CSBA scenario is the SLA between the client Travel-Agency and the cloud service providers offering the Reserve Fly, the Reserve hotel and the payment cloud services. Upon receiving a request placed by the customer 'C', a process instance is created. For this instance, the process execution starts with the activity 'Reserve Fly' (S1). Then, the SLA monitor is called and the software services are invoked if they are available. The maximum duration of the Response time of the whole process should be less than 20 seconds, a violation of the respective SLA occurs, as it can be seen in Figure 2.

4 BACKGROUND

Numerous tasks are reached by data mining. They can be classified in descriptive tasks which are the association rules in our case and predictive tasks which is here the decision Tree used for the prediction from execution logs. In this section, we first introduce the decision Tree, a commonly used data mining technique in order to build predictions models from execution logs to be able to predict potential violation and react to it proactively. This is followed by an overview of association rules.

4.1 Decision Tree

Objective: classification of people or things into groups by recognizing patterns. The user or the expert has always a tendency to structure or classify data into groups of similar objects called classes. For this purpose, he uses distance measurements in order to evaluate the belonging of an object to a class. The most known classification methods are nearest neighbor and decision trees. A decision tree seeks to represent the studied objects in a tree, according to a hierarchy of attributes. Decision trees are popular because they provide a synthetic representation of data. They are graphical representations of a classification procedure aiming to derive a result from a test of attributes (internal nodes of the tree). A node represents a class, an arc represents a partitioning predicate of the source class and the leaves of the tree are the classes we want to predict or explain their statements. CART (Breiman et al., 1984) and C4.5 are among the best known algorithms for generating decision tree. These algorithms generate classification rules easy to interpret by the user. The generated rules can be used to build predictive models. A decision tree is used to classify records by hierarchical division into subclasses.

4.2 Association Rules

Objective: associating what events are likely to occur together.

Association models aim to discover relationships or correlations in a set of items. Association rules is a data mining technique intending to find associated values in a given dataset and serving decision making. It has the following form: A->B (S %), (C %). This rule means that tuples satisfying conditions in **A** also satisfy conditions in **B**.

A rule is always given with two measures (the support (S) and the confidence (C)) describing its strength and its interestingness. The support (equation (1)) is the percentage of transactions that satisfy A and B among all the transactions of the transactions base. The confidence (equation (2)) is the percentage of transactions that verify the consequent of a rule among those that satisfy the antecedent (premise) data.A rule is always given with two measures (the support (S) and the confidence (C)) describing its strength and its interestingness. The support (equation (1)) is the percentage of transactions that satisfy A and B among all the transactions of the transactions base. The confidence (equation (2)) is the percentage of transactions that verify the consequent of a rule among those that satisfy the antecedent (premise) data.

$$Support\ A \rightarrow P(A \cup B) \qquad (1)$$

Confidence A $\rightarrow$ B = P (B/A) = Support AUB/Support (A) (2)

The extraction of association rules is the generation of the interesting rules with support and confidence greater than minimum thresholds of support and confidence. The process of extracting association rules involves two distinct phases. Firstly, the items having a support level that exceeds a certain threshold are segregated. Secondly, the most frequent items are combined in order to generate associations (Chelghoum, 2004).

5 A FRAMEWORK FOR PROACTIVE LEARNING FROM SLA VIOLATIONS

Figure 3 portrays a high-level architectural view of the proposed cross-layer self-adaptation framework. The framework is based on MAPE-K adaptation loop, introduced by IBM as an efficient and novel approach for self-adaptation in autonomic computing. As shown in Figure 3, the MAPE-K adaptation loop comprises of five main components corresponding to its acronym, which will be discussed in the following. The main focus of this paper is on the Analysis component of the MAPE-K loop, where a proactive learning approach is proposed (cf. Section 5) to predict potential QoS violations based on historical execution logs and react accordingly to avoid/prevent the predicted violation. Therefore, the upcoming sections will focus on presenting the details of these two main components.

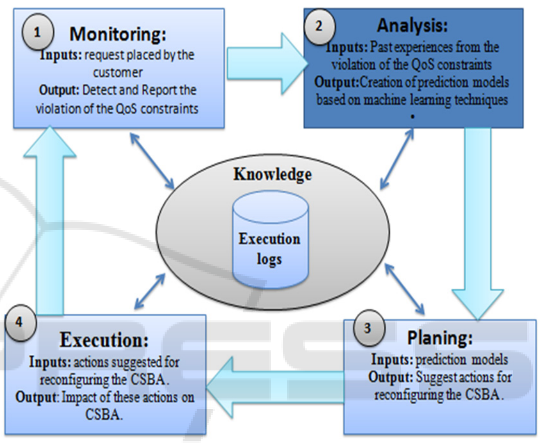

1
Monitoring:
Inputs: request placed by the customer
Output: Detect and Report the violation of the QoS constraints
2
Analysis:
Inputs: Past experiences from the violation of the QoS constraints
Output:Creation of prediction models based on machine learning techniques
Knowledge
Execution logs
4
Execution:
Inputs: actions suggested for reconfiguring the CSBA.
Output: Impact of these actions on CSBA.
3
Planing:
Inputs: prediction models
Output: Suggest actions for reconfiguring the CSBA.

Figure 3: High-level architectural view of the proposed proactive monitoring framework.

Knowledge: SLA is a predominant entity in cloud service based systems (CSBS). In CSBS, clients rent services from providers instead of buying services. This means that a CSBS is a composition of a list of rented services. Thus, SLA becomes critical to both service clients and providers and it needs to be monitored constantly while a CSBS is running with the aim to not only detect violations but also prevent them. As in CSBS a number of providers are involved, detecting and resisting violations (of multiple SLAs that engage different providers form different locations) are enormously challenging. Therefore, an efficient and effective approach is of a paramount importance for CSBS. As CSBS rely on third party cloud service providers, an SLA involves a ‘consumer’ and one to many ‘providers’.

Monitoring: Monitoring of SLA compliance is of crucial importance to the proposed framework. Monitoring is intended to be in a near real-time fashion in order to take corrective actions before it is too late (Yehia et al. 2014). Our intention also is to predict possible SLA violation and avoid them before they occur. To tackle the monitoring task, we will depend on complex event processing (CEP) technology.

Analysis: Analysis is based on the monitoring results; the analysis component is responsible for performing complex data analysis and reasoning, by the continuous interaction with the knowledge component. Based on information extracted from the historical traces, predictions and recommendations are provided for running instances. Such predictions and recommendation rely on data mining techniques, more specifically, decision trees.

Planning: Once a violation is predicted, the planning component takes the hand over. To deal with such a predicted violation, the planning component starts by searching for alternative solutions in order to avoid the occurrence of the predicted violation. The planning component will attempt to adapt the smallest possible set of services without directly targeting a re-engineering process of the whole system.

Execution: Based on the results of the prediction models constructed in the previous planning component, the execution component is responsible for selecting the adaptation plan (in the form of recommendations as passed from the planning component) with the highest probability of success. This evaluation will iteratively enhance the quality of the predication models by better learning (the feedback loop in Figure 3).Our work in this component is ongoing.

6 THE PROPOSED LEARNING APPROACH IN CLOUD ENVIRONMENTS

In this section, we present the details of the proposed learning approach, which combines different existing techniques ranging from learning approaches to decision tree learning, to provide predictions, at runtime, about the achievement of business goals in a Business Process (BP) execution trace. In the following sections, we provide an overview of the approach. Section 7 discusses the implementation of the proposed approach as a Proof-Of-Concept (POC) of the realization of the proposed approach.

6.1 The Proposed Learning Approach

The process of learning from SLA violation and making dependencies precedence between different SLAs violations in CSBAs have been identified as major research challenges in Cloud environments.

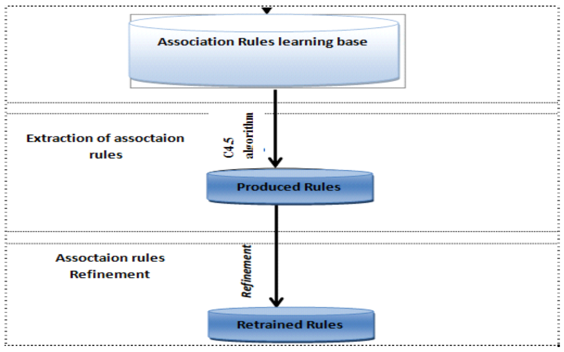

Association Rules learning base
Extraction of assoctaion rules
C4.5 alorithm
Produced Rules
Assoctaion rules Refinement
Refinement
Retrained Rules

Figure 4: Proposed SLA Violation Learning approach: Architecture overview.

SLA does not contain information about the dynamicity of the system. In other words, it is independent of the context of the business process, and it contains information about the service behavior or quality provided by the service which we aim to exploit. SLAs are not mathematically defined. That means that the semantics of the SLA elements and metrics are defined in natural languages, which makes it harder to understand the semantic of QoS, and it is usually dependent on the client and provider contract. Thus, being precise and formal about SLA semantic is necessary. SLAs violations come from different kind of failures, determining the appropriate type of actions to be taken when predicting an SLA violation is equally important.

First Phase: Learning phase: It is a continuous evolving process. The association rules extraction is explained as follows:

Given: a set of historical BP event logs of SLA violations.

Find: Association rules

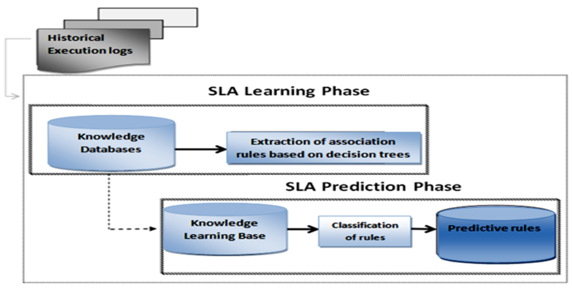


Figure 5: Learning Phase.

Our method of Association Rules (AR) determination goes through three steps:

- The first step is to discover frequent itemsets.
- The second step is dedicated to find out ARs on the basis of the first step outputs.
- The third step is the refinement of the extracted ARs.

These steps are depicted in the diagram shown in Figure 5. Two steps are combined in order to carry out ARs. They are respectively the generation of frequent itemsets and the extraction of association rules. The frequent itemsets are extracted as defined in Algorithm 1.

Algorithm 1: Extraction and refinement of AR.

```
Inputs: Execution logs
Outputs: ARs
      Reject-rule: = Boolean
Begin
      Reject-rule: =False
Do
 (1) Find all the frequent itemsets
     by C4.5 execution (execution
     logs)
 (2) Find all the possible ARs by
     C4.5 execution
 (3) Save the obtained ARs
End
 (4) Save the rules set containing
     only correct rules
End
```

The input corresponds to the set of historical data generated by the previous SLA violation of CSBA. The AR contains QoS property of SLA in the antecedent and the violated SLA as consequent of the rule. The proposed formula of the AR is given in equation:

Onevent**if**condition**then**(perform)action

The next phase is the prediction phase (described below), in that phase some historical executions of CSBA are necessary to bootstrap the prediction. The concrete amount of instances that are necessary, depend both on the expected quality of prediction and on the size and complexity of the service composition.

Second Phase: SLA Prediction

The objective of this phase is to (i) predict potential violation, and to (ii) construct/build the best configuration as the recommendation from what have been detected and learned based on the previous learning phase.The set of such entries is presented in the Decision Tree of the Figurewhere the output is the frequent set of dependencies.The Prediction algorithm gives precise predictions and avoids unnecessary adaptations. Generally, the approach that predicts SLA violations is based on the idea of predicting concrete SLO values based on whatever monitoring information is already available. In order to identify which data should be used to train which model, some domain knowledge is necessary. However, dependency analysis can be used to identify the factors that have the biggest influence on the respective SLOs.

The association rules prediction is explained as follows:

<u>Given:</u> ARs extracted.

<u>Find:</u> Predictive ARs.

The process of this phase is shown in Algorithm 2:

Algorithm 2: Prediction of predictive AR.

```
Inputs: ARs
Outputs: Predicted ARs
      Reject-rule: = Boolean
Begin
      Reject-rule: =False
Do
 (5) Compare all the obtained ARs in
     different time intervals
 (6) Suggest predicted Rules
End
 (7) Save the predicted rules set
     containing only correct rules
End
```

6.2 Exemplifying on the Running Scenario

In this section, the prediction of violations is applied on the Travel agency CSBA scenario (described in section 3). The rules below are an outcome of the Decision Tree mining the data sets sent as inputs from an Excel file of the Travel-Agency scenarios is described in (section 3).

Example of rule

IF sum RT2+RT3 ≤ 10 = no **And** RT3≤ 6**And**RT2 ≤ 11 **And** RT1≥ 5 **Then** Violation = Viol (P1, P6), the configuration will be: {Response time **≤ 20, 1CPU, 2GB RAM}.**

As presented in Figure 2, IaaS is an infrastructure as a service that is a rented service from amazon service provider with 1 CPU and 2 GB Ram and it promises PaaS is a platform as a service that presents a rented IIS (internet information services), with 1 CPU and 2 GB Ram and it promises to satisfy response time <20 sec. The Global SLA service promises to satisfy response time <20 sec, giving 10 sec as a response time for

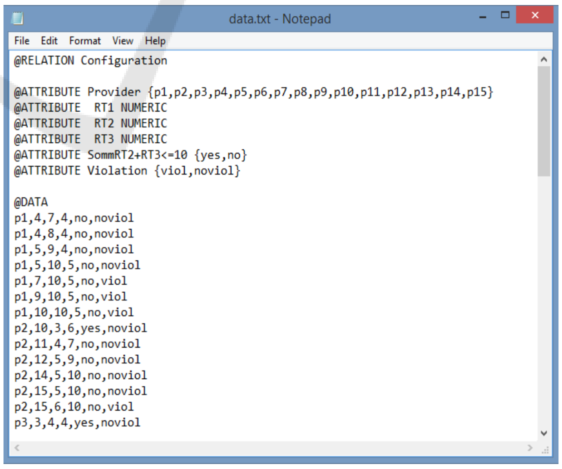
Table 2: Proactive Actions.

Violation ID	**Violation Type**	**Action Type**	**Action**
Viol1	Violation Qos	Reparation	Raise the RAM capacity to 2GB
Viol 2	Availability	Substitution	Change violated service
Viol 3	Security	Reparation	Adapt the service to security police

each of ‘Reserve Hotel’ and 5 sec for ‘Reserve Flight’, and 5 sec for ‘Payment Application’ service. The response time for example could be violated when any of these application services has an internal violation in response time at the level of SaaS, PaaS or IaaS. In order to avoid such situation, the SLA manager acts proactively based on history of completed activities. At the same time, another monitoring component detects that there is an I/O failure at the SaaS layer as S1 has produced a wrong output. A specific rule is triggered that derives the best strategy which consists of executing another instance of *the web service* on a more powerful server with a better memory and CPU allocation. (Amazon (3cpu, 3GB RAM)). Assume that a Monitoring Component, running at the server where *the web service* is executed, detects that the available main memory is not sufficient (IaaS layer) for *the web service*. At this stage, proactive actions are suggestions to be taken based on some predefined suitable actions for each type of violation the system may encounter. In Table 2, we can see that each violation has a violation type that could be availability or security depending on the type of violation. The actions taken are of two types, namely surgery and elastic actions.

7 IMPLEMENTATION

To demonstrate the applicability and feasibility of our approach, we developed a prototype[2]using JAVA. We trigger the execution of 100 process instances using a test client. For each of these instances we select the concrete supplier service and shipper service randomly in order to ensure that history data used for learning contains metrics data on each of these services. During process instance execution, the previously specified metrics are measured and saved in the knowledge database. Then, for each checkpoint a decision tree is learned using the J48 algorithm. For the implementation of the Predictor, we rely on the WeKa J48 implementation of the C4.5 algorithm, which takes as input a‘.arff’ file and builds a decision tree as shown in Figure 6: Text files of real data. The ‘.arff’ file contains a list of typed variables (including the target variable) and, for each trace prefix (e.g., for each data snapshot), the corresponding values are also maintained. The resulting Decision Tree is then analyzed to generate predictions and recommendations as shown in Figure 6. The configuration manager is responsible for configuring the CSBA. The proactive actions suggested by the proactive engine are mapped into the configuration manager to take the action. The action taken is then stored in the knowledgebase. The algorithm searches in the database (as shown in (Figure 7) for suitable actions that can be used. Below in figure 8 is a part of the Excel file that we used for our decision. For example, as shown in the Table 3 since the violation is Response Time, then the suitable action is to add 1 CPU to the violating service.



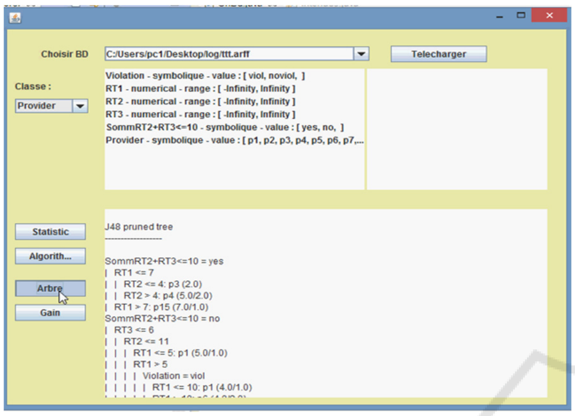
```
@RELATION Configuration

@ATTRIBUTE Provider {p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13,p14,p15}
@ATTRIBUTE  RT1 NUMERIC
@ATTRIBUTE  RT2 NUMERIC
@ATTRIBUTE  RT3 NUMERIC
@ATTRIBUTE SommRT2+RT3<=10 {yes,no}
@ATTRIBUTE Violation {viol,noviol}

@DATA
p1,4,7,4,no,noviol
p1,4,8,4,no,noviol
p1,5,9,4,no,noviol
p1,5,10,5,no,noviol
p1,7,10,5,no,viol
p1,9,10,5,no,viol
p1,10,10,5,no,viol
p2,10,3,6,yes,noviol
p2,11,4,7,no,noviol
p2,12,5,9,no,noviol
p2,14,5,10,no,noviol
p2,15,5,10,no,noviol
p2,15,6,10,no,viol
p3,3,4,4,yes,noviol
```

Figure 6: Text files of real data.

We evaluated experimentally the model’s performance and accuracy. The experiments were performed on a machine with quad-core CPU 2.6 GHz, 8GB RAM and Mac OS X operating system. This experiment evaluates the algorithm’s raw

[2] A video demonstration is available at: https://www.youtube.com/watch?v=oDEFYGBPdH0

relevant and absolute accuracy. The static metrics precision and recall is measured while fluctuating the interval size from 4 to 20 events. Figure 8 shows that relevant precision is 1 for small intervals and falls while increasing the interval size, while absolute precision fluctuates similarly at lower levels (as more irrelevant sub-patterns are discovered).

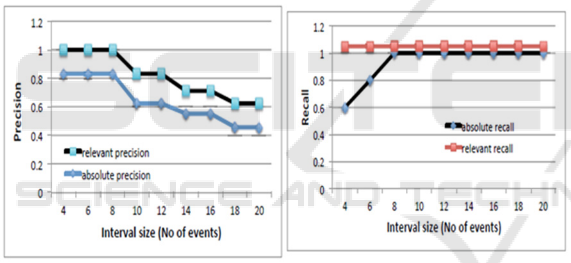


```
C:/Users/pc1/Desktop/log/ttt.arff
```

```
Violation - symbolique - value : [ viol, noviol, ]
RT1 - numerical - range : [ -Infinity, Infinity ]
RT2 - numerical - range : [ -Infinity, Infinity ]
RT3 - numerical - range : [ -Infinity, Infinity ]
SommRT2+RT3<=10 - symbolique - value : [ yes, no, ]
Provider - symbolique - value : [ p1, p2, p3, p4, p5, p6, p7,...
```

```
J48 pruned tree
------------------

SommRT2+RT3<=10 = yes
|  RT1 <= 7
|  |  RT2 <= 4: p3 (2.0)
|  |  RT2 > 4: p4 (5.0/2.0)
|  RT1 > 7: p15 (7.0/1.0)
SommRT2+RT3<=10 = no
|  RT3 <= 6
|  |  RT2 <= 11
|  |  |  RT1 <= 5: p1 (5.0/1.0)
|  |  |  RT1 > 5
|  |  |  |  Violation = viol
|  |  |  |  |  RT1 <= 10: p1 (4.0/1.0)
```

Figure 7: A screenshot of ARs Extraction.



(a) Precision (b) Recall

Figure 8: Evaluation results.

8 CONCLUSION AND FUTURE WORK

In this paper, we proposed a proactive framework for learning from SLA violations in Cloud Service Based application. We proposed a proactive approach that uses business process execution logs to learn from past violations and extracts knowledge between violated SLA's, This is based on building predictions models that is capable of predicting potential violations before their occurrence, as well as recommending proactive response recovery/preventive actions. As future work directions, we plan to design and implement the planning and Execution components of the comprehensive framework for automatic cross-layer self-adaptation of service-based business processes running on cloud environments, proposed in this paper. Future work will also focus on validating the approach by applying it to large-scale case studies from diverse problem domains.

REFERENCES

Boniface M., PhillipsS. C., A. Sanchez-Macian, and Surridge M.. Dynamic service provisioning using GRIA SLAs. In ICSOC'07 workshops, 2007.

Brandic, I.,Towards self-manageable cloud services. In *33rd Annual IEEE COMPSAC'09,* 2009.

Breiman, L., Friedman, J.H., Olshen, R.A., Stone, and C.J. (1984). Classification andRegression Trees. *Edition Wadsworth & Brooks.*Montery, California, ISBN 0-412-04841-8, 358 pages.

Chelghoum, N., (2004). Fouille de données spatiales, Un problème de fouille de donnéesmulti-tables. *Thèse de doctorat présentée et soutenue publiquement à l'université deVersailles Saint-Quentin-en-Yvelines* U.F.R de sciences Par Nadjim CHELGHOUM le16 décembre 2004.

Fugini,M., Siadat, H.,SLA contract for Cross-Layer monitoring and adaptation. In S. Rinderle-Ma, S. Sadiq, and F. Leymann, editors, *BPM Workshops*, pages 412--423., 2010.

Han, J., Kamber, M., Pei, J., , Data Mining: Concepts and Techniques. *(3rd ed.)Morgan Kaufmann*, USA (2011).

Leitner, P., Michlmayr, A., Rosenberg, F., Dustdar, S., Monitoring, Prediction and Prevention of SLA Violations in Composite Services, *icws, pp.369-376, 2010 IEEE SCC,* 2010.

Peter, M., Timoth, y G., "The NIST definition of cloud computing," 2011.

Schmieders, E., Micsik, A., Oriol, M.,Mahbub, K., and KazhamiakinR., Combining SLA prediction and cross layer adaptation for preventing SLA violations". *In Proceedings of the 2nd Workshop on Software Services: Cloud Computing and Applications based on Software Services*, Timisoara, Romania, June 2011.

Tao, C., Rami, B., Xin, Y., Online QoS Modeling in the Cloud: A Hybrid and Adaptive Multi-Learners Approach, *the 7th IEEE/ACM UCC,* London, UK, 2014.

Vaitheki, K.,Urmela, S., A SLA violation reduction technique in Cloud by Resource Rescheduling Algorithm (RRA), *International Journal of Computer Application and Engineering Technology* Volume 3-Issue 3, July 2014. Pp. 217-224.

Vincent, C.,Emeakaroha, Marco, A., S.,Netto, IvonaBrandic, César, A., F., De Rose, Application-Level Monitoring and SLA Violation Detection for Multi-Tenant Cloud Services, *Emerging Research in Cloud Distributed Computing Systems,* 2015.

Yehia, T., Rafiqul, H., Dinh Khoa, N., Béatrice, F., :PAEAN4CLOUD: A Framework for Monitoring and Managing the SLA Violation of Cloud Service-based Applications. *CLOSER 2014:* 361-371.