软件即服务应用框架中配置的设计与实现

张 雷 扈 飞 (北京邮电大学 智能通信软件与多媒体北京重点实验室 北京 100876)

Configuration Design and Realization in SaaS Application Framework

Lei Zhang, Fei Hu (Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing 100876)

Abstract: The challenges for Software as a Service (SaaS) application are how to provide a set of methods to enable tenants to configure the application and how to ensure that after their configuration, the applications they subscribed will run accurately. In this study, we focus on application configuration which is one of the most significant features of SaaS and discuss the differences between configuration and customization. Furthermore, we bring up a more configurable Framework based on workflow management and rule management. In order to prove that the framework is able to provide more flexible configuration capabilities for tenants, we implement a conference management system (confOnline) based on this framework.

Key words: SaaS; workflow configuration

1 Introduction

1.1 Background introduction

SaaS is a new delivery model for software. In SaaS model, customers can subscribe or unsubscribe from the software provider as they want and pay, only for the actual use of the software. Providers offer the same application to different customers. Each of the tenants can use the application in a way as if it would only serve this particular one. Currently, SaaS model is a preferable solution for most small and medium companies (SMC) who need Office Automation (OA) support but can't afford the high development and maintenance cost.

Offering an application which can't be configured severely limits the amount of potential customers. Since the requests of the tenants are different, the application must be customizable and configurable to meet the needs of each particular customer.

The configuration of SaaS application can be described by Fine-grained level and Coarse-grained level. Fine-grained level configuration enables tenants to configure the parameters of the application based on pre-define functionality modules. However, Coarse-grained level configuration enables tenants to configure the execution order of the functionality modules.

1.2 Related works

Currently, SaaS providers have not yet established the best practices about supporting SaaS development style, nor are there application standards. However, there are some companies providing advanced and practical SaaS applications and many papers discussing the constitution of SaaS application framework and how to put it into practice.

Salesforce[1] is the most successful SaaS application so far which implements customer relationship management(CRM)based on SaaS application framework. This successful application brings out a flexible framework to meet tenants' special needs and provides a set of interfaces to enable tenants to connect with their existed system. There are also some successful SaaS providers such as Refs.[2,3].They are both conference management sys- tems based on SaaS application framework and have many tenants using their application.

Although all of these providers having provided all-round services to satisfy their tenants, there are also some deficiencies waiting to be improved. The configuretion abilities of these applications are still staying on the Fine-grained level. If tenants need to frequently change the business workflows to meet their temporary requests, SaaS providers can't roll out all of the workflows for thousands of tenants in a short period of time. In addition, most of the tenants can't develop expansions independently because of lacking of computer technology. To deal with this problem, we need to design a new framework which supports Coarse-grained level configuration to provide better configuration capabilities to meet tenants' needs.

Ref.[4] presents an approach using enterprise application Integration (EAI) pattern to execute an integration solution. The executable EAI patterns can be seen as reusable components and serve different tenants. Ref.[5] shows a configurable SaaS application using service composite architecture (SCA). It provides an approach to add a new service component into the existed application easily by using SCA and service data object (SDO). The most significant view of Ref.[5] is using SOA to realize the business logic parts of an application and delivering it based on SaaS framework.

Refs.[6,7] discuss how to implement a new application and deploy it rapidly by using SaaS application platform.

Ref.[8] presents a SaaS application based on mobile and ubiquitous computing environment. It describes an organization structure of SaaS architecture and the responsibilities of client side and server side respectively. As Ref.[8] emphasizes, the temporary states should be saved at client side and keep server side stateless as far as possible.

1.3 The organization of article

This paper is divided into several parts. In Section 1, we introduce some background information, related work, and the organization of this paper. In Section 2, we discuss the characteristics of SaaS and the SaaS framework. In Section 3, we bring up a set of methods to realize the configuration of SaaS framework. At last, we give a conclusion and introduce the future work.

2 The SaaS application framework

2.1 Key characteristics of SaaS

A well-designed SaaS application should be scalable, multi-tenancy and configurable. Scalability means maximizing concurrency, and using application resources more efficiently. Multi-tenancy is the very significant characteristic. The application will accommodate tenants from dozens to thousands. This characteristic requires SaaS application framework to maximize the sharing of resources among tenants, but is still able to differentiate data which belonging to different customers. Configuration means that SaaS application framework should provide adequate approaches to enable tenants to change the system for adapting to their special needs.

2.2 SaaS maturity model

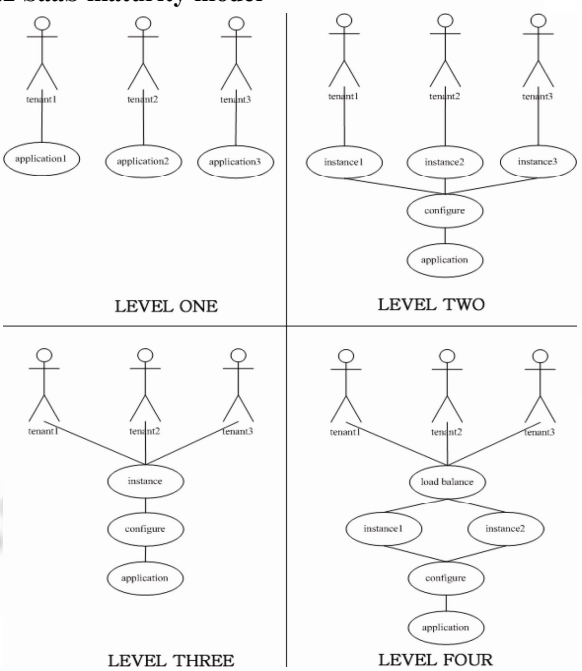

Fig. 1　The SaaS maturity model

There are four levels of SaaS maturity model as showed in Fig.1. At the first level, each tenant has its own customized version of an application. At the second level, the SaaS provider hosts a separate instance for each tenant and allows them to change the application behaviors by providing detailed configuration options. All instances in level two use the same code implementation. Level three shows that the SaaS provider runs a single instance to serve all tenants with configur-

able metadata to provide the unique user experience for each one. At the final level, SaaS provider hosts multiple tenants on a load-balanced farm of identical instances. This level describes a SaaS framework which is scalable to an arbitrarily number of tenants. confOnline is based on the final level.

2.3 The framework of confOnline

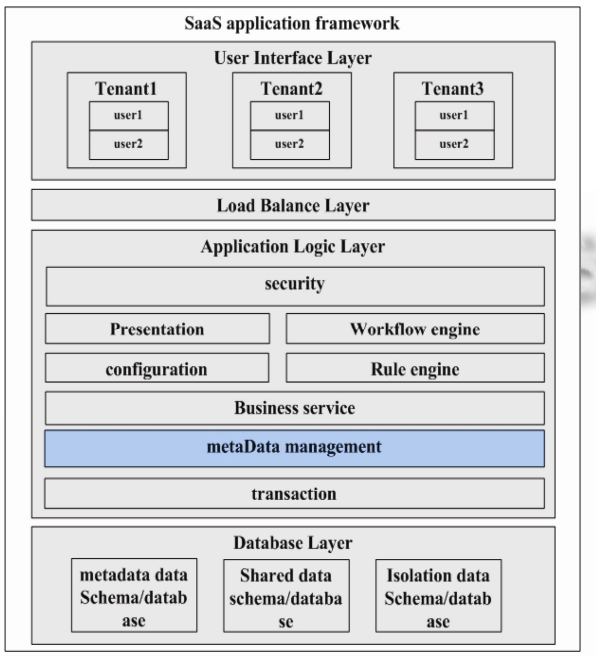

Fig.2 SaaS application framework

This application framework has four layers: User Interface layer, Load Balance layer, Application Logic layer and Database layer. Tenants can log in the system using browser through the User Interface layer. At the second layer, load-balance layer gives SaaS framework much more scalable capability to ensure tenants enjoying high-quality services. As shown in Fig.2, the applica- tion logic layer is constituted by a number of services such as security, presentation, and configuration. At the database layer, we combine "the shared database and separate schema approach"[9] with "the shared database and shared schema approach"[9] to design the database.

3 The Configuration of confOnline

3.1 configuration and customization

SaaS applications are usually developed with highly standardized software functionalities to serve as many clients as possible. But most of enterprise software applications need to be tailored more or less so as to effectively serve a specific client.

The widely used approaches of tailoring software are configuration and customization. Currently configuration and customization plays more and more important role in SaaS application. However they are very similar in the conceptual point of view. From software perspective, there are many obvious differences between them.

Configuration usually supports to adapt for variances through setting pre-defined parameters. Tenants are able to use tools to change parameters or application functions within pre-defined scope without changing the code. At present, most of SaaS applications focus on providing parameter configuration not functionality configuration.

Customization involves SaaS application source code changes to create a new functionality module that is beyond the configuration limit.

Comparing with configuration, customization is a much more costly approach. For example, reallocating resources and infrastructure to match new software code version; managing much longer lifecycle brought by code development, debugging, testing and deployment. Because of these, we should avoid customization by using configuration to meet tenants' needs and enlarge the configuration scope as far as possible to provide better configuration capacities to tenant.

3.2 Scheduling mechanism

To work out a much more configurable framework, we bring up a scheduling mechanism to realize this target.

We divide the requests from tenants into three types:

• Presentation requests: this type of requests asks SaaS application to show information which is needed without changing the state of business service. Presentation request is supported by presentation module.

• Configuration request: this type of requests asks to change the configuration of SaaS application. Because of isolating configuration module from business module, configuration module is able to handle requests without changing the state of business service also.

• Business request: this type of requests changes the states of business services. For example, an author submits a paper. This action will activate a lifecycle to manage this paper's information. In this process, the application will distribute resources to persistent the information of lifecycle. Through analysis, tenants' new demands are reflected in the requirements for the new inputs, since the application is regarded as a black box to them. By using workflow engine and rule engine, SaaS application can easily handle such new requests and schedule the application services to support them.

As we have introduced, each type of requests has its own order to deal with. Here we will focus on the sequence for handling business request in this framework.

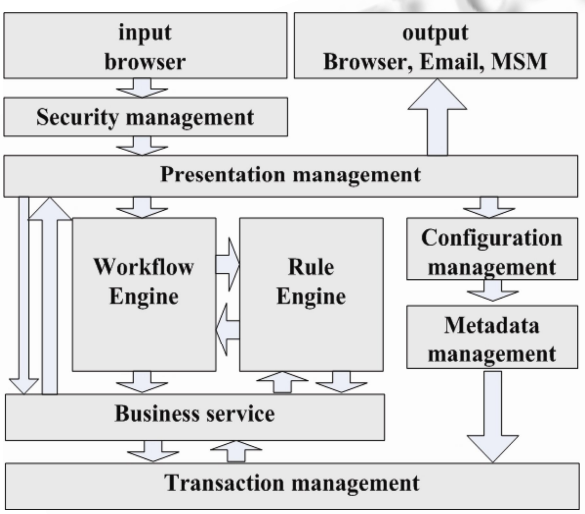

Fig. 3 Scheduling mechanism in confOnline

Every business process has its workflow description. Using modular design method, we can analyze the business processes and divide them into several functionality modules. After such a deal, most of the modules are stateless and reusable. By scheduling these business logic modules, workflow engine is able to assemble a new process easily and execute it rightly. Rule engine which manages the business rules acts as the decision-making node of workflow engine. When business requests from tenants are received by the application, workflow engine will ask rule engine to check the state of related process. Through analyzing the related rules, rule engine will determine how to proceed with the next step and send the response to workflow engine. After that, workflow engine will activate the related business logic module with the guidance of rule engine.

This mechanism adds a scheduling layer between requests from tenants and the concrete application logic. This layer is similar to the orchestration layer in SOA which manages web services. Using such scheduling layer, both SaaS providers and tenants are able to configure the business processes in Coarse-grained level.

3.3 The scope of configuration

In this application framework, tenants can make configuration in four broad areas:

User Interface: SaaS application enables tenants to change the presentation style such as CSS. In addition, they also can regulate the order of functionality widgets to meet their needs.

Workflow and Rule: Tenants can create, delete and update the process definition or the rule definition at runtime.

Extension to data model: Each tenant has its unique data model. This SaaS application provides some common data models to meet tenants' basic needs and also allows the tenants to create some new data model to fit to their special needs.

Security Control: It allows tenants to manage the organization structure of the SaaS application. By using authorization, tenants can create a new role and grant it to the target user. Also, they can change the functionality modules access limitation to realize Security Mechanism by using authentication.

From configuration implement perspective, only workflow configuration belongs to Coarse-grained level configuration.

3.4 Configuration realization

In this framework, we combine Coarse-grained configuration and Fine-grained configuration to form a more flexible and mature configuration strategy.

To realize Coarse-grained configuration, we bring up a design form to restrict the business application logic. As we have introduced, each business process is composed with several functionality modules. To introduce the form clearly, we use formalization method to describe workflow model and the relationships between nodes:

P = {N, E, T, C};

P:Process definition N:Node

E:Entity T:Transition
C:Context

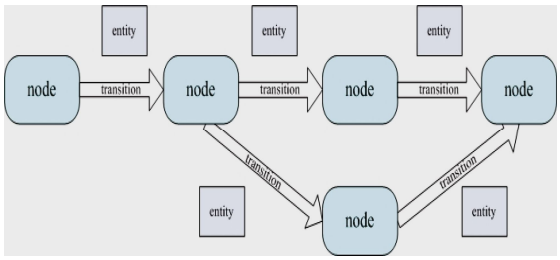

Fig. 4 Workflow model descriptions

Each process contains several nodes. One node represents one business functionality module. The context which acts as restrictions to process definitions describes the relationships between nodes. As shown in Fig.4, each transition carries one entity which includes enough information to insure that the next node can finish its job based on it. Using such workflow design form, we can separate the variable points from business functionality modules and save them in entities. Based on this operation, new workflow definition can be easily constructed by such stateless nodes.

Each node of a process must be supported by the implements of presentation logic, application logic and database logic[10]. To deal with this limitation, we should analyze the business domain and provide a common functionality set to meet the basic needs of tenants. By using the common interfaces which according to the design form, we can extend the set without changing the existed functionality modules.

We use jbpm which is an open source project to realize the workflow engine in this application. Refer to jbpm designer plug-in for eclipse, we design and implement web workflow designer for tenants using Flex. By using the web workflow designer, tenants who must comply with the workflow context restriction can update a process definition or create a new one.

For instance, one tenant wishes to change the default conference lifecycle process. He wants to delete the node from the process definition which configures the author's data model. The operations are very easy to this tenant to achieve his goal. Firstly, he should choose the node and delete it from the process. Secondly, create a new transition to connect the two nodes neighbored with the deleted one. Thirdly, this tenant configures the transition. This step will be supervised by the workflow context restriction. Fourthly, submit this new process definition to server. If this new definition is according to the context restriction, it will be saved by workflow engine.

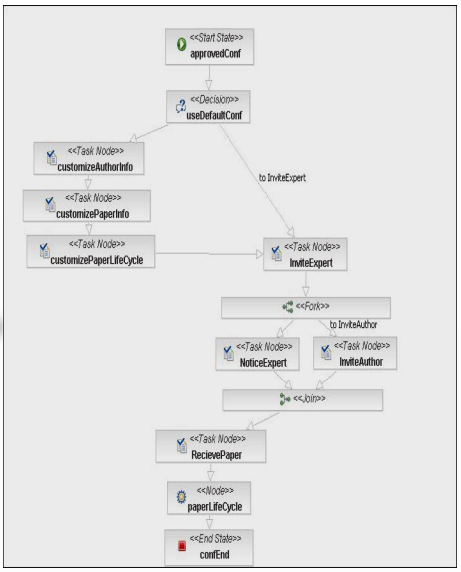

Fig.5 Conference lifecycle configurations

We realize the Fine-grained configuration from two aspects. They are rule configuration and data model configuration.

Rule engine which manages the business rules of the application plays a very important role in this framework. To practice rule configuration, we use template pattern to design the rule templates. Each tenant has a rule file which constructed by the rule templates. In this application, we use drools which is an open source project to realize the rule engine. Rule engine also provides a set of program interfaces for workflow engine and business functionality modules to check related rules.

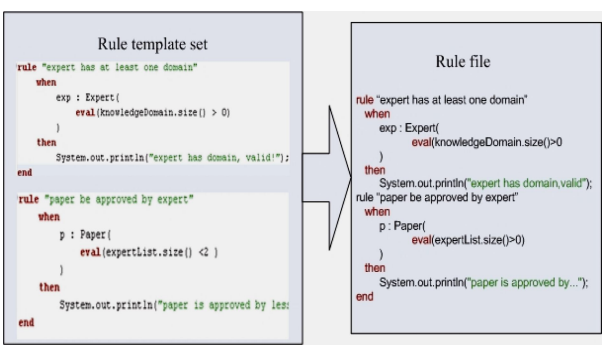

Fig.6 Rule templates and rule file

Tenant can easily check the rules and update them through browser.

A standard default database is created for a new tenant as part of the provisioning process. Tenants can modify the tables to meet their special needs based on database logic. To realize data model configuration, we use metadata to describe the fields of a table and use Id-value pairs to record the value of a field. Tenants also are able to use data model designer to add a new field into related table and define it through browser.

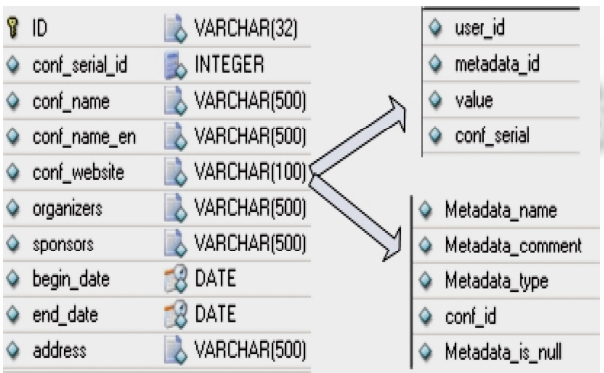

Fig.7　Data model design

4　Conclusion

In this paper, we introduce the framework which enables tenants to configure the application at Coarse-grained level and Fine-grained level. In our framework, we use workflow engine to direct the functionality modules and use rule engine to manage the business rules. This mechanism increases the flexibility of SaaS application.

The most important point of the paper is how to enable tenants to configure the application better. In the future, we will focus on the restriction formalization for workflow definitions.

References

1 http://www.salesforce.com.

2 http://myreview.lri.fr/.

3 http://cmt.research.microsoft.com/cmt/.

4 Scheibler, Thorsten,Mietzner, Ralph, Leymann, Frank. EAI as a Service-Combining the Power of Executable EAI Patterns and SAAS Enterprise Distributed Object Computing Conference, 2008. EDOC'08.12th International IEEE, 2008(9):107－116.

5 Mietzner R, Leymann F, Papazoglou, M.P. Defining Composite Configurable SaaS Application Packages Using SCA, Variability Descriptors and Multi-tenancy Patterns Internet and Web Applications and Services, 2008.ICIW'08. Third International Conference, 2008, (6):156－161.

6 Zhang Q, Liu SJ, Meng XX. The research and implementation of turning conference management system into a service. Asia-Pacific Service Computing Conference,The 2nd IEEE, 2007(s):521－526.

7 Espadas, Javier, Concha, David, Molina, Arturo. Application development over software-as-a-service Platforn. Software Engineering Advances, 2008. ICSE A '08. The Third International Conference, 2008,(10):97－104.

8 Anerousis N, Mohindra A. The Software- as-a- Service Model for Mobile and Ubiquitous Computing Environments. Mobile and Ubiquitous Systems: Networking & Services, 2006 Third Annual International Conference, 2006(7):1－6.

9 Multi-Tenant Data Architecture ttp://msdn.icrosoft.com/en-us/library/aa479086.aspx.

10 Sun W, Zhang X, Guo CJ, Sun P. Su H. Software as a Service Configuration and Customization Perspectives. Congress on Services Part II, 2008. SERVICES-2. IEEE, 2008(s):18－25.