International Journal on Future Revolution in Computer Science & Communication Engineering
Volume: 3 Issue: 12
ISSN: 2454-4248
388 – 391

Execution Time Prediction for a Web Service Instance

A.Vani Vathsala
Dept of Computer Science and Engineering
CVR College of Engineering
Hyderabad, India
atluriv@yahoo.com

***Abstract*—** Availability of services on Internet has provided unique opportunity to customers as well as providers for conducting e-business. This new business paradigm can succeed provided selection of services is accomplished to customer satisfaction in terms of service delivery time as well as service quality. Instead of leaving it to service provider to declare, we propose a strategy for forecasting execution time of the web service being called. The paper deals with model details and proposes a framework for implementation of the model. We also demonstrate its usability in scenarios like checkpointing web services.

Keywords- *web service, instance, prediction, execution time*

*****

I. INTRODUCTION

The role of web services in today's world is continuously growing. This is not only due to the use of web services by organizations having distributed operations but also for its operational ease. Selection of a web service by consumers is typically guided by two factors: quality of service and timeliness. In this work we investigate the second factor to predict execution time of a web service.

The factors that mostly contribute to service execution time include i) message processing time due to different sizes of request and response messages. ii) database query processing time iii) waiting time at the web server due to multiple requests to the same server at the same time. The role of network delay is not considered here, since it contributes to response time but not to service execution time. It has been investigated in our work [7]. In a nutshell, we investigate in this paper, the factors that contribute to service execution time.

We propose a model that takes into account all the above mentioned factors and estimates the execution time of a given web service instance. We also propose a framework to implement the proposed model.

Details of the proposed model are provided in Section III. The remainder of the paper is organized as follows: Section II presents related work; section IV presents our framework to predict the execution of a web service, Section V provides an application of the predicted execution time, where in we demonstrate the usage of predicted execution time in Checkpointing Web services, and Section VI presents our future work and concludes the paper.

II. RELATED WORK

Prediction of response time and QOS attributes of web services has been studied using different techniques [3,4,5]. Zhang Li et al[3] propose to employ similarity mining and prediction from users' experience.Ch Ram Mohan Reddy et al[4] model web services using Unified Modeling Language, Use Case Diagram, Sequence Diagram, Deployment Diagram and identify the bottleneck resources and there by predict the performance metrics. Moreno Marzolla et al[5] propose an approach for performance assessment of Web Service workflows based on the involved BPEL constructs. For predicting web service performance, all these research works do not consider the factors that lead to increased waiting times and processing times at the server side.

Hence in this paper we concentrate on factors which affect the execution time of a web service at a given instance of time which is very much required for accurate prediction.

III. PREDICTING EXECUTION TIME OF A WEB SERVICE INSTANCE

We propose to predict the execution time of a web service instance by first identifying the factors that influence the execution time of the web service instance. Then we propose a strategy that uses these factors and predicts the execution time of the given web service instance.

A. Factors affecting the Execution time of a web service instance

The Service execution time e^s_t of an instance S invoked at time t can be split into the following four factors:

$$e^s_t = m^s_t + w^s_t + drpt^s_t + E$$

where

m^s_t = Message processing time

w^s_t = Waiting time

$drpt^s_t$ = Data base request processing time

E = Time taken to execute its lines of code

 ISSN: 2454-4248
388 – 391

Message Processing Time: A web service request message is a SOAP message in which the input parameters and the operation requested are encoded in XML. The SOAP messages go through serialization and deserialisation when they are sent and received over the internet. Hence the factors contributing to message processing time include time required for serialization, m_s and deserialisation, m_{ds}. Thus the message processing time m^s_t can be defined as follows:

$$m^s_t = m_{ds} * c_1 + m_s * c_2 \qquad where$$

m_s = Serialization time taken for a message of size *a* bytes.

m_{ds}= Deserialization time for a message of size *a* bytes.

Size of Request message = $D_1 = c_1 * a$

Size of Response message = $D_2 = c_2 * a$

Waiting Time: A service may have several instances at a given time, coming from different customers. No of instances of the same web service that are running at time *t* when the current request is made also affect the execution time of the web service instance. Such parallel requests increase the scheduling overhead and also waiting time in the queues and hence we estimate such waiting time for a service request.

In practice waiting time of a service is linked to the time period of a day, since different number of requests appear at different periods of a day. For example in usual business hours, there would be several parallel requests but later the number decreases. Considering the patterns in service requests we have identified three different periods of a day; Peak time, Normal time, and off-peak time. The time at which the service call is made determines the time slot to which the call belongs.

For each category of time period average waiting time is estimated from service execution log maintained at corresponding UDDI. Table I shows an example scenario.

TABLE I
WAITING TIME VALUES

Time slot	Average Waiting time
Peak Time	20 secs
Normal Time	8 secs
Off-peak Time	2 secs

Let d_1 indicate the average waiting time in the peak time (20 secs in the above example). Similarly let d_2,d_3 indicate the average waiting time in normal time and off-peak time respectively. Then the waiting time, w^s_t, of the new request made at time *t* can be estimated as:

$$w^s_t = \begin{cases} d_1 \text{ if } t \in PeakTime \\ d_2 \text{ if } t \in NormalTime \\ d_3 \text{ if } t \in OffPeakTime \end{cases}$$

Database Access: "Does the current request result in time consuming database access?" is another important factor to be considered. Database request processing time for a web service *(drpt)* may be also be estimated similarly depending on the number of requests received by the database server. The database request processing time is divided in to three similar time slots/periods: peak time, normal time and off peak time. A DBMS utility service provides us with average request processing time for each of these time slots. Let r_1, r_2, r_3 represent the average request processing times in these three time slots respectively. Depending on the time *t* at which request is made, the average request processing time $drpt^s_t$ can be either r_1 *or* r_2 *or* r_3. If there are *n* such requests, the database request processing time = $n * drpt^s_t$

B. *Strategy for predicting the Execution time*

While estimating the execution time of a web service instance, we can associate weights with each of the above factors, in order to adjust the error that might be associated with their estimations. Let W_1, W_2, W_3 represent the weights that we would like to associate with the factors: Message processing time, waiting time due to multiple simultaneous instances, database request processing time respectively. Thus the estimation of execution time, e^s_t, of an instance of an atomic web service made at time t can be further refined as:

$$e^s_t = E + (W_1 * (m_{ds} * c_1 + m_s * c_2)) + (W_2 * w^s_t) + (W_3 * n * drpt^s_t)$$

From among the above values, the values which are same for every execution instance of the web service and hence the values that can be predetermined are given in Table II. All the remaining quantities which are listed in Table III change for every instance of service execution.

TABLE II
PREDETERMINABLE QUANTITIES USED IN ESTIMATION OF EXECUTION TIME

Quantity	Description
E	Execution Time of its lines of code
m_s	Serialisation time taken by message of size a bytes
mds	Deserialisation time taken by message of size a bytes
W1,W2, W3	Weights
d1, d2,d3	Average waiting times
r1, r2, r3	Average request processing times

TABLE III

 ISSN: 2454-4248
388 – 391

NON-PREDETERMINABLE QUANTITIES USED IN ESTIMATION OF EXECUTION TIME

Quantity	Description
$D_1 = c_1 * a$	Size of request message
$D_2 = c_2 * a$	Size of response message
w^s_t	Waiting time
$drpt^s_t$	Database request processing time
n	No of Database requests

IV. FRAMEWORK FOR EXECUTION TIME PREDICTION

Let S_1 be the web service whose execution time is to be predicted. Let U be the UDDI in which the web service S_1 publishes its WSDL. Each web service should maintain a web service monitor. At the end of execution of every instance of the web service, the monitor sends all the call details to the UDDI concerned as an XML message.

The UDDI has a ETPrediction service P which receives these messages and stores all these call details in its database. (Refer to Figure 1.

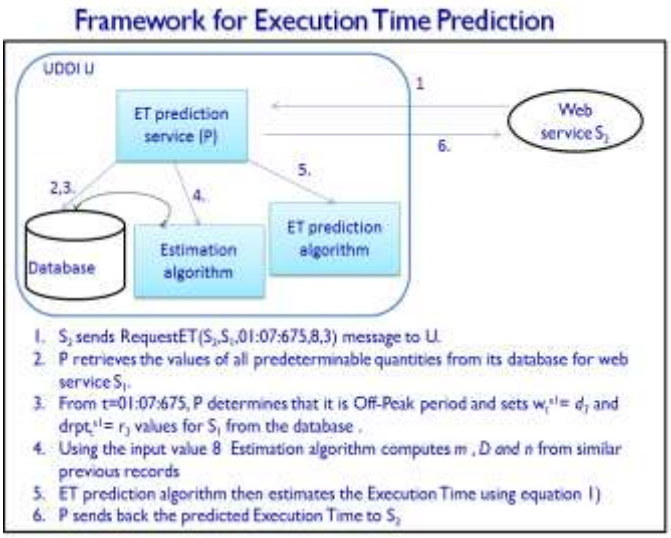

Figure. 1. Execution time prediction steps

The XML message containing the call details is described by the following XML Schema.

```
<xs:element name = "ServiceCallDetails">
<xs:complexType>
     <xs:sequence>
      <xs:element name="ServiceProviderID"
            type="xs:string"/>
      <xs:element
            name="TimeStampOfRequestor"
            type="xs:string"/>
      <xs:element
            name="RequestMessageSize"
            type="xs:integer"/>
      <xs:element
            name="ResponseMessageSize"
            type="xs:integer"/>
      <xs:element
            name="RecordedExecutionTime"
            type="xs:string"/>
      <xs:element
            name="NoofDatabaseRequests"
            type="xs:integer"/>
      <xs:element name="input"
            type="xs:string"/>
     </xs:sequence>
   </xs:complexType>
</xs:element>
```

If a web service/client, say S_2 would like to know the predicted execution time of the web service, S_1, then S_2 should send RequestET message to the UDDI. Request message should contain the following values: (RequestorId, ServiceProviderID, Timestamp,input values). In order to estimate the execution time using equation 1), the ETPrediction service P fetches the quantities mentioned in Table II from its database. The quantities mentioned in Table have to be determined by P.

Size of request and response message could be estimated from the call detail records. Similarly, no of required database requests can also be estimated from similar call detail records. Similar records to the current execution instance could be extracted based on similarity between input values. Any of the similarity calculation measures like Euclidean distances method can be used for finding similar records. The ETPrediction service P invokes the Estimation Algorithm in order to estimate these values.

Depending on the timestamp value t, P extracts the waiting time w^{s1}_t and Database request processing time $drpt^{s1}_t$ from Table II in its database. Thus after obtaining all the required values, ETPrediction service P invokes ETPrediction Algorithm which computes the estimated execution time e^{s1}_t of S_1 at time t using equation 1). P then sends back e^{s1}_t to S_2. The process is depicted in Figure 1.

V. APPLICATION OF EXECUTION TIME PREDICTION

Calling a web service includes several steps and incurs considerable cost and time at run time. When the calls are nested and if there is any kind of failure the entire sequence of calls has to be re-invoked causing considerable delay in response which results in degradation of quality of the service provided.

Our Call based checkpointing policy [6] aims at avoiding expensive re-invocations of web services and hence we propose that a web service must Checkpoint its state when it sends a request or a response message to another web service call.

 ISSN: 2454-4248
388 – 391

In order to improve the performance of composed web services with Call based checkpoints, we propose Execution Prediction based Checkpointing scheme. For each service call, this scheme decides, considering the Predicted Execution Time, e^s_t, and Mean Time Between Failures (MTBF), $mtbf^s$ of the called web service , whether a checkpoint has to be taken on making a call to the service.

MTBF of a service is an average measure of the time duration for which the service can run without failure. MTBF of a service has to be made public by the service itself by placing the MTBF in its WSDL (Web Services Description Language) file. This MTBF can then be used by the service requestors to implement the checkpointing policy. When a web service/ client S_0 calls a web service S_1, e^{s1}_t of S_1 is obtained by S_0 from the concerned UDDI under which S_1 publishes its WSDL. UDDI hosts the service which predicts the Execution Time of web services as explained in section IV above.

Execution Time Prediction based Checkpointing Rule:

If $e^{s1}_t < mtbf^{s1}$ then S_1 will execute within its MTBF and eventually send back the reply to S_0. In such a case S_0 need not take a checkpoint while calling S_1 with an anticipation S_1 might fail before completing its execution. Else S_1 might fail before sending a reply to S0 and hence S_0 must take a checkpoint before calling S_1.

VI. CONCLUSION

In this paper we have proposed a model to predict the execution time of a web service instance. Execution time of each instance of a web service varies, due to the varying amounts of a)waiting time at the server side due to multiple outstanding requests, b) database request processing time, if any, and c) message processing time due to different sizes of request and response messages. We have also proposed a framework for predicting the execution time. We have even demonstrated the use of Execution Time prediction for checkpointing web services. As part of future work we would like to implement the proposed frame work and provide strong experimental support to our approach.

VII. ACKNOWLEDGMENTS

I wish to extend my sincere thanks to guide University of Hyderabad and CVR College of Engineering for providing me the facilities to carry out my research and encouraging me through out.

REFERENCES

[1] Nuno Laranjeiro, Marco Vieira, and Henrique Madeira. *"Predicting Timing Failures in Web Services"*. Springer-Verlag Berlin, Heidelberg 2009

[2] Zhengdong Gao, Gengfeng Wu. *"Combining QoS-based Service Selection with Performance Prediction"*. Proceedings of the 2005 IEEE International Conference on e-Business Engineering (ICEBE05)

[3] Zhang Li Zhang Bin Na Jun Huang Liping Zhang Mingwei. "*An Approachfor Web Service QoS Prediction Based on Service Using Information"*. International Conference on Service Sciences (ICSS), 2010 pp. 324 to 328.

[4] Ch Ram Mohan Reddy, D Evangelin Geetha, KG Srinivasa,T V Suresh Kumar, K Rajani Kanth. "*Early performance prediction of web services"*. International Journal on Web Service Computing (IJWSC), Vol.2, No.3, September 2011.

[5] Moreno Marzolla, Raffaela Mirandola.Performance "*Prediction of Web Service Workflows"*. QoSA07 Proceedings of the Quality of software architectures 3rd international conference on Software architectures, components, and applications Springer Verlag Berlin, Heidelberg 2007.

[6] Vani Vathsala *A. " Global Checkpointing of Orchestrated Web Services"*. 1st International Conference on Recent Advances in Information Technology (RAIT), 2012 pp. 461-467.

[7] Vani Vathsala and Hrushikesha Mohanty, "*Using hmm for predicting response time of web services*", In the Proceedings of the CUBE International Conference, Pune, India, pp 520–525, 2012.