International Journal of u- and e- Service, Science and Technology
Vol.9, No. 2 (2016), pp.127-136
http://dx.doi.org/10.14257/ijunesst.2016.9.2.14

A Service Pipeline Based On Service Composition Model

Weng Yu and Cheng Wenyi

School of Information Engineering, Minzu University of China
Beijing, China
dr_wengyu@126.com

Abstract

With the development of computer science and software engineering, the technology of the service composition has already become the mainstream of the application integration in the Internet environment of distributed interactive technology. Even though, there still be some problems that in some certain occasion. Pipeline technology has been used widely in industry level and micro instruction level, while service pipeline has not draw as much attention as the former. In this paper, the authors combine service composition technology and pipeline technology, and propose a service pipeline based on service composition model. The model can not only reuse variety of hardware and software, but also improve the flexibility of service platform greatly. The model has been validated in the experimental system.

Keywords*: service composition, service pipeline, service workflow*

1. Introduction

Recent years, more and more organizations and institutions tend to integrate their advantage resource and high-quality application into a variety of forms of service to open to the public as the developing of new concepts such as web service, cloud computing, pervasive computing, etc [1].

In the software service platform, we can get more value-added services and make the system flexibility by using service composition technology. With small services together into large size with more business meaning, users only need to aware of the interface and function of the service, instead of knowing the composition and structure of the complex service. At present, the service composition has already become the mainstream of the application integration in the Internet environment of distributed interactive technology [2].

The service component model is not only conducive to code reuse, and makes system integration easier and more flexible, the late maintenance more convenient and cost less. In the service component model ,each function module are encapsulated as components, or become a plug-in. Hara and Fujita proposed a proxy-based approach of software component reuse way which make the software components be reused as the middleware [3]. Maolin Tang and Lifeng Ai proposed a service composition method based on genetic algorithm under constraints, and achieved good results in the evaluation experiments [4]. Chao Ma and Yanxiang He propose a service composition visual and formal tool which improves the usability of service composition [5].

Since the source data varies, there is still problems in the multi-source data service composition and providing flexible service:①Single model such as Map-Reduce are not suitable for all the types of data processing . Map Reduce model based on the idea of divide and conquer, decentralized operation and then merge the results, this model does not meet the requirement of real-time application and the process of complex operations. ②With the accumulation of years of research, it often requires a lot of repetition of previous work, when the existing technology to new platforms or migrate to new system.

ISSN: 2005-4246 IJUNESST

③User demand changes and cannot be accurately predicted. Different users may have different current demand because of its own characteristics and different situation. Even to same user, the demand will gradually change because of its own development[6].The change of user requirements is hard to predict, to make the service on demand, flexible service delivery is very important to service platform[7].

In order to solve the above problems in the combination of multi-source data service platform, and low cost integration of all aspects of the existing resources, improve the efficiency of data processing, build a quick and flexible service platform, this paper proposed a service composition model based on service pipeline. The model relay on a a library of basic services, connect distributed environment and running basic services with segment system, to provide variety of complete service services in service pipeline.

Although there will be bottlenecks and run conflict in pipeline technology, proper use of this technology can significantly improve the throughput of the system, improve the efficiency of the system. It is an effective method to improve system performance and efficiency to build the application or machine level pipeline. Mean while, organizing data and computing resources in the form of services in a distributed environment can integrate all aspects of resources security and flexibility.

In the chapter 3, we describe the main problems and general process, and then propose solutions and design principle in chapter 4. The chapter 5 will verify the efficiency of the model through the design contrast test. Chapter 6 of this paper makes brief summary the research content, and proposed the future work.

2. Problem Description and General Analysis

In the design and implementation process, we must properly handle the following problems:

① How to package the service so that it can be run as a segment of the pipeline;
② How to solve the problem of the pipeline running time;
③ How service composition management.

2.1. Service Pipeline

Service pipeline is a pipeline system based on the service component. A service component serves as a segment of the service pipeline. A number of service components associated business build up a pipeline ,which offer service to top and improve the overall efficiency and throughput of the system.

The basic of Service pipeline system is a lot of service components. The best service is the one which has relatively simple functions and small run-time services are called basic services. And they are called basic service.. And the time of basic services in a single run should not be too long, because one or more of the basic services constitute a segment of the pipeline and the time needed to run each segment has a significant impact on the efficiency of the pipeline.

We construct a basic services library to meet the requirements of service components，integrate the reusable resources and make varieties of servers fusion and standard. The basic services library acquires basic services components by the way of segmentation and encapsulation. Concentrated mainly in the following ways:

Firstly, package the existing independent function module. Many operation functions can be implemented by different forms, such as Java run-time in jar format, DLL dynamic link library of windows environment, so dynamic link library and the Linux environment and the NET run-time, etc. It is difficult to integrate in the same environment, because they have respective operation environment and different standard external interface. For those existing resources, we can encapsulate them and exposed unified standard interface to make it run in a distributed environment set up to run the necessary environment of

hardware devices, in a distributed environment for the system to provide standardized services as a whole. In this way we can reuse these modules and reduced the cost of re-coding greatly.

Secondly, package the existing open or has been authorized to localized Web service encapsulation. Different web services provide ways to call and interface specification which is not the same. In one system, a web service will often use multiple times, and may be used to multiple web service. The standard of these web services such as interface and call parameters are set by the service provider. And the user can't ensure that these specifications will stand still. Therefore, the local packaging services, unification of upper interface specification, providing services to upper system can reduce the dependencies of system on the stability of external characteristics. At the same time, it also can enhance the system security.

Thirdly, refined service when add services. A service can often be refined according to the function service.M_1 M_2, M_3, M_4... M_n were refined into services C_1, C_2, C_3, C_4,... C_n, then the total number of basic services C_{total} <= C_1+C_2+ C_3+C_4+... +C_n. This is because that the service M_i and service M_j required in function or functions may overlap. Basic services will be repeated after refining. The repeated basis services do not have to be repeated in the foundation library. When the basic service is enough big, you just need to add a few basic services even needn't add basic services and implement the service according to based service composition in the basic service library when adding services.

This method can not only make full use of existing software and hardware resources, but also isolate the user and the service form, the underlying hardware deployment. It can provide unified standard services for users. In principle, each service component in service pipeline can run on a separate machine nodes. In order to make different service components run on the machine nodes controllable and can exchange the necessary information with other nodes, It need a section of the system serve as an intermediate layer segment.

Segment system is deployed on each calculation node, which makes the machine and the services running on the machine work like a segment. Functional mode pipeline segment system is shown in figure 1.

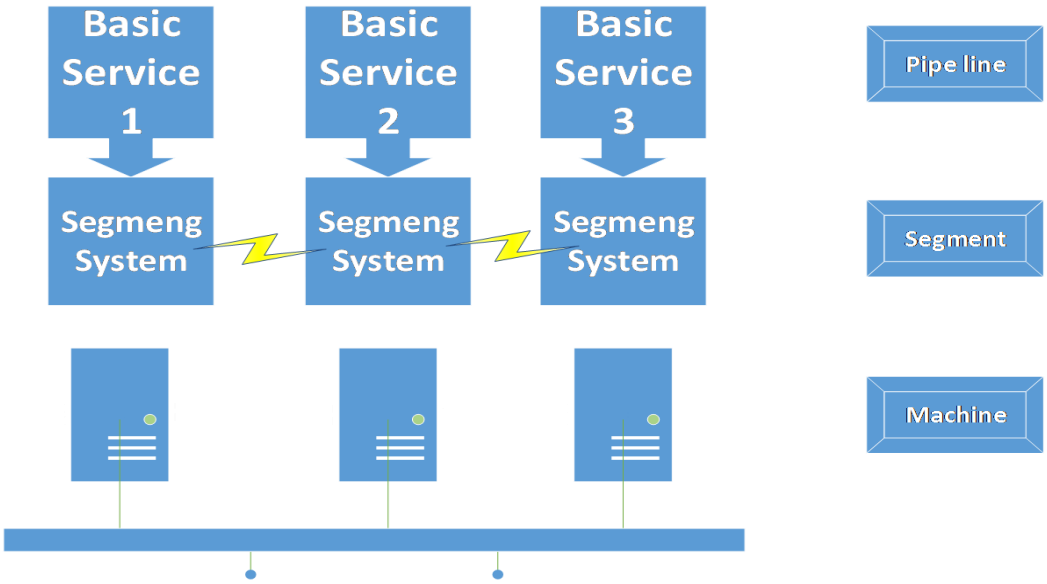

Figure 1. An Example of Segment System

Segment system isolates the hardware and service, which is very important layer in the entire system. Some basic services require a large memory to run, which is suitable for running in the configuration memory deployed. Some services require the high floating

point arithmetic, which can be deployed in better arithmetic processor machines first. Some services need to access external networks (such as calling an external web service), and you may need to have a dual-network machine. The right combination of services and machinery can take the best advantage of hardware and software.

A service can be divided into a number of basic services according to the principle of the service composition. Responsibilities pipeline scheduling system is to realize the pipeline, organizational lines, pipeline static optimization and dynamic adjustment of the pipeline. Service composition system should avoid foreseeable conflicts and avoid the loop when it designed the pipeline. Pipeline scheduling program monitors pipeline running time, dynamic detect bottlenecks and used repetition to open the bottleneck.

2.2. Service Composition

A complex service will be split performed by a number of basic services. The service maybe has some sub-chains, which may constitute a service pipeline [8]. So the service chain has been merged into multiple several service pipelines and basic services.

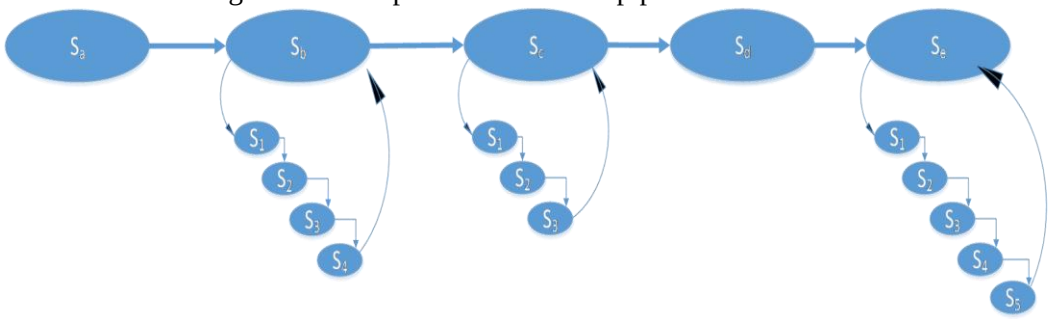

Figure 2. Refinements and Service Composition

A virtual service node will proxy a service strings which were merged, and perform a service on the top. Pipeline system for the upper layer is transparent. When the service was managed, we can manage a pipeline by proxy node. When we look for the sub-strings which can constitute a pipeline in the basic service chain, we can use the forward maximum matching method to scan the service chain and find out the basic service pipelines. Process is as follows:

① Take the m services of the service chains to be slit from left to right as the matching string .And m is the number of basic services of longest pipeline in the services pipelines table which the system can support.

② Look up the services pipelines table and match successfully. If the match is successful, the match string will be considered as a service pipeline and be syncopated out. It also will be designated the agent node. If the match is unsuccessful, this match has the last word removed and the rest of the string as a new matching string match again, repeat the above process until all service lines cut separation of strings so far.

2.3. Summaries of Problems

The main problem of service pipeline is how to build a pipeline and how to deal with the problem in run-time. The service component serves as a segment pipeline. There is a great fluctuation in run time according to the different input data, so it will affect the efficiency of the pipeline. Services composition needs not only the flexible, but also to cope with service pipelines. Therefore method of division and combination is a better choice. We propose three main problems and analysis them one by one. Based on our analysis, the service composition model is shown in figure 3:

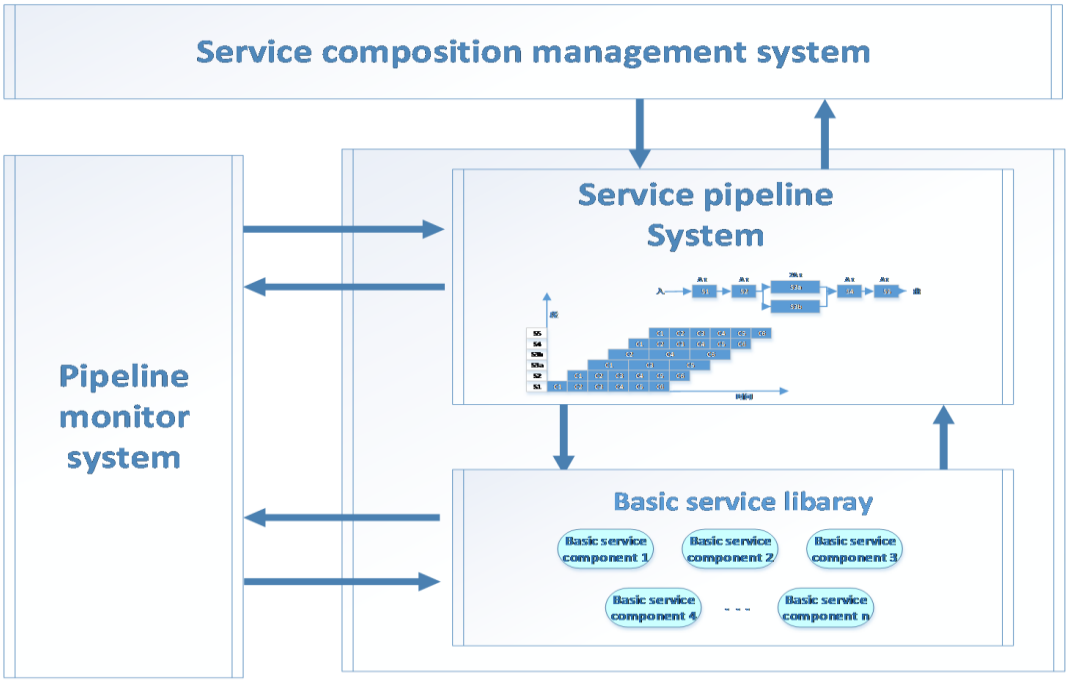

Figure 3. Service Composition Model Based on Service Pipelines

3. A Service Composition Model based on Service Pipeline

3.1. Service Pipeline System

The service pipeline reuses existing software by repackaging to build a basic service library. The basic services will be called by java code with JNI technique. The repackaged services should not only implement certain interfaces, but also mark run-time features of the service such as usage of the system resource, necessary run-time environment etc... Services in basic service library are loaded in a segment system to work as a segment in a pipeline. The segment system of service pipeline system is composed by input module, output module, processing module, and control module.

The input module has two input buffer to deposit the input data of the task to be processed. There is a receive thread in input module to deal with the send request from previous segment. The request will be accepted only when there is at least one buffer is empty. The received data will be stored in the buffer whose idle time is the longest. The buffer Ia and Ib constitute a circular queue, and the task execution in strict accordance with the first-in, first-out strategy.

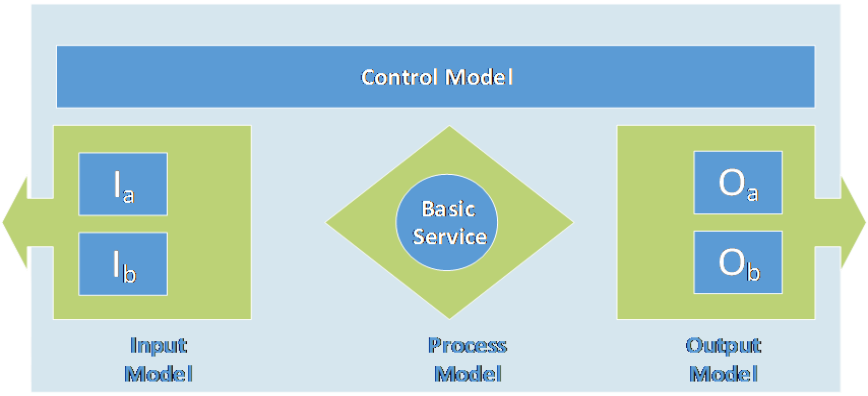

Figure 4. The Architecture of Service Pipeline

Output module includes two output buffer, temporary storage module processing after processing data, and send to the next section of the system in order. Buffer O_a and O_b is also a Circular queue, the data will be sent in accordance with the first-in, first-out strategy. When exist non-empty buffer, output module send a data transfer request to next segment, and the send the older buffer to next segment after get "accept" sign. Output module accept output data from process module only when there exits at least one buffer is empty. The new received data will be stored in the buffer whose data is send earlier.

The process module should get permission of output from output module first, and fetch input data from input module next. The control module is responsible for coordinating the three modules work together, and feedback nodes' status to monitor node in monitoring pipeline. Monitoring information transmitted by XML. After the service pipeline system syncopate the handle of large data into multiple smaller and more fundamental services, some of the hardware or service had special requirement in operating system can be deployed to the machines which to meet the requirement. These nodes can be used to deal with data of these type and services, so that the hardware resources can be fully used.

3.2. Service Pipeline Dispatching System

The pipeline scheduling system is responsible for the communication and connection between the two coordination segment systems. Service pipeline scheduling system itself deployed on the platform as a system-critical service. The proxy nodes of service pipeline use XML to record and configure the running of pipeline [9]. Each calculation node and the proxy node in pipeline get feedback of information by the mechanisms of questions and answers. Proxy node will visit all the nodes of its pipeline at intervals actively. The nodes asked feedback timely node status of the node running, including memory usage, recent long run of a task, the data flow, and other information. A simple feedback XML is shown in figure 5.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<node>
    <Id>3</Id>
    <Ram>800/1024</Ram>
    <TimeForLastTask>3518</TimeForLastTask>
    <DataTraffic>4862</DataTraffic>
    <CurrentStatus>Busy</CurrentStatus>
    <!-- .... -->
</node>
```

Figure 5. The Feedback Information of Pipeline Node

Proxy node retrieves the operational status of the current node by the feedback of XML [10]. According to the comprehensive analysis of operational status data, proxy node of can dynamically schedule the nodes in pipeline.

4. Validation Experiments

4.1. Deployment Environment

The experimental model is deployed in the distributed network environment which is constituted by several machines. Distributed environment is the classic master-slave architecture. Master node is responsible for monitoring the operation of the entire system and dispatching system resources. Slave nodes deployed by a variety of basic services, or equipped with a virtual external services. The system operating environment and the external network are disconnected, so the internal cluster refuse directly access to external networks. Because the basic service library contains some of the external network web Service existed and some services need to feedback data to the service requester, all machines equipped with these services need to have the ability to access external networks.

Comparative system will be deployed in a preferred configuration server. In addition, in order to further study the performance of the system, the system will achieve several extensions. Extended system logic structure as shown above, but cluster size expanded. We chose Web GIS [11, 12] as our demo service. It was deployed on a cluster of 10PC. Following the principle of “equal cost”, we designate a workstation whose price close to our 10 PC. The demo system will be extended into 12PC, 24PC, and 36 PC. The Expansion system will compare with the original system.

4.2. Experiment Results and Analyses

In the "equal costs" principle, we verify the performance of the system by comparing the degree of concurrency, scalability, response time, stability.

Table 1. Overall System Performance Statistics

	Number of Machines	Average Response time	Maximum Concurrency	Cost Accounting
Comparison System	1	1.33	1200	￥22,000
The original system	12	0.53	6200	￥22,000
Expansion System 1	24	0.45	9500	￥44,000
Expansion System 2	36	0.37	1400	￥66,000

As shown in table 1, the performance of GIS services based pipeline significantly better than traditional service model. And the larger the cluster, the system response time is shorter. We also found that, with the expansion of the cluster size, a single PC system performance gains will be reduced. Taking the problems that expansion of cluster will bring energy consumption and equipment costs into account, in the application, we need to find a balance between performance and cost and choose the appropriate size to gain the performance met the requirements. The stability is different in different scale clusters. You can assess the stability of the system by comparing the different scales' clusters with varying average response time in stable condition under concurrency.

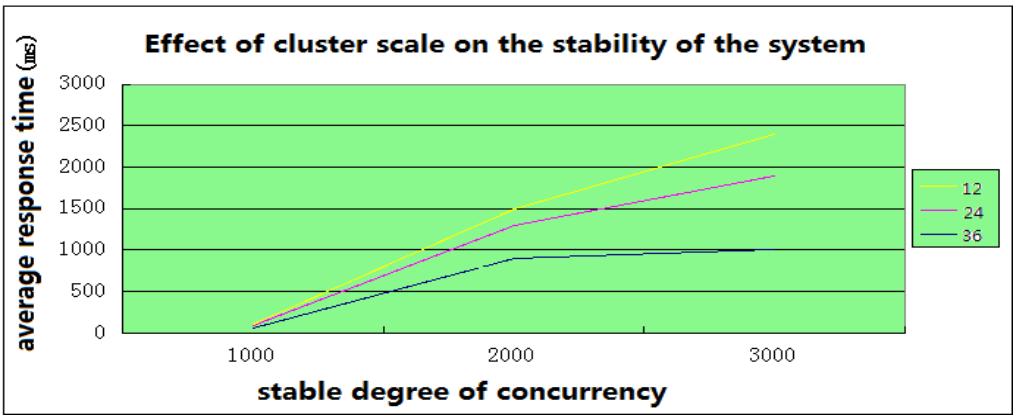

Figure 6. the Effect of Cluster Scale on the Stability

The greater clusters scale becomes and the more frequent the concurrency occurs, the more slow the system average response time increase. In other words, the larger the scale of cluster becomes, the stronger the stability of the system is. We also found a rule which is similar to the previous table. With the expansion of the cluster, the enhancement of stability which a single PC brought to cluster is reduced, too.

In summary, the performance of service composition model based on service pipeline is superior to traditional service model. With cluster size increasing, system performance will increase and the stability of the system will be enhanced. With the cluster scale expanded, the growth of the stability will decrease gradually.

5. Conclusions & Future Work

This article proposed a service pipeline l based onservice composition mode. As a part of model, basic service library is built by packaging the existing reusable software module and Web Service. The service composition use agent-based management of the service composition isolated the upper and lower of system, making the system efficient and easy to use. An evaluation demo system is built to validate the performance. Experimental results show that the proposed model is superior to the traditional service model, can greatly improve the performance and stability of the system.

According to the experimental test data, we will save energy and get the most cost-effective in the next step. At the same time, more applications will be deployed on this platform, such as the ancient southwest minority GIS, GIS and other minority traditional sports, data storage services, statistics service and ancient architectural panorama database applications.

Acknowledgements

Thank all the members of the Central University for Nationalities Information Engineering College 908 laboratory support for the system. Especially Fenghao Joe , Ziwei Huang, and Peng Guo. they provide a lot of valuable suggestions and ideas in systems analysis and design time. This paper is supported by “Construction of a comprehensive database of Development and a information service platform for Minority Areas”, “State Ethnic Affairs Commission-Leading talents supported projects”, “Ministry of Education of the People's Republic of China-Supported by Program for New Century Excellent Talents in University”.

References

[1] A. O. Akingbesote, "A quality of service aware multi-level strategy for selection of optimal web service, International Conference on Adaptive Science and Technology (ICAST), **(2013)**.
[2] H. M. T. A. Ariyarathna and D. N. Ranasinghe, "JXTA based parallel service invocation model for peer to peer web service composition", 7th IEEE International Conference on Industrial and Information Systems (ICIIS), **(2012)**.
[3] H. Hara, S. Fujita and K. Sugawara, "Reusable software components based on an agent model", Seventh International Conference on Parallel and Distributed Systems, **(2000)**.
[4] T. Maolin and A. Lifeng, "A hybrid genetic algorithm for the optimal constrained web service selection problem in web service composition", IEEE Congress on Evolutionary Computation (CEC), **(2010**
[5] C. Ma and Y. He, "An Approach for Visualization and Formalization of Web Service Composition", International Conference on Web Information Systems and Mining, **(2009)**.
[6] S. Mokarizadeh, "Utilizing Web Services Networks for Web Service Innovation", IEEE International Conference on Web Services (ICWS), **(2014)**.
[7] W. Yong, "An actor-based language to unifying web service orchestration and web service choreography", International Conference on Computer Science and Information Processing (CSIP), **(2012)**.
[8] Z. Zibin, "Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization", Services Computing, IEEE Transactions, vol. 6, no. 3, pp. 289-299.
[9] L. Yan, "Folksonomy-Based In-Depth Annotation of Web Services", IEEE 8th International Symposium on Service Oriented System Engineering (SOSE), **(2014)**.
[10] Z. Yilei, "WSPred: A Time-Aware Personalized QoS Prediction Framework for Web Services", IEEE 22nd International Symposium on Software Reliability Engineering (ISSRE), **(2011)**.
[11] D. Xia, X. Xie and Y. Xu, "Web GIS server solutions using open-source software", Software for Scientific Computation (OSSC), **(2009)**.
[12] L. Tao, L. Meng, J. Fang, J. Li, Z. Chen and D. Chen, "Research and Realization of Web GIS Framework Based on XML", International Workshop on Education Technology and Training, **(2008)**.

Authors

Weng Yu, He received PhD in Information Sciences (2010) from University of Science and Technology Beijing. Now he is associate professor of computer sciences at Information Engineering Department, Minzu University of China. His current research interests include distributed computing, cloud computing, and network public opinion analysis.

Cheng Wenyi, He Cheng wenyi is currently working toward the Master degree in Software engineering at the School of Minzu University of China. His current research interests are cloud computing and Service composition.

