Business Logic Structure of SaaS-based E-commerce System

Kai-feng LI, Hao-yu WEN, Zhao-jun YANG
Department of Economics and Management, Xidian University, Xi'an 710071, China
(wenhaoyu@gmail.com)

Abstract - **The functional requirements of multiple tenants are supported by a single instance in a SaaS System, but the differences between tenants make it difficult to unify in same business logic. Based on the construction of SOA and using BPEL to deploy different processes on application servers, a business process customization model for the application layer is proposed and the specific solution of process customization and implementation are discussed with the combination of Web Service.**

Keywords - **multi-tenant system, business logic, system extension, SaaS, SOA**

I. INTRODUCTION

In traditional information systems, customization or secondary development is often required during the implementation so that system functions can adapt to customer demands.

As a multi-tenant software operation model, Software as a Service (SaaS) [1] has features of simple, easy maintenance, affordable and so on. Due to the sharing of the same conventional structure of application layer by multi-tenant, SaaS systems usually only meet the common needs of most users. But from the perspective of the market, information system users often have characteristics consistent with its own business logic which requires SaaS systems to provide a certain degree of customization according to customer demands [2]. It can be said that satisfying individual business needs with users' configuration together with secondary development is one of the core elements to promote the wider use of SaaS model [3].

The customization of the application layer of SaaS systems is more complex than that of traditional information systems, especially in its specific technical implementation to balance the customization of business logic and the sharing among systems.

Code modification and application re-deployment are usually necessary in the customization of traditional information systems. However, this customized model is unrealistic in SaaS Systems because of the sharing of the same application instance by all tenants. With a tenant customizing the application, all tenants will be affected, and the services may be interrupted during the system updating. With the increasing number of tenants, the interruption will occur more and more frequently, which could result in very serious service quality issues [4]. Thus, the configurability of the business process at runtime is one of the critical needs of SaaS applications. In this paper, how to realize the expansibility of the application layer based on Service-Oriented Architecture (SOA) is discussed, according to the characteristics of SaaS systems.

II. FEASIBILITY ANALYSIS ON SOA-BASED SAAS APPLICATION LAYER CONSTRUCTION

As an IT architectural style, SOA supports converting business to a group of mutually linked services or repeatable business tasks which can be accessed over the network when needed. Besides, with the combination of services, specific business processes can be achieved, so that the business logic can quickly adapt to the changing objective conditions and needs [5]. This feature is exactly what the SaaS application layer needed to achieve the customizability. Therefore, SOA architecture can be used as the infrastructure of SaaS applications [6].

Generally, Web Services Definition Language (WSDL) is used in SOA to describe service interfaces to make them dynamic. In addition, when building a SOA-based SaaS application layer model, a business process which consists of multiple services is required to be defined, so that the business process can be associated with software processes.

Business Process Execution Language (BPEL) [7] is one of the key technologies in SOA to implement the definition and automation of business processes. Internally, it calls services line with Web Service specifications and describes the interaction and collaboration between them; externally, it provides voluntary process services and maps relations between business processes and software processes [8].

The process, defined by BPEL, is essentially the arrangement of a series of individual non-state services to make them execute in accordance with established business rules. Usually, the interaction with external services is involved in the process with its interactive interfaces described by WSDL. In this way, the seamless compatibility between BPEL and Web Services is achieved; meanwhile the solution for the definition of sub processes or nested processes is also provided [9]. Considering simplicity and reusability, the business logic of the SaaS application layer should be provided in the Web Service mode and the definition of service interfaces should be abstract without relation to service binding, service quality definition and so on, because these definitions associated with process implementation can be

Supported by Fundamental Research Funds for the Central Universities(NO.72104994)

978-1-4244-6484-5/10/$26.00 ©2010 IEEE

determined during the deployment of BPEL processes. Therefore, the features of BPEL provide the technical basis for the scalability and configurability of business processes in SaaS applications.

In a multi-tenant SaaS system, separated process instances are created by different tenants, so that it is necessary to keep the instances isolated to ensure the system security. In the execution environment of BPEL processes, multiple BPEL process instances can be generated by a same process template, while the instance information will not be preserved by services called by BPEL [10]. When the BPEL process interacts with a service asynchronously, how to transmit information belonged to the same interaction to the correct process instance is a problem which must be resolved to build the SOA-based application layer. Meanwhile, the association mechanism between messages and process instances is required to be established to make the process instances depend on business data carried by messages. Tenants or system extension developers can define a set of service attributes as correlation set wherein a BPEL process instance must be uniquely identified, so that the BPEL runtime environment could route messages to the process instance that matches the correlation set. Different correlation sets can be defined for different interfaces of the process to flexibly adapt to various SaaS business needs.

III. BUSINESS PROCESS CONFIGURATION OF SAAS APPLICATION LAYER

A. *Customization Model of SaaS Application Layer*

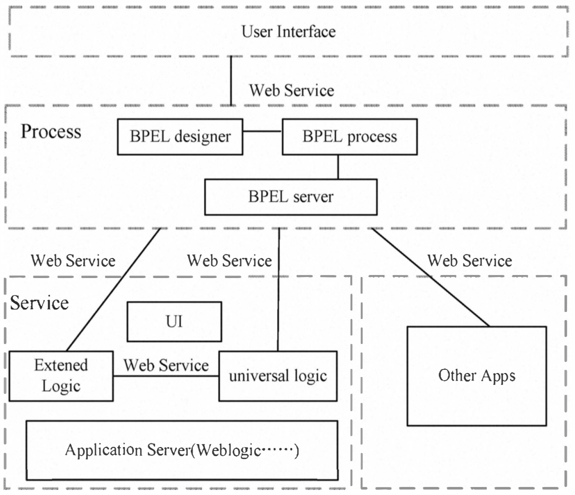

Fig. 1. Customization model of SaaS application layer

In construction of SaaS applications, some common logics within the business processes can be extracted to models for reusing in process modeling. With structural active components defined by BPEL, the basic process model can be described. On this basis, a complex process model can be easily described by the combination of multiple basic process models. The customization model of the SaaS application layer is established combined with its requirements and characteristics, as shown in Fig. 1.

As shown in Fig. 1, this model is logically divided into three parts: service, process and UI.

1) *Service module*: Running the specific business logic which is accomplished by the appropriate Web Service development technology. During the development, mature design patterns can be used to construct relevant service programs and customized demands can be satisfied by using interface classes. In this part, the universal logic module is responsible for the implementation of the default business logic for all tenants, while the extended logic module is in charge of the implementation of the customized business logic.

2) *Process module*: Using technology associated with BPEL to accomplish the arrangement and customization of processes. The BPEL server is responsible for running process instances and calling the corresponding services.

3) *UI module*: In this part, a comparatively simple process definition interface is provided to transform the business description form users into standard BPEL process description.

With the customization model of the SaaS application layer, user-defined and universal business processes can be transformed into executable BPEL processes, and specific functions can be realized by running BPEL instances that deployed on the application server to call the relevant Web Services.

B. *BPEL Process Design and Implementation*

Generally, service descriptions, message types, service connection types and other relevant information should be created before the process modeling. Based on this, workflow templates are created by process modeling tools. Process logics include the following two types of operations.

1) *Original process and data operation.* It is used for calling and receiving calls from Web Services, including methods such as calling other Web services by <invoke>, using <receive> to receive requests and wait for the messages from the client to call the BPEL process, generating responses of synchronous operations by <reply>, manipulating data variables by <assign>, indicating the fault and exceptions by <throw>, waiting for a period time by <wait> and terminating the whole BPEL process by <exit>.

2) *Structured operation.* It is used to control which operation step should be performed in a BPEL process according to schedule. It involves using <sequence> to define a set of activities called in sequence, <flow> to define a set of activities invoked in parallel, <switch> to accomplish branches, <while> to define cycles and <scope> to define the nesting.

In the following, the feasibility of BPEL process design and implementation is illustrated by an instance of B2C online shopping system.

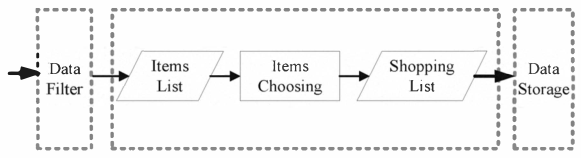

Fig. 2. Shopping process A

Merchant A sells products on a first come first served basis. Customers cannot order the product which is sold out. The process is shown in Fig. 2.

In order to attract customers, merchant B carries out a promotion that customer with spending more than N (such as $100) can obtain a random selected present. The process is shown in Fig. 3.

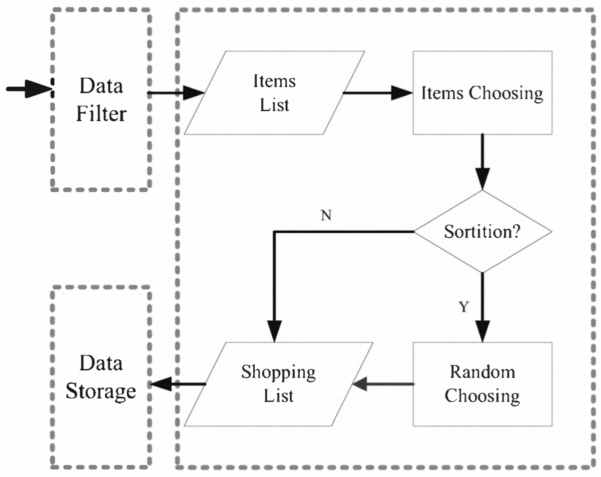

Fig. 3. Shopping process B

It can be seen that the shopping processes of A and B are the same except the sortition process of merchant B, such process differences are widely existed in SaaS applications.

Shopping process model, shown in Fig. 4, is constructed to satisfy the need of the process customization. The process definition of the application is described by XML and the process assignment is received from the service description file by the BPEL engine which can also start the process by calling corresponding services. In this example, the application receives input variables from the corresponding schema, completes the business logic verification through the implementation of business rules and returns output variables when the operations are completed.

In this shopping process described by a WSDL file, the process model receives tasks firstly, and then gets input parameters. After that it calls the Validation Service according to the defined business rules to decide whether the trade satisfies the conditions or not and finally returns result after the verification. The business rules above can be various for different situations, including which service should be called in order to accomplish business functions. Meanwhile, program codes such as Java can also be embedded in the BPEL process to complete partial business logics.

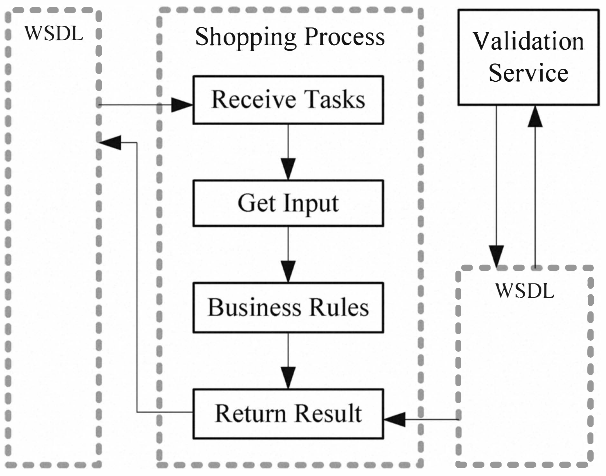

Fig. 4. Shopping process model based on BPEL

There are three basic activities in the sequence of the shopping process A: receiving product lists, calling the shopping Web Service to obtain the treated shopping lists and returning the generated shopping list to customers.

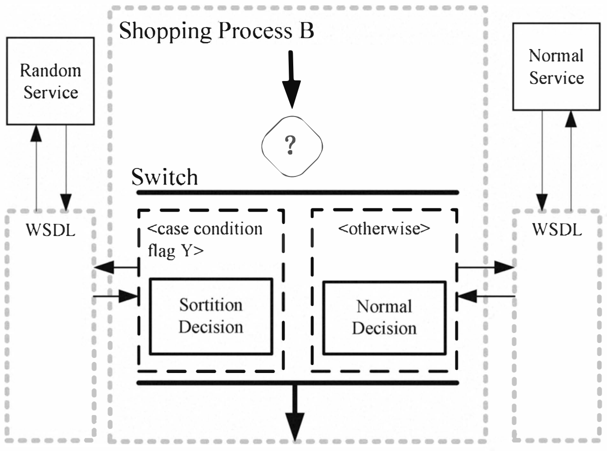

Fig. 5. Branch model in shopping process B

In normal circumstances, the shopping process of merchant B is the same as merchant A, and consequently, the shopping Web Service can still be invoked during the process. But because of the existence of the sortition activity, a branch process controlled by a switch is necessary. For customers who achieve the goal, their shopping lists will be handled by a random selector and the result will be returned to the shopping lists after the sortition is done. For the differences in shopping processes between A and B, a BPEL model can be constructed as shown in Fig. 5.

During the application deployment, each tenant can customize business processes according to its needs by the association between the tenant's configuration files and the specific BPEL process instance. With case analysis, it shows that it is feasible to accomplish customization of the SaaS application layer in this way.

IV. CONCLUSION

The SaaS application layer is analyzed to encapsulate the system in multiple fine-grained services to meet the operation requirements for the application layer of SaaS information system flexibly. BPEL is used to complete the design and deployment of processes. The highly adaptable, extensible and customizable SaaS application layer constructing model can be established by the dispatch of Web Service via BPEL with its feasibility verified by the instance of e-commerce system.

REFERENCES

[1] Ahmed Elfatatry, Paul Layzell. Software As A Service: A Negotiation Perspective. Proceedings of the 26th Annual International Computer Software and Applications Conference 2002 IEEE: 501-506.

[2] M. Turner, D. Budgen, and P.Brereton. Turning software into a service. Computer, pp. 38–44, 2003.

[3] Dan Ma. The Business Model of "Software-As-A-Service". 2007 IEEE International Conference on Services Computing, SCC 2007: 701-702.

[4] F.Chong and G.Carraro. Architecture strategies for catching the long tail. MSDN Library, Microsoft Corporation, 2006.

[5] Eric Newcomer, Greg Lomow. Understanding SOA with web services. Addison Wesley Professional, 2004.

[6] Yuan Zhi-jun, Xia Hong-xia. Research of online software system development solution based on SaaS. Computer engineering and design, 2009, 30(11):2717-2714.

[7] T. Andrews, Curbera. Business process execution language for web services, version 1.1. Standards proposal by BEA Systems, International Business Machines Corporation, and Microsoft Corporation, 2003.

[8] Mark Endrei, Jenny Ang, Ali Arsanjani et al. Patterns: Service-Oriented Architecture and Web Services. IBM Corporation Redbook 2004:26.

[9] Tobias Anstett, Frank Leymann, Ralph Mietzner, Steve Strauch. Towards BPEL in the Cloud: Exploiting Different Delivery Models for the Execution of Business Processes. pp.670-677, 2009 IEEE Congress on Services - I, 2009

[10] Pasley, James. How BPEL and SOA are changing web services development. IEEE Internet Computing, Volume 9, Issue 3, May/June 2005:60-67.