A SLA-BASED DYNAMICALLY INTEGRATING SERVICES SAAS FRAMEWORK

Chia-Hsien Wen **Hei-Ru Shiau** **Chen-Yen Wang** **Szu-Yen Wang**

Dept. of Computer Science
and Information Management
Providence University, Taiwan
chwen@pu.edu.tw g9871002@pu.edu.tw

Dept. of Computer Science
and Information Management
Providence University, Taiwan
g9771001@pu.edu.tw sywang2@pu.edu.tw

ABSTRACT

SaaS cloud computing is gaining momentum in recent years with more and more successful application adoptions. Processes of these applications often comprise a series of services mapping to users' work flows and a consumer has to designate a particular provider for each service when deploying an application. As long as the provider fails to deliver the service, all application processes which include the service will discontinue their execution even though some other providers may offer the same service. In this paper, we propose a SaaS framework to support dynamic service composition based on QoS criteria of SLA. Each service is brokered and delivered to best fit the criteria specified when it is requested. Applications running on this framework will therefore obtain optimal performance.

Keywords: cloud computing, SaaS, SLA

1. INTRODUCTION

With perceived vision of computing to be the fifth utility for providing public services after electricity, water, gas, and telephony, cloud computing has successfully caught the attention of IT market recently. [1] Cloud computing is defined as "A style of computing in which scalable and elastic IT-enable capabilities are delivered as a service to one or multiple customers using Internet technologies." by Gartner in 2009. [2] Not only can it satisfy users' requests that need computing resources, regardless of hardware, software or applications, but also it enables users to utilize services without knowledge of the supporting infrastructure. Under the concept of "cloud service", computing resources over Internet are transformed as various services. [3] According to marketing characteristics, cloud services can be classified into three service types, namely Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). [4] Paying some affordable price based on the service level agreement (SLA) brokered between providers and consumers, business and users may access computing services from anywhere in the world through Internet on demand instead of maintaining their own computing equipments. [5]
Some SaaS cloud service architectures have been proposed with a payment mechanism. No matter what domain they are applied, their design are focused on creating clouds on which providers may dynamically offer variable level services on demand according to predefined quality of service (QoS) parameters and SLAs. To maintain the equilibrium of service supply and service demand, the service level a user got depends on the price she paid. [6-9]
In a SOA application environment, SaaS cloud services are available on various distributed platforms and are accessed across the Internet. [10] SOA is a flexible loosely-couple, open-standard, and process-centric design principle used during system development and integration. Integrating and reusing existing cross-platform services published by service provider and registered in the SOA broker, a user may pile up an application she needs. SOA based services are generally exposed as web services which follow the W3C standards such as WSDL, SOAP and UDDI. [11] To create, deploy, and orchestrate services in a SOA environment, the Enterprise Service Bus (ESB) is employed as a key component of the SOA infrastructure. The ESB is in charge of passing messages among SOA services and applications on heterogeneous platforms. [12, 13] A dynamic reconfigurable ESB service routing (DRESR) framework was proposed to enable the dynamic routing path construction and message routing. [14]
Business applications are developed typically according to business processes, each of them is composed of a series of business services. Web Services Business Process Execution Language (WS-BPEL), an XML based programming language, came up with a way to describe high level business processes by businesses themselves. [15] Business services in BPEL are themselves exposed as web services available within the SOA. Service compositions, asynchronous interactions among web services participating in a process, are coordinated by the use of the BPEL engine. [16, 17]
A designate application service, requested by a consumer, is usually delivered from a specific provider in a SaaS environment currently. SLAs generally serve as the service quality evaluation criteria between consumers and providers. As more

and more providers may provide equivalent services, users are doomed to making choice from several competing providers statically or dynamically. It is necessary to introduce a service evaluation mechanism in a SaaS platform to find out a preferable service from a massive of candidate services according to quality of service (QoS) parameters such as response time and performance.

Kertesz et al proposed an SLA-based resource virtualization approach for on-demand service provision. They advocated that resources required by each service of a business application may be brokered dynamically by means of agreement negotiation, brokering and service deployment. [18] However, their architecture belongs to a PaaS approach because applications running on their architecture are prepared by users themselves.

In this paper we propose a SaaS framework to support dynamic service composition based on SLA. The service composition is automatically generated by integrating those services which are chosen from candidate services evaluated by QoS criteria of SLA. Employing the proposed framework, a consumer will always get a most suitable service composition and perform an application process smoothly even though some providers are failed to deliver services.

2. MOTIVATING EXAMPLE

A motivating example of clinical visiting process is used to explain how the SaaS framework is designed with SLA -confined service composition. The hypothetico-deductive approach has been a clinical decision making process for a long time. It embeds strategies for data collection and interpretation in an iterative process for physicians to treat their patients. [19] We simplify the approach into a sequential clinical visiting process as shown in Figure 1 for convenience in explanation. This refined process is suitable for both outpatient visits and inpatient visits.

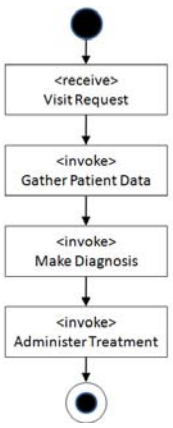
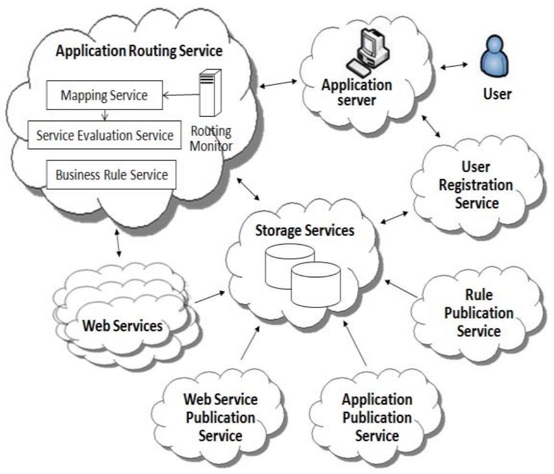


Figure 1: a clinical visiting process

At the beginning of a visit, the physician has to collect data for making diagnosis. After she validates the visiting patient is the right person making appointment, she may review the patient's history of present illness, past history and check results of recent tests, such as laboratory test, radiologic studies, and ECG. According to patient's complains, she may give the patient some physical examination and/or more test orders. Integrating all data collected, she will make diagnosis and will administer treatment for this patient. During the physician collects data and makes diagnosis, she may employ some computerized physician order entry (CPOE) services coupled with medical support services. There are also various databases for evidence based medicine while the physician administers treatment for a patient. In a traditional clinical information system, a physician is unable to use preferable supporting services because such supporting services are typically designed as static functions. If a clinical information system is set up in a SaaS cloud computing environment, more suitable clinical supporting services will be invoked automatically based on predefined QoS criteria of SLA.

3. THE FRAMEWORK

Figure 2 shows the proposed SLA-based dynamically integrating services SaaS framework. Besides the home application server, storage service and web services, the framework consists of four kinds of services, i.e. user registration service, web service publication service, application publication service, and application routing service. At the beginning of an application, a user has to log on to her home application server and then select a function to trigger a series of services.



Application Routing Service
Mapping Service
Service Evaluation Service
Business Rule Service
Routing Monitor
Application server
User
User Registration Service
Storage Services
Web Services
Rule Publication Service
Web Service Publication Service
Application Publication Service

Figure 2: The SLA-based dynamically integrating services SaaS framework

User registration service is the first service to proceed. After completing authentication on the application server, the user's operation request with her personal information will be sent to a LDAP server. The server then returns a response to decide if the user is authorized to perform the requested function by way of directory service. Figure 3 (1.1) shows the flow of user registration service.

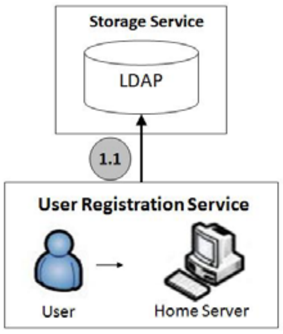


Figure 3: Flow diagram of User Registration Service

Web service publication service comes up with a SOA based mechanism for web service designers publishing their services. Figure 4 depicts the flow of web service publication service. In order to publish a web service, a designer has to wrap service information up in a WSDL file, as shown in figure 4 (2.1), and registers it to a UDDI, the web service registry (2.2). In the meantime, a WSDL Engine will parse attributes , such as service name, parameter name, type, input, output, URL, service price and server location record, of the WSDL file and insert them into the routing path mapping table (RPMT) (2.3).

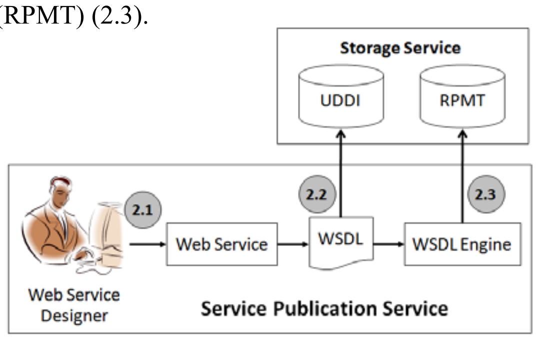


Figure 4: Flow diagram of Web Service Publication Service

Application publication service is used to publish abstract business processes for enterprise applications. An abstract process, generally written in BPEL, serves as a process template which consists of a sequence of web services to model the behavior of a business process. Figure 5 shows the flow of application designers publishing their business processes. Whenever an application designer completes a BPEL file which defines an abstract business process, it will be submitted to the BPEL engine (3.1). BPEL Engine then parses the BPEL file and generates an abstract routing path (ARP) into the abstract routing table (ART) for this business process (3.2). The application designer also has to specify the authorization of each business process and keep such information in the LADP (3.3).

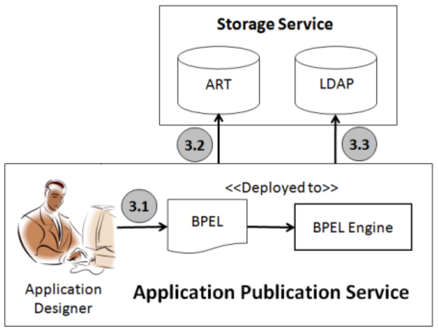


Figure 5: Flow diagram of Application Publication Service

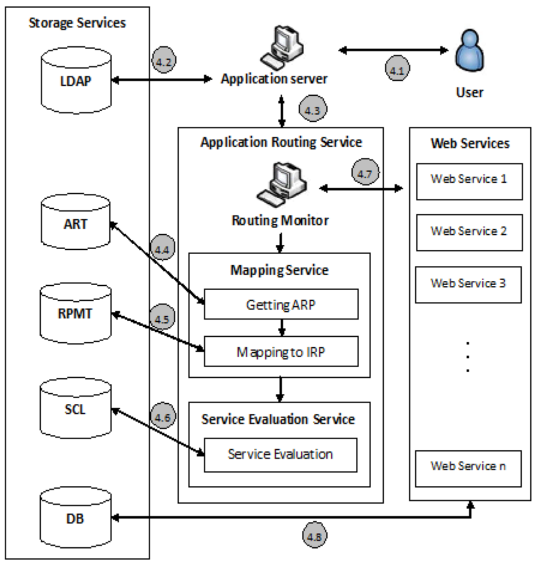
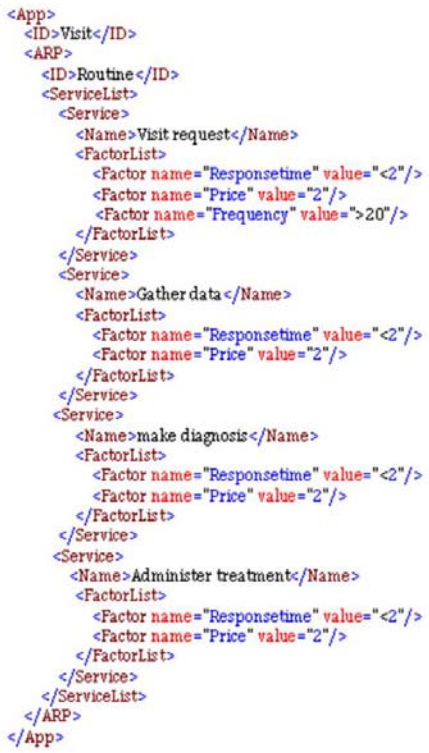


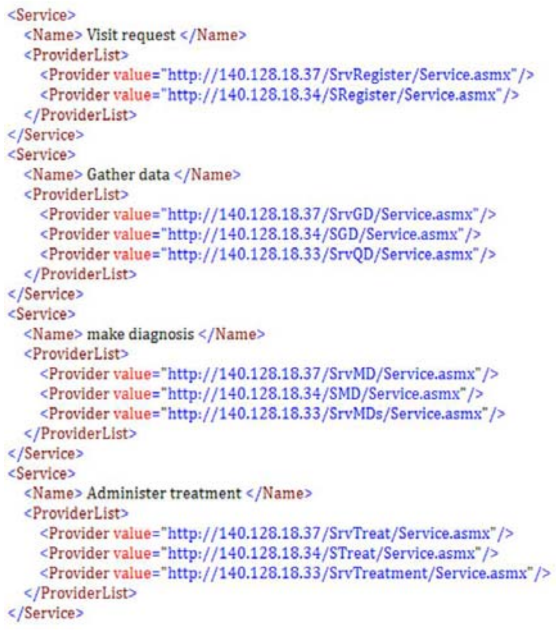
Figure 6: Flow diagram of Application Routing Service

As shown in figure 6, application routing service integrates with web services and storage services to perform an application process. A user connects to her home application server and requests a function to start an application process (4.1). After being authenticated by the LDAP (4.2), the application server sends user's process (function) request to the routing monitor of application routing service (4.3). An abstract routing path (ARP) corresponding to the user's process request will be extracted from the ART (4.4). Since web services of an ARP are represented in service names, they have to be mapped into URLs by way of routing path map table (RPMT) in advance (4.5). Several instance routing paths (IRPs) will be emerged from an ARP for a web service may be offered by several service providers. These IRP candidates are further evaluated based on QoS criteria of SLA, specified in service contention list (SCL), to choose an appropriate IRP for execution (4.6). Routing Monitor routes web services according to the chosen IRP and return result to user (4.7). While a web service is performing, it will access data from application databases (4.8).



```
<App>
  <ID>Visit</ID>
  <ARP>
    <ID>Routine</ID>
    <ServiceList>
      <Service>
        <Name>Visit request</Name>
        <FactorList>
          <Factor name="Responsetime" value="<2"/>
          <Factor name="Price" value="2"/>
          <Factor name="Frequency" value=">20"/>
        </FactorList>
      </Service>
      <Service>
        <Name>Gather data</Name>
        <FactorList>
          <Factor name="Responsetime" value="<2"/>
          <Factor name="Price" value="2"/>
        </FactorList>
      </Service>
     <Service>
        <Name>make diagnosis</Name>
        <FactorList>
          <Factor name="Responsetime" value="<2"/>
          <Factor name="Price" value="2"/>
        </FactorList>
      </Service>
     <Service>
       <Name>Administer treatment</Name>
        <FactorList>
          <Factor name="Responsetime" value="<2"/>
          <Factor name="Price" value="2"/>
        </FactorList>
      </Service>
    </ServiceList>
  </ARP>
</App>
```

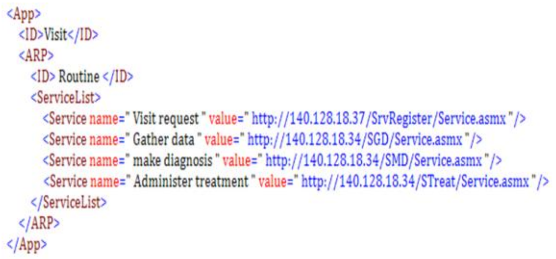
Figure 7: Example ARP of clinical visiting process



```
<Service>
  <Name> Visit request </Name>
  <ProviderList>
    <Provider value="http://140.128.18.37/SrvRegister/Service.asmx"/>
    <Provider value="http://140.128.18.34/SRegister/Service.asmx"/>
  </ProviderList>
</Service>
<Service>
  <Name> Gather data </Name>
  <ProviderList>
    <Provider value="http://140.128.18.37/SrvGD/Service.asmx"/>
    <Provider value="http://140.128.18.34/SGD/Service.asmx"/>
    <Provider value="http://140.128.18.33/SrvQD/Service.asmx"/>
  </ProviderList>
</Service>
<Service>
  <Name> make diagnosis </Name>
  <ProviderList>
    <Provider value="http://140.128.18.37/SrvMD/Service.asmx"/>
    <Provider value="http://140.128.18.34/SMD/Service.asmx"/>
    <Provider value="http://140.128.18.33/SrvMDs/Service.asmx"/>
  </ProviderList>
</Service>
<Service>
  <Name> Administer treatment </Name>
  <ProviderList>
    <Provider value="http://140.128.18.37/SrvTreat/Service.asmx"/>
    <Provider value="http://140.128.18.34/STreat/Service.asmx"/>
    <Provider value="http://140.128.18.33/SrvTreatment/Service.asmx"/>
  </ProviderList>
</Service>
```

Figure 8: Example candidate IRPs of clinical visiting process

Figure 7 shows an example of ARP for clinical visiting process depicted in section 2. This ARP will be mapped into IRPs, as shown in figure 8, based on the RMPT. All IRPs are then evaluated according to the SCL and the most appropriate one is chosen as shown in figure 9.



```
<App>
  <ID>Visit</ID>
  <ARP>
    <ID> Routine </ID>
    <ServiceList>
      <Service name=" Visit request " value=" http://140.128.18.37/SrvRegister/Service.asmx "/>
      <Service name=" Gather data " value=" http://140.128.18.34/SGD/Service.asmx "/>
      <Service name=" make diagnosis " value=" http://140.128.18.34/SMD/Service.asmx "/>
      <Service name=" Administer treatment " value=" http://140.128.18.34/STreat/Service.asmx "/>
    </ServiceList>
  </ARP>
</App>
```

Figure 9: Example final IRP of clinical visiting process

4. DISCUSSIONS

Application routing service stated in previous section maps an application process into a sequence of services by way of mapping service and service evaluation service. Mapping service gets the ARP corresponding to a process and translates it into IRPs by mapping the service name into corresponding URLs. Service evaluation service then evaluates all translated IRPs and chose the most appropriate one according to the SCL. There exist some arguments in electing the right IRP. The first one comes from the necessity of online negotiation and online brokering. The Kertesz's architecture advocated using online negotiation and brokering to accomplish resource virtualization. It only aims at finding a provider to offer a suitable platform to deploy the on demand service software a consumer had prepared. However, application software provision is the target of our consideration and it is difficult to specify software requirement in the SLA. Online negotiation and brokering is therefore excluded from our approach. A consumer has to find out some qualified application service providers to be candidates in advance of application execution.

The second argument is about contents of SCL. Whether an application is robust or not depends on the performance of service providers elected. Contents of SCL are used to evaluate candidate IRPs to choose the most appropriate one. In other words, the SCL has to include some attributes which may decide the service provider to be chosen. These attributes are determined from Qos specified in SLA. Service response time and price are general examples of such attributes. Nevertheless, service level of a provider is not always consistent. A mechanism has to be employed for dynamically updating attributes, such as response time, to reveal current status of each provider.

5. CONCLUSIONS

Integrating concepts of BPEL, SOA, ESB and cloud computing, this paper has proposed a SaaS framework which may dynamically provide on-demand services for enterprise applications. While deploying an application, a consumer is responsible for decomposing each application process into a sequence of services and finding out some suitable providers for each service instead of entrusting all services of an application to a single provider. The most appropriate provider will be brokered to deliver the on demand service if its current service quality best matches the requested QoS specified in SLA. Every application will run smoothly on this proposed platform in spite of any provider fails to offer the requested service normally. To finalize the stated essential architecture of the Saas application platform and to apply it in commercial clouds will be our future work.

6. ACKNOWLEDGMENTS

This paper is based upon work supported by the National Science Council (NSC), Taiwan, under grants NSC98-2410-H-126-010-MY2, NSC97-3114-E-007-001- and NSC98-2218-E-007-005-.

7. REFERENCES

[1] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems, Vol.25, No.6, pp.599–616, 2009.

[2] Gartner, "Experts define cloud computing: Can we get a little definition in our definitions? , " http://www.gartner.com (accessed on: 13 April 2010), 2009.

[3] A. Lenk, M. Klems, J. Nimis, S. Tai, "What's inside the cloud? An architecture map of the cloud landscape," Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing, pp.23-31, 2009.

[4] L.M. Vaquero, L.R. Merino, J. Caceres, M. Lindner, "A break in the clouds: towards a cloud definition," ACM SIGCOMM Computer Communication Review, Vol.39, No.1, pp.50-55 ,2009.

[5] P. Patel, A. Ranabahu, A. Sheth, " Service Level Agreement in Cloud Computing," Workshops at Object-Oriented Programming, Systems, Languages & Applications, 2009.

[6] M.A. Vouk, "Cloud computing issues, research and implementations," In 30th International Conference on Information Technology Interfaces (ITI 2008), Cavtat/Dubrovnik, Croatia, pp.31-40, 2008

[7] C. Vecchiol, S. Pandey, R. Buyya, "High-performance cloud computing: A view of scientific applications," Proceedings of the 10th International Symposium on Pervasive Systems Algorithms and Networks, 2009.

[8] A. Rosenthal, P. Mork, M.H. Li, J. Stanford, D. Koester, P. Reynolds, "Cloud computing: A new business paradigm for biomedical information sharing," Journal of Biomedical Informatics, Vol.43, No.2, pp.342-353, 2010.

[9] R. Buyya, R. Ranjan, R.N. Calheiros, "Modeling and simulation of scalable cloud computing environments and the CloudSim Toolkit: Challenges and opportunities," Proceedings of the 7th High Performance Computing and Simulation (HPCS 2009) Conference, 2009.

[10] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, et al., "Above the clouds: A berkeley view of cloud computing," Technical report, EECS Department, University of California, Berkeley, 2009.

[11] K. Kawamoto, D.F. Lobach, "Proposal for fulfilling strategic objectives of the U.S. roadmap for national action on decision support through a service-oriented architecture leveraging HL7 services," Journal of the American Medical Informatics Association, Vol.14, No.2, pp.146-155, 2007.

[12] R. Wullianallur, K. Someswar, "Interoperable electronic health records design: Towards a service-oriented architecture," e-Service Journal, Vol.5, No.3, pp.39-57, 2007.

[13] J. Hu, F.E. Luo, J. Li, X. Tong, G. Liao, "SOA-based enterprise service bus," 2008 International Symposium on Electronic Commerce and Security, pp.536-539, 2008.

[14] X. Bai, J. Xie, B. Chen, S. Xiao, "DRESR: Dynamic routing in enterprise service bus," In ICEBE '07: Proceedings of the IEEE International Conference on e-Business Engineering, pp.528–531, 2007.

[15] B. Eidson, J. Maron, G. Pavlik, R. Raheja, "SOA and the future of application development," Proceeding of the First International Workshop on Design of Service-Oriented Application (WDSOA), pp.1-8, 2005

[16] C. Peltz, "Web services orchestration and choreography," IEEE Computer, Vol.36, No.8, pp.46-52, 2003.

[17] F. Curbera, Y. Goland, J. Klein, F. Leymann, D. Roller, S. Thatte, et al., "Business process execution language for web services, Version 1.0," Standards proposal by BEA Systems, International Business Machines Corporation, and Microsoft Corporation, 2002.

[18] A. Kertesz, G. Kecskemeti, I. Brandic, "An SLA-based Resource Virtualization Approach for On-demand Service Provision," Virtualization Technology in Distributed Computing, Barcelona, Spain, June 2009, pp.27-34.

[19] E.H.Shortliffe, L.E. Perreault, G. Wiederhold, L.M. Fagan, "Medical Informatics Computer Applications in Health Care and Biomedicine (2nd ed.)," Springer-Verlag New York, Inc., 2001.