A Multi-tenant Web Application Framework for SaaS

Wonjae Lee
Broadcasting & Telecommunications Convergence Research Lab
Electronics and Telecommunications Research Institute
Daejeon, South Korea
russell@etri.re.kr, leewonjae.kr@gmail.com

Min Choi
School of Information and Communication Engineering
Chungbuk National University
Cheongju, South Korea
mchoi@cbnu.ac.kr

***Abstract*—Software as a Service (SaaS) is a software delivery model in which software resources are accessed remotely by users. Enterprises find SaaS attractive because of its low cost. SaaS requires sharing of application servers among multiple tenants for low operational costs. Besides the sharing of application servers, customizations are needed to meet requirements of each tenant. Supporting various levels of configuration and customization is desirable for SaaS frameworks. This paper describes a multi-tenant web application framework for SaaS. The proposed framework supports runtime customizations of user interfaces and business logics by use of file-level namespaces, inheritance, and polymorphism. It supports various client-side web application technologies.**

Keywords-Software as a Service (SaaS); Platform as a Service (PaaS); Multi-tenancy; Cloud Computing.

I. INTRODUCTION

Software as a Service (SaaS) is a software delivery model in which software resources are accessed remotely by clients. The subscription business model of SaaS is attractive to small and medium enterprises.

SaaS providers can exploit benefits of the economies of scale when computing infrastructures are shared among multiple tenants [1] [3]. In addition to shared computing resource, tenant-specific customizations are required for many application domains [1]. User interfaces, business logics, data models, and other elements of an application need to be customized to meet requirements of each tenant.

Industry-specific customizations and department-specific customizations are also desirable. The companies in a certain industries may share common customization requirements for a specific application.

II. RELATED WORK

Paper [1] has provided a configuration and customization competency model and a methodology framework for SaaS. The authors of [2] presented approaches and principles to support multi-tenancy. Force.com has a metadata-driven software architecture to provide multi-tenancy [4]. Application codes and customization codes are treated as metadata in the Force.com platform. Those codes are dynamically executed to provide tenant-specific customizations.

While previous works support multi-tenancy in various ways, they tend to be too complex and to lack generality. In configuration-oriented SaaS frameworks [1] [2], only a limited number of client-side technologies can be used and the expressive power is limited by configuration boundaries. Complex configuration tools are required for such frameworks.

III. A MULTI-TENANT WEB APPLICATION FRAMEWORK

We have designed and implemented a multi-tenant web application framework for SaaS. The framework supports customization, scalability, and sharing of computing resources.

A. Architecture

The component & connector view [4] of the multi-tenant framework is illustrated in Fig. 1.

The SaaS application server serves requests from clients. When a client sends a request to the SaaS application server, the SaaS application server sends software data requests to the software data server. When the software data server sends required data to the SaaS application server, the SaaS application server dynamically executes the received user interface codes and business logic codes. Then the SaaS application server sends a response message to the client.

When the SaaS application server executes user interface codes and business logic codes, it may access database that contains user data.

The software data includes user interface codes and

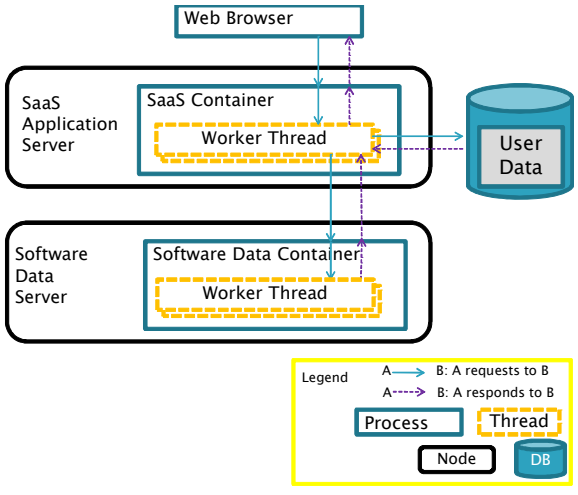

Figure 1. Architecture of the multi-tenant web application framework.

978-0-7695-4755-8/12 $26.00 © 2012 IEEE
DOI 10.1109/CLOUD.2012.27

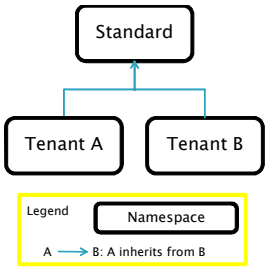

Figure 2. Namespaces for customization.

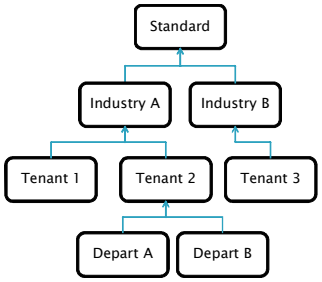

Figure 3. Multilevel inheritance for customization.

business logic codes. To be more specific, Hypertext Markup Language (HTML) files, image files, Cascading Style Sheets (CSS) files, JavaScript files, Chameleon template files [5], and Python program files [6] are treated as software data. The Chameleon template engine is used to provide server-side dynamic pages. The Python program files are used to implement business logics.

This architecture promotes scalability and availability of the system. A SaaS provider can run multiple SaaS application servers to serve many clients.

The dynamic execution of user interface codes and business logic codes enables runtime customizations without restarting of application servers.

The Pyramid web framework [7] is used to implement both the SaaS application server and the software data server.

B. Namespaces and Customization

In the proposed multi-tenant framework, namespaces, inheritance, and polymorphism are used to implement tenant-specific customizations. In this framework, files in namespaces are inherited and overridden.

When a software provider develops a web application, the developed web application files reside in the standard namespace of the application. When tenant A and B subscribe to the application, namespaces for the tenants are created as in Fig. 2. The namespace of each tenant inherits files from the standard namespace. When there is no file in the tenant A's namespace, the tenant A will use the standard application as is without any customization. When customization is required, the tenant A can override files of the standard namespace by uploading files into the namespace of the tenant A.

Namespaces can be used to support various levels of the configuration and customization competency model [1]. Simple user interface customizations can be easily done. A tenant can replace images by overriding image files. When a tenant wants to change the look and formatting of a HTML document, overriding Cascading Style Sheets files will do the job. Neither HTML files nor Chameleon template files need to be modified because URLs (Uniform Resource Locator) will continue to work even when CSS files or images files are overridden.

The proposed multi-tenant framework also supports multilevel inheritance as in Fig. 3. In the proposed framework, industry-specific customizations and department-specific customizations are possible. Multilevel inheritance will promote code reuse when industry-specific or department-specific customizations are required.

IV. CONCLUSIONS

We have proposed the multi-tenant web application framework for SaaS. The framework provides an effective customization method for user interfaces and business logics. The customization method supports various levels of customization through the use of file-level namespaces, inheritance, and polymorphism. Industry-specific, tenant-specific, and department-specific customizations are supported by multilevel inheritance. It promotes reuse of user interface and business logic components. The customizations can occur in runtime without restarting application servers. In the framework, multiple tenants share a single application server, and this lowers operational costs. The system can scale by adding SaaS application servers. This also improves the availability of the system. The framework is not tied to any specific client-side technologies. Its expressive power is not limited by any predefined boundaries. A tenant can use various JavaScript libraries by uploading libraries into the tenant's namespace. The performance evaluation has showed that the performance impact due to supporting multi-tenancy can be minimized when software data are cached in the SaaS application server.

REFERENCES

[1] Wei Sun, Xin Zhang, Chang Jie Guo, Pei Sun, Hui Su, "Software as a Service: configuration and customization perspectives," Congress on Services Part II, 2008. SERVICES-2. IEEE, pp.18-25, 23-26 Sept. 2008, doi: 10.1109/SERVICES-2.2008.29

[2] Chang Jie Guo, Wei Sun, Ying Huang, Zhi Hu Wang, Bo Gao, "A framework for native multi-tenancy application development and management," E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, 2007. CEC/EEE 2007, pp.551-558, 23-26 July 2007, doi: 10.1109/CEC-EEE.2007.4

[3] Craig D. Weissman, Steve Bobrowski, "The design of the force.com multitenant internet application development platform," Proceedings of the 35th SIGMOD international conference on Management of data, pp.889-896, June 29-July 02, 2009, Providence, Rhode Island, USA doi:10.1145/1559845.1559942

[4] P. Clements, F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, R. Nord, and J. Stafford. Documenting Software Architectures, Views and Beyond. Addison Wesley, 2002, pp. 103-123.

[5] Chameleon templates. http://chameleon.repoze.org/

[6] Python Programming Language. http://python.org/

[7] Pyramid web framework. http://www.pylonsproject.org/