특집 06

소프트웨어 서비스 플랫폼 개발과 그 활용에 관한 연구

이지현·원희선·허성진·최 완 (ETRI)

목 차 »

1. 서 론

소프트웨어를 사용하는 추세가 라이선스 구매 후 설치하여 사용하는 데에서 벗어나 웹 상에서 빌려 쓰는 개념으로 변화해 가고 있다. 본 논문의 목적은 온라인 공유 호스팅 환경을 통해 다중 사용자(multiple users) 그룹에게 어플리케이션 서비스를 제공하기 위한 기술 개발에 관한 것이다. 이는 소프트웨어를 서비스로 제공하기 위한 Software as a Service(SaaS) 플랫폼과 관련된다.

SaaS란 온라인 환경에서 제공되는 요구형 SW (on-demand software)를 제공하기 위한 기술로서 SaaS를 도입하면 온라인 상에서 서비스를 임대하여 사용하고 그에 대한 비용을 지불하는 것이 가능하다. 더불어 운용 서버와 SW를 외부 운용 시스템으로부터 제공받기 때문에 직접 운용 환경을 구축하거나 유지 보수할 필요가 없어 SW 개발에 필요한 호스팅 투자비용을 거의 들이지 않고 최신 IT 기술을 사용하고 유지보수를 할 수 있다.

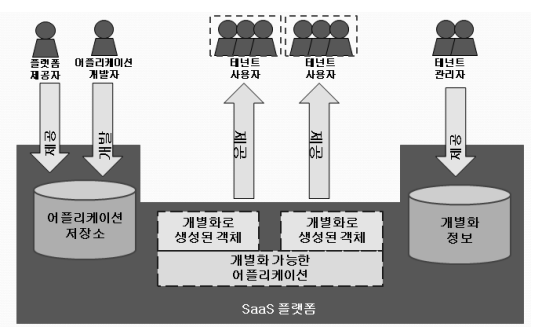
1.1 ASP와 SaaS 비교

SaaS는 기존의 ASP(Application Service Provider) 서비스와 공통점과 차이점이 존재한다. 두 모델 간의 공통점은 서비스 제공자가 SW를 온라인에서 호스팅을 통해 제공하고 서비스 사용자는 인터넷을 통해 SW를 사용한다는 점에 있어 공통점이 존재한다. 차이점으로는 SaaS의 경우 서비스를 사용하는 조직에서 직접 SW를 자체 개별화(customize)할 수 있는 기능을 제공받는 반면 ASP는 서비스 제공자가 기업별 SW를 개별적으로 개별화하여 제공한다는 점에서 사용 방식이 다르다. SaaS의 이런 특징은 다수의 사용자들이 서비스 형태의 SW를 사용할 때 어플리케이션의 기능, 화면 구성, 데이터 저장 구조를 직접 변경하거나 새로운 요소를 추가할 수 있게 한다. 특히 SaaS의 개별화는 SW 서비스를 사용하는 여러 사용자 조직들에게 공유 호스팅 환경을 통해 SW를 제공할 수 있어 여러 사용자에게 SW를 공급할 수 있어 경제성을 실현한다는 장점이 존재한다.

1.2 SaaS 생태계

SaaS에서 사용되는 중요 용어를 살펴보고 SaaS 생태계에 대해 살펴본다. SaaS 어플리케이션은 주로 기업용 어플리케이션으로 개발되고 있다. 이들 어플리케이션의 특징은 조직, 기관, 회사에 속한 여러 사용자가 사용한다는 점이다. 여기서 어플리케이션 사용자 집단은 SaaS에서 테넌트(tenant)라 불리는데 테넌트는 중앙에서 제공되는 플랫폼을 임대하여 사용하는 주체가 된다. 테넌트의 관리자는 플랫폼 기반의 SW를 자사의 요구 사항에 따라 직접 개별화하여 필요로 하는 기능을 갖도록 수정하고 업데이트한다. 수정이 완료되면 테넌트는 단말 사용자들에게 서비스 형태로 SW를 제공한다. 이 경우 공급자에 의한 SW 개별화가 아닌 자사가 직접 요구 사항에 따라 개별화할 수 있어 사용자 중심의 SW 변경이 가능하다.

(그림 1)은 공급자가 중앙에서 제공하는 SaaS 플랫폼을 기반으로 어떤 이해당사자들이 SW를 어떻게 개발하는 지 보여준다.



(그림 1) SaaS 생태계

SaaS 플랫폼은 어플리케이션을 온라인 서비스로 제공하고 테넌트가 직접 서비스를 개별화하기 위해 필요한 기술들의 집합이다. SaaS 플랫폼의 이해당사자들의 역할은 <표 1>과 같다.

〈표 1〉 SaaS 이해당사자의 역할

이해당사자	역할 설명
플랫폼 제공자	⊙ SaaS 플랫폼 개발/제공 ⊙ 개별화 기능 제공
어플리케이션 개발자	⊙ 멀티테넌트 어플리케이션 개발
테넌트 관리자	⊙ 어플리케이션 개별화
테넌트 사용자	⊙ 최종 어플리케이션 사용

플랫폼 제공자는 SaaS 플랫폼을 개발하여 제공하고 테넌트 별 사용자 인터페이스, 비즈니스 로직, 데이터를 구성할 수 있도록 중앙에서 관리 및 지원되는 개별화 기능을 제공한다. 어플리케이션 개발자는 멀티테넌트 어플리케이션을 개발하여 제공한다. 멀티테넌트 어플리케이션은 SaaS 플랫폼을 기반으로 동작하고 테넌트 관리자가 개별화하기 전 SW로서 테넌트 별 수정을 필요로 한다. 그리고 테넌트 관리자는 사용 계약을 맺은 서비스에 대해 테넌트 요구 사항을 반영하여 어플리케이션을 수정한다. 테넌트 사용자는 테넌트에 속하며 서비스로 제공되는 최종 어플리케이션을 사용하는 주체가 된다.

2. SaaS 플랫폼 상세 설계 고려 사항

SaaS 생태계에는 멀티테넌트가 존재하고 SaaS 지원 플랫폼을 제공한다. 그리고 SaaS 플랫폼은 멀티테넌시(multitenancy)를 지원하는데 기반을 둔다. 멀티테넌시는 다수개의 기업에 대해 단지 하나의 동일한 어플리케이션 객체를 동작시켜 멀티테넌트에게 필요한 서비스를 제공하는 것을 의미한다[1]. 플랫폼 제공자는 멀티테넌시가 지원되는 공유 플랫폼을 제공하고 다수의 테넌트는 이를 사용함으로써 기존의 소프트웨어 서비스를 개발 및 유지 보수하는 비용을 낮출 수 있으므로 SaaS 플랫폼을 통해 규모의 경제를 추구할 수 있

다. 공유 플랫폼을 통한 테넌트 별 어플리케이션을 생성하기 위해 SaaS 플랫폼은 테넌트에게 SaaS 어플리케이션을 개별화하기 위한 구성변경성(customizability)을 제공해야 한다. 그리고 다중 테넌트들이 어플리케이션의 구성을 변경하기 위한 편의 기능을 이용하여 테넌트 관리자는 스스로 기업 요구사항에 맞는 서비스를 쉽게 생성할 수 있어야 한다. SaaS 플랫폼과 SaaS 어플리케이션을 개발하는데 고려해야 할 사항을 SaaS 성숙도 레벨, 멀티테넌트 플랫폼 아키텍처, 멀티테넌트 지향 어플리케이션 개발을 통해 알아보자.

2.1 SaaS 성숙도 레벨

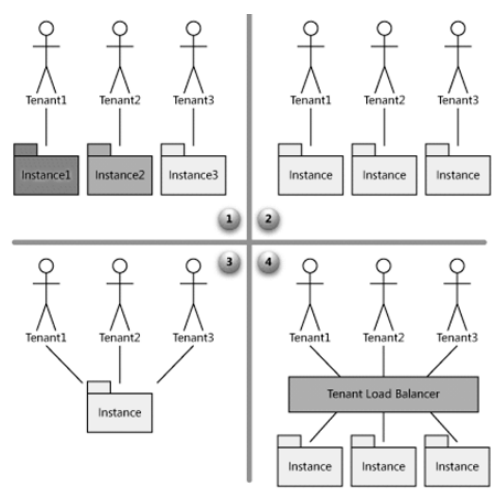
본 논문에서는 (그림 2)와 같이 SaaS 성숙도 레벨을 4가지로 분류한다.

	Level 1	Level 2	Level 3	Level 4
개념 모델	APP Instances OS, Middleware Hardware	APP Instances VM/Partition/OS Hardware	APP Instances OS, Middleware Hardware	Tenant Load Balance APP Instances OS, Middleware Hardware
자원 공유	X	O	O	O
Configuration	X	O	O	O
다중 테넌트 지원	X	X	O	O
메타데이타 지원	X	X	O	O
시스템 확장성	X	X	X	O
국내외 기술수준	국내 업체		Salesforce.com WaveMaker	

(그림 2) SaaS 성숙도 레벨의 특징

각 레벨 별 특징으로 레벨 1은 테넌트 별로 전용 어플리케이션, 미들웨어, 하드웨어를 제공한다. 레벨 2는 공유 하드웨어 시스템 위에서 동작하는 어플리케이션을 테넌트 별로 제공하며, 테넌트 별로 어플리케이션을 개별화한다. 레벨 3은 어플리케이션이 동작하기 위한 하드웨어 및 미들웨어의 공유가 이뤄지고 메타데이터를 통한 개별화 정보를 관리함으로써 멀티테넌트가 공유 플랫폼을 통해 개별화된 어플리케이션을 제공받을 수 있다. 레벨 4는 레벨 3에 시스템 확장성이 가미된 구조로 부하 분산이 가능해져 테넌트가 필요로 할 때 효율적으로 확장된 어플리케이션과 플랫폼을 제공한다.

위의 SaaS 성숙도 레벨은 6가지로 분류된 Forrester 리서치의 SaaS 성숙도 레벨[2]과 4 가지로 분류된 Microsoft의 SaaS 어플리케이션 속성[3]에 기반을 두어 정의되었다. 참고로 Forrester 리서치의 SaaS 성숙도 레벨은 SaaS가 적용되지 않은 레벨 0, 테넌트에게 어플리케이션 인스턴스가 동작하는 전용 서버를 제공하고 같은 방식으로 개별화하는 레벨 1, 같은 소프트웨어 패키지를 호스팅하며 사용자 별로 개별화된 레벨 2, SaaS 지원 기능을 갖는 어플리케이션을 개발하여 제공하며 제한적 개별화가 가능한 레벨 3, 잘 정의된 비즈니스 어플리케이션 뿐 아니라 테넌트별 확장 기능을 개발할 수 있는 레벨 4, 레벨 4보다 특화되도록 멀티테넌트 지원 동적 오케스트레이션(orchestration)이 가능한 레벨 5로 분류된다. (그림 3)은 Microsoft의 SaaS 어플리케이션 성숙도 레벨을 표현한다.



(그림 3) Microsoft의 SaaS 어플리케이션 성숙도 레벨 [2]

Microsoft는 SaaS 어플리케이션의 속성에 따라 4개 레벨로 분류한다. 레벨 1은 테넌트 별로 서로 다른 어플리케이션 인스턴스를 따로 제공하고, 레벨 2는 테넌트들에게 같은 코드로 구성된 어플리케이션 인스턴스를 따로 제공하고 각각 개별화하며, 레벨 3은 테넌트들에게 설정 가능한(configurable) 메타데이터와 함께 하나의 공통된 어플리케이션을 제공하고, 레벨 4에서는 부하 분산 기능을 두어 테넌트들이 효율적으로 SaaS 어플리케이션과 플랫폼이 확장될 수 있도록 한다.

2.2 멀티테넌트 플랫폼 아키텍처

SaaS 플랫폼은 어플리케이션을 온라인 서비스로 제공하며 테넌트 측 개발자의 개발 용이성을 확대하는데 필요한 기술들을 스택으로 표현한다. SaaS 플랫폼에 대해 서로 다른 구성 요소를 갖는 3가지 참조 아키텍처를 정리하고 각 아키텍처의 특징을 살펴본다[4,5].

(그림 4) SaaS 플랫폼 참조 아키텍처[4]

(그림 4)의 (1)은 공유 프로세스 기반의 아키텍처이다. 데이터 플랫폼으로 불리는 데이터베이스는 물리적으로 분리되고 다수 테넌트의 트랜잭션이 하나의 특정 어플리케이션 서버에서 수행되는 구조이다. 테넌트 별 어플리케이션에 대한 동작 요청 시 테넌트의 트랜잭션은 메타데이터를 데이터베이스에서 가져와 동적으로 사용자 인터페이스, 비즈니스 로직, 데이터를 구성한다. 이 때 물리적 메타데이터를 해석할 수 있는 특별 가상 머신이 필요하다.

(그림 4)의 (2)는 SaaS 플랫폼의 전체를 공유하는 것이 특징이다. 공유 프로세스 기반의 아키텍처와 다른 점은 물리적인 데이터베이스를 테넌트별로 분리하지 않고 하나의 데이터베이스를 논리적 데이터베이스로 공유하여 사용한다는 점이다.

(그림 4)의 (3)은 위의 SaaS 플랫폼 전체를 공유하는 것을 전제로 멀티테넌트들에게 어플리케이션의 구성변경성을 지원함으로써 맞춤화된 멀티테넌시 지원 아키텍처를 추가적으로 제공한다. 이는 어플리케이션 설계 및 구현에 구성 변경 알고리즘을 반영하고 이러한 설계 및 구현 요건들이 해석되는 범용 어플리케이션 플랫폼(즉, 범용적인 SaaS 플랫폼)을 제공한다. 이 아키텍처는 제시된 세 가지 SaaS 플랫폼 중 설계 및 구현 수준이 가장 높다[6]. Salesforce.com은 이 아키텍처를 지원하고 한국전자통신연구원의 SaaSpia™ 역시 범용적인 플랫폼을 지원하도록 개발 중에 있다.

2.3 멀티테넌트 지향 어플리케이션 개발 및 개별화

SaaS 플랫폼의 핵심 기술은 멀티테넌트 어플리케이션을 개발하여 테넌트 별 요구사항이 반영되도록 하는데 있다. 멀티테넌트 어플리케이션은 테넌트의 요구사항을 반영하여 테넌트가 원하는 대로 구성이 변경된 어플리케이션을 생성할 수 있다는 것이다. 이를 위해 SaaS 플랫폼은 멀티테넌트 어플리케이션을 개발하기 위한 환경과 테넌트 별 어플리케이션 구성변경을 위한 설정 환경이 필요하다. 개발 환경은 멀티테넌트가 사용할 어플리케이션을 테넌트 별 개별화할 수 있는 부

분을 포함하여 개발한 후 웹 서버에 배포하여 서비스로 제공한다.

멀티테넌트 지향 어플리케이션은 2.1절에서 정의된 SaaS 성숙도 레벨 3과 4가 되어야 공유 자원을 기반으로 개별화 어플리케이션을 제공할 수 있다. 레벨 3을 지원하는 멀티테넌트 어플리케이션의 동적 설정은 하나의 어플리케이션 인스턴스에 테넌트 별로 변경하거나 추가하는 웹 페이지와 기능이 런타임에 얼마나 효과적으로 테넌트 별로 저장되고 요청 시 서비스 될 수 있는가에 달려 있다. 덧붙여 레벨 4를 지원하는 멀티테넌트 어플리케이션은 테넌트가 멀티 어플리케이션을 사용하거나 공유 자원의 확장이 필요한 경우 부하를 분산시켜 효과적으로 확장될 수 있도록 지원하는 기능이 레벨 3의 성숙도에 추가된다.

테넌트 요구사항들은 SaaS 플랫폼에서 테넌트 데이터로 관리되고 테넌트 별 서비스가 요청될 때 테넌트 데이터와 멀티테넌트 어플리케이션 코드를 이용해 동적으로 서비스를 생성하여 테넌트에게 제공한다.

테넌트 별 어플리케이션을 개발하기 위해서 개별화 가능한 부분은 기존 연구들을 통해 어플리케이션을 구성하는 사용자 인터페이스, 데이터 스키마, 비즈니스 로직으로 정하고 있다[7,8]. 설정 환경은 웹 서버에 배포된 서비스로 테넌트 별로 수정하거나 추가하고 싶은 사용자 인터페이스, 비즈니스 로직, 데이터 저장 구조, 사용자 접근 정보 등을 동적으로 반영하여 개별화된 서비스를 만들어 테넌트 사용자들에게 제공한다.

2.4 테넌트 데이터 관리

테넌트 별 어플리케이션은 테넌트 별로 확장될 부분을 포함하는 멀티테넌트 어플리케이션과 테넌트 데이터가 결합하여 생성된다. 테넌트 데이터는 멀티테넌트 어플리케이션을 개별화할 때 생성되며 분산 데이터베이스 기법, 공유 데이터베이스-분산 스키마 기법, 공유 데이터베이스-공유 스키마 공유 서버 기법에 대해 살펴본다[9].

공유 자원과 멀티테넌트 어플리케이션 코드는 일반적으로 하나의 서버를 통해 공유되나 분산 데이터베이스 기법은 테넌트 별로 분리된 데이터베이스를 사용하여 데이터를 저장하는 기법이다. 분산 데이터베이스 기법은 테넌트 데이터를 개별 데이터베이스로 나눠 저장하여 데이터에 대한 높은 보안이 제공된다. 하지만 시스템 오류가 발생 시 테넌트 데이터를 복원하기가 쉽지 않고 서버 증설과 유지보수 비용이 증가하기 때문에 데이터 분리가 강력히 필요한 은행 정보나 의료 정보 관리를 위해 사용된다.

공유 데이터베이스-분산 스키마 기법은 멀티테넌트가 같은 데이터베이스를 사용하고, 각 테넌트가 자신의 테이블을 가지고 테넌트 데이터를 저장하고 관리하는 기법이다. 이 기법은 분산 데이터베이스 기법만큼 쉽게 구현할 수 있고 분산 데이터 기법보다 데이터베이스에 더 많은 테넌트 정보를 저장할 수 있다. 하지만 상당히 작은 수의 데이터베이스 테이블을 사용하는 어플리케이션의 경우에 적합하다.

공유스키마 방식은 테넌트 테이블을 확장시키기 위해 각 테이블은 추가적인 테이블 칼럼이 있고 모든 테넌트의 데이터가 하나의 스키마에 저장되는 구조를 제공한다.

3. SaaSpia™ 플랫폼 개발 현황

한국전자통신연구원은 다양한 어플리케이션 SW를 중소기업에 저비용, 고효율로 온라인 서비스화하기 위한 SaaS 플랫폼 개발과제를 2009년

부터 수행하고 있다. SaaSpia™는 SaaS와 Utopia를 결합한 것으로 개발된 SaaS 플랫폼의 이름이다. SaaSpia는 범용적인 SaaS 플랫폼 아키텍처 기반으로 SaaS 성숙도 레벨 3 수준의 어플리케이션을 개발하고 운영할 수 있는 플랫폼 기술을 개발한다. 본 절에서는 멀티테넌트 지원 플랫폼 개발 기술과 개발 현황에 대해 살펴본다.

3.1 SaaSpia™ 소개

SaaSpia™는 동일한 SW와 데이터베이스를 이용하여 서비스하고 테넌트 별 접근 독립성과 어플리케이션 개별화를 지원하여 테넌트 별 자사만의 서비스를 생성하여 사용할 수 있게 하는 멀티테넌트 지원 플랫폼이다.

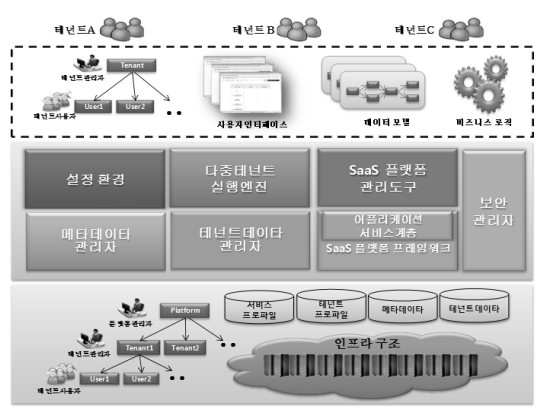
SaaSpia는 SaaS 설정 도구, 멀티테넌트 실행 엔진, 메타데이터 관리자, 플랫폼 통합 관리도구로 구성된다. SaaS 설정 도구는 SaaSpia 기반의 어플리케이션을 사용하는 테넌트가 자사에서 사용할 서비스를 만들기 위한 기능을 제공하는 도구이다. 어플리케이션에서 설정할 수 있는 사용자 인터페이스, 데이터 스키마, 비즈니스 로직 및 워크플로우에 대해 테넌트 필요에 따른 설정이 가능하며 이를 통해 SaaSpia 기반으로 개발된 어플리케이션이 변경된다. SaaS 어플리케이션 설정 도구가 이와 같이 동작하기 위해 SaaSpia는 어플리케이션 설계 및 구현 시 구성 변경을 미리 고려하여 표현하고 실행 요청 시 동적으로 해석될 수 있는 알고리즘과 실행 엔진을 제공한다.

멀티테넌트 실행 엔진은 멀티테넌시 지원 아키텍처의 테넌트 데이터를 해석할 수 있는 기술의 구현이다. 이로써 테넌트 별 어플리케이션 개별화 정보와 멀티테넌트 어플리케이션을 기반으로 테넌트 별 독립적인 서비스를 생성하여 사용자에게 제공한다.

메타데이터 관리자는 하나의 데이터베이스로 멀티테넌트의 논리적 데이터를 저장하고 관리하기 위한 기술을 구현한 것이다. SaaSpia 설정 도구와 실행 엔진을 통해 테넌트 어플리케이션 생성에 필요한 메타데이터 접근을 지원하고 테넌트 별로 설정된 요소들을 테넌트 데이터로 저장하고 관리한다. 또한 어플리케이션 실행 시 메타데이터 지원 API를 이용해 테넌트 별로 분리된 데이터를 제공하며 테넌트 별 추가 확장 가능한 여분의 데이터 필드를 제공한다.

플랫폼 통합 관리 도구는 SaaSpia 플랫폼 관리자를 위한 시스템, 테넌트, 사용자 및 서비스의 통합 관리 도구로서 실행 중인 서비스 별 플랫폼 사용 정보의 실시간 모니터링 제공, 테넌트 별 플랫폼 사용 정보 관리 기능 등을 제공한다.

SaaSpia 플랫폼의 구조는 (그림 5)와 같이 표현되며 SaaS 설정 환경, 다중테넌트 지원 실행 엔진, 메타데이터 관리자, 플랫폼 관리 도구 등의 구성 요소를 보여준다.



(그림 5) SaaSpia™ 플랫폼의 구조

3.2 멀티테넌트 어플리케이션 생성 및 설정 기술

SaaSpia™ 플랫폼은 기존의 웹 어플리케이션

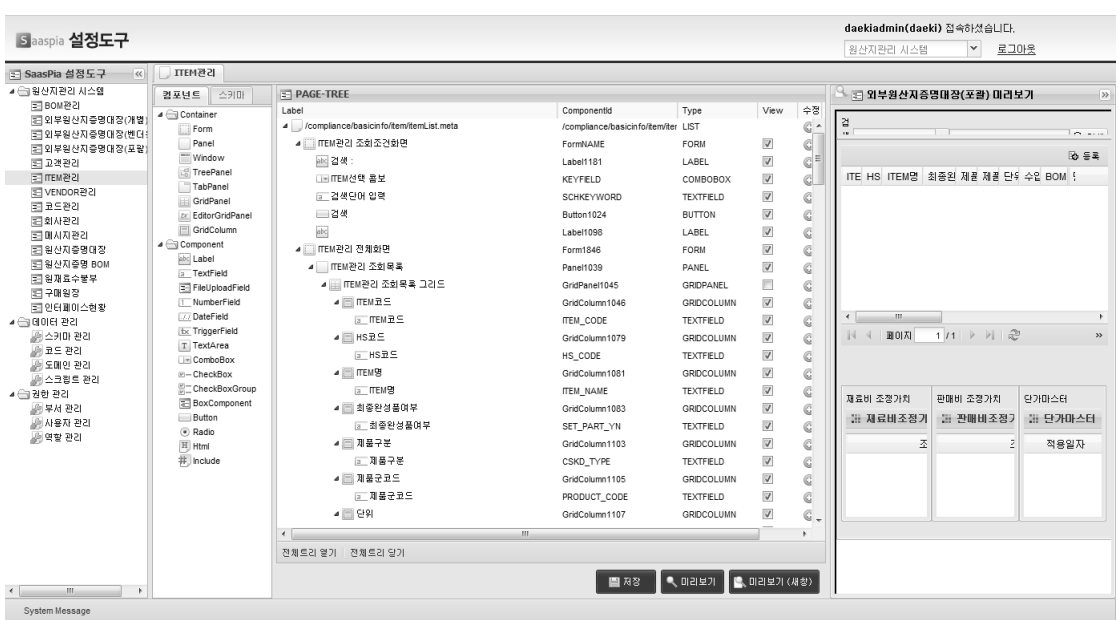
이 멀티테넌트 구조로 동작하기 위해 변환하는 것을 지원한다. SaaSpia 플랫폼 상에 통합되기 위해 기존 웹 어플리케이션은 ExtJS와 Spring 기반의 자바 구현 언어로 구성되어 있어야 한다. SaaS 설정 도구 사용 전에 웹 페이지 생성 기능을 통해 기존 웹 어플리케이션은 메타데이터로 변환되고 멀티테넌트 공유 데이터베이스에 저장되어야한다.

SaaSpia 설정도구는 테넌트 별 요구사항을 SaaS 기반 멀티테넌트 어플리케이션에 반영할 수 있는 환경을 제공한다. 어플리케이션에서 설정 가능한 부분은 사용자 인터페이스, 데이터 스키마, 비즈니스 로직 단위로 관리된다. SaaSpia 어플리케이션 설정 도구는 (그림 6)과 같다.

사용자 인터페이스의 경우 ExtJS 기반의 자바 스크립트로 구현된 웹 페이지로 웹 페이지를 구성하는 컴포넌트들이 트리 모델로 계층화되어 관리된다. 테넌트 별로 웹 페이지는 페이지를 구성하는 컴포넌트의 위치 재조정, 레이블 변경 및 관련 테이블 정보를 변경함으로써 설정한다. 설정된 정보는 페이지 단위로 조합되어 렌더링 되어 사용자에게 웹 페이지로 제공된다.

데이터 스키마는 공유 스키마 기반의 테넌트 데이터 관리 기법으로 관리된다. 어플리케이션에 공통적으로 제공되는 데이터 스키마와 테넌트 별 필요에 의해 확장 가능한 스키마를 제공한다. 데이터 스키마 설정 시 모든 테넌트들은 공통적으로 고정필드를 제공받아 필요한 타입으로 정의하여 사용되고 테넌트 별 확장필드 사용 여부가 이때 결정된다.

비즈니스 로직은 UI를 통해 데이터베이스와 주고받는 데이터의 처리 및 가공에 관련된 정보로서 스크립트 기반과 자바 코드 기반의 비즈니스 로직을 변경하여 설정할 수 있는 기능을 제공한다. 스크립트 기반의 비즈니스 로직은 UI 컴포넌트에 연결되어 간단한 수준의 연산을 설정하여 처리되도록 한다. 자바 코드 기반의 비즈니스 로직은 데이터 처리 관련 생성, 읽기, 갱신, 삭제와



(그림 6) SaaSpia 어플리케이션 설정 도구

관련된 기능을 SQL로 처리한 후 추가적으로 데이터 가공을 위해 필요한 로직을 설정하여 UI 컴포넌트를 통해 동작이 요청된다.

워크플로우 설정은 비즈니스 로직이 일련의 처리할 업무 흐름으로 표현될 수 있는 경우 처리해야 할 업무의 상세 정보, 담당자, 처리 조건을 명시하고 정의된 워크플로우의 흐름대로 수행될 수 있도록 한다. 워크플로우 개별화는 워크플로우를 구성하는 액티비티의 추가, 액티비티 속성 변경, 연결된 UI 매핑 정보 변경을 통해 수행된다.

3.3 사례 연구

SaaSpia™ 플랫폼을 기반으로 자유무역협정(이하 FTA) 원산지관리시스템 등의 적용 서비스를 선정하고 개발하여 사례 연구를 개발하였는데 현재 시범 서비스 단계에 있다.

원산지 관리는 90년대 이후 자유무역협정(FTA)이 체결되면서 FTA에 적극적인 대응하고자 필요한 기술이다. SaaS 플랫폼의 사례로 개발하게 된 배경을 살펴보자. FTA 협정이 늘고 있고 기업이 수출에서 관세 혜택을 받기 위해 체계적인 관리가 필요했으나 협정이 추가될 때마다 변경되거나 다르게 적용되는 기준 때문에 원산지관리를 위한 어플리케이션은 수정되거나 새롭게 개발되어야 했다. 하지만 이를 관리하고 개발하기 위한 인력과 자금이 수출입 중소기업에게 있어 어려운 실정이다[10]. SaaSpia 기반 원산지관리시스템은 FTA 대응이 어려운 중소기업에게 중앙집중식 공유 기반 시스템을 제공하며 저비용으로 효율적인 온라인 SW 서비스 사용이 가능했다.

또한 개별적인 요구사항에 맞게 서비스를 설정하여 기업별 개별화된 서비스를 구축할 수 있어 여러 협력업체와 협업하는 중소기업들이 온라인 SW를 임대하여 사용할 수 있고 신속하게 원산지 관리 과정에 필요한 서비스를 저비용에 구축할 수 있었다. 특히 완제품을 생산하는 업체에서 협력업체가 제공한 원산지 증명서를 확인하여 완제품의 원산지 판정에 필요한 정보를 관리하고 원산지 최종 증명서를 출력하는데 적극 이용될 수 있어 그 유용함을 확인할 수 있었다.

4. 관련연구

SaaS 플랫폼 개발 기술은 상용화되어 클라우드 컴퓨팅 플랫폼으로 국외에서 제공되고 있다. 국내에 비해 국외 SaaS 플랫폼 개발은 SaaS 기술 성숙도 레벨이 안정된 3~4에 이르고 있고 온라인에서 SaaS 어플리케이션을 개발하는 환경과 협업 환경을 제공하는데 앞서 있는 상태이다. 이 절에서 국외에서 현재 제공되는 두 가지 플랫폼에 탑재된 기능들과 그 기능들의 특징에 대해 살펴본다.

4.1 Salesforce.com

Salesforce.com은 SaaS 플랫폼 개발의 선두업체로 자체 개발한 엔터프라이즈용 클라우드 컴퓨팅 플랫폼인 Force.com을 기반으로 여러 다양한 기업용 어플리케이션을 제공하고 있다. Force.com은 높은 확장성(scalability)과 생산성을 제공함으로써 다양한 어플리케이션을 서비스하고 개발할 수 있다. 이러한 이유로 Force.com은 가장 성숙한 클라우드 어플리케이션 플랫폼으로 평가받고 있다[11].

Force.com에서는 비즈니스 어플리케이션을 개발하기 위한 환경으로 비즈니스 어플리케이션 개발을 위한 Appforce, 웹사이트 개발을 지원하는 Siteforce, SaaS 기반 비즈니스 어플리케이션 개발을 위한 ISVforce를 제공한다. SaaS 어플리케이션의 유통을 위해 AppExchange를 통한 SaaS

마켓플레이스를 제공한다.

Force.com은 비즈니스 어플리케이션을 개발하기 위한 자체 개발 언어로 Visualforce, Apex, SOQL(Salesforce Object Query Language)을 제공하여 차례대로 사용자 인터페이스, 비즈니스 로직, DB query 개발에 이용한다. 이들 개발 언어를 통해 Force.com에서 CRM 등의 다양한 어플리케이션을 테넌트 요구사항에 맞게 개별화하거나 새롭게 개발할 수 있다.

웹 어플리케이션은 대부분 데이터 처리 중심의 어플리케이션이기 때문에 데이터베이스에 저장된 데이터에 대한 효과적인 저장 및 인출이 중요한데 Salesforce.com은 클라우드 컴퓨팅을 위한 DBMS인 Database.com을 제공한다. Database.com은 표준 기반의 API를 통해 어떤 플랫폼이나 어떤 언어로든 클라우드 DBMS 서비스에 접근할 수 있게 하여 클라우드 DBMS 기반으로 동작할 수 있는 어플리케이션을 개발한다. Database.com은 Sales Cloud, Service Cloud, Force.com의 일부로서 현재 제공되고 있다.

4.2 Wavemaker

WaveMaker Software에서는 민첩한 어플리케이션 개발 환경(rapid application development environment)인 WaveMaker를 제공함으로써 웹 2.0 어플리케이션의 빠른 개발을 유도한다. WaveMaker와 WaveMaker 어플리케이션은 기업 표준 및 Dojo, Spring, Hibernate, JAXWS, JSON-RPC, Acegi와 같은 오픈 소스 기술에 기반을 둔다.

WaveMaker는 내부적으로 WaveMaker Studio와 WaveMaker Enterprise Server로 구성된다. 이 중 WaveMaker Studio는 클라우드 어플리케이션을 개발하기 위해 시각화된 도구로서 클라우드 어플리케이션을 구성하는 위젯, 폼, 차트 등의 웹 페이지를 구성하는 컴포넌트를 웹 브라우저 상에서 드래그와 드롭을 통해 개발할 수 있도록 지원한다. WaveMaker Studio의 개발 결과는 표준 Java 플랫폼에 배포되고, Java IDE를 사용하여 수정 및 확장할 수 있다.

WaveMaker Enterprise Server는 표준 Java 컴포넌트를 이용한 데이터 접근, 보안, 어플리케이션 확장을 지원한다. 또한 WaveMaker 어플리케이션이 테넌트 별로 제한된 DB에 접근할 수 있는 아키텍처를 제공하여 멀티테넌트 환경을 지원한다. 또한 WaveMaker에서 개발된 어플리케이션은 웹 어플리케이션 서버뿐만 아니라 Amazon EC2, Rackspace, OpSource, Eucalyptus 등의 호스팅 환경에 배포되고 클라우드 인프라 관리 솔루션인 RightScale에 의해 관리될 수 있다.

WaveMaker 버전으로 Community Edition은 무료로 아파치 오픈 소스 라이선스를 가지며, Enterprise Edition은 유료로 추가적인 기능이 제공되며 기술 컨설팅이 지원된다. 또한 Enterprise Edition에서는 역할 기반 보안, LDAP 지원, 상용 DB 접근 및 자동화된 멀티테넌트 지원 기능이 제공된다.

5. 결 론

본 논문에서는 SaaS를 서비스로 제공하기 위한 SaaS 플랫폼 개발을 위한 상세 설계 시 고려해야할 사항들에 대해 살펴보았다. 현재 한국전자통신연구원에서 개발되고 있는 SaaSpia™ 플랫폼의 개발 현황을 통해 멀티테넌시를 지향하는 플랫폼 핵심 기술의 구현 및 원산지관리 분야로의 적용 사례를 살펴보았다. SaaSpia™ 플랫폼은 FTA 원산지 관리와 관련 있는 민간 기업들로 기술 이전 중에 있으며, 플랫폼 최적화와 비즈니스 로직 확장을 지원하기 위해 국내 자체 기술로 개발 중에 있다.

참 고 문 헌

[1] Gartner, “Hype Cycle for Software as a Service”, 2010.

[2] Stefan Ried, “Forrester's SaaS Maturity Model,” Forrester Research, August, 2008.

[3] Frederick Chong, Gianpaolo Carraro, “Architecture Strategies for Catching the Long Tail,” MSDN, April, 2006.

[4] Yefim V. Natis, David W. Cearley, Eric Knipp, “Salesforce.com at an Inflection Point”, Gartner, Jan., 2011.

[5] 이지현, 박경현, 이상민, 이원재, 오병택, 강성주, 정문영, 양경아, 원희선, 허성진, 최완, “SaaS 플랫폼 기술 및 개발 동향,” 전자통신동향분석 제26권 제5호, 2011년.

[6] 이지현, 허성진, “이해당사자의 관점에서 분류한 SaaS 플랫폼 역량”, 제 32회 한국정보처리학회 추계학술대회 논문집 제16권 2호, 2009.

[7] “The Forece.com Multitenant Architecture,” Force.com

[8] Wei Sun, Xin Zhang, Chang Jie Guo, Pei Sun, Hui Su, “Software as a Service: Configuration and Customization Perspectives,” Proc. of IEEE Congress on Services Part II, pp.18-25, 2008.

[9] Frederick Chong, Gianpaolo Carraro, Roger Wolter, “Multi-Tenant Data Architecture,” MSDN, 2006.

[10] 한국무역협회 및 지식경제부, “사례를 통해 배우는 FTA 원산지 길라잡이”, 2009.

[11] Rick Greenwald, “Forece.com Developer Guide-Advanced Programming Techniques for Cloud Computing,” Salesforce.com, 2009.

저 자 약 력

이 지 현

이메일 : jihyun@etri.re.kr

- 1998년 성균관대학교 정보공학과(학사)
- 2000년 포항공과대학교 컴퓨터공학과(석사)
- 2001년~현재 한국전자통신연구원 선임연구원
- 관심분야 : 서비스 기반 소프트웨어, 소프트웨어 아키텍처 상충 관계 분석 등

원 희 선

이메일 : hswon@etri.re.kr

- 1990년 연세대학교 전자공학과(학사)
- 1992년 KAIST 전산학과(석사)
- 1992년~1999년 한국방송 기술연구소 연구원
- 2000년~현재 한국전자통신연구원 선임연구원
- 관심분야 : 온라인 SW 플랫폼, 데이터 마이닝

허 성 진

이메일 : sjheo@etri.re.kr

- 1990년 경북대학교 전자공학과(학사)
- 1992년 경북대학교 컴퓨터공학과(석사)
- 1999년 경북대학교 컴퓨터공학과(박사)
- 1999년~2001년 창신대학 인터넷정보과 전임강사
- 2001년~현재 한국전자통신연구원 책임연구원
- 2009년~현재 한국전자통신연구원 SW서비스연구팀장

최 완

이메일 : wchoi@etri.re.kr

- 1981년 경북대학교 전자공학과(학사)
- 1983년 KAIST 전산학과(석사)
- 1988년 전자계산기조직응용 기술사
- 1985년~현재 한국전자통신연구원 책임연구원
- 2011년~현재 한국전자통신연구원 클라우드컴퓨팅연구부장