

# 서비스지향 개발 프로세스의 기능기반 모델에 대한 연구

유춘근<sup>\*</sup>, 임철우<sup>\*\*</sup>, 강병욱<sup>\*\*\*</sup>

## 요 약

서비스 지향 아키텍처(Service Oriented Architecture, 이하 SOA)는 독립적으로 실행 가능하며, 외부 인터페이스를 통해서만 접근 가능한 서비스를 재사용 단위로 하는 재사용 패러다임이다. 이러한 SOA를 적용하여 시스템을 개발하기 위한 노력이 지속적으로 시도 되고는 있지만, SOA를 기반으로 하는 표준화된 방법론은 아직 정립되지 않고 있다. 그러므로 SOA를 이용한 시스템 개발은 현재 많은 어려움에 직면하고 있다. 본 논문에서는 레거시 시스템을 지원하기 위한 기능기반 모델의 서비스 지향 개발방법론을 제안한다. 그리고 기존의 제안된 SOA 방법론의 분석을 통하여 레거시 시스템에 적용 가능한 기능 중심의 재사용성을 정형화된 방법을 제안함으로써 서비스 재활용도를 높일 수 있는 가이드라인을 제공한다.

## A Study of The Function-Oriented Model for SOA Development Process

Chungun Yu<sup>\*</sup>, Chulwoo Rhim<sup>\*\*</sup>, ByungWook Kang<sup>\*\*\*</sup>

## ABSTRACT

Service-oriented architecture(SOA) is a paradigm where reusable units are independently executable and only accessible through outer interfaces. Standardized methodology based on SOA has not been decided yet although a lot of efforts have been made for systems using SOA. Therefore Development of system using SOA is facing many difficulties currently. In this paper, service oriented methodology of function based model for supporting legacy system is proposed. Standardized methods for function based reusability through analysing SOA methodology are proposed and guide lines for improvement of service reusability are presented.

**Key words:** Service Oriented Architecture(서비스지향 아키텍처), Legacy System(레거시 시스템), Development Process(개발프로세스)

## 1. 서 론

최근 비즈니스 환경은 외부의 변화 및 내부의 요구사항에 대해 보다 빠른 대응을 요구하고 있다. 이는 기업의 성공적인 운영을 위해 예측되어진 변화뿐만 아니라 예측하지 못한 변화에 대해서도 빠르게 대응할 수 있는 능력을 요구하고 있기 때문이다. 이

에 따라 IT환경 또한 기업의 판단에 따라서 비즈니스 프로세스 변화가 빠르게 수행되어야 하지만 기존의 방식으로는 많은 시간과 추가비용이 발생하게 된다. 또한 “올바른 방향을 유지하는 동시에 신속하게 움직이는 것”이 중요시 되어 빠른 대응과 함께 올바른 대응이 요구되고 있다. 그렇기 때문에 기존의 방법론으로는 이와 같은 요구사항을 만족시키는 데에는 한

※ 교신저자(Corresponding Author): 강병욱, 주소: 경북 경산시 대동 214-1 영남대학교 공대본관 313호(712-749), 전화: 053)810-3914, FAX: 053)810-4630, E-mail: bwkang@yu.ac.kr  
접수일: 2010년 12월 21일, 수정일: 2011년 3월 17일  
완료일: 2011년 4월 23일

<sup>\*</sup> 정회원, 영남대학교 컴퓨터공학과  
(E-mail: tonko96@gmail.com)

<sup>\*\*</sup> 정회원, 제주관광대학 산학협력단  
(E-mail: brain95@ynu.ac.kr)

<sup>\*\*\*</sup> 정회원, 영남대학교 컴퓨터공학과

계가 있다. 이러한 비즈니스 환경변화의 대응하려는 다양한 방안 중 하나가 서비스 지향 아키텍처(Service Oriented Architecture, 이하 SOA)이다[1,2].

기존의 객체지향 개발방식과 컴포넌트 기반 개발방식은 프로세스의 변화에 신속한 대응이 어려우며, 특정 애플리케이션에 국한되어 있다. 또한 애플리케이션을 위한 컴포넌트의 의미론적 통합에 어려움이 있어 최근의 환경변화에 부합하지 못하는 문제를 가지고 있기 때문에 SOA가 가지는 느슨한 결합성(loosely coupling), 비즈니스 요구사항 일치성(business requirement alignment), 블랙박스 형태(black box form)와 같은 개발 접근 방식들은 이러한 문제들을 해결할 수 있는 대안의 하나로써 제시되고 있으며, 이들은 시스템을 서비스 단위로 구분하여 식별하고 개발하고자 하는 방법론이다. 그렇기 때문에 SOA는 독립적으로 실행 가능하며, 외부 인터페이스를 통해서만 접근 가능한 서비스를 재사용 단위로 하는 재사용 패러다임 중 하나이다[3,4]. SOA가 느슨한 결합을 제공함으로써 유연성을 높이고 표준 인터페이스를 기반으로 하여 이기종 환경에서의 민첩성을 높일 수 있다[5]. 이러한 SOA를 활용하여 시스템을 개발하고자 하는 많은 노력이 있었으나 표준화된 방법론이 존재하지 않아서 개발 프로세스 관리와 애플리케이션의 체계적 품질관리가 현실적으로 어렵다. 기존의 방법론들에서는 서비스에 대한 식별과 조합이 중요한 문제로 인식되지 않았다. 그렇기 때문에 서비스 식별에 대한 구체적인 방법을 제안하지 못하였다.

본 논문에서는 이러한 문제를 해결하고자 레거시 시스템을 지원하기 위한 기능기반 모델 중심 비즈니스 프로세스 모델을 제안한다. 그리고 기능기반이라는 시스템에 적용되어 개발될 수 있는 재사용 가능한 서비스 단위와 시스템을 비즈니스에 맞춰 유연하게 공유가 가능하다는 것을 의미한다. 제안하는 프로세스는 서비스 개발 계획단계, 비즈니스 도메인 모델링 단계, 기능 서비스 구현전략결정단계, 서비스 개발 및 통합단계 순으로 4단계와 세부 활동 및 각 활동을 통한 산출물들을 정의한다.

논문의 구성은 다음과 같다. 2장에서는 SOA를 지원하는 기존 개발 프로세스에 대해서 알아보고, 3장에서는 본 논문에서 제안하는 기능기반 중심 프로세스에 대해서 세부 활동 및 각 활동을 통해 생성되는

산출물들을 정의한다. 4장에서는 기존의 프로세스와 본 논문에서 제안하는 프로세스를 비교·분석하여 평가하며, 마지막으로 5장에서는 본 논문의 결론을 맺는다.

## 2. 관련연구

SOA를 위한 개발 프로세스는 OOAD(Object Oriented Analysis and Design)나 CBS(Composite Business Services)에서 찾을 수 없는 추가적인 활동이나 산출물이 요구된다. 현재 알려진 SOA를 지원하는 소프트웨어 개발 방법론으로는 가트너 그룹에서 제시한 SODA(Service Oriented Development of Application), IBM의 SOAD(Service Oriented Analysis and Design), RUP(Rational Unified Process)와 XP(eXtreme Programming)의 특징을 모아 만든 SOUP(Service Oriented Unified Process), 그리고 SOMA(Service Oriented Modeling and Architecture)가 있다.

SODA[6,7]는 웹서비스의 등장에 따라 가트너 그룹에서 SOA를 지원하는 개발방법으로 제안했다. SODA는 SOA를 구현하기 위해 서비스를 설계하고 개발 및 조합하는 일련의 과정에 대한 원칙과 방식을 제시하고 있다. 그리고 어떻게 서비스를 설계하고 개발 또는 조합할 것이며, 서비스를 이용한 애플리케이션 개발에 필요한 방법론은 없는지 등에 대한 문제를 제안하고 있다. 이 방법론은 기존의 CORBA(Common Object Request Broker Architecture)가 성공을 거두지 못한 이유가 개념을 충분히 지원할 수 있는 개발 방법론이 존재하지 않았기 때문이라고 분석하면서, SOA의 개발 개념을 충실히 반영하고자 했다. 이 방법에서는 시스템을 개발하고 배포하기 위한 기본 메커니즘으로 OOD(Object Oriented Design) 및 CBD(Component Based Development)에 대한 개념을 포함하고 있으며, 객체와 컴포넌트들을 서비스로 구성하기 위해 사용한다.

SOAD[8]는 OOAD와 EA(Enterprise Application) 프레임워크, BPM(Business Process Modeling)과 같은 기존의 모델링 개념들을 SOA에 적용시킨 프로세스이다. 이들의 개발절차나 표기법 등은 SOA에서 요구하는 사항을 부분적으로 지원하지만, 전반적인 지원이 미흡하다. 그러므로 추가적인 활동과 산출물

들이 필요하다. SOAD는 일반적인 SOA 모델링 기법인 하향식 접근과 래핑이나 리팩토링 전략을 적용하기 위한 상향식 접근을 결합한 방법을 제시한다.

SOUP[9,10]는 RUP와 XP의 특징만을 모아서 만든 소프트웨어 개발 프로세스로 어떤 조직이든지 SOA 프로젝트에 경우에 적용할 수 있다고 언급하고 있다. 이 방법은 소프트웨어 개발단계를 6개의 단계로 나누고 각 단계에서 수행해야 되는 활동들을 정의했다. 전체 프로세스는 SOA 배치와 관리의 두 부분으로 구분되어진다. 이 때, SOA 배치 부분에 대한 기반은 RUP, 관리부분에 대한 기반은 XP에 둔다.

SOMA[9,11,12]는 SOA를 지원하기 위한 모델링, 분석 설계 기술 및 활동을 포함한다. 이를 위해 SOA의 계층과 각 계층의 요소를 정의하고 각 계층에서 구조적인 결정을 하도록 제안한다. 이 기법도 역시 하향식 접근과 비즈니스 중심의 상향식 접근법을 절충하는 방법이다. 상위수준의 비즈니스 프로세스 기능은 크기가 큰 서비스로 식별되며, 좀 더 크기가 작은 서비스는 기존의 존재하는 기능을 중심으로 식별하도록 한다. 또한 시스템 내에 있으나 외부 비즈니스 협력자나 사용자들이 원하는 기능을 외부로 끌어내어 어댑터나 래퍼로 만들 수 있도록 결정하는 방법을 포함하고 있다.

앞에서 상술한 여러 가지 개발 방법론들이 가지고 있는 문제점을 분석하면 다음과 같다. 첫 번째는 서비스 단위나 세부절차에 대해 명시하지 않고 있다. 두 번째는 비즈니스 프로세스 서비스 조합이나 서비스 저장소 등에 대한 요구사항을 지원하지 못하고 있다. 세 번째는 서비스 식별에 대한 구체적인 활동 제시가 없다. 마지막으로 네 번째는 각 방법론에서 프로세스는 서비스만을 중심으로 하여 고려하고 있다.

기존의 방법을 이용하여 개발을 할 경우에는 서비스 공급자 중심으로 서비스를 개발해야 할지, 또는 서비스 저장소의 서비스를 어떻게 요청해서 사용해야 하는지, 레거시 시스템을 어떻게 래핑을 해야 하는지에 대한 문제를 고려해야 한다.

SOA는 객체를 기반으로 하는 객체지향 개발이나 CBD와는 다른 개발 단위 서비스를 사용하고 있다. 그렇기 때문에 SOA는 객체나 컴포넌트 보다는 더 추상적인 개념인 서비스 단위의 지원이 이루어져야 한다. 그리고 SOA의 계층을 고려할 때, 기존의 개발 프로세스가 엔터프라이즈 컴포넌트 계층을 위한 것

이라면, SOA를 지원하기 위한 프로세스는 비즈니스 프로세스와 엔터프라이즈 컴포넌트 계층의 중간에서 두 가지 모두 지원해야 한다. 다시 말하면 비즈니스 프로세스가 반영된 서비스 개발이 이루어져야 한다[13].

서비스 개발에 필요한 절차와 방법을 정의하고, 각 단계를 수행하기 위해 필요한 가이드라인을 제공을 위한 연구[14]는 사용자가 서비스 개발을 위한 분석 및 모델링에 절차를 제시하고 있다. 이 방법은 기존의 레거시 시스템에서 제공하는 기능을 합병하거나 또는 보다 작은 기능 단위로 분해하여 새로운 기능을 제공하기 어렵다. 그리고 기존의 레거시 시스템이 보유하고 있는 산출물을 충분히 활용하지 못한다. 또한 개발 방법론을 분석에 있어서 난해함을 가져오거나 보다 많은 시간을 요구 할 수 있게 된다. 그러므로 본 논문에서는 이와 같은 문제점을 해결하기 위해서 레거시 시스템 개발을 지원하기 위한 기능기반 모델의 서비스 지향 개발방법론을 제안한다.

### 3. SOA 서비스 모델링 프로세스

CORBA, DCOM, COM+, EJB, J2EE, .NET등을 기반으로 하는 분산 컴포넌트들은 특정 기술에 종속적이지만 상대적으로 연관도가 높아 동일한 플랫폼이나 프레임워크 내부에서의 재사용성이 뛰어나다. 그러나 다양한 환경에서 개발된 기존의 컴포넌트들을 재사용하고자 할 때는 개발된 플랫폼과 구현된 언어에 의존성이 강하기 때문에 컴포넌트간의 상호 운영에 있어서 문제가 발생할 수 있다. 이러한 점을 보완하기 위해 본 논문에서는 비즈니스 프로세스를 반영할 수 있도록 SOA를 이용하여 재사용 가능한 서비스 단위와 시스템의 비즈니스 환경에 맞춰 유연하게 공유가 가능한 서비스 기능중심의 모델링 프로세스를 제안한다. 제안하는 프로세스는 서비스 구현을 하기 위한 가이드라인을 포함한다. 그리고 프로세스 절차 및 산출물에 대한 항목들을 제시하여 이에 따른 개발 수립을 할 수 있도록 한다.

#### 3.1 SOA 서비스 모델링 프로세스

그림 1은 기존의 객체지향 또는 컴포넌트 기반 개발방법론의 절차이다. SOA의 개발 절차는 그림 1의 절차와 크게 다른 점이 없다. 서비스 공급자의 경우

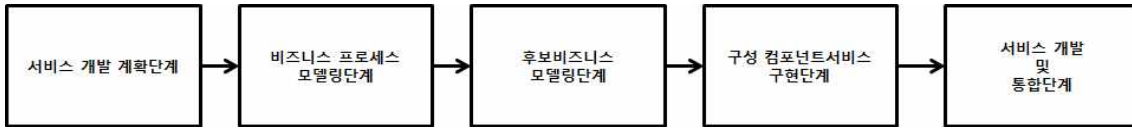


그림 1. 일반적인 소프트웨어 개발 프로세스

일반적인 소프트웨어 개발 방법론을 따르지만 서비스 요청자의 경우에는 서비스 저장소에서 조건에 맞는 서비스를 검색하고 해당 조건에 일치하는 단위서비스에 따라 진행되는 프로세스가 다르게 된다.

그림 2는 본 논문에서 제안하는 전체 프로세스로서 유연하고 적응력 있는 새로운 비즈니스 모델이 필요한 경우와 레거시 시스템을 활용한 비즈니스 모델이 필요한 경우를 조건으로 하여 두 가지의 유형으로 나누고 있다. 첫 번째의 경우는 계획단계에서부터 레거시 시스템을 고려하지 않고 서비스를 구현을 통해 새로운 시스템을 개발하는 경우에 적용한다. 두 번째의 경우는 서비스 식별을 위한 후보서비스가 정의 가능하고, 서비스 적용을 위한 정보가 제공된다고 판단되는 경우 적용한다. 그리고 레거시 시스템에서 사용할 후보서비스 많은 경우와 레거시 시스템의 재사용이 전략적으로 필요한 경우, 레거시 시스템에서 기능별 후보서비스를 위해 조합, 분리가 가능한 경우를 조건으로 둔다.

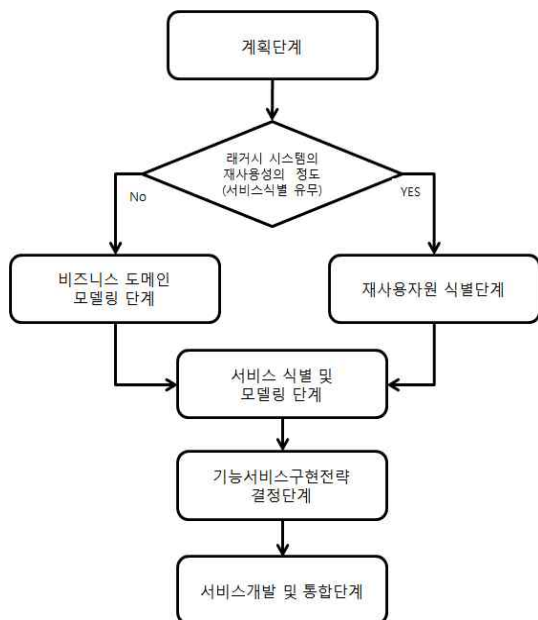


그림 2. 제안한 시스템의 프로세스

본 논문에서 제안하는 서비스 모델링 프로세스는 SOA를 구현하기 위해 필요한 프로세스와 그에 따른 산출물들을 정의하며, 프로세스의 각 단계를 수행하기 위해 필요한 세부지침을 제공한다. 그렇기 때문에 사용자가 레거시 시스템에 기반 두어 SOA를 적용하는 경우와 일반적인 소프트웨어를 개발하기 위한 모델링과 분석에 SOA를 적용할 수 있다. 만약 처음 개발하는 프로세스 경우는 서비스의 제공자의 입장에서 개발하게 된다. 이렇게 개발하는 경우는 일반적인 소프트웨어 개발단계와 비즈니스 도메인 모델링 분석부터 시작한다. 그리고 기존의 시스템을 비즈니스 서비스로 재사용하기 위하여 절충식 접근방식을 이용하여 SOA로 전이하는 체계적인 방법으로 개발하게 된다.

본 논문에서 제안하는 프로세스는 서비스 개발 계획단계, 비즈니스 도메인 모델링단계, 기능서비스 구현결정단계, 서비스 개발 및 통합의 4단계로 구성되며, 세부적인 활동과 작업들에 따라 각 단계에서 산출물들이 포함된다. 여기에서 세부 태스크들은 각 단계의 목표를 달성하기 위하여 행해지는 일련의 활동들이다. 그림 3은 본 논문에서 제안하는 SOA의 모든 프로세스를 나타내고 있다.

### 3.1.1 서비스 개발계획 단계

이 단계의 목표는 개발 프로세스를 결정하는 것이다. 그림 4에서 보는 바와 같이 새로운 시스템을 개발하는 경우에는 비즈니스 환경 분석을 통해 프로세스를 결정한다. 그러나 레거시 시스템에서 개발해야 하는 경우에는 비즈니스 환경분석과 레거시 시스템 분석을 통해 프로젝트의 범위와 목적을 결정하여야 하고, 이를 바탕으로 개발프로세스를 결정한다.

#### (1) 비즈니스 환경분석

조직의 비즈니스 목적을 위해 업무간의 구성관계 및 상호 유기적인 흐름 나타내는 업무 프로세스를 비롯하여 조직의 구성을 하향식으로 계층적으로 분석한다. 여기에서 산출되는 면담결과서는 전체적인

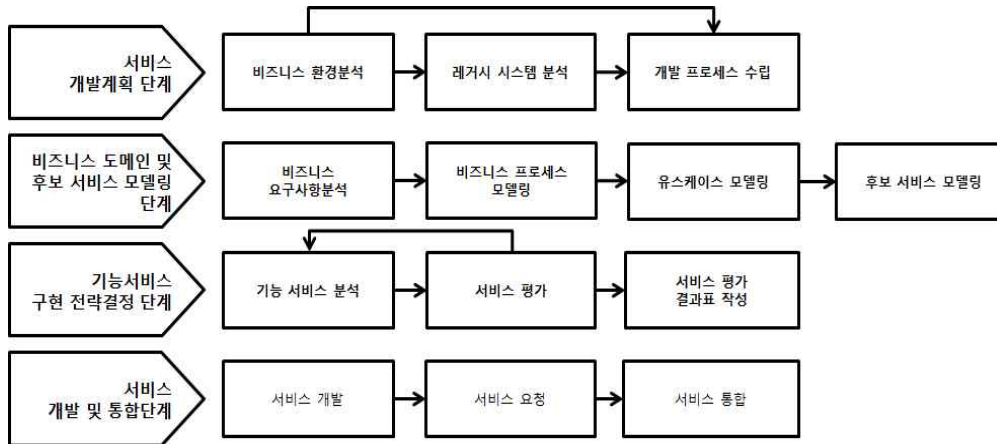


그림 3. 제안하는 서비스 지향 개발 프로세스

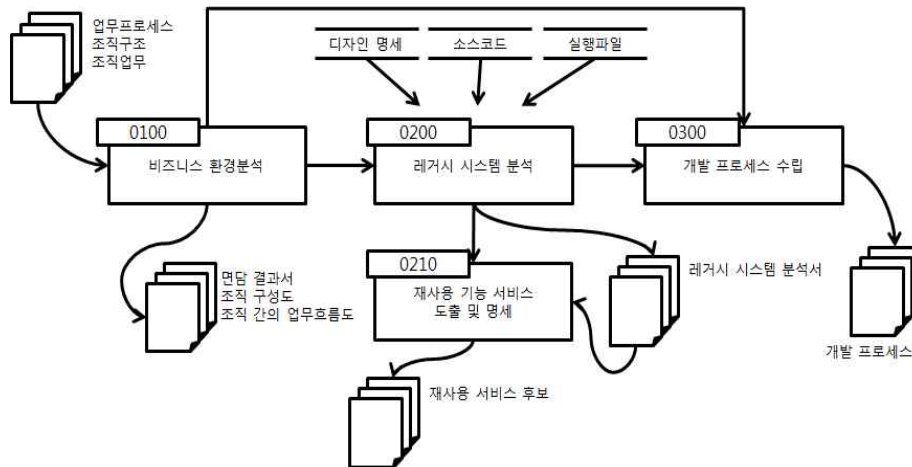


그림 4. 서비스 개발계획 및 산출물 흐름도

시스템 계획을 잡기위한 업무와 업무 흐름을 파악하기 위한 것이고, 조직 간의 업무흐름도는 각 업무간의 구성관계 및 상호 유기적인 흐름을 기술하는 것이다. 이 활동을 통하여 조직의 구조와 각 조직 간의 업무 흐름이 파악되어야 한다.

(2) 레거시 시스템 분석

현조직의 업무 흐름을 분석하고 지원하는 시스템을 파악한다. 이를 위해 레거시 시스템에 가장 적합한 담당자를 선별하고, 담당자와 면담을 통해 시스템의 기능을 업무단위로 분석한다. 또한 각 업무의 기능별로 대응하는 레거시 시스템과 목표 시스템과의 상관관계를 파악하여 개발 프로세스를 수립한다. 그리고 레거시 시스템의 재사용성을 고려한다. 그림 5

는 가용산출물에 대한 제어흐름을 나타내는 활동 다이어그램(activity diagram)이다. 분석 자료에 따라 이루어지는 활동으로서 디자인 명세, 소스코드 및 실행파일

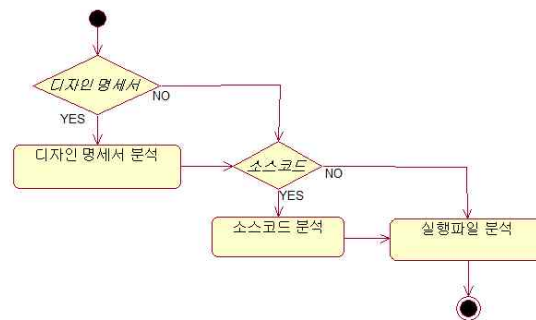


그림 5. 가용산출물에 따른 프로세스

행파일로 흐름을 나타낼 수 있으며, 가용 산출물이 모두 있는 경우를 재사용성이 높은 후보서비스로 선정될 수 있다.

(3) 개발프로세스 수립

비즈니스 환경 분석과 레거시 시스템의 분석 결과를 토대로 앞에서 제시한 그림 2의 프로세스를 적용하여 개발한다.

3.1.2 비즈니스 도메인 및 후보서비스 모델링 단계

비즈니스 프로세스 참여하는 각 단계들은 비즈니스 프로세스 모델링을 상세히 분석하여 이를 공통적으로 이해할 수 있도록 한다. 이때 사용하는 방법은 새로운 소프트웨어 개발방법 또는 레거시 시스템의 후보 서비스를 래핑을 통해 서비스화 하는 방법, 그리고 서비스 저장소에 공개된 서비스를 요청하여 사용하는 방법이 있다. 이 단계의 절차에 따라 소프트웨어 구현 방법이 결정되고, 그 다음에 수행될 다음 단계가 결정된다.

그림 6은 비즈니스 요구사항을 분석하여 비즈니스 도메인 및 후보서비스 모델링 단계의 흐름을 통하여 프로세서 모델링과 유스케이스 모델링을 기준으로 서비스를 식별한다. 식별된 컴포넌트에 대한 서비스 모델을 생성한다. 이전 단계에서 만들어진 유스케이스 다이어그램을 사용하여, 서비스를 식별한 다음 서비스에 대한 명세 나타내며, 웹서비스에 적용할 경우 서비스 기술언어인 WSDL에 들어갈 항목을 기술한 문서인 인터페이스 명세서, 서비스를 다이어그램

으로 형식으로 도식화하여 관계를 명확히 하는 서비스 아키텍처 다이어그램, 식별된 기능을 단위 서비스 목록을 작성한 기능 서비스 목록이 산출된다.

(1) 비즈니스 요구사항분석

비즈니스들을 전체적인 관점에서 이해할 수 있도록 목표 비즈니스 프로세스 모델에 관련된 정보들을 상세히 분석한다. 수립된 비즈니스 요구사항을 분석하여 목표 비즈니스 도메인에서 비즈니스에 참여하는 이해관계를 식별한다. 그리고나서 목표 비즈니스 프로세스 모델의 이름, 설명, 비즈니스 영역 등의 정보를 포함하는 비즈니스 기술서와 모델링에 참여하는 모든 사람들이 공통으로 사용할 용어에 대한 정보를 기술한 비즈니스 용어 사전을 작성한다.

(2) 비즈니스 프로세스 모델링

앞에서 분석된 비즈니스 프로세스 모델을 상세하게 분석하여 도출된 이해 관계자와 이들 간의 비즈니스 시나리오를 기반으로 비즈니스 프로세스 모델을 정의한다.

(3) 유스케이스 모델링

유스케이스 다이어그램을 이용해서 비즈니스에 참여하는 이해관계자들과 이들 사이에 이루어지는 비즈니스 프로세스를 기술함으로써 비즈니스 프로세스들을 보다 명확히 정의할 수 있다. 비즈니스 요구사항분석과 비즈니스 모델링에 따라 작성된 시나리오를 정제하여 유스케이스 다이어그램을 작성한다. 유스케이스 다이어그램은 시스템의 기능성을 사

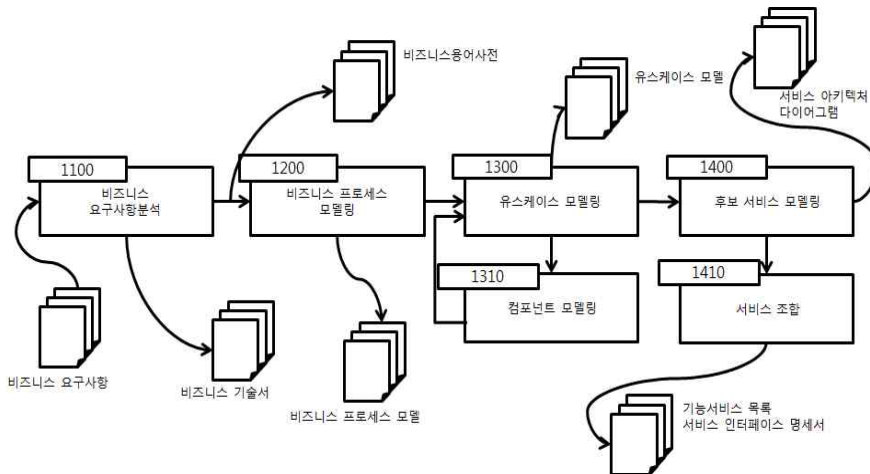


그림 6. 비즈니스 도메인 및 후보 서비스 모델링 및 산출물 흐름도

용자 입장에서 표현한 것이다. 유스케이스 모델은 비즈니스의 관점에서 행위자와 유스케이스 간의 관계를 보여준다. 그림 7은 유스케이스 다이어그램의 예를 나타내고 있다.

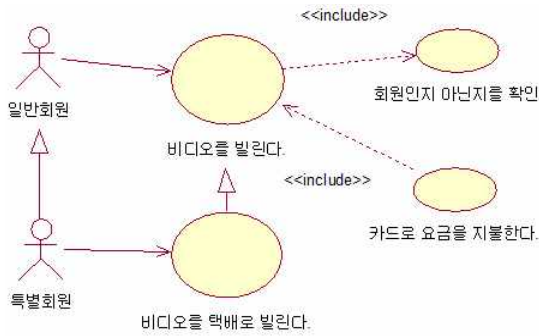


그림 7. 유스케이스 다이어그램 예제

(4) 컴포넌트 모델링

이 단계에서는 비즈니스 프로세스 모델링과 유스케이스 모델링을 기준으로 서비스를 식별하며 식별된 구성 컴포넌트 서비스에 대한 서비스 모델을 생성한다. 서비스를 식별할 때에는 이전 단계에서 만들어진 유스케이스 다이어그램을 사용한다. 서비스 컴포넌트는 의미론적으로 비슷한 기능을 수행 할 수 있는 유스케이스를 재구성하여 식별한다. 또한 컴포넌트 모델링 단계에서 재사용후보 명세서, 레거시 시스템 분석을 통하여 하나의 기능이 여러 핵심 기능을 처리하는 1:n의 관계 또는 n:n의 관계를 가지고 있는 것을 분석하여 클러스터링을 통해 하나의 컴포넌트가 된

다. 클러스터링된 컴포넌트들은 재사용성이 높은 후보서비스로 선정이 된다. 표 1은 기능들을 클러스터링하기 위한 지침이다. 이 단계에서 후보서비스로 선정되면 후보 서비스모델링단계를 통과한다.

그림 8은 레거시 시스템에서 재사용 서비스 후보 명세서와 레거시 시스템의 정보를 기준으로 계획단계의 조직도와 업무 흐름도를 참고하여 서비스화하면서 재사용 가능한 자원을 식별해 놓은 레거시 시스템 분석서, 비즈니스 프로세스 모델, 지침서를 가지고 클러스터링 하는 과정이다.

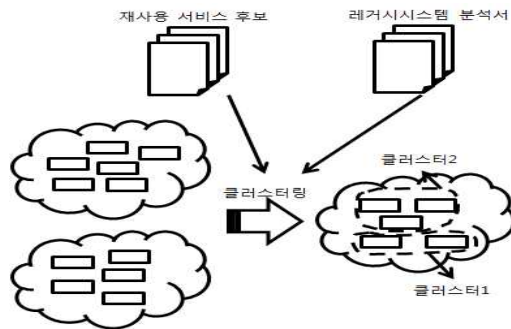


그림 8. 기능들을 클러스터링하는 과정

(5) 후보서비스 모델링

유스케이스 모델링과 컴포넌트 모델링에 의해 식별된 서비스에 대하여 다이어그램 형식으로 하여 후보 서비스들의 관계와 데이터의 입출력, 메시지의 흐름을 나타낸다. 이 과정을 통해 후보서비스에 대한 상세 정보를 모델링한다. 표 2는 후보서비스의 가치

표 1. 클러스터링하기위한 지침

클러스터링 지침
■ 서로 다른 기능이 같은 기능을 호출할 경우, 각 기능은 하나의 클러스터에 포함되어야 한다.
■ 여러 개의 기능이 같은 데이터베이스를 사용할 경우 하나의 클러스터에 포함되어야 한다.
■ 서로 다른 기능이 같은 기능을 호출할 경우 각 기능은 하나의 클러스터에 포함되어야 한다.

표 2. 후보서비스 선정지침

후보서비스 선정지침
■ 특정 플랫폼에 종속적이지 않는 기능. 독립성 및 표준을 따르지 않아 이식성이 떨어짐
■ 비즈니스 환경변화에 대해 확장성이 떨어지지 않는 경우
■ 복잡한 인터페이스를 필요로 하지 않는 경우
■ 높은 품질의 기능이 제공이 보장되는 경우
■ 재사용성이 잠재성이 높아 ROI를 기대할 수 있는 경우

를 높이기 위해 후보서비스를 선정하기 위한 지침이다. 식별된 단위 서비스는 모델링 될 수 있으며, 웹서비스에 적용할 경우 웹서비스에 접근하는 인터페이스 WSDL로 기술하여 서비스 저장소에 등록 할 수 있도록 한다. 후보서비스 모델링을 통해 입출력 데이터나 메시지의 흐름에 대한 정보를 보다 구체적으로 표현한 결과의 산출물인 서비스 아키텍처 다이어그램을 산출한다.

(6) 서비스 조합

후보 서비스의 비즈니스 규칙과 비즈니스 논리에 따라 비즈니스와 후보 서비스를 조합한다. 서비스의 조합은 세 가지 유형(병합, 생성, 분리)으로 정제 가능한 기준을 둔다. 병합은 두 후보 서비스를 표 3의 지침에 따라 하나의 후보서비스로 병합되며, 생성은 두 후보서비스의 공통적인 기능을 새로운 후보서비스로 생성된다. 마지막으로 분리는 하나의 후보서비스를 두 개 이상의 후보서비스로 분리한다. 이 과정에서의 산출물은 식별된 기능 단위의 서비스 목록인 기능서비스목록과 서비스에 대한 명세 나타내며, 웹서비스에 적용할 경우 서비스 기술언어인 WSDL에 들어갈 항목을 기술한 문서인 서비스 인터페이스 명세서가 작성된다.

3.1.3 기능서비스 구현 전략결정 단계

이 단계는 서비스 인터페이스명세서와 기능서비

스 목록을 기준으로 처음부터 개발하는 방법과 레거시 시스템을 래핑하여 개발방법이 정해지고 난 후 수행된다. 이때 산출물로 서비스평가 기준서이며, 이를 바탕으로 해당 서비스를 평가한다. 그림 9은 기능서비스 구현전략 및 산출물의 흐름을 보여준다.

(1) 기능서비스 분석

기능 서비스 분석단계에서는 시스템이 수행해야 할 처리나 가져야 할 기능 및 시스템이 작동하면 가져야 되는 특징인 비기능 서비스 목록과 서비스 인터페이스 명세서를 바탕으로 기능별 서비스를 분석한다. 여기에서 생성되는 산출물은 서비스 평가항목과 평가하는 기준을 정의한 서비스평가 기준서이다.

(2) 서비스 평가

이전 단계에서 생성된 서비스 평가 항목과 각 항목을 평가하는 기준을 정의한 서비스 평가 기준서와 기능 서비스를 분석한 결과를 토대로 기능별 서비스에 대한 평가를 시행한다. 서비스 평가를 통해서 레거시 시스템의 후보서비스를 래핑하여 서비스를 할 것인지 또는 처음부터 개발할 것인지, 아니면 다른 공급자에서 서비스를 요청할 것인지를 결정한다.

(3) 서비스 평가결과표 작성

기능서비스 목록에 따라 분석과 평가 결과를 보여 줄 수 있는 표 4의 서비스 평가 결과표를 작성한다.

표 3. 서비스 조합지침

서비스 조합지침	
병합	<ul style="list-style-type: none"> <li>■ 하나의 후보서비스가 다른 후보서비스를 포함하고 있을 경우</li> <li>■ 하나의 후보서비스가 실행을 위해 다른 후보서비스에 의존도가 다소 높을 경우</li> </ul>
분리	<ul style="list-style-type: none"> <li>■ 하나의 후보서비스가 큰 기능 제공하며, 여러 개의 독립적인 기능으로 분리될 수 있는 경우</li> </ul>
생성	<ul style="list-style-type: none"> <li>■ 하나의 후보서비스와 다른 후보서비스중 공통적인 기능을 포함하고 있는 경우 또 다른 후보서비스를 생성할 수 있다. 생성 후 나머지 기능이 독립적인 경우 새로운 후보서비스로 생성할 수 있다.</li> </ul>

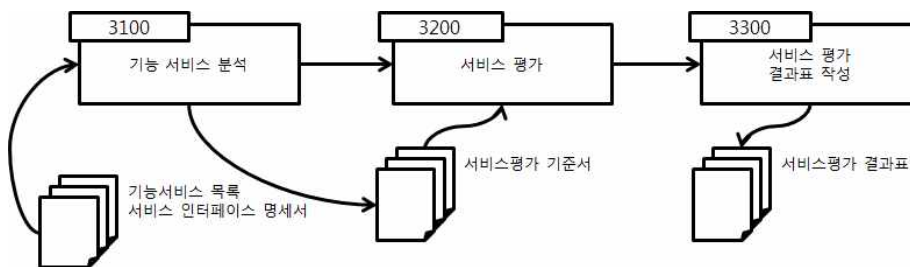


그림 9. 기능서비스 구현전략결정 및 산출물 흐름도



표 4. 서비스 평가 결과표

서비스 목록	재사용정도 가중치 : 1	ROI 가중치 : 0.6	기능의 중요성 가중치 : 0.8	시장성 가중치 : 0.6	etc 가중치 : 0.5	평가값
F_01	1	1	1	1	1	0.7

평가항목 중 재사용정도는 후보서비스가 얼마나 많은 곳에 제공되는지를 나타내며, ROI는 서비스로 전이할 경우 투자대비 수익을 나타내고, 기능의 중요성은 얼마나 많은 기능을 제공하는지를 나타낸다. 마지막으로 시장성은 잠재적인 발전가능성을 나타낸다.

서비스평가 항목의 점수는 -1~1점 사이의 값을 가지도록 한다. 수식 1은 서비스 평가항목에 점수를 부여하여 각 항목에 맞는 가중치를 포함하여 평가값을 구하는 식이다. 가중치는 평가항목의 중요성에 따라 0~1사이의 값을 부여할 수 있다. 높은 평가값이 서비스가 될 확률이 높다.

$$\text{평가값} = \frac{(\text{재사용도} \times 1) + (\text{ROI} \times 0.6) + (\text{기능의 중요성} \times 0.8) + (\text{시장성} \times 0.6) + (\text{etc} \times 0.5)}{\text{평가항목의 총갯수}} \quad (1)$$

3.1.4 서비스 개발 및 통합단계

그림 10은 새로운 서비스를 개발하는 과정을 나타내고 있다. 이 경우에는 전통적인 개발 프로세스가 그대로 적용된다. 만일 레거시 시스템을 재활용하는 경우에는 레퍼를 통해 선발된 우선순위가 높은 후보 서비스들을 SOA에 맞도록 서비스화 하는 작업이 필요하다. 또한 공개된 서비스를 사용할 경우에는 서비스 저장소를 검색하여 해당 서비스를 요청한다. 서비스들의 구현전략이 결정되면 새로 개발되는 서비스

의 경우에는 기존 개발프로세스를 따르게 되며 분석 및 설계를 통한 서비스 명세와 전체 비즈니스 프로세스에 서비스 평가 결과에 따른 개발할 서비스와 요청할 서비스를 정의하여 전체 통합된 서비스에 대한 서비스 통합 명세서가 산출되게 된다.

(1) 서비스 개발

새로운 서비스의 개발은 서비스 공급자의 입장에서 고려한다. 이 경우에는 일반적인 소프트웨어 개발 프로세스를 그대로 적용한다. 여기에서는 새로 개발된 서비스의 모델링한 결과를 포함하는 서비스 명세서가 산출된다.

(2) 서비스 요청

이 활동은 서비스 평가를 통하여 외부의 서비스를 사용하겠다고 결정되어진 경우에 수행된다. 이 경우에는 서비스저장소를 검색하여 요청 가능한 서비스 선택한다. 이 활동에는 요청하려는 서비스 검색을 통해 선택을 한 후, 다른 서비스 공급자가 개발한 서비스를 이용하여 새로운 서비스를 개발하는 경우이다.

(3) 서비스 통합

이 단계는 비즈니스 프로세스에 따라 새로 개발된 서비스와 레거시 시스템에서 개발된 서비스를 사용하려고 할 때 요청된 서비스를 통합하는 단계이다. 서비스 통합을 위해서는 복합적인 비즈니스 서비스

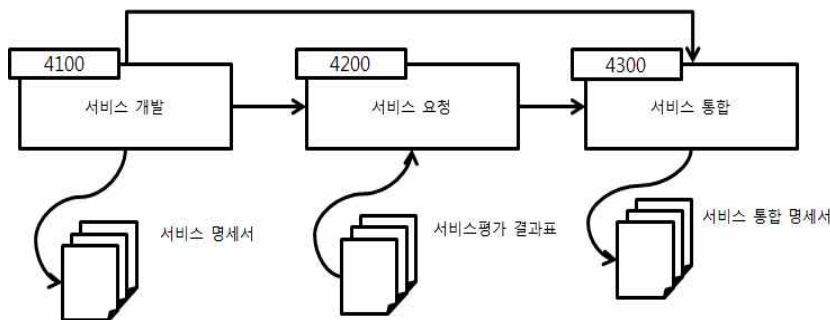


그림 10. 서비스 개발 및 통합단계

들을 비즈니스 관점에서 정렬, 통합, 실행해야 하는 지에 대한 전략이 고려되어야한다. 이때 생성되는 산출물은 개발할 서비스와 요청할 서비스를 정의하여 전체 통합된 서비스에 대한 명세가 포함된 서비스 통합 명세서이다.

#### 4. 제안 방식의 고찰

이 장에서는 기존의 개발방법론과 본 논문에서 제안한 개발방법론을 비교·분석한다. 표 5는 기존 소프트웨어인 개발 방법인 객체지향 및 CBD와 본 논문에서 제안하는 SOA개발 프로세스의 재사용 관점에서 비교한 결과이다. 비교방법은 재사용단위, 처리단위, 입출력방법, 식별방법, 재사용 방법을 비교하였다. 객체지향 방식의 개발 방법론에서 식별방법은 시스템구축의 블록이 클래스가 되며, 논리보다는 자료를 바탕으로 구성된다. 객체는 유스케이스 명세서를 보고 명사 또는 명사구를 기준으로 식별한다. 이 방식에서의 재사용 방법은 클래스들 간의 상호의존관계를 클래스의 재사용 단위로 사용하기 때문에 현실적으로 어렵다. CBD 방식의 개발 방법론에서 식별방법은 요구사항분석을 통해 만들어진 분석모델을 통해 식별하며 연관된 객체를 조합하여 컴포넌트를 구성한다. 이 방식에서의 재사용 방법은 컴포넌트의 재사용을 추구하지만 이기종 플랫폼간의 상호 호환이 불가능하다. 본 논문에서 제안하는 방법은 기존 개발방법이 가지고 있는 문제점을 극복하기 위해서 레거시 시스템의 자원을 활용하고 구현전략 및 클러

스터링을 통해 프로세스의 재사용 전략을 적용하였다. 그 결과로 기존의 개발방법론에 비해서 재사용단위의 높음을 알 수 있다.

표 6는 기존의 SOA개발 방법론과 본 논문에서 제안하는 방법론을 비교 분석한 결과이다. 각 방법론을 평가하기 위해서 세부 평가기준[15,16]을 가지고 평가하였다. 각 방법론의 항목이 평가 기준에 따라 적합성 여부를 판단하고 support(○), semi-support(△), non-support(×)의 세단계로 구분하여 평가한다. SOA지원 개발 방법론에서 서비스 구현전략과, 서비스 식별, 산출물을 중심으로 SOA에서의 개발 단위가 되는 서비스를 식별하는 방법과 각 단계 산출물의 연관관계, 식별된 기능의 구현 전략이 고려되었는지를 비교·분석하였다. 기존의 방법론들은 개발 방법들은 대부분 하향식 개발 방법을 사용하고 있으며 산출물과 지침을 제공하고 않고 있다. 그러나 본 논문에서 제안한 방법은 상향식과 하향식을 절충한 방법을 사용했으며, 각 단계별로 산출물 및 지침서를 제공하고 있다. 또한 적합성은 본 논문에서 완전한 검증을 거치지 않았기 때문에 semi-support로 표기하였다.

표 7은 기존의 개발 방법론과 본 논문에서 제안한 방법론을 세부 평가기준[15,16]을 이용하여 평가한 결과를 나타내고 있다. 여기에서 해당 평가항목을 지원 하면 support(○), 그렇지 않으면 non-support(×)로 분류 하였다. 레거시 시스템을 지원하는 SOA개발 프로세스에서 제시되어야 하는 서비스 식별 및 구현전략이 제공되었는지를 기존에 평가항목에 추

표 5. 기존 개발방법론과 제안된 방법론 비교분석

비교항목	OOD	CDB	제안방법론
단 위	클래스	컴포넌트	서비스
처리단위	합 수		비즈니스 프로세스
입 출 력	데이터		메시지
식별방법	시스템구축의 블록이 클래스가 됨. 동작보다는 객체, 논리보다는 자료를 바탕으로 구성. 유스케이스 명세서를 보고 명사 또는 명사구를 기준으로 식별	요구사항 분석을 통해 만들어진 분석 모델을 통해 식별. 연관된 객체를 조합하여 컴포넌트 구성	비즈니스 시나리오를 반영. 유스케이스를 재구성하여 비즈니스 서비스 식별
재사용단위	하	중	상
재사용 방법	클래스들 간의 상호의존관계로 클래스를 재사용단위로 사용 어려움	컴포넌트의 재사용을 추구하나 이기종 플랫폼간의 상호호환 불가능	레거시 시스템의 자원 활용과 구현 전략결정 및 클러스터링을 통한 재사용 전략 제시

표 6. 기존 SOA개발 프로세스와 제안된 프로세스 비교분석

	SODA[5,6]	SOAD[7]	SOMA[8,10,11]	SOUP[8,9]	[13]방법론	제안 방법론
모델링 기술	×	○	×	×	○	○
개발방법	상/하향식	하향식	상/하향식	하향식	하향식	절층식
산출물	×	×	×	×	○	○
산출물/ 지침서	× / ×	× / ×	× / ×	× / ×	단계별 산출물 / ×	단계별 산출물 / ○
세부 활동	통합된 라이프사이클에 대한 언급 없음	계층화된 개발단계	세부 활동 정의 없음	세부 활동 정의 없음	정제된 유스케이스 모델	각 단계별 세부 활동 정의
서비스 식별	비즈니스적 측면의 요구사항기술	서비스 식별 및 정의에 객체기반 기술을 사용	비즈니스 도메인의 기능적 분해를 통해 식별	×	유스케이스기반 서비스 식별	기능적 요구사항을 통한 유스케이스 기반 서비스 식별
서비스 구현전략 결정	×	×	×	×	○	○
적합성	○	○	○	○	△	△

가하여 비교하였다. 세부 평가항목은 SOA 개발 프로세스를 중점을 두어 세부 활동이 고려되었는지에 대해서 비교·분석하였다. 대부분의 방법들은 산출물이 없다. 그렇기 때문에 출력 산출물들을 정하고 어떤 과정으로 완료되었는지 기준이 없으나, 제안된 방법은 각 단계별 산출을 정하고 세부 활동을 통해

표 7. 기존 SOA개발 방법론과 제안된 방법론에 대한 세부 평가항목

평가 항목	SO DA	SO AD	SO MA	SO UP	제안 방법론	비 고
1. 각 프로세스에 대해서 순서를 가지고 연결되었는가?	○	○	○	○	○	명세서
2. 각 프로세스의 출력 산출물을 정하고 어떤 과정으로 완료되는지 기준을 제시하였는가?	×	×	×	×	○	명세서 (다이어그램)
3. 비즈니스 프로세스 분류 체계에 대하여 각 계층에 대한 기준이 성립되었는가?	○	○	○	○	○	명세서
4. 비즈니스 프로세스를 분해하는 기준을 제시하였는가?	×	×	○	×	○	명세서 (다이어그램)
5. 서비스 정제에 대한 기준을 제시하였는가?	○	○	○	○	○	명세서
6. 서비스 후보 목록보다 정제된 서비스 목록이 재사용성을 보장하는가?	×	○	×	×	○	명세서 (다이어그램)
7. 서비스의 정제에 대한 기준을 제시하였는가?	○	○	○	×	○	명세서
8. 서비스 정의에 대해서 비즈니스 프로세스에서의 정의된 정보를 모두 활용하는가?	○	○	○	×	○	명세서 (문서)
9. 서비스 사용자 관점에서 모델링 영역과 제공자 관점에서의 모델링 영역을 구분하였는가?	×	○	○	○	○	명세서 (문서)
10. 정의된 서비스에 대해서 서비스 사용자가 이해 가능한 단위인가?	○	○	○	○	○	명세서

기준을 제시하였다.

## 5. 결론 및 향후과제

SOA는 기업의 인프라의 복잡성 및 유지비용을 최소화하여 기업의 생산성과 유연성을 극대화할 것으로 기대되어 산업계의 꾸준한 사용이 증가하고 있다. 기업은 SOA로 변화하기 위하여 기존의 시스템을 버리고 새로운 SOA를 도입하기에는 비용과 시간 면에서 많은 위험이 있기 때문에 기존의 레거시 시스템을 최대한 이용하면서 점차적으로 SOA로 변화하기를 원하고 있다. SOA의 대표적인 방법론인 SOUP, SOAD, SODA, SOMA에 대하여 특징을 비교분석하였다. 이 대표적인 방법론은 서비스 식별, 명세, 산출물 및 구현 절차에 대한 구체적인 사항이 제시되지 않고 있다.

본 논문에서는 제안한 SOA개발 프로세스 기존의 개발방법과 달리 각 단계에 대하여 단계별 세부 활동과 산출물, 지침서를 포함하고 있으며 서비스를 식별하기 위한 가이드라인과 식별된 서비스에 대한 구현 전략 결정에 대한 가이드라인도 제시하고 있다. 제안된 프로세스는 4단계로 구성되어있으며, 각 단계의 세부적인 활동과 그에 따른 산출물 및 지침서를 정의하였다. 또한 레거시 시스템을 재사용을 위해 기능을 서비스 단위로 구분하였으며, 구현된 서비스는 UML 모델링을 통해 재구성하여 비즈니스 서비스로 식별하였다. 이렇게 함으로써 기존의 개발 방법론보다 더 높은 재사용성을 구현할 수 있었다. 그리고 레거시 시스템의 자원 활용과 구현 전략 결정을 통해서 레거시 시스템의 재사용성을 향상시키는 방법으로 제안하였다. 더욱이 기존에 개발 방법과는 다르게 서비스 개발에 필요한 구체적인 절차와 방법에 대한 가이드라인을 제시하였다. 따라서 이를 이용하면 SOA 기반의 개발과 재사용을 통한 개발시간의 단축 및 개발비용을 줄일 수 있을 것으로 사료된다. 후 연구과제로는 본 논문에서 제안한 방법론을 이기종간의 P2P 환경에서 실제 개발에 적용하여 재제안한 개발 방법론과 기존 개발 방법론을 비교·검토할 것이다.

## 참 고 문 헌

- [1] K. Channabasavaiah, K. Holley, and E. M.

- Tuggle, "Migrating to a Service Oriented Architecture, Part 1," *IBM Whitepaper*, 2003.
- [2] F. Jammes. and H. smit, "Service Oriented Paradigms in Industrial Automation," *IEEE Transaction on Industrial Informatics*, Vol.1, No.1, pp.62-70, 2005.
- [3] Erl, T., *SOA Principles of Service Design*, Prentice Hall, 2007.
- [4] 박동식, 신호준, and 김행곤, "SOA기반의 웹서비스 컴포넌트 개발에 관한연구," 한국멀티미디어학회 논문지, 2004, pp.1496-1504
- [5] A. Brown, S. Johnston and K. Kelly, "Using Service Oriented Architecture and Component Based Development to Build Web Service Applications," *Rational Software Corporation*, 2002.
- [6] Samir Nigam, "Service Oriented Development of Applications(SODA) in Sybase Work-space," *Sybase Whitepapers*, 2005.
- [7] D. Plummer, "Service Oriented Development of Application:SODA Pops," *Gartner's Internet Strategies Commentary*, COM-129640, 2001.
- [8] O. Zimmermann, P.Krogdahl, and C. Gee, "Elements of Service Oriented Analysis and Design," *IBM developer Works*, 2004.
- [9] Grady Booch, *Object Oriented Analysis and Design with Applications*, Addison Wesley Professional, 1994.
- [10] K. Mittal, "Service Oriented Unified Process (SOUP)," 2006, <http://www.kunalmittal.com/html/soup.shtml>
- [11] 노갑수, "SOA(Service Oriented Architecture) 구축전략(2)," 정보통신연구진흥원, 2006.
- [12] A. Arsanjani, "Service Oriented Modeling and Architecture(SOMA)," *IBM Developer Works*, 2004.
- [13] 윤홍란, 김유경, and 박재년, "유스케이스기반 웹서비스식별 방법," 한국정보과학회 컴퓨터종합학술대회 논문집, 2005, pp. 352-355.
- [14] 김유정 and 윤홍란, "SOA를 위한 서비스지향 개발 프로세스," *한국전자거래학회지* 제12권 제

2호, 2007.

- [15] Java Native Interface, <http://java.sun.com>
- [16] IEEE Computer Society and ACM, "Guide to the Software Engineering Body of Knowledge (SWEBOK)," *IEEE Computer Society*, 2004.



**유 춘 근**

2005년 2월 경산대학교 정보처리  
학과 이학사  
2007년 2월 영남대학교 컴퓨터공  
학과 공학석사  
2007년 3월~현재 영남대학교 컴  
퓨터공학과 박사과정

관심분야: 소프트웨어공학, UML, SOA, Ubiquitous Sensor Network



**임 철 우**

1999년 2월 제주대학교 통신공학  
과 공학사  
2002년 2월 제주대학교 통신공학  
과 공학석사  
2009년 8월 영남대학교 컴퓨터공  
학과 공학박사

2011년 3월~현재 제주관광대학 산학협력단  
관심분야: 소프트웨어공학, UML, SOA, Mobile Ad-hoc  
Network, Ubiquitous Sensor Network



**강 병 옥**

1970년 2월 영남대학교 전기공학  
과 공학사  
1977년 2월 영남대학교 전자공학  
과 공학석사  
1994년 2월 경북대학교 전자공학  
과 공학박사

1973년~1976년 영남대학교 전자계산연구소(SE)  
1977년~1978년 영남전문대학 전자과 전임강사  
1979년~현재 영남대학교 전자정보공학부 컴퓨터공학  
전공 교수  
관심분야: 소프트웨어공학, UML, SOA, Ubiquitous  
Sensor Network