Applying Data Mining Algorithms to Calculate the Quality of Service of Workflow Processes

Jorge Cardoso

Department of Mathematics and Engineering
9000-390 Funchal, Portugal
jcardoso@uma.pt

Abstract. Organizations have been aware of the importance of Quality of Service (QoS) for competitiveness for some time. It has been widely recognized that workflow systems are a suitable solution for managing the QoS of processes and workflows. The correct management of the QoS of workflows allows for organizations to increase customer satisfaction, reduce internal costs, and increase added value services. In this paper we show a novel method, composed of several phases, describing how organizations can apply data mining algorithms to predict the QoS for their running workflow instances. Our method has been validated using experimentation by applying different data mining algorithms to predict the QoS of workflow.

Keywords: Quality of Service, Data Mining, Business Process, Workflow.

1 Introduction

The increasingly global economy requires advanced information systems. Business Process Management Systems (BPMS) provide a fundamental infrastructure to define and manage several types of business processes. BPMS, such as Workflow Management Systems (WfMS), have become a serious competitive factor for many organizations that are increasingly faced with the challenge of managing e-business applications, workflows, Web services, and Web processes. WfMS allow organizations to streamline and automate business processes and reengineer their structure; in addition, they increase efficiency and reduce costs.

One important requirement for BMPS and WfMS is the ability to manage the Quality of Service (QoS) of processes and workflows [1]. The design and composition of processes cannot be undertaken while ignoring the importance of QoS measurements. Appropriate control of quality leads to the creation of quality products and services; these, in turn, fulfill customer expectations and achieve customer satisfaction. It is not sufficient to just describe the logical or operational functionality of activities and workflows. Rather, design of workflows must include QoS specifications, such as response time, reliability, cost, and so forth.

One important activity, under the umbrella of QoS management, is the prediction of the QoS of workflows. Several approaches can be identified to predict the QoS of

workflows before they are invoked or during their execution, including statistical algorithms [1], simulation [2], and data mining based methods [3, 4].

The latter approach, which uses data mining methods to predict the QoS of workflows, has received significant attention and has been associated with a recent new area coined as Business Process Intelligence (BPI). In this paper, we investigate the enhancements that can be made to previous work on BPI and business process quality to develop more accurate prediction methods.

The methods presented in [3, 4] can be extended and refined to provide a more flexible approach to predict the QoS of workflows. Namely, we intend to identify the following limitations that we will be addressing in this paper with practical solutions and empirical testing:

1. In contrast to [4], we carry out QoS prediction based on path mining and by creating a QoS activity model for each workflow activity. This combination increases the accuracy of workflow QoS prediction.
2. In [4], time prediction is limited since workflow instances can only be classified to "have" or "not to have" a certain behavior. In practice, it means that it is only possible to determine that a workflow instance will have, for example, the "last more than 15 days" behavior or will not have that behavior. This is insufficient since it does not give an actual estimate for the time a workflow will need for its execution. Our method is able to deduce that a workflow wi will probably take 5 days and 35 minutes to be completed with a prediction accuracy of 78%.
3. In [4], the prediction of the QoS of a workflow is done using decision trees. We will show that MultiBoost Naïve Bayes outperforms the use of decision trees to predict the QoS of a workflow.

This paper is structured as follows: In Section 2, we present our method of carrying out QoS mining based on path mining, QoS activity models, and workflow QoS estimation. Section 3 describes the set of experiments that we have carried out to validate the QoS mining method we propose. Section 4 presents the related work in this area. Finally, section 5 presents our conclusions.

2 Motivation

Nowadays, a considerable number of organizations are adopting workflow management systems to support their business processes. The current systems available manage the execution of workflow instances without any quality of service management on important parameters such as delivery deadlines, reliability, and cost of service.

Let us assume that a workflow is started to deliver a particular service to a customer. It would be helpful for the organization supplying the service to be able to predict how long the workflow instance will take to be completed or the cost associated with its execution. Since workflows are non-deterministic and concurrent, the time it takes for a workflow to be completed and its cost depends not only on which activities are invoked during the execution of the workflow instance, but also

depends on the time/cost of its activities. Predicting the QoS that a workflow instance will exhibit at runtime is a challenge because a workflow schema w can be used to generated n instances, and several instances w_i ($i \leq n$) can invoke a different subset of activities from w. Therefore, even if the time and cost associated with the execution of activities were static, the QoS of the execution of a workflow would vary depending on the activities invoked at runtime.

For organizations, being able to predict the QoS of workflows has several advantages. For example, it is possible to monitor and predict the QoS of workflows at any time. Workflows must be rigorously and constantly monitored throughout their life cycles to assure compliance both with initial QoS requirements and targeted objectives. If a workflow management system identifies that a running workflow will not meet initial QoS requirements, then adaptation strategies [5] need to be triggered to change the structure of a workflow instance. By changing the structure of a workflow we can reduce its cost or execution time.

3 QoS Mining

In this section we focus on describing a new method that can be used by organizations to apply data mining algorithms to historical data and predict QoS for their running workflow instances. The method presented in this paper constitutes a major and significant difference from the method described in [4]. The method is composed of three distinct phases (figure 1) that will be explained in the following subsections.

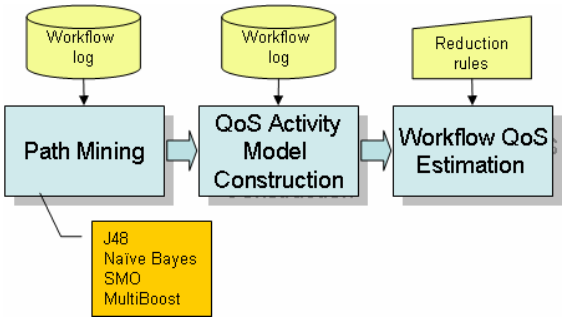

Fig. 1. Phases of workflow QoS mining

In the first phase, the workflow log is analyzed and data mining algorithms are applied to predict the path that will be followed by workflow instances at runtime. This is called path mining. Path mining identifies which activities will most likely be executed in the context of a workflow instance. Once we know the path, we also know the activities that will be invoked at runtime. For each activity we construct a QoS activity model based on historical data which describes the runtime behavior (duration and cost) of an activity. In the last phase, we compute the QoS of the overall workflow based on the path predicted and from the QoS activity models using a set of reduction rules.

3.1 Path Mining

As we have stated previously, the QoS of a workflow is directly dependent on which activities are invoked during its execution. Different sets of activities can be invoked at runtime because workflows are non-deterministic. Path mining [6, 7] uses data mining algorithms to predict which path will be followed when executing a workflow instance.

Definition (Path): A path P is a continuous mapping P: [a, b] $\rightarrow C^o$, where P(a) is the initial point, P(b) is the final point, and C^o denotes the space of continuous functions. A path on a workflow is a sequence $\{t_1, t_2, \ldots, t_n\}$ such that $\{t_1, t_2\}$, $\{t_2, t_3\}$, $\ldots$,$\{t_{n-1}, t_n\}$ are transitions of the workflow and the t_i are distinct. Each t_i is connected to a workflow activity.

A path is composed of a set of activities invoked and executed at runtime by a workflow. For example, when path mining is applied to the simple workflow illustrated in figure 2, the workflow management system can predict the probability of paths A, B, and C being followed at runtime. Paths A and B have each 6 activities, while path C has only 4 activities. In figure 2, the symbol ⊕ represented non-determinism (i.e., a xor-split or xor-join).

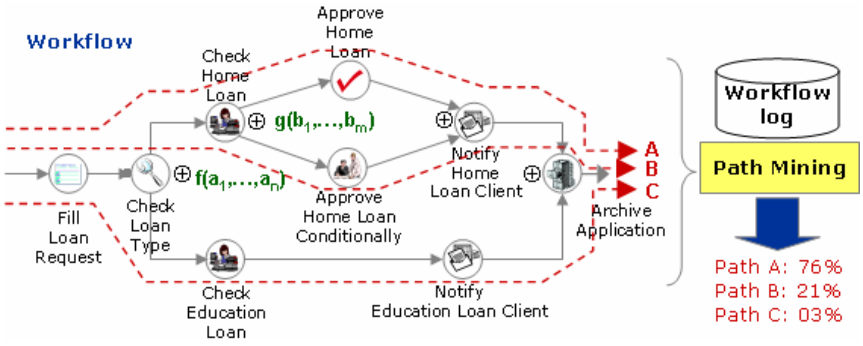

Fig. 2. Path mining

To perform path mining, current workflow logs need to be extended to store information indicating the values and the type of the input parameters passed to activities and the output parameters received from activities. The values of inputs/outputs are generated at runtime during the execution of workflow instances. Table 1 shows an extended workflow log which accommodates input/output values of activity parameters that have been generated at runtime. Each 'Parameter/Value' entry as a type, a parameter name, and a value (for example, string loan-type="car-loan").

Additionally, the log needs to include path information: a path describing the activities that have been executed during the enactment of a process. This information can easily be stored in the log. From the implementation perspective it is space efficient to store in the log only the relative path, relative to the previous activity, not

the full path. Table 1 shows the full path approach because it is easier to understand how paths are stored in the log.

Table 1. Extended workflow log

	Workflow log extension	
…	**Parameter/Value**	**Path**
…	int SSN=7774443333; string loan-type="car-loan" …	…
…	string name=jf@uma.pt; …	{FillLoanRequest, CheckLoanType, CheckCarLoan, ApproveCarLoan, NotifyCarLoanClient, ArchiveApplication}
…	…	…

During this phase, and compared to [3, 4], we only need to add information on paths to the log. Once enough data is gathered in the workflow log, we can apply data mining methods to predict the path followed by a process instance at runtime based on instance parameters. In section 4.2, we will show how the extended workflow log can be transformed to a set of data mining instances. Each data mining instance will constitute the input to machine learning algorithm.

3.2 QoS activity model construction

After carrying out path mining, we know which activities a workflow instance will be invoking in the near future. For each activity that will potentially be invoked we build what we call a QoS activity model. The model includes information about the activity behavior at runtime, such as its cost and the time the activity will take to execute [1].

Each QoS activity model can be constructed by carrying out activity profiling. This technique is similar to the one used to construct operational profiles. Operational profiles have been proposed by Musa [8, 9] to accurately predict future the reliability of applications. The idea is to test the activity based on specific inputs. In an operational profile, the input space is partitioned into domains, and each input is associated with a probability of being selected during operational use. The probability is employed in the input domain to guide input generation. The density function built from the probabilities is called the operational profile of the activity. At runtime, activities have a probability associated with each input. Musa [9] described a detailed procedure for developing a practical operational profile for testing purposes. In our case, we are interested in predicting, not the reliability, but the cost and time associated with the execution of workflow activities.

During the graphical design of a workflow, the business analyst and domain expert construct a QoS activity model for each activity using activity profiles and empirical knowledge about activities. The construction of a QoS model for activities is made at

design time and re-computed at runtime, when activities are executed. Since the initial QoS estimates may not remain valid over time, the QoS of activities is periodically re-computed, based on the data of previous instance executions stored in the workflow log.

The re-computation of QoS activity metrics is based on data coming from designer specifications (i.e. the initial QoS activity model) and from the workflow log. Depending on the workflow data available, four scenarios can occur (Table II): a) For a specific activity *a* and a particular dimension *Dim* (i.e., time or cost), the average is calculated based only on information introduced by the designer (Designer $Average_{Dim}(a)$); b) the average of an activity *a* dimension is calculated based on all its executions independently of the workflow that executed it (Multi-Workflow $Average_{Dim}$ (a)); c) the average of the dimension *Dim* is calculated based on all the times activity *a* was executed in any instance from workflow *w* (Workflow $Average_{Dim}(t, w)$); and d) the average of the dimension of all the times activity *t* was executed in instance *i* of workflow *w* (Instance $Average_{Dim}(t, w, i)$).

Table 2. QoS dimensions computed at runtime

a)	$QoS_{Dim}(a) =$	Designer $Average_{Dim}(a)$
b)	$QoS_{Dim}'(a) =$	wi_1* Designer $Average_{Dim}(a)$ + wi_2* Multi-Workflow $Average_{Dim}(a)$
c)	$QoS_{Dim}(a, w) =$	wi_1* Designer $Average_{Dim}(a)$ + wi_2* Multi-Workflow $Average_{Dim}(a)$ + wi_3*Workflow $Average_{Dim}(a, w)$
d)	$QoS_{Dim}(a, w, i) =$	wi_1* Designer $Average_{Dim}(a)$ + wi_2* Multi-Workflow $Average_{Dim}(a)$ + wi_3* Workflow $Average_{Dim}(a, w)$ + wi_4* Instance Workflow $Average_{Dim}(a,w, i)$

Let us assume that we have an instance *i* of workflow *w* running and that we desire to predict the QoS of activity $a \in w$. The following rules are used to choose which formula to apply when predicting QoS. If activity *a* has never been executed before, then formula a) is chosen to predict activity QoS, since there is no other data available in the workflow log. If activity *a* has been executed previously, but in the context of workflow w_n, and w != w_n, then formula b) is chosen. In this case we can assume that the execution of *a* in workflow w_n will give a good indication of its behavior in workflow *w*. If activity *a* has been previously executed in the context of workflow *w*, but not from instance *i*, then formula c) is chosen. Finally, if activity *a* has been previously executed in the context of workflow *w*, and instance *i*, meaning that a loop has been executed, then formula d) is used.

The workflow management system uses the formulae from Table II to predict the QoS of activities. The weights wi_k are manually set. They reflect the degree of correlation between the workflow under analysis and other workflows for which a set of common activities is shared. At this end of this second phase, we already know the activities of a workflow instance that will most likely be executed at runtime, and for each activity we have a model of its QoS, i.e. we know the time and cost associated with the invocation of the activity.

3.3 Workflow QoS Estimation

Once we know the path, i.e. the set of activities which will be executed by a workflow instance, and we have a QoS activity model for each activity, we have all the elements required to predict the QoS associated with the execution of a workflow instance.

To compute the estimated QoS of a process in execution, we use a variation of the Stochastic Workflow Reduction (SWR) algorithm [1]. The variation of the SWR algorithm that we use does not include probabilistic information about transitions. The SWR is an algorithm for computing aggregate QoS properties step-by-step. At each step a reduction rule is applied to shrink the process. At each step the time and cost of the activities involved is computed. This is continued until only one activity is left in the process. When this state is reached, the remaining activity contains the QoS metrics corresponding to the workflow under analysis. For the reader interested in the behavior of the SWR algorithm we refer to [1].

For example, if the path predicted in the first phase of our QoS mining method includes a parallel system, as show in Figure 3, the parallel system reduction rule is applied to a part of the original workflow (Figure 3.a) and a new section of the workflow is created (Figure 3.b).

A system of parallel activities t_1, t_2, …, t_n, an *and* split activity t_a, and an *and* join activity t_b can be reduced to a sequence of three activities t_a, t_{1n}, and t_b. In this reduction, the incoming transitions of t_a and the outgoing transition of activities t_b remain the same. The only outgoing transitions from activity t_a and the only incoming transitions from activity t_b are the ones shown in the figure below.

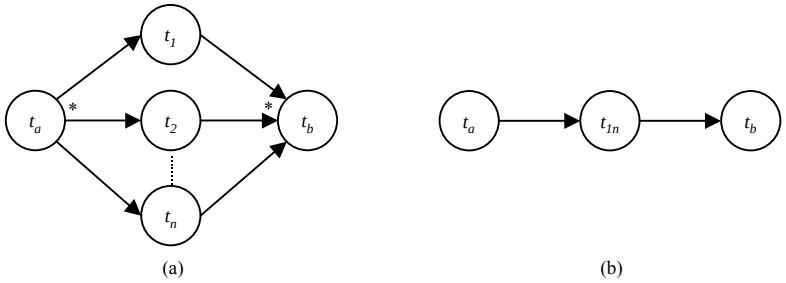

Fig. 3. Parallel system reduction

The QoS of the new workflow is computed using the following formulae (the QoS of tasks t_a and t_b remain unchanged):

$$\text{Time}(t_{1n}) = \text{Max}_{i \in \{1..n\}} \{\text{Time}(t_i)\} \text{ and}$$
$$\text{Cost}(t_{1n}) = \sum_{1 \le i \le n} \text{Cost}(t_i)$$

Reduction rules exist for sequential, parallel, conditional, loop, and network systems [1]. These systems or pattern are fundamental since a study on fifteen major workflow management systems [10] showed that most systems support the reduction rules

presented. Nevertheless, additional reduction rules can be developed to cope with the characteristics and features of specific workflow systems.

Our approach to workflow QoS estimation – which uses a variation of the SWR algorithm – addresses the third point that we raised in the introduction and shows that the prediction of workflow QoS can be used to obtain actual metrics (e.g. the workflow instance *w* will take 3 days and 8 hours to execute) and not only information that indicates if an instance takes "more" than *D* days or "less" than *D* days to execute.

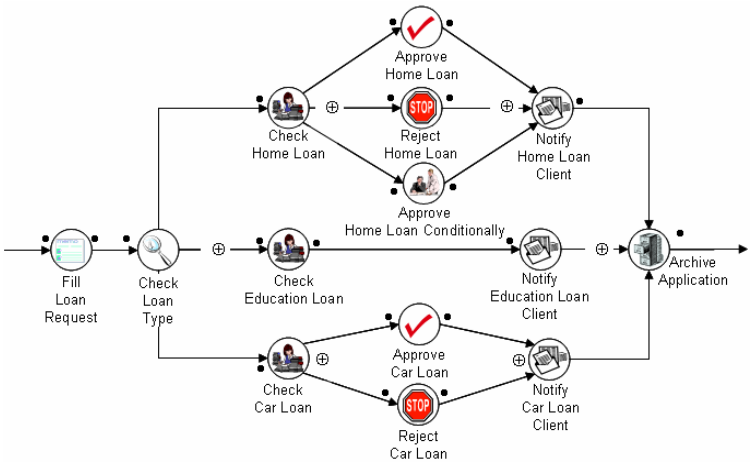

Fig. 4. The loan process

4 Experiments

In this section, we describe the data set that has been used to carry out workflow QoS mining, how to apply different data mining algorithms and how to select the best ones among them, and finally we discuss the results obtained. While we describe the experiments carried out using the loan process application (see Figure 4), we have replicated our experiments using a university administration process. The conclusions that we have obtained are very similar to the one presented in this section.

4.1 Workflow scenario

A major bank has realized that to be competitive and efficient it must adopt a new and modern information system infrastructure. Therefore, a first step was taken in that direction with the adoption of a workflow management system to support its processes. One of the services supplied by the bank is the loan process depicted in Figure 4. While the process is simple to understand, a complete explanation of the process can be found in [6].

4.2 Path mining

To carry out path mining we need to log information about the execution of workflow instances. But before storing workflow instances data we need to extended our workflow management log system, as explained in section 3.1, to store information indicating the values of the input parameters passed to activities and the output parameters received from activities (see [6, 7] for an overview of the information typically stored in the workflow log). The information also includes the path that has been followed during the execution of workflow instances.

To apply data mining algorithms to carry out path mining, the data present in the workflow log need to be converted to a suitable format to be processed by data mining algorithms. Therefore, we extract data from the workflow log to construct data mining instances. Each instance will constitute an input to machine learning and is characterized by a set of six attributes:

income, loan_type, loan_amount, loan_years, Name, SSN

The attributes are input and output parameters from the workflow activities. The attributes income, loan_amount, loan_years and SSN are numeric, whereas the attributes loan_type and name are nominal. Each instance is also associated with a class (named *[path]*) indicating the path that has been followed during the execution of a workflow when the parameters were assigned specific values. Therefore, the final structure of a data mining instance is:

income, loan_type, loan_amount, loan_years, Name, SSN, [path]

In our scenario, the path class can take one of six possible alternatives indicating the path followed during the execution of a workflow when activity parameters were assigned specific values (see Figure 4 to identify the six possible paths that can be followed during the execution of a loan workflow instance).

Having our extended log ready, we have executed the workflow from Figure 4 and logged a set of 1000 workflow instance executions. The log was then converted to a data set suitable to be processed by machine learning algorithms, as described previously.

We have carried out path mining to our data set using four distinct data mining algorithms: J48 [11], Naïve Bayes (NB), SMO [12], and MultiBoost [13]. J48 was selected as a good representative of a symbolic method, Naïve Bayes as a representative of a probabilistic method, and the SMO algorithm as representative of a method that has been successfully applied in the domain of text-mining. Multiboost is expected to improve performance of single classifiers with the introduction of meta-level classification.

Since when we carry out path mining to a workflow not all the activity input/ouput parameters may be available (some activities may not have been invoked by the workflow management system when path mining is started), we have conducted experiments with a variable number of parameters (in our scenario, the parameters under analysis are: income, loan_type, loan_amount, loan_years, name, and SSN) ranging from 0 to 6. We have conducted 64 experiments (2^6); analyzing a total of 64000 records containing data from workflow instance executions.

Accuracy of path mining. The first set of experiments was conducted using J48, Naïve Bayes, and SMO methods with and without the Multiboost (MB) method. We obtained a large number of results that are graphically illustrated in figure 5. The chart indicates for each of the 64 experiments carried out, the accuracy of path mining.

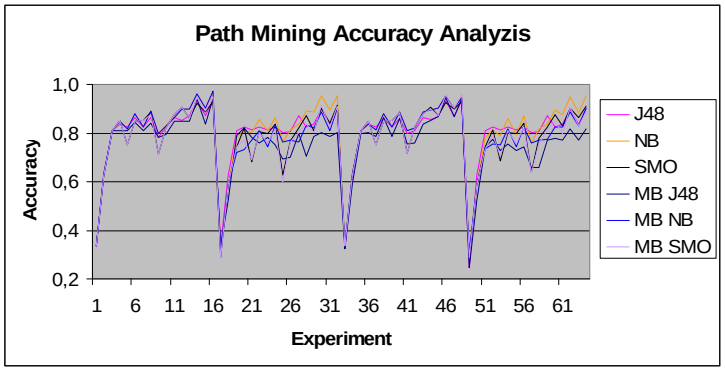

Fig. 5. Accuracy analysis of path mining

The chart indicates, for example, that in experiment nº12, when we use two parameters to predict the path that will be followed by a workflow instance from Figure 4, we achieve a prediction accuracy of 87,13% using the J48 algorithm. Due to space limitation, the chart in Figure 4 does not indicate which parameters or the number of parameters that have been utilized in each experiment.

Table 3. Summary results of accuracy analysis of path mining

	J48	**NB**	**SMO**
Avg acc.	75,43%	78,84%	77,79%
Min acc.	24,55%	30,84%	29,04%
Max acc.	93,41%	96,41%	93,11%
	MB J48	**MB NB**	**MB SMO**
Avg acc.	79,74%	81,11%	78,28%
Min acc.	24,55%	30,84%	29,04%
Max acc.	94,61%	97,31%	96,11%

For reasons of simplicity and as a summary, we computed the average, the minimum, and the maximum accuracy for each method for all the experiments carried out. The results are shown in Table 3.

On average the Naïve Bayes approach performs better than all other single methods when compared to each other. When the number of parameters is increased, the accuracy of Naïve Bayes improves. It can be seen that all the methods produced more accurate results when a more appropriate set of parameters was proposed. The worst results were produced by the J48 and SMO algorithms. It is safe to assume that these algorithms overfitted and were not able to find a generalized concept. That is probably a result of the nature of the dataset that contains parameters and that introduced noise. These results address the third point that was raised in the introduction and show that path prediction using MultiBoost Naïve Bayes outperforms the use of decision trees.

Next we added the meta-level of the multiboost algorithm and repeated the experiments. As expected, the multiboost approach made more accurate prognoses. All the classifiers produced the highest accuracy in experiment 16, since this experiment includes the 4 most informative parameters (i.e. income, loan_type, loan_amount, and loan_years). In order to evaluate which parameters are the most informative, we have used information gain.

4.3 QoS activity model construction

Once we have determined the most probable path that will be followed by a workflow at runtime, we know which activities a workflow instance will be invoking. At this stage, we need to construct a QoS activity model from each activity of the workflow. Since this phase is independent of the previous one, in practice it can be carried out before path mining.

Since we have fourteen activities in the workflow illustrated in Figure 4, we need to construct fourteen QoS activity models. Each model is constructed using a profiling methodology (profiling was described in section 3.2). When carrying out activity profiling we determine the time an activity will take to be executed (i.e. Activity Response Time (ART)) and its cost (i.e. Activity cost (AC)). Table 4 illustrates the QoS activity model constructed for the Check Home Loan activity in Figure 4 using profiling.

Table 4. QoS activity model for the Check Home Loan activity

	Static QoS model		
	Min value	Avg value	Max value
Time (min)	123	154	189
Cost (euros)	4,80	5,15	5,70

This static QoS activity model was constructed using activity profiling. When a sufficient number of workflows have been executed and the log has a considerable amount of data, we re-compute the static QoS activity at runtime, originating a

dynamic QoS activity model. The re-computation is done based on the functions presented in Table 2. Due to space limitations we do not show the dynamic QoS activity model. It has exactly the same structure as the model presented in Table 4, but with more accurate values since they reflect the execution of activities in the context of several possible workflows.

4.4 Workflow QoS Estimation

As we have already mentioned, to compute the estimated QoS of a workflow in execution, we use a variation of the Stochastic Workflow Reduction (SWR) algorithm. The SWR aggregates the QoS activity models of each activity step-by-step. At each step a reduction rule is applied to transform and shrink the process and the time and cost of the activities involved is computed. This is continued until only one activity is left in the process. When this state is reached, the remaining activity contains the QoS metrics corresponding to the workflow under analysis. A graphical simulation of applying the SWR algorithm to our workflow scenario is illustrated in Figure 6.

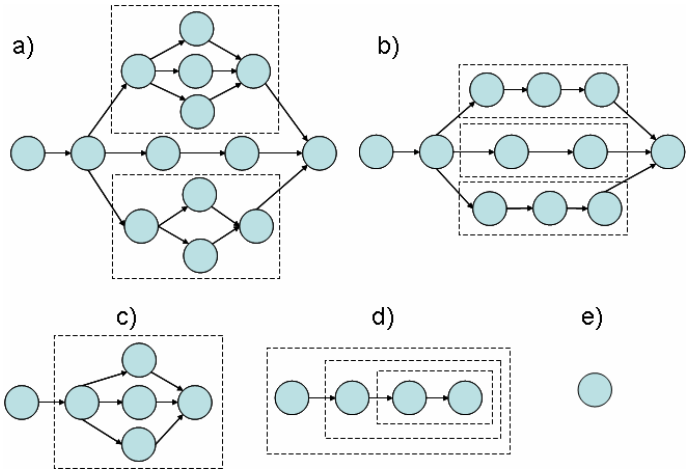

Fig. 6. SWR algorithm applied to our workflow example

The initial workflow (a) is transformed to originate workflow b) by applying the conditional reduction rule to two conditional structures identified in the figure with a box (dashed line). Workflow b) is further reduced by applying the sequential reduction rule to three sequential structures also identified with a box (dashed line). The resulting workflow, workflow c), is transformed several times to obtain workflow d) and, finally, workflow e). The final workflow (e) is composed of only one activity. Since at each transformation step SWR algorithm aggregates the QoS activity models involved in the transformation, the remaining activity contains the QoS metrics corresponding to the initial workflow under analysis.

4.5 QoS experimental results

Our experiments have been conducted in the following way. We have selected 100 random workflow instances from our log. For each instance, we have computed the real QoS (time and cost) associated with the instance. We have also computed the predicted QoS using our method. The results of QoS prediction for the loan process are illustrated in Figure 7.

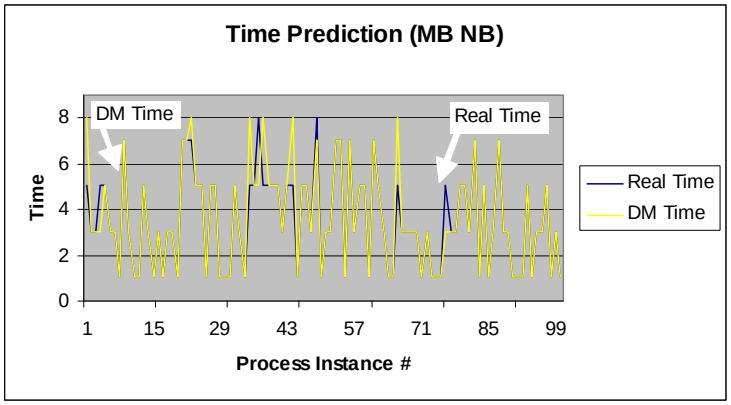

Fig. 7. QoS prediction for time

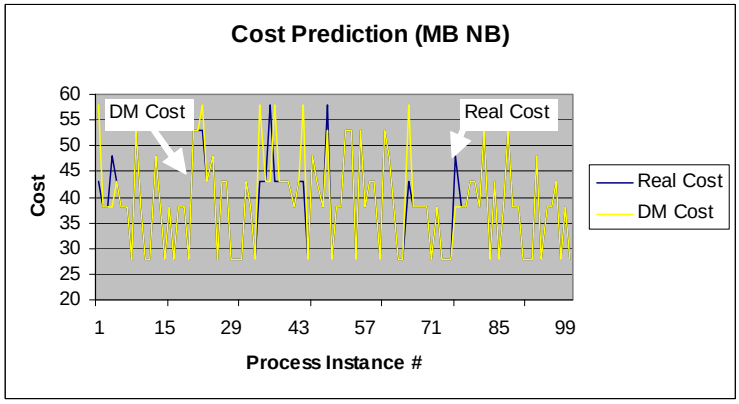

Fig. 8.. QoS prediction for cost

The results clearly show that the QoS mining method yields estimations that are very close to the real QoS of the running processes.

5 Related Work

Process and workflow mining is addressed in several papers and a detailed survey of this research area is provided in [14]. In [3, 4], a Business Process Intelligence (BPI) tool suite that uses data mining algorithms to support process execution by providing several features, such as analysis and prediction is presented. In [15] and [16] a machine learning component able to acquire and adapt a workflow model from observations of enacted workflow instances is described. Agrawal, Gunopulos et al. [17] propose an algorithm that allows the user to use existing workflow execution logs to automatically model a given business process presented as a graph. Chandrasekaran et al., [2] describe a simulation coupled with a Web Process Design Tool (WPDT) and a QoS model [1] to automatically simulate and analyze the QoS of Web processes. While the research on QoS for BMPS is limited, the research on time management, which is under the umbrella of QoS process, has been more active and productive. Eder et al. [18] and Pozewaunig et al. [19] present an extension of CMP and PERT frameworks by annotating workflow graphs with time, in order to check the validity of time constraints at process build-time.

6 Conclusions

The importance of QoS (Quality of Service) management for organizations and for workflow systems has already been much recognized by academia and industry. The design and execution of workflows cannot be undertaken while ignoring the importance of QoS measurements since they directly impact the success of organizations. In this paper we have shown a novel method that allows us to achieve high levels of accuracy when predicting the QoS of workflows. Our first conclusion indicates that workflow QoS mining should not be applied as a one-step methodology to workflow logs. Instead, if we use a methodology that includes path mining, QoS activity models, and workflow QoS estimation, we can obtain very good prediction accuracy. Our second conclusion indicates that the MultiBoost (MB) Naïve Bayes approach is the data mining algorithm that yields the best workflow QoS prediction results.

References

[1]. Cardoso, J., et al., *Modeling Quality of Service for workflows and web service processes.* Web Semantics: Science, Services and Agents on the World Wide Web Journal, 2004. **1**(3): p. 281-308.

[2]. Chandrasekaran, S., et al. *Service Technologies and their Synergy with Simulation*. in *Proceedings of the 2002 Winter Simulation Conference (WSC'02)*. 2002. San Diego, California. p. 606-615.

[3]. Grigori, D., et al., *Business Process Intelligence.* Computers in Industry, 2004. **53**: p. 321–343.

[4]. Grigori, D., et al. *Improving Business Process Quality through Exception Understanding, Prediction, and Prevention*. in *27th VLDB Conference*. 2001. Roma, Italy.

[5]. Cardoso, J. and A. Sheth. *Adaptation and Workflow Management Systems*. in *International Conference WWW/Internet 2005*. 2005. Lisbon, Portugal. p. 356-364.

[6]. Cardoso, J., *Path Mining in Web processes using Profiles*, in *Encyclopedia of Data Warehousing and*

Mining, J. Wang, Editor. 2005, Idea Group Inc. p. 896-901.
[7]. Cardoso, J. and M. Lenic, *Web Process and Workflow Path mining using the multimethod approach.* Journal of Business Intelligence and Data Mining (IJBIDM). submitted., 2005.
[8]. Musa, J.D., *Operational Profiles in Software-Reliability Engineering.* IEEE Software, 1993. **10**(2): p. 14-32.
[9]. Musa, J.D., *Software reliability engineering: more reliable software, faster development and testing*. 1999, New York: McGraw-Hill.
[10]. Aalst, W.M.P.v.d., et al., *Workflow patterns homepage*. 2002, http://tmitwww.tm.tue.nl/research/patterns.
[11]. Weka, *Weka*. 2004.
[12]. Platt, J., *Fast training of support vector machines using sequential minimal optimization*, in *Advances in Kernel Methods - Support Vector Learning*, B. Scholkopf, C.J.C. Burges, and A.J. Smola, Editors. 1999, MIT Press: Cambridge, MA. p. 185-208.
[13]. Webb , I.G., *MultiBoosting: A Technique for Combining Boosting and Wagging.* Machine Learning, 2000. **40**(2): p. 159-196.
[14]. Aalst, W.M.P.v.d., et al., *Workflow Mining: A Survey of Issues and Approaches.* Data & Knowledge Engineering (Elsevier), 2003. **47**(2): p. 237-267.
[15]. Herbst, J. and D. Karagiannis. *Integrating Machine Learning and Workflow Management to Support Acquisition and Adaption of Workflow Models*. in *Ninth International Workshop on Database and Expert Systems Applications*. 1998. p. 745-752.
[16]. Weijters, T. and W.M.P. van der Aalst. *Process Mining: Discovering Workflow Models from Event-Based Data.* in *13th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC 2001)*. 2001. Amsterdam, The Netherlands. p. 283-290.
[17]. Agrawal, R., D. Gunopulos, and F. Leymann. *Mining Process Models from Workflow Logs*. in *Sixth International Conference on Extending Database Technology*. 1998. Valencia, Spain: Springer. p. 469-483.
[18]. Eder, J., et al. *Time Management in Workflow Systems*. in *BIS'99 3rd International Conference on Business Information Systems*. 1999. Poznan, Poland: Springer Verlag. p. 265-280.
[19]. Pozewaunig, H., J. Eder, and W. Liebhart. *ePERT: Extending PERT for workflow management systems*. in *First European Symposium in Advances in Databases and Information Systems (ADBIS)*. 1997. St. Petersburg, Russia. p. 217-224.