논문 2010-1-6

서비스 유연성을 높이기 위한 유스케이스 가변성 분석 기반의 SOA 서비스 모델링

김유경*, 도경구*, 주영도**

UseCase Variability Analysis based Service Modeling to Promote Adaptability for Service-Oriented Architecture

Yukyong Kim*, Kyung-Goo Doh*, Young-Do Joo**

요 약

서비스지향 아키텍처(SOA) 기반의 시스템이 효율적이고 성공적으로 자리 잡기 위해서는 적절한 추상화와 잘 정의된 인터페이스를 가지는 올바른 서비스 구성이 필수적이다. 하나의 유스케이스는 시스템을 구성하는 개체의 내부 구조를 드러내지 않고 시스템의 행위를 정의할 수 있으므로 좋은 서비스 후보가 될 수 있다. 또한 소프트웨어 프로덕트 라인의 가변성 분석 방법을 적용하여 유스케이스들 사이의 가변점과 의존관계를 모델링함으로써, 보다 적응성 높은 서비스 도출이 가능해진다. 본 논문은 SOA 기반의 시스템을 개발하기 위해 필요한 서비스 모델링 방법을 제공하기 위한 것으로 적합한 추상화 수준에서 잘 정의된 인터페이스를 갖는 서비스를 추출을 위해 유스케이스의 가변성 분석을 이용한 서비스 식별 및 정의 과정을 제안한다.

Abstract

The key challenge in developing Service-Oriented Architecture based systems with efficiency and success is the identification of right-grained services having a well-defined interface. A Use Case can be a good service candidate because it is used to define the behavior of a system without revealing the entity's internal structure. To extract highly adaptable services, variability analysis of Software Product Line Engineering can apply to modeling variation points and dependency between Use Cases. This paper is to present a service modeling method to develop SOA based systems with right-grained and well-defined services. The method include a procedure for identifying and defining services based on variability analysis of Use Cases.

한글키워드 : 서비스지향 아키텍처, 서비스 모델링, 가변성 분석, 유스케이스

I. 서 론

최근 IT 화두 중의 하나는 서비스 지향 아키텍처(Service-Oriented Architecture, SOA)이다. SOA는 소프트웨어 재활용성과 단위 기능들 간

* 한양대학교 컴퓨터공학과
(email: {yukyong, doh}@hanyang.ac.kr)
** 강남대학교 컴퓨터미디어공학부
(email: ydjoo@kangnam.ac.kr)
접수일자: 2010.3.21 수정완료: 2010.4.22

의 상호호환성에 중심을 둔 소프트웨어 설계 방식으로 기업의 생산성과 유연성을 보장하는 차세대 소프트웨어 아키텍처로 기대를 모으고 있다. SOA 기반의 시스템이 효율적이고 성공적으로 자리 잡기 위해서는 적절한 추상화(right-grained)와 잘 정의된(well-defined) 인터페이스를 가지는 올바른 서비스 구성이 필수적이며 이를 위한 서비스 모델링은 시스템 개발 전체의 성패를 좌우할 수 있는 가장 중요한 요소이며 시작점이 된다.

모델링 관점에서 본다면, 가장 큰 도전과제는 잘 설계된 의미 있는 서비스 추상화에 대한 체계적인 구성이다. 즉, SOA를 구성하는 서비스를 어떻게 식별하고 기술하는지가 가장 중요한 문제라고 할 수 있다. 지금까지 서비스 모델링은 고객과 프로젝트 환경에 맞추어 여러 방법론과 기술들을 조합해서 이루어졌으며, 주로 UML 모델들이 분석과 초기 설계 단계에서 매우 중요한 역할을 하고 있다. 특히 유스케이스(Use Case) 모델은 시스템에서 수행되어야 할 작업들을 전체적인 관점에서 이해할 수 있도록 외부 시스템에 대한 시스템의 경계(boundary)와 인터페이스를 시각화하고 시스템이 제공하는 기능(functionality)에 대한 개요를 제공한다 [1]. 하나의 유스케이스는 시스템을 구성하는 개체의 내부 구조를 드러내지 않고 시스템의 행위를 정의할 수 있으므로 좋은 서비스 후보가 될 수 있다 [2].

소프트웨어 재사용 개념은 ADT(Abstract Data Type)로부터 객체, 컴포넌트 그리고 현재의 서비스까지 계속 진화해오고 있다. 서비스는 동적이며 느슨하게 결합된 특성을 갖고 있으며, 소프트웨어 프로덕트 라인(Software Product Line, SPL)에서 제공하는 가변성 관리 기술을 유용하게 활용할 수 있는 컴포넌트와 다른 몇 가지 가변성들이 존재한다. SOA 서비스의 가변성은 플랫폼의 선택, 인터페이스 기술, 서비스 수준 협약(Service Level Agreement, SLA) 및 보안 정책 등이 있다. 따라서 유스케이스 다이어그램에 프로덕트 라인 공학의 가변성 분석 기법을 적용하여 유스케이스 가변성을 모델링한다면, 서비스의 가변성으로 대응시킬 수 있고 이를 통해 적응성 높은 서비스를 정의할 수 있다.

본 논문은 SOA 기반의 시스템을 개발하기 위해 필요한 서비스 모델링 방법을 제공하기 위한 것으로 적합한 추상화 수준에서 잘 정의된 인터페이스를 갖는 서비스를 추출을 목표로 한다. 이를 위해 유스케이스의 가변성 분석을 이용한 서비스 식별 및 정의 과정을 제안한다.

II. 관련 연구

1. SOA와 서비스 모델링

서비스는 서비스 소비자의 관점에서 봤을 때 비즈니스 적으로 재활용 가능한 단위의 기능을 캡슐화하며 그 자체로서 독립적이어야 하며 서비스를 사용함에 있어 제약이 없어야 한다. 그림 1은 SOA의 핵심적인 요소를 이루는 서비스 계층이다 [3]. 소프트웨어 프로그램으로 구축된 어플리케이션 계층이 가장 하위에 있고, 비즈니스 전문가에 의해 잘 구축된 비즈니스 프로세스 계층이 가장 상위에 있다. 이 두 계층을 연결하는 서비스 계층은 어플리케이션 서비스와 비즈니스 서비스 및 서비스 연동에 관여하는 오케스트레이션(Orchestration) 서비스로 나뉜다.

서비스 모델링은 개발하게 될 어플리케이션 시스템을 재사용 및 상호운용이 가능한 서비스들로 구성하여, 시스템이 제공하는 기능을 구현 할 수 있도록 설계하는 것이다. 지금까지 IBM의 SOMA (Service-Oriented Modeling and Architecture)를 비롯하여, SODA (Service

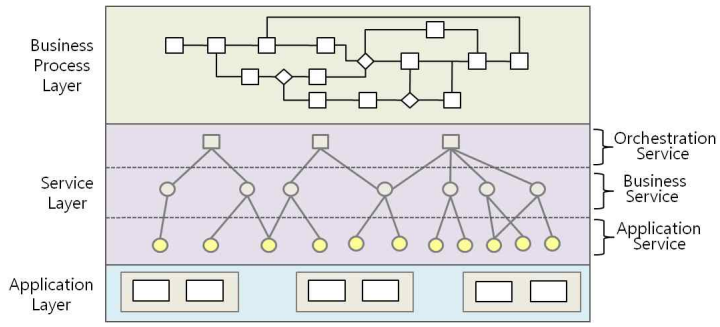

그림 1. 서비스지향 아키텍처의 서비스 계층
Figure 1. Service layers of Service-Oriented Architecture

Oriented Development of Applications), SOUP(Service Oriented Unified Process), 그리고 SOAD(Service Oriented Analysis and Design)등의 서비스 모델링을 포함한 SOA 기반 개발 방법론 및 서비스 식별에 관한 방법들이 제시되어 왔다 [4-10]. 그러나 서비스 기반 개발 절차상의 서비스 식별을 포함한 모델링 과정을 하향식(Top-down), 상향식(Bottom-up), 또는 절충식 (Hybrid)으로 구분할 뿐 세부적인 절차나 활동을 명시하지 않고 있다 [11]. 또한 대부분의 상향식 방법들은 컴포넌트 기반으로 서비스를 식별하여 정의하고 있다. 그러나 서비스는 비즈니스 작업 흐름을 반영하는 단위가 되어야 하므로, 비즈니스 프로세스를 기반으로 서비스 모델링이 이루어져야 한다. 유스케이스는 시스템의 기능을 표현하는 모델로서, 유스케이스 기술서에 비즈니스 프로세스를 포함하고 있으므로 적절한 서비스 후보가 될 수 있다 [12]

2. SPL 가변성 모델링

SPL은 소프트웨어 자산의 적극적이고 효과적인 재사용을 도모함으로써, 어플리케이션 개발 비용을 절감할 수 있다. SPL의 능력과 가변성을 인식하는 것은 프로덕트 라인 자산의 성공적인 재사용을 이끄는 주된 요인이 된다. SPL의 어플리케이션마다 변경될 수 있는 부분이 가변성(variability)으로서, 고객의 관점에서 인식할 수 있는 모든 종류의 변경 가능성을 내포할 뿐만 아니라 기본적으로 모든 기능적인 면(functional aspects)을 포함하게 된다.

SPL 가변성은 가변점(variation points), 가변치(variants), 그리고 그들 사이의 의존관계(dependency)로 표현한다. SPL의 어플리케이션에 대한 요구사항이 정의될 때 가변점과 가변치는 반드시 고려되어야 하며, 이들이 명시적으로 표현되어야 한다. 또한 가변점의 공통적인 특성을 표현하는 가변치는 "mandatory"로, 변경되는 부분은 "option" 또는 "alternative"가 되어야 한다.

가변성 요소들 사이의 관계는 의존관계로 표현되며, 특성모델링(feature modeling)에서는 8가지 유형의 의존관계를 정의하여 사용한다. 이 가운데 프로덕트 라인의 가변성을 표현하기에 적합한 가변성 요소들 사이의 상호의존관계(interdependency) 유형은 4가지로 요약되며, 그림 2에 잘 나타나 있다 [13, 14]. 한 가변치의 바인딩(binding)에 요구되는 가변치(required

variant)와 배제되어야 할 가변치(excluded variant)의 관계를 나타내는 Requires 의존성과 Exclusive 의존성이 있고, 한 가변치의 바인딩이 또 다른 가변치에 긍정적인 영향을 주는지 부정적인 영향을 주는지에 따른 관계를 나타내는 Hints 의존성과 Hinders 의존성이 있다. 이 관계들 중 Requires, Hints, Hinders 의존성은 단일방향성(unidirectional)을 갖고, 반면 Exclusive 의존성은 양방향성(bidirectional)이다.

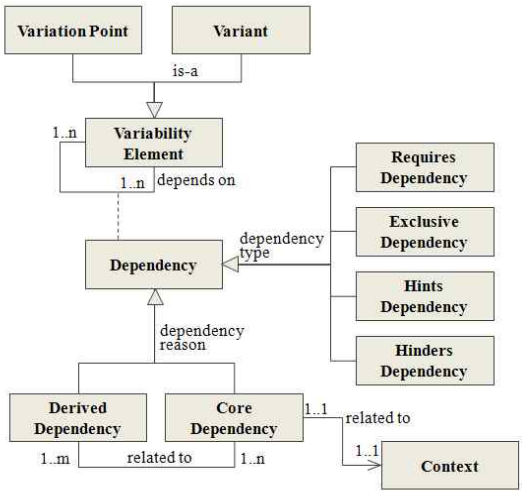

그림 2. SPL 의존관계 유형

Figure 2. Dependency types in SPL

위의 4가지 유형은 가변성 요소들인 가변점과 가변치의 의존성에 대한 구분이며, 가변점과 가변치들 사이의 상호의존성은 가변치와 가변점 사이의 의존관계, 가변치들 사이의 의존관계, 그리고 가변점과 가변점 사이의 의존관계로 구분된다. 특히, 가변치는 반드시 하나 이상의 가변점과 연관되어 있어야 하며, 가변치와 가변점 사이의 의존관계는 Requires나 Exclusive가 될 수 있다.

SPL 가변성 모델은 도메인의 특성과 가변성 요소들 사이의 관계를 표현하는 구조지향의 모델이다. 반면 유스케이스는 시스템 행위나 활동(activity)의 순서를 표현하고, 이들 활동들 사이의 동적인 특성과 의존관계를 모델링한다. 따라서 SPL의 가변성 요소들을 명시적으로 표현하기 위해서 유스케이스 모델이 확장되어야 한다.

III. 유스케이스 가변성 분석을 통한 서비스 모델링

1. 유스케이스 모델의 가변성

UML 유스케이스 다이어그램은 사용자와 시스템간의 상호작용을 모델링 하는 것으로, 유스케이스 모델이 시스템의 공통적인 기능적 특성과 가변적인 특성을 나타낼 수 있다면, 유스케이스로부터 SOA를 구성하는 후보 서비스를 쉽게 도출할 수 있다.

유스케이스 다이어그램이 가변성 요소들을 표현하기 위해서는 [15]의 연구와 같이 가변점과 가변치를 표현할 수 있도록 UML 표기법을 확장해서 사용할 수 있다. 본 논문에서는 가변성 요소와 관계를 표현할 수 있도록 UML Use Case 메타모델을 확장하여 사용한다.

유스케이스는 다른 유스케이스들과 포함(include), 확장(extend), 상속(generalization) 관계로 연관되어 있다. 확장 관계를 위해 “extension point”가 추가로 사용된다 [1]. 본 논문에서 가변점은 ”extension point“에 대한 특수화(specialization)로, 가변치는 스테레오 타입 <<variant>>로 정의한다. 가변점과 가변치 사이에는 기본적으로 “mandatory” 관계로 정의하고, “option”과 “alternative”는 스테레오 타입을 사용한다. 그림 3은 확장된 유스케이스 메타모델이다.

유스케이스의 가변성은 어플리케이션마다 변경될 수 있는 속성, 로직, 워크플로우 부분이 된

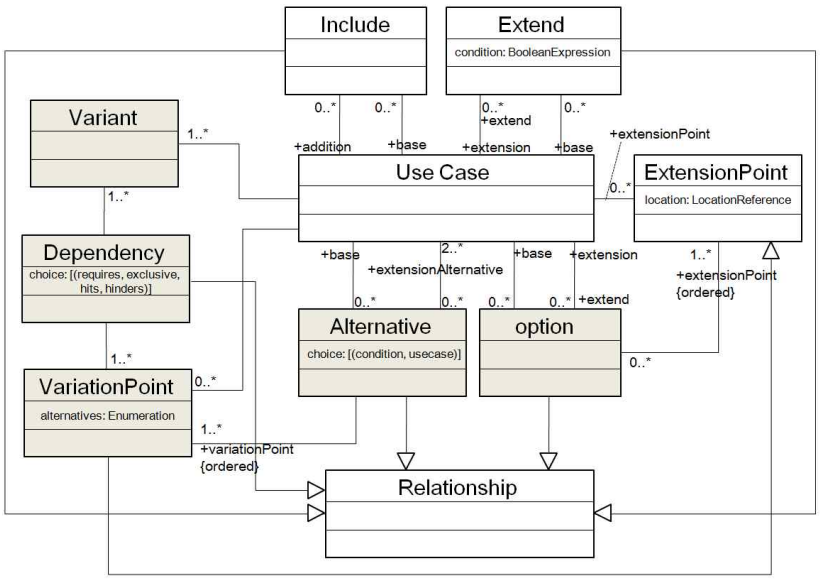

그림 3. 확장된 유스케이스 메타모델

Figure 3. Extended Use Case meta-model

다. [13]에서는 유스케이스 다이어그램에 가변성을 표현하기 위해, 사용 관점(usage view), 도메인 관점(domain view), 그리고 기술적 관점(technical view)의 3가지 주요 관점들을 정의하고 있다.

본 논문에서는 서비스 후보로서 유스케이스의 가변성을 고려하기 위해, 사용 관점을 이용한다. 사용 관점은 사용자(end-user)와 시스템의 제공자 또는 관리자의 문제를 해결하기 위해 필요한 고려사항들에 대한 것이다. 즉, 사용 관점은 시스템이 사용자에게 제공해야 하는 모든 기능(functionality)을 다룬다. 사용 관점에서 필요한 기능은 특정 정보 또는 데이터를 요구하게 된다. 이 경우 직접적인 의존관계와 함께 우회적인(indirect) 의존관계가 존재할 수 있다. 우회적인 의존관계는 추적이 가능하기 때문에, 직접적인 의존 관계들뿐만 아니라 전이(transitive)를 통해서 발견할 수 있는 모든 우회적인 의존관계들도 함께 고려해야 한다.

2. 가변성을 이용한 서비스 모델링

서비스 모델링은 서비스 식별을 통해 후보 서비스를 도출하고, 서비스 인터페이스 정의 및 정제를 통해 최종 서비스를 정의함으로써 이루어진다. 서비스 식별은 분석설계자의 경험과 직관이 많이 요구되는 과정이지만, 단 한 번의 분석 설계자의 결정에 따라 종료되는 프로세스가 아니며, 상호작용과 설계를 거치는 정제의 과정을 통해 보다 나은 서비스를 도출할 수 있도록 하는 것이 중요하다.

먼저 유스케이스 가변성 모델을 이용한 서비스 후보 식별은 다음과 같은 과정을 거친다.

- 유스케이스 다이어그램을 통해 사용자와 시스템 간의 상호작용을 분석하면서 서비스 경계를 선택한다.

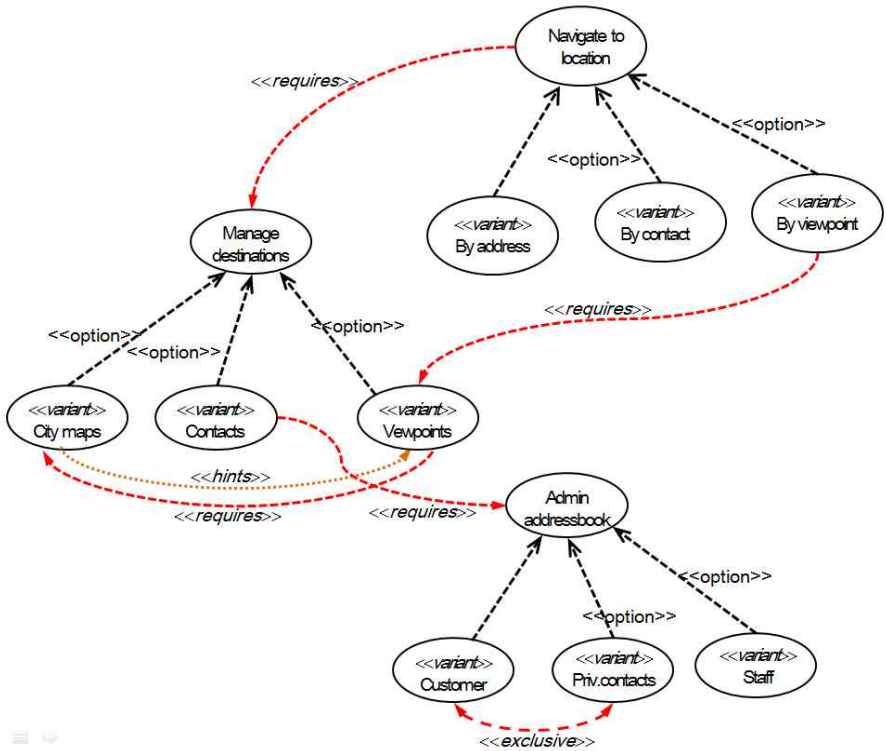

그림 4. 가변성 요소를 포함한 유스케이스 예제

Figure 4. An example of Use Case diagram including variability elements

- 액터별로 한 액터가 상호작용하는 유스케이스들을 묶어 하나의 서비스로 식별하거나, 액터들이 비슷하게 상호작용하는 유스케이스들을 묶어 하나의 서비스로 식별한다. 이때 액터별로 인터페이스는 분리 할 수 있다. 액터의 활동 단위로 서비스 오퍼레이션 후보가 된다.
- 유스케이스들 간의 관계에 따라 서비스를 식별하는 경우, 유스케이스 사이의 의존관계를 고려하여 서로 응집력이 높은 유스케이스들을 하나의 단위로 클러스터링 해야 한다. “mandatory″와 ”requires″의 경우 하나의 서비스 단위로 정의되어야 서비스 사이의 약결합 특성을 유지할 수 있다.

각 서비스 후보는 서비스 인터페이스 정의 및 오퍼레이션 정제를 통해 서비스로 식별되어 WSDL 명세를 작성하게 된다. 다음은 서비스 정제를 위한 가이드라인이다.

- 유스케이스에서 기본흐름, 선택흐름, 예외흐름의 각 스텝이 비즈니스적인 의미를 가지는 독립적인 단위 업무를 수행하는 경우, 하나의 서비스 오퍼레이션으로 정의한다.
- 액터에 의해 수행되는 하나의 활동이 두 개 이상의 비즈니스적인 의미를 갖는 단위 업무를 수행하는 경우, 두 개 이상의 서비스 오퍼레이션으로 정의한다.

비즈니스 프로세스 모델의 수준이 너무 상위

레벨이어서 입도를 조정해야 할 경우, 서비스 오퍼레이션을 유즈케이스 정의서에서 식별하고 액티비티 다이어그램(TO-BE)의 비즈니스 프로세스를 수행하는데 누락된 것은 없는지 검증하고 보완한다 [16].

- 서비스 오퍼레이션이 시스템에서 처리되는 경우는 정의하지 않는다.
- 하나의 서비스 오퍼레이션에서 너무 많은 기능을 수행하지 않도록 한다.
- 서비스 요청자가 한번에 요청하고, 수행할 수 있는 업무 단위로 정의한다.
- 동일한 위치(장소)에서 한번에 수행할 수 있는 활동 단위로 정의한다.
- 사용자에게 비즈니스적인 의미를 지니는 최소의 처리단위로 정의한다.

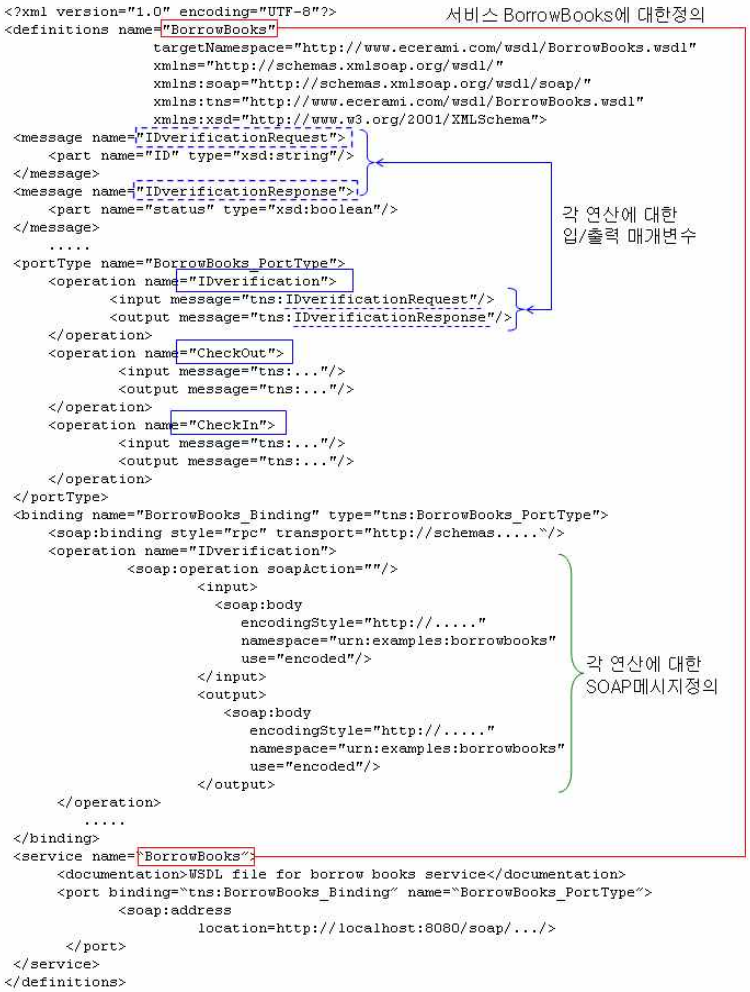

그림 5. 서비스의 WSDL 명세

Figure 5. WSDL description for a service

정제된 서비스 오퍼레이션을 클러스터링 하여 재사용성을 고려하여 최종 서비스로 도출한다. 이런 정제 과정은 각 서비스의 역할과 기능을 사용자에게 좀 더 분명하게 하기 위해 필요하다. 유스케이스 가변점과 의존관계를 바탕으로 서비스들 사이의 상호작용을 분석함으로써 적절한 추상화 수준을 유지하는 서비스들을 정의하기 위한 것이다. 필요하다면, 기존의 서비스들을 수정하거나 제거할 수 있고, 새로운 서비스를 생성 할 수도 있을 것이다. 이런 반복적인 정제 과정을 거쳐 그림 5와 같은 최종적인 서비스들에 대한 WSDL 명세가 생성된다.

IV. 분석 및 평가

다음 표 1과 같이 서비스 모델링에 대한 다양한 방법론과 본 논문에서 제안한 방법과 비교해 보았다. 기존의 방법들은 매우 다양한 서비스 분류 체계와 이론적 배경들을 가지고 있다. 서비스를 식별하기 위해 상향식 접근방법을 사용하여 정보 시스템을 활용하기도 하며, 하향식으로 비즈니스 프로세스를 포함하는 사용자 요구사항을 분석하는 절차를 거치기도 한다. 기존의 방법론과 비교해 본 결과, 본 논문에서 제안한 서비스 모델링 절차는 서비스 후보를 도출하는 과정에서 비즈니스 측면을 더 고려하였다. 유스케이스의 변경 가능한 특징들로부터 공통적인 부분을 분리하여 그룹화 함으로써, 적응력 높은 서비스를 도출하도록 하였다. 이를 통해 적응력 높고 적절한 추상화 수준을 유지하는 서비스를 도출함으로써, 보다 유연한 SOA 서비스 기반 어플리케이션 구축을 유도할 수 있다.

본 논문에서 제안한 방법을 검증하기 위해 간

표 1. 서비스 식별 방법의 비교
Table 1. Comparison of approaches regarding service identification

	Jain et al. [17]	T. Erl [3]	M. Bell [10]	제안 방법
이론적 배경	상향식. 기존 정보 시스템과 기능에 대한 분석 방법 도입	절충식. 비즈니스 프로세스와 정보 시스템 병행 분석	상향식. 기존 정보 시스템의 분석 방법 도입	하향식. 유스케이스 모델과 비즈니스 프로세스 분할 방법 도입
서비스 분류 체계	단위 서비스와 조합 서비스의 묵시적 구분	11 서비스 유형 분류	6 가지 개념적 서비스 카테고리	서비스 계층구조 형성: 서비스, 활동, 오퍼레이션
SOA 모델링	서비스 개발 프로세스에 초점. 서비스 모델링은 부수적인 단계	모델링 가이드라인 포함. SOA 개발 방법론에 초점	서비스 카테고리와 명세화에 초점	서비스 식별과 정제를 통한 서비스 모델링
특징	7 단계로 구성. 간단한 문서	과도한 문서화. 사례 연구를 통한 가이드라인 제공	의사결정 트리를 사용한 형식화	고적응성 서비스 모델링 방법 제공

단한 송장발부를 포함한 주문처리 시스템(Order-to-invoice business system)에 대한 사례 연구를 수행하고 있다. 이 작업의 목적은 주문처리 시스템의 유스케이스 모델의 가변성 모델을 생성하여, 서비스 식별 작업에서 적절한 추상화 수준을 유지하는 서비스들을 얼마나 수월하게 도출할 수 있는지를 평가하는 것이다. 이를 위해 기존의 방법론에서 제안한 가이드라인 및 도구를 통해 생성된 서비스 모델과 본 논문에서 제안한 유스케이스 가변성 모델로부터 생성된 서비스의 오퍼레이션 수준을 비교하게 될 것이다.

V. 결론 및 향후 연구과제

본 논문에서는 SOA 분석 및 설계를 위한 서비스 모델링 방법을 제안하였다. 적합한 추상화 수준에서 잘 정의된 인터페이스를 갖는 서비스를 추출을 목표로 한다. 이를 위해 유스케이스의 가변성 분석을 이용한 서비스 식별 및 정의 과정을 제안하였다. 제안된 방법은 유스케이스 모델의 가변성 특징을 모델링하여, 후보 서비스를 도출하고 정제한 후 서비스 명세를 생성하는 과정을 포함하고 있다.

서비스 식별 과정에서 발생한 오류는 상세 설계 및 구현에 영향을 주고 전파되므로, 서비스 식별은 SOA 솔루션의 모델링에서 가장 중요한 활동이다. 따라서 향후 연구의 방향은 제안된 식별 방법을 형식화(formalization)하고, 식별 패턴을 체계화하여 통합된 모델링 방법으로 개선하는 것이다.

참 고 문 헌

[1] OMG, "Unified Modeling Language, Ver. 1.5", Available at http://www.uml.org, March 2003.

[2] Yukyong Kim and Kyung-Goo Doh, "The Service Modeling Process Based on Use Case Refactoring", Lecture Notes in Computer Science, vol. 4439, pp. 108-120, 2007.

[3] Erl, T., SOA Principles of Service Design, Prentice-Hall PTR, 2007.

[4] O. Zimmermann, V. Doubrovski, J. Grundler, and K. Hogg, "Service-Oriented Architecture and Business Process Choreography in an Order Management Scenario", Proc. of 20th conference on OOPSLA, pp.301-312, San Diego, CA, USA, Oct. 16-20, 2005.

[5] A. Arsanjani, "Service-Oriented Modeling and Architecture: How to identify, specify, and realize services for your SOA", IBM developerWorks, Nov. 2004.

[6] G. Kreizman, "How to Build a Business Case for Service-Oriented Development of Applications in Government", Gartner Industry Research, Sep. 2005.

[7] K. Mittal, "Service Oriented Unified Process (SOUP)", IBM Journal, June 2005.

[8] A. Parikh, R. Pradhan and N. Shah, "Modeling of Web Services: A Standards-Based Approach", Software Magazine, May 2004.

[9] D. Fensel, D. Bussler, Y. Ding, and B. Omelayenko, "The Web Service Modeling Framework WSMF", Journal of Electronic Commerce Research and Applications, vol. 1, no.2, pp.113 - 137, 2002.

[10] M. Bell, Service-Oriented Modeling : Service Analysis, Design, and Architecture, John Wiley & Sons Inc., 2008.

[11] 이현주, 최병주, 이정원, "서비스 지향 아키텍처를 위한 컴포넌트기반 시스템의 서비스 식별", 정보과학회논문지 : 소프트웨어 및 응용, 제35권 제2호, 70-80쪽, 2008년 2월.

[12] 김유경, 주영도, "SOA 적용을 위한 비즈니스 프로세스 모델 기반의 서비스 모델링",

한국소프트웨어감정평가학회 논문지, 제4권, 제1호, 27-40쪽, 2008년 5월.

[13] S. Bühne, G. Halmans, and K. Pohl, “Modeling Dependencies between Variation Points in Use Case Diagrams”, Proc. of 9th Workshop on Requirements Engineering - Foundations for Software Quality, pp.59-69, 2003.

[14] K. Kang et al, "Feature-Oriented Domain Analysis (FODA) Feasibility Study", Technical Report No. CMU/SEI-90-TR-2, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, November 1990.

[15] G. Halmans, and K. Pohl, “Communicating the Variability of a Software Product Family to Customers”, Software and Systems Modeling, vol. 2, pp.15-36, Mar. 2003.

[16] 한상우, 박선희, 노재호, “Service-Oriented Architecture 적용을 위한 서비스 식별 기법”, 한국정보과학회지, 제24권, 제11호, pp.27-31, 2006.

[17] H. Jain, H. Zhao, N. Chinta, "A Spanning Tree Based Approach to Identifying Web Services", International Journal of Web Services Research, vol. 1, no. 1, pp.1 - 20, 2004.

저 자 소 개

김 유 경(金裕卿)

1991년 2월 숙명여자대학교 수학과 이학사
1994년 8월 숙명여자대학교 전산학과 이학석사
2001년 8월 숙명여자대학교 컴퓨터과학과 이학박사
2001년 9월～2005년 8월 숙명여자대학교 정보과학부 초빙교수
2005년 9월～2006년 8월 미국 University of California Davis, Post-doc.
2006년 9월～현재 한양대학교 컴퓨터공학과 연구교수
<주관심분야>SOA, 웹서비스, SW품질평가

주 영 도(朱映度)

1983년 한양대학교 전자통신학과(학사)
1988년 미국 University of South Florida 컴퓨터공학과 (석사)
1995년 미국 Florida State University 컴퓨터과학과(박사)
1984년～1985년 한국무역협회 전자계산소 시스템 프로그래머
1995년～2000년 KT 연구개발본부 연구 팀/실장
2000년～2005년 시스코 시스템즈 코리아 통신사업부 담당 상무
2005년～2006년 화웨이 기술 코리아 부사장
2007년～현재 강남대학교 컴퓨터미디어공학부 교수(e-mail: ydjoo@kangnam.ac.kr)
<주관심분야> 데이터베이스, 지능형시스템, 초고속정보통신망, 통신망관리등

도 경 구

1980년 한양대학교 산업공학과(학사)
1987년 미국 아이오와주립대학교 전산학과(석사)
1992년 미국 캔사스주립대학교 전산학과(박사)
1993.4~1995.8, 일본 Aizu 대학, 교수
2000.9~2001.12, 스마트카드연구소 대표이사
2005.3~2006.2, 미국 캘리포니아대학 데이비스캠퍼스 방문교수
1995.9~현재, 한양대학교 안산캠퍼스 컴퓨터공학과 교수