CBD기반 SOA 도입 방안과 요소 기술 연구

임철홍
SK C&C S/W공학 센터

A Study on the Adoption of the CBD based SOA and Elemental Technologies

Im, Chol Hong
SK C&C S/W Engineering Center
E-mail : imich@skcc.com

Abstract

The SOA (Service Oriented Architecture) is whooped up and many Organizations have considered the SOA as the next generation architecture. Due to misunderstanding of the SOA, they think that the CBD is not useful anymore and may be completely replaced by the SOA. Because of misunderstanding of the SOA, they think that the CBD is not useful anymore and may be completely replaced by the SOA. In this paper, we presented the critical success factor, technology and successful construction methods of the CBD based SOA. So, we expect that this paper provides the guides for adopting and realizing the SOA.

KEYWORD: SOA, CBD, ESB, 서비스, 컴포넌트, SCA

1. 서론

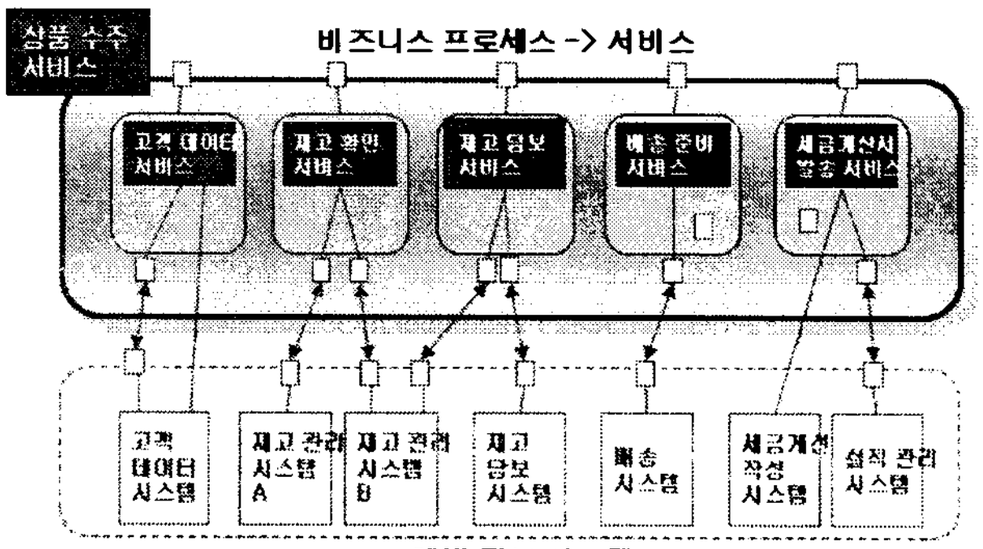
하루가 다르게 변화하고 있는 비즈니스 및 IT 환경에서 경쟁사 보다 더욱 민첩하고 유연하게 대응하는 것이 중요한 경쟁력으로 인식 되고 있다. 이러한 환경에서 많은 기업들이 차세대 시스템 아키텍처로 SOA(Service Oriented Architecture)[1]의 도입을 고려하고 있다. 하지만, SOA는 도입만 하면 모든 문제가 해결 되는 것은 아니다. SOA 도입 목적을 분명하게 수립하고, SOA를 통해 효과를 얻을 수 있는지에 대한 검토가 선행 되어야 한다.

본 논문에서는 차세대 시스템과 같은 핵심 정보시스템 구축을 컴포넌트 기반인 CBD (Component Based Development) 방법론을 활용하여 수행하면서 이를 통해 SOA 구축을 위한 컴포넌트 확보와 시스템 아키텍처 기반을 마련하고, 이를 기반으로 SOA를 구축하기 위한 성공요인과 구축 방안을 제시하였다. SOA의 구축은 단위 정보시스템 관점이 아닌 전체 조직, 파트너 및 고객을 포함하는 단위에서 비즈니스 프로세스를 목표로 하여 CBD기반 정보시스템과 기 구축된 Legacy시스템과의 효율적인 통합을 통하여 구성 할 수 있다. SOA의 구축은 Big bang에 의하여 단시간적에 구축이 불가능하며 단계적이고 점진적인 방법에 의해서 최적화되고 완성 될 수 있다.

2. SOA 도입 전략

2.1 SOA 도입의 목적

SOA는 전사적인 관점에서 조직의 비전, 전략 및 프로세스를 설계하고, 이를 지원하기 위하여 상호운영성을 보장하면서 유연하고 민첩성(Agility) 있는 시스템의 구축과 운영을 목적으로 한다. 이를 위해서 조직의 내/외부 정보시스템들과 업무 처리 담당자는 프로세스 관점에서 서로 끊임이 없이 실 시간적으로 정보를 교환 하면서 동작 되어야 한다. (그림1)은 프로세스 관점에서 SOA가 도입되어 활용 되는 모습을 보여 주고 있다.



(그림1) SOA 도입 방안 사례

2.2 SOA 도입 요건

SOA 도입의 예상 효과들이 많지만, SOA 도입 시에 고려 되어야 할 요건들이 있다.

첫 번째 요건으로는 올바른 SOA도입 관점이다. SOA의 도입은 업무 프로세스 관점에서 고려가 되어야 한다. SOA에서 정의 하고 있는 서비스는 개별 컴포넌트들이 모여서 만들어진 추상화도가 높은 컴포넌트 단위이다. 따라서 특정 정보시스템 하나에 국한되어 서비스의 식별과 활용이 어렵고 설계와 구현이 CBD와 달라지는 점이 없다. 따라서, 현재의 C/S나 웹 기반 시스템을 전면 개편하여 SOA 시스템으로 전환 하려는 시도는 SOA에 대한 올바른 도입 관점이 아니다.

두 번째로는 대상 시스템의 성격을 고려한 도입이 필요하다. 현재 많은 정보 시스템이 사용자의 입력을 바로 DB로 처리하는 형태이다. 이러한 경우 내부적인 업무 구현이 대부분 DB 인터페이스에 활용이 되며, 공통적으로 활용 가능한 컴포넌트 및 서비스의 식별은 어렵다.

세 번째로는 성능에 대한 고려 이다. 이 부분은 현재의 기술적인 한계 때문에 고려 되어야 한다. 특정 시스템 내부에서 수행되는 처리 속도와 여러 시스템을 거쳐서 분산 환경에서 처리 되는 속도는 많은 차이가 난다. 따라서, 시스템 내부에서 처리가 가능한 기능을 공통컴포넌트 구축 및 활용을 위하여 분산 환경으로 구현 할 경우에는 성능 부분에 대한 고려가 충분히 선행 되어야 할 것이다.

2.3 SOA 도입 전략

SOA를 지원하는 Application 형태는 개별적인 컴포넌트가 서비스를 구성하고, 서비스들이 연계되어 Composite 컴포넌트를 구성 한다. 컴포넌트의 가장 작은 구성 단위는 CBD에서 활용 되는 컴포넌트 단위이기 때문에 CBD가 SOA을 구현하는 시작점인 것을 알 수 있다. CBD와 SOA는 전혀 다른 별개의 개념이 아니라, CBD의 개념을 단위 조직 및 정보시스템에 국한 되는 것이 아니라, 업무 프로세스를 구성하는 내/외부 시스템까지 확장 시킨 개념 이다. 다음 (표1)에서 CBD와 SOA의 개념을 비교하였다[2].

특성	SOA	CBD
관점	기업 내/외부 통합	기업 내부 특정 시스템
프로세스	프로세스에 집중	컴포넌트에 집중
컴포넌트 특성	서비스 컴포넌트 (높은 추상화 수준)	기능/기술 컴포넌트 (기술적 관점)
모델링 기법	컨설팅 기법(IDEF 등), UML, EA	UML
목표	비즈니스 목표와 연결 목적	시스템 관점 목표
Platform	이기종 통합 연계	J2EE, .NET 개별적 연계
연계방식	Loosely Coupling (SOAP)	Tightly Coupling (Serialization)
Interface	공개적인 인터페이스 (WSDL 활용)	개별 인터페이스

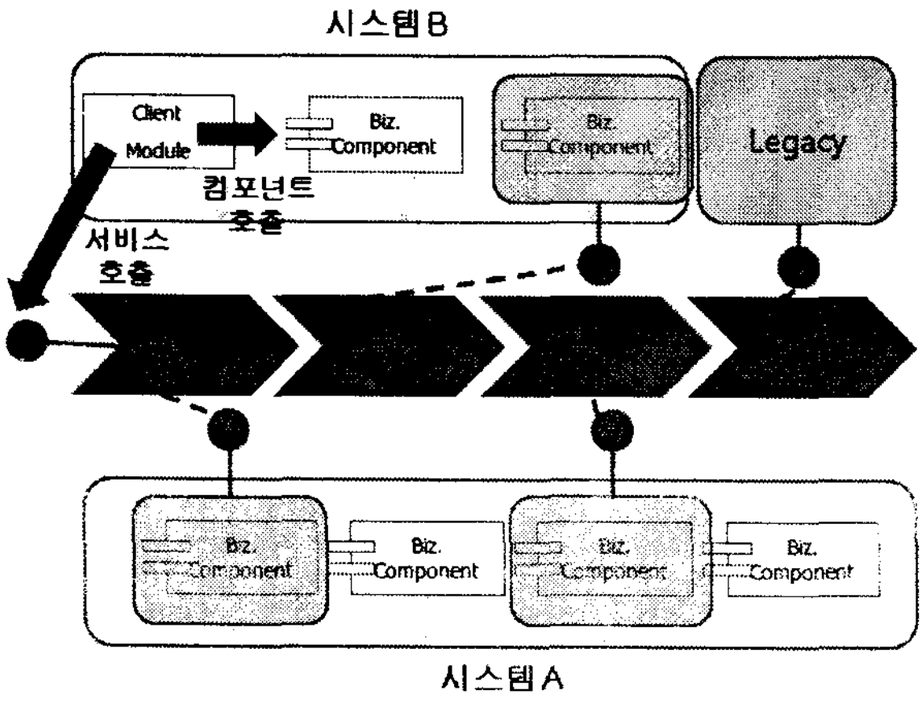
(표1) SOA와 CBD의 특성 비교

SOA의 도입은 CBD기반으로 시스템을 새롭게 구축하거나 기존에 구축된 정보시스템을 확장 발전 시켜 단일 시스템뿐만 아니라, 내/외부 정보시스템을 포함한 협업적인 프로세스를 포괄하여 처리 속도의 단축과 자동화 수준을 높여서 RTE (실시간 기업 환경)을 실현 할 수 있도록 기반을 제공 한다. SOA도입을 위한 필수 요건으로 SOA 구현을 위한 컴포넌트가 확보가 되어야 하며, 이를 위하여 CBD기반 SOA프로젝트를 수행하는 것이 필요 하다.

3. CBD기반 SOA구축 성공 요건

3.1 SOA 구축 목적과 범위 수립

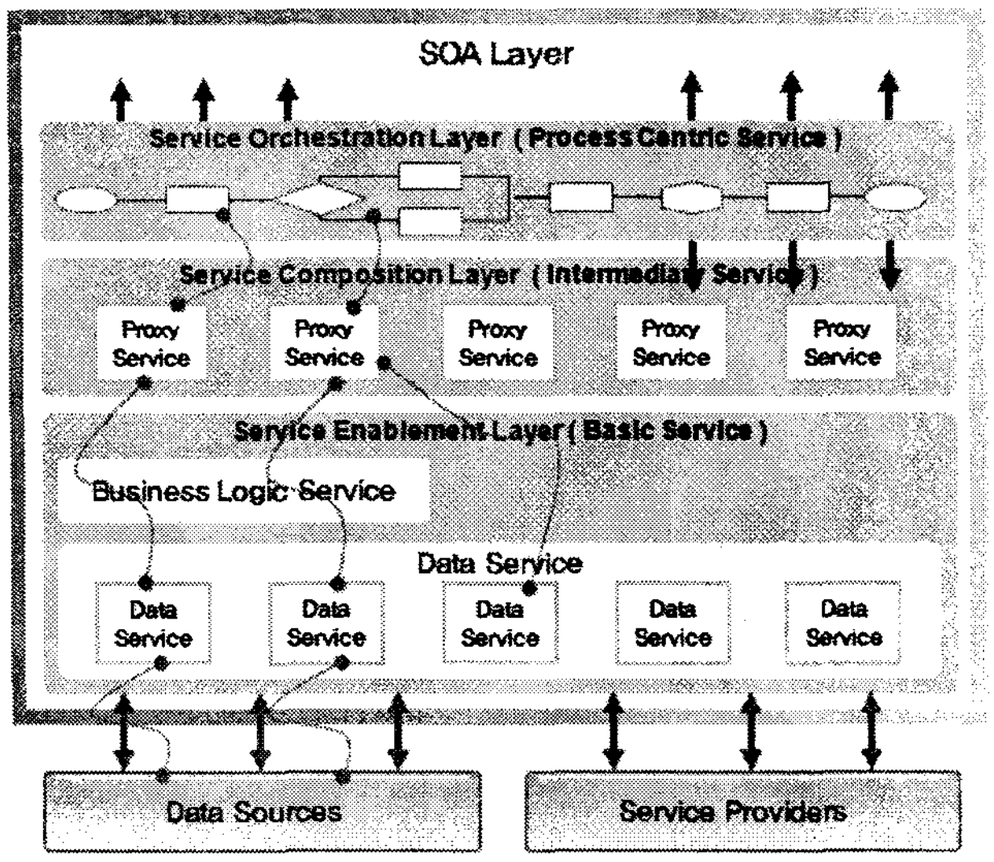
모든 Application 환경을 SOA로 구축하는 것은 많은 비용이 소요되고, 복잡한 아키텍처를 가지며, 원하는 성능도 얻기가 어렵다. SOA의 구축의 성공 요건으로는 SOA도입의 목적 수립과 목적에 맞는 범위를 선정하는 것이다. 목적에 부합 되는 비즈니스 프로세스를 선정하고, 활용 되는 서비스를 식별 하는 것과, 구축된 비즈니스 프로세스를 활용하는 환경을 제공 해야 한다. 다음 (그림2)에서 시스템A, 시스템B에서의 컴포넌트가 외부에서 호출 가능하도록 노출 되어 있고, 프로세스에 맞추어 복합 컴포넌트 형태의 서비스를 구성하고 있다.



(그림2) 복합 컴포넌트 구성

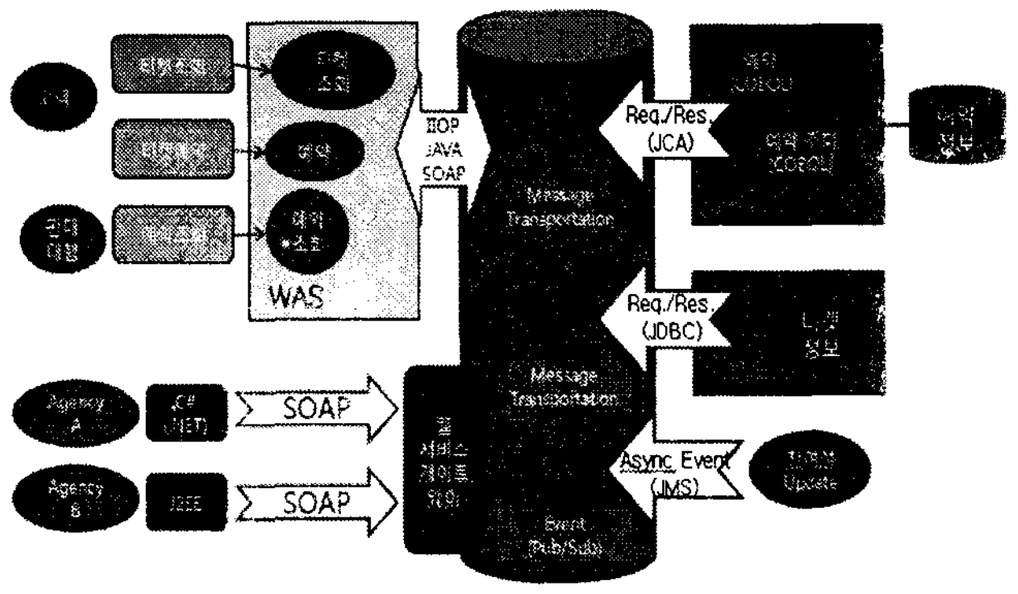
3.2 통합 관점의 아키텍처 수립

SOA의 구축은 다른 시스템 들과의 연계와 통합을 전제로 구성이 되며, 다양한 내/외부 시스템과의 통합을 필요로 한다. 따라서, 시스템간의 상호 운영성을 보장 하면서, 데이터, 서비스 및 프로세스 기반의 통합을 보장해야 한다. 다음 (그림3)은 상위 수준의 통합 아키텍처 모델[3] 이다.



(그림3) 통합 아키텍처 모델

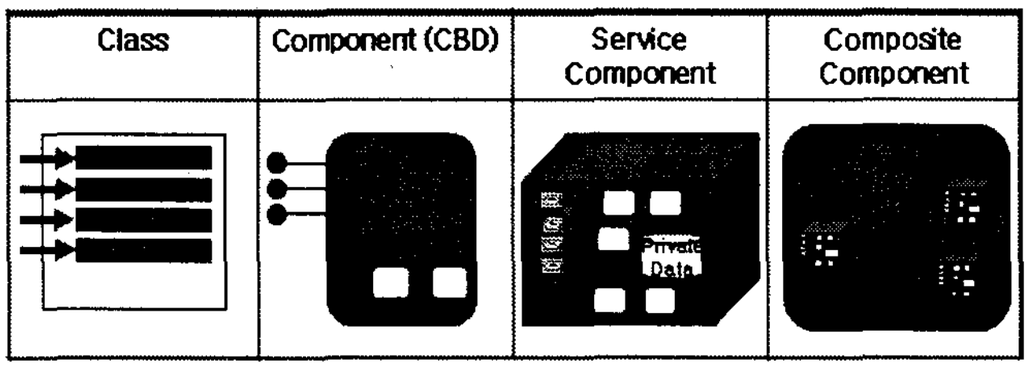
내/외부에 존재하는 많은 정보시스템 자원들과 효율적인 통합을 위해서는 인터페이스 표준화와 신뢰성 있는 통신을 지원 해야 하며, 이러한 역할을 수행하는 미들웨어로 ESB(Enterprise Service Bus)가 있다. ESB는 SOA환경에서 여러 컴포넌트들 간의 연결을 지원하고, Message에 대한 Routing과 신뢰성 통신을 보장 한다[4]. (그림4)에서 ESB를 활용한 서비스 통합을 설명하고 있다.



(그림4) ESB 활용

3.3 효율적 컴포넌트 설계 및 활용

SOA는 CBD단위의 컴포넌트를 확장하여 서비스 단위의 기능을 지원하면서, 서비스의 발견과 사용이 유용하도록 지원하여 공통 서비스의 제공과 재사용 및 변화에 대한 유연성을 제공하고자 하는 목적을 가지고 있다. 컴포넌트는 다음 (그림5)와 같은 형태로 분류가 가능 하다.



(그림5) 컴포넌트 분류

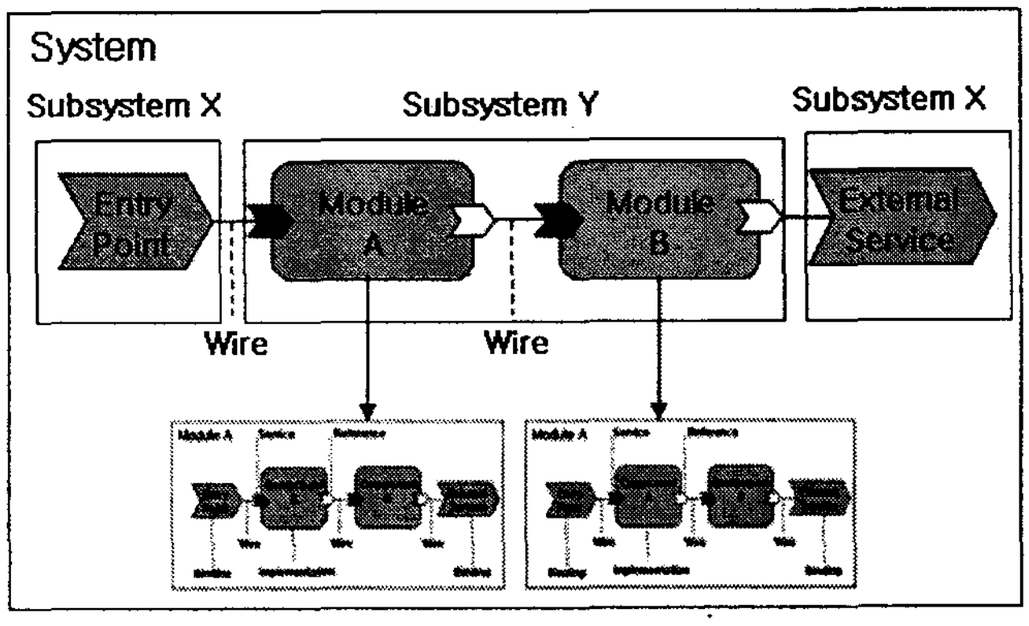
SOA에서의 컴포넌트는 복합 컴포넌트 형태로 구성이 가능하도록 Loosely Coupling 기능을 제공하여 쉽게 조립과 활용이 가능하도록 지원하여야 하며, 다양한 인터페이스 요건을 수용하여 통합을 지원해야 합니다. 다음 (표2)는 일반적인 컴포넌트 품질 기준[5]을 SOA 단위로 발전시킨 서비스 및 복합 컴포넌트에 대한 기능 요건 이다.

특성	설명
상호운영성	이기종 환경의 상호운영성을 보장하여 플랫폼 독립적인 동작 필요
통합성	연동이 필요한 다양한 기술에 대한 인터페이스 보장 필요
컴포넌트 조합성	컴포넌트간의 조립이나 참조가 쉽도록 지원
Loosely-Coupling	메시지 기반 연계나 Annotation을 통하여 Coupling을 줄이도록 지원
추상화	제공 인터페이스가 기술적인 관점보다는 비즈니스 관점에서 의미가 있도록 추상화 수준 제공 필요

(표2) SOA 컴포넌트 요건

SOA Application은 ESB를 포함한 서비스들과의 효율적 연계가 필요하다. 이러한 분산 시스템들을 포함하는 공통의 컴포넌트 규격을 필요로 하게 되었다. SCA(Service Component Architecture)[6]는 현재의 CBD형태의 컴포넌트를 확장하여 다양한 통합 기술을 수용하면서, Loosely Coupling을 효율적으로 지원하는 프레임워크의 형태로 구성되어 있다. SCA는 IBM, BEA, Oracle, Sybase, SAP 등의 벤더들이 공동 참여하고 있는 새로운 컴포넌트 규격이다.

(그림6)에서 단위 서비스들은 SCA 표준에 맞추어 서로 연계 되어 모듈 단위의 복합 서비스를 구성하고 있다.

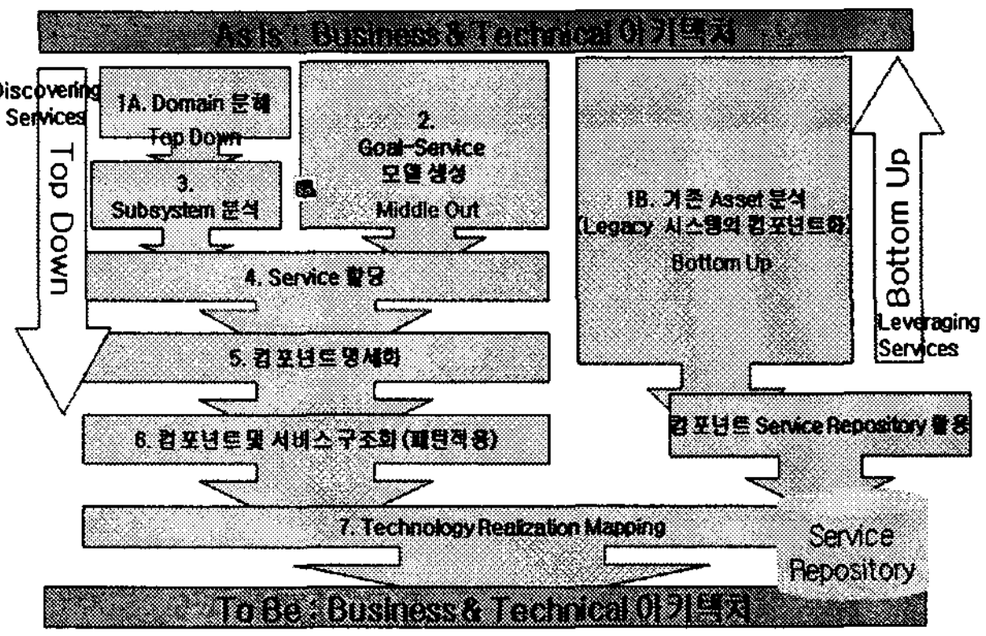


(그림6) SCA 컴포넌트 규격

4. CBD기반 SOA 구축 방안

4.1 SOA 방법론 활용

SOA 기반 시스템 구축을 위해서는 분설-설계-구현-테스트 및 인수에 이르는 적절한 방법론의 선정과 활용이 필요 하다. 전사적 관점의 SOA 방법론은 비전 수립과 프로세스 재설계와 같은 컨설팅 기법들을 포함하고 있으며, 설계 및 구축 단계는 CBD방법론의 Activity를 활용 하고 있다. 다음 (그림7)은 IBM의 SOMA방법론[7]의 전체 프로세스 이다.

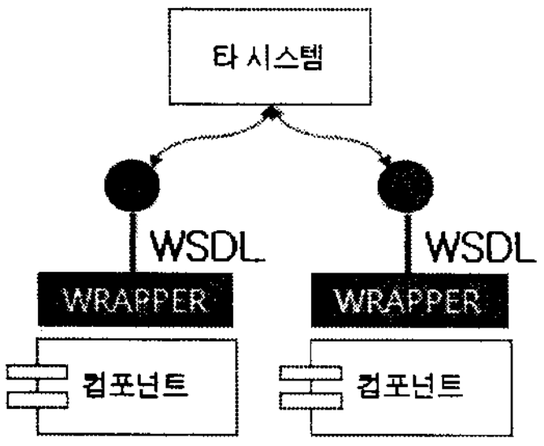


(그림7) IBM SOMA 방법론

개별적인 컨설팅 프로젝트가 병행되지 않을 경우, CBD방법론을 기반으로 하여 SOA도입을 위해 필요한 Activity를 추가하여 전체 방법론 프로세스를 Tailoring하여 활용 하는 방법이 적절 하다.

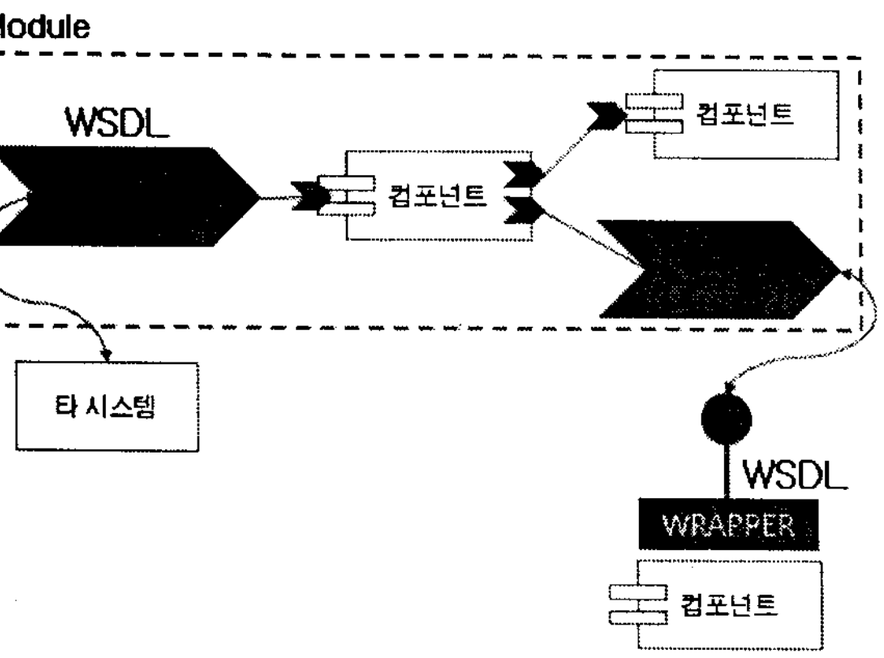
4.2 서비스 구축 방안 (Provider 측면)

컴포넌트를 활용하여 서비스를 구축하기 위한 방법은 CBD 컴포넌트 활용, ESB 기반 복합 컴포넌트 구축, SCA와 같은 SOA 컴포넌트 활용으로 나뉠 수 있다. (그림8)에서 CBD 컴포넌트의 경우 Wrapper를 활용하여 별도의 코딩 없이 웹 서비스 형태로 노출이 가능하다.



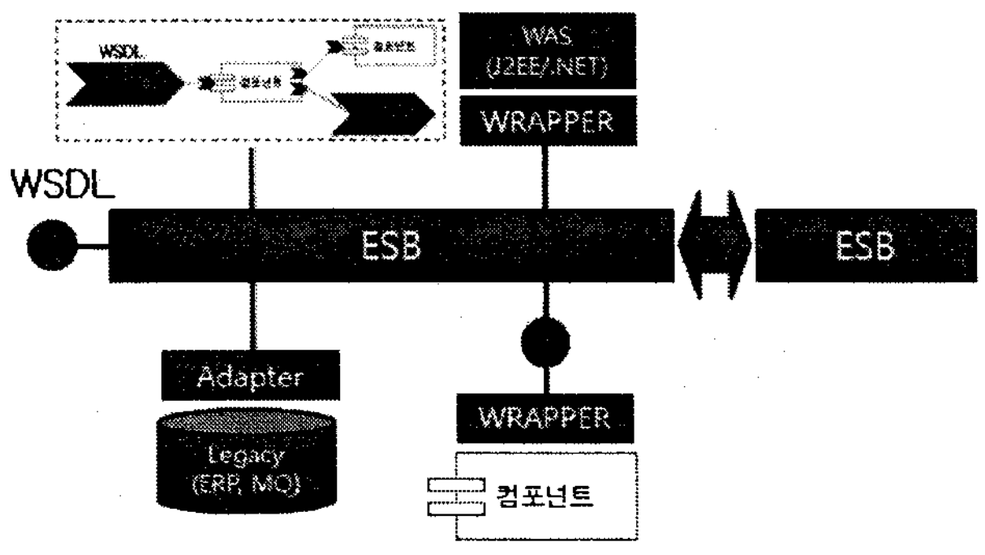
(그림8) 컴포넌트 기반 서비스

하나 이상의 컴포넌트를 활용하여 서비스를 구현하여야 하는 경우에는 별도의 컴포넌트 규격을 필요로 하게 된다. 다음 (그림9)는 SCA를 기반으로 서비스를 구현 방안 이다. 외부 서비스를 효율적으로 활용 할 수 있으며, 여러 컴포넌트를 Loosely Coupling하게 조립하여 활용이 가능 하다.



(그림9) SOA 컴포넌트 기반 서비스

내/외부 적으로 많은 시스템과의 연동이 필요하고, 이 기종의 기술과 프로토콜들의 변환이나 통합이 필요하며, 서비스의 체계적인 관리와 등록이 필요한 경우 ESB를 활용하여 서비스를 구현하는 것이 효율적이다. 다음 (그림10)은 ESB를 활용한 서비스 구현 방안 이다. 단위 컴포넌트를 활용한 서비스나 SOA 컴포넌트, Legacy, 다른 ESB 등을 활용한 다양한 통합과 관리를 지원 한다.



(그림10) ESB 활용 서비스 구현 방안

서비스 Provider를 구축하기 위한 방안을 결정할 경우에는 서비스의 특성, IT 환경의 특성을 고려하여 결정 하여야 하며, 세가지 모두를 동시에 활용하는 것도 가능 하다. Provider를 구축하는 방안에 대한 비교표는 다음 (표3)과 같다.

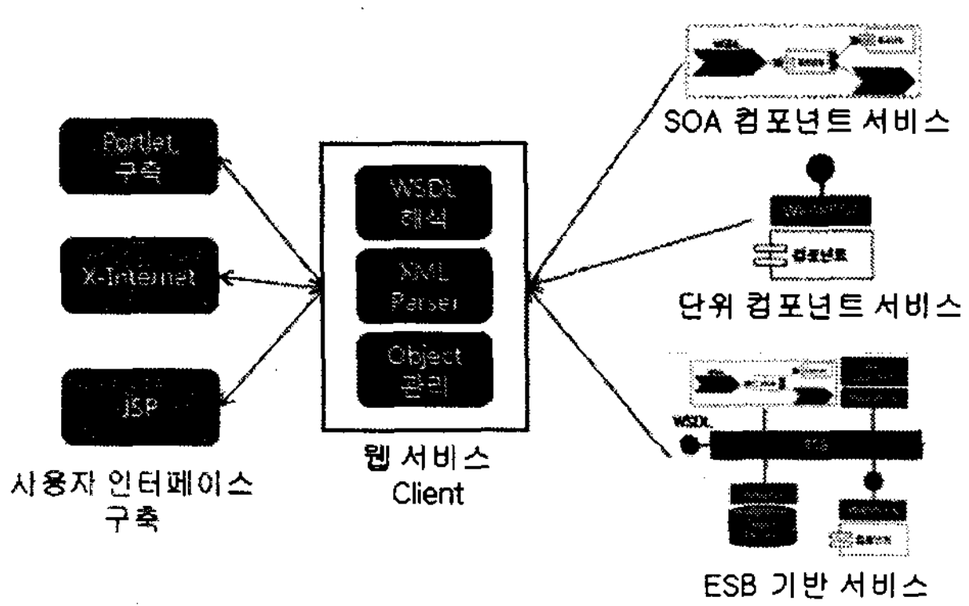
특성	단위 컴포넌트	SOA 컴포넌트	ESB 활용
컴포넌트	단일	복합	복합
구현	단순	복잡	복잡
Interface	P2P	P2P	단일화
통합	웹 서비스	다양	대부분 지원
서비스	10개 내외	10개 내외	50개 이상
장점	쉽게 구현	CBD지원 Local Binding가능	EAI지원

(표3) Provider 구축 방안 비교

4.3 서비스 활용 방안 (Consumer 측면)

구축된 서비스들은 모두 WSDL(Web Service Description Language)의 형태로 인터페이스가 노출되게 된다. WSDL에 인터페이스를 위한 모든 정보

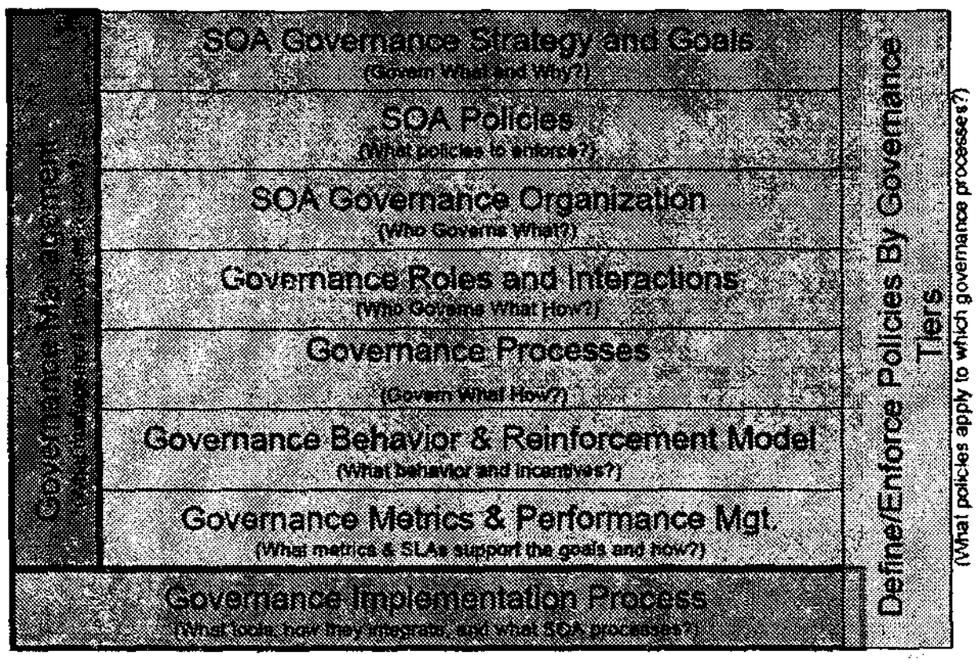
(In/Out 메시지, 지원 Operation, 서비스 위치)가 포함되어 있으므로, 웹 서비스 Client를 Eclipse WTP와 같은 툴을 활용하여 쉽게 자동화 하여 구축이 가능 하다. 구축된 웹 서비스 Client를 통해서 Portal, X-Internet, JSP와 같은 형태로 사용자 인터페이스를 구축하고 활용이 가능 하다. 다음 (그림 11)은 웹 서비스 Client형태로 구축하여 서비스를 활용하는 방안을 설명하고 있다.



(그림11) 서비스 활용 방안

4.4 SOA Governance 도입

SOA의 도입은 기존의 조직과 IT환경의 변화를 필요로 한다. IT환경에 적극적으로 대처하기 위하여, 새로운 조직과 업무 프로세스를 필요로 하게 된다. 서비스에 대한 비즈니스적인 목표와의 Align, 서비스 Life Cycle 관리, 효율적인 운영 프로세스, 서비스 및 아키텍처 담당 조직과 역할 등의 정책 수립과 실행이 필요하게 된다. (그림12)는 이러한 SOA Governance에 대한 참조 모델이다[8].



(그림12) SOA Governance 참조 모델

5. 결론

본 연구에서 CBD를 기반으로 SOA로 확장, 구축 하는 방안을 제시 하였다. SOA가 단지 새로운 통합 기술의 측면이 아니라, 비즈니스적인 관점에서 실제 IT의 가치를 높이고자 도입이 되고 있다.

성공적인 SOA의 구축을 위해서는 분명한 목표와 범위의 선정이 필요하며, 통합 아키텍처의 구축과 효율적인 컴포넌트의 활용이 필요 하다. 이를 달성하기 위하여 SOA 방법론을 활용한 핵심 서비스의 식별과 모델링, ESB를 활용한 서비스 구축, SCA를 활용한 SOA 컴포넌트의 활용이 필요하다. SOA를 통하여 정보시스템은 유연성과 민첩성을 확보하게 되고, 빠른 변화와 실 시간적인 처리가 중요한 현재의 IT환경에서 경쟁력을 갖추게 하는데 핵심적인 역할을 하게 될 것으로 전망 한다.

[참고문헌]

[1] Thomas Erl, "Service-Oriented Architecture: Concepts, Technology, and Design", Prentice Hall PTR, 2005.

[2] 임철홍, "기술과 응용 측면을 고려한 SOA 도입 방안", 경영과 컴퓨터, Aug, 2006.

[3] Capgemini, "Capgemini's Services Architecture Framework".

[4] David A. Chappell, "Enterprise Service Bus", O'Reilly, 2004.

[5] 김남희, 신석규, "소프트웨어 컴포넌트 시험 기술 동향", TTA저널 제88호.

[6] SCA Specification, "http://www.osoa.org/display/Main/SCA+Resources"

[7] IBM Redbooks, "Patterns: Service-Oriented Architecture and Web Services", 2004.

[8] AgilePath Corp., "Introducing AgilePath's SOA Governance Reference Model v. 1.0", 2006.