Modeling and Simulation of Quality of Service for Composite Web Services

Gregory A. Silver
Angela Maduko
Rabia Jafri
John A. Miller
Amit P. Sheth
Department of Computer Science
University of Georgia
Athens, GA 30602, USA

ABSTRACT

As businesses begin to link Web services to create new functionality in the form of composite Web services, known as Web processes, it will become increasingly important to have a way of measuring their quality of service (QoS). To this end, we present a method that uses a predictive QoS model to compute the QoS for Web processes in terms of performance, cost and reliability. The ability to compute QoS for a Web process enables an organization to tune the process. Tuning Web processes presents an interesting problem. During the act of tuning, a business may want to explore many different configurations of the Web process in order to answer "what-if" questions. Composing and evaluating the QoS for many different configurations may be prohibitive in terms of time and costs. We present a simulation based technique to overcome this challenge to tuning Web processes.

Keywords: Quality of Service, Simulation, Web Process, Web Service, Web Service Composition.

1. INTRODUCTION

Individual Web services, while useful, are limited in the functionality that they can provide. This limitation can be overcome by creating Web processes. Web processes are created by composing Web services, i.e., combining existing services to create new services. Customers select Web services based on their ability to meet some set of criteria – this activity is also termed Web service discovery. To this end, it is necessary for the customer to have a way of evaluating the service. We present a method that uses a predictive quality of service (QoS) model to compute the QoS for Web processes. The QoS model [1] computes the QoS for a Web process based on the attributes of the individual services that make up the process. The model consists of three dimensions: time, cost and reliability. The primary focus of this paper will be on the time and cost dimensions.

Businesses can use the time, cost and reliability metrics as they evaluate Web processes based on their requirements. If, at execution time, Web processes do not meet the QoS requirements that have been specified for them, modifications may need to be made. Simulation can be used as an effective tool for evaluating the possible configurations of a Web process [2].

The method presented in this paper allows a developer to evaluate the potential performance of a Web process using simulation, without actually enacting the Web process. Using this method, the developer can quickly answer "what-if" questions about the Web process by modifying it and using simulation to evaluate the modification. The Service Composition and Execution Tool (SCET) [3], a part of the METEOR-S system [7], is used to compose a Web process and generate simulation model specifications that can be processed by the JSIM simulation environment [4].

The structure of this paper is as follows: Section 2 describes related work in the area of Web services QoS, while Section 3 discusses the use of SCET and JSIM in developing a Web process. Using simulation to predict the performance of deployed Web processes is discussed in Section 4, and Section 5 proposes a method for determining the cost of a Web process. Conclusions and future work are presented in Section 6.

2. RELATED WORK

As the popularity of Web service technology grows, both providers and users of the services are becoming increasingly aware of the importance of QoS. Work has begun in several areas related to Web service QoS. Research taking place in the LSDIS lab at the University of Georgia suggests that several components of a model used to compute, analyze and monitor QoS metrics for workflows may be applied to Web processes [1]. This work promotes the idea that the QoS for Web services that make up a Web process can be computed in a way similar to that of computing the QoS for atomic tasks in a workflow.

Work is also beginning to take place in the area of QoS enabled Web services [5], [6], [8]. Web service providers need to publish information about the expected QoS of their services and service requestors need to have the ability to search for Web services that meet both their QoS and functional requirements. This requires a language capable of describing QoS measurements and an architecture that can support the discovery of QoS aware services [5]. Obtaining QoS measurements for Web processes presents several problems related to the distribution and autonomy of Web services. Simulation of Web processes can be used to help overcome these problems [3].

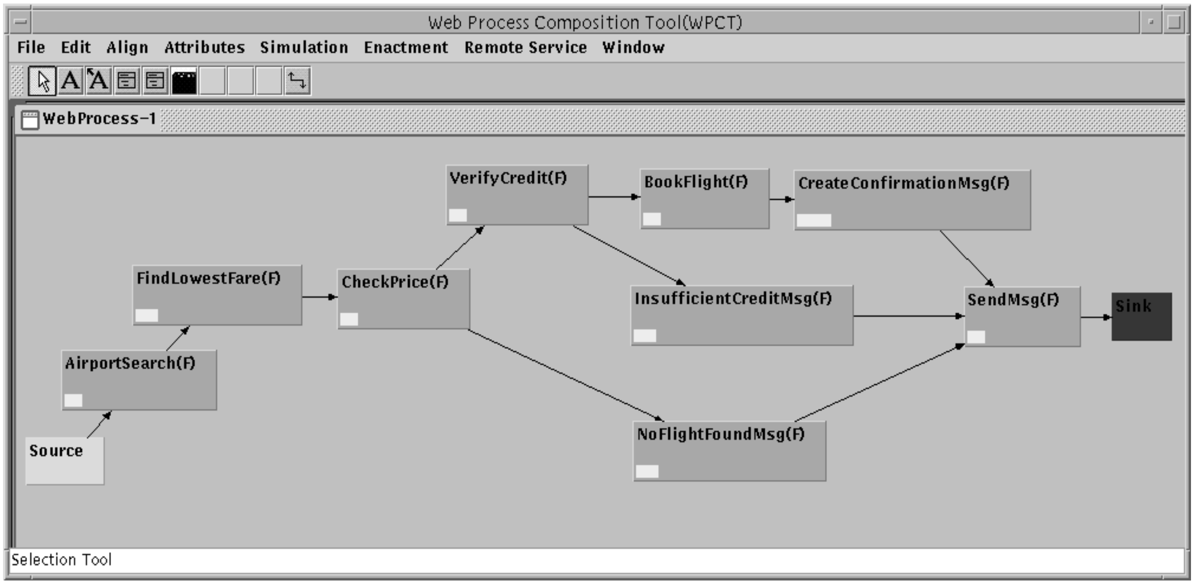

Figure 1: SCET Web Process Design

3. WEB PROCESS DEVELOPMENT

The SCET tool of the METEOR-S system provides a graphical interface which allows the user to draw a directed graph that represents a Web process [3] as shown in figure 1. The tool generates a WSFL specification for the Web process based on the information that was entered into the directed graph. It also has the ability to use the information entered into the graph to generate a JSIM model specification [4] and a Perl script for enacting the process.

The Web process depicted in Figure 1 is composed of individual Web services. The AirportSearch Web service locates departure and destination airports. LowCostFare identifies the carrier with the lowest fare. CheckPrice returns a value of true, if the fare found is less than the limit specified by the requestor. If CheckPrice returns a value of true, the customer's credit card is validated; otherwise, a message is sent to the requestor indicating that no fare within the limit was found. If the VerifyCredit service returns a value of true, the BookFlight service is executed; otherwise, a message is sent to the requestor indicating that the customer does not have enough credit to purchase the ticket. When the flight is booked, a confirmation is sent to the requestor.

We use the QoS model developed by Cardoso, et al. [1] to evaluate the time dimension of the QoS for this Web process. In order to evaluate the time dimension of the Web process, we must compute the response time of each Web service in the process. The response time consists of *service time*, *delay time* and *waiting time* [1], [3]. Service time is the time that it takes the service to execute. Delay time is the time required to send and receive the messages, and waiting time is the time that Web service invocations wait due to load on the system. Response time can be calculated as follows:

$$T(s) = S(s) + D(s) + W(s) \qquad (1)$$

The methods used to compute the service, delay and wait times are described by Chandrasekaran et al. [3]. The average response times for the Web services in Figure 1, obtained over 100 runs for each service, are shown in Figure 2.

Service	D(s)	W(s)	S(s)	T(s)
AirportsSearch	0.164	0.230	0.691	1.085
LowCostFare	0.161	0.213	0.705	1.079
CheckPrice	0.182	0.183	0.636	1.001
VerifyCredit	0.124	0.102	0.526	0.752
BookFlight	0.050	0.102	0.870	1.022
CreateConfirmation	0.043	0.124	0.937	1.104
InsufficientCreditMsg	0.043	0.178	0.878	1.099
NoFlightFoundMsg	0.054	0.108	0.926	1.088
SendMail	0.065	0.080	1.047	1.192

Figure 2: Response Times for Web services

Information related to a Web service's QoS time dimension should be published by the service provider. Once the response time information for each Web service within a process is known, the Stochastic Workflow Reduction (SWR) algorithm developed by Cardoso et al. [1] can be used to compute the response time for the Web process. Six distinct reduction rules are applied to a Web process until only one atomic Web service remains. This remaining Web service contains the QoS metrics corresponding to the process being analyzed. For the Web process depicted in Figure 1, the transitional probabilities are set to 1 for the sequential portions of the process and empirically derived probabilities are assigned to the transitions for each conditional branch. Since the only subsystems in this particular process are sequential and conditional ones, we need to apply only the sequential and conditional reduction rules to calculate the response time for the Web process as shown in Figure 5. The sequential rule reduces a system of two sequential services to one service, whereas the conditional rule reduces a system of conditional services to that of three sequential services.

The sequential reduction rule is applied to the Web services BookFlight and CreateConfirmationMsg in Figure 1, to generate Seq1. When these Web services are replaced with the generated Seq1, a situation such as depicted in Figure 3 results. The conditional reduction rule can not be applied to the parallel system in Figure 3 since the SendMsg service has an incoming transition from outside of this system (i.e., the incoming transition from the NoFlightFoundMsg service). Therefore, we introduce a null Web service (i.e., a Web service node without any associated actions) as a direct predecessor of the SendMsg service which takes over the incoming transitions of that service as shown in Figure 4. This null Web service has no other effect on the composed Web process. The conditional reduction rule is then applied to Seq1 and InsufficientCreditMsg to generate Cond1. Replacing these with Cond1 results in the situation depicted in Figure 5. Recursively applying the sequential / conditional reduction rules in the same way generates the desired Web Process as shown in the equations of Figure 6.

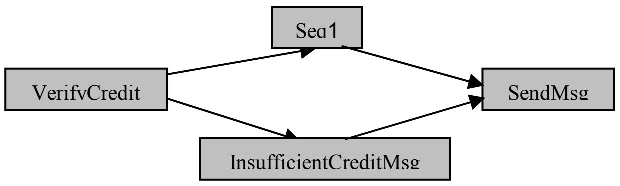

Figure 3

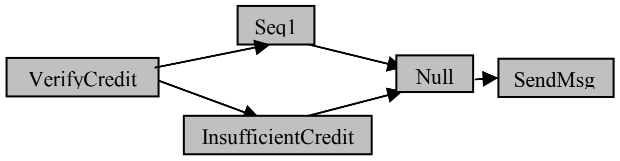

Figure 4

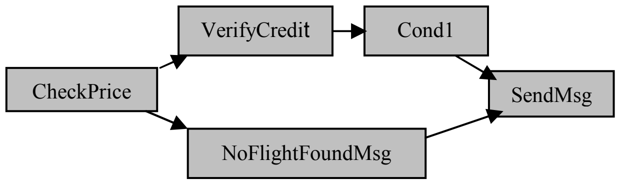

Figure 5

The SCET tool also has the ability to generate JSIM simulation model specifications. Each node in the Web process graph represents a JSIM resource. The Web process specification maps to a JSIM model specification in the following way:

- Web services are modeled as JSIM Facilities.
- Source nodes are modeled as JSIM Sources.
- Sink nodes are modeled as JSIM Sinks.
- Data and Control links between Web services are modeled as JSIM Transports.
- Messages are modeled as JSIM SimObjects.

Each Source in the JSIM model specification requires a minimum of two parameters, a mean inter-arrival time, and a distribution that characterizes the inter-arrival time. The developer can enter these parameters from the graphical designer of the SCET tool. The mean inter-arrival time represents the average amount of time between the arrival of messages to be sent to the first Web service in the Web process.

Each facility in a JSIM model specification also requires a minimum of two parameters: a mean service time and a distribution that characterizes the service time.

4. WEB PROCESS SIMULATION

Once the JSIM model specification has been created, it can be used to simulate the Web process, but it should be validated in order to show that it accurately represents the behavior of the actual Web process. Our first step in validating the JSIM model for the Web process in Figure 1 was to verify the reliability of the input data.

The input data for the model consisted of the mean inter-arrival time for the source node, mean service times of each Web service and their related probability distributions. The mean service time for each Web was gathered by executing the Web service 100 times and taking an average of the service times.

The probability distribution for each Web service was determined by plotting a histogram for service times of the Web services and hypothesizing a distribution based on the shape of the histogram.

The hypothesized distribution was then validated using the Kolmogorov-Smirnov goodness-of-fit test at 0.05 level of significance. Figures 7 and 8 show the results of a Kolmogorov-Smirnov goodness-of-fit test for the VerifyCredit Web service using the normal distribution. The null hypothesis is not rejected since the observed test statistic 0.044 is not greater than the critical value 0.136. The inter-arrival time mean and probability distribution were known because the execution of the Web process was controlled by a testing engine.

Our next step was to validate the input-output transformations. Simulation is executed using the mean service time and distribution function for each Web service as input to the JSIM model. When the simulation is executed, it is expected that the average wait times generated via simulation will be consistent with the average wait times observed when the actual Web services were executed. The simulation was executed several times with each replication using a different seed. The sample wait time for each of the replications was compared to the observed wait times from the Web services. Figure 9 shows the hypothesis tested for one of the Web services. The observed average wait time for VerifyCredit(Z_1) is compared to the model service time for VerifyCredit(Y_1). We performed a t-test with $\alpha = 0.05$ and $n = 10$, where n is the number of replications of the

T(Seq1) = T(BookFlight) + T(CreateConfirmationMsg)
*T(Cond1) = T(Seq1) * p(Seq1) + T(InsufficientCreditMsg) * p(InsufficientCreditMsg)*
T(Seq2) = T(VerifyCredit) + T(Cond1)
*T(Cond2) = T(Seq2) * p(seq2) + T(NoFlightFoundMsg) * p(NoFlightFoundMsg)*
T(Web Process) = T(AiportSearch) +T(LowCostFare) + T(CheckPrice) + T(Cond2) + T(SendMsg)

Where T(X) represents the cost of Web service X and p(X) represents the transition probability of Web service X.

Figure 6: Response Time Calculation for a Web Process

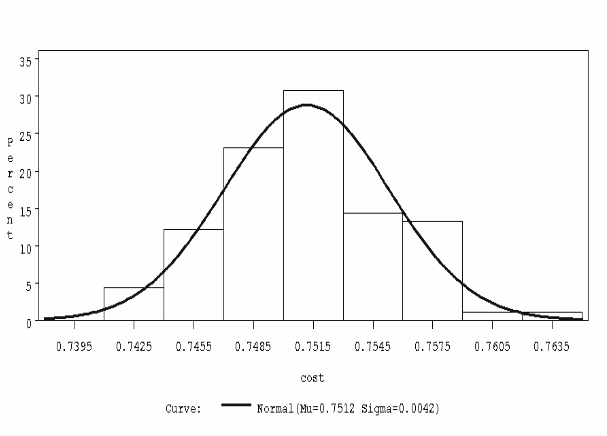

Figure 7

Kolmogorov-Smirnov Goodness-of-Fit Test for Normal Distribution	
Mean	0.751162
Standard Deviation	0.004152
Test Statistic D	0.04428375
Level of Significance	0.05
Critical Value	0.1365
P - Value	>0.150

Figure 8

Average Wait Time: $Z_1 = 0.102$

Hypothesis: $H_0 = 0.102$

$H_1 \neq 0.102$

Sample Mean Wait Time: $Y_1 = 0.092$

Test Statistic: $t_0 = -1.443$

Figure 9: Testing H_0 for Web service VerifyCredit

simulation model, to determine whether or not to reject H_0 for each of the Web services in the process

Hypotheses similar to the one in Figure 9 were tested for each of the Web services in order to validate the JSIM model. None of the hypotheses were rejected, indicating that the model accurately represents the behavior of the actual Web process. Figure 10 contains the results of hypothesis tests for several of the Web services.

5. COST

The primary focus of this paper has been the QoS time component, but as mentioned earlier the QoS of a Web process should consist of cost and reliability as well as time.

The cost dimension of the QoS for a Web process refers to the cost incurred when the process is executed. In order to calculate the cost for the entire process, we first need to define the cost for an individual Web service. This cost can be defined from two perspectives: that of the service provider and that of the consumer of the service. From the service provider's point of view, this would include the cost associated with the hardware and software resources needed to set up and run the service as well as the expense of hiring personnel to maintain and update the service and its interface [1]. From the service consumer's perspective, the cost of the Web service includes not only the fee charged by the provider of the service, but also the hardware and software resources used to invoke it and the amount of time taken to execute it. We are interested in the cost of the Web service only from the consumer's perspective. Since the service invocation is usually just a function call and the overhead associated with it is negligible, it may be disregarded in our measurement of cost. Moreover, since time has already been defined as a distinct dimension of the QoS model, it too, can be eliminated from the cost calculations. Hence, we end up defining the cost of an individual Web service purely in monetary terms, i.e., the amount of money one would need to spend in order to make use of the service. Since Web services are still a relatively new concept/technology, the majority of these services are being offered free of charge. However, some of the companies hosting these services are now demanding a fee for their use and others are following suit. There are two prevalent charging policies:

1) Charge per use;
2) Lease-based charging.

In the first case, the user pays a fee for each invocation of the service, whereas in the second case, he makes a payment that allows him to invoke the service any number of times within a time period (e.g., a week, a month or a year) specified by the service provider. If the expected number of invocations that would be made to the service within the lease period is known, an estimate of the cost of each invocation can be obtained as follows:

$$\frac{Lease\quad Cost}{E(Number\quad of\quad Invocations)} \tag{2}$$

Once the costs of the individual Web services have been calculated, the cost dimension of the QoS for the Web process can be calculated using the QoS model developed by Cardoso et al. [1].

The JSIM simulation environment is being enhanced to allow the model developer to associate a cost with each JSIM facility.

Simulated Data									
	AirportSearch	LowCostFare	CheckPrice	BookFlight	FlightConfirm	InsufficientCredit	NoFlight	VerifyCredit	SendMsg
	0.251	0.245	0.213	0.111	0.133	0.121	0.063	0.120	0.062
	0.205	0.202	0.170	0.152	0.000	0.217	0.058	0.094	0.069
	0.210	0.201	0.170	0.089	0.121	0.064	0.107	0.064	0.051
	0.137	0.136	0.106	0.117	0.138	0.184	0.016	0.085	0.087
	0.224	0.212	0.178	0.096	0.122	0.151	0.198	0.069	0.067
	0.260	0.253	0.219	0.101	0.121	0.228	0.154	0.102	0.195
	0.211	0.209	0.172	0.064	0.086	0.145	0.086	0.081	0.074
	0.233	0.227	0.200	0.107	0.127	0.222	0.114	0.129	0.057
	0.250	0.239	0.210	0.115	0.132	0.126	0.142	0.103	0.118
	0.207	0.198	0.166	0.092	0.101	0.135	0.086	0.075	0.116
Mean Wait Time	0.219	0.212	0.180	0.104	0.108	0.159	0.102	0.092	0.090
StdDev	0.035	0.033	0.033	0.023	0.041	0.053	0.053	0.021	0.044
Observed Data									
Expected Wait Time	0.230	0.213	0.183	0.102	0.124	0.178	0.108	0.102	0.080
t-statistic	-1.009	-0.076	-0.249	0.332	-1.224	-1.117	-0.336	-1.443	0.690
alpha = .05	2.26 -2.26	2.26 -2.26	2.26 -2.26	2.26 -2.26	2.26 -2.26	2.26 -2.26	2.26 -2.26	2.26 -2.26	2.26 -2.26
Decision	do not reject	do not reject	do not reject	do not reject	do not reject	do not reject	do not reject	do not reject	do not reject

Figure 10: t-test Results for Web services

Since JSIM facilities are used to represent Web services during simulation, the cost of the facility will become the cost of the Web service.

6. CONCLUSIONS AND FUTURE WORK

Quality of service is an important factor in determining the utility of a Web service. As businesses begin to create new functionality in the form of composite Web services, QoS becomes increasingly more important.

This paper has presented a method for evaluating the QoS of composite Web services along with tools to enable their evaluation.

Simulation is an important part of the evaluation process, as it allows the developer to explore and tune the Web process without experiencing the cost of having to enact the process for many different configurations.

The JSIM simulation environment does not currently support the QoS reliability dimension, but has the components needed to implement such functionality. Since Web services are represented as facilities in a JSIM model, reliability may in some cases be modeled using JSIM signal objects. JSIM signals are able to increase and decrease the number of service units in a JSIM facility. A signal can be used to alternately decrease the number of service units in a facility from one to zero and then from zero to one. When the number of units in a facility is zero, it cannot process any simulation entities and can be considered unavailable, and when the number of units is increased from zero to one, the facility can operate normally and is considered available.

The following JSIM enhancements are either in process or planned for the near future:

- Simulation of Web service costs;
- Simulation of Web service reliability;
- Ability to dynamically adjust the load on JSIM facilities representing Web services;
- Ability to specify message delay times for JSIM transports used to connect JSIM facilities representing Web services.

These enhancements along with the methods presented in this paper promise to provide a fully functional environment for evaluating the QoS of Web services.

REFERENCES

[1] Jorge Cardoso, John Miller, Amit Sheth and Jonathan Arnold, **Modeling Quality of Service for Workflows and Web service Processes**, Technical Report UGA-CS-TR-02-002, LSDIS Lab, Computer Science Department, University of Georgia, May 2002.

[2] Senthilanand Chandrasekaran, Gregory Silver, John A. Miller, Jorge Cardoso and Amit P. Sheth, **Web service Technologies and their Synergy with Simulation**, Proceedings of the 2002 Winter Simulation Conference), San Diego, California (December 2002) pp. 606-615.

[3] Senthilanand Chandrasekaran, John A. Miller, Gregory Silver, I. Budak Arpinar and Amit P. Sheth, **Performance Analysis and Simulation of Composite Web services**, Electronic Markets: The International Journal of Electronic Commerce and Business Media*,* Vol. 13, No. 2, Spring 2003, Taylor and Francis Publishing. (to appear)

[4] John A. Miller, Rajesh Nair, Zhiwei Zhang and Hongwei Zhao, **JSIM: A Java-Based Simulation and Animation Environment**, Proceedings of the 30th Annual Simulation Symposium, Atlanta, Georgia (April 1997) pp. 31-42.

[5] Peter Farkas and Hassan Charaf, **Web services Planning Concepts, Journal of WSGC**, Vol. 11, No. 1.

[6] Anbazhagan Mani, Arun Nagarajan, **Understanding Quality of Service for Web services**, http://www106.ibm.com/developerworks/webservices/library/ws-quality.html

[7] **METEOR-S: Semantic Web Services and Processes, Applying Semantics in Annotation, Quality of Service, Discovery, Composition, Execution**, http://lsdis.cs.uga.edu/proj/meteor/SWP.htm

[8] Daniel A. Menascé, **QoS Issues in Web Services,** 6 (6), November/December 2002, pp. 72-75.