韓國IT서비스學會誌
第9卷 第2號
2010年 6月

서비스 온톨로지 기반 SOA 개발 방법론*

최고봉** · 박세권*** · 류승완****

SOA Development Methodology Based on Service Ontology*

Kobong Choi** · Seikwon Park*** · Sungwan Ryu****

■ Abstract ■

Service-Oriented Architecture (SOA) is a new trend for the design of application architecture to enhance the degree of share and reuse with the concept of service. It comes from the current situation where the business environments are changing rapidly and therefore competitions are getting bitter. To cope with such business challenges, business (and/or applications) architecture needs considerably flexibility and reusability, and that's why SOA is accepted as one of the most effective framework for the business applications in these days.

In this paper we propose an analysis and design methodology for the applications of SOA. To implement the proposed methodology, the service ontology needs to be defined first, and the tasks such as service profiling, workflow design and service orchestration need to be followed.

To validate the expected effects on flexibility and reusability, the proposed methodology was compared with CBD (Component Based Development).

Keyword : Service Oriented Architecture, SOA, Service Ontology, Ontology, Analysis and Design Methodology

논문투고일 : 2010년 04월 18일 논문수정완료일 : 2010년 06월 05일 논문게재확정일 : 2010년 06월 10일

* 본 연구는 중앙대학교 연구기금으로 이루어 졌음.

** 중앙대학교 대학원 정보시스템학과 석사과정

*** 중앙대학교 산업과학대학 정보시스템학과 교수, 교신저자

**** 중앙대학교 산업과학대학 정보시스템학과 교수

1. 서 론

최근 비즈니스 환경은 외부 변화 및 내부 요구사항에 대한 보다 빠른 대응을 요구하고 있다. 이는 기업의 성공적인 운영을 위해 예측되어진 변화뿐만 아니라 예측하지 못한 변화에 대해서도 빠르게 대응 할 수 있는 능력을 요구하고 있기 때문이다. 이에 따라 IT 환경 또한 기업의 판단에 따른 비즈니스 프로세스 변화에 빠르게 대처 할 수 있도록 요구하고 있으나 기존의 방식으로는 변화를 위한 많은 시간과 추가 비용이 발생 된다. 또한 “올바른 방향을 유지하는 동시에 신속하게 움직이는 것”이 중요시 되어 빠른 대응과 함께 올바른 대응이 요구되고 있으나 이를 기존의 방법론으로 대처하는 것은 한계가 존재한다. 이러한 비즈니스 환경 변화의 대응하려는 다양한 방안 중에 하나가 Service Oriented Architecture(이하 SOA)이다[1, 2].

SOA는 사용자 애플리케이션의 기능을 서비스 형식으로 전달하는 분산 시스템 구축의 한 접근 방법으로, 느슨한 결합(Loosely Coupled)[3]을 통해 유연성을 높이고 표준 인터페이스를 기반으로 하여 이기종 환경에서의 비즈니스 민첩성을 높일 수 있다[4]. 이에 따라 SOA를 활용하여 시스템을 개발하고자 하는 많은 노력이 있었으나 아직까지 SOA를 위한 표준 방법론이 존재하지 않아 개발 프로세스 관리와 어플리케이션의 체계적 품질관리가 현실적으로 어려운 실정이다.

기존의 구조적 방법론에서는 시스템을 기능(프로세스) 단위로, 객체 지향 방법론에서는 시스템을 객체 단위로, 컴포넌트 기반 방법론(Component Based Development(이하 CBD))은 설계 단위를 컴포넌트로 하고 있다. 그러나 구조적 방법론과 객체지향 방법론은 프로세스 변화에 대한 신속한 대응이 어렵고 컴포넌트 기반 방법론은 어플리케이션을 위한 컴포넌트의 의미론적(Semantic) 통합에 어려움이 있어 최근의 환경변화에 부합하지 못하는 문제를 가지고 있다. SOA는 이러한 문제들을 해결할 수 있는 대안의 하나로써 시스템을 서비스 단위로 구분하여 식별하고 개발하는 방법론으로, 서비스에 대한 식별과 조합이 중요한 이슈이다.

〈표 1〉 방법론별 시스템 단위 관점

방법론	시스템 단위
구조적 방법론	기능
객체지향 방법론	객체
CBD	컴포넌트
SOA	서비스

본 논문에서는 온톨로지를 이용하여 서비스를 정의하고 식별함으로써 서비스의 재사용성과 유연성을 높일 수 있는 방법론을 제안하고자 한다.

본 논문의 구성은 다음과 같다. 제 2장에서는 SOA의 개요와 온톨로지 개요를 간단히 살펴보고 온톨로지와 SOA의 연관성에 대하여 설명하였다. 제 3장에서는 제안하는 방법론의 절차를 설명하고 제 4장에서는 본 방법론의 검증을 통한 분석결과를 정리하였다. 제 5장은 결론에 해당하는 장으로, 본 논문을 요약하였다.

2. 관련연구

2.1 SOA 개요

SOA란 기존 애플리케이션의 기능을 비즈니스적인 의미가 있는 기능 단위로 묶어 표준화된 호출 인터페이스를 통해 서비스라는 소프트웨어 컴포넌트 단위로 재조합한 후, 이 서비스들을 서로 조합하여 업무 기능을 구현한 애플리케이션으로 만들어내는 소프트웨어 아키텍처이다[5, 6].

서비스는 특정 기술이나 플랫폼에 종속되지 않는 표준 인터페이스 기반의 기능을 갖는 단위로써, 주요 특징으로는 느슨한 결합, 조합성, 조밀성 그리고 발견 가능성이 있다. 또한, 서비스 공급자, 서비스 요청자, 서비스 저장소라는 세 요소가 유기적으로 연계되어 있는 특징을 갖는다[7, 8].

서비스 단위로 시스템을 구성하기 때문에 기존의 구조적 방법론, 객체지향 방법론 그리고 CBD 방

법론에서 사용된 방법론들은 서비스 단위 식별이나 설계에 적용 될 수 없으므로 SOA를 위한 새로운 방법론들이 제안되고 있으며 주요 SOA 방법론을 살펴보면 다음과 같다.

SODA(Service Oriented Development of Application)는 서비스를 설계하고 개발 및 조합하는 일련의 과정에 대한 원칙과 방식을 제시하고 있으나 서비스 단위나 세부 절차에 대한 구체적인 방법은 언급되지 않았다[9].

SOAD(Service-Oriented Analysis and Design)는 OOAD와 EA프레임워크, BPM과 같은 기존의 모델링과 개념을 SOA에 적용시킨 방법론으로, 부분적으로 SOA 방법론을 지원하지만 추가적인 활동과 산출물이 필요하다[10].

SOUP(Service Oriented Unified Process)는 RUP와 XP의 특징을 모아서 SOA에 적용시킨 방법론으로써 6단계로 정의하였으나 비즈니스 프로세스, 서비스 규약, 서비스 저장소에 대한 요구사항을 지원하지 않는다[11].

SOMA(Service-Oriented Modeling and Architecture)는 SODA의 확장된 방법론으로써 IBM이 제안하였다. 이 방법론은 모델링, 분석, 설계 기술 및 활동을 포함하고 있으나 서비스 식별의 구체적인 방법을 명시하고 있지 않다[12].

2.2 온톨로지(Ontology)

온톨로지란 용어 사이의 관계를 정의하는 일종의 사전과 같은 것으로 Gruber의 정의에 의하면 “온톨로지란 공유된 개념에 대한 정형화된 명세이다.”

온톨로지를 SOA에 적용시킴으로써 얻을 수 있는 장점으로는 서비스의 정의와 식별의 유연성과 보다 명확한 서비스 정의가 가능하다는 것이다[13].

3. 제안하는 개발방법론

3.1 서비스 온톨로지

SOA의 적용을 위해서는 서비스를 정의하고 식별해야 하는데, 이를 위해 본 논문에서는 온톨로지의 개념을 이용하여 [그림 1]의 서비스 온톨로지를 제안한다.

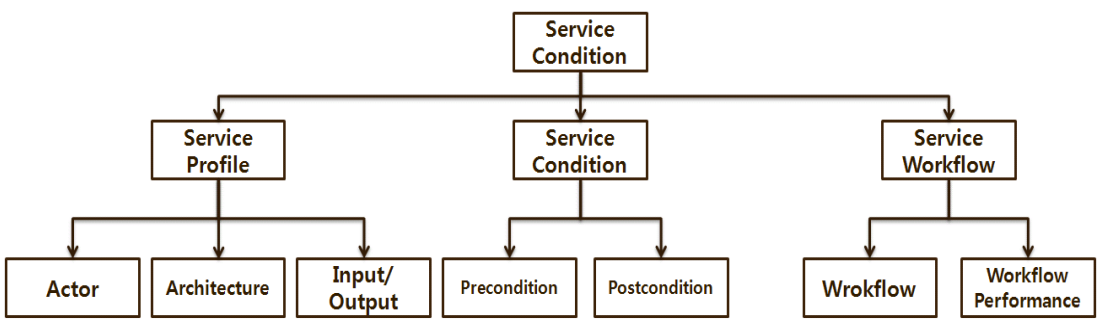
본 논문에서 제안하는 서비스 온톨로지의 서비스는 서비스 프로파일, 서비스 컨디션 그리고 서비스 워크플로우로 구성되는데 각각에 대한 설명은 다음과 같다.

3.1.1 서비스 프로파일(Service Profile)

서비스 프로파일은 서비스와 서비스를 구분하고 식별하는 요소로써 서비스를 설명할 수 있는 요소를 담고 있다. 서비스 프로파일의 구성요소는 액터와 입출력 데이터, 아키텍처이다.

액터는 서비스를 제공하는 제공자와 서비스를 소비하는 소비자 등 서비스에 참여하는 객체의 정보를 담고 있는 부분이다.

입출력 데이터(Input/output Data)는 서비스가 가



[그림 1] 제안하는 서비스 온톨로지(Service Ontology)

진 기능을 실현하기 위해 입력 데이터를 서비스 외부로부터 받고, 서비스의 기능이 실현된 후 출력 데이터를 생성하게 된다. 즉 서비스 사이에 주고받는 데이터들에 대한 정보를 담고 있는 부분이다.

아키텍처(Architecture)는 서비스 실현을 위하여 필요한 기반 기술들의 정보를 담고 있는 것으로 기반 기술들로는 DB, 네트워크, 운영체제, 플랫폼, 그리고 메시징 기술로 구분된다.

3.1.2 서비스 컨디션(Service Condition)

서비스는 실행되기 위한 선행 조건이 존재하며 실행된 이후에도 서비스를 종료하기 위한 조건이 존재하는데, 서비스 컨디션에서는 이러한 서비스 실행과 종료를 위한 조건의 정보를 담고 있는 부분으로써 서비스의 시작과 끝을 표현 한다.

서비스 컨디션의 정보는 서비스와 서비스를 구분하는 기준이 되며 서비스 검색 시 서비스 선택의 조건으로 이용된다.

3.1.3 서비스 워크플로우(Service Workflow)

서비스 워크플로우는 서비스를 구성하고 있는 태스크들의 조합으로써 서비스 내의 태스크들의 순차를 담고 있다. 서비스 워크플로우는 워크플로우와 워크플로우 퍼포먼스로 구분되는데 워크플로우는 태스크의 실행 순서를 플로우 다이어그램 형태로 저장한다.

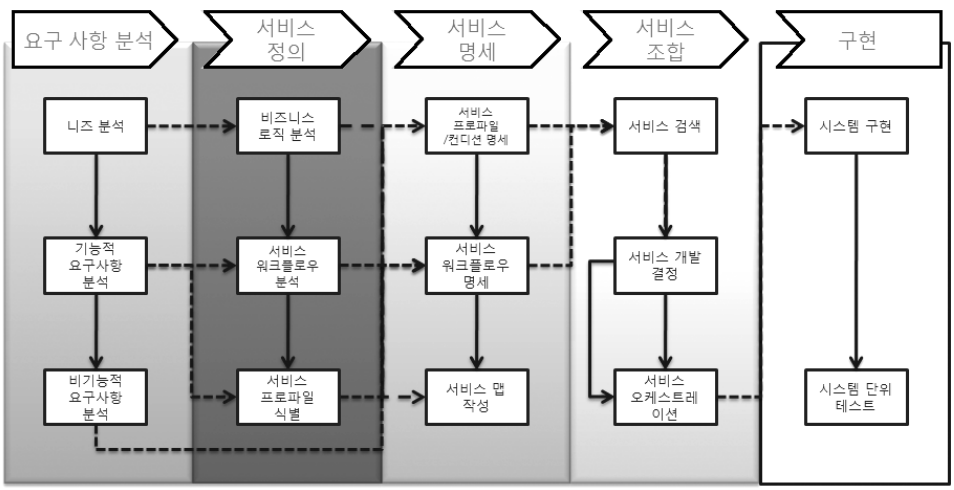
워크플로우 퍼포먼스는 서비스의 비기능적 요구사항을 만족시키기 위하여 각 태스크들이 가져야 할 태스크 품질의 정보를 담고 있는데 응답속도, 신뢰도 등이 이 부분에 해당된다.

3.2 제안하는 개발 방법론 프로세스

본 논문에서 제안한 SOA 개발 방법론은 [그림 2]와 같이 크게 요구사항분석, 서비스 정의, 서비스 명세, 서비스 조합 그리고 구현의 5단계로 이루어지고 각 단계별 산출물로 구성되며, 각 단계는 세부 태스크들로 구성 된다. 세부 태스크들은 각 단계의 목표를 달성하기 위하여 행해지는 일련의 활동들이다.

3.2.1 요구사항 분석 단계

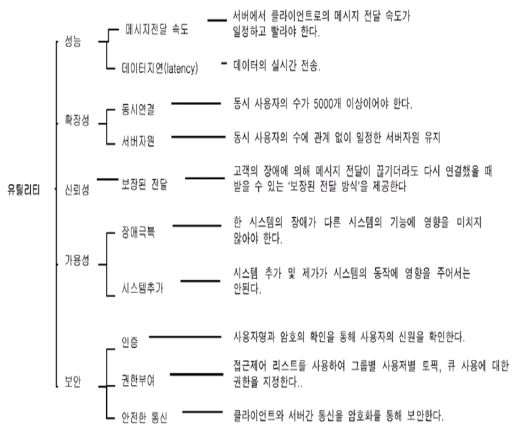
요구사항 분석 단계에서는 개발 시스템의 이해당사자를 식별하여 그들의 니즈들로부터 요구사항을 추출하는 단계이다. 요구사항 분석을 위해서 유즈케이스 다이어그램(Use Case Diagram) 과 유즈케이스 명세서(Use Case Specification)를 작성하게 된다.



[그림 2] 제안하는 방법론 프로세스

요구사항 수집 시 이해당사자들로부터 각 요구사항에 대한 선행조건과 사후조건에 대한 정보를 획득하여 유즈케이스 명세서에 작성한다. 이 정보는 다음 단계에서 수행 하게 되는 비즈니스 로직 분석에서 서비스와 서비스를 구분하는 기준을 제공한다.

또한 추출된 요구사항은 기능적 요구사항과 시스템 요구사항으로 구분하여 추출하게 된다. 시스템 요구사항은 비기능적 요구사항들을 만족시키기 위한 시스템 구성요소 기술수준 결정에 사용되는데, 시스템 요구사항 정의에는 [그림 3]의 유틸리티 트리(Utility Tree)를 이용한다.



[그림 3] 유틸리티 트리 예[14]

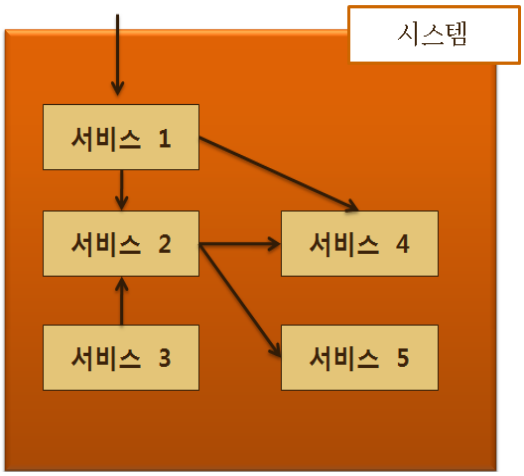
3.2.2 서비스 정의 단계

서비스 정의 단계에서는 시스템의 비즈니스 로직을 분석하여 시스템을 구성하는 서비스를 식별하고 서비스를 정의하는 단계이다.

- 비즈니스 로직 분석

비즈니스 로직은 시퀀스 다이어그램(Sequence Diagram)을 이용하여 운영 시나리오(Operation Scenario)를 정의한다. 이 때 요구사항 분석 단계에서 작성한 유스케이스 명세서와 운영 시나리오를 바탕으로 서비스를 식별하게 되는데 서비스 식별 기준은 기능을 수행하는 단위로써 동일한 서비스 컨디션별로 식별한다.

예를 들어 쇼핑몰의 경우 쇼핑몰 시스템을 구성하는 서비스는 로그인, 결제, 장바구니 등의 서비스들로 식별 할 수 있다. [그림 4]는 이 단계의 산출물로써 서비스들 간의 관계 정보를 표현한다. 또한 해당 태스크가 완료되면 서비스 온톨로지의 서비스 컨디션이 정의된다.



[그림 4] 서비스 관계도

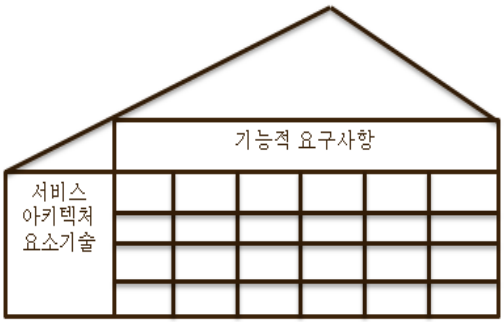
- 서비스 워크플로우 분석

식별된 서비스는 해당 기능을 실행하기 위해 어떤 워크플로우를 가지게 되는지를 식별하는 단계이다. 워크플로우 식별은 이전 태스크에 정의한 운영 시나리오를 이용하여 정의하게 된다. 식별된 서비스 워크플로우는 서비스 명세 단계에 사용된다. 또한 이 단계에서 워크플로우의 성능을 결정하는데 시스템 요구사항을 만족시킬 수 있도록 워크플로우들의 성능을 정의한다.

- 서비스 프로파일 식별

본 태스크는 서비스에 참여하는 액터와 서비스에 사용되는 입력데이터, 서비스를 실행함으로써 산출되는 출력데이터, 그리고 서비스를 구성하는 아키텍처를 식별한다. 서비스 아키텍처 식별에서는 [그림 5]와 같은 HoQ를 이용하는데 HoQ(House

of Quality)는 기능적 요구사항을 만족시키기 위해 필요한 요소기술을 식별한다.



[그림 5] 아키텍처 식별 HoQ

이 단계에서는 기능적 요구사항을 만족시키기 위하여 특정 요소기술이 필요한지 아닌지의 여부만을 식별하게 되는데, 예를 들어 로그인, 결제 그리고 장바구니 서비스 등을 구현하기 위해서는 DB, 네트워크, 운영체제, 플랫폼, 그리고 메시징 기술 중 어떤 기술이 필요한지를 식별한다. 이 단계가 완료되면 서비스 온톨로지의 서비스 프로파일을 정의하게 된다.

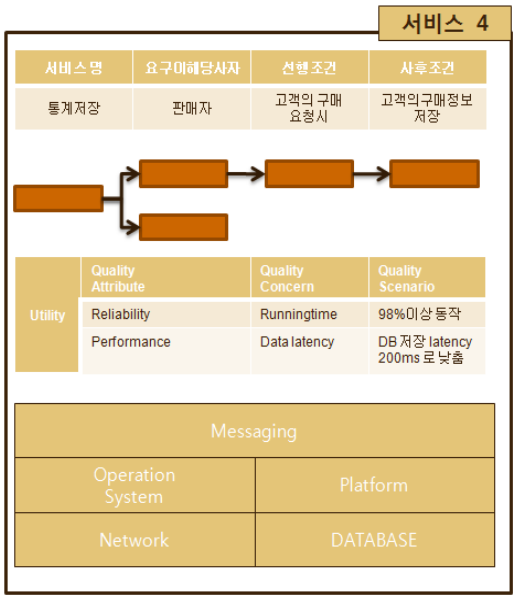
3.2.3 서비스 명세 단계

이 단계에서는 지금까지 정의한 서비스 온톨로지의 각 요소에 대하여 정형화된 형태로 명세한다. 이 단계에서 서비스 요소기술들의 명세를 위해 다른 형태의 HoQ를 이용하게 되는데 [그림 5]의 HoQ에서 기능적 요구사항을 대신하여 시스템 요구사항을 이용하고, [그림 5]에서 식별된 서비스 아키텍처 요소기술 각각에 대해서 시스템 요구사항을 만족할 수 있는 성능을 식별하고 이를 명세한다.

[그림 6]은 서비스 명세의 산출물이다. 완료된 서비스 명세는 각각의 서비스와 맵핑하여 서비스 맵을 만들고 이는 서비스 조합에서 서비스 선택의 기준이 된다.

3.2.4 서비스 조합 단계

서비스 조합 단계에서는 앞서 정의한 서비스 맵을



[그림 6] 서비스 명세 산출물

기준으로 서비스를 검색하며, 조건에 맞는 서비스가 존재하지 않는 경우 개발을 결정하여 서비스 오케스트레이션(Orchestration)을 완성하게 된다.

이 단계에서의 서비스 검색 기준은 선행조건과 사후조건, 그리고 입출력 데이터이다. 또한 검색된 서비스는 시스템 요구사항들의 우선순위를 정하고 그것을 만족시키는 서비스를 선택하게 된다.

조건을 만족시키는 서비스가 발견되지 않을 경우에는 서비스의 명세대로 새로운 서비스를 설계하게 되며 검색된 서비스와 새로이 설계된 서비스들을 조합하여 서비스 오케스트레이션을 완성하게 된다.

서비스 오케스트레이션에서는 논리적인 서비스 연합(Service Federation)과 호출순서, 에러처리 등의 제어를 포함한다.

3.2.5 구현 단계

구현단계에서는 앞서 정의한 서비스 오케스트레이션을 토대로 새롭게 설계된 서비스들을 개발하고 서비스들을 인터페이스로 연결하여 시스템을 구현한다. 구현이 완료되면 시스템 단위 테스트를

통하여 초기 요구사항과 서비스 워크플로우의 퍼포먼스 기준을 만족하는지를 확인한다.

4. 제안하는 방법론의 평가

제안하는 방법론의 효용성 평가를 위해 기존의 방법론 중 SOA와 가장 유사한 CBD 방법론과 비교하였다. 평가단은 S기업(매출액 2조 원 이상, 종업원 5000명 이상) 근무자 중 경력 10년 이상의 전문가들로 구성하였으며 평가항목은 개발기간, 비용, 고객만족, 재사용성의 4가지 항목에 대해 7점 척도로 평가하였다. 또한 개발방법론을 대형, 중형, 소형 시스템 개발로 구분하여 각각을 평가 하였다.

시스템 규모의 분류는 대형 시스템의 경우 전사적 차원의 시스템을 의미하는 것이며, 중형 시스템은 2~3개의 업무 연관을 가지는 부서에서 사용되는 시스템, 그리고 소형 시스템의 경우 단일 부서에서 사용되는 시스템을 의미한다.

〈표 2〉 시스템 분류

구 분	설 명
대형시스템	전사적 차원
중형시스템	2~3개의 연관 부서
소형시스템	단일 부서

평가는 전문가 채점 결과의 평균을 비교 한 것과 AHP 가중치를 부여하여 계산 한 것, 두 가지 형태로 실시하였다. 대형, 중형, 소형 시스템에 대한 평가 결과는 차례로 <표 3>~<표 5>와 같다.

평가 결과 제안하는 방법론이 기존 CBD 방법론보다 대체로 더 좋은 것으로 나타났다. 이러한 결과의 원인으로는 CBD의 단점에서 찾을 수 있다.

CBD에서는 시스템 개발의 요소를 표준적인 부품들로 보고 필요한 부품을 조립하듯이 개발하는 방법론이다. 하지만 CBD의 개발 사상은 동일 시스템에서의 중복 개발을 줄일 수 있었으나 이기종 시스템간의 연계를 지원하지 않아 재사용의 한계가 존재한다. 또한 시스템 요소를 하위 단계까지

〈표 3〉 대형 시스템에서의 평가 결과

구 분	평균		가중 평균		
	CBD	제안하는 방법	가중치	CBD	제안하는 방법
개발 기간	5.8	6.4	0.382	2.22	2.45
비용	5.2	5.4	0.206	1.07	1.11
고객 만족	5	6	0.346	1.73	2.08
재사 용성	5.2	6.2	0.066	0.34	0,41
합계	5.3	6		5.36	6.05

〈표 4〉 중형 시스템에서의 평가 결과

구 분	평균		가중 평균		
	CBD	제안하는 방법	가중치	CBD	제안하는 방법
개발 기간	4.4	5.6	0.382	1.69	2.14
비용	4	5	0.206	0.82	1.03
고객 만족	4.2	5.8	0.346	1.45	2.01
재사 용성	5.2	5.8	0.066	0.34	0.38
합계	4.45	5.55		4.31	5.56

〈표 5〉 소형 시스템에서의 평가 결과

구 분	평균		가중 평균		
	CBD	제안하는 방법	가중치	CBD	제안하는 방법
개발 기간	3.4	4	0.382	1.30	1.53
비용	3	3.4	0.206	0.62	0.70
고객 만족	4.4	5.2	0.346	1.52	1.80
재사 용성	4.4	4.6	0.066	0.29	0.30
합계	3.8	4.3		3.73	4.33

정의하여 개발 기간과 개발 비용이 증가하는 단점이 있다. 이에 반해 본 연구에서 제안한 방법론은 표준인터페이스 기반으로 서비스를 정의함으로써

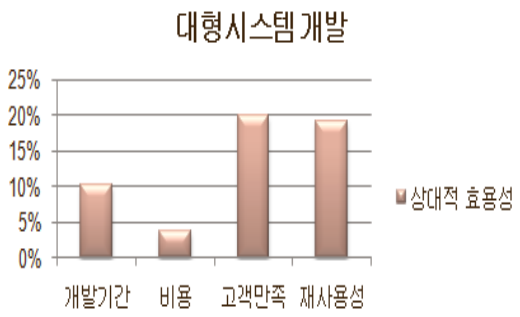
이기종 시스템 간의 상호운영을 지원하며, 시스템의 구성요소가 컴포넌트보다 큰 서비스를 이용함으로써 시스템을 정의하는 기간과 비용을 단축할 수 있다.

제안한 방법론의 효용성의 절대적인 평가는 대형 시스템 개발 시 가장 높은 것으로 나타났으며 중형 시스템, 소형 시스템 순이었다. 대형 시스템에서의 절대적 가치가 더 큰 것은 최근 시스템이 복잡하고 다양해짐에 따라 이기종간 연동의 필요성이 급증하고, 비즈니스 환경이 급변함에 따라 재사용성의 중요성이 증가하였기 때문이다. 따라서 전통적인 구조적 방법론보다는 CBD, SOA와 같이 재사용성을 추구하는 개발방법론을 대형 시스템에 적용하여 개발하는 것이 적합함을 보여주는 것으로 사료된다. 또한 CBD는 동종 플랫폼에서 제한적으로 재사용할 수 있는 것에 비하여 SOA는 이기종 플랫폼간의 연계를 지원하는 특성으로 인해 대형 시스템 개발 시 CBD 방법론보다 제안한 방법론의 평가점수 절대 값이 더 높이 나타난 것으로 추정된다.

이에 반해 소형 시스템 개발에서는 CBD와 제안하는 방법론이 다른 규모의 시스템 개발보다 비교적 효용성이 낮은 것으로 평가됐다. 특히 CBD는 7점 척도의 평가에서 평균 3점대의 점수를 획득하는데 그쳤는데, 이는 상대적으로 재사용성 요소가 다른 규모의 시스템보다 적어 개발 기간과 비용의 증가가 시스템 개발에 부담으로 작용한 결과로 유추해 볼 수 있다. 이러한 부담은 본 연구에서 제안한 방법론에서도 동일하게 적용되지만 평가를 통하여 CBD보다 부담을 완화할 수 있음을 보여 주고 있다.

상대적 효용성의 크기는 가중 평균을 이용하여 계산하였을 때 대형 12.8%, 중형 29.2%, 소형 16.1%로 나타났다.

절대적 수치가 가장 높은 대형 시스템개발에서 상대적 효용성이 낮은 것은 기존의 CBD 방법론이 현재 대형 시스템 개발에 효율이 높은 것으로 인정되고 있어 제안한 방법론이 CBD 방법론보다 높은 점수를 획득하였지만 한계효율(Marginal Efficiency)이 적은 것으로 판단된다. 세부 항목별 점수는 비교는 [그림 7]과 같다.



[그림 7] 대형 시스템 개발시상대적 효용성

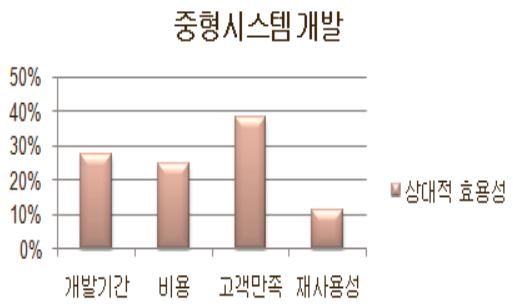
세부 항목별 점수를 보면 고객만족과 재사용성에서 상대적으로 높은 점수를 받은 것으로 나타나는데 재사용성이 높은 것은 이기종간의 연계를 지원하지 않는 CBD보다 이를 지원하는 SOA가 재사용될 확률이 더 높기 때문인 것으로 판단된다. 개발 비용측면에서는 CBD와 제안하는 방법이 큰 차이가 없는 것으로 평가되었다.

중형 시스템 개발에서는 상대적 효용성이 CBD보다 30% 정도 높게 평가 되었다. 이는 CBD가 소형 시스템과 중형 시스템에서 비교적 낮은 점수로 평가된 반면에 제안한 방법론은 중형 시스템에서 높은 점수를 획득하였기 때문인데 세부 항목별 상대적 효용성은 [그림 8]과 같다.

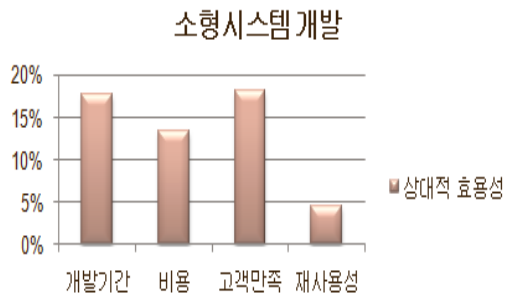
중형 시스템에서 상대적 효용성이 높은 부분은 개발기간과 비용부분인데 이는 CBD 방법론으로 시스템을 개발하기 위해서는 전체 시스템의 세부적인 사항을 모두 분석하여 아키텍처 모형을 만들어야하는 것으로 인해 개발 기간과 비용이 증가되는 단점이 있지만 제안한 방법론은 비즈니스 프로세스 모델을 만들어 서비스 단위의 재사용이 가능하므로 개발기간과 비용 면에서 CBD 방법론 보다 효율적으로 평가된 것으로 판단된다.

소형 시스템 개발 시 세부 항목별 상대적 효용성은 [그림 9]와 같은데 소형 시스템에서도 상대적

효용성이 높게 평가 된 것은 개발 기간과 비용인 것으로 볼 때 중형 시스템과 그 맥락을 같이하는 것으로 판단된다.



[그림 8] 중형 시스템의 개발시 상대적 효용성



[그림 9] 소형 시스템 개발시 상대적 효용성

마지막으로 상대적 효용성 중 모든 시스템에서의 상대적 효용성이 매우 높게 나타난 고객만족의 경우 기존 CBD는 컴포넌트 단위의 재사용으로 인하여 시스템의 성능에 대해서 정량적 측정이 어려워 개발 이후 고객만족이 낮았는데, 제안한 방법론은 고객의 요구사항을 정량적으로 변환하여 요구조건을 만족시키는 서비스 단위로 재사용하기 때문에 시스템 개발 시 고객의 요구 사항과 일치된 시스템 개발이 가능하여 고개만족에서 상대적 효용성이 높게 평가된 것으로 판단된다.

5. 결 론

SOA는 급변하는 비즈니스 환경에서 기업의 민첩한 대응을 가능케 하는 새로운 IT 전략의 대안으로, 기업의 기존 자산의 활용이나, 비용 절감 등의 효과를 줄 것으로 기대를 모으고 있다.

그러나 서비스 기반의 개발을 위한 표준 방법론이 존재하지 않아 서비스 재사용율이 낮으며 SOA에 대한 개발 경험을 축적할 수 없었다. 또한 SOA를 이용한 정보시스템 개발 시 현재 시스템 개발의 진척도를 가늠할 수 없어 프로젝트 관리에 어려움이 존재 하였다. 이에 본 논문에서는 온톨로지를 이용하여 서비스를 식별하고, 서비스를 기반으로 시스템을 개발하는 SOA 방법론을 제안하였다.

제안한 방법론에서는 서비스를 온톨로지를 정의하고 그를 이용하여 서비스를 식별하여 서비스를 구성함으로써 서비스 오케스트레이션을 완성하였다.

제안한 방법론을 기존의 CBD 방법론과 효율성 측면에서 비교하였는데 비교 결과 대형 시스템 개발에서는 이기종 간 연계를 지원하는 장점으로 인하여 재사용성이 높아졌으며, 중·소형 시스템 개발에서는 서비스 단위 재사용으로 인하여 개발기간과 비용의 면에서 많은 개선의 효과가 있는 것으로 나타났다. 또한 고객만족 지표에서는 모든 시스템 개발에서 높은 점수를 획득하였는데 이 또한 서비스 단위의 재사용으로 인하여 정량적인 요구사항의 만족여부를 평가하여 재사용하기에 품질에 대한 만족이 높은 것으로 평가되었다.

본 논문에서는 서비스 식별에 온톨로지를 적용함으로써 서비스의 유연성을 높여 재사용성을 향상시킬 수 있는 방법을 제시하였는데, 이는 어플리케이션 개발 방법론에서 의미론적 접근에 대한 방향성을 제공할 수 있을 것으로 기대된다.

참 고 문 헌

[1] 고현희, 궁상환, 박재년, “아키텍처 기반 설계 방식에 대한 평가기능이 통합된 소프트웨어 설계 방법론”, 정보과학회논문지, 소프트웨어 및 응용, 제34권, 제7호(2007).

[2] Arsanjani, A., “Service Oriented Modeling and Architecture(SOMA)”, IBM Developer

Works, 2004.

[3] Brown, A., S. Johnston, and K. Kelly, “Using Service Oriented Architecture and Component Based Development to Build Web Service Applications”, Rational Software Corporation, 2002.

[4] Channabasavaiah, K., K. Holley, and E. M. Tuggle, “Migrating To a Service Oriented Architecture, Part 1”, IBM Whitepaper, 2003.

[5] Chung, S., P. Young, and J. Nelson, “Service Oriented Software Reengineering : Bertie3 as Web Services”, Proceedings of the 2005 IEEE International Conference on Web Services, IEEE Computer Society, 2005.

[6] Jammes, F. and H. Smit, “Service Oriented Paradigms in Industrial Automation”, *IEEE Transactions on Industrial Informatics*, Vol. 1, No.1(2005), pp.62-70.

[7] Korotkiy, M. and J. Top, “Onto-SOA : From Ontology-enabled SOA to Service-enabled Ontologies”, International Conference on Internet and Web Application and Services (ICIW), Guadeloupe, 2006.

[8] Korotkiy, M. and J. Top, “From Ontology Enabled SOA to Service-enabled Ontologies”, Telecomunications, 2006.

[9] Lewis, G., E. Morris, L. O’Brien, D. Smith, and L. Wrage, “SMART : The Service Oriented Migration and Reuse Technique”, Software Engineering Institute, Carnegie Mellon University, 2005.

[10] Mittal, K., “Service Oriented Unified Process (SOUP)”, available from http://www.kunalmittal.com/html/soup.shtml, 2006.

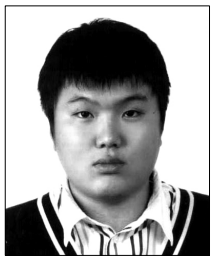
[11] Niblett, P. and S. Graham, “Events and Service Oriented Architectures : The OASIS Web Services Notification Specifications”, *IBM Systems Journal*, Vol.44, No.4(2005).

[12] Plummer, D., “Service Oriented Development Applications : SODA Pops, Gartner’s Internet Strategies Commentary”, COM-129640, 2001.

[13] Thomas, E., “Service Oriented Architecture (SOA) : Concepts, Technology, and Design”, Prentice Hall, 2005.

[14] Zimmermann, O., P. Krogdahl, and C. Gee, “Elements of Service Oriented Analysis and Design”, IBM Developer Works, 2004.

◈ 저 자 소 개 ◈



최 고 봉 (ca_ckb@hanmail.net)

중앙대학교 정보시스템학과에서 2008년 정보학사를 취득하였다. 현재는 중앙대학교 정보시스템학과 석사과정 재학 중이다. 주요 연구 관심분야는 SOA, 서비스, 정보시스템 개발방법론, 온톨로지 등이다.

박 세 권 (psk3193@cau.ac.kr)

서울대학교 공과대학과 대학원 산업공학과에서 1978년과 1981년에 공학사(BS)와 공학 석사(MS)를 취득하였으며, Texas A&M 대학교 대학원 산업공학과에서 1985년에 산업공학공학 박사(Ph.D.)를 취득하였다. 1978년부터 1981년까지 한국전자통신연구소에서 연구원으로 근무하였으며, 1985년부터 1987년까지 한국전자통신연구원 통신망계획부에서 선임연구원으로, 1987년부터 1990년까지 농촌경제연구원에서 농림수산부 소프트웨어하우스 실장(수석연구원)으로 농업농촌정보화 하부구조 구축 연구를 수행하였다. 1990년부터 현재까지 중앙대학교 정보시스템학과에 재직중이며 연구 관심분야는 시스템공학 등이다.

류 승 완 (ryu@cau.ac.kr)

고려대학교 산업공학과에서 1988년과 1991년에 각각 공학사와 공학 석사를 취득하였으며, 뉴욕주립대(SUNY at Buffalo) 산업공학과에서 2003년에 공학박사를 취득하였다. 1991년부터 1993년까지 LG전자 영상미디어연구소에서 주임연구원으로 근무하였으며, 1993년부터 2004년까지 한국전자통신연구원 이동통신연구단에서 선임연구원으로 근무하면서 2세대 CDMA 디지털 이동통신, 3세대 이동통신 IMT-2000, 4세대 이동통신 시스템 연구를 수행하였다. 2004년부터 중앙대학교 정보시스템학과에 부임하여 정보통신관련 과목을 담당하고 있으며, 2004년부터 2007까지 한국전자통신연구원 이동통신연구단에서 초빙연구원으로 이동통신 연구를 수행하였다. 주요 연구 관심 분야는 이동통신시스템 설계 및 성능분석, 무선 MAC 프로토콜, 차세대 이동통신 서비스 및 비즈니스 모델 개발 등이다.