An Empirical Comparison of Methods to support QoS-aware Service Selection

Bice Cavallo
Univ. of Naples, Federico II,
Univ. of Sannio, Italy
bicecava@unisannio.it

Massimiliano Di Penta,
Gerardo Canfora
Dept. of Engineering –
RCOST,
Univ.of Sannio, Italy
{dipenta,canfora}@unisannio.it

ABSTRACT

Run-time binding is an important and useful feature of Service Oriented Architectures (SOA), which aims at selecting, among functionally equivalent services, the ones that optimize some QoS objective of the overall application. To this aim, it is particularly relevant to forecast the QoS a service will likely exhibit in future invocations.

This paper presents an empirical study aimed at comparing different approaches for QoS forecasting, namely the use of average and current values, linear models, and models based on time series. The study is performed on QoS data obtained by monitoring the execution of 10 real services for 4 months.

Results show that, overall, the use of time series forecasting has the best compromise in ensuring a good prediction error, being sensible to outliers, and being able to predict likely violations of QoS constraints.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures—*Service-oriented architecture (SOA)*

General Terms

Performance, Experimentation

Keywords

Quality of Service, Service Composition and Binding, Time Series Forecasting, Empirical Study

1. INTRODUCTION

Ultra-late binding [5, 25] and discovery of new services [17, 22] are two relevant features of Service Oriented Architectures (SOA). Late binding is a mechanism which allows to bind a request coming from a service composition—hereby referred as *abstract service*—to one of the (possibly) multiple *concrete services* available and able to satisfy the specific request. For example, a flight booking request can be forwarded to services belonging to different airlines, or an e-book search request can be forwarded to services belonging to different e-libraries.

In this context, the choice of the service to invoke might depend on non-functional properties of the services, i.e., Quality of Service (QoS) attributes. According to Std. ISO 8402 [13] and ITU [14], QoS may be defined in terms of attributes such as price, response time, availability, and reputation. Also, it can be possible to define and use domain-specific properties, e.g., resolution or color depth for an image processing service [4]. In summary, QoS-aware binding means that, among functionally equivalent services, one can choose the cheapest one, the fastest, or the one that realizes a good compromise between time and cost.

QoS-awareness implies the enactment of different mechanisms, for which approaches and tools have been developed in recent and past years, such as:

1. monitoring mechanisms to collect QoS and functional information about service invocations, and to trigger recovery actions whenever needed [1, 20, 21];
2. approaches to estimate the QoS of a service composition given the QoS of services that participate in the composition [6, 15];
3. approaches to enact dynamic binding and to determine the (near) optimal set of bindings for a service composition [5, 25].

QoS-aware composition and binding is generally performed considering, for the candidate services, a likely QoS value that can be either the one declared by the service provider and exposed as part of the service description, or one obtained by performing a prediction based on past observed values.

The paper reports results from an empirical study aimed at comparing different approaches for QoS prediction. Specifically, the approaches being compared are: (i) the averaged value from past monitored QoS data; (ii) a linear model; (iii) the last observed—hereby referred as "current" QoS value; and (iv) the use of time series forecasting, specifically of Auto Regressive Integrated Moving Average (ARIMA) series, with and without smoothing the data being used to train the model.

The study has been performed upon QoS data—in this paper we restrict our attention to response time—collected

PESOS '10, May 1-2, 2010, Cape Town, South Africa

by invoking and monitoring 10 real services for for 4 months every hour. Results indicate that, although averaged value, last observed value and time series produce good QoS predictions, time series forecasting is less sensible to outliers than the current value, and differently from the averaged value it can also be used to predict likely Service Level Agreement (SLA) constraint violations.

The paper is organized as follows. Section 2 provides an overview of different QoS forecasting mechanisms, providing, in particular, details for the time series forecasting approach. Section 3 describes the empirical study. Results are reported in Section 4, while Section 5 discusses the study threats to validity. After an overview of the related literature in Section 6, Section 7 concludes the paper and outlines directions for future work.

2. APPROACHES FOR QoS FORECASTING

This section overviews the different approaches for QoS prediction this paper is aimed at comparing. For all the approaches, we consider a series of QoS values—related to any particular attribute X one is interested to use for QoS-aware selection—defined as:

$$\hat{X}(t_0), \hat{X}(t_1), \ldots, \hat{X}(t_k), \hat{X}(t_{k+1}), \ldots, \hat{X}(t_{k+n}) \quad (1)$$

where t_0 is the time when the first observation is available, t_k is the current time, and n is the number of future values one wants to predict. In this paper we focus on response time and consider predictions at 1-step ahead—i.e., how the service will respond one hour from now.

For most of the approaches, before starting to make predictions, it is needed to have enough training data points available to build the prediction model. For our experiments, we consider as initial training set the first 500 values of the time series. Once n predictions have been performed, the training set is augmented with the next n actual values of the time series, the model is estimated again, and the subsequent n values are then predicted.

2.1 Average of past values

With this model, the forecasting of n-steps ahead at time t_k is based on the average of the previously observed $k+1$ values:

$$X(t_{k+n}) = \sum_{i=0}^{k} \frac{\hat{X}(t_i)}{k+1} \quad (2)$$

where $X(t_{k+n})$ is the predicted value at time t_{k+n}, while $\hat{X}(t_i)$ is the observed value at time t_i.

2.2 Current value

This prediction model is very simple, as it just uses the last observed QoS value:

$$X(t_{k+n}) = \hat{X}(t_k) \quad (3)$$

2.3 Linear model

This approach builds a linear regression model, using the minimum least squares method, over the previously observed $k+1$ values, and builds an equation

$$X(t_i) = a \cdot t_i + b \quad (4)$$

which can be used to predict future values of the time series.

2.4 Time Series Forecasting

In this section we provide a definition of ARIMA time series, and then explain how it can be used for forecasting. More details can be found in [2].

Time series observations can be considered as realizations of a *stochastic process*, i.e., a collection of random variables ordered in time and defined at a set of time points. In particular, a stochastic process is a purely random process (i.e., *white noise*) if it consists of a sequence of random variables $\{Z_t\}$ which are mutually independent and identically distributed. By definition, it follows that purely random processes have constant mean and variance and thus a *white noise* process has a zero mean and a constant variance σ_Z^2.

Let $\{Z_t\}$ be a *white noise*, a process $\{X_t\}$ is said to be an autoregressive process of order p ($AR(p)$) if:

$$X_t = \alpha_1 X_{t-1} + \ldots + \alpha_p X_{t-p} + Z_t \quad (5)$$

where $\{\alpha_i\}$ are constants. This is similar to a multiple regression model, but $\{X_t\}$ is not regressed on independent variables but on past variables of $\{X_t\}$.

Suppose that $\{Z_t\}$ is a discrete purely random process with mean zero and variance σ_Z^2, then a process $\{X_t\}$ is said to be a moving average process of order q (MA(q)) if

$$X_t = Z_t + \beta_1 Z_{t-1} + \ldots + \beta_q Z_{t-q} \quad (6)$$

where $\{\beta_i\}$ are constants. Once the backward shift operator B is defined as

$$B^j X_t = X_{t-j} \quad (7)$$

a moving average process can be expressed as:

$$X_t = \Theta(B) Z_t \quad (8)$$

where

$$\Theta(B) = 1 + \beta_1 B + \ldots + \beta_q B^q \quad (9)$$

Broadly speaking, a $MA(q)$ explains the present as the mixture of q random impulses, while an $AR(p)$ process builds the present in terms of the past p events. A useful class of models for time series is obtained by combining MA of order p and AR processes of order q into a mixed autoregressive moving-average (ARMA) process of order (p, q), defined as follows:

$$X_t = \alpha_1 X_{t-1} + \ldots + \alpha_p X_{t-p} + Z_t + \beta_1 Z_{t-1} + \ldots + \beta_q Z_{t-q} \quad (10)$$

where $X(t)$ is the original series and $Z(t)$ is a series of random pulses which are assumed to follow the normal probability distribution.

A very important assumption in time series analysis is stationarity. A time series is said to be strictly stationary if the time series has its statistical properties (e.g., mean, variance, autocorrelation) constant over the time.

Box and Jenkins introduced a generalization of ARMA processes to deal with the modeling of non-stationary time series [2]. In particular, if in equation (10) X_t is replaced by $W_t = \nabla^d X_t$, it is possible to describe certain types of non-stationarity time series. Such a model is called ARIMA (Auto Regressive Integrated Moving Average) because the model is fitted to the time series W_t that has to be summed (i.e., integrated) to provide a model for the non-stationary

data. In other words, applying term-by-term differencing d times to an $ARIMA(p,d,q)$ process gives a stationary ARMA(p,q) process. Writing

$$W_t = \nabla^d X_t = (1-B)^d X_t \quad (11)$$

the general process ARIMA(p,d,q) is of the form

$$W_t = \alpha_1 W_{t-1} + \ldots + \alpha_p W_{t-p} + Z_t + \ldots + \beta_q Z_{t-q} \quad (12)$$

2.4.1 Time Series Forecasting

A wide variety of time series prediction procedures and diagnostic tests are available [3, 7, 11, 24]. This paper relies on the Box and Jenkins approach [2], which entails the steps described below:

1. *Identifying the presence of trends.* As described above, ARMA time series requires the time series to be stationary. If this is not the case, an ARIMA time series need to be used instead. To this aim, we use the Augmented Dickey-Fuller test (ADF) to check whether the time series contains a trend. Specifically, the following null hypothesis is tested H_0 *the time series is not stationary.*

2. *Identifying seasonal (periodic) component of the series.* This is done by analyzing the spectral decomposition of the time series. If the spectrum contains frequency peaks, we consider the highest peak as frequency of the time series.

3. *Model identification.* The observed time series is analyzed to select an ARIMA(p,d,q) process that appears to be the most appropriate; this requires the identification of the p,d,q parameters. While d is selected by differenentiating the series d times until a stationary series (according to the ADF test) is obtained, the p and q parameters are estimated using an iterative procedure—based on the *Akaike Information Criterion (AIC)*—aimed at determining the model that better fits the data used to train it.

4. *Estimation.* The actual time series is modeled using the ARIMA(p,d,q) process previously defined; this requires the estimation—using again the AIC criterion—of the $\{\alpha_i\}$ and $\{\beta_j\}$ coefficients defined by the equations (12).

2.5 Smoothed time series

To avoid the training data set being polluted by outliers and noise, we smooth it using a kernel smoothing function. A kernel smoother is a statistical technique that allows to estimate a real function from its observations. Given a random sample $X_1, \ldots, X_n$, a kernel density estimator is defined as:

$$k(x,h) = \frac{1}{nh} \sum_{i=1}^{n} D\left(\frac{||x - X_i||}{h}\right) \quad (13)$$

where $||\cdot||$ is the Euclidean norm, D is the kernel function, and h the kernel bandwidth. D is a probability distribution function, often unimodal and symmetric around the origin. Examples of kernel functions are the uniform distribution, the Gaussian distribution, or the Triangular distribution.

Table 1: Response time for S_1 and S_2

t	$X[ms]$ S_1	$X[ms]$ S_2	t	$X[ms]$ S_1	$X[ms]$ S_2
1	1000	2100	16	900	1000
2	1500	2100	17	1500	1000
3	2000	1500	18	2100	1400
4	1700	1500	19	1500	1400
5	1400	1500	20	1000	1400
6	900	1500	21	1000	1500
7	1500	1500	22	1500	1500
8	2100	1500	23	2000	1500
9	1500	900	24	1700	1700
10	1000	900	25	1400	1700
11	1000	900	26	900	1700
12	1500	1000	27	1500	2000
13	2000	1000	28	2100	2000
14	1700	1000	29	1500	2000
15	1400	1000	30	1000	2100

Also, simple examples of smoothers are the nearest neighbor smoother, where a value of the smoothed function is estimated as the average of the neighbors, or the local (linear or polynomial) regression, where segments of the function are approximated using a line instead of a constant value (used by the nearest neighbor smoother). The bandwidth is a scaling factor, which controls the level of smoothness of the density estimate. Higher h values make the resulting function smoother, while lower values make it rougher. Clearly, choosing the right bandwidth is crucial to avoid under-smoothing or over-smoothing.

In this paper, smoothing is performed by applying a Daniels [9] kernel smoothing function having a bandwidth of 5, calibrated through a trial-and-error procedure. This introduces in the smoothed data a lag of $2 \cdot 5 = 10$ points, i.e., the first 10 points of the series are not smoothed, as they are used to estimate the smoothing function parameters.

2.6 Why using Time Series for QoS-aware Service Selection?

After having described the various QoS prediction approaches, this section describes the claimed advantages of time series forecasting in this context. Let S_1 and S_2 be two functionally equivalent services, exhibiting the response time X, collected once every 24 hours for 30 days—shown in Table 1. Let us assume a response time constraint $X \leq 1600$. The goal of a QoS-aware composition middleware would be to select the service that, with higher likelihood, will meet such a QoS constraint in future invocations. Both S_1 and S_2 are able to meet the constraint $\hat{X} \leq 1600$ in 70% of the cases, and the minimum, maximum and average response times are 900, 2100 and 1460 for both S_1 or S_2. Thus, simple descriptive statistics would not allow us to determine which service is better. Moreover, using the average value (1460) as a predictor would never allow for forecasting likely QoS constraint violations.

By applying the Box and Jenkins approach described in Section 2.4.1, we obtain that the spectrum of the time series of S_1 contains a frequency peak at 0.2—i.e., the series has a period every $1/0.2 = 5$ days—and that ARIMA(2,0,1) and ARIMA(1,0,0) are good models for the time series of S_1 and S_2 respectively. Thus, we could forecast the future value of X for S_1 and S_2, by means of these ARIMA models, and finally, we select the service between S_1 and S_2 with higher probability to meet the QoS constraint.

Other than the capability of prediction constraint violation, time series have other advantages with respect to sim-

Table 2: Characteristics of the monitored services

Service	Description	URI	R. time constr. [s]	
			C1	C2
Amazon	Searches for books in the online book store	http://www.xmlme.com/WSAmazonBox.asmx	5.6	5.8
Google	Searches into the Google search engine	http://api.google.com/search/beta2	3.5	4.5
BLiquidity	Provides information on liquidity in a banking system	http://webservices.lb.lt/BLiquidity/BLiquidity.asmx	2.1	2.3
Currency Converter	Performs a currency conversion using the current quotation	http://www.webservicex.com/CurrencyConvertor.asmx	4.6	5.0
Stock Quote	Reports quotations of stocks	http://www.swanandmokashi.com/HomePage/WebServices/StockQuotes.asmx	2.0	2.2
Fast Weather	Reports weather info for a given city	http://ws2.serviceobjects.net/fw/FastWeather.asmx	3.0	3.5
Quote of the Day	Reports a random quote every day	http://www.swanandmokashi.com/HomePage/WebServices/QuoteOfTheDay.asmx	3.6	4.0
GetJoke	Outputs a random joke	http://www.interpressfact.net/webservices/getJoke.asmx	2.2	2.3
Hiperlink Extractor	Extracts hyperlinks from web pages	http://www.atomic-x.com/xmlservices/HyperlinkExtractor.asmx	3.4	3.6
XML Daily Fact	Returns a daily fact with an emphasis on XML Web Services and the use of XML within the Microsoft .NET Framework	http://www.xmlme.com/WSDailyXml.asmx	1.8	2.0

ple averaged models. In fact, for simple models, either one has to build a model with infinite memory (i.e., averaging all the available values), or has to determine in an ad-hoc manner the number of previous values on which the average should be computed. Instead, time series forecasting is based on a model which order is determined by means of a well-defined calibration method.

3. EMPIRICAL STUDY

The *goal* of this study is to compare the capabilities of different approaches for QoS prediction in the context of dynamic, QoS-aware service selection for SOA. The *focus* is on selecting the services that provide the best QoS, while avoiding SLA violations. The *perspective* is of researchers interested to investigate the applicability of time series forecasting to deal with data obtained from service monitoring, as well as of service integrators wanting to develop better QoS-aware service composition/selection middlewares. The *context* consists of monitored QoS data collected by invoking 10 real services every hour for about four months. The list of services used for the study is reported in the three left-most side columns of Table 2[1].

Specifically, we collected data by sending an invocation to operations of these services and then recording (i) whether the service was available or not, (ii) the SOAP output message, (iii) the observed response time, (iv), the throughput, as the size of the SOAP message divided by the response time, (iv) whether or not the service generated an exception. In this paper, we limit our investigation to the response time. Before being used for the study, we pruned out from the time series outliers, i.e., values above $Q3 + 1.5 \cdot IQR$, where $Q3$ is the third quartile (75% percentile) of the response time distribution and IQR the interquartile range.

The research questions this study aims at addressing are the following:

- **RQ1:** *What is the prediction error produced by the different approaches?*
- **RQ2:** *To what extant the approaches can be used to forecast QoS violations?*

All the analyses of this study have been performed using the R statistical environment[2], and specifically the R *arima* model for time series forecasting, and the *kernel* and *kernapply* functions for kernel smoothing.

To address **RQ1**, for each prediction performed, we compute the relative error defined as:

$$e(t_i) = \frac{|x(t_i) - \hat{x}(t_i)|}{\hat{x}(t_i)} \tag{14}$$

and we analyze it using boxplots and descriptive statistics. We also test whether the various methods perform differently using the non-parametric Kruskal-Wallis test. To perform a pair-wise comparison of pairs of approaches, we use the (non-parametric) Mann-Whitney test. In this case, however, to avoid biasing, we apply the Bonferroni correction, i.e., we consider as significant p-values less or equal to 0.05 divided by the number of possible pairwise comparisons. Other than testing the presence of a significant difference among the different methods, it is of practical interest to estimate the magnitude of such a difference. To this aim, we use the Cohen d effect size [8], which indicates the magnitude of a main factor treatment effect on the dependent variables (the effect size is considered small for $0.2 \leq d < 0.5$, medium for $0.5 \leq d < 0.8$ and large for $d \geq 0.8$). For independent samples, it is defined as the difference between the means (M_1 and M_2), divided by the pooled standard deviation ($\sigma = \sqrt{(\sigma_1^2 + \sigma_2^2)/2}$) of both groups: $d = (M_1 - M_2)/\sigma$.

As we mentioned above, predictions have been performed considering response times sampled every hour. However, we also did the experiments by sampling every 24 hours—that we omit due to the limited space—and results were consistent with the ones with samples collected every hour.

To address **RQ2**, we check, for two different constraints—reported in the right-most side columns of Table 2—whether the predicted QoS value violates or not the constraint. Then,

[1]Some of these services might not be available anymore, or the URL might be outdated.

[2]http://www.r-project.org

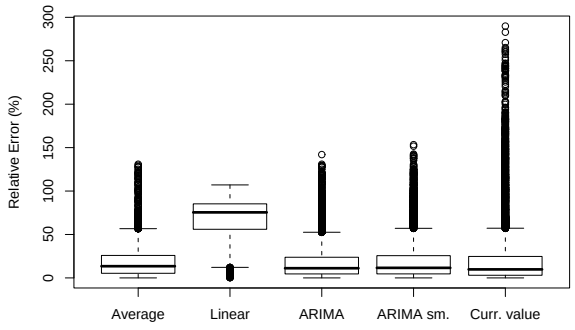

Figure 1: Boxplots of relative errors for one-step ahead prediction

we compute the percentage of correctly predicted QoS violations. Clearly, a model constantly predicting a violation would achieve a prediction rate of 100%, however such a model would not be able, at the same time, to exhibit a low prediction error.

4. RESULTS

This section reports results of our empirical study to answer the research questions formulated in Section 3. Data for verification/replication are available on-line[3].

4.1 RQ1: What is the prediction error produced by the different approaches?

Before applying time series forecasting approaches, it is needed to perform a stationarity test. As explained in Section 2, this is done using the ADF test. The test indicates that all time series are stationary (p-value < 0.01 in all cases), thus ARIMA $(p, 0, q)$ (ARMA) models can be used.

Boxplots of relative prediction errors (1-step ahead) are shown in Figure 1, while mean and standard deviation of prediction errors are reported in Table 3 for each service separately. The boxplots highlight that different approaches exhibit significantly different prediction errors, as also confirmed by the Kruskal-Wallis test (p-value$<$0.01). The boxplots clearly show that the linear model produces a prediction error significantly higher than other methods (p-value$<$ 0.01, according to the Mann-Whitney test), with a high effect size $d > 2$. The error exhibited by the ARIMA model is significantly lower than those of current value and average value (p-value < 0.01), although in this case, as it can also be seen from the figure, the effect size is small: $d = 0.14$ for the current value and $d = 0.11$ for the averaged value. Also, ARIMA performs significantly better without smoothing the time series (p-value$<$ 0.01) but, again, the effect size is small (d=0.10). Thus, differently from other applications, where smoothing turned out to be useful for time series forecasting [12], it is not the case here. Finally, Figure 1 shows that the current value model produces a high number of outliers, exhibiting a very high prediction error, as confirmed by the high standard deviation reported in Table 3.

[3]http://www.rcost.unisannio.it/mdipenta/QoS-tseries-data.tgz

In conclusion, by looking at the prediction error only, we can say that the average value model and the ARIMA (without smoothing) model outperform other models, while smoothing does not introduce any advantage; the current value model, while being good in terms of 1-step prediction errors, goes worse for long-term prediction errors and exhibits a large variation as well as a high number of outliers.

4.2 RQ2: To what extant the approaches can be used to forecast SLA violations?

RQ1 has indicated that, overall, a QoS prediction model using the simple average value of past observations could be as good as using a more complex ARIMA model. However, a good prediction model should not only be able to allow for selecting the service exhibiting the best QoS, but also to forecast possible SLA violations. Let us consider the two response time constraints shown in the right-side of Table 2, and let us compute the percentage of cases in which the constraint violation was predicted. Results shown in Table 4 indicate that the most useful model for this purpose is the current value, although ARIMA is still able to perform a reasonably good number of predictions. In fact, for constraint $C1$ the average ratio of correct predictions is 19%, 22%, and 37% for ARIMA, ARIMA smoothed, and current value respectively, with a significantly higher percentage of correct predictions for the current value than for ARIMA (p-value=0.04), while there is no significant difference between ARIMA smoothed and current value (p-value=0.1) For constraint $C2$, the average ratio of correct predictions is 19%, 18% and 30% for ARIMA, ARIMA smoothed, and current value respectively, with a significantly higher ratio of correctly predicted violations for the current value model than for ARIMA smoothed (p-value < 0.01), and than for ARIMA (p-value=0.02). The reason why the current value outperforms ARIMA is, in our understanding, due to the relatively low capability ARIMA has to predict values strongly deviating from the average, and thus violations. The table does not show columns for the average and linear model, as they were not able to predict violations in any case, for the reasons explained in Section 2.6. In general, however, results show that guaranteeing a good prediction of SLA violations is still a challenging issue, thus further models able to better deal with this issue should be investigated.

5. THREATS TO VALIDITY

This section discusses the main threats to the validity of our study.

Threats to *construct validity* depend on how QoS was measured. This paper considers response time measures performed from the service consumer's side, thus can be affected by load in the local network or on the Internet. However, this still reflects a realistic situation, as the response time measured is the one perceived by the service consumer, and thus the one on which the service choice should be based (e.g., a service that can be reached through a network path with higher bandwidth should be preferred).

Threats to *conclusion validity* concern the statistics being used in this study. We mainly used non-parametric tests to address the study research questions, as they do not impose constraints on the data set distribution.

Threats to *external validity* concern the generalization of results. Although we considered a large number of services belonging to different domains and made available by dif-

Table 3: Average and standard deviation of relative prediction error for 1-step ahead prediction

Service	Average		Linear		ARIMA		ARIMA sm.		Curr. value	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
BLiquidity	22.29	12.64	51.43	24.59	21.94	13.29	22.59	15.90	26.91	33.40
FastWeather	33.68	16.71	78.58	7.70	32.89	17.04	32.86	20.14	49.20	49.84
Google	20.84	13.48	68.83	14.38	20.14	13.53	20.48	16.04	24.30	27.55
QuoteOfTheDay	41.09	22.20	72.62	15.48	29.97	23.59	42.48	29.60	29.03	43.87
XMLDailyFact	9.57	6.43	57.63	18.84	8.98	6.60	9.69	7.00	9.92	10.98
Amazon	11.07	14.08	85.15	6.39	10.99	14.19	11.43	14.30	15.46	17.44
CurrencyConverter	9.38	18.33	84.86	5.91	9.38	18.33	9.38	18.34	12.75	20.26
GetJoke	15.70	8.59	59.56	14.47	11.13	8.26	11.31	9.14	12.59	13.74
StockQuotes	6.41	5.74	69.46	21.84	5.62	5.62	6.05	6.08	6.41	8.52
HyperlinkExtractor	15.82	11.83	78.73	11.22	16.98	12.23	17.61	13.23	21.94	22.18

Table 4: Percentages of correctly predicted constraint violations

Service	Constraint 1			Constraint 2		
	ARIMA	ARIMA sm.	Curr. value	ARIMA	ARIMA sm.	Curr. value
BLiquidity	11	25	29	6	12	25
FastWeather	3	17	15	100	91	33
Google	0	0	20	0	0	15
QuoteOfTheDay	69	40	76	41	17	69
XMLDailyFact	7	7	31	0	1	18
Amazon	2	21	36	1	8	25
CurrencyConverter	0	0	44	0	0	21
GetJoke	21	20	34	8	8	30
StockQuotes	51	55	47	15	18	30
HyperlinkExtractor	30	35	38	17	21	34

ferent service providers, further experiments are desirable. Also, future studies should consider other QoS attributes than response time.

6. RELATED WORK

QoS-Aware service composition [5, 6, 18, 25, 26] aims at selecting component services to optimize the overall QoS and meet some constraints imposed by the SLA. This is often done using QoS aggregation formulae [6] and optimization techniques [5, 25]. In the aforementioned approaches, it is assumed the availability of pre-existing QoS information of component services, thus QoS-aware service selection and composition is based on these declared values.

In recent years, several papers have investigated approaches for the QoS forecasting and the prediction of SLA violations. Leitner *et al.* [16] proposed an approach for predicting SLA violations at runtime by means of machine learning regression techniques. Vu *et al.* [23] proposed a QoS-aware semantic service selection and ranking solution that exploit a trust and reputation management method to to predict the future quality of a service. Finally, Gao and Wu [10] proposed the application of ANN (Artificial Neural Networks) to predict service performance.

In the context of QoS management, Nobile *et al.* [19] proposed an architecture of QoS proxy for RT-RPCs (Real-Time Remote Procedure Calls) that uses the Box-Jenkins [2] time series forecasting procedure to predict future traffic characteristics of RT-RPCs that pass through the proxy. Zhu *et al.* [28] used a linear prediction to forecast the performance of parallel/distributed discrete event simulation (PDES). Zeng *et al.* [27] adopted a time series forecasting algorithm, in particular an exponential smoothing method, to forecast the values of the Key Performance Indicators (KPIs). In our work, however, we found that smoothing did not help to perform better forecasting, and that, in practice, seasonal phenomena could not be observed on real response time data collected from various services.

Indeed, some of the aforementioned pieces of work have identified the possibility of using various prediction approaches, including time series forecasting [19, 27]. With respect to the existing state-of-the-art, however, the present work, in the authors' knowledge, is the first that reports an empirical evaluation and comparison—upon time series of real observations made in four months—of various forecasting approaches, and discussing pros and cons of various approaches for different purposes, i.e., obtaining a low prediction errors and being capable of predicting SLA violations.

7. CONCLUSION AND WORK-IN-PROGRESS

This paper reported an empirical study aimed at comparing different QoS prediction models on time series of response time collected by monitoring invocations of 10 services for 4 months. Results of the study indicate that ARIMA time series forecasting constitutes a good compromise in terms of achieving a low relative prediction error and being capable to predict SLA violations. Simpler models such as the average of the previously observed QoS also exhibit a low prediction error, but cannot be used to predict SLA violations. Models based on the current value, instead, exhibit high prediction variability, while the usage of smoothing does not help to improve the predictions.

The study also shows limits of time series forecasting in the context of QoS prediction, limits not always highlighted by previous works in the field:

- the benefit in terms of prediction error introduced by time series forecasting with respect to simpler model is significant though limited in terms of magnitude;
- the presence of seasonal phenomena is, in practice, rarely observed;

- predicting SLA violations is still challenging. In general, when the time series exhibit outliers, it is quite hard to perform accurate predictions, although the ARIMA time series forecasting is more robust to that phenomenon.

Work-in-progress aims at investigating other forecasting models, such as non-linear time series which could be used to better predict SLA violations, or models accounting for periodicity of the time series (e.g., response times decrease during weekends or increase during peak hours), which was not observed in the current data set.

8. REFERENCES

[1] L. Baresi, C. Ghezzi, and S. Guinea. Smart Monitors for Composed Services. In *Proc. 2nd International Conference on Service Oriented Computing (ICSOC'04)*, pages 193–202, New York, USA, 2004.

[2] G. Box and M. Jenkins. *Time Series Forecasting Analysis and Control*. Holden Day, San Francisco (USA), 1970.

[3] R. G. Brown. *Smoothing, Forecasting and Prediction*. Prentice Hall, 1963.

[4] G. Canfora, M. Di Penta, R. Esposito, F. Perfetto, and M. L. Villani. Service composition (re)binding driven by application-specific QoS. In *Proc. of Service-Oriented Computing - ICSOC 2006, 4th International Conference, Chicago, IL, USA, December 4-7, 2006*, volume 4294, pages 141–152. LNCS, Springer, 2006.

[5] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani. A framework for QoS-aware binding and re-binding of composite web services. *Journal of Systems and Software*, 81(10):1754–1769, 2008.

[6] J. Cardoso, A. P. Sheth, J. A. Miller, J. Arnold, and K. J. Kochut. Modeling quality of service for workflows and Web service processes. *Web Semantics Journal: Science, Services and Agents on the World Wide Web Journal*, 1(3):281–308, 2004.

[7] C. Chatfield. *The Analysis of the Time Series*. Chapman & Hall Rc, 1996.

[8] J. Cohen. *Statistical power analysis for the behavioral sciences (2nd ed.)*. Lawrence Earlbaum Associates, Hillsdale, NJ, 1988.

[9] H. Daniels. The estimation of spectral densities. *J. Roy. Statist. Soc*, (24):185–198, 1962.

[10] Z. Gao and G. Wu. Combining QoS-based service selection with performance prediction. In *IEEE International Conference on e-Business Engineering*, 2005.

[11] A. Harvey. *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge University Press, 1989.

[12] I. Herraiz, J. M. González-Barahona, and G. Robles. Forecasting the number of changes in Eclipse using time series analysis. In *Fourth International Workshop on Mining Software Repositories, MSR 2007, Minneapolis, MN, USA, May 19-20, 2007, Proceedings*, page 32, 2007.

[13] ISO. *UNI EN ISO 8402 (Part of the ISO 9000 2002): Quality Vocabulary*. ISO, 2002.

[14] ITU. *Recommendation E.800 Quality of service and dependability vocabulary*. ITU, 1994.

[15] M. Jaeger, G. Rojec-Goldmann, and G. Mühl. QoS Aggregation in Web Service Compositions. In *Proc. of the IEEE International Conference on e-Technology, e-Commerce and e-Services (EEE 2005)*, pages 181–185, Honk-Kong, China, Mar. 2005. IEEE.

[16] P. Leitner, B. Wetzstein, F. Rosenberg, A. Michlmayr, S. Dustdar, and F. Leymann. Runtime prediction of service level agreement violations for composite services. In *Proceedings of the 3rd Workshop on Non-Functional Properties and SLA Management in Service-Oriented Computing (NFPSLAM-SOC'09), November 23, Stockholm, Sweden*, 2009.

[17] S. B. Mokhtar, D. Preuveneers, N. Georgantas, V. Issarny, and Y. Berbers. EASY: Efficient semAntic Service discoverY in pervasive computing environments with QoS and context support. *Journal of Systems and Software*, 81(5):785–808, 2008.

[18] X. T. Nguyen, R. Kowalczyk, and J. Han. Using dynamic asynchronous aggregate search for quality guarantees of multiple web services compositions. In *ICSOC, LNCS Springer*, volume 4294, 2003.

[19] P. N. Nobile, R. R. F. Lopes, C. E. Moron, and L. C. Trevelin. QoS proxy architecture for real time RPC with traffic prediction. In *11th IEEE Symposium on Distributed Simulation and Real-Time Applications*, 2007.

[20] M. Pistore and P. Traverso. Assumption-based composition and monitoring of web services. In *Test and Analysis of Web Services*, pages 307–335. Springer, 2007.

[21] G. Spanoudakis and K. Mahbub. Non-intrusive monitoring of service-based systems. *Int. J. Cooperative Inf. Syst.*, 15(3):325–358, 2006.

[22] G. Spanoudakis, A. Zisman, and A. Kozlenkov. A service discovery framework for service centric systems. In *2005 IEEE International Conference on Services Computing (SCC 2005), 11-15 July 2005, Orlando, FL, USA*, pages 251–259, 2005.

[23] L.-H. Vu, M. Hauswirth, and K. Aberer. QoS-based service selection and ranking with trust and reputation management. http://dip.semanticweb.org/documents/manfred-hauswirth-coopis-semanticdiscovery.pdf. Last access: 11 January 2010.

[24] M. West and P. Harrison. *Bayesian Forecasting and Dynamic Models*. Springer-Verlag, 1989.

[25] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. QoS-aware middleware for web services composition. *IEEE Transactions on Software Engineering*, 30(5):311–327, 2004.

[26] L. Zeng, H. Lei, J.-J. Jeng, J.-Y. Chung, and B. Benatallah. Policy-driven exception-management for composite web services. In *IEEE CEC*, 2005.

[27] L. Zeng, C. Lingenfelder, H. Lei, and H. Chang. Event-driven quality of service prediction. In *ICSOC, LNCS Springer, Heidelberg*, volume 5364, 2008.

[28] S. Zhu, Z. Du, Y. Chen, X. Chai, and B. Li. QoS enhancement for PDES grid based on time series prediction. In *The Sixth International Conference on Grid and Cooperative Computing*, 2007.