Journal of the Korea Academia-Industrial cooperation Society
Vol. 14, No. 2 pp. 868-877, 2013

http://dx.doi.org/10.5762/KAIS.2013.14.2.868

소프트웨어 재사용성 향상을 위한 PaaS 기반 클라우드 컴포넌트 통합 연구

김철진[1*]
[1]인하공업전문대학 컴퓨터시스템과

A Integration Research of Cloud Component based on PaaS for Enhancing Software Reusability

Chul-Jin Kim[1*]
[1]**Dept. of Computer Systems and Engineering, Inha Technical College**

요 약 본 논문은 클라우드 컴퓨팅 환경에서 개발 재사용성을 향상시켜줄 수 있는 PaaS 기반의 클라우드 서비스를 제공하는 것이다. PaaS 기반의 클라우드 서비스는 개발 측면에서 제공될 수 있는 플랫폼 클라우드 서비스로서 기존의 개발 도구나 관리 도구 서비스의 수준을 넘어서 재사용 프레임워크 서비스를 제공한다. 이러한 재사용 프레임워크 서비스는 다양하게 분산되어 있는 서비스를 활용하여 개발의 재사용성을 향상시켜 줄 수 있을 것이다.

Abstract This paper will provide the cloud service based on PaaS that can enhance reusability of development in the cloud computing environment. The cloud service based on PaaS is the cloud service of platform in the side of development, which provide the reusable framework service that is beyond the existing development tool or management tool service. This reusable framework service will be enhanced reusability using a variety of distributed services.

Key Words : Cloud Computing, PaaS(Platform as a Service), Reusability, Reusable Framework

1. 서론

현재 클라우드 컴퓨팅 서비스의 주요현안은 스토리지 기반의 데이터 공유 서비스로 국한된다는 것과 개인정보 보호 및 데이터 안정성에 대한 이슈일 것이다.

구글, 애플, 마이크로소프트, 아마존, AT&T, 등의 기업들이 제공하는 클라우드 컴퓨팅 서비스는 대부분 데이터를 공유하기 서비스가 주류를 이룬다. 물론 인프라 서비스를 제공하지만, 자체 서비스로 활용하는 경우가 대부분이다. 이와 같이 현재 클라우드 컴퓨팅 서비스 모델 측면에서 SaaS(Software as a Service) 나 IaaS(Infrastructure as a Service) 모델 형태의 서비스을 제공하고 있으나, PaaS(Platform as a Service) 모델 형태의 서비스 제공은 미약한 상황이다[1,2,3]. 개발자에게 제공되는 PaaS 형태의 클라우드 서비스는 현재 소프트웨어 개발 도구, 어플리케이션 배치, 실행, 관리할 수 있는 서비스를 제공한다. 개발 측면에서의 기본적인 서비스를 제공할 뿐, 생산성 측면에서 개발 효율을 향상시킬 수 있는 PaaS기반 클라우드 서비스 아키텍쳐 연구가 요구되고 있다.

기업의 소프트웨어 개발은 촉박한 개발 일정으로 인해 업무 요구사항에 집중하며, 향후 재사용성 및 유지보수에 대한 고려를 하지 못하는 실정이다. 이에 따라 [4]에서와 같이 소프트웨어 유지보수로 인하 비용이 기하급수적으로 증가하고 있다. 본 논문에서는 업무 요구사항 외에 품질 요구사항을 고려하여 개발할 수 있는 개발 플랫폼 서비스를 제공하므로 소프트웨어의 생산성을 향상시켜 줄

이 논문은 2012학년도 인하공업전문대학 교내연구비지원에 의하여 연구되었음.
*Corresponding Author : Chul-Jin Kim (Inha Technical College)
Tel: +82-10-6398-7471 email: cjkim@inhatc.ac.kr
Received January 21, 2013 Revised February 5, 2013 Accepted February 6, 2013

것이다.

본 논문의 2장에서는 관련연구로 클라우드 컴퓨팅 기술과 기존 클라우드 컴퓨팅 아키텍쳐에 대해 살펴보고, 3장에서는 재사용 기반 PaaS 클라우드 컴포넌트 구조 및 개발 프로세스를 제시한다. 4장에서는 제시한 PaaS 기반 클라우드 개발 플랫폼 컴포넌트에 대한 사례연구를 통해 적합성을 검증한다.

2. 관련 연구

2.1 클라우드 컴퓨팅 기술

클라우드 컴퓨팅은 컴퓨팅 자원의 공유, 다양한 서비스의 연동, 등을 통해 사용자에게 서비스를 효율적으로 제공하기 위한 컴퓨팅 체계라고 할 수 있다[5]. 가상 공간에서 모든 정보들이 공유되어 소프트웨어, 플랫폼, 인프라 환경 등을 사용자의 요구에 따라 활용하게 함으로써 정보의 통제와 관리, 검색, 조정이 혁신적인 수준으로 발전할 수 있게 한다.

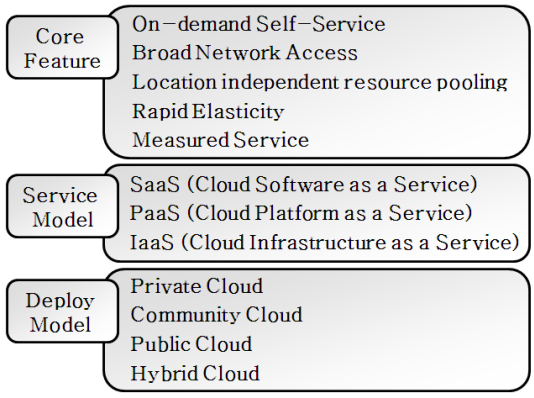
클라우드 컴퓨팅의 주요기술은 Table 1과 같이 대규모 정보 인프라의 필요성과 다양한 서비스 연결, 그리고 사용자 편의를 위한 제공하기 위한 기술들로 구성된다.

[Table 1] Core Technology of Cloud Service

Technology	Core Technology
Grid Computing	Distribution Technology
Utility Computing	Usage Measurement, Billing, User Account
Virtualization Technology	Resource Pool, Hypervisor, Virtual I/O, Partition Mobility
SOA	OpenAPI, WebService
SLA	Service Level Mgt.
Security and Privacy	Firewall, Intrusion Prevention, Access Control

그리드 컴퓨팅(Grid Computing)은 분산 환경에서 공유를 통한 컴퓨팅 유휴자원의 상호 활용을 위한 기술이며, 유틸리티 컴퓨팅(Utility Computing)은 사용자가 컴퓨터를 직접 구매하지 않고 사용한 만큼의 비용을 지불하기 위한 기술이다. 가상화 기술은 서버의 수를 빠른 시간에 확대 배치할 수 있게 할 수 있으므로 컴퓨팅 자원의 구축과 활용 방식이 보다 새로운 체계로 변형 시킬 수 있다. 사용자 접근성이 강조된 웹 서비스 기술과 서비스 연계의 용이성을 위한 SOA(Service Oriented Architecture), 공급자와 사용자 간의 질적 보장을 위한 SLA(Service Level Agreement) 등이 체계를 이루어 클라우드 서비스를 제공할 수 있다.

미국 국립기술표준원 NIST(National Institute of Standard Technology)에서는 클라우드 컴퓨팅의 특징에 대해 Fig. 1에서와 같이 5가지의 핵심적인 특성(Essential Characteristics)과 4가지 배치 모델(Deployment Model), 그리고 3가지 서비스 모델(Service Model)로 정의하고 있다[6].



[Fig. 1] Features of Cloud Service(NIST)

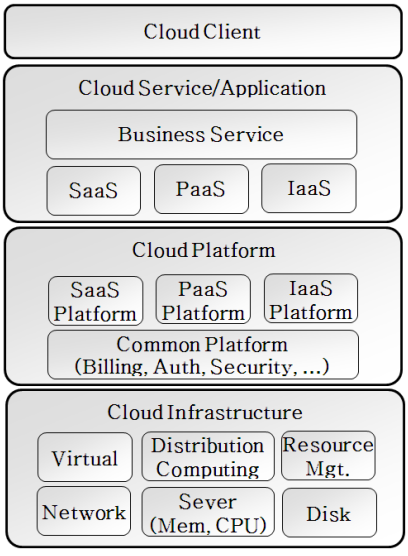
서비스 모델에서 SaaS는 기업이나 다수의 개인 사용자에게 공통으로 필요한 소프트웨어 서비스를 제공하는 형태이다. 제공되는 서비스의 종류는 온라인 검색 서비스, 지도 서비스, 블로그, 위키, 웹 메일 등이 있다. PaaS는 개발자에게 어플리케이션이나 어플리케이션 서비스를 편리하게 개발할 수 있도록 소프트웨어 개발 도구, 어플리케이션 배치, 실행, 관리할 수 있는 서비스를 제공한다. 본 연구에서는 서비스 모델 중에 PaaS를 기반으로 하고 있다. IaaS는 컴퓨팅 자원을 구축하지 않고 필요한 컴퓨팅 자원의 일부 또는 전부를 서비스 받을 수 있도록 제공하는 모델이다. 일반적으로 가상화 기술을 바탕으로 사용한 자원을 자동 계측하여 과금하는 유틸리티 컴퓨팅 방식에 따라 서비스를 제공한다.

배치 모델에서 Private Cloud는 기관이나 기업 내부의 제한된 사용자만을 위하여 구축되는 클라우드 서비스를 말하며, Public Cloud는 모든 사용자가 이용할 수 있도록 공개하는 서비스를 의미한다. Community Cloud는 특정 사용자만의 접근을 통해 서비스를 제공하는 형태이며, Hybrid Cloud는 2가지 이상의 배치모델을 조합하여 서비스를 제공하는 형태를 말한다.

2.2 클라우드 컴퓨팅 아키텍쳐

클라우드 컴퓨팅의 핵심기술을 기반으로 계층적 아키텍쳐는 Fig. 2와 같다.

클라우드 인프라 계층에서 서버, 스토리지, 네트워크가 가상화 기술의 대상이된다. 분산 컴퓨팅에서는 독립적인 파일 시스템 및 데이터베이스를 단일 시스템으로 인지하고 접근할 수 있도록 하며 대용량 데이터들에 대한 빠른 처리 속도를 제공해 줄 수 있다. 구글이 제공하는 Hadoop[7]가 대표적인 기술이다. 클라우드 컴퓨팅에서 자원 관리는 이용자들에게 SLA 기반의 사용자 가상 컴퓨팅 환경을 제공하고, 제공된 가상 시스템을 모니터링하며, 사용자 서비스별 자원 활용도에 따라 동적 자원 할당 및 동적 스케쥴링을 제공해 준다.



[Fig. 2] Cloud Computing Architecture

클라우드 플랫폼 계층에서는 사용자들이 클라우드 컴퓨팅에 사용자 고유의 응용 또는 인터넷 서비스를 구축하기 위한 인터페이스를 제공한다. 프로그래밍 언어의 인터프리터 환경 등과 같은 소프트웨어 개발 환경들과 서비스들의 API를 제공한다. 또한 보안 및 프라이버시, 자원 유틸리티(과금), 사용자 인증 등의 기능을 제공한다.

클라우드 컴퓨팅에서 제공하는 주요기능은 백업기능, 동기화 기능, 자원 관리 기능, 서비스 주문 기능, 인증 기능을 제공한다. 백업 기능은 사용자의 문서, 멀티미디어 자료 등의 모든 정보 자원들을 저장할 수 있으며, 사용자 디바이스에 복원해야 할 때, 사용자의 과거 정보 자원을 복원할 수 있다. 동기화 기능은 사용자의 데이터에 대해 언제 어디서나 다양한 디바이스에서 해당 데이터를 접근할 수 있도록 허용한다. 자원 관리 기능은 개발 사용자의 가상 컴퓨팅 자원을 모니터링하며, 고가용성을 보장해 준다. 서비스 주문 기능은 클라이언트에 클라우드 서비스를 제공하기 위해 용이한 사용자 인터페이스를 제공한다. 사용자 인증 기능은 사용자 인증 인터페이스를 제공하여 사용자의 데이터 접근에 대한 신뢰성을 제공할 수 있도록 한다.

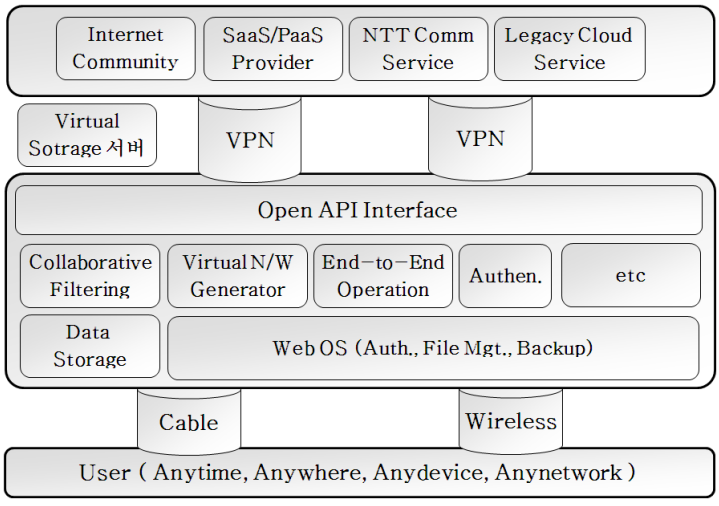
2.3 PaaS 기반 클라우드 서비스

Google App Engine[8]은 PaaS 기반 클라우드 서비스로서 2008년부터 Google 인프라 상에서 사용자가 웹 서비스를 개발할 수 있도록 다양한 API를 서비스하고 있다.

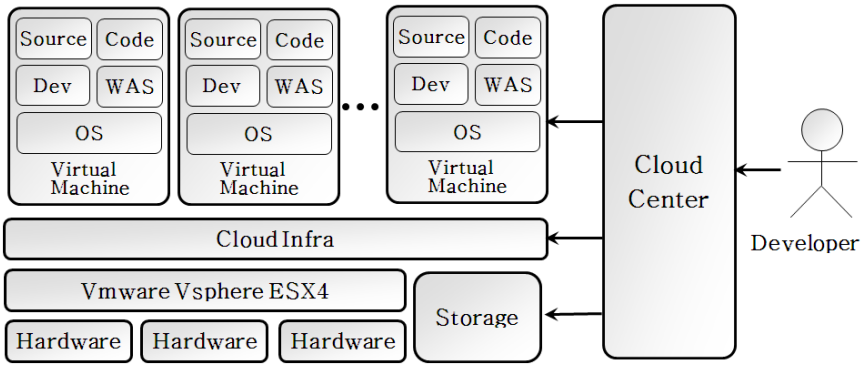
- Python Runtime API : 응용프로그램이 실행되는 Python 환경 제공(CGI, Sandbox, caching, logging)
- Data Repository API : 확장 가능한 데이터저장소 및 효율적인 사용 방법 제공
- Image API : 이미지 데이터 조작 서비스 제공
- Mail API : 응용프로그램으로 이메일 전송 기능 제공
- Memcache API : 분산 메모리 캐시
- URL loading API : 응용프로그램에서 다른 인터넷 호스트에 접근 기능 제공
- User API : 사용자 응용프로그램을 Google 계정과 통합하기 위한 기능 제공

VANADIS SaaS Platoform[9]은 NTT 그룹(NTT Data)에서 제공하는 PaaS 기반 어플리케이션 개발환경 클라우드 서비스로서 개발 및 운영환경, 금융기관 결제제휴, 싱글사인온 등의 개발 API를 서비스 한다. Fig. 3에서 NTT는 Open API를 활용하여 SaaS용 클라우드 서비스를 개발할 수 있다. 인증, 파일관리, 백업 등의 기능을 제공하여 개발자는 개발 시스템 구축에 대한 투자 없이 클라우드 서비스 개발이 가능하다.

Cloud Center[10]는 Fig. 4에서와 같이 PaaS와 IaaS 를 동시에 제공하는 하이브리드 형태의 서비스로서 개발자 자신의 개발 요구사항에 맞추어 CPU, RAM, HDD 등 하드웨어 시스템 및 OS를 포함하여 OS Type, 소프트웨어 개발환경을 선택이 가능하다. 또한 Cloud Center에서는 Uploader를 이용하여 기존에 개발된 웹 스크립트 소스를 특정 구조에 맞추어 개발 후 Zip 형태의 압축 파일로 전송하면 Cloud Center는 IaaS 환경의 PaaS 플랫폼에 업로드 하여 가상의 클라우드 머신에서 바로 웹 어플리케이션 서비스를 제공할 수 있다. Cloud Center 연구는 단순 개발 환경 서비스를 제공하며 본 연구에서 제안하는 재사용성 향상을 위한 서비스와 차이를 보인다.



[Fig. 3] NTT Cloud Service



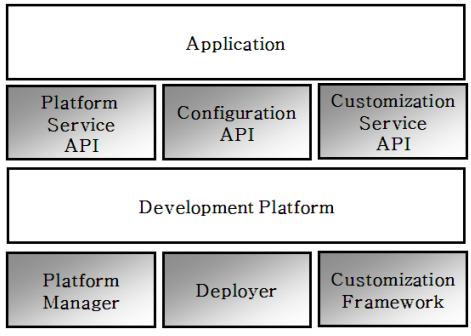
[Fig. 4] Cloud Center Architecture

3. 재사용 기반 PaaS 클라우드 컴포넌트

재사용을 위한 PaaS 기반 클라우드 컴포넌트의 구조와 설정 서비스 및 커스터마이제이션 서비스를 제공하기 위한 구조를 제시한다.

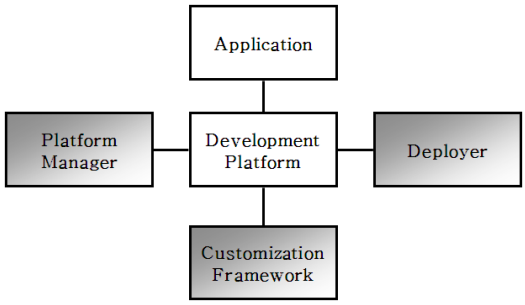
3.1 재사용 PaaS 클라우드 컴포넌트 구조

재사용 PaaS 클라우드 컴포넌트는 개발을 위한 서비스를 제공해야 하므로 개발 플랫폼 및 개발 클라우드 서비스를 제공해야 한다. Fig. 5에서와 같이 개발 플랫폼(Platform)을 중심으로 플랫폼을 제공하기 위한 플랫폼 관리자(Platform Manager), 그리고 플랫폼을 개발자 환경에 설치하기 위한 배포자(Deployer), 그리고 가변적인 부분을 제공할 수 있도록 커스터마이제이션 프레임워크(Customization Framework)으로 구성된다. 이러한 개발 플랫폼을 기반으로 플랫폼 서비스 API, 설정 서비스 API, 커스터마이제이션 서비스 API를 제공하여 컴포넌트를 변경할 수 있도록 제공한다.



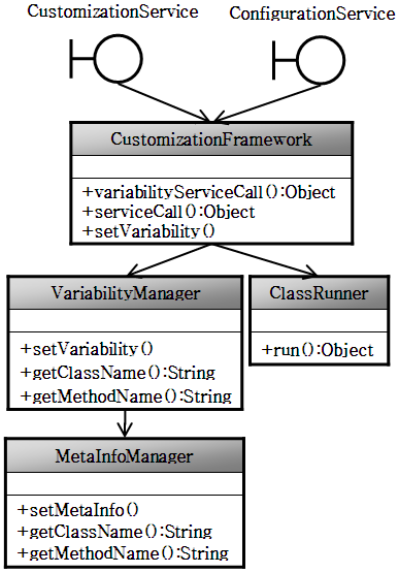
[Fig. 5] Layer Architecture of Reusable PaaS Cloud Component

Fig. 6에서와 같이 개발 플랫폼을 중심으로 플랫폼 관리자와 플랫폼 배포자, 그리고 커스터마이제이션 프레임워크를 이용하여 어플리케이션 개발이 이용될 수 있도록 개발 플랫폼 서비스를 제공할 수 있다.



[Fig. 6] Factors of Reusable PaaS Cloud Component

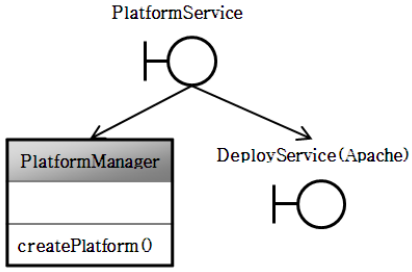
개발 플랫폼 서비스를 제공하기 위한 커스터마이제이션 서비스와 설정 서비스를 제공하기 위한 클래스 다이어그램은 Fig. 7과 같다. 커스터마이제이션 서비스와 설정 서비스는 모두 ‘CustomizationFramework’ 클래스를 이용하여 서비스를 설정한 후에 변경할 수 있다. ‘VariabilityManager' 클래스는 설정 시 클래스 정보와 함수 정보를 설정하거나 호출할 수 있는 기능을 제공하며, 'MetaInfoManager' 클래스는 설정된 클래스 정보와 함수 정보를 메타 정보로 저장하여 관리하는 역할을 담당한다. 이렇게 저장된 서비스들은 ’ClassRunner' 클래스를 통해 호출되어 실행될 수 있다.



[Fig. 7] Class Diagram of Customization Service and Configuration Service

커스터마이제이션 서비스와 설정 서비스는 개발 자 환경에 제공되어 서비스될 수 있도록 패키징(Packaging)되어 서비스 되어야 한다. Fig. 8에서와 같이 'PlatformManager' 클래스는 플랫폼을 생성하여 배포하기 위한 서비스를 제공한다. 이때 배포는 각 웹 서버에서 제공하는 배포 방식을 활용할 수 있다. 본 논문에서는 아파치 톰캣(Apach Tomcat) 웹 서버의 배포 도구를 활용한다.

설정 서비스와 커스터마이제이션 서비스는 RESTful (Representational State Transfer) 서비스 방식으로 제공한다.



[Fig. 8] Class Diagram for Platform Deployment Service

설정 서비스를 제공하기 위한 API는 Fig. 9에서와 같이 정의한다. 설정 서비스명은 ‘ConfigurationService'이며, 서블릿으로 정의한다.서비스 홈 폴더인 ‘PlatformA'는 개발 플랫폼 명이 된다. 플랫폼 명은 배포시 생성해야 하며 배포 파일도 동일하게 정의한다. 전달되는 데이터는 가변성 식별자('VAR_ID'), 변경하고자 하는 클래스 명(‘CLASS’), 그리고 함수명('METHOD')으로 구성된다.

```
http://localhost:8080
 /PlatformA/ConfigurationService
  ?VAR_ID=SampleID
  &CLASS=test.client.ClassB
  &METHOD=method_2
```

[Fig. 9] Configuration Service API

커스터마이제이션 서비스 API는 Fig. 10과 같이 정의한다. 커스터마이제이션 서비스명은 ‘CustomizationService' 이며, 전달되는 데이터는 서비스 받고자 하는 클래스명(’CLASS‘), 함수명(’METHOD‘), 입력 데이터(’PARAM‘), 가변성 식별자(’VAR_ID‘)로 구성된다.

설정 서비스와 커스터마이제이션 서비스의 차이는 운영하기 위한 서비스를 정의하는 것이 설정 서비스이며, 정의된 서비스를 가변성 식별자를 지정하여 변경하는 것이 커스터마이제이션 서비스이다.

```
http://localhost:8080
  /PlatformA/CustomizationService
   ?CLASS=test.client.ClassB
   &METHOD=method_2
   &PARAM=
   &VAR_ID=SampleID
```

[Fig. 10] Customization Service API

플랫폼 서비스 API는 Fig. 11과 같이 정의한다. 플랫폼을 정의하여 패키징된 플랫폼을 배포할 수 있다. 서비스명은 'PlatformService' 이며, 전달되는 데이터는 접근 정보('ID', 'PW')와 플랫폼 명('PlatformName')으로 구성된다. 각 개발자 마다 유일한 플랫폼 명을 정의하여 개발자 환경에 배포하여 설치 후 이용할 수 있다.

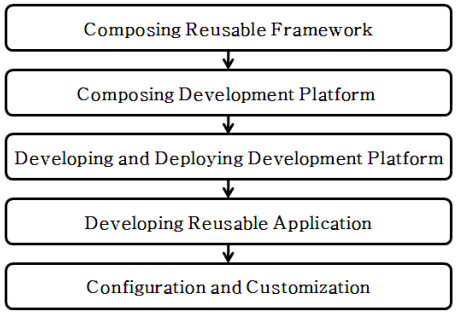
플랫폼 서비스 API에서 정의된 플랫폼 명이 설정 서비스(Fig. 9)나 커스터마이제이션 서비스(Fig. 10)를 호출할 때 사용되는 폴더명이다.

```
http://localhost:8080
  /PaaS_Cloud_Service_for_Variability
  /PlatformService
    ? ID=cjkim
    &PW=1234
    &PlatformName=PlatformA
```

[Fig. 11] Platform Service API

3.2 PaaS 클라우드 컴포넌트 개발 프로세스

PaaS 클라우드 컴포넌트 개발 프로세스는 Fig. 12에서와 같이 재사용 프레임워크 구성한 후 어플리케이션 컴포넌트가 포함된 개발 플랫폼을 구성한다. 이렇게 구성된 개발 플랫폼을 배포하여 재사용 어플리케이션을 개발하며, 기존 어플리케이션을 설정 및 커스터마이제이션 한다.



[Fig. 12] Development Process of PaaS Cloud Component

(1) 재사용 프레임워크 구성 (Composing Reusable Framework)

PaaS 기반의 개발 플랫폼 서비스를 제공하기 위한 프레임워크를 구성하여 패키징하는 단계이다. 엔진 클래스 및 설정 서비스를 위한 UI를 패키징한다.

(2) 개발 플랫폼 구성 (Composing Development Platform)

플랫폼 배포 파일을 생성하는 단계이다. 재사용 프레임워크와 재사용 컴포넌트를 패키징한다. 이 단계에서 패키징되는 컴포넌트는 재사용될 수 있는 배포 단계의 서비스 컴포넌트이다. 이 컴포넌트는 개발자에 의해 커스터마이제이션될 수 있다.

(3) 개발 플랫폼 생성 및 배포(Developing and Deploying Development Platform)

개발 플랫폼 접근권한과 플랫폼 명을 생성한 후, 생성된 개발 플랫폼 배포 파일을 배포한다. 본 논문에서는 아파치 톰캣 웹 서버의 배포 도구를 이용한다.

플랫폼 생성 서비스를 호출하면 Fig. 13과 같이 'PlatformManager' 클래스의 'createPlatform()' 함수를 호출하여 접근 권한를 체크한 후 개발 플랫폼 내의 개발 플랫폼을 위한 'CustomizationFramework' 클래스 등의 내부 클래스에 대한 객체를 생성한다. 이때 각 개발자 별로 플랫폼 객체가 생성된다.

(4) 재사용 어플리케이션 개발 (Developing Reusable Application)

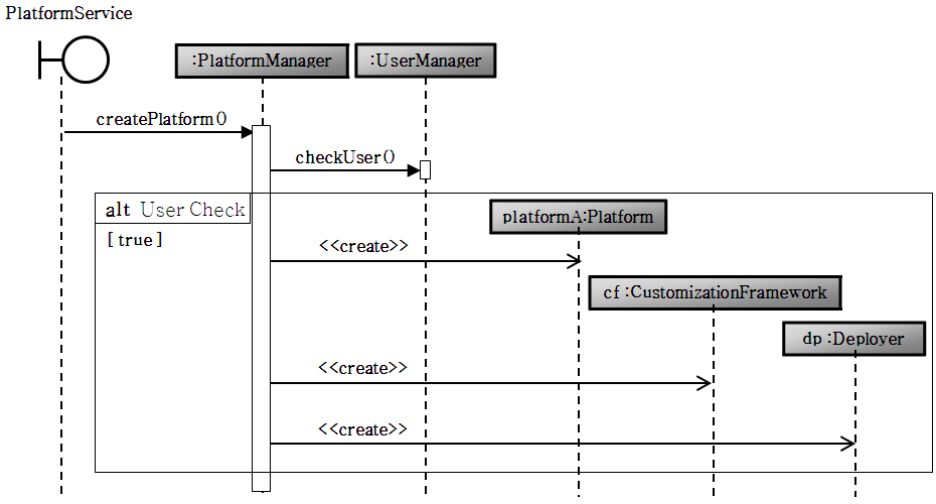
가변적으로 처리될 수 있도록 어플리케이션을 개발한다. 'CustomizationFramework' 클래스를 이용하여 어플리케이션을 개발한다. 배포된 재사용 컴포넌트를 이용하여 커스터마이제이션 API를 이용하여 개발한다.

(5) 설정 및 커스터마이제이션 (Configuration and Customization)

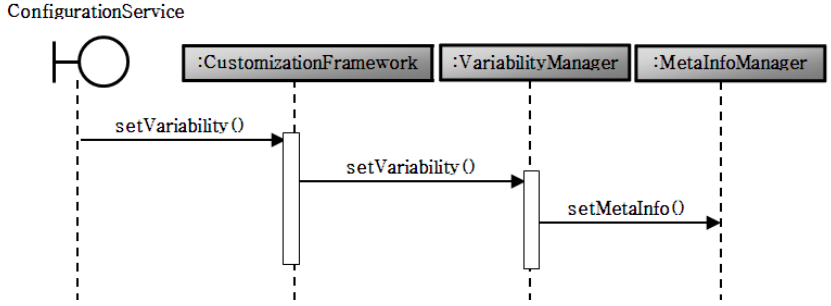
배포된 플랫폼을 기반으로 서비스를 설정하거나 서비스를 호출할 수 있다. 설정 서비스 API와 커스터마이제이션 API를 이용하여 호출 할 수 있다.

가변성 설정 서비스는 Fig. 14와 같이 'Configuration-Service' API를 이용하여 배포된 개발 플랫폼 내의 클래스를 호출한다. 'CustomizationFramework' 클래스, 'VariabilityManager' 클래스, 'MetaInfoManager' 클래스는 설정 서비스를 제공하기 위한 기능을 제공한다.

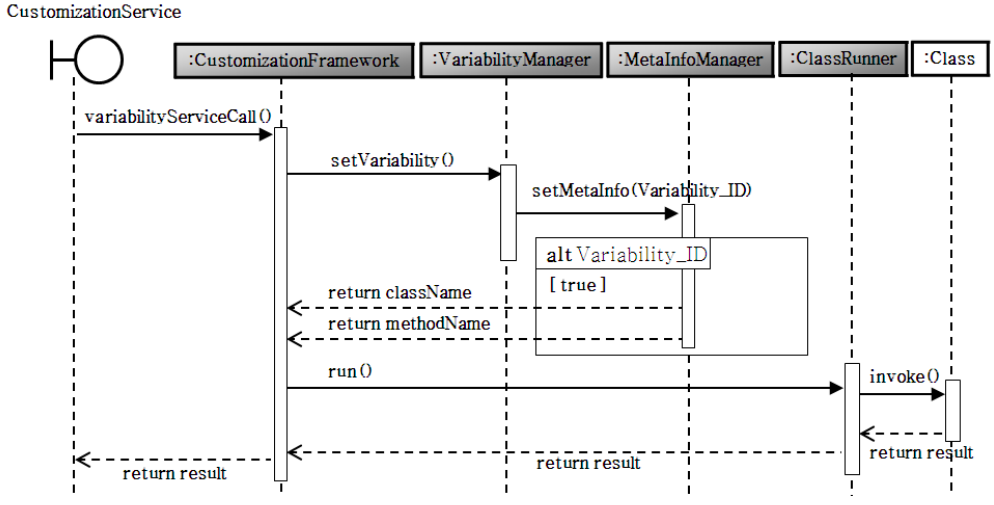
커스터마이제이션 서비스는 Fig. 15와 같이 'CustomizationService' API를 이용하여 배포된 플랫폼 내의 클래스를 이용하여 변경할 수 있다. 가변성 ID를 식별하여 요구하는 서비스를 가변적으로 변경하여 제공할 수 있다.



[Fig. 13] Service Flow of Platform Creation



[Fig. 14] Service Flow of Variation's Configuration



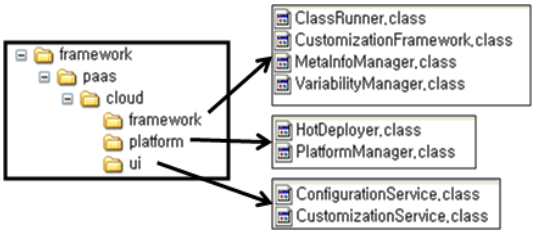
[Fig. 15] Service Flow of Customization

4. 실험 및 평가

사례 연구에서는 본 논문에서 제시한 PaaS 클라우드 컴포넌트 구조를 개발 프로세스에 따라 구현하여 서비스의 적합성을 검증한다. 본 사례 연구는 J2EE 기반의 아파치 톰캣 웹 서버 기반에서 구축하는 것을 전제로 한다.

(1) 재사용 프레임워크 구성

재사용 프레임워크 패키지를 구성하기 위해 Fig. 16과 같이 엔진 클래스 'paas.cloud.framework.*'와 UI 클래스(서블릿) 'paas.cloud.ui.*' 를 jar로 패키지하여 'framework.jar'를 구성한다.



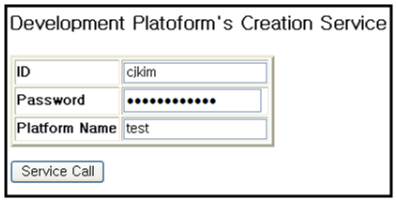
[Fig. 16] Reusable Framework's Organization

(2) 개발 플랫폼 구성

- 개발 플랫폼 배포 파일을 생성하기 위해 어플리케이션 컴포넌트 파일은 '.../test/WEB-INF/classes' 폴더에 복사한다.
- 재사용 프레임워크 패키지인 'framework.jar'는 '.../test/WEB-INF/lib' 폴더에 복사한다.
- 아파치 톰캣 서버의 설정 정보 파일인 'web.xml'에는 설정 UI를 통해 설정 및 커스터마이제이션을 할 수 있도록 서블릿을 등록 한다.
- 'web.xml' 파일을 '.../test/WEB-INF' 폴더에 복사한다.
- 플랫폼 배포 파일 'test.war'을 생성한다.

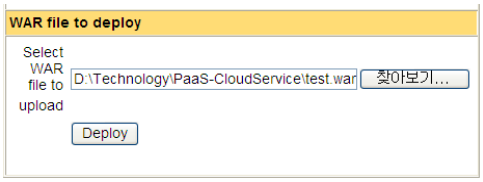
(3) 개발 플랫폼 생성 및 배포

플랫폼 서비스 API(Fig. 13)를 이용하여 Fig. 17과 같이 플랫폼 접근 권한(ID, 암호)와 플랫폼 명을 입력하여 플랫폼을 생성한다. 플랫폼명은 플랫폼 파일명과 동일하게 생성하여 생성된 폴더로 식별가능하게 한다.



[Fig. 17] Creation Service of Development Platform

플랫폼의 기본 정보가 생성되면 아파치 톰캣 웹 서버의 'Web Application Manager' Fig. 18를 이용하여 개발 플랫폼 패키지인 'test.war'를 배포한다. 배포 파일을 형태로 폴더가 생성되므로 배포 파일명이 'test'가 플랫폼 명이 된다.



[Fig. 18] Web Application Manager of Apach Tomcat

```
package test.client;

import paas.cloud.framework.CustomizationFramework;

public class ClassA {
CustomizationFramework cf = new CustomizationFramework();

public String method_1(String data) {
        //..
        Object result = (Object)cf.variabilityServiceCall("var_id", new String[]{data});

        return (String)result;
}

public String method_1() {

        //..
        Object result = (Object)cf.variabilityServiceCall("var_id");
        return (String)result;
}}
```

[Fig. 19] Developed Application applying Variation

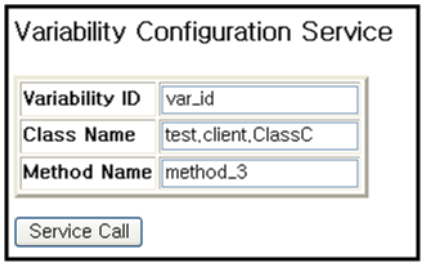
(4) 재사용 어플리케이션 개발

가변적으로 처리될 수 있도록 어플리케이션 'ClassA'를 개발한다. Fig. 19와 같이 'CustomizationFramework' 클래스의 'variabilityServiceCall()' 함수를 이용하여 가변적으로 서비스를 호출한다. 서비스를 호출할 때 가변성 식별자를 입력하여 서비스가 가변적으로 변경될 수 있도록 호출할 수 있다.

(5) 설정 및 커스터마이제이션

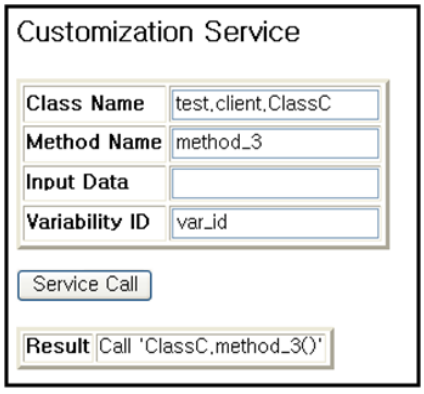
가변성이 포함된 서비스를 호출하거나 커스터마아제이션 서비스를 호출하여 변경할 수 있다. 서비스 호출은 'http://../test/hello.jsp'와 같이 플랫폼 명이 포함된 API를 호출한다.

Fig. 20에서와 같이 새로운 서비스를 등록하고자 할 경우 가변성 ID와 클래스명 함수명을 입력한 후 서비스를 호출하면 등록된다.



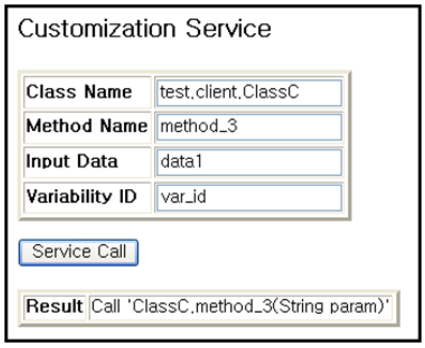
[Fig. 20] Configuration Web Service of Variability

Fig. 21에서와 같이 설정 서비스에서 등록한 가변성 ID을 통해 다른 서비스로 변경할 수 있다. 'ClassC' 클래스의 입력 데이터가 없는 'method_3' 함수로 설정하여 'var_id' 가변성 ID를 이용하여 개발자가 서비스를 이용할 수 있다.



[Fig. 21] Customization Web Service(1)

Fig. 22는 'var_id' 가변성 ID에 입력 데이터가 있는 'method_3' 함수로 변경한다. 개발자가 'var_id' 가변성 ID를 사용하여 개발할 경우 최종적으로 변경된 클래스와 함수가 적용되어 서비스된다.



[Fig. 22] Customization Web Service(2)

개발자에게 모바일 서비스를 통해서도 커스터마이제이션 서비스를 제공할 수 있음을 증명하기 위해 Fig. 23과 같이 안드로이드 기반 모바일 환경에서 적용하였다.

[Fig. 23] Customization Mobile Service

개발자에게 웹 서비스와 모바일 서비스를 통해 설정 및 커스터마이제이션 서비스를 제공할 수 있는 개발 플랫폼을 제공하여 기존 컴포넌트가 개발자에 의해 변경될 수 있음을 실험하였다.

본 사례 연구를 통해 본 논문에서 제시한 PaaS 기반의 개발 플랫폼 컴포넌트를 구축할 수 있음을 파악하였으며, 개발 플랫폼을 개발자에게 제공하여 커스터마이제이션할 수 있음을 증명하였다. 따라서 본 논문에서 제시한 PaaS

개발 플랫폼 클라우드 컴포넌트가 구현 적합함을 증명하였다.

5. 결론 및 향후 연구과제

PaaS 기반의 개발 플랫폼은 개발 환경을 서비스 하므로 개발 환경 구축으로 인한 비용 절감 및 개발 생산성을 향상 시켜 줄 수 있을 뿐만 아니라 개발의 통합 관리가 가능하게 할 수 있다. 본 논문에서는 PaaS 기반의 개발 플랫폼 컴포넌트 구조와 설정 서비스 및 커스터마이제이션 서비스 API를 제공하여 웹 및 모바일 상에서 서비스 할 수 있음을 증명하였다. 향후 연구과제로는 PaaS 기반의 개발 플랫폼 서비스를 통해 개발된 서비스들의 재사용성을 측정할 수 있는 메트릭을 연구하여 재사용성을 정량적으로 평가할 수 있는 방안을 제시한다.

References

[1] ITU-T FG Cloud, http://www.itu.int/en/ITU-T/focusgroups/cloud.

[2] Marco Carugi, Jamil Chawk, Kangchan Lee, Draft deliverable on Introduction to the cloud ecosystem: definitions, taxonomies, use cases, high level requirements and capabilities, ITU-T FG Cloud 6#, 2011. 7.

[3] Lee K. C., W3C DAP(Device API and Policy) Standardization Trends, IT Standard Weekly, 2009. 12.

[4] Boehm, B.W. et al., “Some Experience with Automated Aids to the Design of Large-Scale Reliable Software”, IEEE Trans. On Software Engineering, 1975.
DOI: http://dx.doi.org/10.1109/TSE.1975.6312826

[5] Kim H. G., Lee Y. S., “Trends and Future Prospects of Cloud Computing Service ”, THE JOURNAL OF KOREA INFORMATION AND COMMUNICATIONS SOCIETY, p:31-34, 2010

[6] Peter Mell and Tim Grance, "The NIST Definition of Cloud Computing", Version 15, 10-7-09, 2010.

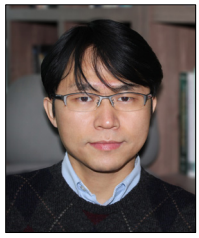
[7] Dhruba Borthakur, "The Hadoop Distributed File System: Architecture and Design", The Apache Software Foundation, 2007.

[8] Google App Engine, http://code.google.com/intl/ko-KR/appengine.

[9] NTT VANADIA SaaS Platform, http://www.ntt.co.jp/saas/index.html.

[10] Lee J. H., Park J. Y., and Huh E. N., “PaaS environment on Cloud System”, Korean Society for Internet Information, 2010 Conference Proceeding, p:247-250, 2010.

김 철 진(Chul-Jin Kim) **[종신회원]**



- 2004년 2월 : 숭실대학교 대학원 컴퓨터학과 (공학박사)
- 2004년 4월 : 가톨릭대학교 컴퓨터정보공학부 강의전담교수
- 2004년 3월 ~ 2009년 2월 :삼성전자 책임연구원
- 2009년 3월 ~ 현재 : 인하공전 컴퓨터시스템과 조교수

<관심분야>
컴포넌트 기반 개발 방법론, 컴포넌트 커스터마이제이션, 모바일 서비스, 클라우드 컴퓨팅