Modeling and Analyzing Adaptive Energy Consumption for Service Composition

Guisheng Fan[1,2], Huiqun Yu[1], Liqiong Chen[3]
[1]Department of Computer Science and Engineering
East China University of Science and Technology, Shanghai 200237, China
[2]Shanghai Key Laboratory of Computer Software Evaluating and Testing, Shanghai 201112, China
[3] Department of Computer Science and Information Engineering
Shanghai Institute of Technology, Shanghai 200235, China
Corresponding author:lqchen@sit.edu.cn

***Abstract*—In this paper, Petri nets are used to model the different components of service composition, and form the energy consumption model of service composition based on the relationship between components, Agent is also introduced in the energy consumption management process. Then, an adaptive energy consumption strategy are proposed to dynamically ensure that service composition can get the lowest energy consumption. The operational semantics and related theories of Petri nets help establish the correctness of our proposed method. We have also performed two simulations to evaluate our proposed approach. Results show that it can help reveal the structural and behavioral characteristics of energy consumption in service composition.**

***Index Terms*—Service composition, Agent, Petri nets, energy consumption**

I. Introduction

Service composition provides a mechanism for distributed software integration, which primarily concerns the requests of users that cannot be satisfied by any available service[1]. This increased usage of service composition, together with the increasing energy costs[2]. The energy management is highly complex. First, energy consumption is inherently complicated due to the scale, heterogeneity, and concurrent user services that share a common set of service. Second, the operating environment of service composition is dynamic, it is difficult to plan energy consumption schema at design time. When service composition starts to execute, composition can not achieve the goal because of wrong design of composition or can not meet the required energy consumption. The failure will result in a waste of computing resources and increase energy consumption. Therefore, it requires that energy consumption management of service composition must have a certain adaptive ability, which means that service composition can dynamically select the service, in order to complete its execution successfully and meet the required energy consumption.

Contributions. This paper investigates how to model and analyze the energy consumption of service composition based on user requirements. Below summarizes our main contributions: (1) Petri nets[3] are used to model different components of service composition. Agent[4] is introduced in the energy consumption management process. (2) We evaluate the energy consumption of service composition from the operational level(in addition to service, component, connector and Agent). Based on the model, we propose an adaptive energy consumption strategy to dynamically ensure that service composition can get the lowest energy consumption. (3) The operational semantics and related theories of Petri nets help establish the effectiveness of our proposed method.

The rest of this paper is organized as follows. Section II presents our energy consumption evaluation, and Section III describes how we model the different components of service composition. Next, we show how to analyze the constructed model(Section IV), and then evaluate the proposed method via simulation in Section V. Finally, Section VI surveys related work, and Section VII concludes.

II. Energy consumption evaluation of service composition

A. Requirements of service composition

As the function of service composition is composed of a number of independent tasks according to a certain composition rules. Each task has a number of available services. The energy consumption of service under each state can be got by testing and simulation. The interaction between components, services, Agents, and subsystems is defined as the connector, and the connector is viewed as the basic element of service composition, so the connector can be seen as service. The energy consumption of connector is considered.

Definition 1: The energy consumption requirement of service composition is a 7-tuple: Ξ =(WS, AG, AL, RE, TW, RA, OP):

(1)The set of service WS = $\{WS_1, WS_2, \ldots, WS_n\}$, WS = ($type$, op, ra, ep, ei), $type$, op are the type and operations of service, $ep(op)$ = (etp, enp) is the execution time and per unit of energy consumption of the operation. ei = (ty, $func$) is the set of service interface.

(2) The connector AL =$\{al_1, al_2, \ldots, ag_n\}$, al= ($data$, oa, nl,li), $data$ is the transmission content, oa is the associated service or Agent, nl is the energy consumption, li is used to describe the input and output interface.

(3) The set of Agent AG = $\{ag_1, ag_2, \ldots, ag_n\}$, ag = (eg, lg, om, op, gi), $eg \subseteq WS$, $g \subseteq AL$ is the set of service and connector of ag; $om : AG \times RA \rightarrow \{Start, Sleep, reV, raV\}$ is the possible operations that Agent can do for service, op is the set of operation of Agent, gi is the set of interface of ag, ag may have several input and output interfaces.

(4) The set of component C = $\{C_1, C_2, \ldots, C_n\}$, C = (et, tp, rl), $et : C \rightarrow WS^*$ is the set of available service of C_i, $tp : C \rightarrow op^*$ is the set of operations that need be realized, rl is a relation function between the components, the main

(DOI reference number: 10.18293/SEKE2015-236)

relationship is sequence(>), choice(+), parallel relationship (‖). $VW(WS_i, C_j) = \{C_k | WS_i \in et(C_k) \cup C_k \in C_j\}$ is the set of operation of WS_i for component C_j.

In service composition, the service will realize its function by executing a series of operations. Because the operation of service may be different, which will make the energy consumption of service be different too. Therefore, the energy consumption analysis based on service level may cause large errors. This paper will describe the energy consumption of service composition based on the operation level.

B. Energy consumption evaluation

Because the component in service composition may invoke different services to realize its function, which will make the reachable states of energy consumption model be different. Let C_i invoke $OP(WS_j, C_i)$ of WS_j to realize its function, we will evaluate the energy consumption in the following.

(1)Energy consumption of component C_i

The energy consumption of the operation $op_f \in OP(WS_j, C_i)$ of component C_i is $en(op_f) = etp(op_f) \times etp(op_f)$.

The energy consumption of component C_i is:

$$EN(C_i) = \sum_{op_f \in op(ae_j, tk_i)} en(op_f) \tag{1}$$

(2)Energy consumption of service

The energy consumption of service is:

$$EN(WS_j) = \sum_{C_i \in C(ae_j)} EN(C_i) \tag{2}$$

The average energy consumption that service using to realize the function of component is:

$$avg\_EN(WS_j) = \frac{EN(WS_j)}{|C(WS_j)|} \tag{3}$$

(3)Energy consumption of Agent and service composition

The energy consumption of Agent is composed by service and connector, so the energy consumption of Agent is:

$$EN(ag_j) = \sum_{WS_j \in WS_{ac} \cap WS(ag_i)} EN(ae_j) + \sum_{la_j \in AE_{ac} \cap lg(ag_i)} en(la_j) \tag{4}$$

The energy consumption of service composition is:

$$EN = \sum_{ag_i \in AG} nl(ag_i) = \sum_{ae_j \in AE_{ac}} EN(ae_j) + \sum_{la_j \in AE_{ac}} en(la_i) \tag{5}$$

We can quantitatively evaluate the energy consumption of service composition by using the above formula.

III. Modeling service composition

A. Modeling service composition

In this section, we will use Petri nets to model different components of service composition, then construct the energy consumption model of service composition based on its execution process. In addition, we mark the service, Agent, connector in the front of place and transition.

a) Modeling service: The model of service is modeled as following, p_i^I, p_{ie}^I, p_o^O, p_{dt}^O are used to describe the startup, suspending, running and overtime interface, t_i, t_e and t_{ie} are used to describe the startup, finished and suspending operation. The execution of op_i in service is: $t_{a,i}$ is used to describe the execution of the operation, $ct(p_{a,i}) = etp(op_i)$ is the execution time of operation, $en(t_{a,i}) = enp(op_i)$. If the service is in the overtime or suspended state, then invoke the transition t_p to make the operation be in the interrupted position $p_{p,i}$. If the transition can re-execute ($p_{r,i}$), then invoke transition $t_{r,i}$ to make the operation be in the available position ($p_{w,i}$).

b) Modeling component: The model of component is shown in Fig.1: First, we introduce p_w to store the set of available service, $\forall WS_j \in WS$, there is $d_j^w \in M_0(p_w)$. If C_i gets the input parameter $p_{s,i}$, then invoke the transition $t_{pp,i}$ to allocate the appropriate service for C_i, and the component will be in the waiting for execution(P_{wa}^i). If C_i gets the input parameter $p_{s,i}$, then no available service can realize the function of component C_i and $t_{fa,i}$ is used to output the fault.

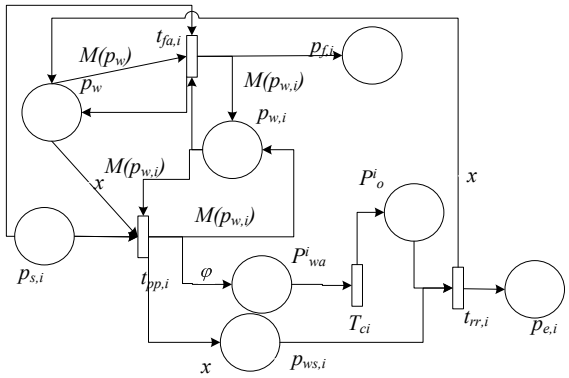

Fig. 1. The model of component

c) Modeling connector: The model of connector is shown in Fig.2, let two basic elements of connector be A and B. Then the system will introduce two interfaces to send and receive the info from A, B.

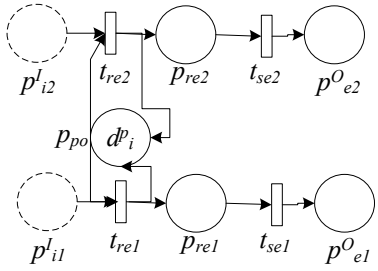

Fig. 2. The model of connector

d) Modeling Agent: The model contains the execution of component and connector in Agent, that is, the execution environment of Agent. Agent will get the information from the environment and compute the startup service based on the adaptive energy consumption strategy. t_{st} is used to start the service, while t_{ns} is used to make the remaining service be in the suspended state. t_e represents the termination of Agent.

The service will do the related adjustment when it receives the instruction of Agent.

The steps for constructing energy consumption model are. Constructing the model of services based on their attributes. Introducing t_{st} and p_{st} to describe the beginning operation and position. t_{en} and p_{en} are used to describe the termination operation and position of whole application. Computing the set of startup service by using the adaptive strategy, and initializing the place in the model. Setting initial marking $M_0(p_s) = \varphi$, while setting the priority of transition t_{dt} and transitions in the system level be 4. In this paper, we divide the priority of transition into 5 level, which can be adjusted according to the actual requirements.

IV. Adaptive energy consumption strategy and analysis

In this section, we will propose an adaptive energy consumption strategy for service composition based on Agent. Then dynamically compute the set of startup service based on the state and attributes of the current service.

A. Adaptive energy consumption strategy

A series of operations that service ws is used to realize the function of C_i are called the path of ws. Let the path that WS_i uses to realize the function of component C_j be $Lat(WS_i, C_j)$=$\{Lat_1, Lat_2, \ldots, Lat_n\}$, LaC=$\{op_{k,1}, op_{k,2}, \ldots, op_{k,f}\}$. The energy consumption of path Lat_i is:

$$En(Lac) = \sum_{ap_{k,l} \in Lat_k} enp(op_{k,l}) \times etp(op_{k,l}) \tag{6}$$

The average energy consumption that service WS_i uses to realize the function of component C_j is:

$$Avg\_en(wsi, C_j) = \frac{\sum_{Lat_k \in Lat(ae_i, TK_j} en(Lat_k)}{|Lat(ae_i, TK_j|} \tag{7}$$

B. Analysis technique

In this section, we will verify the correctness of proposed method based on the reachable states of energy consumption model and the operation semantics of Petri nets.

Theorem 1. *Let $R(\Omega)$ be the reachable state of energy consumption model Ω, $\forall S \in R(\Omega), \forall WS_i \in WS$, the set of startup service is WS_{ac}, then:*

(1) If $WS_i \in WS_{ac}$, $\exists S \in R(\Omega)$, which makes $|M(WS_{i\bullet}p_e)| = 1$.

(2) If $WS_i \notin WS_{ac}$, $\exists S \in R(\Omega)$, which makes $|M(WS_{i\bullet}p_p)| = 1$.

Proof: If $WS_i \in WS_{ac}$, we may set $WS_i \in eg(ag_j)$, that is, WS_i belongs to the corresponding Agent. We can set $ag_{j\bullet}P_i^i \in ag_{j\bullet}t_{st}^{\bullet}$ in the modeling process of ag_j, therefore, $S_1 \in R(S_0)$, which makes $M_1(ag_{j\bullet}p_i^I) \neq \emptyset$ and $M_1(ag_{j\bullet}p_{ie}^I) = \emptyset$, because $M_0(ag_{j\bullet}p_{st})$=$(d_j^a,1)$ and $ag_{j\bullet}p_{st}^{I\bullet}$=$ag_{j\bullet}t_{st}$, therefore, $M_1(ag_{j\bullet}p_{st})$=$(d_j^a,1)$. Form the modeling process of service, we can get that transition $WS_{j\bullet}t_i$ can be fired, while $WS_{j\bullet}t_{ie}$ isn't enable. Therefore, $\exists S \in R(\Omega)$, which makes $|M(WS_{i\bullet}p_e)| = 1$. Similarly, we can prove (2) establishes from the modeling process that Agent does the suspended operation for service. ■

Theorem 1 explains that the energy consumption model can correctly describe the interaction between the Agent and service, such as, the system get the info of service, Agent startups and suspends the service.

Theorem 2. *Let $R(\Omega)$ be the reachable state of energy consumption model Ω, $\forall S \in R(\Omega), \forall WS_i \in WS$, then:*

(1) If $|M(WS_{i\bullet}p_p)| = 1$, $\exists S' \in R(\Omega)$, which makes $|M'(WS_{i\bullet}p_e)| = 1$.

(2) If $|M(WS_{i\bullet}p_{p,j})| = 1$, $\exists S' \in R(\Omega)$, which makes $|M'(WS_{i\bullet}p_{g,j})| = 1$.

Proof: $\forall WS_i \in WS$, because $|M(WS_{i\bullet}p_p)|$= 1, therefore, ws is in the suspended state under S. Let WS_i restart after a period of time, We will prove the suspend of WS_i will not affect the restart of it, we can prove it from two two aspects: First, we will prove WS_i can be in the running position again. Because $WS_{i\bullet}p_i^{I\bullet}$ = $\{WS_{i\bullet}t_i, WS_{i\bullet}t_r\}$, ${}^{\bullet}WS_{i\bullet}t_r$ = $\{WS_{i\bullet}p_i^I, WS_{i\bullet}p_p\}$, so $WS_{i\bullet}t_r$ has the right to fire and the priority of $WS_{i\bullet}t_r$ is 0. Therefore, the firing of $WS_{i\bullet}t_r$ under S is effective, we may set that S_1 will reach S_2 by firing $WS_{i\bullet}t_r$, then there is $|M_2(WS_{i\bullet}p_w)|$ = $|M_1(WS_{i\bullet}p_i^I)| = 1$, because ${}^{\bullet}WS_{i\bullet}t_i$ = $\{WS_{i\bullet}p_i^I, WS_{i\bullet}p_w\}$, therefore, the firing of $WS_{i\bullet}t_i$ under S_2 is effective. Second, we can prove that all operations can be executed properly, $\forall op_f \in op(WS_i)$ may be in $p_{a,f}$ or $p_{e,f}$ when WS_i is suspended. Therefore, $\exists S' \in R(S)$ which makes$|M'(WS_{i\bullet}p_e)|$ = 1.

We can prove the sub-proposition(2) in the similar way. ■

Theorem 2 illustrates that the service and its operations can execut properly when Agent needs to restart the suspended service, thus realizing its function.

V. Examples

In order to evaluate the effectiveness of proposed method, we will use experimental platform Windows 7, C# language to implement a prototype for analyzing the constructed model. First, we will generate available services as service resource. Each service at least has the basic information such as energy consumption, the function and the set of operation, etc.

Experiment 1. The goal of this experiment is to analyze the effectiveness of adaptive strategy, the specific steps are:

(1) Taking 400 services, 50 components, 10 Agents and dividing them into 10 groups, each group corresponds to a Ligo system, then do Step (2)-(3) to each Ligo system.

(2) Selecting 10 groups of service, then compute the energy consumption of each Ligo system[5] based on each set of available service;

(3) Using adaptive strategy to compute the set of startup service of service composition, then compute the whole energy consumption of each system based on the set of startup service.

The results of Experiment 1 are shown in Fig.3, we can get: (1) the energy composition of the set of service will change when the attributes of service has changed. (2) the overall energy consumption of service composition can be reduced by using the adaptive strategy, the highest reduction is 34.5% , and the lowest reduction is 3 %.

Experiment 2. Experiment 2 analyzes the relationship between the state space and available service, the steps are:

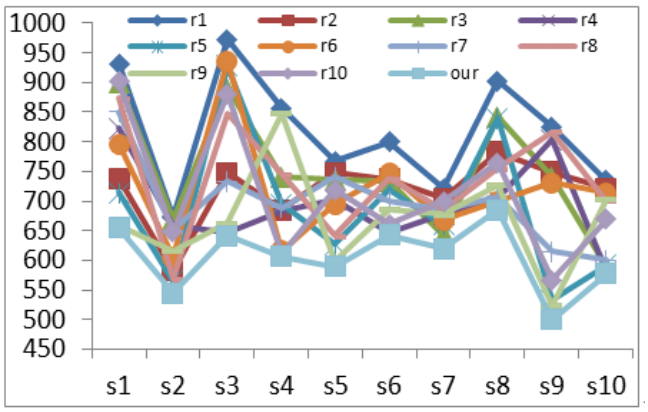

Fig. 3. Experimental results of Experiment 1

(1) Taking 20 services, 5 components, 1 Agent to construct the resource of Ligo system;

(2) Adding the service to system by 10 each time, that is, 30, 40, 50, 60, 70, 80, 90, then compute the set of startup service and the number of reachable states of model.

From the results of Experiment 2, we can get that: the state space of energy consumption model is 32 in all cases, that is, the increase of available service will not effect the state space of energy consumption model.The state of model will not increase with the available service increasing. The proposed method can be used to model and analyze the energy consumption of large scale service composition.

VI. Related Works

There is a vast amount of research available on adaptive designs for different areas. The work in [6] presents a system architecture to monitor, interpret and analyze system events in order to implement self-healing and self-adaptive systems. The architecture presented in [7] is focused on service level agreement management in a Service Oriented Architecture. The approaches defined in the above don't consider the different components of service composition.

Agent system design has emerged as a powerful approach to perform tasks or solve problems in a decentralized environment. A framework for building an adaptive Learning Management System has been proposed in [8]. Reference [9] introduces an Agent-oriented Model-Driven Architecture. Agents use hierarchical business knowledge models with business process rules at the top, business rules to control policy and logic in the middle and a base layer defining business concepts. Later, the authors design a flexible method that supports a range of coordinator components[10]. However, the approaches defined in the above don't consider the operation of service, and they don't involve in how to filter the service.

Many research efforts for service oriented computing have adopted formal methods techniques to leverage its mathematically precise foundation for providing theoretically sound and correct formalisms. A novel method for runtime monitoring of composite services is proposed by [11], they employ process algebra as the primary formalism to express specifications. A similar approach is given by [12], the authors present a CSP-based workflow framework for intelligently navigating service composition. We have proposed an approach to constructing the reliable service composition[13]. Almost all of the aforementioned formalisms cover basic and structured activities of service composition, but they are unable to ensure that the constructed model can meet the users' specific requirements, such as energy consumption. Meanwhile, the approaches defined in the above can not solve the problem of dynamically selecting available service.

VII. Conclusion

In this paper, we proposed new solutions for optimizing energy consumption for service compositions. The special features of the proposed model include: (1) Petri nets are used to describe different components of service composition. Agent is introduce to the energy consumption management process of service composition. (2) The adaptive energy strategy is proposed, which is used to dynamically select the available service that meets the requirement for component, thus reducing the energy consumption of service composition. (3) The operational semantics and related theories of Petri nets help prove the correctness of proposed method. Finally, we conduct several experiments to evaluate the effectiveness of the proposed method.

Acknowledgment

The work is partially supported by the NSF of China under grants No. 61173048 and 61300041. Research Fund for the Doctoral Program of Higher Education of China under Grants No. 20130074110015. The Fundamental Research Funds for the Central Universities under Grant No.WH1314038.

References

[1] S. Marstona, Z. Lia, S. Bandyopadhyaya, et al. Cloud computing-the business perspective. Decision Support Systems. 2011, 51(1): 176-189.

[2] C. Wang, M. de Groot, P. Marendy. A Service-Oriented system for optimizing residential energy use. IEEE International Conference on Web Services(ICWS 2009), IEEE Computer Society, Washington, DC, USA, 2009:735-742.

[3] M. TADAO. Petri nets: properties, analysis and application. Proceedings of the IEEE. 1989, 77(4):540-581.

[4] C. Antonio, C. Massimo, G. Salvatore, V. Seidita. Agent-Oriented software patterns for rapid and affordable robot programming. Journal of Systems and Software, 2010,83(4):557-573.

[5] G. Juve, A. Chervenak, E. Deelman, et al. Characterizing and profiling scientific workflows. Future Generation Computer Systems, 2013, 29(3): 682-692.

[6] I. Al-oqily, B. Subaih, S. Bani-Mohammad, et al. A survey for self-healing architectures and algorithms. Processings of the 9th International Multi-Conference on Systems, Signals and Devices (SSD), IEEE Computer Society, Washington, DC, USA, 2012: 1-5.

[7] R. R. Aschoff, A. Zisman. Proactive adaptation of service composition. In: processing of the 2012 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), IEEE Computer Society, Washington, DC, USA, 2012: 1-10.

[8] Y. Mahkameh, B. Ardeshir. A context-aware adaptive learning system using agents. Expert Systems with Applications, 2011,38(4): 3280-3286.

[9] X. Liang, G. Des. Agent model: software adaptivity using an agent-oriented model-driven architecture. Information and Software Technology, 2009,51(1):109-137.

[10] J. Lian, S. Shatz, X. He. Flexible coordinator design for modeling resource sharing in multi-agent systems. Journal of Systems and Software, 2009,82(10):1709-1729.

[11] K. Mohsen, J. Saeed. WSCMon: Runtime monitoring of web service orchestration based on refinement checking. Service Oriented Computing and Applications. 2012,6(1):33-49.

[12] X. Song, W. Dou, J. Chen. A workflow framework for intelligent service composition. Future Generation Computer Systems. 2011, 27(5): 627-636.

[13] G. Fan, H.Yu, L.Chen, D.Liu. Petri net based techniques for constructing reliable service composition. Journal of Systems and Software, 2013, 86(4): 1089-1106.