

Level 2 SaaS Platform and Platform Management Framework

Jihyun Lee, Sung Jin Hur

Cloud Computing Research Department, ETRI, South Korea

jihyun@etri.re.kr, sjheo@etri.re.kr

Abstract— Monitoring helps to understand resource utilization in cloud computing and detect usage pattern of diverse service users. As SaaS platform allows tenants to customize software services for their requirements, the services become more complex. Due to the complexity of customized services, monitoring for runtime information encounters new challenges and the need for monitoring for multi-tenants is increasing. In this paper, we propose architecture for monitoring run-time execution and managing SaaS, which represents SaaS hosting foundation in addition to common monitoring interest. Currently, we are implementing the monitoring framework on ETRI SaaS platform, SaaSapia and describe the feasibility study about extensions to be implemented, which are required for monitoring the services being shared by multi-tenants and which are planned to be implemented in 2011.

Keywords— SaaS platform, management framework, monitoring, platform architecture, multi-tenant

I. INTRODUCTION

The platforms in the form of an infrastructure have been existed, which provided authentication, billing, integrated operations, and management. However, the platforms were not the implementation of cloud computing paradigm. In the perspective of reflecting diverse requirements from the traditional platforms, individual enterprise was provided with an individually customized service for its customization requirements. However, individual customization is inefficient in terms of cost. Therefore, Software as a Service platform is expected to compete with Platform as a Service. As if to aim at these differences, Microsoft has developed Microsoft CRM Dynamics as a SaaS (Software as a Service) platform and another Pass Platform named Azure.

Currently, ETRI is in the progress to research and develop Level 2 SaaS platform, which is equipped with the facilities such as core framework, customization, and platform management tool.

From 2009, ETRI SaaS platform, SaaSapia has been scheduled to open the service for origination management in

This work was supported by the IT R&D program of MIS/IITA. [2009-S-033-01, Development of SaaS Platform for Software Service of Small and Medium Enterprises]

October in 2010. This paper described a set of features in the SaaSapia and platform management subsystem, which provides platform-core framework and enables the management for services, systems, tenants, and users. This paper consists of the following sections: Chapter 2 describes cloud computing architecture. Chapter 3 proposes the SaaS hosting foundation concerned with monitoring real-time executions and managing SaaS. In Chapter 4, related works are described and we conclude in Chapter 5.

II. CLOUD COMPUTING ARCHITECTURE

Cloud-based business application, i.e. SaaS is served based on one of the cloud computing architectures. The cloud computing architecture can be separated into 3 types shown in Figure 1.

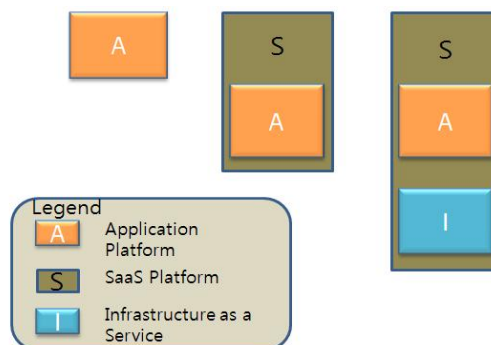


Figure 1. 3 Types of Cloud Computing Architecture

The 3 types of the cloud computing architecture are an application-led platform, a SaaS platform containing customization framework, and a SaaS platform including an application platform and an infrastructure such as storages, physical data centers, and etc. for delivering and scaling on-demand services [1][2].

Among of these, we select the second type of a SaaS platform based on an application platform and name this type as the level 2 SaaS platform.

A. Level 2 SaaS Platform

SaaS platform is a set of technologies and services used to develop, deploy, integrate, and deliver SaaS applications [3]. The architecture of level 2 SaaS platform is composed of

applications, application services, and an application platform, which is shown in Figure 2.

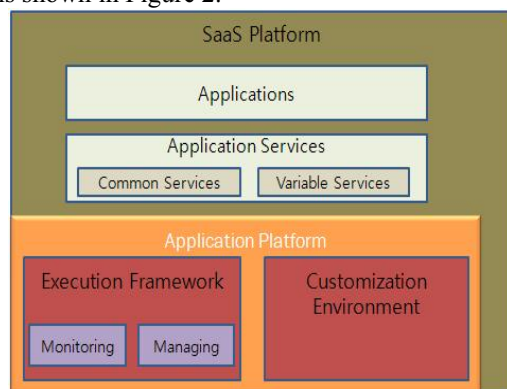


Figure 2. Architecture of Level 2 SaaS Platform

The method to enable extensibility in the level 2 SaaS platform, new applications can be added without any changes of platform code. Because application services are designed to be associated with customized services, tenant administrators can make changes of the generic applications by describing all the functional variations of the application services.

The customization environment in the Figure 2 provides the facilities required to construct custom variants. To address the challenges in multi-tenant based monitoring and management, we propose the monitoring and management subsystem of execution framework and provide a monitoring model for obtaining real-time statistics. In addition, we are considering the extension of the execution framework in order to provide the differentiated services following the service contracts of multi-tenants.

B. Needs of Platform Management Focusing on Efficient Multi-Tenancy

In SaaS, the core concept is multi-tenancy, which has a meaning that the services are provided to multiple organizations, where the end users of individual organization can access and use their enterprise services after an original SaaS application service is customized. Since the SaaS platform has multiple tenants together with another multiple end users per tenant, the important platform feature is in the monitoring and managing the complex service stakeholders. For the monitoring and management, the SaaS platform should recognize the following situations, for example, who are using SaaS application services, what kinds of services and resources need to be supplemented, and what type of errors are occurring in the customized services.

In order to deal with management issues in multi-tenant environment, the SaaS provides management facilities to administrate tenants, tenant users, services, and systems, which are corresponding to the functions such as customizing services, managing error codes, logging exceptions and error messages, and managing tenant data repository.

Moreover, the management architecture of the SaaS platform is required to focus on efficient multi-tenancy to provide the services for both administrative users and end users in each tenant.

C. Multi-Tenant Based Platform Monitoring

Our level 2 SaaS platform is established with program code, application services, customization framework, and execution framework. Among them, the execution framework plays two roles such as monitoring and managing. The part of monitoring is focusing on acquiring real-time statistics about a system, tenants, users, and services. Because the monitored data are generated when multi-tenants is about to use and in use of services, each target of monitoring is concerned with the activities from multi-tenants.

Therefore, the existing common monitoring for traditional software and web services is different from the monitoring SaaS platform in this paper and need to be extended.

III. PLATFORM MONITORING AND MANAGEMENT FRAMEWORK

In this chapter, we describe the platform management tool and the execution framework including monitoring and managing facilities, which is important one system of the SaaS being developing in ETRI.

A. 4 Category Monitoring based on Multi-tenancy

The modules of the monitoring real-time execution inside of the execution framework are corresponding to monitoring framework; and the monitoring is concerned with 4 categories of functions in the platform management tool.

The structures of the execution framework and the platform management tool are shown in Figure 3, some of the tool is under development in this year and others are going to be implemented and extended until 2011.

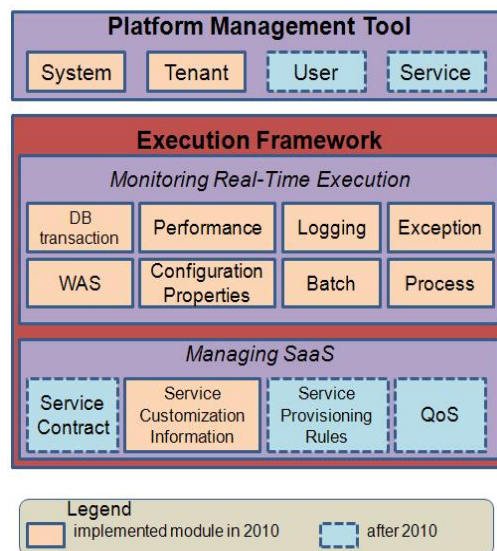


Figure 3. Framework for Managing Platform

The modules in “in scope” represent those of the already implemented in this year from the monitoring system and the modules in the managing SaaS are scheduled for implementation. We mainly deal with the monitoring real-time execution and the functionality of that will be described below.

1) **Monitoring a System:** The system resources such as CPU, memory, session, and threads are monitored, which are shown in graphical charts and utilized to identify the degree of utilization of service execution related system resources and decide whether the status of a running system is enough to provide other new tenants with services. And the monitored data can be also shown for a specific web application server, corresponding to filtering conditions such as keywords and periods.

2) **Monitoring Tenants:** In the SaaS platform, multi-tenants are sharing system resources and common parts of application services in one instance, and receive services based on different conditions of their contracts to services; therefore, it is necessary to monitor how many tenants are provided with the customized services and to manage which tenants are approved for the license or rejected.

3) **Monitoring Users:** Since each tenant has more than one service users, the end-users should be registered according to the information of user groups and role hierarchy in the tenant. Therefore, the function of monitoring users includes registering end users of a tenant, providing privileges, and tracing usage patterns of when users use services.

4) **Monitoring Services:** When each user use services, the function of monitoring services collects usages metered by use and offer the monitored data to service developers for upgrading and modifying services to have more enhanced quality.

B. Run-time Monitoring Enabler

The execution framework of the SaaSPlia involves run-time monitoring enablers such as a resource monitoring library, a contract checker, a log analyser, a process filter, and a service probe [4]. The relationships between the execution framework and the platform management tool in Figure 4 of the next page.

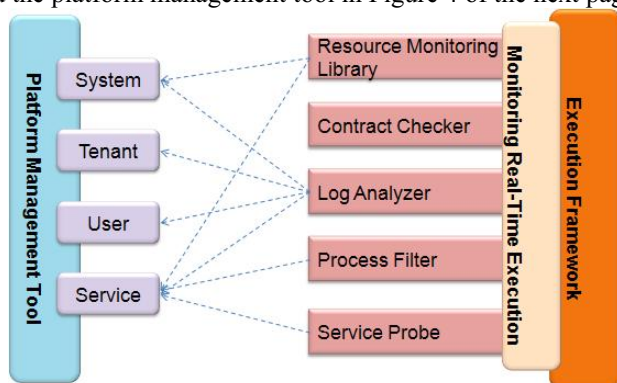


Figure 4. Relationship between Run-time Monitoring Enablers and the SaaSPlia Execution Framework

C. Tenant and System Management

We designed the sub-system for managing tenants to cover the approval or refusal of the requests for registering tenants from general organizations and the requests for providing the formally service agreed companies with the other services. Table 1 shows the implemented functions of the SaaSPlia platform management tool.

TABLE 1. PLATFORM MANAGEMENT MENUS

Category	Menu Name	2 nd Level Menu Name
Tenant Management	Registration approval	
	Tenant profiling	
	Service approval	
System Management	Performance monitoring	Overall
		WEB/DB
		Statistics
	Log level management	
	Error history management	
	Error Code management	
	WAS management	
	Web properties management	
	Batch history management	
	Process management	

The system management sub-system include the functions for monitoring system performance and managing log level, history for errors and batch jobs, WAS, properties for configuring web, and running processes.

The design and implementation is based on model-view-controller architecture of Spring Framework. We chose Eclipse, Tomcat, Oracle, Java, and ExtJS respectively for a development tool, a web server, a database system, an implementation language, and a JavaScript GUI library.

IV. IMPLEMENTATION

We use the implementation languages such as Java JDK 6, Ext JS 3.3, and Spring Framework 3.0 and all configurations of implementation artifacts including source codes are managed with Subversion. Oracle is used for DMBS server. And testing activities are conducted with JUnit and Mantis bug tracker.

For example, the screen shots in Figure 5 and 6 represent system monitoring from the platform management tool. Figure 5 shows some of detailed information which are monitored for DB queries through iBatis from the application server instance, SW11.

index (ex.)	total execution number	total execution time	max /min /average execution time
infrastructure.was.wasList	2	0.0170	0.0140 0.0030 0.0095
infrastructure.was.delete-was	1	0.0030	0.0030 0.0030 0.0030
uimetaDatabase.selectAttrList	19	20.4870	1.7930 0.0060 1.0782
infrastructure.domain.selectDomainCc	5	0.0170	0.0040 0.0030 0.0034
errorCode.COUNT_SQL	7	0.0240	0.0040 0.0030 0.0034
errorHis.selectPagingList	6	0.0590	0.0110 0.0050 0.0073
infrastructure.was.search-was-pag	20	0.0700	0.0050 0.0020 0.0035
monitorStat.FIND_BY_DAILY_STAT_S	11	0.0590	0.0140 0.0030 0.0053
infrastructure.was.insert-was	1	0.0240	0.0240 0.0240 0.0240
infrastructure.code.updateCodeValue	2	0.0080	0.0040 0.0020 0.0030
infrastructure.codeType.updateCode1	5	0.0160	0.0080 0.0020 0.0032
batchHis.FIND_BY_PAGE_SQL	10	0.0510	0.0130 0.0030 0.0060
framework.errorHis.insert	81	0.6400	0.2510 0.0020 0.0079

Figure 5. Monitoring DB Transactions

Figure 6 shows the information of the registered tenants and the rejected applicants for service licenses.

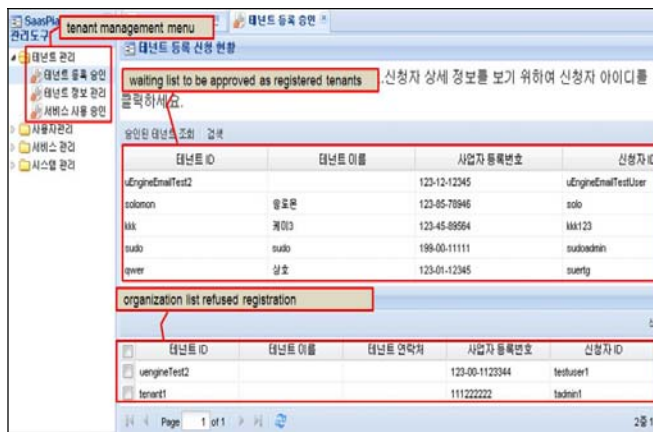


Figure 6. Managing Registration of Tenants

V. RELATED WORKS

A. Monitoring Web Services

When people monitor web services, they mainly focus on monitoring performance. And the monitoring target is the event stream obtained from the BPEL engine instead of services itself [5]. Therefore, web service monitoring framework is aiming at monitoring of service interaction protocols [6].

B. Monitoring Software as a Service Platform

On the other hand, the monitoring SaaS is focusing on how the application services are working in the SaaS platform. However, research works regarding to monitoring in SaaS is not common. We consider that research works concerned with monitoring SaaS platform has not been open from the company such as Salesforce.com, Google, and etc. We can say that the research field for monitoring SaaS is unattended and new because it was not begun extensively in SaaS.

The existing tool kits such as Jennifer [7] and InfraRed [8] in open source can be candidate for monitoring extensive performance of SaaS because they are considerably good at performance monitoring.

VI. CONCLUSIONS

We suggest architecture for monitoring run-time execution and managing SaaS, which represents SaaS hosting foundation in addition to common monitoring concerns. The monitoring framework on the SaaS platform is being implemented and the feasibility managing SaaS platform was described.

Future works will be done by performing following activities. For example, the modules included in the "Managing SaaS" of the "SaaS Hosting Foundation" are out of scope of our project in this year. The facilities such as metering, billing, and managing SLA should be implemented in the SaaS platform as the modules in the managing SaaS system and integrated with the monitoring system. It is important to add the facility regarding to whether tenants' customization requirements are satisfied in the running services. In addition, it should be necessarily monitored whether the predefined rules and constraints are followed for multi-tenant efficient service provision and management.

REFERENCES

- [1] Yefim V. Natis, "Reference Architecture for Multitenancy: Enterprise Computing "in the Cloud," Gartner, 2008.
- [2] Abhijit Dubey, Junaid Mohiuddin, Aadarsh Baijal, "Emerging Platform Wars in Enterprise Software," McKinsey&Company, 2007.
- [3] Guoling Liu, "Research on independent SaaS platform," ICIME 2010, pp. 110-113, 2010.
- [4] J. Shao, H. Wei, Q. Wang, H. Mei, "A Runtime Model Based Monitoring Approach for Cloud," Cloud Computing, pp. 313-320, 2010.
- [5] K. Mahbub, and G. Spanoudakis, "Run-time Monitoring of Requirements for Systems Composed of Web-Services: Initial Implementation and Evaluation Experience," Web Services, vol. 1, pp. 257-265, July 2005.
- [6] Z. Li, Y. Jin, and J. Han, "A Runtime Monitoring and Validation Framework for Web Service Interactions," Software Engineering, pp. 10 -79, April 2006.
- [7] Jennifer, <http://www.jennifersoft.com/docs/apm-jennifer-installation-file-download.html>
- [8] InfraRed, infrared.sourceforge.net