# 유비쿼터스 융·복합 인프라로서의 서비스 지향 기술 및 전망

건국대학교 민덕기* · 이용환**

## 1. 서 론

소프트웨어 기술 간의 상호운용성 문제와 서비스들 간의 융합 문제를 지원하기 위해 데이터 전송 기술의 표준화와 함께 다양한 이기종의 애플리케이션들에 대한 서비스화 기술 즉, SOA(Service-Oriented Architecture)관련 기술들의 등장으로 인해 소프트웨어 기술 시장은 점차적으로 소프트웨어 시장의 재편을 예고하고 있으며 기존 소프트웨어 시장의 침체된 상황을 정리하고 신규 시장의 확대와 시장의 활성화를 유도할 것으로 예상된다[1-4].

SOA는 1996년 Gartner 그룹(Schulte & Natis, 1996)에 의해 소개된 이래로 특별한 주목을 받지 못하다가 SOA 개념을 실제로 구현하는 웹 서비스가 등장하면서, 근래에 다시 중요한 패러다임이자 소프트웨어 아키텍처로 주목받고 있다. 그런데, 웹 서비스가 SOA를 실현시킨 유일한 기술은 아니다. 분산 컴퓨팅이나 RPC 기반 미들웨어 시스템들(예: CORBA, DCOM, RMI 등)도 SOA 개념을 구현한 기술들이다. CORBA, DCOM, COM+, EJB, J2EE, 그리고 .NET 등을 기반으로 하는 분산 컴포넌트는 특정 기술에 종속적이면서 상대적으로 연관도가 높아(Tightly Coupled) 동일한 플랫폼이나 프레임워크 내부에서의 재사용성은 뛰어나다. 그러나 이기종 환경에서 개발된 기존 컴포넌트들을 재사용하고자 할 때는 플랫폼과 구현 언어에 의존성이 강하여 상호운용의 문제가 발생한다는 단점이 있다. SOA는 이를 보완하기 위한 소프트웨어 아키텍처 중의 하나로서 구현 기술로부터의 독립성 및 유연성이 높아 재사용성 뿐만 아니라 이기종 시스템간 통합 구축에도 적합하다. 즉, SOA는 표준화된 인터페이스와 함께 XML 기반의 인터넷 표준 메시징 프로토콜을 이용한 약결합(Loosely Coupled)연결 방식을 제공함으로써 플랫폼에 보다 유연하면서 표준화된 아키텍처와 개발 방법을 제공한다[2,5,6]. 이런 장점들 때문에 최근에는 엔터프라이즈 서비스뿐만 아니라 그리드 서비스, 모바일 서비스, 디바이스 서비스, 유비쿼터스 서비스 등 다양한 산업 및 기술에도 XML 기반의 서비스화 경향이 두드러지고 있다.

이 글에서는 융·복합 인프라로서의 SOA가 무엇이고 어떤 특징을 가지고 있는지를 설명한다. 또한, SOA의 역할과 요건에 대해 기술하고 마지막으로 최근 서비스 지향 기술의 발전 전망에 대해 기술한다. 본 글의 구성은 다음과 같다. 2장에서는 융·복합 인프라로서의 서비스 지향 기술의 특징, 역할, 그리고 요건에 대해 기술하고 3장에서는 서비스 지향기술 발전 전망에 대해 기술한다. 마지막으로 4장에서는 결론을 기술한다.

## 2. 융·복합 인프라로서의 서비스 지향 기술

### 2.1 융통합 인프라의 역할과 요건

최근 정부에서는 서비스, 인프라, 신 성장 동력을 유기적으로 연계하고, IT 산업의 선순환 구조를 확립하여 산업전체의 동반성장을 이루어 내는 것을 목적으로 IT 839 전략을 내놓았다[3]. 이 전략은 새로운 IT 서비스를 도입하여 인프라에 대한 투자를 유발하고, 이를 바탕으로 홈 네트워크, 텔레매틱스 등 신 성장 동력 산업이 동반 성장할 수 있도록 하는 IT산업 기반을 확립하자는 것이 주요 내용이다[2]. IT839 전략의 발전모델은 3단계로 구성되어 있으며 현재 IT839는 아직 1단계 수준에 머물고 있지만, 각 영역별 기술이 어느 정도의 성숙도를 갖추어 감에 따라 점차 무게중심이 2단계 발전 모델인 상호운용성 이슈로 넘어가고 있으며 이와 더불어 최근에는 사용자의 관점에서 필요한 새로운 서비스 창출을 위하여 연계 통합 시범 사업의 필요성이 점차 커지고 있다[2]. 통합이나 상호 호환성 차원에서 중요시 되고 있는 서비스 지향 융·복합 인프라의 역할을 정의해 보면 다음과 같다.

* 종신회원
** 정 회 원

### 2.1.1 표준 분산 컴퓨팅

서비스 지향 융·복합 인프라는 기본적으로 원격에 존재하는 소프트웨어들 사이에 서로 상호작용하게 하는 표준 분산 컴퓨팅 환경을 지원해야 하다. 표준 분산 컴퓨팅을 지원하기 위해서는 표준서비스 정의체계, 표준인터페이스, 그리고 표준통신프로토콜이 필요하다. 표준서비스 정의체계의 경우 분산 컴퓨팅 환경 상에서 서비스 제공자의 서비스 정보를 기술하는 것을 표준화함으로써 서비스에 대한 인식을 보다 원활히 할 수 있다. 표준인터페이스는 접근의 편의성을 높일 수 있으며, 다양한 확장 기능을 지원할 수 있다. 표준통신 프로토콜은 전송 메시지 포맷의 표준화를 통해 정보 교환의 호한성과 효율성을 높일 수 있다.

### 2.1.2 이음새 없는 디바이스와 서비스의 융. 통합

서비스 지향 융·복합 인프라는 이음새 없는 디바이스와 서비스의 융·복합을 제공해야 한다. 즉, 디바이스 및 서비스들이 사용하는 서로 다른 네트워크 프로토콜들을 지원함으로써 이들이 제공하는 각종 정보 서비스들을 이해하고 상호 간의 원활한 통신 및 정보 교환을 지원할 수 있어야 한다.

### 2.1.3 표준 미들웨어 서비스 제공

서비스 지향 융·복합 인프라는 표준 미들웨어 서비스를 제공할 수 있어야 한다. 서비스 지향 융·복합 인프라는 응용 도메인별로 공통적으로 요구되는 미들웨어 서비스들(예를 들면, 보안, 트랜잭션, 디렉터리, 프로세스 관리서비스)을 지원해야 한다. 미들웨어 서비스들은 크게 로컬 상에서 지원되는 서비스들과 원격에서 지원되는 서비스들로 나누어볼 수 있다. 로컬 상에서 지원되는 미들웨어 서비스는 하나의 시스템 내에서 제공되는 표준 서비스들을 대상으로 보안, 트랜잭션 등과 같은 서비스들을 제공해야 하며, 원격에서 지원되는 미들웨어 서비스의 경우에는 네트워크상의 모든 표준 서비스들을 대상으로 서비스 센터 등과 같은 전문적인 관리 체계를 통해 디렉터리, 상호호환성, 품질관리 등의 서비스들을 제공해야 한다.

### 2.1.4 자동화된 서비스 연합

서비스 지향 융·복합 인프라는 자동화된 서비스 연합(Automatic Service Integration)이다. 즉, 서비스 지향 융·복합 인프라는 검색된 여러 응용 서비스(Domain Specific Service) 중에서 최적의 서비스를 동적으로 결합하여 사용하도록 자동화 된 서비스 통합 인프라를 제공한다. 자동화 서비스 인프라는 크게 동적 서비스 정의 및 설정, 융합 등으로 구성되며 이중 동적 서비스 정의 및 설정의 경우 기존 서비스 정보를 기반으로 표준화된 서비스 정보를 산출한 뒤 이를 동적으로 재구성하여 표준 분산 컴퓨팅 환경에서 운용할 수 있도록 하는 환경을 제공한다. 이밖에 융합 서비스 지원은 기존 서비스들의 의미론적 정보 결합을 통해 여러 서비스들이 결합된 신규 서비스로 제공할 수 있도록 한다.

서비스 지향 융·복합 인프라는 네트워크상에 존재하는 다양한 디바이스와 서비스를 융·통합하는 소프트웨어 기반 구조로서 그림 1과 같이 크게 서비스 융·복합 인프라, 표준 분산컴퓨팅 인프라, 그리고 서비스관리 인프라의 세 가지 요건을 갖추어야 한다. 이 세가지 요건을 정리해 보면, 먼저 서비스 융·복합 인프라는 서로 다른 네트워크 프로토콜들을 사용하는 디바이스나 서비스들을 인지하고 융·복합하여 하나의 표준화된 서비스 구조로 제공하는 인프라이다. 이는

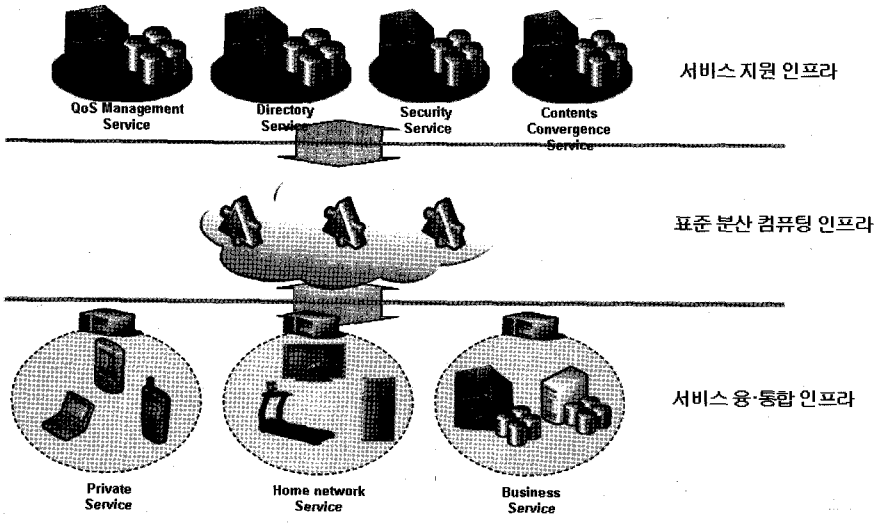


그림 1 서비스 지향 융·복합 인프라의 3가지 요건

융·복합을 지원하기 위해 각각이 사용하는 네트워크 프로토콜의 지원, 표준 인터페이스 제공, 콘텐츠 양식의 자동변환 등을 통해 서비스들 간의 정보 교환 및 제어가 가능해 지도록 하는 기능을 포함한다. 두 번째는 표준분산컴퓨팅 인프라는 디바이스와 서비스 들 간에 투명성을 지원하기 위해 표준 기반의 분산 컴퓨팅 요소를 제공하는 환경을 말한다. 이는 서비스의 정의에서부터 검색, 호출에 이르기까지의 모든 분산 컴퓨팅 과정을 하나의 표준 기술로 적용함으로써 이기종 플랫폼들 간의 상호 호환성을 보장할 수 있는 환경을 지원한다. 세 번째로, 서비스관리 인프라는 표준 분산 컴퓨팅 인프라 상에서 운용되는 서비스들을 효율적으로 관리하기 위한 기능들과 이들 간에 공통적으로 요구되는 미들웨어 서비스들을 지원하는 환경을 의미한다.

### 2.2 융·복합 인프라 후보기술로서 SOA의 특징

시스템 통합이나 호환성은 이기종 분산 시스템이나 애플리케이션간의 통합을 의미하는 것으로서 기업 환경의 변화에 따라 비즈니스 프로세스를 수시로 융통성 있고 민첩성 있게 변화 시킬 수 있는 정보 시스템 구축 능력을 말한다. 이러한 시스템 통합이나 호환성을 가능케 가능케하는 융·복합 인프라의 후보기술로서 TCP/IP기반, RPC 기반, ORB기반, 그리고 XML기반이 있다. TCP/IP는 분산 컴퓨팅 환경 구축의 가장 근간이 되는 기술로 단순한 구조와 빠른 속도를 보장하는 장점을 가진다. 하지만 프로토콜 구조의 특성상 각 어플리케이션별로 프로토콜을 확장해야하며 표준화된 인터페이스가 없다. 따라서 서비스 기반의 분산 컴퓨팅 환경을 제공하기에는 많은 비용과 어려움이 따른다. RPC 기반의 분산 컴퓨팅 환경은 처음으로 서비스 기반의 분산 컴퓨팅개념을 도입한 기술 환경으로 단순히 TCP/IP기반의 네트워크 모델위에 서비스를 요청하는 프로토콜을 제공한다. RPC 기반의 분산 컴퓨팅 기술을 객체지향 환경에 적합하게 변형시킨 것이 ORB 기술이다. ORB 기반의 분산 컴퓨팅 환경은 서비스 정의, 전용 프로토콜 지원 그리고 미들웨어 서비스 등을 지원한다. ORB 기반의 기술로는 CORBA, DCOM, COM+, EJB, J2EE, 그리고 .NET 등과 같은 기술들이 사용될 있다. 이들 ORB 기술들은 특정 기술에 종속적이다. 하지만, 상대적으로 연관도가 높아(Tightly-Coupled) 동일한 플랫폼이나 프레임워크 내부에서의 통합이나 재사용성은 높다. 그러나 이기종 환경에서 개발된 기존 컴포넌트들을 재사용 및 연계 통합하고자 할 때는 플랫폼과 구현 언어에 의존성이 강하여 상호운영의 문제가 발생한다는 단점을 가지고 있다. 이를 보완하기 위한 나온 기술이 XML 기반의 분산 컴퓨팅 기술인 SOA 기술이다. XML 기반의 SOA기술은 XML기술이 가지는 유연한 확장성과 구조화된 데이터 정의 기능의 장점을 통해 서비스의 정의, 전송, 확장, 관리 등을 용이하다. 따라서 구현 기술로부터의 독립성 및 유연성이 높아 재사용성 뿐만 아니라 이기종 시스템 간 통합 구축에도 적합하다. XML 기반의 분산 컴퓨팅 환경은 하지만 성능적인 측면에 있어 다른 여타 분산 컴퓨팅 기술에 비해 많이 뒤처진다는 단점을 가진다.

표 1은 후보기술을 융·복합 인프라가 가져야한 요건별 비교 항목에 대하여 비교한 것이다. 비교 분석 및 평가를 볼 때에 서비스 지향 융·복합 인프라를 구축하기에 가장 용이한 기술은 XML기반의 SOA 분산 컴퓨팅 환경이라 할 수 있다. 특히 플랫폼 독립적인 서비스 제공과 확장의 용이성은 네트워크상의 다양한 서비스들을 융·복합하려는 서비스 지향 융·복합 인프라의 목적에 가장 잘 부합되는 기술로 고려될 수 있다.

XML 기반의 SOA 분산 컴퓨팅 기술이 융·복합 인프라서의 충분한 가능성을 가졌는지 SOA의 특성을 살펴보자. SOA는 서비스들을 기반으로 한 소프트웨어 아키텍처로서 '애플리케이션의 기능(Function)들을

표 1 융·통합 인프라 후보기술의 비교분석

| 분류 | 비교항목 | TCP/IP기반 | RPC 기반 | ORB 기반 | XML 기반 |
|---|---|---|---|---|---|
| 분산 컴퓨팅 인프라 | 정의언어 존재여부 | x | x | o | o |
| | 정의언어의 확장용이성 | x | x | x | o |
| | 서비스개념 존재여부 | x | o | o | o |
| | 플랫폼 독립적 서비스 개념정의부와 플랫폼 의존적 서비스 바인딩부의 분리여부 | x | x | x | o |
| | 통신 프로토콜 정의방법 | o | o | o | o |
| | 분산 컴퓨팅을 위한 표준 프로토콜 존재여부 | o | x | o | o |
| | 미들웨어 서비스를 위한 분산 표준 프로토콜의 확장가능성 및 용이성 | x | x | o | o |
| | 다양한 통신 프로토콜과의 바인딩여부 | o | x | x | o |
| | 안전한 전송여부 | o | o | o | o |
| 서비스 관리 인프라 | 디렉토리 서비스 | x | x | o | o |
| | 상호발견 서비스 | x | x | o | o |
| | 트랜젝션 서비스 | x | x | o | o |
| | 비즈니스 프로세스 서비스 | x | x | o | o |
| | 복합 서비스 | x | x | x | o |
| | 보안 서비스 | x | x | o | o |
| 융·복합 인프라 | 관리 서비스 | x | x | o | o |
| | 개방형 표준지원 | x | o | o | o |
| | 플랫폼 독립성 | x | x | x | o |
| | 성능 | o | o | x | x |
| | Firewall 통과여부 | x | x | x | o |

사용자에 적합한 크기로 공개한 서비스들의 집합으로 이의 제공, 사용에 관한 정책이나 적용 또는 프레임워크'로 정의할 수 있다[7,8,9]. SOA 기술을 구현한 구현기술로는 웹 서비스(Web Services)가 대표적이다[10]. 이러한 융통합 인프라로서의 SOA 기술의 특징은 다음과 같다.

#### 2.2.1 표준 서비스 인프라(Standard Service Infra)

SOA는 표준 서비스 인프라를 제공한다. 즉, 제공하는 서비스에 대한 인터페이스를 XML 기반의 표준 형태(예, 웹 서비스의 경우 WSDL)로 기술 할 수 있으며, 메시지도 XML 기반의 표준 메시지 포맷(예, 웹 서비스의 경우 SOAP)을 사용하여 통신하게 된다. 제공 서비스가 인터넷과 같은 글로벌 네트워크상에 어디든 존재 할 수 있다. 서비스 사용자는 표준화된 서비스 발견 방식(예, 웹 서비스의 경우 중앙 집중 방식의 UDDI나 분산방식의 WS-Discovery)에 따라서 어디서든지 원하는 서비스를 찾아 사용이 가능하다.

#### 2.2.2 약결합(Loosely-Coupled) 형태의 서비스 연결

SOA는 약결합 형태의 서비스 연결을 지원한다. 서비스 간 결합도가 높을수록 한 모듈의 변화가 다른 모듈에도 영향을 주어 파문효과(Ripple Effect)를 일으키게 되는데, 파문 효과가 클수록 시스템을 유지보수하기가 어려워진다. 약결합 형태의 서비스 연결을 위해 SOA는 먼저 서비스 인터페이스와 서비스 내부설계 및 구현을 분리하는 Black Box 형태의 서비스를 지향한다. Black Box형태의 서비스에서 클라이언트는 서비스가 제공하는 인터페이스가 명시하는 규약들만을 이해하고 서비스 내부의 여러 규약들에 대해서는 이해할 필요가 없다. 두 번째로 약결합 형태의 서비스 연결을 위해서는 서비스 레지스터리(Registry)와 관련해 서비스 발견, 동적 바인딩을 지원함으로서 서비스 위치 투명성을 제공한다. 세 번째로 약결합 형태의 서비스 연결을 위해 SOA는 여러 인프라(예: 플랫폼, 운영체제, 그리고 구현 언어)에 독립성을 제공해야 하며 운영 중에 서비스 발견, 융합(Composition), 확인(Verification) 등을 제공한다.

#### 2.2.3 서비스 재사용성

그림 2와 같이 SOA는 서비스 재사용을 위하여 비즈니스 서비스모델과 적합한 정도로 큰 크기(Granularity)의 서비스를 제공한다. 서비스의 재사용성 문제는 이미 객체지향이나 컴포넌트 기반기술의 주요 이슈중의 하나이다. 서비스 요청자와 제공자가 동의한 한 개 이상의 인터페이스를 가지고 있다는 면에서 서비스는 클래스나 컴포넌트와 동일하다. 하지만, 서비스는 클래스나 컴포넌트와 같이 어떤 객체의 타입에 대한 어떤 것이 아니고 스키마(Schema)에 대한 것이며 또한, 메서드 호출에 대한 것이 아니라 메시지에 대해 것이다. 더불어, 서비스 크기(Granularity)는 객체나 컴포넌트 보다 일반적으로 크다. 이러한 서비스의 크기에 대한 결정은 서비스에 대한 중요한 설계상의 결정문제이다. 서비스 크기(Granularity)가 잘못된다면 그들이 필요한 것 이상의 더 많은 서비스 호출을 해야 하는 불상사가 생길 수 있으며 또한 서비스 레벨상의 보안 관련 문제가 생길 수 있다. 그리고 잘못 사용된 경우 어떤 메서드로부터 특정 사용자를 제한하는 것이 불가능할 수 도 있다. 하지만 이러한 서비스 크기(Granularity)를 결정할 때 중요한 문제는 서비스의 유지 보수를 해치지 말아야 한다는 것이다.

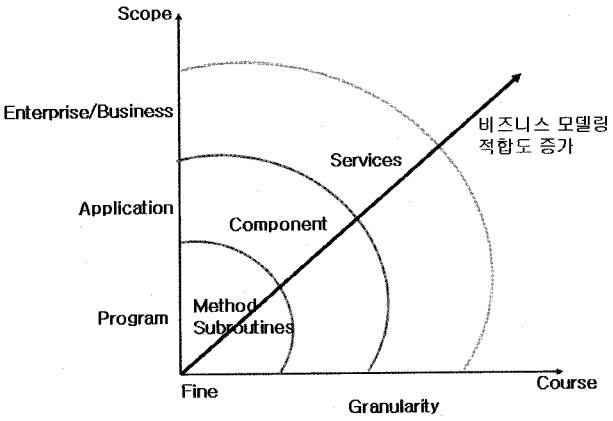


그림 2 범위(Scope)와 크기(Granularity) 관점에서 서비스 재사용성

#### 2.2.4 공통 표준 서비스들

SOA는 여러 서비스들에서 공통적으로 사용할 수 있는 표준 서비스들을 제공한다. 공통 표준 서비스들에는 각 비즈니스 도메인별 공통 서비스도 존재하지만 융·복합을 위해 필요한 기술적인 표준 서비스들도 포함한다. 이들 기술적인 표준 서비스들에는 여러 서비스들이 필요하지만 최근에 가장 이슈가 되고 있는 대표적인 것으로는 신뢰성 있는 메시지 서비스와 트랜잭션 서비스, 안전한 보안 서비스, 그리고 동적인 비즈니스 프로세스 실행 서비스(예, 웹 서비스의 경우 WSBPEL)와 같은 것들이 있다. SOA가 가지는 재사용성이나 품질속성 측면에서 고려할 때 공통 표준 서비스들은 중요한 의미를 가지고 있으며 서비스 지향 기술의 발전과 성숙을 위해 중요한 기술들이다.

그림 3은 SOA가 가져야 할 이러한 특징을 반영한 서비스 기반 아키텍처의 모습이다. 즉, SOA는 표준

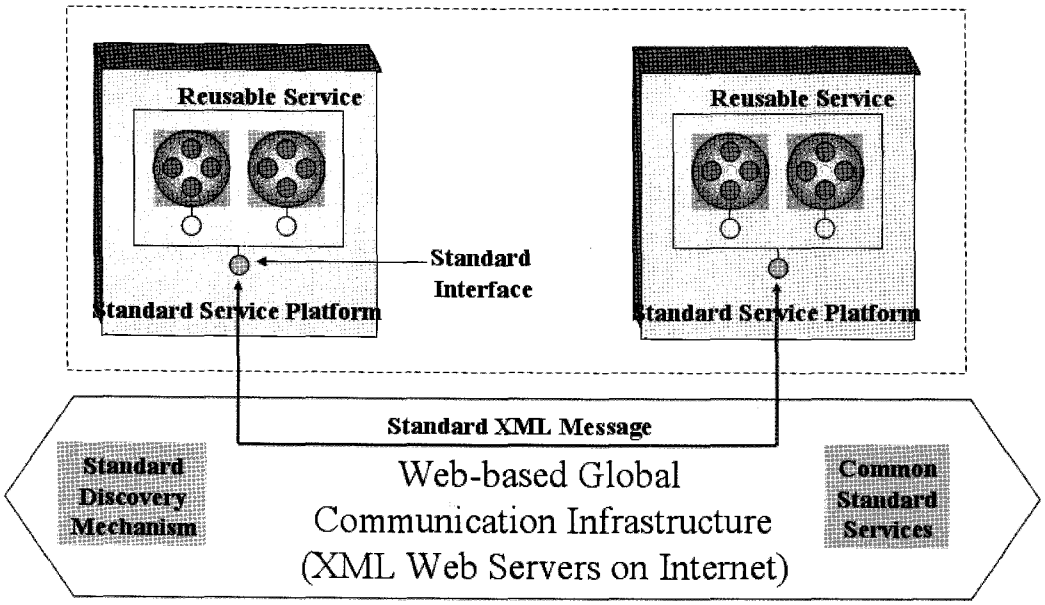


그림 3 Service-Oriented Architecture의 구조

서비스 플랫폼, 표준 발견 메커니즘, 표준 XML 메시지, 그리고 공통 표준 서비스들이 존재한다. 표준 서비스 플랫폼 안에는 서비스에 관련된 표준 인터페이스와 구현이 존재한다. 서비스들은 인터페이스 표준명세를 사용해 기술되며 서비스 크기(Granularity)에 의해 재사용성이나 유지보수성에 많은 영향을 미치게 된다. 일반적으로 객체나 컴포넌트 보다 인터페이스의 크기가 일반적으로 크며 서비스는 한 개 이상의 객체나 컴포넌트 형태로 구현된다. 두 번째로 SOA는 서비스들의 위치에 관계없이 서비스들을 발견하고 호출하기 위해서 웹 기반의 글로벌 통신 인프라를 제공해야 한다. 웹 기반의 통신 인프라는 표준 발견 메커니즘, 표준 XML기반 메시지 포맷, 공통 표준 서비스들을 포함하고 있다. 또한, 정의된 표준 XML 기반 표준 메시지 포맷을 기반으로 서비스들에 대한 접근을 용이하게 하기 위해서는 웹 기반으로 해야 한다. 그 이유는 기존의 많은 방화벽들은 외부에 서비스를 제공하는 웹 서버에 대해서 기본적으로 접근을 허용하기 때문에 방화벽의 접근 제약성을 어느 정도 극복할 수 있기 때문이다.

## 3. 서비스 지향 기술의 발전 전망

융·복합 인프라로서의 서비스 지향 기술의 지능화, 이벤트화, 버스와, 임베디드화 등의 방향으로 발전하고 있으며, 모발산업 및 그리드 산업의 경우 이러한 방향으로 변화를 시도하고 있다.

### 3.1 서비스 지능화 (Intelligent Services)

SOA의 구현 기술 중의 하나인 웹서비스는 웹에서 정보를 표현할 수 있는 정적인(Static) 기능에 서비스라는 동적인(Dynamic) 기능을 추가한 것이다. 많은 사람들이 웹서비스를 통해 분산 환경의 컴포넌트를 통합할 수 있는 가능성을 언급하고 있고, 실제로 웹 환경은 이를 바탕으로 급속히 변화하고 있다. 그럼에도 불구하고, 현재의 웹서비스는 구문적(Syntactical) 관점에서 서비스를 기술해야 하는 제약을 갖고 있다. 이러한 특성은 사람의 개입 없이 서비스 환경을 변경 혹은 확장하는 것을 쉽지 않게 하여 웹서비스의 확산을 가로막는 요인으로 지적되고 있다. 이러한 문제는 지식관리(Knowledge Management), EAI(Enterprise Application Integration) 및 전자상거래 분야에 걸쳐 폭넓게 자리 잡고 있다. W3C에서는 다양한 애플리케이션 사이의 의미적 수준의 상호운용, 검색, 자동화를 위해 서비스의 지능화를 가능케 하는 시맨틱 웹을 제안하고 있다. 시맨틱 웹 환경은 분산되고 이종적인 정보를 지능적인 방법으로 접근 가능하게 하고, 사용자의 요구와 이용 가능한 정보 자원 사이에서 소프트웨어를 이용한 중개(Mediation)를 제공한다. 따라서 웹서비스의 잠재적 능력을 활용하기 위해서는 시맨틱 웹 기술과의 접목이 요구된다. 시맨틱 웹 기술은 사용자의 요구 조건에 맞는 서비스의 탐색과 재구성을 위한 자동화를 제공한다. 따라서 차세대 웹서비스 기술은 웹서비스 기술과 시맨틱 웹 기술이 접목된 시맨틱 웹서비스, 즉 지능형 웹 서비스 방향으로 나아갈 것이다. 그림 4은 이러한 지능형 웹 서비스 방향을 나타내고 있다.

현재의 웹 서비스는 UDDI와 WSDL을 기반으로 하고 있는데 이들 기술은 의미 정보를 표현하지 못하고

있다. 즉, 전송 바인딩, 교환 메시지 정의와 커뮤니케이션 프로세스 정의와 같은 요소를 포함하고 있지만 자동적인 서비스 인지(Recognition), 비교(Comparison),

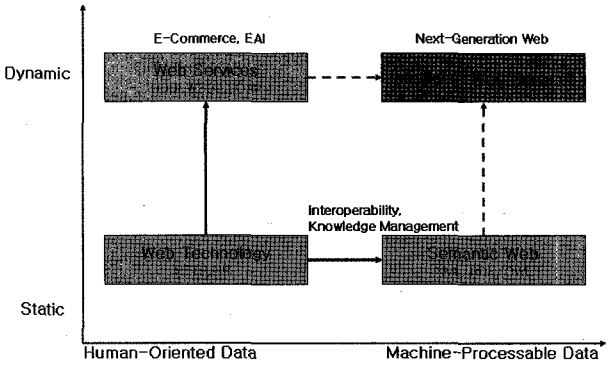


그림 4 웹기술의 발전방향

구성(Configuration)과 같은 기능은 제한적으로 제공하고 있다. 서비스에 대한 충분한 명세를 지원하지 않는 환경에서 서비스 제공자나 요청자는 요구 사항에 맞는 정보를 탐색하는 것이 어려울 뿐만 아니라 중개 시스템을 통한 서비스도 만족할 수준을 얻기 힘들다. 시맨틱 웹 서비스에서 온톨로지는 서비스 명세를 위한 핵심적인 역할을 한다. OWL 온톨로지 기반의 서비스 언어인 OWL-S는 웹 서비스의 특성과 수용력을 기술하고 있다. OWL-S는 서비스 특성에 대한 추론과 에이전트가 처리할 수 있는 의미 수준의 마크업 언어를 제공한다.

### 3.2 서비스 이벤트화(Event-Based Services)

유비쿼터스 서비스는 상황에 맞는 적시에 찾아가는 서비스를 제공하는 것으로서 상황에 맞는 서비스 혹은 상황을 예측하여 능동적인 서비스를 제공해야 한다. 이를 위해 유비쿼터스는 개인의 주변에서 발생하는 다양한 정보, 즉 위치정보, 환경정보 등과 사용자의 행동 패턴과 같은 이력정보, 더불어 개인과 직접적인 관계는 없으나 개인에게 영향을 미칠 수 있는 이벤트 등 실세계에서 발생하는 다양한 이벤트 정보를 종합적으로 분석하여 서비스를 연동하고 있다. 또한, 비즈니스 환경 변화에 유연하고 민첩한 대응을 위해 기업은 기업 내. 외부적인 프로세스의 지연요소를 제거하고, 비즈니스 과정상에서 발생하는 이벤트와 비즈니스에 영향을 줄 수 있는 외부 이벤트를 조기에 감지와 신속하게 대응하기 위한 IT 구조를 요구하고 있다. 또한, 실시간(RTE) 구현을 통해 기업 내부 및 외부에서 발생하는 중요 이벤트들에 조직이 신속히 대응하기 위해 최근 기업들은 이벤트 중심으로 비즈니스 프로세스를 구성하고 있다.

융·복합 인프라로서의 SOA는 다양한 이벤트를 종합적으로 분석해서 실시간적으로 서비스를 제공하기 위해 EDA(Event-Driven Architecture)를 포함하여 SOA 2.0이라고 명명하고 있다. 정부조사 기관에 의하면 현재는 대기업과 중소기업의 10% 미만이 SOA와 EDA 기반으로 응용을 구축하고 있고, 패키지 응용에서도 5% 미만에서 적용되고 있지만 2009년도에는 새로운 패키지 응용과 비즈니스 응용의 25% 이상이 SOA와 EDA기반으로 구축될 것으로 전망하고 있다[11,12]. SOA와 EDA는 상호보완적인 서비스 구축 방법으로 SOA가 서비스 인터페이스를 이용하여 요청(Request) 및 응답(Response)을 통해 서비스를 연동하는 것이라면, EDA는 이벤트에 대한 감지(Sense) 및 대응(Respond) 모델이다. 표 2는 SOA와 EDA와의 비교이다. SOA는 클라이언트에 의해 서비스가 제어되며 순차적으로 실행되는 반면, EDA는 이벤트 수신자가 대응여부를 결정하며, 이벤트가 동시에 여러 곳으로 전달이 가능하고 또한 비동기 방식으로 전달이 가능하므로 이벤트 발생에 의한 대응이 동적으로 구성된다.

표 2 SOA와 EDA 비교

| 구분 | SOA | EDA |
|---|---|---|
| 상호규약 | 서비스 인터페이스 정보 | 이벤트 규격 정보 |
| 연결방식 | 1 : 1 | N : N |
| 흐름제어 주체 | 클라이언트 | 이벤트 수신자 |
| 흐름제어 방식 | 순차경로 | 동적/병렬/비동기 방식 |
| 새로운 입력에 대한 대응 | 진행 중에는 새로운 입력을 차단 | 진행 중에서 새로운 입력에 반응 |

〈자료〉: Gartner, 2005.

### 3.3 서비스 버스화(Enterprise Service Bus)

다양한 기술로 구현되어 있는 기업 인프라를 SOA 기반으로 통합하기위한 솔루션으로 Enterprise Service Bus(ESB)가 관심을 끌고 있다. Enterprise Service Bus(ESB)는 통합이라는 관점에서의 프로젝트를 진행시에 비용을 낮추기 위한 새로운 아키텍처이다. 또, 통합 프로젝트의 기술과 경제성을 근본적으로 변화시키기 위하여 SOA의 원리와 웹서비스가 가진 잠재 능력을 이용한다. ESB의 가장 큰 특징과 Paradigm의 전환점을 들라면 역시 Service 기반의 아키텍처와 기존의 제품군 위주의 아키텍처에서 이제는 전사적 Business Integration이라는 (BI) 하나의 큰 개념의 대두라 하겠다. 즉, 기존의 모든 역량은 이제 기업의

Business 통합이라는 대 주제 안에 묶이게 되고, 그것을 가능케 하는 기반 기술은 Service 기반인 Enterprise Service Bus에 의해 연계된다.

ESB는 비즈니스 내에서 서비스, 어플리케이션, 자원을 연결/ 통일하는 미들웨어라고 할 수 있다. 또한, 사내외적으로 제반 비즈니스 어플리케이션의 기능을 타 어플리케이션에 의해 재사용하고자 하는 데 기반이 되는 프레임워크이기도 하다[13]. 기존의 통합에 대한 새로운 방법론이며, 서로 다른 언어와 프로그래밍 모델에서 개발된 소프트웨어의 통합과 연결을 가능케 하여 준다. 여기에는 물론 웹서비스와 메시징기반의 전송과 라우팅도 포함된다. ESB의 상위 계층에는 User Interaction Service, Application Service, Information Service, Process Service, Community Integration Service등의 다섯 서비스 계층이 블록화 되어 있고, 어떠한 형태의 통합도 가능케 하는 미들웨어 역할을 해준다. ESB의 Universal Connectivity Layer를 갖고 있어, 기업 간 Integration의 스케일과 Scope를 더욱 확장한다.

### 3.4 서비스 유틸리티화(Services on Demand)

서비스 유틸리티화(Utility Computing)란, IT산업 역시 수도나 전기처럼 기간이 되는 산업으로 발전할 것이라는 점에 착안하여, 수도나 전기를 사용하고 사용료를 내는 것처럼 컴퓨터의 유틸리티를 사용한 만큼(On Demand) 요금을 지불하는 방식을 말한다. 하드웨어 종량제 판매라고도 하며, 2001년 미국의 Hewlett Packard가 고성능 유닉스 서버 슈퍼 돔을 출시하며 도입한 이후, IBM과 SUN 등 다른 대형 컴퓨터 업체에서도 적극적으로 개발에 나서고 있다.

하드웨어의 경우, 대용량의 서버나 스토리지를 제공하되, 사용량을 늘려갈 때마다 단계적으로 공급회사에서 키를 제공하는 방식이다. 소프트웨어도 ASP(Application Service Provider)의 형태로 인터넷을 통해 고객이 필요한 솔루션을 빌려서 사용하는 방식이다. IBM은 차세대 컴퓨터 전략으로 e비즈니스 온 디멘드(e-Business On Demand)를 내세우고 있다. 최근에는 중견 및 중소기업의 IT환경에 적합하도록 설계된 e-비즈니스 온 디맨드 방식의 업무 솔루션 패키지인 VNS-ES1과 VNS-IB를 선보였다. 이러한 솔루션 패키지는 고객에게 ERP, 전자구매, 인터넷 뱅킹 등 종합적이고 일관된 서비스를 제공하는 서비스이다. 이 솔루션은 벨류넷 서비스(Value-Net Service) 개념에 기반을 둔 것으로, IBM의 파트너 사의 솔루션과 긴밀하게 연결함으로써, 파트너 사와의 가치 네트워크(Value-Network)를 지속적으로 확장하여 향후 요구사항에도 적절한 대응이 신속하게 이루어질 수 있도록 했다. SOA는 이러한 온 디멘드(On-Demand)방식의 서비스 사용을 가능케 하며 네트워크 통신의 속도가 빨라질수록 미래의 소프트웨어는 더욱 유틸리티화의 방향으로 나아갈 것이다.

### 3.5 서비스 임베디드화(Service Embedded)

현재 서비스의 범위는 기존에 기업의 비즈니스 서비스에서 디바이스의 서비스로 그 범위가 확장되어 가고 있다. 특히 디바이스 성능의 향상은 기능적 확장으로 이어지면서 서비스의 임베디드화를 촉진시키고 있다[14]. DP4WS[15]는 이러한 서비스의 임베디드화의 대표적인 기술적 사례로 디바이스와 서비스의 개념을 논리적으로 동일한 영역으로 추상화 하고 있다. DP4WS(Device Profile for Web Services)는 웹서비스의 확장 스펙을 이용해 디바이스의 서비스 운용환경을 정의하는 기술로서 기존의 UPnP 1.0 스펙을 참조하여 웹서비스 기반의 UPnP 모델을 구현하고 있으며, 향후 UPnP 2.0 모델로 정의될 예정이다.

DP4WS는 크게 디바이스 웹서비스의 정의, 발견, 제어 및 이벤트 처리 등으로 구성되며, 암호화된 메세지 정보를 주고받는다. 디바이스 웹서비스 정의의 경우 WSDL를 사용해 정의되며, WS-Discovery를 통해 디바이스의 웹서비스를 검색하게 된다. 또한 검색된 디바이스의 웹서비스는 WS-Metadata스펙을 통해 서비스의 메타데이터 정보를 주고받으며 SOAP를 통해 제어된다. WS-Eventing스펙은 디바이스의 이벤트 메커니즘을 구현하는데 이용되며 SOAP의 MTOM(Message Transmission Optimization Mechanism) 스펙을 통해 고용량의 데이터를 전송하기도 한다. 그밖에 디바이스간의 통신 보안과 인증처리를 위해 AES/TLS, HTTP Authentication, SHA1, UUID, X.509.v3 등의 보안 메커니즘이 사용된다. 그림 5는 DP4WS의 구조를 보여주고 있다.

DP4WS와 더불어 최근에는 센서나 구동장치와 같은 다양한 장치들을 분산 IT 엔터프라이즈 시스템들과 통합하려는 서비스 기반 디바이스 아키텍처(Service-Oriented Device Architecture)[16] 관련 기술들이나 표준들이 등장하고 있다. 그림 6과 같이 SODA는 모델러 혹은 프로그래머들이 센서나 구동장치와 같은 장치들을 우리가 알고 있는 엔터프라이즈 SOA의 비즈니스 서비스처럼 모델링하고 구현할 수 있게 하는 것을 목표로 하고 있다[16]. 이를 위해 SODA는 물리적인 영역과 디지털 영역 사이의 경계 영역에 집중하고

있다. SODA가 집중하는 경계영역은 예를 들면, RFID 태그 리더의 경우 물리적인 현상을 상응하는 디지털 데이터로 변환하는 영역이며 반대로 구동장치의 경우에는 디지털 신호나 데이터를 물리적인 현상으로 변환하는 영역을 말한다.

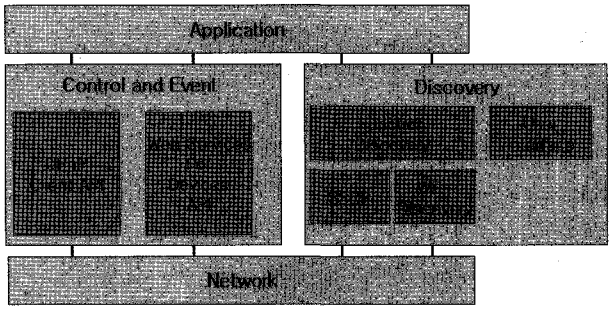


그림 5 DP4WS의 구조

SODA는 이러한 목표를 달성하기 위해 SODA의 구현 내부에 Device Adaptor, Bus Adaptor, Device Service Registry와 같은 3개의 핵심 컴포넌트들로 구성되어 있다. 먼저 Device Adaptor는 장치와의 연결, 프로토콜, 장치와의 인터페이스 등을 담당하며 다른 한편으로는 디바이스의 서비스 모델들을 추상화함으로서 프로그래머나 모델러들이 세부적인 장치에 대한 내용을 다루지 않고도 해당 서비스들을 사용할 수 있게 한다. 두 번째로 Bus Adaptor는 장치 서비스의 추상화된 모델을 엔터프라이즈에서 사용하는 특정 SOA 바인딩 메커니즘을 사용함으로서 네트워크 프로토콜을 통해 데이터를 이동 시킬 수 있게 한다. 이는 추상화된 서비스 관련 정보들을 엔터프라이즈에서 사용하는 다양한 프로토콜이나 데이터 포맷에 무관하게 데이터를 공유할 수 있게 한다. 마지막으로, Device Service Registry는 SODA 서비스들을 발견하고 접근할 수 있게 한다.

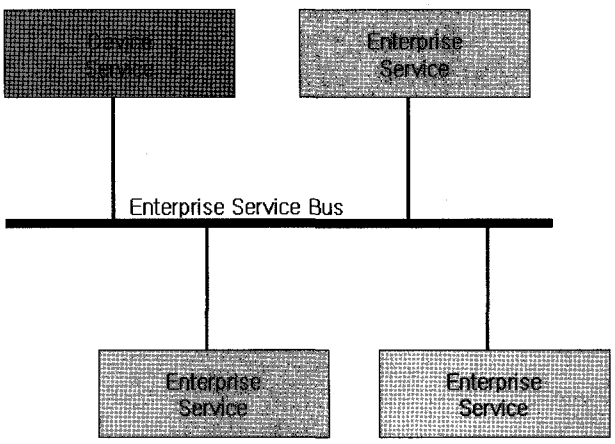


그림 6 장치서비스들을 SOA의 서비스처럼 모델링

### 3.6 모바일 서비스화(Mobile Services)

SOA의 등장과 함께 최근 모바일 환경에서는 모바일 디바이스의 개념을 서비스 지향 구조상에 하나의 서비스 구성 요소로 정의하고 모바일 디바이스 상에서 요구되는 서비스 구성을 플랫폼에 독립적이고 동적인 운용이 가능하도록 하고 있다. 노키아는 2003년도에 들어서 이를 지원하기 위한 차세대 노키아 모바일 소프트웨어 플랫폼을 개발하기 시작했다. 특히 디바이스와 리모트 시스템간의 메세징 전송 구조를 지원하기 위해 HTTP, XML, SOAP 등과 같은 오픈 표준 기술들로 구성된 노키아 디바이스 웹서비스 프레임워크(NDWF: Nokia's Device WebService Framework)를 제공하고 있다. 참고로, 노키아 디바이스 웹서비스 프레임워크는 웹서비스 엔진을 기반으로 하여 서비스 구동 서비스, 기본 서비스(서비스 발견, 디렉터리, 정책 서비스), 확장 서비스(과금, 위치추적, 메세징서비스 등) 등으로 구성되어 있다. 최근에 들어서는 OASIS와 Liberty Alliance등에서 제시하고 있는 웹서비스의 확장 스펙들(OASIS: WS-Security, Liberty Alliance: ID-FF/IDWSF)을 도입하여 운영 중인 서비스 상에 적용하고 있다.

OMA(Open Mobile Alliance)는 Mobile Web Service 워킹 그룹을 통해 OMA 서비스 환경(OSE: OMA Service Environment)에서의 웹서비스 지원 스펙을 제시하고 있다. OSE는 Enabler라는 인터페이스 구조를 이용해 필요로 하는 기능들을 서비스형태(OMA 서비스)로 제공하는 시스템 환경으로 모바일 디바이스 상에서 요구되는 서비스 구성을 플랫폼에 독립적이고 동적인 운용이 가능하도록 한다. Mobile Web Service는 OSE의 Enabler의 한 일종으로 OSE상에서 제공되는 OMA 서비스들을 웹서비스로 변환하는 기능과 OSE 상에 신규 웹서비스를 제공하는 기능을 지원한다.

### 3.7 그리드 서비스화(Grid Services)

그리드 쪽에서도 서비스 지향 기술을 도입하고 있다. 현재, 그리드 기술 표준화는 GGF가 추진하고 있는데 1999년 미국을 중심으로 시작한 그리드 포럼에 유럽 아시아의 많은 나라들이 참여하여 2000년 11월에 조직된 유일한 그리드 관련 국제 표준화 단체이다. GGF에는 7개 영역(Application, Programming Model and Environments, Architecture, Data, Security, Information Systems and Performance, Peer-to-Peer, Scheduling Resource Management)이 있고, 영역

마다 워킹그룹(Working Group: WG)과 리서치그룹(Research Group: RG)이 표준화 활동을 주도하고 있다. GGF는 표준화 활동의 진행과 소프트웨어 개발에 대한 개별적 프로젝트를 추진하고 있다. 글로버스 툴키트는 일종의 업계표준(defect standard)으로서 유명하다.

2002년 2월에 캐나다의 토론토에서 개최된 GGF4에서 글로버스 프로젝트와 IBM은 그리드 기반의 SOAP(Simple Object Access Protocol), WSDL(Web Service Description Language) 등의 웹서비스를 기반으로 그리드의 모든 기능을 서비스로 제공하는 새로운 구조 OGSA(Open Grid Services Architecture)를 제안하였다. 글로버스 툴키트는 미국 알콘즈 국립연구소와 남캘리포니아 대학을 중심으로 한 대학, 그리고 IBM, MS와 같은 기업들이 참가한 글로버스 프로젝트 팀에 의하여 개발되고 있다. 2002년 10월에 배포된 글로버스 툴키트 2.2가 최신판이다.

글로버스 프로젝트에서도 OGSA에 대응하는 미들웨어 글로버스 툴키트 3.0(GT3)의 개발을 진행하고 있으며, 2003년 1월에는 OGSI부분을 포함한 GT3 알파판을 공개하였다. 글로버스 이외에도 유럽의 UNICORE 프로젝트 등이 OGSA를 구현하고 있다. 그리드의 커뮤니티는 신속한 논의를 통하여 OGSA의 기본 부분인 OGSI(Open Grid Services Infrastructure)에 대한 스펙의 제1판을 2003년 3월에 GGF에 제출하였다. 이와 같은 과정에서 그리드 기술의 표준화를 검토하는 많은 WG가 설립되었다. 그 다음 버전인 GT4는 OGSI는 WSRF(Web Service Resource Framework)이라는 웹서비스 기반으로 대치될 예정이며 따라서 OGSA도 완전 웹서비스 플랫폼 위에서 돌아가는 그리드 웹서비스의 아키텍처로 변화되고 있다.

### 3.8 웹 2.0 플랫폼 서비스화(WEB 2.0 Platform Services)

웹 2.0은 인터넷 환경에서 사용자 측면에서 클라이언트 브라우저와 서버간의 통신에 주로 적용되는 반면, SOA는 기업환경에서 서로 서버 간의 통신에 적용된다. 웹 2.0과 SOA는 다른 배경에서 시작되었지만, 기술요소와 목표는 매우 흡사하다.

첫 번째 유사점으로, 웹 2.0은 SaaS (Software As A Service)라는 개념으로서 설치형 소프트웨어가 아닌 서비스 형태의 소프트웨어라는 개념으로 발전하고 있으며, SOA는 서비스 기반으로 IT 아키텍처를 구축하여 실시간 기업을 지원하기 위한 방법이다. 두 번째 유사점으로, 표면적으로는 웹 2.0의 주요기술 요소인 REST, AJAX, RSS, ATOM과 SOA의 WSRP, WS-*, SOAP, BPEL, BPM, ESB 등 기술 요소들은 관련이 없는 것처럼 보이지만 그 하부 구조에는 XML, HTTP, SOAP 등 현재 IT 기술의 원자적인 표준 기술들을 공유하고 있다. 뿐만 아니라, 그러한 표준 기술들에 대한 응용 기술들은 이전처럼 단일 벤더나 소수 기업들이 주도하는 것이 아니라 오픈소스나 OASIS 같은 표준화 그룹에서 이끌어가고 있기 때문에 웹 2.0이나 SOA 모두 세부 기술은 동일한 기술 스택을 사용한다고 해도 과언이 아니다. 세 번째 유사점으로, 사용자 인터페이스 개선으로 웹 2.0 은 이미 AJAX나 Rich Client 등의 기술을 통해서 여러 포털 사이트에서 사용자의 편의성을 제고하고 있으며, 기업에서는 SOA의 클라이언트 기술로서, 웹 2.0의 AJAX 기술이나 Rich Client 제품들이 SOA에 적용되고 있다. 네 번째 유사점으로, SOA의 복합 애플리케이션(Composite Application)과 웹 2.0의 매쉬업(Mashup)은 비슷한 유형의 통합 개념이다. 복합 애플리케이션이란 기업 내에 기존의 서로 다른 시스템에서 존재하는 애플리케이션을 프로세스에 의해 유기적으로 연결하는 방식의 통합 방법이다. 즉 서로 다른 시스템에 존재하는 애플리케이션을 프로세스 중심으로 통합하는 방법이라고 할 수 있다. 웹 2.0의 매쉬업은 네트워크에 구글, 네이버나 아마존 같은 오픈 API를 제공하는 업체들로부터 API를 조합하여 새로운 비즈니스 시스템을 구축하는 방법이다.

SOA와 웹 2.0은 서로 다른 영역과 사용주체로 인하여 다른 듯 보이지만 기술 스택이나 구현 방법 그리고 그 목적에서는 많은 공통점을 가지고 있다. SOA와 웹 2.0은 애플리케이션을 서비스로 파악하는 것을 기본 사상으로 하고 있어, 서로 경쟁하는 개념이 아닌 상호 보완하여 융합되는 방향으로 전개되고 있다. 즉, 웹 2.0이 인터넷 사용자에게 새로운 데스크탑 환경을 제공해준 것처럼 이제 엔터프라이즈 시스템에도 웹 2.0의 기술들이 SOA 와 결합하여 실시간 기업에 요구되는 IT 조직의 대응성을 향상시켜 주게 될 것이다.

## 4. 결 론

이 글에서는 융·복합 인프라로서의 SOA의 여러 특징들에 설명했다. SOA는 표준 서비스 인프라를 기반으로 약결합 형태의 서비스들을 연결하며 특정 프로그램이나 어플리케이션 범위를 초과한 엔터프라이즈에서 서비스의 재사용성 측면을 가지고 있다. 이들 재사용성이나 품질 속성 측면에서 공통 표준 서비스들을

가지고 있어야 한다. 이런 특징을 가진 SOA가 수행해야 하는 역할과 요건은 먼저 표준 분산 컴퓨팅 지원, 유비쿼터스 환경에서 디바이스와 서비스의 융·복합 지원, 표준 미들웨어 서비스 제공, 자동화된 서비스 연합, 그리고 서비스 지원 혹은 관리를 제공해야 한다. 유비쿼터스 환경에 존재하는 이질적이고 다양한 디바이스와 서비스를 융·복합 시키는 인프라로서의 서비스 지향 기술들의 발전 전망은 점차적으로 서비스 지능화(Intelligent), 서비스 이벤트화(Event-Based Services), 서비스 버스화(Enterprise Service Bus), 서비스 유틸리티화(Service On Demand), 서비스 임베디드화(Embedded Services)로 가고 있으며 모바일 산업, 그리드 산업 등 다양한 산업이 서비스화되어 가고 있다. 최근 관심을 끌고 있는 웹 2.0도 웹 플랫폼이 서비스화 되어가는 모습을 보여주고 있다.

## 참고문헌


[1] K.Channabasavaiah,K. Holley,E. M.Tuggle, Jr., "Migrating to a service-oriented architecture"

[2] 한국전산원, IT 839전략의 서비스 지향 융·통합 인프라로서 웹서비스 추진방안 연구

[3] M. P. Papazoglou, "Service-Oriented Computing: Concepts, Characteristics and Directions," Proceedings of the 4th Int'l Conference on Web Information Systems Engineering (WISE'03), pp.3-12, December 2003.

[4] F. Jammes and H. Smit. Service-Oriented Architectures for Devices - the SIRENA View. In INDIN'05, Perth, Australia, 2005.

[5] D. Krafzig, K. Banke, and D. Slama, Enterprise SOA: Service-Oriented Architecture Best Practices, Prentice Hall, 2005.

[6] T. Erl, Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall, 2005.

[7] T. Erl, Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services, Prentice Hall, 2004.

[8] CIO Council (2004). Service Component-Based Architecture Version 2.0, June.

[9] C. Koch, "How SOA Really Works," CIO, Aug. 2005; www.cio.com/blog_view.html?CID=10591.

[10] 이경하·이규철 (2004). "SOA(Service-Oriented Architecture)와 웹 서비스." 「한국정보과학회지」, 22(10): 5-10.

[11] Daryl C. Plummer, "Software Architecture Will Evolve from SOA and Events to Service Virtualization," Gartner, Mar. 2005.

[12] Brenda M. Michelson, "Event-Driven Architecture Overview," Patricia Seybold Group, Feb. 2006.

[13] Erl, T. (2005). Service-Oriented Architecture (SOA): Concepts, Technology, and Design, Prentice Hall PTR.

[14] M. Ditze, G. Kaemper, I. Jahnich, and R. Bernhardi-Grisson. Service-based Access to Distributed Embedded Devices through the Open Service Gateway. In INDIN'04, Berlin, Germany, 2004.

[15] Microsoft. Device Profile for Web Services, 2005

[16] Scott Deugd, Randy Carrol, Kevin E. Kelly, Bill Millet, and Jeffrey Ricker, "SODA: Service-Oriented Device Architecture", Pervasive Computing, IEEE CS and IEEE ComSoc, pp94-97, 2006.06.



**민 덕 기**

1986 고려대학교 산업공학과(학사)
1991 Michigan State(석사)
1995 Michigan State(박사)
1995 건국대학교 교수 부임
2006~현재 건국대학교 교수 재임
관심분야 : Distributed Systems, Ubiquitous Computing, Software Architecture, Web Services
E-mail : dkmin@konkuk.ac.kr

**이 용 환**

1997 건국대학교 행정학과(학사)
1999 건국대학교 컴퓨터공학부 (공학석사)
2006 건국대학교 컴퓨터공학부 (공학박사)
2003~2005 (주)인터넷커머스코리아 연구소장
2005~2006 동덕여대 겸임교수
2006~현재 건국대학교 연구교수
관심분야 : CBD, Ubiquitous Computing, Embedded Software
E-mail : yhlee@konkuk.ac.kr