Response time analysis for composite Web services

Serge Haddad
Laboratoire Lamsade
Université de Paris Dauphine
Place du Mal. de Lattre de Tassigny
75775 cedex 16, France
serge.haddadd@lamsade.dauphine.fr
Phone Number: (331) 44 05 40 6
Fax Number: (331) 44 05 40 91

Lynda Mokdad
Laboratoire Lamsade
Université de Paris Dauphine
Place du Mal. de Lattre de Tassigny
75775 cedex 16, France
lynda.mokdad@lamsade.dauphine.fr
Phone Number: (331) 44 05 40 60
Fax Number: (331) 44 05 40 91

Samir Youcef
Laboratoire Lamsade
Université de Paris Dauphine
Place du Mal. de Lattre de Tassigny
75775 cedex 16, France
samir.youcef@lamsade.dauphine.fr
Phone Number: (331) 44 05 40 60
Fax Number: (331) 44 05 40 91

Abstract—[1] **Service Oriented Computing (SOC) strives for applications with services as the fundamental items of design, and Web services acting as the enabling technology. Web services use open XML-based standards and are becoming the most important technology for communication between heterogenous business applications over Internet. In this paper, we focus on mean response times. Thus we propose analytical formulas for mean response times for structured BPEL constructors such as *sequence*, *flow* and *switch*. We propose also a response time formula for *multi-choice* pattern which is a generalization of switch constructor. Contrarily to previous studies in the literature, we consider that the servers can be heterogenous and the number of invoked elementary Web services can be variable.**

Keywords: composite Web service, BPEL constructors, response times, analytical formulas.

I. INTRODUCTION

Service Oriented Computing (SOC) introduced the concept of software as services, which can be integrated and reused by other applications. Service providers publish their services in Universal Description Discovery and Integration (UDDI). These services are searched by potential clients, therefore reducing time-to-market of a product. Service Oriented Architecture (SOA) [14] provides a paradigm to use capabilities that may be under the control of different ownership domains. Interoperability among service providers is ensured by open protocols and standards like Web Service Description Language (WSDL) [15] and Simple Object Access Protocol (SOAP) [16].

Elementary Web services offer only limited capabilities. Thus a composite Web service composes elementary Web services in order to achieve a complex request. This composite service controls the coordination between elementary services. This process is called Web service orchestration and is transparent for Web service clients. Composite service activities may be defined by control flow graphs and data graphs.

For a service provider, it is important to (upper) bound the mean response time of a request given some request load and some architectural environment. Furthermore, this computation should be performed before the deployment of the service [17]. Moreover in case of a composite service, this performance evaluation also depends on hypotheses about the invoked elementary services.

In a recent paper, Menasce [1] studied the response time of Web services with the same statistical characteristics and where the number of invoked Web services is constant. The objective of this work is to overcome these two limitations. So our study takes into account different statistical characteristics for the services and a random number of invoked services. The former extension is required in order to handle heterogeneous servers w.r.t. the performance criterion. The latter one captures the fact that the number of invocations depends on the parameters of the request which are used as a filter for invocation.

More specifically, in this paper, we give analytical formulas for the mean response time of structured BPEL constructors (like sequence, flow, switch) and of multi-choice pattern.

This work is organized as follows. Section II presents related work. Section III details the different structured BPEL constructors. Section IV presents analytical formulas for response time of these constructors. in section V, we give the response time formula for multi-choice pattern which is a generalization of switch constructor. Numerical results are given in section VI. Finally, section VII concludes and gives some perspectives to this work.

II. RELATED WORK

Most of the work in the domain of performance of Web services is concentrated towards composite web services and their response time. The execution of a composite service have been studied as a fork-join model in [1]. This model states that a single Internet application invokes many different Web services in parallel and gathers their responses from all these launched services in order to return the results to a client. Server times for composite database Web services has been studied in [2], which follows a fork-join model of execution. The author proposes that while performing a join operation, servers with slow response times can be eliminated to maximize the performance. The work is more oriented towards studying fork-join model in order to understand the merger of results from various servers.

[1] This work is supported by ANR Perso

The exact analysis of fork and join system, under some hypothesis, can be found in [3]. These hypothesis state that the number of servers is equal to two, the job arrival is Poisson process and the tasks have exponential service time distribution. Nelson and Tantawi [4] proposed an approximation in the case where the number of servers is greater or equal to two and homogeneous exponential servers. Thereafter, a more general case is presented in [6] [7], where arrival and service process are general. An upper and lower bound are obtained by considering respectively $G/G/1$ and $D/G/1$ queuing parallels systems.

XML and SOAP protocols have been tested for their execution time and throughput [8],[9],[10] by executing and measuring response time of SOAP-based Web services. Latency of SOAP implementations currently available has been presented in [8] and are compared with existing protocols such as RMI, RMI/IIOP or CORBA/IIOP. XML-based protocols for Web services have been critically studied in [9] and binary encoded protocol has been proposed instead of text-based XML ones. Klingemann and al. [11] use a continuous Markov chain to estimate the execution response time and the cost of workflow. In [11], the authors propose an algorithm witch determines the QoS of a Web service composition by aggregating the QoS dimensions of the individual services, based on a collection of workflow patterns defined by Van der Aalst's and al. [13]. These QoS include upper and lower bounds of execution time as well as throughput. In order to improve the availability of Web services, Cotroneo and al. [11] propose a new architecture of middleware which is suitable for increasing the service availability for a group of premium users. In [12], we have studied end-to-end response time for composite Web services representing a factor of Internet overhead in the execution model.

Contrarily to these previous studies, where the servers are not supposed heterogenous and their number is always constant, the objective in this paper is justly to overcome their limitations. Thus, we propose analytical formulas for mean response time of composite Web services assuming that servers are heterogenous and the number of invoked elementary Web services can be variable.

III. BPEL CONSTRUCTORS

Business Process Execution Language for Web services (BPEL4WS) has been built on IBM's WSFL (Web Services Flow Language) and Microsoft's XLANG (Web services for Business Process Design) and combines accordingly the features of a block structured language inherited from XLANG with those for directed graphs originating from WSFL [5]. The language BPEL is used to model the behavior of both *executable* and *abstract* processes.

- An abstract process is a not an executable process and which is a business protocol, which use process descriptions that specify the mutually visible message exchange behavior of each of parts involved in the protocol, without revealing their internal behavior.
- An executable process specifies the execution order between a number of activities constituting the process, the partners involved in the process, the messages exchanged between these partners and the fault and exception handling specifying the behavior in cases of errors and exceptions.

In the BPEL process each element is called an activity which can be a primitive or a structured one. The set { *invoke, receive, reply, wait, assign, throw, terminate, empty* } are primitive activities and the set {*sequence, switch, while, pick, flow, scope*} are structured activities.

In this paper, we are interested on the *sequence*, *flow* and *switch* activities also called constructors. The description of each of them is given on an example described as below .

As service composition implies several elementary Web services. The relationships among the individual Web services may be represented by a connected and directed graph $G = (X, V)$ where $X = \{s_i, s_j, \cdots, s_n\}$ is the set of elementary Web services and $V = \{(s_i, s_j) : s_i, s_j \in X \wedge s_i \text{ can invoke } s_j\}$. $G = (X, V)$ is called the **composition graph**. Figure 1, which is reproduced from [1], presents a composition graph example implying six individual Web services s_1, s_2, s_3, s_4, s_5 and s_6.

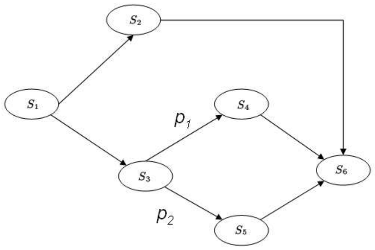

Fig. 1. An example of composition graph

In the following, we give the description of the constructor that we are interested and we denote by $\overline{R}$, the mean response time. Thus, $\overline{R}_{s_i}$ is the mean response time of the web service s_i.

- *Sequence*. Web service is activated as a result of the completion of one of a set of mutually exclusive predecessor activities. These activities may be listed with the XML <sequence> tag, that is, in lexical order. *Example*: s_2 and s_6 in Figure 1. The response time for this constructor in this example is given by:

$$\overline{R}_{sequence(s_2,s_6)} = \overline{R}_{s_2} + \overline{R}_{s_6}$$

- *flow*. It represents a point in the process where a single thread of control splits into multiple threads of control which can be executed in parallel (fork). This pattern is supported by BPEL using XML <flow> tag. *Example*: s_1 in Figure 1 which can invoke s_2 and s_3 in parallel. The response time for this constructor in this example is given by:

$$\overline{R}_{flow(s_2,s_3)} = Max(\overline{R}_{s_2}, \overline{R}_{s_3})$$

- *switch*. A probability value p on an outgoing arrow from s_i to s_j indicates that s_i invokes s^j with probability p. If no value is indicated, the probability is assumed to be 1. *Example*: s_3 in Figure 1 which can invoke s_4 with probability p_1 or s_5 with probability $p_2 = 1 - p_1$. The response time for this constructor in this example is given by:

$$\overline{R}_{switch(p_1,(s_4,s_5))} = p_1\overline{R}_{s_4} + p_2\overline{R}_{s_5}$$

In the following, we give analytical formulas for response times to each considered constructor.

IV. Response Times of structured BPEL constructors

In this section, we give analytical formulas for mean response times for structured BPEL constructors and we consider the case that the execution time of each elementary Web service s_i of a composite Web service S, is exponentially distributed and we consider also that the number of invoked elementary services are random.
Thus, we consider in the following the basic control patterns supported by BPEL standard. More specifically, the control patterns considered are: sequence, parallel split (flow), exclusive choice (switch).

A. *Computation for the* **sequence** *constructor*

the constructor *sequence* correspond to a sequential execution of s_1 to s_n elementary Web services. The analytical formulas of mean response time $E(T^{sequence})$ is given by:

$$E(T^{sequence}) = \sum_{i=1}^{n} E(T_i) \tag{1}$$

Proof: The execution time of composite Web service S composed by n elementary web services is given by: $T^{sequence} = \sum_{i=1}^{n} T_i$ which is easier to derive from equation (1). ■

Case of homogeneous servers. In the case of $T_i, i \in \{1, ..., n\}$ are random variables with exponential distributions with rate λ for each T_i, the mean response time of composite Web service S is trivial and is given by:

$$E(T_{exp}^{sequence}) = \sum_{i=1}^{n} \frac{1}{\lambda} = \frac{n}{\lambda}$$

Case of heterogenous servers. As we said before, we overcome the limitation of other studies by considering that the servers are heterogeneous. Thus, we consider that the execution time of k elementary services s_i follow an exponential distribution with rate μ and the execution time of $n - k$ services follow an exponential distribution with rate λ. Thus, the response time for a composite Web service S is given by:

$$E(T_{exp}^{sequence}) = \frac{n-k}{\lambda} + \frac{k}{\mu}$$

B. *Computation for the* **flow** *constructor*

One of the most important benefits of the component approach is the reuse. In the WSDL language, the elementary Web services are conceptually limited to relatively simple features that can be modelled by collection of operations. However, for some kind of applications, it is necessary to combine a set of Web services into composite web services. Thus, in this section, we are interested to the mean response time of a composite Web service S which is composed by n elementary services invoked in parallel. In [1], the author give an analytical formula for the response time of flow constructor but he suppose that n is fixed. Our contribution is to consider that n is random. In addition, we generalize the results given in [1] where the author consider that only one execution time of an elementary service is different. Our contribution is also to consider that we can have k elementary service times with rate μ and $k - n$ others with rate λ with k not fixed.

In the following, we give an analytical expression for the mean response time:

$$E(T^{flow}) = \sum_{i=1}^{n} \int_0^{\infty} t f_i(t) \prod_{j \neq i}^{n} F_j(t) dt \tag{2}$$

where:

$$T^{flow} = Max\{T_i, i = \overline{1, n}\}$$

As we assume that the random variables T_i are independents, the cumulative function of random variable T^{flow} is given by:

$$F(T^{flow}) = P(T^{flow} \leq t) = \prod_{i=1}^{n} F_i(t)$$

So the probability density of T^{flow} is:

$$f_{T^{flow}}(t) = \sum_{i=1}^{n} f_i(t) \prod_{j \neq i}^{n} F_j(t) \tag{3}$$

Thus $E(T^{flow})$ can be derived easily.

Case of exponential distributions. We give in the following the mean response time analytical formula where the random variables T_i, $i \in \{1, ..., n\}$ are exponentially distributed with rates λ_i.

$$E(T_{exp}^{flow}) = \sum_{i=1}^{n} \lambda_i \sum_{X \in \mathcal{P}(E \setminus \{i\})} \frac{(-1)^{|X|}}{(\sum_{j \in X} \lambda_j + \lambda_i)^2} \tag{4}$$

Proof:

From equation 3, the probability density of random variable T_{exp}^{flow} is given by:

$$f_{T_{exp}^{flow}}(t) = \sum_{i=1}^{n} \lambda_i e^{-\lambda_i t} \prod_{j \neq i}^{n} (1 - e^{-\lambda_j t})$$

The average response time is:

$$E(T_{exp}^{flow}) = \sum_{i=1}^{n} \lambda_i \int_0^{\infty} t e^{-\lambda_i t} \prod_{j \neq i}^{n} (1 - e^{-\lambda_j t}) \ dt$$

As we have:

$$\forall n \geq 2, \prod_{j \neq i}^{n} (1 - e^{-\lambda_j t}) = \sum_{X \in \mathcal{P}(E \setminus \{i\})} (-1)^{|X|} e^{-(\sum_{j \in X} \lambda_j t)}$$

Thus we obtain that the mean response time for a composite Web service S is given by the following formula:

$$E(T_{exp}^{flow}) = \sum_{i=1}^{n} \lambda_i \sum_{X \in \mathcal{P}(E \setminus \{i\})} \frac{(-1)^{|X|}}{(\sum_{j \in X} \lambda_j + \lambda_i)^2}$$

■

Case of homogeneous servers. In the case of all elementary service times are exponentially distributed with the same rate λ_i (i.e $\forall i \in \{1, ..., n\}$), $\lambda_i = \lambda$), the response time for S given in [1] is:

$$E(T_{exp}^{flow}) = n \sum_{k=0}^{n-1} \frac{(-1)^k}{\lambda(1+k)^2} \tag{5}$$

Case of heterogeneous servers. Our generalization is to consider that k elementary service times follow an exponential distribution with rate μ and $n - k$ elementary service times follow an exponential distribution with rate λ and with k not fixed and we consider a factor g which is the slowdown factor such that $\mu = g\lambda$. With these assumptions, the response time of S is as follows:

$$E(T_{exp}^{flow}) = R_1 + R_2 \tag{6}$$

Where,

$$\begin{cases} R_1 = k\mu \sum_{j=0}^{n-1} \sum_{m=0}^{j} (-1)^j \frac{C_{i-k}^{m} C_{k-1}^{j-m}}{[m\lambda + (j+1-m)\mu]^2} \\ R_2 = (n-k)\lambda \sum_{j=0}^{n-1} \sum_{m=0}^{j} (-1)^j \frac{C_{n-1-k}^{m} \times C_{k}^{j-m}}{[(m+1)\lambda + (j-m)\mu]^2} \\ \qquad\qquad\qquad\qquad et \end{cases}$$

This equation (6) is easily derived by the equation (4) by considering that $\lambda_i = \mu$, $\forall i \in \{1, ..., n-k\}$ and $\lambda_i = \lambda$, $\forall i \in \{n-k+1, ..., n\}$.

C. *Computation for the* **switch** *constructor*

In this case, we consider that we have one choice of n elementary Web services. Let $P(Y = i)$ the invocation probability of elementary Web i, with $\sum_{i=1}^{n} P(Y = i) = 1$. In this case, the response time of *switch* constructor is given by the following analytic formula:

$$E(T^{switch}) = \sum_{i=1}^{n} P(Y = i) E(T_i) \tag{7}$$

with $E(T_i)$ the mean response time of service i.

Proof: First we compute the probability density of the random variable T^{switch}. The cumulative distribution function of the variable T^{switch} is defined as: $F_{T^{switch}}(t) = P(T^{switch} \leq t)$.

According to the total probability theorem, we can write:

$$F_{T^{switch}}(t) = \sum_{i=1}^{n} P(T^{switch} \leq t \setminus Y = i) P(Y = i)$$

Thus, probability density function of random variable T^{switch} is given by:

$$f_{T^{switch}}(t) = \sum_{i=1}^{n} f_{T_i}(t) P(Y = i)$$

The definition of the average of T^{switch} allow to deduce the result given in equation (7). ■

Case of exponential distribution. As in this paper, we consider the case of exponential distribution time for each elementary service time, thus the formula for mean response time is given by:

$$E(T_{exp}^{switch}) = \sum_{i=1}^{n} \frac{P(Y = i)}{\lambda_i} \tag{8}$$

Case of heterogeneous servers. As well as in the case of the previous presented constructor, we give in the following the response time for the case that the execution times of elementary services are not the same:

$$E(T_{exp}^{switch}) = \frac{1}{\mu} \sum_{i=1}^{n-k} P(Y = i) + \frac{1}{\lambda} \sum_{i=n-k+1}^{n} P(Y = i) \tag{9}$$

In the next section, we are interested to multi-choice pattern which is not supported directly by BPEL, but it can be implemented using the links controls inherited from WSFL.

V. Computation for the **multi-choice** pattern

The multi-choice pattern allows the invocation of a subset of elementary services among the n possible. Take for example the case of a booking flights operated as follows: Web services invoked depend on two criteria namely the city of departure and destination. Next, according to these cities, agencies providing this trip are invoked on parallel. The number of services, and relied on is random.

Let N the random variable for the number of invoked services and $P(N = i)$ the probability that the number of invoked service is equal to i, with n maximum number of the invoked services.

In this case, the response time of composite web service S is given by the following formula:

$$E(T^{multichoice}) = \sum_{i=1}^{n} [P(N = i) E(T_{S^i})] \tag{10}$$

Where $E(T_{S^i})$ is the mean response time for composite Web service S when i elementary services are invoked.

Proof: First, we give the cumulative function $F_{T^{multichoice}}(t)$ of random variable $T^{multichoice}$. $F_{T^{multichoice}}(t) = P(T^{multichoice} \leq t)$.
From totaly probability theorem, we can obtain:

$$F_{T^{multichoice}}(t) = P(\bigcup_{i=1}^{n}\{P(T^{multichoice} \leq t) \wedge N = i\})$$

The events $(N = i,\, i \in \{1, ..., n\})$ are incompatible, so:

$$F_{T^{multichoice}}(t) = \sum_{i=1}^{n} P(T^{multichoice} \leq t \wedge N = i)$$

thus,

$$F_{T^{multichoice}}(t) = \sum_{i=1}^{n} P(T^{multichoice} \leq t \setminus N = i)P(N = i)$$

So:

$$F_{T^{multichoice}}(t) = \sum_{i=1}^{n} P(T_{S^i} \leq t)P(N = i)$$

The cumulative function of $T^{multichoice}$ is:

$$F_{T^{multichoice}}(t) = \sum_{i=1}^{n} F_{T_{S^i}}(t)P(N = i)$$

We can derive the probability density $f_{T^{multichoice}}$ of $T^{multichoice}$ and we obtain:

$$f_{T^{multichoice}}(t) = \sum_{i=1}^{n} f_{T_{S^i}} P(N = i)$$

■

Case of exponential distribution. As, we consider the case that the elementary service execution times are exponentially distributed with rate λ and the invocation probability of elementary service s_i is p, thus the mean response time for composite Web service S can be easily derived from equation (10) and is given as follows:

$$E(T_{exp}^{multichoice}) = \frac{n}{\lambda}\sum_{i=1}^{n} C_n^i p^i (1-p)^{n-i} \sum_{k=0}^{i-1} \frac{(-1)^k}{(1+k)^2} \quad (11)$$

Case of heterogeneous servers. We give also the analytical formula for composite Web service response time where we consider two classes of elementary services. The execution time in each class is the same. N^1 (resp. N^2) is the random variable which defined the number of elementary services in class 1 (resp. class 2). The mean response time formula is also derived from equation (10) and is given by:

$$E(T_{exp}^{multichoice}) =$$

$$\sum_{i=1}^{n} P(N^1 = i) \sum_{j=0}^{k} E(T^{multichoise}(i,j))P(N^2 = j/N^1 = i) \quad (12)$$

With:

$$\begin{cases} P(N^1 = i) = C_n^i p^i (1-p)^{n-i}. \\ P(N^2 = j/N^1 = i) = \frac{C_k^j \times C_{n-k}^{i-j}}{C_n^i}. \end{cases}$$

VI. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this section, we present some numerical computation and results that we obtained, when two classes of elementary Web services are considered. Indeed, some elementary web service execution times are exponentially distributed with rate μ and others with rate λ.

As in [1], let first define a heterogenous coefficient noted g, such as $\mu = g\lambda$. It's clear that if $g = 1$, then all of elementary Web services belong to the same class (i.e. the elementary Web services are homogeneous). However, if $g > 1$, it means that Web services belong to the second class are slower then services belong to the first class. For simplicity, we assume that the probability of elementary Web services invocation is p for all services. So, the response time of *multi-choice* pattern is given by the following equation:

$$E(T_{exp}) = \sum_{i=1}^{n} P(N^1 = i) \sum_{k=0}^{s} E(T_{i,k})P(N^2 = k/N^1 = i) \quad (13)$$

where:

$$\begin{cases} P(N^1 = i) = C_n^i p^i (1-p)^{n-i}. \\ P(N^2 = k/N = i) = \frac{C_s^k \times C_{n-s}^{i-k}}{C_n^i}. \end{cases}$$

It's clear that when $g = 1$, the synchronization time is the same for any value for the number of elementary Web services belong to the second class denoted N^2.

In figure 2, we give the response times by varying the slowdown factor g and where we consider different values of the number of elementary services for second class. We denote by s this number which takes these values ($s = 5$, $s = 10$, $s = 15$, $s = 20$).

In figure 3, we give the response times by varying the the number of elementary services for second class and we consider the case of $g = 10$, $g = 15$ and $g = 20$.

From figure 2, we can conclude two things. One for any value of N^2, the synchronization response time increases exponentially with the heterogeneous coefficient g. Second, when $g = 1$ the response time of the composite Web service is the same for any value of the elementary Web services belong to the second class.

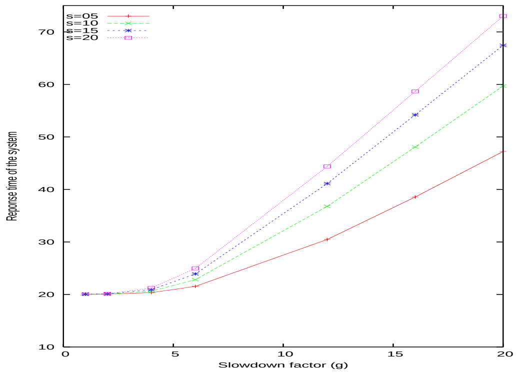

Fig. 2. Response times for composite Web service

From figure 3, we can notice that the waiting time increase in logarithmic way with invocation probability p. It's clear, also, that the response time increases logarithmic way with the number of invoked Web services.

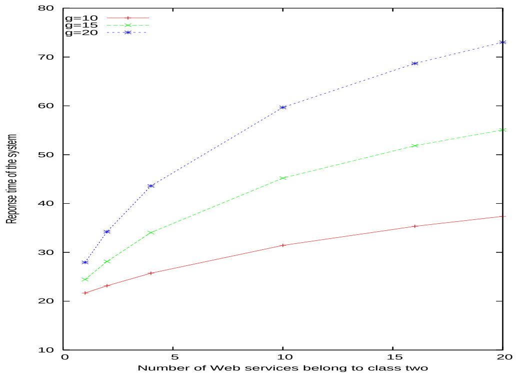

Fig. 3. Response time for composite Web service

VII. CONCLUSION

Web services rely on open protocols and standards like eXtensible Markup Language (XML); Simple Object Access Protocol (SOAP); Universal Description, Discovery and Integration (UDDI); and Web Services Description Language (WSDL). Composite Web services combine the power of existing component Web services to form new Web services. One challenge of these composite Web services is the guarantee of the Quality of Service (QoS) for an adhesion of the clients. In this paper we are interested to the response times of composite Web service. We have proposed analytical formulas for mean response times for structured BPEL constructors such as *sequence*, *flow* and *switch*. We have also given a response time formula for a generalized case such as *multi-choice* pattern. We have proposed an extension of literature results. First, the generalization for the case where the number of invoked elementary Web services is random. Second, we have considered that the server times follow exponential distribution with different rates. In perspective, we plan to consider the case of no exponential distribution times for elementary web services but we will consider heavy tailed distribution and we will give the analytical formulas for BPEL constructors. Also, we will consider more complex composite web services.

REFERENCES

[1] D.A. Menasc, "Response-time analysis of composite Web services", IEEE Internet computing, vol. 8, No. 1, pp. 90-92, 2004

[2] M. Sharf, "On the response time of the large-scale composite Web services", Proceedings of the 19th International Teletraffic Congress (ITC 19), Beijing, 2005

[3] S. Hahn and L. Fatto, "Two parallel queues created by arrivals with two demands", Applied Mathematics, Vo. 44, pp. 1041-1053, October, 1984

[4] R. Nelson and A.N. Tantawi, "Approximate Analysis of Fork/Join Synchronization in Parallel Queues", IEEE Transaction Computer, vol. 37, No. 6, pp. 739-743, 1998

[5] F. Curbera, "Business Process Execution Language for Web Services", Version 1.0, IBM, 2002

[6] F. Bacelli, A.M. Makowski and A. Shwartz, "The fork and join Related Systems with Synchropnization constraints: Stochastic Ordering and Computable Bounds", Applied Probability, pp. 629-660, July, 1983

[7] F. Bacelli and A.M. Makowski, "Simple computable bounds for the fork-join qeue", Proceeding Information Scinece, pp. 436-441, March, 1985

[8] D. Davis and M. P. Parashar, "Latency Performance of SOAP Implementations", IEEE Computer Society, pp. 407-415, 2002

[9] S. Paul, P.G. Santiago, K. Kobuske, H. Marc and P. Eduardo, "Fast Web Services, Web site: http://wwww.java.sun.com/developper/technicalArticles/WebServices/fastWS/", 2006

[10] C. Kohlhoff and R. Steele, "Evaluating SOAP for High Performance Business Applications: Real-Time Trading Systems", In Proceedings of WWW'03, Budapest, Hungry, 2003

[11] G. Kiczales, "Aspect-oriented programming", ACM Comput. Surv., pp. 154, http://doi.acm.org/10.1145/242224.242420, 1996

[12] M.U.Bhatti, S.Youcef, L.Mokdad and V.Menfort, "Simulation-based Response Time Analysis of composite Web Services", In Proceedings 10th IEEE international Multitopic conference, INMIC, pp. 1-7, juin 2006

[13] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros, "Workow patterns", Technical report FIT-TR-2002-2, Faculty of IT, Queensland University of Technology, July 2002. Accessed from http://www.tm.tue.nl/it/

[14] F. Curbera and M. Duftler, R. Khalaf, W. Nagy, N. Mukhi and Sanjiva Weerawarana, "Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI", IEEE Internet Computing, vol. 6, No. 2, pp. 86-93, 2006

[15] "Web Services Description Language, WSDL", Web site at: http://www.w3.org/TR/wsdl, http://www.w3.org/TR/wsdl, 2007

[16] "The SOAP specification, which is published (and endorsed) by the World Wide Web Consortium, is available on-line at http://www.w3.org/TR/SOAP/", http://www.w3.org/TR/wsdl, 2007

[17] Heiko Ludwig, "Web Services QoS: External SLAs and Internal Policies - Or: How do we deliver what we promise", citeseer.ist.psu.edu/699287.html

[18] Daniel A. Menasce, "QoS Issues in Web Services, IEEE Internet Computing, Vol. 6, No. 6, 2002, issn. 1089-7801, pp. 72-75, http://dx.doi.org/10.1109/MIC.2002.1067740, IEEE Educational Activities Department, Piscataway, NJ, USA

[19] Anbazhagan Mani and Arun Nagarajan,"Understanding quality of service for Web services", http://www-106.ibm.com/developerworks/library/ws-quality.html, 2002

[20] Baskett, F. and Chandy, K.M. and Muntz, R.R and Palacios-Gomez, F., "Open, closed and mixed networks of queues with different classes of customers", J. ACM, Vo 22, No 2, pp. 248-260, 1975

[21] D. A. Menasce and V. Almeida, "Capacity Planning for Web Services: metrics, models, and methods", 2001

[22] P. Wohed, W.M.P. van der Aalst, M. Dumas and A.H.M. ter Hofstede, "Pattern based analysis of BPEL4WS", technical report FIT-TR, 2002