

서비스 컴포넌트 아키텍처

SK C&C 임철홍

1. 요약

국내외적으로 차세대 아키텍처인 SOA(Service Oriented Architecture)에 대한 관심이 높아지고 있으며, 수년 내에 SOA가 일반적인 아키텍처가 될 것으로 전망되고 있다. 하지만, SOA를 실제로 구축하고 활용하기에는 많은 장벽들이 존재하고 있다. 본 논문에서는 SOA기반의 Application을 구축하기 위해 필요한 컴포넌트 규격에 대한 연구와 함께 기존의 분산 객체 컴포넌트를 확장하여 비즈니스 컴포넌트로 구축 가능하게 지원하는 SCA(Service Component Architecture)를 기반으로 SOA를 구축하는 방안을 제시하여, SOA의 장점인 유연성과 상호운용성을 보장하는 Application 구축에 활용이 가능하도록 하고, 이론 및 사상적인 측면에서 벗어나 현실화된 Architecture로써 SOA가 도입될 수 있도록 하고자 한다.

2. 서론

SOA(Service Oriented Architecture)에 대한 기대와 관심이 높아지면서, SOA를 지향하는 표준과 솔루션의 출시가 활발하다. 현재 상황에서 완전한 SOA의 적용 및 구축은 어렵고 여전히 SOA는 이상적인 형태를 지향하는 것이라는 의견도 많다. SOA는 Service를 기반으로 구축되는 것인데, 이러한 Service에 대한 표준을 찾기가 어렵다. 웹 서비스의 경우 상호운용성을 보장하는 Messaging에 대한 표준 중심이며, business Logic을 포함하여 컴포넌트 명세라고 하기에는 부족한 점이 많다.

이러한 문제점을 해결하기 위하여 SCA(Service Component Architecture)[1]가 등장하게 되었다. SCA는 SOA에 기반한 시스템과 Application을 구축하기 위한 모델을 정의하고 있으며, Language와 Solution에 독립적으로 구현이 가능하다. SCA는 BEA, IBM, IONA, ORACLE, Sybase, SAP, Siebel 등 주요 WAS, DB, Business Application

의 주요 Vendor가 참여하고 있는 OSOA(Open Service Oriented Architecture)[2]에 의해서 표준이 제정되었다.

본 논문에서는 컴포넌트 품질 모델을 기반으로 이러한 품질 요구 사항들을 기존의 컴포넌트와 SCA가 어떻게 충족하고 있는지를 분석하고, SCA를 활용하고 SOA를 구축하기 위한 방안을 제시하고자 한다.

3. SOA 컴포넌트 요구 사항

3.1 SOA Application 특성

SOA는 분산 객체의 형태로 존재하는 컴포넌트들간에 Message 통신을 통해 정보를 교환하는 느슨하게 연결된(Loosely Coupling) 형태의 S/W Architecture이다[3]. 개별 컴포넌트들은 미리 약속된 표준 메시지(주로 SOAP[4] 활용)를 송수신 하며 그림 1처럼 동작을 수행한다.

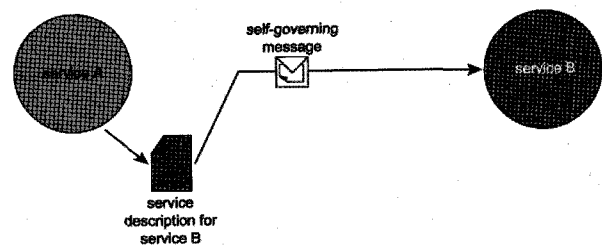


그림 1 컴포넌트간 표준 메시지 교환

SOA의 Loosely Coupling한 결합 성격을 활용하여 SOA Application은 XML기반의 메시지를 각 컴포넌트에 전송하여 입력/수정/삭제/조회의 처리가 가능하다. Web Service를 조합하여 Process 기반의 처리를 지원하는 ESB(Enterprise Service Bus)[5]를 활용하여 Composite Component형태의 Application을 구성하는 모습은 그림 2와 같다. ESB는 Function 중심의 단위 서비스를 연결 조합하게 되고, 이러한 ESB를 포함하는 Application 들은 SOA Application의 형태로 동작하게 된다. Process 기반으로 컴포넌트의

호출과 동작을 제어하는 형태를 Service Orchestration 이라고 하며, 대표적인 표준으로 BPEL(Business Process Execution Language)[6]가 있다.

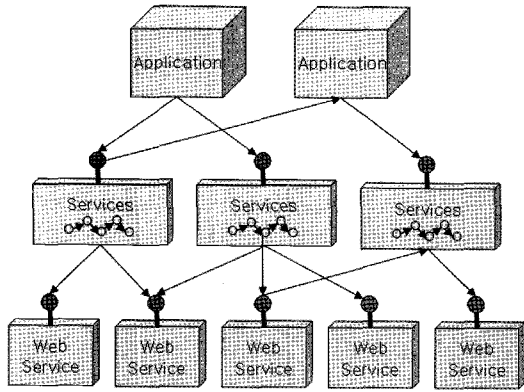


그림 2 SOA Application

3.2 SOA 컴포넌트 요구 사항

SOA 환경에서 적합한 컴포넌트 환경을 정의하기 위하여 컴포넌트 품질 모델을 기반으로 요구 사항을 정의 하고자 한다. ISO-9126의 범용적인 S/W품질 모델을 컴포넌트에 맞추어 정의 하고자 하는 연구가 진행되어 왔다. 한국정보통신기술협회 (TTA)[7]에서 다음 표 1과 같이 컴포넌트 품질 모델을 정의 하였다.

표 1 컴포넌트 품질 모델

품질 요소	
기능성	요구 사항에 명시 되어 있는 기능들이 충분히 구현 되었는지를 측정
재사용성	재사용 단위의 컴포넌트가 동일한 도메인을 대상으로 범용적으로 이용될 수 있는지에 대한 척도(특화성, 결합성, 범용성)
준수성	컴포넌트를 개발함에 있어서 지켜야 하는 일종의 규약이나 표준의 준수 여부 (도메인 표준 준수성, 플랫폼 준수성)
품질 항목	
적합성	요구된 기능들이 구현된 정도
결합성	컴포넌트 각각이 다른 컴포넌트와의 연동시에 컴포넌트 간에 발생하는 상호 작용의 정도
범용성	컴포넌트가 가지는 기능성들이 표준 도메인 모델에 잘 맞는지의 정도
특화성	컴포넌트가 가지는 기능을 수행하는데 필요한 로직, Attribute, Workflow에 대한 가변성이 얼마나 구현 되었는지의 정도
도메인 표준 준수성	컴포넌트가 주어진 도메인에 존재하는 표준들을 얼마나 준수하는지의 정도
플랫폼 준수성	컴포넌트가 미리 정해진 플랫폼을 얼마나 준수하는지의 정도

정의된 컴포넌트 품질 모델을 기반으로 SOA에 적합한 품질 요구 사항을 정의 하면 다음 표 2와 같다. 상호운용성과 이식성에 대한 품질 모델을 별도로 정의 하기도 한다.

표 2 컴포넌트 품질 요구 사항

품질 요구 사항	
결합성	Loosely Coupling 지원하여 유연성 확보
특화성	로직, Attribute, Workflow에 대한 변경 시에 코딩을 최소화 하여 구현이 가능해야 함
도메인 표준 준수성	도메인 기반 컴포넌트 제작과 활용이 가능해야 함
플랫폼 준수성	J2EE, .NET, Legacy에 대한 표준을 지원 하여야 함 상호운용성과 이식성을 보장해야 함

4. 기존 컴포넌트 활용 SOA 구축 방안

4.1 컴포넌트 표준

현재의 컴포넌트 표준은 주로 분산 환경을 지원하는 분산 객체 컴포넌트를 중심으로 발전 되어 왔다. 이러한 분산 객체 컴포넌트의 활용이 점점 많아 지고 있으며, SOA를 구현 하는 방안에 있어서 Web Service를 기반으로 구현이 되고 있다. 다음 표 3은 분산 객체 컴포넌트를 구현하는 표준을 설명하고 있다.

표 3 분산 객체 컴포넌트 표준

표준	설명
CORBA	- 산업계 최초의 다중 언어 기반 컴포넌트 표준 - IDL(Interface Definition Language) 지원 - 여러 플랫폼에서 동일하게 활용 가능
EJB	- 자바 기반의 컴포넌트 표준 - Enterprise Java Beans - 분산 트랜잭션 지원 - Local Interface 지원, 성능 증대 - Annotation 및 POJO지원 (EJB3) - RMI (Remote Method Invocation)과 CORBA의 IIOP지원
COM	- Microsoft에서 제정한 컴포넌트 표준 - 컴포넌트 간의 인터페이스 기반 활용 지원 - 분산 환경 지원 하는 DOM규격과 트랜잭션을 지원하는 COM+ 규격 발전
Web Service	- SOAP, WSDL, UDDI와 같은 표준으로 구성 - WS-*의 확장 표준 제정 - Java Web Service의 경우 RMI를 활용하여 통신

위의 분산 객체 컴포넌트는 분산 환경 및 트랜잭션을 지원하는 것이 주요 기능이다. CBD방법론(Component Based Development)을 활용하여 비즈니스 컴포넌트를 구성한다면, Business Logic과 이러한 분산 환경

을 지원하는 컴포넌트를 활용하여 전체 컴포넌트를 Design해야 한다.

다음 그림 3은 EJB를 활용하여 전체 비즈니스 컴포넌트를 구성하고 있다. 컴포넌트를 구성하는 여러 Java Beans들은 EJB Bean에 의해서 호출이 되며 (Façade의 역할 수행), Tightly하게 Coupling되어 있다. 이렇게 전체적인 컴포넌트 규격이 없기 때문에 컴포넌트의 구축이 어려우며, 이미 구축된 컴포넌트도 기술 및 플랫폼에 종속되어 활용성이 떨어지게 된다. 웹 서비스를 활용하는 경우에도 EJB의 경우와 비슷한 문제점이 발생 되게 된다. 인터페이스의 역할을 하는 웹 서비스 컴포넌트의 경우 loosely coupling을 지원 하지만, Business Logic을 가지고 있는 Class와는 여전히 Tightly Coupling을 가지게 된다.

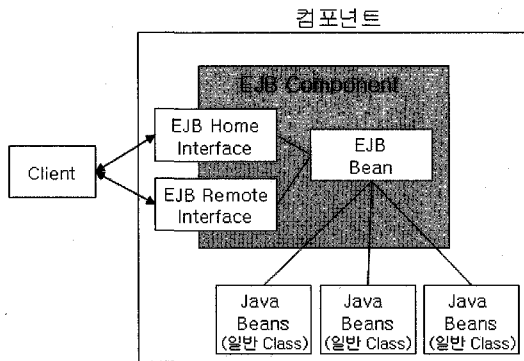


그림 3 EJB활용 컴포넌트 구축 방안

현재 활발하게 진행 되고 있는 오픈 소스 프로젝트를 중심으로 POJO (Plain Old Java Object)를 활용한 Framework 기반 컴포넌트의 활용이 활발 하다. 이들은 Business Logic에 집중하며, Light Weight하며 간단한 컴포넌트 환경을 구현하고자 노력하고 있다. Spring, HiveMind와 같은 프로젝트 들이 대표적이다. 이들은 Dependency Injection(8) 개념을 통해서 컴포넌트들간의 loosely coupling을 지원 하고 있다. 하지만, 표준 자체가 Framework API에 종속되는 단점이 있다.

4.2 분산 객체 컴포넌트 기반 SOA 구축 방안

기존의 SOA 구축을 위한 Architecture를 살펴보면 그림 4와 같다.

컴포넌트들은 EJB, POJO, COM+의 형태로 구축이 되고, 이들간의 연결은 여전히 Tightly Coupling한 형태로 이루어 지게 된다. 컴포넌트, 웹 서비스, BPEL Processor등이 개별적인 표준을 가지고 연결되어 있는 형태이다. Web Service를 통하여 서로 통신하거나 BPEL Processor와 통신이 가능하지만 전체적인 컴포넌트를 형성하는 표준이 없으며, 기술을 활

용하기 위해서도 API를 직접 활용하는 등의 많은 작업이 필요 하다. 또한 솔루션이나 API에 대한 종속성을 가지기 때문에 구현된 Application의 유연성, 상호 운용성 및 재사용성에 제약을 받게 된다.

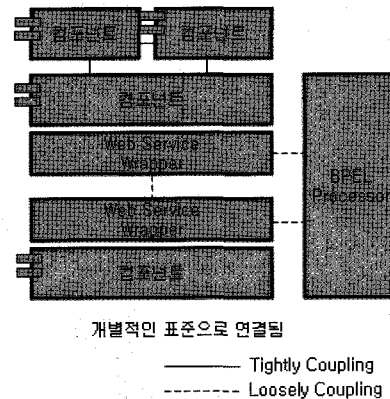


그림 4 SOA구축을 위한 기존의 Architecture

5. SOA 컴포넌트 표준인 SCA

5.1 SCA의 구성

그림 5는 SCA의 전체 Architecture를 설명하고 있다(1). 여러 Business Logic 컴포넌트들이 서로 연관 되어 구성 되고 있으며, 통신이나 연동을 위한 처리를 위한 Layer 및 Transaction 등의 Infrastructure Capability를 제공하는 Layer가 있다. 각 Business Logic 컴포넌트는 Adapter나 미들웨어 등에 쉽게 연동이 가능한 형태이다. 실제로 Business Logic을 수행하는 컴포넌트에 초점을 맞추고 있는 형태이다.

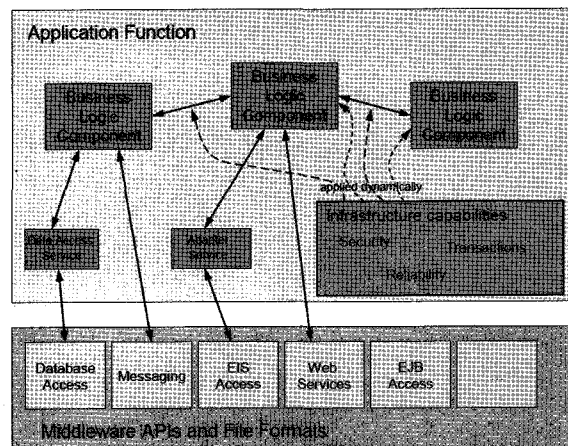


그림 5 SCA 개념도

세부적인 컴포넌트 구성도는 그림 6과 같다. 컴포넌트의 구현과 활용 되는 기술과는 공통된 Implementation Layer를 활용하여 추상화 하고 있다. 즉, 컴포넌트의 구현은 바뀌지 않고 해당 기술들을 바꾸어 사용 할 수

있다는 장점이 있다. 이를 위하여 인터페이스에 활용하는 기술에 대한 정의를 XML형태의 설정 파일(sca.module)을 통하여 기술 한다.

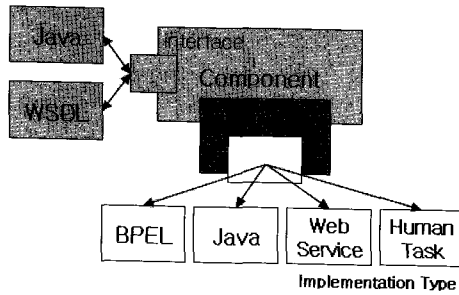


그림 6 SCA 컴포넌트 구성도

① 컴포넌트 Implementation

Business 컴포넌트는 Coarse grained 형태로 내부적으로 여러 단위 컴포넌트로 구성 되어 있다. 개별 단위 컴포넌트는 다른 컴포넌트에게 호출/사용 되도록 service (interface)를 제공하여 서로간에 reference 가 될 수 있도록 지원 한다. 지원 Type으로는 Java Interface(POJO지원)과 WSDL Port Type(BPEL 지원)이 존재한다. 내부적인 연결(Local Binding)로는 Java Interface를 활용하고 외부 연결로는 WSDL를 통한 웹 서비스나 BPEL을 통한 연결이 가능하도록 지원 하고 있다. 다음 그림 7은 컴포넌트의 구성을 설명하고 있다.

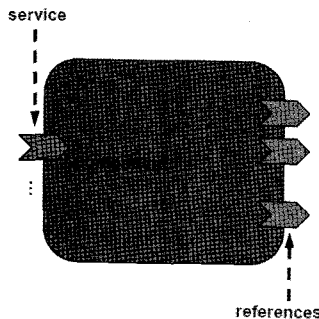


그림 7 컴포넌트 구성

② Assembly

기존의 컴포넌트 환경과 비교하여 가장 다른 점은 컴포넌트간의 연결을 지원하여 복합 컴포넌트(Composite Component)가 형성 되도록 지원한다는 점이다.

개별 컴포넌트는 Service와 Reference를 활용하여 여러 컴포넌트를 묶어서 Module을 형성하게 된다. 각 Module에는 Entry Point와 External Service가 존재하여 다른 Module과 연계가 가능하다. 여러 Module이 연계되어 Subsystem을 형성하고 이들 Subsystem간의 연결을 통한 System단위의 컴포넌트 형성이 가능하다.

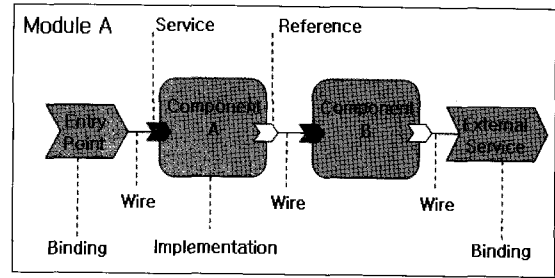


그림 8 Module 단위 Assembly

System에도 Entry Point와 External Service가 존재하여 다른 System과의 연계가 가능하다. 단위 컴포넌트가 연결되어 Module Assembly를 구성하고, Subsystem이 연결되어 System Assembly를 구성한다. 다음 그림 8은 Module Assembly를 설명하는 그림이다.

System Assembly의 경우 그림 9는 Module Assembly로 이루어진 Module간의 연결을 통하여 전체 System이 구성된다. 내부적으로 Module간의 연결을 통한 Subsystem을 구성하고 Subsystem이 Entry Point와 External Service를 가지고 있다.

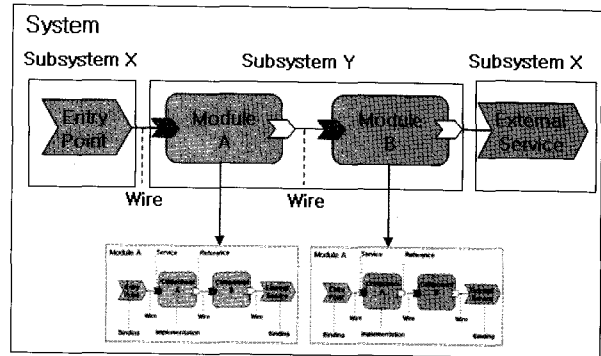


그림 9 System 단위 Assembly

③ Binding

Binding은 Web services, JMS/Messaging, CORBA IIOP and DB Stored Procedures 등의 기술이 활용되며, entry points 와 external services에서 주로 활용 되어, 내 외부적인 컴포넌트간의 인터페이스를 통합하는 역할을 수행 한다.

5.2 SCA를 활용한 SOA구축 방안

SCA를 활용하여 SOA를 구축하는 경우에는 Module, Subsystem 단위의 컴포넌트 명세를 따르게 되며, 컴포넌트들간의 연결도 Loosely Coupling형태를 가지게 된다. Binding의 경우도 직접적인 API 사용을 추상화 시킬 수 있으므로, Application의 유연성 및 재사용성을 향상 시킬 수 있으며, 여러 컴포넌트가 연결된 복합 컴포넌트(Composite Component)의 형성과

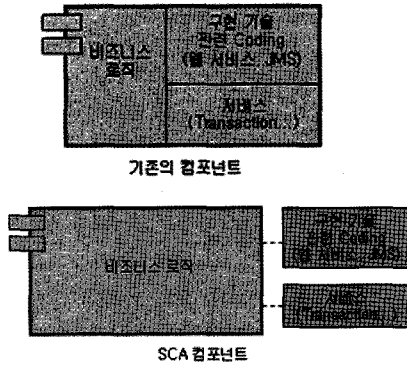


그림 10 기존의 컴포넌트와 SCA 컴포넌트 비교

사용이 가능하도록 지원 한다. 컴포넌트, 웹 서비스, BPEL Processor에 대한 개별 표준을 수용하면서 전체적으로 동일한 표준 하에 구성이 되도록 지원 한다. 그림 10은 이것을 설명하고 있다. 기존 컴포넌트에 포함되던 구현 기술 관련 Coding을 XML 설정 형태로 컴포넌트 밖으로 분리해 놓았기 때문에, 구현 기술이 바뀌어도 컴포넌트를 그대로 재사용 할 수 있게 된다.

다음 그림 11은 SCA를 활용하여 SOA를 구축하는 방안을 설명하고 있다. 내부의 컴포넌트들이 Loosely Coupled 형태로 구성이 되며, Subsystem을 포함한 모든 구성 요소들이 표준 명세 하에서 구성되고 동작하게 된다. Composite 컴포넌트 전체에 대한 표준 명세를 지원 할 수 있다.

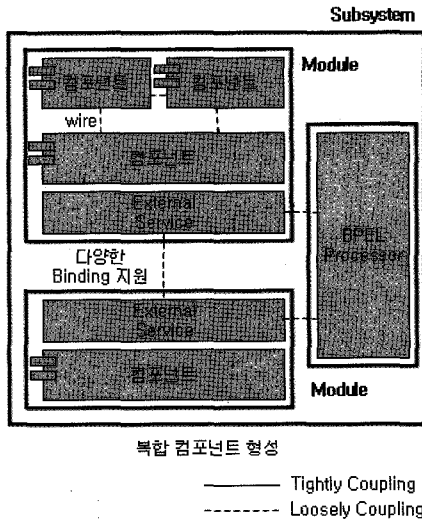


그림 11 SCA를 활용한 SOA 구축 Architecture

6. 컴포넌트 품질 요구 사항 평가

6.1 SCA의 컴포넌트 품질 특성

SCA는 기존의 컴포넌트가 가진 문제점을 해결할 수 있도록 지원 하고 있으며, SOA 환경에 적합하도록 기능을 제공 한다. 컴포넌트 품질 모델에 기반하여

표 4 SCA 컴포넌트 품질 특성

SCA 컴포넌트 품질 특성	
결합성	- Loosely coupling 지원 - 단위 컴포넌트, Module, Subsystem 단위를 지원하여 컴포넌트 간의 결합을 표현하여 복합 컴포넌트 구성 지원 - Business Logic과 구현 기술을 분리 하여 기능의 응집도가 높음 - 컴포넌트간의 wire 기능을 제공하여 관계성을 표현
특화성	- Workflow 기능을 활용 하여 컴포넌트의 사용을 편리하게 적용 가능 (BPEL표준 수용) - 사용 기술에 대하여 설정 파일 형태로 관리하여 활용 가능
도메인 표준 준수성	- Business Logic과 활용 기술을 분리하여 도메인 기반 컴포넌트 제작이 가능함
플랫폼 준수성	- Business Logic과 구현 기술을 분리 하여 이식성이 우수함 - 웹 서비스 기반 메시징 표준을 준수하여 상호운용성이 높음 - J2EE, .NET, Legacy 등의 플랫폼에 독립적으로 적용이 가능

SCA가 가진 특성은 표 4와 같다.

위와 같은 컴포넌트 품질 특성을 통하여 SOA 기반 구축을 지원하여 구축된 Application은 SOA의 특성인 재사용성, 유연성, 이식성, 구축 용이성이 높은 품질 특성을 가지게 된다[9].

6.2 기존 컴포넌트와 특성 비교

SCA는 기존의 컴포넌트 규격을 발전시켜 분산 환경과 인터페이스 기반의 활용성을 지원하면서 Business Logic에 집중하면서 컴포넌트간의 연계를 통한 복합 컴포넌트 구현이 가능하도록 지원 하고 있다. 다음 표 5는 기존의 컴포넌트와 SCA컴포넌트의 특성을 비교하고 있다.

표 5 SCA와 기존 컴포넌트 특성 비교

특성	기존 컴포넌트	SCA 컴포넌트
기능	분산 환경 지원 인터페이스 제공	기존 컴포넌트 기능과 Business Logic 지원
형태	단일 컴포넌트	복합 컴포넌트
바인딩	Tightly	Loosely Coupling
구성	기술과 Logic 혼재	기술과 Logic 분리
요소 기술	단일 기술 지원 (IIOP 또는 RMI등)	통합적인 기술 지원 (JCA, 웹 서비스, BPEL 등)
활용	EAI 기능 활용	SOA 구축

7. 결 론

본 연구에서 SCA에 대한 컴포넌트 품질 특성을 분

석하고, SCA를 활용하여 SOA를 구현 할 수 있는 방안을 제시 하였다. SCA는 현재 version 0.9의 상태이며, 현재 활용 가능한 기술은 SCA 컴포넌트 Binding, 웹 서비스이다. BPEL, EJB의 경우 향후 버전에서 지원 할 예정이다[1].

SCA를 활용할 경우 SOA에서 활용하는 복합 컴포넌트 전체에 대한 명세를 제공하고 있으며, 웹 서비스, BPEL 등에 대한 표준을 준수하여 상호 운영성 확보가 가능하다. 기존의 컴포넌트 표준이 분산 객체의 형태를 지원하고자 하는데 중점을 두었다면, SCA는 여러 기술 요소를 단일화된 표준으로 통합하면서 Business Logic에 집중하여 컴포넌트가 구현 될 수 있도록 지원 하고 있다.

SCA가 이러한 장점을 가지고 있지만, 벤더 중심으로 제정 된 표준이고, 현재 초기 상태이기 때문에 시장에서 활발하게 적용될 지 여부는 아직 시간을 필요로 하는 부분이다. 하지만, SCA가 기존의 컴포넌트의 부족한 점을 보완하여 기술과 비즈니스 관점에서 통합적인, 획기적인 시도를 하고 있으며, 이러한 시도는 향후에 제정될 표준에 많은 영향을 주게 되고 더욱 완성도가 높은 표준의 등장도 가능하리라 생각 한다.

참고문헌

- [1] SCA Specification, "http://www.osoa.org/display/Main/SCA+Resources".
- [2] OSOA (Open Service Oriented Architecture), "http://www.osoa.org".
- [3] Thomas Erl, "Service-Oriented Architecture: Concepts, Technology, and Design", Prentice Hall PTR, 2005.

- [4] SOAP version1.2, "http://www.w3.org/2000/TR /Group".
- [5] David A. Chappell, "Enterprise Service Bus", O'Reilly, 2004.
- [6] BPEL, "http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel".
- [7] 김남희, 신석규, "소프트웨어 컴포넌트 시험 기술 동향", TTA저널 제88호.
- [8] Martin Fowler, "Inversion of Control Containers and the Dependency Injection pattern ", www.martinfowler.com/articles/injection.html.
- [9] 임철홍, 홍도석, 최정준, "ESB 기반 SOA Application의 S/W 아키텍처적 관점에서의 평가와 개선 방안", IT서비스 학회지 제5권 2호.

임 철 홍



1996 고려대학교 재료공학과(공학학사)
 1997~1999 한승 정보시스템 MIS연구원
 2000~2002 한국 디지털라인 기술연구소 연구원
 2002~2005 SK 커뮤니케이션즈 컨텍스트 기술연구소 책임연구원
 2003 고려대학교 전자컴퓨터공학과 (공학석사)
 2004 정보처리 기술사(전자계산조직응용)
 2005~현재 SK C&C S/W공학센터 과장
 관심분야: SOA, 시맨틱 웹, EAI, CBD
 E-mail : imich@skcc.com

한국소프트웨어공학 학술대회

- 일 자 : 2007년 2월 22 ~ 24일
- 장 소 : 용평리조트
- 내 용 : 논문발표 등
- 주 최 : 소프트웨어공학 연구회
- 상세안내 : http://www.sigse-kiss.or.kr