http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 9 (3): 422-429, 2010
ISSN 1812-5638

Improving QoS of Web Service Composition by Dynamic Configuration

[1,2]Huaizhou Yang and [1]Zengzhi Li
[1]Institute of Computer Architecture and Networks, Xi'an Jiaotong University, Xi'an 710049, China
[2]State key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract: In order to achieve self-adaptation to run-time environment and self-management of Web service composition, an extended hierarchical Petri net is presented to reflect the feasible configuration scheme of composite service. Being different from related works, the dynamic configuration scheme and execution process of Web service composition can be reflected together in the extended model. Furthermore, a rapid QoS calculation strategy of composite service is presented. Some important QoS calculation formulas are presented and proved. Based on the extended model and the QoS calculation strategy, a configuration with optimal QoS can be selected dynamically. The QoS of composite service is improved. The soundness and correctness of the dynamic configuration method are proved by simulation results and corresponding analysis.

Key words: Web service composition, dynamic configuration, Petri Net, QoS improvement

INTRODUCTION

A Web service composition system is comprised of a number of component services, which are usually orchestrated to an executable workflow by a workflow language, such as Business Process Execution Language for Web Service (BPEL4WS). Being different from traditional component system, there often exist a number of alternative component services, which have same functionality but differ in QoS. Although, the problem of Web service selection has been extensively researched in the past few years (Kobti and Wang, 2007; Huang *et al*., 2009), most works regard each service selection as an independent process. In the actual applications, there often exist functional dependent relationships between component services. A service selection is very likely to influence the next service selection and even execution flow. For example, if a certain ticket booking service is selected in a service composition for travel plan, then the acceptable payment services, such as the acceptable banks and types of credit cards, are often restricted. Another example is that an activity of composite service can be accomplished by different component service sets with different quantities of services and execution flows. A component service set should be selected as a whole in this situation.

The service community, which is comprised of the total component services, keeps evolving as new services are added into it. Most services are third party services whose QoS is not guaranteed. In the internet-based environment, because the qualities of the component services are subject to change with the variation of workload or communication delay, the QoS of composite service is fluctuating and the malfunction of component service can disable a composite service. Therefore, the Web service composition system must have the ability of dynamic reconfiguration in order to adapt to the dynamic change of run-time environment (Yen *et al*., 2008).

Reconfigurability is an important feature of self-adaptive system, especially high-assurance system. The problem of dynamic Web service selection in a failure-prone environment is studied by Hwang *et al*. (2008). The Web services are selected dynamically at run time in their work. A composite service configuration scheme is modeled by Petri net (Xiong *et al*., 2008). The composite service can be dynamically reconfigured along with the QoS change of component services. These works have made a good job in modeling of service dependent relationship. However, these solutions remain hard to be integrated with current service orchestration engines. A main reason is the orchestration process of composite service is not considered in their models. The QoS of composite service correlates closely to its flow structure. Two composite services, which are comprised of the same component services, have different QoS when their orchestration processes are different. Without considering the orchestration process, it will be difficult to well evaluate whether the user's QoS requirements are satisfied or not.

Corresponding Author: Huaizhou Yang, Institute of Computer Architecture and Networks, Xi'an Jiaotong University, Xi'an 710049, China

Many theories, such as FSM (Finite State Machine) (Lei and Duan, 2007), Pi-calculus (Liao *et al*., 2005) and Petri nets (Huang *et al*., 2009) have been applied to describe the Web services composition. However, most of them concentrate on model mapping, i.e., modeling through translating composition plan or language into model. They scarcely deal with how to facilitate the QoS improvement. Some works concentrate on performance prediction of composite service (Wang and Tian, 2009; Punitha and Babu, 2008). They are only suitable for the evaluation of different design alternatives and can not satisfy the requirements of adaptive and automatic system management. In order to construct a self-adaptive composite service, different algorithms, such as evolutionary algorithm (Milani *et al*., 2008) and simulated annealing (Liu *et al*., 2009) are presented to accelerate service global selection. However, the service dependent relationship is not considered in these algorithms.

Being different from the related works, we focus on the unitized modeling of composite service configuration scheme and service orchestration process. The service dependent relationships and the possible orchestration processes are synthetically reflected by an extended hierarchical Petri net presented in this study. The model can be easily updated along with the evolution of service community. In order to calculate the QoS of composite service under different configurations, a series of QoS calculation formulas are presented. Different flow structures and QoS attributes are considered in these formulas. The QoS expressions are stored in advance in order to be used by the optimal configuration selection algorithm. By using the rapid QoS calculation strategy, the configuration with optimal QoS can be selected dynamically. The fault-tolerance ability and the QoS of composite service are improved. The final configuration and orchestration process can be directly used by the current composite service engines.

The main contribution of this study is two fold: (a) a unitized modeling method of composite service configuration scheme and orchestration process and (b) a rapid QoS calculation strategy for some important QoS attributes.

THE MODELING OF DYNAMIC CONFIGURATION SCHEME AND ORCHESTRATION PROCESS

Definition 1: Extended Hierarchical Petri Net (EHPN): An EHPN is a tuple $(P, T, F, K, W, M_0, \Gamma_c)$, where:

- P is a finite set of places. T is a finite set of transitions. $P \cup T \neq \emptyset$, $P \cap T = \emptyset$
- $T = T_d \cup T_c \cup T_m$, $T_d \cap T_c = \emptyset$, $T_c \cap T_m = \emptyset$, $T_d \cap T_m = \emptyset$. T_d is a set of dummy transitions, T_c is a set of concrete transitions, T_m is a set of macro transitions. $\forall t_d \in T_d$, t_d can be associated with a choice probability. $\forall t_m \in T_m$, t_m is an EHPN with unique input transition $t_i$and unique output transition t_0. $T_i, t_0 \in T_d$
- $F \subseteq (P \times T) \cup (T \times P)$ is finite set of arcs
- $K: P \rightarrow N^+$ is a capability function. N^+ is the set of positive integers
- $W: F \rightarrow N^+$ is a weight function
- $M_0: P \rightarrow N^+$ is the initial marking, N is the set of natural numbers
- $\Gamma_c: T_c \rightarrow R^+$ is a QoS function, R^+ is the set of positive real numbers

In contrast with basic Petri net, EHPN classifies the transitions into three types, dummy transitions, concrete transitions and macro transitions. Their meanings are explained after Definition 2. A service configuration scheme net is constructed via EHPN.

Definition 2: Service Configuration Scheme Net (SCSN): An EHPN is a SCSN if following conditions are satisfied:

- The model has unique input place p_i and unique output place p_0. $p_i, p_0 \in P$
- $M_0(p_i) = 1$. $\forall_p \in p \setminus \{p_i\}$, $M_0(p) = 0$
- The model is an ordinary Petri net, i.e., all of its arcs weights are 1
- $|T_m| \geq 1$. The model has at least one macro transition
- Every concrete transition is associated with a component service
- Every macro transition model is constructed by a fork-and-join structure

The SCSN is a mixed model of service selection and orchestration process. The system states and the execution conditions are represented by places. The logical control activities in the composition process are represented by dummy transitions. No component service is associated to a dummy transition. The activities of component services are represented by concrete transitions. Only one fixed component service is associated to a concrete transition. The activities of service selection are encapsulated in the macro transitions. The macro transition indicates that an abstract sub-function of composite service can be accomplished by a number of optional component services or service modules. Each branch of fork-and-join structure in a macro transition represents a service selection. The QoS attribute is associated to concrete transitions.

Definition 3: Composite service configuration: A composite service configuration is a final model of SCSN. Only one selection branch in every macro transition is retained in the final model.

There are multi possible configurations in a SCSN. A feasible configuration can be get after one selection branch is retained and others are eliminated in every macro transition. Because different configuration is likely to include different number of component services, the corresponding orchestration process can also be different. Therefore, in contrast with the models of composite service configuration presented in related works (Milani *et al*., 2008; Xiong *et al*., 2008), SCSN reflects not only the possible service selections but also the corresponding orchestration process.

The process to associate every activity of composite service to a component service is called service selection. According to whether a service selection is related to another, the service selections can be classified into two kinds, free selection and restricted selection. As shown in Fig. 1a and b, the two kinds of selections can be modeled by a fork-and-join structure of Petri net. In Fig. 1a, every selection branch only includes one transition. Therefore, the transitions in Fig. 1a can be selected freely. In Fig. 1b, every selection branch includes multi transitions. The service dependent relationships are reflected by the flow structure of branch. The transitions in a branch must be selected as a whole. For example, the set $\{t_1, t_2\}$ or (t_3, t_4) is selected.

A service orchestration model with static configuration can be transformed to one with dynamic configuration. For example, in Fig. 1c, if there are a number of component services with identical function, either of which can accomplishes the function of t_1, then t_1 is transformed to a macro transition T_1 and T_1 is modeled as a selection structure as shown in Fig. 1a. If either t_2 or t_3 can be accomplished by multiple component services, but there exists mutual restriction between two selections, then t_2 and t_3 are transformed to macro transition T_2 and T_2 is modeled to a selection structure as shown in Fig. 1b. The transformation of merging transitions is based on the fact that the restricted selection relationships usually exist between adjacent sub-functions of service orchestration process. For example, a purchase activity is often followed by a payment activity. If a certain purchase service is selected, then the acceptable payment services are restricted, such as the acceptable banks and types of credit cards.

Some important modeling issues need to be explained.

- In Fig. 1b, one selection branch can include different flow structure and different quantities of transitions in contrast with another, as long as each branch can accomplish the same function. This enables the model to reflect the various service orchestration processes. For example, if some component services form a sub-composite-service which can accomplish an abstract system sub-function, then the sub-composite-service model can be added to corresponding macro transition as a new selection branch
- The macro transitions can exist in a nesting manner, i.e., there probably exist other macro transitions in a macro transition. A complex composite service configuration scheme can be compactly reflected by the hierarchical model
- The composite service configuration scheme and optional orchestration processes are synthetically reflected in a SCSN. Only one selection branch in every macro transition is retained in the final model

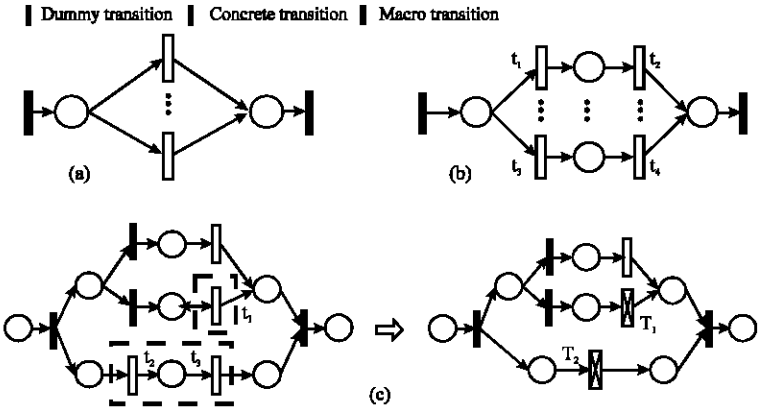

Fig. 1: A modeling example of dynamic configuration by using SCSN, (a) The service selection without dependent relationship, (b) the service selection with dependent relationship and (c) the transformation from static configuration to dynamic configuration

THE SELECTION OF DYNAMIC CONFIGURATION

Here, a rapid QoS calculation strategy of composite service is presented at first. Furthermore, an optimal configuration selection algorithm is designed to select the configuration with best QoS from the service configuration scheme.

The calculation strategy of composite service QoS: Many QoS attributes, such as response time, throughput, cost, reliability, availability, security, accessibility and so on, have been presented to evaluate a Web service. Each component service may have several QoS attributes. To throughput, reliability, availability, security and accessibility, the larger is the value, the better is the QoS. To response time and cost, the smaller is the value, the better is the QoS. In this study, we suppose smaller value indicates better QoS. To larger-is-better attributes, we change their values to negative number. If a component service is unavailable or disabled, the QoS value of it is set to $+\infty$.

Because a selection branch in a macro transition must be selected as a whole, the QoS of the branch needs to be calculated. Without losing the generality, different branch structures need to be considered. As shown in Fig. 2a-d, SCSN is usually comprised of four kinds of basic Petri net structures. The α is a choice probability associated to dummy transition. To different QoS attributes, the QoS calculation formulas of four kinds of structures are different. As shown in Table 1, we propose some feasible estimate formulas to calculate the QoS of basic model structures. $R_t(t)$, $C(t)$, $R_a(t)$ and $T(t)$ are response time, configuration cost, reliability (or availability) and throughput of component services, respectively. The configuration cost indicates the charge being paid for the third-party component services. It can be simply the summing up of all the costs of component services (Xiong *et al*., 2008). Two QoS formulas of loop structure are proved as follow. The other formulas are easy to be understood and verified.

Theorem 1: In a loop structure with a forward transition t_1 and a backward transition t_2, if the probability of t_2 executing is α, then:

- The response time of the loop structure is:

$$\frac{R_t(t_1)+\alpha R_t(t_2)}{1-\alpha}$$

- The reliability or availability of the loop structure is:

$$\frac{(1-\alpha)R_a(t_1)}{1-\alpha R_a(t_1)R_a(t_2)}$$

Proof: As shown in Table 2, the probabilities of loop times and corresponding values of QoS parameters are listed. The sum of these probabilities is:

$$\sum_{n=0}^{\infty}\alpha^n(1-\alpha)=(1-\alpha)\sum_{n=0}^{\infty}\alpha^n=(1-\alpha)\times\frac{1}{1-\alpha}=1$$

The response time of the loop structure is:

$$\begin{aligned}
&\sum_{n=0}^{\infty}\alpha^n(1-\alpha)\{R_t(t_1)+n[R_t(t_2)+R_t(t_1)]\}\\
&=\sum_{n=0}^{\infty}\alpha^n(1-\alpha)R_t(t_1)+\sum_{n=0}^{\infty}\alpha^n(1-\alpha)n[R_t(t_2)+R_t(t_1)]\\
&=(1-\alpha)R_t(t_1)\sum_{n=0}^{\infty}\alpha^n+(1-\alpha)[R_t(t_2)+R_t(t_1)]\sum_{n=0}^{\infty}n\alpha^n\\
&=(1-\alpha)R_t(t_1)\times\frac{1}{1-\alpha}+(1-\alpha)[R_t(t_2)+R_t(t_1)]\alpha\sum_{n=1}^{\infty}n\alpha^{n-1}\\
&=R_t(t_1)+(1-\alpha)[R_t(t_2)+R_t(t_1)]\alpha\times\frac{1}{(1-\alpha)^2}\\
&=\frac{R_t(t_1)+\alpha R_t(t_2)}{1-\alpha}
\end{aligned}$$

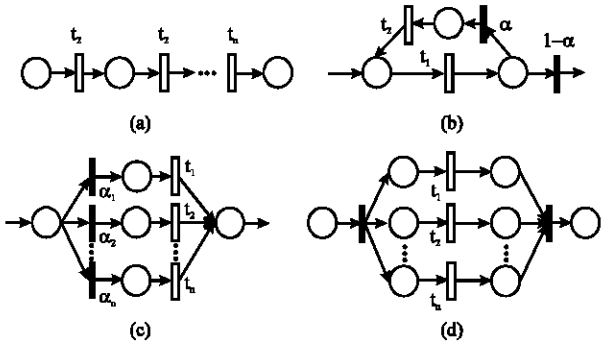

Fig. 2: The basic flow structures of SCSN, (a) sequence, (b) loop, (c) choice and (d) concurrent

Table 1: The QoS calculation formulas of basic model structures

QoS parameters	Sequence	Loop	Choice	Concurrent
Response time	$\sum_{i=1}^{n}R_t(t_i)$	$\frac{R_t(t_1)+\alpha R_t(t_2)}{1-\alpha}$	$\sum_{i=1}^{n}\alpha_i R_t(t_i)$	$\mathrm{Max}\{\bigcup_{i=1}^{n}R_t(t_i)\}$
Configuration cost	$\sum_{i=1}^{n}C(t_i)$	$C(t_1)+C(t_2)$	$\sum_{i=1}^{n}C(t_i)$	$\sum_{i=1}^{n}C(t_i)$
Reliability or availability	$\prod_{i=1}^{n}R_a(t_i)$	$\frac{(1-\alpha)R_a(t_1)}{1-\alpha R_a(t_1)R_a(t_2)}$	$\sum_{i=1}^{n}\alpha_i R_a(t_i)$	$\prod_{i=1}^{n}R_a(t_i)$
Throughput	$\mathrm{Min}\{\bigcup_{i=1}^{n}T(t_i)\}$	$(1-\alpha)T(t_1)+\alpha\mathrm{Min}\{T(t_1),T(t_2)\}$	$\sum_{i=1}^{n}\alpha T(t_i)$	$\mathrm{Min}\{\bigcup_{i=1}^{n}T(t_i)\}$

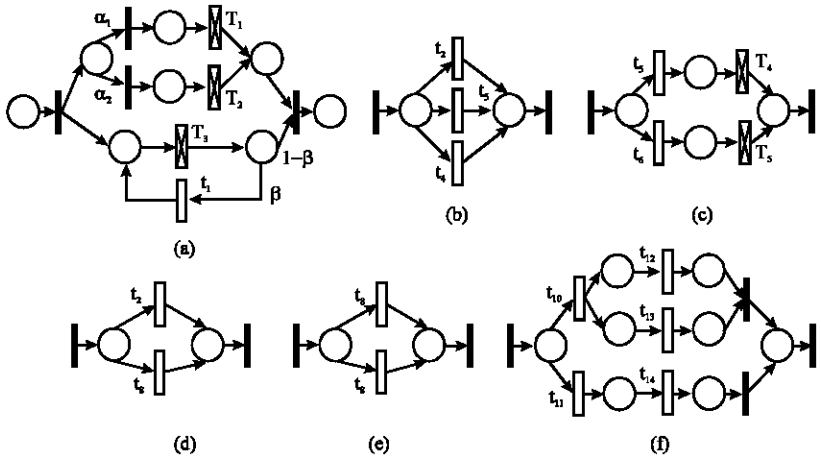

Fig. 3: An example of SCSN, (a) the main model, (b) the model of T_1, (c) the model of T_2, (d) the model of T_4, (e) the model of T_5 and (f) the model of T_3

Table 2: The list of probabilities of loop times

Loop times	Probability	Response time	Reliability or availability
0	$1-\alpha$	$R_t(t_1)$	$R_a(t_1)$
1	$\alpha(1-\alpha)$	$R_t(t_1)+R_t(t_2)+R_t(t_1)$	$R_a(t_1)R_a(t_2)R_a(t_1)$
2	$\alpha^2(1-\alpha)$	$R_t(t_1)+2[R_t(t_2)+R_t(t_1)]$	$R_a(t_1)[R_a(t_2)R_a(t_1)]^2$
…	…	…	…
n	$\alpha^n(1-\alpha)$	$R_t(t_1)+n[R_t(t_2)+R_t(t_1)]$	$R_a(t_1)[R_a(t_2)R_a(t_1)]^n$
…	…	…	…

The reliability or availability of the loop structure is:

$$\sum_{n=0}^{\infty}\alpha^n(1-\alpha)R_a(t_1)[R_a(t_2)R_a(t_1)]^n$$
$$=(1-\alpha)R_a(t_1)\sum_{n=0}^{\infty}[\alpha R_a(t_2)R_a(t_1)]^n$$
$$=(1-\alpha)R_a(t_1)\times\frac{1}{1-\alpha R_a(t_2)R_a(t_1)}$$
$$=\frac{(1-\alpha)R_a(t_1)}{1-\alpha R_a(t_1)R_a(t_2)}$$

Based on the formulas above presented, the QoS of a SCSN can also be expressed in a hierarchical manner. For example, suppose that the SCSN of a composite service is one as shown in Fig. 3a-f, when the selection branch with minimal response time is selected in every macro transition, the response time of the composite service can be expressed as expression 1-6. In the model of T_2 (Fig. 3c), there exist nesting macro transitions T_4 and T_5. The selection branches in T_4 and T_5 are selected at first and then $\{t_5, T_4\}$ or $\{t_6, T_5\}$ is selected in T_2. In the model of T_3 (Fig. 3f), there are two selection branches, which have different flow structures $\{t_{10}, t_{12}, t_{13}\}$ or $\{t_{11}, t_{14}\}$ is selected in T_3.

(1) $$R_t(SCSN)=\text{Max}[\alpha_1R_t(T_1)+\alpha_2R_t(T_2),\frac{R_t(t_1)+\beta R_a(T_3)}{1-\beta}]$$

(2) $$R_t(T_1)=\text{Min}[R_t(T_1),R_t(T_2),R_t(T_3)]$$

```
Input:  A SCSN
Output:  An optimal QoS configuration
Begin
S = Ø ' S is a stack of macro transitions.
   For each t ∈ Tm(SCSN) t.selected = False
   For each t ∈ Tm (MainModel) Push t into S
   Do While (S ≠ Ø)
   Ti = Pop(S)
   If |Tm(Ti)| = 0\/ (∀t ∈ Tm (Ti, t.selected = True)) Then
   Calculate QoS of every selection branch by formulas stored in
       advance.
   Select the branch with minimal QoS value and set QoS of Ti with
       this value.
       Ti.selected = True
     Else
     Push Ti into S
       For each t ∈ Tm(Ti) Push t into S
   End If
   Loop
End
```

Fig. 4: The optimal configuration selection algorithm

(3) $$R_t(T_2)=\text{Min}[R_t(t_5)+R_t(T_4),R_t(t_6)+R_t(T_5)]$$

(4) $$R_t(T_4)=\text{Min}[R_t(t_7),R_t(t_8)]$$

(5) $$R_t(T_5)=\text{Min}[R_t(t_8),R_t(t_9)]$$

(6) $$R_t(T_3)=\text{Min}\{R_t(t_{10})+\text{Max}[R_t(t_{12}),R_t(t_{13})],R_t(t_{11})+R_t(t_{14})\}$$

These expressions can be stored in advance in order to be used by the optimal configuration selection algorithm. They also can be used to estimate or predict the QoS of composite service. When the model is updated, only limited expressions need to be updated.

The optimal configuration selection algorithm: As shown in Fig. 4, an optimal configuration selection algorithm is designed to achieve the dynamic reconfiguration of composite service and enable the

system to adapt to the dynamic change of component service QoS. In the algorithm, T_m(SCSN) is the set of all macro transitions in SCSN. T_m(MianModel) is the set of macro transitions in the main model (Fig. 3a). $T_m(T_i)$ is the set of macro transitions in the macro transition T_i.

Because the macro transitions can exist in a nesting manner, the QoS calculation and the branch selection of the inner macro transitions must be accomplished at first. In the algorithm, a macro transition stack S is used to achieve the depth-first search to the macro transitions and to accomplish the calculation and selection to the inner macro transitions at first.

THE EFFICIENCY EXPERIMENTS OF DYNAMIC CONFIGURATION

Here, in order to evaluate the efficiency and the practicality of the dynamic configuration method presented in this study, three typical service configuration modes are contrasted in the experiments:

- Global dynamic configuration
- Local dynamic configuration
- Static configuration

Global dynamic configuration: When the component services are selected, the dynamic change of QoS and the dependent relationships between component services are considered synthetically. The QoS are compared between different selection branches instead of between individual component services. The component services in a selection branch with optimal QoS are selected as a whole. The dynamic configuration method presented in this study belongs to this kind of mode.

Local dynamic configuration: When there are a number of component services, either of which can be selected to accomplish an abstract sub-function of composite service, the component service with optimal QoS is selected. In this mode, the service selections depend on the local QoS information. The dependent relationships between component services are not considered.

Static configuration: The composite service is comprised of some fixed component services.

The simulation experiments are made on the SCSN model in Fig. 3a-f. The response time is selected as the testing QoS attribute. Suppose that the response time of component services changes dynamically and conforms to a normal distribution, the corresponding concrete transitions and their response time distribution parameters are shown in Table 3. The μ and σ are the mean and standard deviation of response time, respectively. The choice probabilities in the model are set to $\alpha_1 = 0.5$, $\alpha_2 = 0.5$, $\beta = 0.3$.

Table 3: The list of response time distribution parameters

	Response time													
Parameters	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}
μ	8	19	22	25	10	12	13	15	11	9	8	14	13	12
σ	2	4	5	6	3	4	3	5	3	2	2	4	3	3

The experiments and algorithms are achieved by a VB.Net program. The transition set $\{t_1, t_2, t_5, t_7, t_{11}, t_{14}\}$ is selected as the static configuration, which has the minimal mean of response time. The experiment data are acquired and processed by the following steps:

- By using the function randn() in Matlab7.0, one hundred groups of simulation data are created for t_1–t_{14} according to their distribution parameters in Table 3. The total number of these data is 100×14. Every group of data represent the response time of component services at a certain time. These data are stored in text form in order to be used by experiment program
- To every group of data, the corresponding optimal service configuration is determined according to global and local dynamic configuration modes, respectively. The optimal configuration selection algorithm is used in the decision process for optimal configuration
- The response time of composite service on global dynamic configuration, local dynamic configuration and static configuration are calculated, respectively. We get one hundred groups of data. The total number of these data is 100×3. Every group of data represent the response time of composite service in different configuration modes at a certain time

In order to contrast the QoS of different configuration modes, two measurement indexes are set:

- QoS threshold. The QoS threshold is the worst QoS user can tolerate. When the QoS of composite service is worse than the threshold, the system can not satisfy the user's QoS requirement
- The satisfaction rate of service request. Let n be the quantity of user's service requests during a period. Let n' be the quantity of service requests whose QoS requirements have been satisfied. The satisfaction rate of service request is n'/n

When the QoS threshold is set to 35, i.e., the response time of composite service is required to be less than 35, we get the simulation result as shown in Fig. 5.

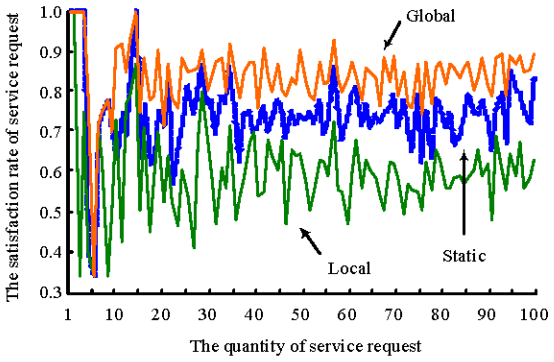

Fig. 5: The simulation result when QoS threshold is 35

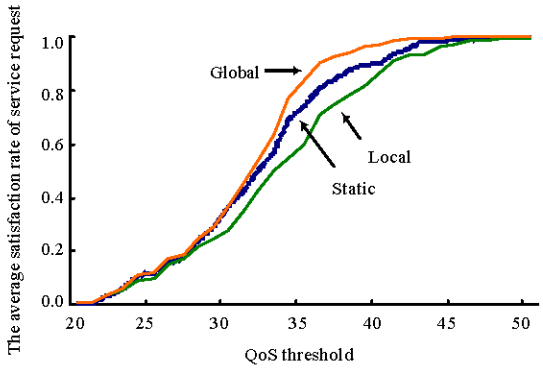

Fig. 6: The average satisfaction rate under different QoS thresholds

The global dynamic configuration has the best satisfaction rate, whose mean is 83.62% and standard deviation is 7.97%. The mean and standard deviation of satisfaction rate in static configuration are 74.19 and 9.71%, respectively. The mean and standard deviation of satisfaction rate in local dynamic configuration are 59.39 and 11.16%, respectively.

The simulation results in Fig. 6 reflect the adaptability of three configuration modes to different QoS requirements. Along with the increase of QoS threshold, the satisfaction rates in three configuration modes also increase. However, the global dynamic configuration has the best satisfaction rate under any QoS threshold. The satisfaction rate of static configuration is better than that of local dynamic configuration. A reason is that the component services with minimal mean of response time are selected to construct the static configuration in the experiment. Another key reason is that the dependent relationships between component services are not considered in local dynamic configuration. In contrast with global dynamic configuration and local dynamic configuration, the static configuration has not fault tolerance feature.

CONCLUSION AND FUTURE WORK

By using the SCSN and the rapid QoS calculation strategy presented in this study, a global dynamic configuration of composite service can be achieved. The configuration scheme of composite service can be easily updated along with the evolution of service community. By taking advantage of the global dynamic configuration, the composite service can adapt to the QoS dynamic change of component services and failure-prone run-time environment. The QoS of composite service is improved. The QoS requirements of users are satisfied to the greatest extent. Service configuration optimization will become a hot research topic. The key problem is how to find an optimal configuration under the multiple functional and non-functional constraints.

A problem is left and still needs to be resolved in the future work. A composite service usually has several QoS attributes. When considering the different QoS attributes, we most probably get different optimal configurations. How to find a suitable configuration to balance the multiple QoS requirements is a difficult and important problem. It will be researched in our future work.

ACKNOWLEDGMENTS

This research was sponsored by National Natural Science Foundation of China NSFC under the Grant No. 60673170 and Beijing University of Posts and Telecommunications State Key Laboratory of Networking and Switching Technology.

REFERENCES

Huang, A.F., C.W. Lan and S.J. Yang, 2009. An optimal QoS-based web service selection scheme. Inform. Sci., 179: 3309-3322.

Hwang, S.Y., E.P. Lim, C.H. Lee and C.H. Chen, 2008. Dynamic web service selection for reliable Web service composition. IEEE Trans. Services Comput., 1: 104-116.

Kobti, Z. and Z. Wang, 2007. An adaptive approach for QoS-aware web service composition using cultural algorithms. LNCS, 4830: 140-149.

Lei, L. and Z. Duan, 2007. An extended deterministic finite automata based method for the verification of composite web services. Chinese J. Software, 18: 2980-2990.

Liao, Z., H. Tan and J. Liu, 2005. Describing and verifying web service using Pi-calculus. Chinese J. Comput., 28: 635-643.

Liu, Q., S.L. Zhang, R. Yang and X.J. Lian, 2009. Web services composition with QoS bound based on simulated annealing algorithm. Chinese J. Southeast Univ., 24: 308-311.

Milani, A., C.H.L. Leung, M. Baioletti and S. Suriani, 2008. An evolutionary algorithm for adaptive online services in dynamic environment. LNCS, 4974: 626-632.

Punitha, S. and C. Babu, 2008. Performance prediction model for service oriented applications. Proceedings of 10th IEEE International Conference on High Performance Computing and Communications, Sept. 25-27, IEEE Computer Society, Washington, DC, USA., pp: 995-1000.

Wang, K. and N. Tian, 2009. Performance modeling of composite web service. Proceedings of Pacific-Asia Conference on Circuits, Communications and System, (PACCS'09), Chengdu, China, pp: 563-566.

Xiong, P.C., Y.S. Fan and M.C. Zhou, 2008. QoS-aware web service configuration. IEEE Trans. Syst. Man Cybernetics, 38: 888-895.

Yen, I., H. Ma, F.B. Bastani and H. Mei, 2008. QoS-reconfigurable web services and compositions for high-assurance systems. Computer, 41: 48-55.