Journal of Korea Multimedia Society Vol. 11, No. 5, May 2008(pp. 710-727)

서비스지향 컴퓨팅 시스템으로의 확장을 위한 컴포넌트 기반의 서비스 식별

최미숙[†], 이서정[††], 이종석[†††], 양승원[††††]

요 약

서비스지향 컴퓨팅 시스템은 시스템의 기능적 단위인 서비스들을 재사용함으로 해서 개발 시간과 노력을 줄이는 특성 때문에 분산 환경이 일반화 되면서 더욱 중요하게 부각되고 있다. 서비스의 재사용은 서비스들 간의 느슨한 결합에 의하여 효과적으로 이루어질 수 있다. 그러나 상속 및 포함 관계와 같은 객체지향 시스템의 강한 연관 관계들은 객체들 간에 강한 결합을 생성한다. 상속 관계와 포함 관계가 없는 컴포넌트 기반의 시스템은 컴포넌트들 간에 느슨한 결합을 생성한다. 그리하여 컴포넌트 인터페이스들에 의해서 제공된 기능을 사용해서 실시간에 서비스지향 시스템의 서비스를 실현한다. 따라서 컴포넌트기반 시스템은 기능적 서비스 단위들을 효율적으로 제공하기 위하여 서비스지향 컴퓨팅 시스템으로 확장될 필요가 있다. 또한, 서비스지향 컴퓨팅 시스템을 지원하는 기존의 방법들은 서비스 계층의 명확한 분류 및 서비스 계층에 따른 명확한 서비스 식별 가이드라인 그리고 서비스 계층 간의 매핑 방법을 제시하지 않고 있다. 따라서 본 논문에서는 비즈니스 관점의 서비스와 구현 관점의 서비스를 계층으로 나누어 분류하고 서비스 식별 가이드라인 및 각 계층의 서비스들 간의 매핑을 제안한다. 즉, 우리는 서비스 계층과 다양한 크기의 서비스 식별 방법을 연구하고, 각 계층의 서비스들 간의 매핑 방법을 도출한다. 이를 기반으로 기존 컴포넌트 기반 시스템을 서비스 지향 컴퓨팅 시스템으로 확장할 수 있다.

Service Identification of Component-Based For Extending Service-Oriented Computing System

Misook Choi[†], Seojeong Lee[††], Jongsuk Lee[†††], Seungwon Yang[††††]

ABSTRACT

Service-oriented computing systems have been issued by their properties of reducing software development time and effort by reusing functional service units. The reusability of services can effectively promote through loose coupling between services. But strong associations of object-oriented systems such as inheritance and aggregation create a rather tight coupling between objects. The component-based systems without inheritance and aggregation create a loose coupling between components. Thus components provide service realization at runtime using the functionality provided by their interfaces. Therefore legacy component-based systems need to have service-oriented computing concept in order to support functional service units efficiently. Also, conventional methods for service-oriented computing system have not suggested the clear classification of service layers, the clear service identification guideline introducing service layers and a service mapping method between serviceces of each layer. Therefore we suggest the service classification and the identification guideline of business view and implementation view introducing layers and propose a mapping between two views. That is, we research service layers, service identification, diversified service sizes and a service mapping method between services of each layer. This can be applied to legacy component-based system to extend to the service-oriented computing system.

Key words: Service Identification(서비스 식별), Service Layers(서비스 계층), Service Units(서비스 단위), Component-Based System(컴포넌트 기반 시스템)

※ 교신저자(Corresponding Author) : 최미숙, 주소 : 전북 완주군 삼례읍 후정리 490(565-701), 전화 : 063)290-1452, FAX : 063)290-1453, E-mail : khc67_kr@hanmail.net
접수일 : 2008년 2월 25일, 완료일 : 2008년 5월 7일
[†] 정회원, 우석대학교 기초및자연과학연구소 연구교수
[††] 정회원, 한국해양대학교 컴퓨터공학부 조교수
(E-mail : sjlee@hhu.ac.kr)
[†††] 우석대학교 컴퓨터교육과 교수
(E-mail : jong1007@nate.com)
[††††] 우석대학교 게임콘텐츠학과 교수
(E-mail : yangy123@mail.woosuk.ac.kr)
※ 이 논문은 2006년도 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임 (KRF-2006-353-D00029).

1. 서 론

현재 대학의 연구 그룹뿐 아니라 소프트웨어 회사들은 서비스지향 컴퓨팅(Service-Oriented Computing) 시스템에 상당한 관심을 기울이고 있다. 서비스지향 컴퓨팅 시스템의 소프트웨어 서비스들은 새로운 표준적 아키텍처 스타일이 되고 있고 소프트웨어 공학의 차세대 스텝으로서 자리잡아가고 있다[1]. 소프트웨어 서비스들은 네트워크 상에서 재사용이 가능한 공유 서비스의 집합으로, 서비스의 인터페이스(interface)와 구현(implementation)을 분리시켜 애플리케이션을 개발한다. 서비스 인터페이스와 구현부를 분리시킴으로 해서 변화하는 사용자의 요구에 애플리케이션들이 쉽게 대체될 수 있고 구현부는 해체됨이 없이 서비스의 조합과 재사용에 의해서 진화될 수 있다[2]. 즉, 서비스지향 컴퓨팅 시스템은 서비스들을 재사용함으로 해서 개발 시간과 노력을 줄이고 빠르게 변화하는 환경에 효과적으로 대처할 수 있다. 서비스지향 컴퓨팅 시스템은 서비스들 사이에 상속 관계와 포함 관계를 지원하지 않으며 서비스들 사이에 느슨한 결합을 통하여 재사용성, 자율성, 상호운용성, 조합성을 증진하는 방법이다. 객체지향 시스템은 클래스들 간에 상속 관계와 포함 관계등을 지원하여 클래스들 사이에 강한 연관 관계를 생성하고 비즈니스 모델링 관점에서, 클래스는 서비스의 크기(granularity) 및 추상화 레벨이 너무 낮다[3]. 따라서 객체지향 시스템은 서비스 지향 시스템의 특성에 부합되지 않는다. 컴포넌트기반 시스템은 상속 관계 및 포함 관계와 같은 강한 연관 관계가 없어 컴포넌트들 사이에 느슨한 결합을 생성하고, 컴포넌트 인터페이스들에 의해서 제공된 기능을 통해 실시간에 서비스를 실현한다[4]. 따라서 서비스지향 컴퓨팅 시스템을 위한 서비스의 설계는 컴포넌트기반 시스템을 통해서 이루어지는 것이 효율적이므로, 컴포넌트 기반 시스템은 서비스 지향 컴퓨팅 시스템으로 확장될 필요가 있다. 또한, 서비스지향 시스템의 특징과 이점을 명백히 나타내려면 서비스지향을 전사적 기업 환경에 효율적으로 적용하고 전파시키는 수단이 필요하다. 이것은 서비스 계층의 추상화를 통해서 바로 실현될 수 있고[5] 각 서비스 계층은 느슨한 결합을 가진 서비스를 포함하여야 한다. 그러나 서비스지향 시스템을 지원하는 기존의 방법들[3,4,6,7,8,9]은 서비스 계층의 명확한 분류 및 각 서비스 계층에 대한 명확한 서비스 식별 가이드라인을 제시하지 않는다. 또한, 서비스 입자의 크기 및 서비스 관점 차이에 의한 의미적 불일치를 해결하고 서비스의 효율적 재사용 및 관리를 위해서, 각 서비스 계층 간 매핑이 필요하다.

따라서, 본 논문에서는 비즈니스 관점의 서비스로부터 구현 관점까지 서비스를 계층으로 나누어 분류하고 각 계층에 대한 서비스 식별 가이드라인 및 서비스 계층 간의 매핑을 제안한다. 본 논문의 2장에서는 컴포넌트기반 시스템과 서비스지향 시스템과의 관계, 서비스지향 아키텍처 그리고 기존의 방법론들을 제시한다. 3장에서는 서비스 계층의 분류 및 서비스 계층에 따른 서비스 식별 가이드라인 그리고 서비스 계층 간의 매핑을 제시한다. 4장에서는 본 논문에서 제안한 방법의 효율성을 분석 평가한다. 5장에서는 본 논문의 결론을 기술한다.

2. 관련 연구

2.1 컴포넌트기반 시스템과 서비스지향 시스템

서비스지향 시스템은 컴포넌트기반 시스템에서 추상화 레벨이 한층 더 높아진 서비스들의 조합에 의해 구축되는 소프트웨어 개발 환경이다. 따라서, 컴포넌트기반 시스템의 인터페이스들을 비즈니스 관점으로 추상화하고, 그들을 중심으로 컴포넌트 기반 시스템을 재 사용하여 서비스지향 시스템을 구축한다. 즉, 서비스지향 시스템 관점에서 보면, 비즈니스 수준의 서비스를 효율적으로 제공하기 위해서 컴포넌트기반 시스템의 컴포넌트를 개발하고 구현한다.

2.2 서비스지향 아키텍처

서비스지향 컴퓨팅 시스템을 위한 서비스지향 아키텍처는 서비스 계층을 별도로 구성한다는 측면에서 다른 아키텍처와 차이가 있다. 시스템의 전체 구조에서 서비스 계층은 기존의 비즈니스 계층과 애플리케이션 계층인 컴포넌트 계층 사이에 추상화된 형태로 존재한다. 서비스 계층이 비즈니스 계층과 애플리케이션 계층인 컴포넌트 계층 사이에 위치함으로써, 서비스는 비즈니스 프로세스 로직뿐만 아니라 물리적인 애플리케이션 로직도 캡슐화할 수 있다[5].

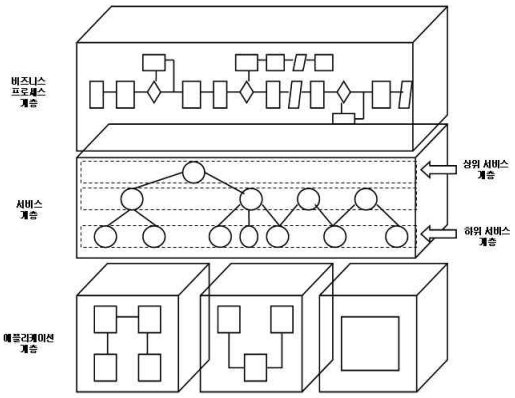

그림 1. 서비스 계층

그림 1에서 제시하듯이 비즈니스 계층과 애플리케이션 계층을 연결하는 중간 계층의 서비스는 상위 서비스가 하위 서비스를 캡슐화하며 서비스의 추상화 정도에 따라 서비스 계층의 구성은 달라진다[5]. 따라서, 서비스의 추상화 정도에 따른 서비스 계층의 분류 및 각 서비스 계층의 서비스 식별 가이드라인이 필요하고 비즈니스 관점의 서비스 계층과 구현 관점의 애플리케이션 계층에 대한 의미적 불일치를 해결하기 위해 매핑이 필요하다. 또한, 서비스 계층의 서비스는 애플리케이션 계층의 재사용 단위인 클래스나 컴포넌트 보다는 기능적 재사용 단위인 서비스 관점의 좀 더 높은 추상화가 필요하다.

2.3 기존의 서비스 식별 방법

[3]의 SOAD(**S**ervice-**O**riented **A**nalysis and **D**esign)는 OOAD, EA, BPM 방법을 혼합 적용하여 서비스지향 아키텍처를 설계하는 방법론이다. SOAD는 서비스지향 아키텍처를 위하여 비즈니스 계층, 서비스 계층, 컴포넌트 계층으로 분류하고 있지만 서비스 계층의 분류 기준 및 서비스 계층에 따른 서비스 식별 방법을 구체적으로 제시하지 않는다. [4]의 SOMA(**S**ervice-**O**riented **M**odeling and **A**rchitecture)는 서비스지향 아키텍처를 지원하는 방법론이다. SOMA는 서비스지향 아키텍처를 위하여 7 단계의 계층으로 나누고 있지만 서비스 계층에 대한 분류 및 서비스 식별 방법을 구체적으로 명시하고 있지 않다. [6]은 휘처분석(Feature Analysis) 방법[10]을 도입하여, 기존의 레거시 시스템을 서비스지향으로 확장하기 위한 서비스 식별 방법을 제안하고 있다. 기존 시스템 분석 및 설계 모델은 대 다수 UML(Unified Modeling Language)기반으로 정의된다. 따라서, 레거시 시스템을 서비스지향으로 확장하기 위해 UML 기반의 모델에 휘처 모델(Feature Model)을 다시 적용해야 하는 어려움이 존재한다. 또한, [6]은 서비스 계층의 분류 및 방법을 제시하지 않는다. [7]은 액터들 사이의 상호 교류에 기반한 서비스 식별 방법으로 서비스 계층에 따른 다양한 추상화 관점의 서비스 식별 방법을 제시하지 않는다. [8]의 SOUP(**S**ervice **O**riented **U**nified **P**rocess)은 기능적 요구 사항과 비기능적 요구 사항을 파악한 후 하향식 방법(Top-Down)으로만 유스케이스 단위의 서비스를 식별하고 서비스 계층에 대한 분류 및 서비스 식별 방법을 구체적으로 명시하고 있지 않다. [9]은 서비스 계층에 대한 분류를 제시하지 않고 유스케이스 단위의 서비스만을 식별한다. 본 논문에서 제시한 기존의 서비스 식별 방법들은 서비스 계층 간의 매핑 방법을 제시하고 있지 않다.

3. 컴포넌트기반의 서비스 식별

본 장은 컴포넌트기반의 시스템을 서비스지향 컴퓨팅 시스템으로 효율적으로 확장하기 위하여 서비스 계층의 분류 및 서비스 계층에 따른 서비스 식별 가이드라인을 제시한다. 또한 각 서비스 계층 간 매핑을 제시한다.

3.1 서비스 계층의 분류

본 논문은 서비스 계층을 비즈니스 관점, 시스템 관점 그리고 구현 관점으로 나누어 분류한다. 비즈니스 관점에서는 서비스를 시스템 컴포넌트 서비스 계층, 비즈니스 컴포넌트 서비스 계층 그리고 유스케이스 서비스 계층으로 나누어 정의하고 시스템 관점에서는 이벤트 서비스 계층으로 정의한다. 또한 구현 관점은 컴포넌트 계층으로 정의 한다.

3.1.1 비즈니스 관점의 시스템 컴포넌트 서비스 계층

서브 시스템 단위의 서비스로 업무 관점의 기능 분할을 통해서 시스템을 적절한 단위 업무로 분할한 후에 그 단위 업무가 비즈니스 컴포넌트 서비스들에 의해서 실현될 경우 그 서비스를 시스템 컴포넌트 서비스 계층의 시스템 컴포넌트 서비스로 정의한다.

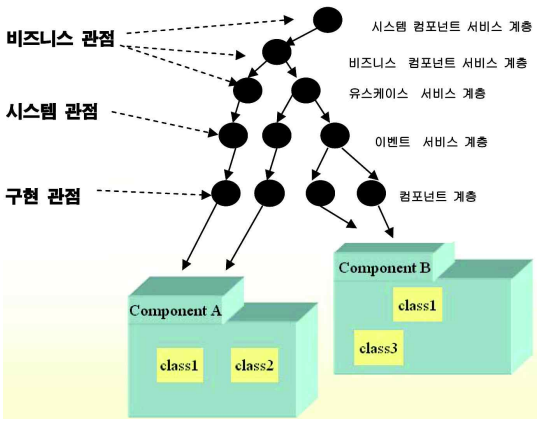

그림 2. 본 논문에서 제안한 서비스 계층

시스템 컴포넌트 서비스 계층은 가장 큰 입자이면서 가장 느슨한 결합도를 가지는 서비스들로 구성된 가장 상위 계층이다. 시스템 컴포넌트 서비스 계층의 시스템 컴포넌트 서비스의 실현은 비즈니스 컴포넌트 서비스들의 조합에 의해서 이루어지므로 비즈니스 컴포넌트 서비스 계층과의 효율적인 추적과 매핑이 가능하다.

3.1.2 비즈니스 관점의 비즈니스 컴포넌트 서비스 계층

상호 의존이 강한 유스케이스들을 그룹화한 단위를 비즈니스 컴포넌트 서비스 계층의 비즈니스 컴포넌트 서비스로 정의한다. 비즈니스 컴포넌트 서비스 계층은 상호 의존이 강한 유스케이스들을 그룹화한 단위의 서비스들이므로 느슨한 결합도를 가지고 있다. 따라서 비즈니스 컴포넌트 서비스 단위의 조합이나 분리 그리고 새로운 비즈니스 컴포넌트 서비스의 추가에 의해서 서비스를 유연하게 조립할 수 있으므로 비즈니스 컴포넌트 서비스의 재 사용이 용이하다. 비즈니스 컴포넌트 서비스 계층의 비즈니스 컴포넌트 서비스의 실현은 유스케이스들에 의하여 이루어지므로 유스케이스를 단위로한 유스케이스 서비스 계층과의 효율적인 매핑과 추적이 가능하다.

3.1.3 비즈니스 관점의 유스케이스 서비스 계층

유스케이스 단위의 서비스를 유스케이스 서비스 계층의 유스케이스 서비스로 정의한다. 비즈니스 관점에서 유스케이스는 단위 서비스 후보가 될 수 있다. 그러나 기존의 서비스 식별 방법들은 유스케이스가 가장 좋은 서비스 후보가 된다[4]라고만 명시하고 있지 서비스로 식별할 수 있는 구체적인 방법은 언급하고 있지 않다. 따라서 본 논문의 3.2.4절에서 유스케이스 서비스 식별 가이드라인을 구체적으로 제시한다. 본 논문에서는 유스케이스를 실현하기 위해서 하나의 이벤트를 포함하는 유스케이스를 단위 유스케이스 서비스로 정의하고 여러 개의 이벤트를 포함하는 유스케이스는 복합 유스케이스 서비스로 정의한다. 따라서 유스케이스 서비스 계층의 유스케이스 서비스의 실현은 시스템 관점의 이벤트들에 의해서 이루어지므로 이벤트를 단위로한 이벤트 서비스 계층과의 효율적인 매핑과 추적이 가능하다.

3.1.4 시스템 관점의 이벤트 서비스 계층

이벤트 단위의 서비스를 이벤트 서비스 계층의 이벤트 서비스로 정의한다. 액터의 요청에 기반한 서비스로서 가장 기본이 되는 원자 서비스를 구성한다. 이벤트 계층의 서비스는 상속관계나 포함관계가 없어 느슨한 결합도를 가지고 있는 컴포넌트 인터페이스에 의해서 실현된다. 따라서 이벤트 서비스 계층의 이벤트 서비스의 실현은 구현 관점의 컴포넌트 인터페이스들에 의해서 이루어지므로 구현 계층인 컴포넌트 계층과 효율적인 매핑과 추적이 가능하다. 이벤트 단위의 서비스의 조합이나 분리 그리고 새로운 이벤트 서비스의 추가에 의해서 서비스를 유연하게 조립할 수 있으므로 이벤트 계층의 서비스에 대한 재 사용이 용이하게 이루어질 수 있다.

3.1.5 구현 관점의 컴포넌트 계층

컴포넌트기반의 시스템은 기능 및 구조적으로 밀접한 연관관계가 있는 클래스들을 그룹화하여 독립적인 단위로 개발한다. 컴포넌트 기반 시스템의 기능 실현은 각 컴포넌트가 포함하고 있는 컴포넌트 인터페이스 메소드에 의해서 실행이 되거나 컴포넌트 간의 인터페이스 메소드들의 상호협력에 의해서 이루어 진다. 따라서 이벤트 서비스 계층의 이벤트 서비스 실현은 컴포넌트기반 시스템이 포함하고 있는 컴포넌트 인터페이스들에 의해서 이루어진다.

3.2 서비스 계층의 서비스 식별

본 절은 서비스 계층에 따른 서비스 식별 가이드라인을 제시하고 온라인 도서 주문 시스템[11]을 중심으로 서비스 식별 방법을 설명한다. 온라인 도서

주문 시스템은 고객이 인터넷을 사용하여 온라인 서점에서 도서 목록을 검색하고 원하는 도서를 장바구니에 담아 주문하는 시스템이다. 다음 그림 3은 온라인 도서 주문 시스템의 유스케이스 다이어그램을 제시한다.

3.2.1 컴포넌트기반의 서비스 식별

본 논문에서 제안하고 있는 컴포넌트기반의 서비스 식별을 위해서 세가지 경우가 존재할 수 있다. 첫번째는 컴포넌트기반의 시스템 분석, 설계 모델이 존재할 경우, 두번째는 객체지향기반의 시스템 분석, 설계 모델이 존재할 경우, 세번째는 시스템을 새롭게 분석, 설계해야할 경우이다. 첫번째의 경우와 두번째의 경우, 모두 본 논문에서 제안한 식별 가이드라인을 적용하여 서비스를 식별할 수 있다. 세번째의 경우는 컴포넌트기반 시스템 분석, 설계 모델을 만든 후 적용한다. 기존의 컴포넌트기반 모델이 잘못 설계되었을 경우에도 본 논문에서 제안한 서비스 식별 가이드라인을 적용하여 더욱 정확하게 서비스를 식별할 수 있다. 본 논문에서는 서비스 계층에 따른 서비스 식별을 위해서 상향식 방법과 하향식 방법을 모두 적용하고 세부 사항은 다음 표 1에서 제시 한다.

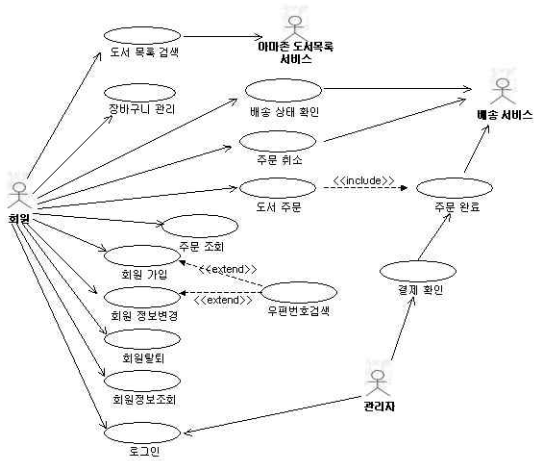

그림 3. 온라인 도서 주문 시스템의 유스케이스 다이어그램

표 1. 서비스 식별을 위한 상향식/하향식 방법의 적용

서비스	시스템 컴포넌트 서비스	비즈니스 컴포넌트 서비스	유스케이스 서비스	이벤트 서비스
식별 방법	상향식, 하향식 방법	하향식 방법	하향식 방법	상향식 방법

서비스 식별을 위한 프로세스는 시스템 컴포넌트 서비스 후보 식별 → 비즈니스 컴포넌트 서비스 식별 → 시스템 컴포넌트 서비스 식별 → 이벤트 서비스 식별 → 유스케이스 서비스 식별순으로 진행한다.

3.2.2 시스템 컴포넌트 서비스

본 절은 시스템 컴포넌트 서비스 식별을 위한 가이드라인을 제시한다.

가이드라인 1 하향식 방법에 의한 기능 분할을 통해서 시스템을 적절한 단위 업무로 분할한 후에 그 단위 업무가 비즈니스 컴포넌트 서비스들의 조합에 의해 실현될 경우 그 기능을 시스템 컴포넌트 서비스로 식별한다.

가이드라인 1에 의해서, 온라인 도서 주문 시스템의 경우, 시스템을 적절한 단위 업무로 분할해 보면 시스템은 도서 주문을 위한 온라인 도서주문관리, 직원의 급여등등에 관한 회계관리, 직원의 인사관리를 위한 인사관리등등이 될 수 있다. 이때 온라인 도서주문관리는 3.2.3절의 그림 7에서 제시하고 있듯이 회원관리, 주문관리, 도서관리, 우편번호관리등의 비즈니스 컴포넌트 서비스들의 조합에 의하여 그 기능이 실현된다. 따라서, 온라인 도서주문관리는 시스템 컴포넌트 서비스가 된다. 온라인 도서 주문 시스템의 경우 식별된 시스템 컴포넌트 서비스는 그림 4와 같다.

3.2.3 비즈니스 컴포넌트 서비스

비즈니스 컴포넌트 서비스는 상호 의존이 강한 유스케이스들의 조합에 의하여 생성되는 서비스이다. 본 절에서는 비즈니스 컴포넌트 서비스 계층의 서비스를 식별하기 위한 가이드라인을 유스케이스 관점과 비즈니스 컴포넌트 관점으로 나누어 제시한다.

3.2.3.1 유스케이스 관점

유스케이스 관점에 의한 비즈니스 컴포넌트 서비

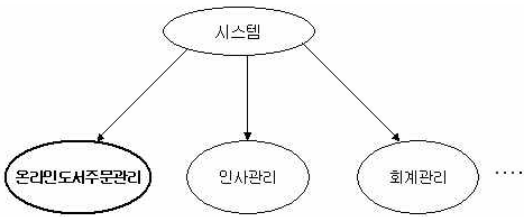

그림 4. 온라인 도서주문 시스템의 시스템 컴포넌트 서비스

스 식별은 1 단계로 유스케이스를 그룹화하여 도출된 결과를 2 단계의 유스케이스 그룹화 가이드라인을 적용하여 비즈니스 컴포넌트 서비스를 식별한다.

<1 단계 유스케이스 그룹화>

가이드라인 1-1 유스케이스들이 각각 다른 프로세스의 흐름으로 처리되지만 여러 개의 클래스들을 참조하지 않고 하나의 클래스만을 동시에 같이 참조하여 기능을 실행한다면 그러한 유스케이스들은 한 클래스만을 대상으로한 기능 실행으로 서로 분리될 수 없으므로 이러한 유스케이스들은 그룹화 한다.

가이드라인 1-2 유스케이스 다이어그램에서 유스케이스들의 관계가 포함관계(include relationship)를 갖는다면 서로 분리될 수 없으므로 그러한 관계의 유스케이스들은 그룹화 한다.

가이드라인 1-3 각각의 유스케이스들이 공유하는 유스케이스들을 그룹화하여 공유 비즈니스 컴포넌트 서비스로 식별한다.

<2 단계 유스케이스 그룹화에 의한 비즈니스 컴포넌트 서비스 식별>

2 단계는 1 단계 유스케이스 그룹화 가이드라인에 의해 도출된 유스케이스들 사이의 클래스 및 클래스 참조 관계의 유형을 고려하여 비즈니스 컴포넌트 서비스를 식별한다.

가이드라인 1-4 임의의 유스케이스가 포함하는 클래스나 클래스들을 다른 유스케이스도 완전히 포함하여 참조하고 참조되는 클래스의 유형이 같다면 그러한 유스케이스들은 참조 클래스에 의한 포함관계이므로 그룹화하여 비즈니스 컴포넌트 서비스로 식별한다.

가이드라인 1-5 임의의 유스케이스가 포함하는 클래스나 클래스들을 다른 유스케이스도 완전히 포함하여 참조하지만 클래스의 참조 유형이 다르다면 그러한 유스케이스들은 서로 다른 참조 클래스 유형에 의하여 완전한 포함관계가 아니고 연관관계이다. 따라서 그들은 분리하여 각각 비즈니스 컴포넌트 서비스로 식별한다.

가이드라인 1-6 임의의 유스케이스가 포함하는 클래스를 다른 유스케이스가 부분적으로 참조한다면 그러한 관계의 유스케이스들은 클래스 부분 참조에 의한 연관관계이므로 서로 분리하여 각각의 비즈니스 컴포넌트 서비스로 식별한다.

가이드라인 1-7 1 단계의 결과로 도출된 유스케이스들이 2 단계의 가이드라인에 의해서 다른 유스케이스들과 그룹화 되지 않는다면 그들은 각각 비즈니스 컴포넌트 서비스로 식별한다.

시스템 컴포넌트 서비스 중 온라인 도서 주문관리 시스템의 경우, 그림 3의 유스케이스 다이어그램과 아래 그림 5의 비즈니스 객체 모델을 참조하여 비즈니스 컴포넌트 서비스를 식별하기 위한 사례를 다음에서 제시한다.

<1 단계 유스케이스 그룹화의 사례>

가이드라인 1-1의 경우

회원가입, 회원정보변경, 회원탈퇴, 로그인 유스케이스는 프로세스의 흐름은 다르지만 모두 하나의 클래스인 회원정보를 참조하므로 회원관리로 그룹화한다.

도서목록검색은 하나의 클래스인 도서정보를 참조한다.

가이드라인 1-2의 경우

그림 3의 유스케이스 다이어그램을 보면 도서주문 유스케이스는 주문완료 유스케이스와 포함관계이므로 그들은 도서주문으로 그룹화 한다.

가이드라인 1-3의 경우

그림 3의 유스케이스 다이어그램을 보면 우편번호 검색 유스케이스는 회원 가입, 회원 정보 변경등의 유스케이스에서 공통으로 참조하는 유스케이스이다. 따라서 우편번호 검색 유스케이스는 공유 비즈니스 컴포넌트 서비스로서 식별된다.

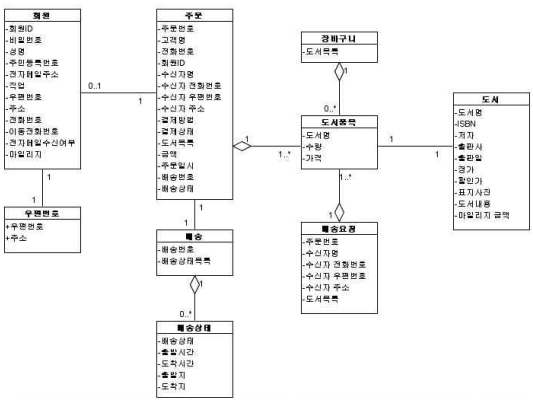

그림 5. 온라인 도서주문 시스템의 비즈니스 객체 모델

<2 단계 유스케이스 그룹화의 사례>

다음 표 2는 1 단계 유스케이스 그룹화의 결과와 각 유스케이스의 실현을 위해서 참조되는 클래스와의 관계를 제시하고 있다. 표 2에서 1이라고 표시 되어 있는 유스케이스는 1 단계 가이드라인의 적용 결과이다. 또한, 각 유스케이스의 실현을 위한 클래스의 부분 참조는 P(Partially) 그리고 완전참조는 F(Fully)로 표 2에 표시되어 있다.

가이드라인 1-4의 경우

장바구니 관리 유스케이스의 참조 클래스들인 도서, 도서품목, 장바구니 클래스들은 도서주문 유스케이스에도 포함된다. 따라서, 가이드라인 1-4에 의해서 장바구니 관리는 도서주문 유스케이스들과 그룹화 된다.

도서주문과 주문취소, 결제확인, 배송상태확인, 주문조회 유스케이스들은 참조 클래스들에 의한 포함관계이므로 그들은 그룹화 된다.

가이드라인 1-5와 1-6의 경우

장바구니 관리, 도서주문 유스케이스들은 도서정보를 부분 참조하지만 도서목록검색은 도서 클래스의 데이터를 완전 참조 즉, 포함하므로 장바구니 관리, 도서주문 유스케이스들과 도서목록검색 유스케이스와의 관계는 도서 클래스 참조의 형태에 의한 연관관계이므로 가이드라인 1-5와 1-6에 의해서 이들은 분리된다.

가이드라인 1-7의 경우

1 단계의 결과로 도출된 우편번호검색, 도서목록검색, 회원관리 유스케이스들이 1-4, 1-5, 1-6의 가이드라인에 의하여 다른 유스케이스들과 그룹화 되지 않기 때문에 그들은 각각 비즈니스 컴포넌트 서비스로 식별한다.

따라서 가이드라인 1-4에 의해서 장바구니 관리, 도서주문, 주문조회, 주문취소, 결제확인, 배송상태확인 유스케이스들은 그룹화 되어 주문관리 비즈니스 컴포넌트 서비스로 식별된다. 가이드라인 1-5와 1-6 그리고 1-7의 결과에 의해서 우편번호검색, 회원관리, 도서목록검색 유스케이스들이 각각 비즈니스 컴포넌트 서비스로 식별된다. 따라서 주문관리, 회원관리, 도서목록검색인 도서관리, 우편번호검색인 우편번호관리 비즈니스 컴포넌트 서비스가 식별된다.

표 2. 유스케이스와 데이터(클래스) 참조와의 관계

유스케이스 서비스	회원	도서	도서품목	장바구니	도서주문	배송요청	배송	배송상태	우편번호
1. 회원 관리	F								F
장바구니 관리		P	F	F					
1. 도서주문	P	P	F	F	F	F	F	F	
주문취소					F	F	F	F	
결제확인					F	F	F	F	
배송상태확인					F		F	F	
주문조회					F				
1. 도서목록검색		F							
1. 우편번호검색									F

3.2.3.2 비즈니스 컴포넌트 관점

비즈니스 컴포넌트가 포함하고 있는 유스케이스들을 그룹화한 단위를 비즈니스 컴포넌트 서비스로 식별한다. 즉, 상호 의존이 강한 클래스들을 그룹화해서 비즈니스 컴포넌트로 식별하고 각각의 기능인 유스케이스들은 비즈니스 컴포넌트를 통해서 그 기능을 실현한다. 따라서 비즈니스 컴포넌트가 포함하고 있는 유스케이스들은 상호 의존이 강한 유스케이스들이므로 유스케이스들을 그룹화한 단위는 비즈니스 컴포넌트 서비스로 식별된다. 다음 가이드라인은 상호 의존이 강한 클래스들을 그룹화 해서 비즈니스 컴포넌트로 식별한 다음 식별된 비즈니스 컴포넌트가 포함하고 있는 유스케이스들을 그룹화 해서 비즈니스 컴포넌트 서비스로 식별하기 위한 가이드라인을 제시한다.

가이드라인 2-1 클래스들 사이의 관계가 포함관계와 상속관계인 경우, 그들은 상호 의존이 강한 관계이므로 그룹화 되어야 한다. 클래스들 사이가 연관관계라 하더라도 임의의 클래스가 다른 클래스에 의해서 의존되어 데이터가 생성될 경우 데이터 생성 관계에 의해서 상호 의존이 강한 관계이므로 그룹화 되어야 한다. 따라서 상호 의존이 강한 클래스들은 그룹화 되어 비즈니스 컴포넌트로 식별되고 각 비즈니스 컴포넌트가 포함하고 있는 유스케이스들은 그룹화되어 비즈니스 컴포넌트 서비스로 식별된다.

가이드라인 2-2 가이드라인 1-3과 같다.

아래의 그림 6은 가이드라인 2-1에 의해서 클래스들 간에 상호 의존 관계를 고려하여 식별된 비즈니스

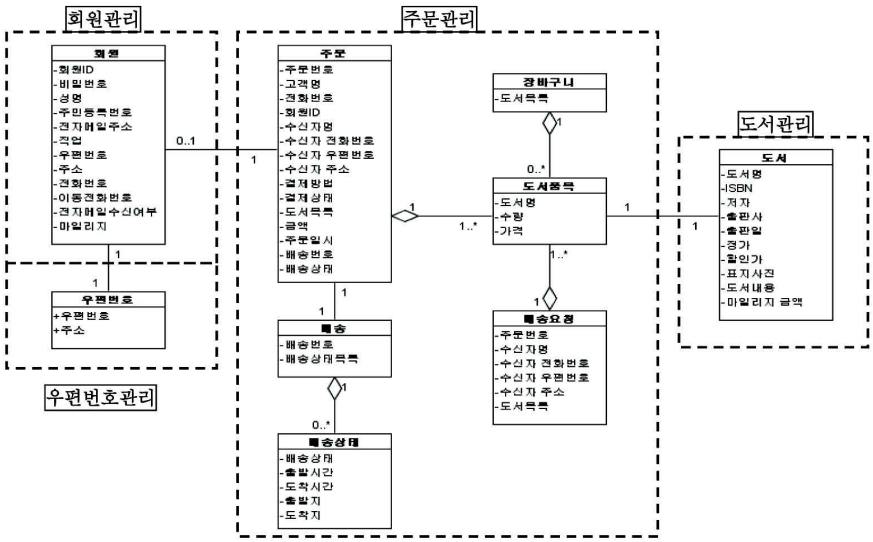

그림 6. 온라인 도서 주문관리 시스템의 비즈니스 컴포넌트

컴포넌트들이다. 즉, 주문 클래스의 데이터가 생성되면 동시에 배송 클래스의 데이터가 생성되고 그들은 1:1 관계이다. 또한 배송 클래스는 다시 배송상태 클래스와 포함관계이다. 따라서, 클래스다이어그램에서 배송 클래스가 주문클래스와 연관관계라 해도 배송 클래스는 주문 클래스에 완전히 의존되어 데이터가 생성되므로 그들은 종속관계이다. 배송 클래스는 배송상태 클래스를 포함하고 있는 포함관계이므로 그들 역시 강한 의존관계를 형성한다. 또한, 장바구니 클래스의 데이터가 생성되면 동시에 도서품목 클래스의 데이터가 생성되므로 그들은 포함관계이다. 주문 클래스는 다시 도서품목 클래스와 포함관계이고 배송요청은 도서품목 클래스와 포함관계이다. 따라서 가이드라인 2-1에 의해서 주문, 배송, 배송상태, 장바구니, 도서품목, 배송요청 클래스는 그룹화되어 주문관리 비즈니스 컴포넌트로 식별된다. 또한, 주문 클래스는 회원 클래스의 일부 데이터를 참조하는 연관관계이므로 그들은 그룹화되지 않는다. 우편번호 클래스 역시 회원 클래스와 데이터를 생성하는 관계가 아닌 데이터를 참조하는 연관관계이므로 그들은 그룹화 되지 않는다. 따라서 그들은 각각 회원클래스를 포함하는 회원관리 비즈니스 컴포넌트와 우편번호 클래스를 포함하는 우편번호관리 비즈니스 컴포넌트로 식별된다. 마지막으로, 도서품목 클래스는 도서 클래스의 데이터를 생성하는 관계가 아닌 도서 클래스의 데이터를 일부 참조하는 연관관계이므로 종속관계가 아니다. 따라서 그들은 그룹화 되지 않으므로 도서 클래스를 포함한 도서관리 비즈니스 컴포넌트가 식별된다.

비즈니스 컴포넌트가 포함하고 있는 유스케이스들을 그룹화한 단위를 비즈니스컴포넌트 서비스로 식별한다. 따라서 다음 그림 7은 각각의 비즈니스 컴포넌트에 의해서 식별된 비즈니스 컴포넌트 서비스와 그들이 포함하는 유스케이스를 제시한다.

3.2.4 유스케이스 서비스

비즈니스 컴포넌트 서비스는 유스케이스를 그룹

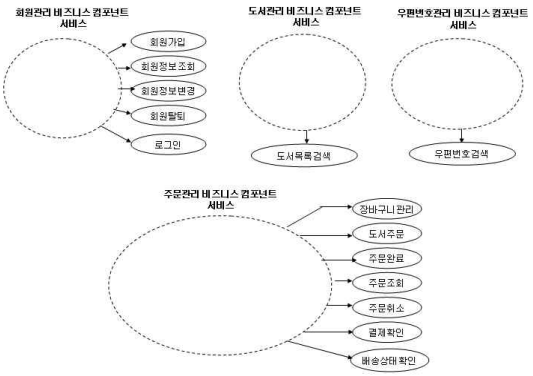

그림 7. 각 비즈니스 컴포넌트 서비스가 포함하고 있는 유스케이스 서비스

화해서 식별하므로 그 다음 서비스 계층은 유스케이스 서비스 계층이 된다. 본 논문에서는 유스케이스 단위의 서비스를 다음과 같이 식별하고 정의한다. 액터의 요청으로부터 그 결과를 출력하는 한 단위가 되는 기능적 서비스 단위를 이벤트라 한다. 하나의 유스케이스는 액터 관점에서 하나의 이벤트를 포함하거나 여러 개의 이벤트를 포함할 수 있다. 액터의 요청은 시스템이 제공해야할 서비스가 되므로 단일 이벤트는 단위 서비스가 된다. 액터 관점의 유스케이스 서비스를 실현하기 위하여 시스템은 단일 이벤트의 실행에 의해서 또는 여러 개의 단일 이벤트들의 조합에 의한 실행에 의하여 그 결과를 제공한다. 따라서 하나의 이벤트를 포함하는 비즈니스 관점의 서비스를 단위 유스케이스 서비스로 정의하고 여러 개의 이벤트를 포함하는 유스케이스는 비즈니스 관점의 복합 유스케이스 서비스로 정의한다. 기존의 컴포넌트 기반이나 클래스 기반의 레거시 시스템은 대다수 UML 기반의 모델을 생성하므로 유스케이스 단위의 서비스를 효율적으로 식별할 수 있다. 유스케이스 서비스 계층의 유스케이스 서비스는 단일 유스케이스 서비스와 복합 유스케이스 서비스로 분류하고 가이드라인은 다음에서 제시한다.

가이드라인 1 하나의 이벤트를 포함하는 유스케이스는 비즈니스 관점의 단위 서비스인 단위 유스케이스 서비스로 식별하고 여러 개의 이벤트를 포함하는 유스케이스는 비즈니스 관점의 복합 유스케이스 서비스로 식별한다.

가이드라인 1에 의해서 온라인 도서 주문관리 시스템의 유스케이스 서비스는 각 유스케이스가 포함하는 이벤트에 의하여 단위 유스케이스 서비스와 복합 유스케이스 서비스로 식별될 수 있으며 그 예는 3.4절의 표 4에서 제시하고 있다. 즉, 회원 조회는 단위 유스케이스 서비스이며 회원가입이나 도서 주문은 복합 유스케이스 서비스이다.

3.2.5 이벤트 서비스

각각의 유스케이스는 액터 관점의 이벤트들을 포함하고 있으므로 이벤트 서비스 계층의 이벤트 서비스 식별은 각 유스케이스당 시퀀스 다이어그램을 이용한다.

가이드라인 1 액터의 요청에 의한 이벤트를 비즈니스 컴포넌트나 비즈니스 컴포넌트들이 처리한 후 그 처리 결과를 비즈니스 컴포넌트가 저장할 경우의 한 단위를 이벤트 서비스로 식별한다.

가이드라인 2 액터의 요청에 의한 이벤트를 비즈니스 컴포넌트나 비즈니스 컴포넌트들이 처리한후 그 처리 결과를 요청한 액터에게 반환하는 한 단위를 이벤트 서비스로 식별한다.

가이드라인 3 액터의 요청에 의한 이벤트를 비즈니스 컴포넌트나 비즈니스 컴포넌트들이 처리한후 그 처리 결과를 다른 액터에게 보낼 경우 이벤트 서비스로 식별한다.

그림 8의 시퀀스 다이어그램에서 보여주고 있듯이 회원가입 유스케이스는 4 개의 이벤트를 포함한다. 회원가입 유스케이스의 기등록회원체크 이벤트, 중복아이디체크 이벤트, 우편번호검색 이벤트는 처리 결과를 다시 요청한 액터에게 반환해 주는것까지 한 단위가 되는 경우이므로 가이드라인 2에 의해서 이벤트 서비스로 식별된다. 회원정보등록 이벤트는 액터의 요청에 의해서 비즈니스 컴포넌트가 처리한 후 그 결과를 저장하는 이벤트이므로 가이드라인 1에 의해서 이벤트 서비스로 식별된다. 따라서 회원가입 유스케이스는 기등록회원체크 이벤트 서비스, 중복아이디체크 이벤트 서비스, 회원정보등록 이벤트 서비스, 우편번호검색 이벤트 서비스가 식별된다.

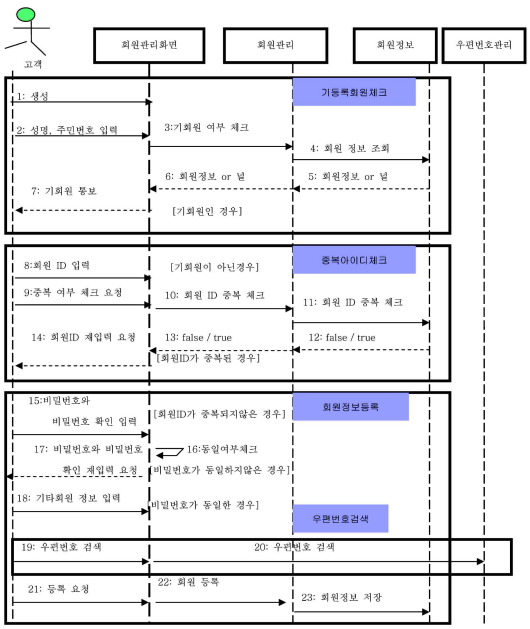

그림 8. 회원가입 유스케이스 서비스의 이벤트 서비스

3.2.6 컴포넌트 계층

그림 9에서 제시하고 있듯이 시스템 관점의 이벤트 서비스는 구현 관점의 컴포넌트 인터페이스와 매핑이되고 이벤트의 실행은 구현 관점의 서비스인 컴포넌트의 인터페이스 메소드나 컴포넌트 간의 인터페이스 메소드들의 상호협력에 의해서 이루어 진다. 따라서, 그림 9는 회원가입 복합 유스케이스 서비스에서 식별된 기등록회원체크 이벤트 서비스, 중복아이디체크 이벤트 서비스, 회원정보등록 이벤트 서비스 그리고 우편번호검색 이벤트 서비스가 각각 구현 계층의 회원관리 컴포넌트의 인터페이스 메소드인 checkRegMbr(), checkDupMb(), registerMbr()와 우편번호관리 컴포넌트의 인터페이스 메소드인 getAddress()로 각각 매핑되는 것을 보여주고 있다.

그림 10은 온라인 도서 주문관리 시스템의 컴포넌트 다이어그램으로 각각의 구현 컴포넌트와 그들의 인터페이스들을 보여주고 있다.

3.3 비즈니스 관점으로부터 구현 관점으로의 서비스 매핑

서비스 계층 간의 의존 관계의 특성을 부여하여

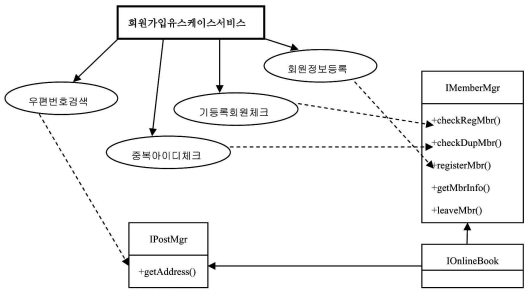

그림 9. 회원가입 이벤트 서비스에 대한 컴포넌트 인터페이스 메소드

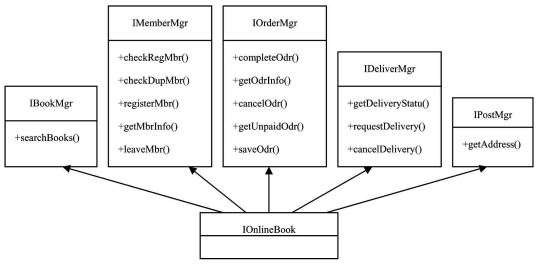

그림 10. 온라인 도서 주문관리 시스템의 컴포넌트 다이어그램

각 서비스 계층의 서비스들 간에 매핑이 잘 정의되고 모델화될 수 있다면 그들은 또 다른 응용을 위해서 효과적으로 재 사용될 수 있고 서비스들을 용이하게 관리할 수 있다. 따라서, 본 장은 서비스 계층 및 서비스 계층 간의 매핑 관계를 명확히 명시하기 위한 추상화 모델을 정의하고 서비스를 효율적으로 관리하기 위한 매핑 테이블을 정의한다.

3.3.1 서비스 계층 간의 매핑을 위한 추상화 모델

상위 서비스 계층의 서비스는 하위 서비스 계층의 서비스에 의존되어 기능을 실현하므로 상위 서비스 계층의 각각의 서비스는 하위 서비스 계층의 서비스들과 매핑이 된다. 따라서 본 절은 서비스 계층 간의 의존적 특성을 부여하여 서비스 계층 및 서비스 계층 간의 매핑을 명확히 정의하기 위한 추상화 모델을 제시한다.

[정의 1] 서비스지향 컴퓨팅 시스템의 서비스 계층

서비스지향 컴퓨팅 시스템은 시스템 컴포넌트 서비스 계층(SL), 비즈니스 컴포넌트 서비스 계층(BL), 유스케이스 서비스 계층(UL), 이벤트 서비스 계층(EL), 컴포넌트 계층(IL) 등 유한 개의 서비스 계층으로 구성된다. 따라서, 서비스 지향 컴퓨팅 시스템을 S라 하면, 시스템은 다음과 같이 정의된다.

$$L, BL, UL, EL, IL\}$$

[정의 2] 시스템 컴포넌트 서비스 계층

가장 상위 계층의 서비스인 시스템 컴포넌트 서비스 계층은 유한 개의 시스템 컴포넌트 서비스들로 구성된다. 시스템 컴포넌트 서비스 계층이 포함하고 있는 시스템 컴포넌트 서비스들을 $\mathfrak{d}_i(\iota = 1...J)$라 하면, 시스템 컴포넌트 서비스 계층인 SL은 다음과 같이 정의 된다.

$$= \{SS_1, SS_2, ..., SS_j\}$$

[정의 3] 비즈니스 컴포넌트 서비스 계층

비즈니스 컴포넌트 서비스 계층은 유한 개의 비즈니스 컴포넌트 서비스들로 구성된다. 비즈니스 컴포넌트 서비스 계층이 포함하고 있는 비즈니스 컴포넌트 서비스들을 $\mathfrak{d}_k(\kappa = 1...J)$라 하면, 비즈니스 컴포넌트 서비스 계층인 BL은 다음과 같이 정의 된다.

$$:S_1, BS_2, ..., BS_l\}$$

[정의 4] 유스케이스 서비스 계층

유스케이스 서비스 계층은 유한 개의 유스케이스 서비스들로 구성된다. 유스케이스 서비스 계층이 포함하고 있는 유스케이스 서비스들을 $US_m(m=1...n)$라 하면, 유스케이스 서비스 계층인 UL은 다음과 같이 정의 된다.

$$UL = \{US_1, US_2, ..., US_n\}$$

[정의 5] 이벤트 서비스 계층

이벤트 서비스 계층은 유한 개의 이벤트 서비스들로 구성된다. 이벤트 서비스 계층이 포함하고 있는 이벤트 서비스들을 $ES_o(o=1...p)$라 하면, 이벤트 서비스 계층인 EL은 다음과 같이 정의 된다.

$$EL = \{ES_1, ES_2, ..., ES_p\}$$

[정의 6] 컴포넌트 계층

컴포넌트 계층은 유한 개의 컴포넌트 인터페이스들로 구성된다. 컴포넌트 계층이 포함하고 있는 컴포넌트 인터페이스들을 $IS_q(q=1...r)$라 하면, 컴포넌트 계층인 IL은 다음과 같이 정의 된다.

$$IL = \{IS_1, IS_2, ..., IS_r\}$$

[정의 7] 시스템 컴포넌트 서비스와 비즈니스 컴포넌트 서비스와의 매핑

각각의 시스템 컴포넌트 서비스는 유한 개의 비즈니스 컴포넌트 서비스들에 의해서 기능이 실현된다. 즉, 시스템 컴포넌트 서비스 계층 SL의 각각의 시스템 컴포넌트 서비스 $SS_i(i=1...j)$들은 비즈니스 컴포넌트 서비스 계층 BL의 유한 개의 비즈니스 컴포넌트 서비스들인 $BS_k(k=1...t)$에 의존된다. 따라서 각각의 시스템 컴포넌트 서비스 $SS_i(i=1...j)$가 의존하는 비즈니스 컴포넌트 서비스들 $BS_k(k=1...t)$에 대한 매핑인 $SS_i_BS_k$는 다음과 같이 정의 된다.

$$SS_i(BS_k) = \{BS_{i.k} \mid BS_{i.k}\ means\ (SS_i, BS_k),\ s.t.\ BS_{i.k} \in BL\}$$
$$= \{BS_{i.1}, BS_{i.2}, ..., BS_{i.k}\}$$

[정의 8] 비즈니스 컴포넌트 서비스와 유스케이스 서비스와의 매핑

각각의 비즈니스 컴포넌트 서비스는 유한 개의 유스케이스 서비스들에 의해서 기능이 실현된다. 즉, 비즈니스 컴포넌트 서비스 계층 BL의 각각의 비즈니스 컴포넌트 서비스 $BS_k(k=1...t)$들은 유스케이스 서비스 계층 UL의 유한 개의 유스케이스 서비스들인 $US_m(m=1...n)$에 의존된다. 따라서 각각의 비즈니스 컴포넌트 서비스 $BS_k(k=1...t)$가 의존하는 유스케이스 서비스들 $US_m(m=1...n)$에 대한 매핑인 $BS_k_US_m$는 다음과 같이 정의 된다.

$$BS_k\_US_m = BS_k(US_m) = \{US_{k.m} \mid US_{k.m}\ means\ (BS_k, US_m),\ s.t.\ US_{k.m} \in UL\}$$
$$= \{US_{k.1}, US_{k.2}, ..., US_{k.m}\}$$

[정의 9] 유스케이스 서비스와 이벤트 서비스와의 매핑

각각의 유스케이스 서비스는 유한 개의 이벤트 서비스들에 의해서 기능이 실현된다. 즉, 유스케이스 서비스 계층 UL의 각각의 유스케이스 서비스 $US_m(m=1...n)$들은 이벤트 서비스 계층 EL의 유한 개의 이벤트 서비스들인 $ES_o(o=1...p)$에 의존된다. 따라서 각각의 유스케이스 서비스 $US_m(m=1...n)$가 의존하는 이벤트 서비스들 $ES_o(o=1...p)$에 대한 매핑인 $US_m_ES_o$는 다음과 같이 정의 된다.

$$US_m\_ES_o = US_m(ES_o) = \{ES_{m.o} \mid ES_{m.o}\ means\ (US_m, ES_o),\ s.t.\ ES_{m.o} \in EL\}$$
$$= \{ES_{m.1}, ES_{m.2}, ..., ES_{m.o}\}$$

[정의 10] 이벤트 서비스와 컴포넌트 인터페이스와의 매핑

각각의 이벤트 서비스는 유한 개의 컴포넌트 인터페이스들에 의해서 구현된다. 즉, 이벤트 서비스 계층 EL의 각각의 이벤트 서비스 $ES_o(o=1...p)$는 컴포넌트 계층 IL의 유한 개의 컴포넌트 인터페이스들인 $IS_q(q=1...r)$에 의존된다. 따라서 각각의 이벤트 서비스 $ES_o(o=1...p)$가 의존하는 컴포넌트 인터페이스들 $IS_q(q=1...r)$에 대한 매핑인 $ES_o_IS_q$는 다음과 같이 정의 된다.

$$ES_o\_IS_q = ES_o(IS_q) = \{IS_{o.q} \mid IS_{o.q}\ means\ (ES_o, IS_q),\ s.t.\ IS_{o.q} \in IL\}$$
$$= \{IS_{o.1}, IS_{o.2}, ..., IS_{o.q}\}$$

[정의 11] 시스템 컴포넌트 서비스로부터 유스케이스 서비스로의 매핑

시스템 컴포넌트 서비스 계층의 각각의 시스템 컴포넌트 서비스 $SS_i(i=1...j)$는 비즈니스 컴포넌트 서비스 계층의 비즈니스 컴포넌트 서비스들인 $BS_k(k=1...t)$로 매핑이 되고 비즈니스 컴포넌트 서비

스 계층의 각각의 비즈니스 컴포넌트 서비스 $BS_k(k=1...l)$는 유스케이스 서비스 계층의 유한 개의 유스케이스 서비스들인 $US_m(m=1...n)$으로 매핑이 된다. 따라서 시스템 컴포넌트 서비스로부터 유스케이스 서비스로의 매핑인 $SS_i\_BS_k\_US_m$ 은 다음과 같이 정의 된다.

$SS_i\_BS_k\_US_m$ 는 위에서 정의한 정의 7에 의하여 다음 식 ①이 정의 되고

$$\_BS_k\_US_m = SS_i(BS_k(US_m)) = BS_{i.k}(US_m) \quad ①$$

위의 식 ①은 정의 8에 의하여 다음 식 ②가 정의 된다.

$$_k(US_m) = US_{i.k.m} \quad ②$$

따라서 위의 식 ①과 ②에 의하여 시스템 컴포넌트 서비스로부터 유스케이스 서비스로의 매핑은 다음 식 ③과 같이 정의 된다.

$$_i = SS_i(BS_k(US_m)) = BS_{i.k}(US_m) = US_{i.k.m} \quad ③$$

where:

$$SS_i \in SL, BS_{i.k} \in BL, US_{i.k.m} \in UL$$

[정의 12] 시스템 컴포넌트 서비스로부터 이벤트 서비스로의 매핑

시스템 컴포넌트 서비스 계층의 각각의 시스템 컴포넌트 서비스 $SS_i(i=1...j)$는 비즈니스 컴포넌트 서비스 계층의 비즈니스 컴포넌트 서비스들인 $_k(k=1...l)$로 매핑이 되고 비즈니스 컴포넌트 서비스 계층의 각각의 비즈니스 컴포넌트 서비스 $_k(k=1...l)$는 유스케이스 서비스 계층의 유한 개의 유스케이스 서비스들인 $_m(m=1...n)$으로 매핑이 된다. 또한 유스케이스 계층의 각각의 유스케이스 서비스 $_m(m=1...n)$는 이벤트 서비스 계층의 유한 개의 이벤트 서비스들인 $_o(o=1...p)$로 매핑이 된다. 따라서 시스템 컴포넌트 서비스로부터 이벤트 서비스로의 매핑인 $SS_i_BS_k_US_m_ES_o$는 다음과 같이 정의 된다.

$_i_BS_k_US_m_ES_o$는 위에서 정의한 정의 11에 의하여 다음 식 ①이 정의 되고

$$= BS_{i.k}(US_m(ES_o))$$
$$= US_{i.k.m}(ES_o) \quad ①$$

위의 식 ①은 정의 9에 의하여 다음 식 ②가 정의 된다.

$$US_{i.k.m}(ES_o) = ES_{i.k.m.o} \quad ②$$

따라서 위의 식 ①과 ②에 의하여 시스템 컴포넌트 서비스로부터 이벤트 서비스로의 매핑은 다음 식 ③과 같이 정의 된다.

$$= US_{i.k.m}(ES_o)$$
$$= ES_{i.k.m.o} \quad ③$$

where:

$$SS_i \in SL, BS_{i.k} \in BL, US_{i.k.m} \in UL, ES_{i.k.m.o} \in EL$$

[정의 13] 시스템 컴포넌트 서비스로부터 컴포넌트 인터페이스로의 매핑

시스템 컴포넌트 서비스 계층의 각각의 시스템 컴포넌트 서비스 $SS_i(i=1...j)$는 비즈니스 컴포넌트 서비스 계층의 비즈니스 컴포넌트 서비스들인 $_k(k=1...l)$로 매핑이 되고 비즈니스 컴포넌트 서비스 계층의 각각의 비즈니스 컴포넌트 서비스 $_k(k=1...l)$는 유스케이스 서비스 계층의 유한 개의 유스케이스 서비스들인 $US_m(m=1...n)$으로 매핑이 된다. 또한 유스케이스 서비스 계층의 각각의 유스케이스 서비스 $_m(m=1...n)$는 이벤트 서비스 계층의 유한 개의 이벤트 서비스들인 $ES_o(o=1...p)$로 매핑이 된다. 이벤트 서비스 계층의 각각의 이벤트 서비스 $_o(o=1...p)$는 컴포넌트 계층의 유한 개의 컴포넌트 인터페이스들인 $_q(q=1...r)$로 매핑이 된다. 따라서 시스템 컴포넌트 서비스로부터 컴포넌트 인터페이스로의 매핑인 $SS_k_US_m_ES_o_IS_q$는 다음과 같이 정의 된다.

$SS_k_US_m_ES_o_IS_q$는 정의 12에 의하여 다음 식 ①이 정의 되고

$$\begin{aligned} SS_i\_BS_k\_US_m\_ES_o\_IS_q &= SS_i(BS_k(US_m(ES_o(IS_q)))) \\ &= BS_{i.k}(US_m(ES_o(IS_q))) \\ &= US_{i.k.m}(ES_o(IS_q)) \\ &= ES_{i.k.m.o}(IS_q) \end{aligned} \quad ①$$

위의 식 ①은 정의 10에 의하여 다음 식 ②가 정의 된다.

$$ES_{i.k.m.o}(IS_q) = IS_{i.k.m.o.q} \quad ②$$

표 3. 서비스 매핑 테이블

비즈니스 관점			시스템 관점	구현 관점
시스템 컴포넌트 서비스 계층	비즈니스 컴포넌트 서비스 계층	유스케이스 서비스계층	이벤트 서비스 계층	컴포넌트 인터페이스 계층
시스템 컴포넌트 서비스 1 ***SS1***	비즈니스 컴포넌트 서비스 1 ***BS1.1***	유스케이스 서비스 1 ***US1.1.1***	이벤트 서비스 1 ***ES1.1.1.1***	인터페이스 메소드 1 ***IS1.1.1.1.1***
		유스케이스 서비스 2	이벤트 서비스 2	인터페이스 메소드 2
			이벤트 서비스 3	인터페이스 메소드 3
			이벤트 서비스 4	인터페이스 메소드 4
			이벤트 서비스 5	인터페이스 메소드 5
			이벤트 서비스 6	인터페이스 메소드 6

따라서 식 ①과 ②에 의하여 시스템 컴포넌트 서비스로부터 컴포넌트 인터페이스로의 매핑은 다음 식 ③과 같이 정의 된다.

$$\begin{aligned} SS_i\_BS_k\_US_m\_ES_o\_IS_q &= SS_i(BS_k(US_m(ES_o(IS_q)))) \\ &= BS_{i.k}(US_m(ES_o(IS_q))) \\ &= US_{i.k.m}(ES_o(IS_q)) \\ &= ES_{i.k.m.o}(IS_q) \\ &= IS_{i.k.m.o.q} \end{aligned} \quad ③$$

$Where$:

$SS_i \in SL$, $BS_{i.k} \in BL$, $US_{i.k.m} \in UL$, $ES_{i.k.m.o} \in EL$, $IS_{i.k.m.o.q} \in IL$

3.3.2 서비스 매핑 테이블

비즈니스 관점으로부터 구현 관점까지의 관점 차이에 의한 의미적 불일치의 해결, 각 서비스 계층에 의한 다양한 서비스 입자 크기의 선택 및 사용자의 동적인 요구에 적합한 서비스의 선택, 서비스들 간의 의존 관계의 명시에 의한 서비스들 간의 추적 관리 및 효율적인 재 사용 그리고 서비스 명세등에 용이하게 적용할 수 있는 방법이 필요하다. 따라서 본 절에서는 3.3.1절에서 정의한 추상화 모델을 적용하여 서비스들 간의 관계를 명시하고 관리할 수 있는 매핑 테이블을 다음 표 3에서 제시한다. 서비스 매핑 테이블에서 서비스 계층 간의 매핑 및 서비스들에 대한 명세는 3.3.1절에서 정의한 추상화 모델을 적용하고 기호 숫자를 이용해서 표현한다.

3.4 온라인 도서 주문관리 시스템 사례

본 절은 온라인 도서 주문관리 시스템의 3가지 관점의 각 계층에서 식별된 서비스들과 서비스들 간의 매핑 사례를 표 4에서 제시한다.

4. 분석 평가 및 기대 효과

서비스지향 컴퓨팅 시스템의 기능적 단위인 서비스들의 재사용은 서비스들 간의 느슨한 결합에 의하여 효과적으로 이루어질 수 있다. 따라서 본 장은 식별된 서비스가 느슨한 결합도를 가지는지를 4.1절과 4.2절을 통해서 분석 평가한다. 또한, 본 논문에서 제시한 서비스 식별 방법에 대한 기대 효과를 4.3절, 4.4절 그리고 4.5절을 통해서 제시한다. 마지막으로 4.6절에서 기존의 서비스지향 컴퓨팅 시스템을 위한 방법론과의 비교 평가를 통해서 제안한 방법의 효율성을 평가한다.

4.1 비즈니스 관점의 식별된 서비스에 대한 독립성

다음 표 5에서 온라인 도서 주문관리 시스템에서 식별된 비즈니스 관점의 서비스에 대한 독립성을 구현 컴포넌트와의 관계를 통하여 보이고자 한다.

위의 표 5에서 제시하고 있는 비즈니스 관점의 비즈니스 컴포넌트 서비스 중 도서관리 서비스, 우편번호관리 서비스는 포함하고 있는 컴포넌트가 단일 컴포넌트의 기능 실현에 의해서 실행되므로 다른 컴포넌트에 전혀 영향을 끼치지 않는다. 또한 회원관리 서비스는 기능 실현을 위해서 회원관리 컴포넌트와 우편번호관리 컴포넌트를 사용하지만 우편번호관리 컴포넌트는 공유 컴포넌트로서 회원관리 컴포넌트가 참조만하는 관계이므로 회원관리 컴포넌트는 우편관리 컴포넌트에 전혀 영향을 끼치지 않는다. 따라서 도서관리 서비스, 우편관리 서비스, 회원관리 서비스는 다른 서비스와의 관계를 고려해볼 때 보다 독립적이므로 느슨한 결합도를 가지고 있다고 할 수

표 4. 온라인 도서 주문관리 시스템의 서비스 식별 결과 및 서비스 매핑

서비스 계층				구현 계층
비즈니스 관점의 서비스			시스템 관점의 서비스	구현 관점의 서비스
패키지 계층		유스케이스 계층	이벤트 계층	컴포넌트 계층
시스템 컴포넌트 서비스	비즈니스 컴포넌트 서비스	유스케이스 서비스	이벤트 서비스	구현 컴포넌트의 인터페이스 메소드
온라인 북 주문 시스템 서비스 ***SS1***	**회원관리 서비스** ***BS1.1***	회원조회 ***US1.1.1***	회원정보조회 ***ES1.1.1.1***	getMemberInfo() ***IS1.1.1.1.1***
		회원가입 ***US1.1.2***	기등록회원체크	checkRegMbr()
			중복아이디체크	checkDupMbr()
			회원정보등록	registerMbr(),
			우편번호 검색	***getAddress()***
		회원정보변경 ***US1.1.3***	회원정보조회	getMemberInfo()
			회원정보변경	registerMember()
			우편번호 검색	***getAddress()***
		회원탈퇴 ***US1.1.4***	회원정보조회	getMemberInfo()
			회원탈퇴	leaveMemeber()
		로그인 ***US1.1.5***	로그인	checkRegMbr()
	주문관리 서비스 ***BS1.2***	장바구니 관리 ***US1.2.1*** 사전조건 : 도서목록검색 실행	장바구니 담기	addCart()
			장바구니 조회	viewCart()
			장바구니 변경	changeCart()
			장바구니 비우기	emptyCart()
		도서주문 ***US1.2.2*** 사전조건 : 장바구니관리 실행	회원정보조회 (회원정보/배송정보)	getMemberInfo()
			결제처리	*외부시스템처리*
			주문정보저장	saveOrder()
			도서주문완료	completeOrder()
		주문완료 ***US1.2.3***	도서주문완료	completeOrder()
		주문조회 ***US1.2.4***	주문조회	getOrderInfos()
			주문상세내역조회	getOrderInfos()
		주문취소 ***US1.2.5***	주문취소	cancelOrder()
		결제확인 ***US1.2.6***	미결제주문조회	getUnpaidOrder()
			주문완료	completeOrder()
		배송상태확인 ***US1.2.7***	배송상태확인	queryDeliveryStatus()
		주문완료 ***US1.2.8***	*주문완료에서 실행*	*requestDelivery()*
			주문취소에서 실행	*cancelDelivery()*
	도서관리 서비스 ***BS1.3***	도서목록검색 ***US1.3.1***	도서목록조회	searchBooks()
			도서상세정보조회	searchBooks()
	우편번호관리 : 공유 서비스 BS1.4	***우편번호검색 US1.4.1***	***우편번호검색***	***getAddress()***

표 5. 비즈니스 관점의 서비스의 독립성

비즈니스 서비스 관점			비즈니스 컴포넌트			
시스템 컴포넌트 서비스	비즈니스 컴포넌트 서비스	유스케이스 서비스	회원관리	주문관리	도서관리	우편번호 관리
온라인 도서 주문 시스템 서비스	회원관리 서비스	회원조회	√			
		회원가입	√			√
		회원정보변경	√			√
		회원탈퇴	√			
		로그인	√			
	주문관리 서비스	장바구니 관리		√	√	
		도서주문	√	√	√	
		주문완료		√		
		주문조회		√		
		주문취소		√		
		결제확인		√		
		배송상태확인		√		
	도서관리 서비스	도서목록검색			√	
	우편관리 서비스	우편번호검색				√

있다. 또한, 비즈니스 컴포넌트 서비스 중 주문관리 서비스는 기능 실현을 위해서 주도적으로 사용되는 컴포넌트가 주문관리 컴포넌트이고 주문관리 컴포넌트에 의해서 참조되는 컴포넌트들이 회원관리, 도서관리 컴포넌트들이다. 이때 회원관리나 도서관리 컴포넌트는 주문관리 컴포넌트가 기능 실현을 위해서 참조만 하는 관계이므로 주문관리 컴포넌트에 의해서 회원관리나 도서관리 컴포넌트는 전혀 영향을 받지 않는다. 따라서, 주문관리 서비스 역시 느슨한 결합도를 가지고 있다. 비즈니스 관점의 서비스 중 시스템 컴포넌트 서비스는 그림 4에서 제시하고 있듯이 업무 도메인의 기능 분할에 의하여 식별되고 느슨한 결합도를 가진 비즈니스 컴포넌트 서비스들의 조합에 의하여 기능이 실현되므로 식별된 시스템 컴포넌트 서비스는 당연히 느슨한 결합도를 가지고 있다고 할 수 있다.

4.2 시스템 관점의 식별된 서비스에 대한 독립성

다음 표 6에서 온라인 도서 주문관리 시스템에서 식별된 시스템 관점의 서비스에 대한 독립성을 구현 컴포넌트와의 관계를 통하여 보이고자 한다.

위의 표 6에서 제시하고 ⊙는 시스템 관점의 이벤트 서비스와 매핑되는 컴포넌트 인터페이스 메소드가 기능을 실현할 때에 내부적으로 참조하는 컴포넌트들을 체크하고 있다. 표 6에서 제시하듯이 표 5에서 제시하고 있는 비즈니스 관점의 유스케이스 서비스 보다는 시스템 관점의 이벤트 서비스가 훨씬 느슨한 결합도를 가지고 있다는 것을 확인할 수 있다. 그 이유는 이벤트 서비스들의 조합에 의하여 유스케이스 서비스가 실현되기 때문이다. 또한 시스템 관점의 이벤트 서비스는 구현 관점의 서비스인 컴포넌트의 각각의 인터페이스 메소드와 1:1 매핑이 된다는 것을 알 수 있다. 따라서 식별된 시스템 관점의 서비스 역시 느슨한 결합도를 가지고 있다고 할 수 있으며 서비스가 보다 독립적이어야 한다는 특성을 만족하고 있다. 따라서 본 장의 4.1, 4.2절에서 제시하고 있듯이 본 논문에서 제안한 방법에 의하여 식별된 서비스는 보다 느슨한 결합도를 가지고 있으므로 서비스 간의 조합이나 분리 그리고 새로운 서비스의 추가나 삭제에 의해서 서비스를 유연하게 조립할 수 있으므로 서비스에 대한 재사용이 용이하게 이루어질 수 있다.

4.3 서비스의 재사용 단위

본 논문은 다양한 크기를 가진 서비스와 서비스 간의 의미적 불일치를 해소하기 위하여 비즈니스 관

표 6. 시스템 관점의 서비스의 독립성

시스템 관점의 서비스	구현 관점의 서비스	컴포넌트			
이벤트 서비스	구현 컴포넌트의 인터페이스 메소드	회원 관리	주문 관리	도서 관리	우편 관리
회원정보조회	getMemberInfo()	√			
기등록회원체크	checkRegMbr()	√			
중복아이디체크	checkDupMbr()	√			
회원정보등록	registerMbr()	√			⊙
회원정보변경	registerMbr()	√			⊙
회원탈퇴	leaveMemeber()	√			
로그인	checkRegMbr()	√			
장바구니담기	addCart()		√	⊙	
장바구니조회	viewCart()		√		
장바구니변경	changeCart()		√		
장바구니비우기	emptyCart()		√		
주문정보저장	saveOrder()	⊙	√		
도서주문완료	completeOrder()		√		
주문조회	getOrderInfos()		√		
주문상세내역조회	getOrderInfos()		√		
주문취소	cancelOrder()		√		
미결제주문조회	getUnpaidOrder()		√		
배송상태확인	queryDeliveryStatus()		√		
도서목록조회	searchBooks()			√	
도서상세정보조회	searchBooks()			√	
우편번호검색	getAddress()				√

점, 시스템 관점 그리고 구현 관점으로 나누어 서비스의 계층 및 타입을 분류하였다. 3 가지 관점의 서비스 중 구현 관점의 서비스는 구현을 위한 컴포넌트 계층이므로 서비스에 대한 재사용 단위는 비즈니스 관점과 시스템 관점의 서비스이다. 비즈니스 관점과 시스템 관점의 서비스의 크기를 비교해 보면 비즈니스 관점이 포함하고 있는 서비스들 중 시스템 컴포넌트 서비스, 비즈니스 컴포넌트 서비스, 유스케이스 서비스 순서로 서비스 입자의 크기가 작다. 또한, 시스템 관점의 서비스인 이벤트 서비스는 비즈니스 관점의 유스케이스 서비스 보다 서비스 입자의 크기가 작다. 또한 이들은 서비스 단위 마다 느슨한 결합도를 가지고 있으므로 다양한 관점에서 다양한 크기를 가진 서비스 단위별로 기능적 재사용 단위가 될 수 있다. 따라서 사용자의 서비스의 크기에 따른 동적인 서비스의 재사용 요구에 효율적으로 대처할 수 있다.

4.4 컴포넌트기반 시스템에서 서비스 지향 컴퓨팅 시스템으로의 확장

컴포넌트기반의 서비스 식별 방법을 제시하고 비즈니스 관점의 서비스로부터 구현 관점까지 서비스들 간에 매핑 방법을 제시했으므로 컴포넌트기반 시스템을 서비스지향 시스템으로 용이하게 확장할 수 있다.

4.5 서비스들 간의 추적성

각 관점에 대한 각 계층의 서비스들 간의 매핑이 가능함으로 해서 비즈니스 관점으로부터 구현 관점까지 효율적으로 추적이 가능하다. 또한, 비즈니스 관점과 구현 관점 사이의 서비스 간에 의미적 불일치를 해소하고 유지 보수 단계에서 서비스 관리를 용이하게 할 수 있다.

표 7. 기존 방법과의 비교 평가

비교기준 \ SOA 방법론	SOAD[3]	SOMA[4]	SOUP[8]	M4SOD[9]	제안한 방법
제안	IBM 2004. 6	IBM 2004. 11	Knual Mittal 2005	YUN 2005	Our Method
영향받은 기술	OOAD, EA, BPM	SOA	RUP, XP	CBD, SOA	OOAD, CBD, SOA
서비스 계층의 분류기준 제시	언급없음	언급없음	언급없음	언급없음	분류기준 제시
서비스 식별방법 제시	구체적이지 않음	구체적이지 않음	구체적이지 않음	유스케이스기반 서비스 식별 방법만 제시	식별방법 제시
서비스 계층간 매핑방법 제시	언급없음	언급없음	언급없음	언급없음	매핑방법 제시

4.6 비교 평가

본 절은 기존 방법과의 비교 평가를 표 7을 통해서 제시하고자 한다.

5. 결 론

서비스지향 시스템을 위한 기존의 방법들은 서비스 관점 및 서비스 입자의 크기에 따른 서비스를 계층별로 분류하거나 분류 가이드라인을 제시하고 있지 않다. 또한, 서비스 식별 방법과 비즈니스 관점으로부터 구현 관점까지 각 계층간 매핑 방법을 제시하지 않고 있다. 그러나, 효율적인 서비스지향 시스템을 위해서 서비스의 추상화 정도에 따른 서비스 계층의 분류 및 각 계층에 대한 서비스 식별 방법이 필요하고 비즈니스 관점의 서비스 계층과 구현 관점의 애플리케이션 계층에 대한 의미적 불일치를 해결하기 위해 매핑이 필요하다. 따라서 본 논문에서는 비즈니스 관점과 구현 관점의 서비스에 대한 의미적 불일치를 해결하고 서비스 계층의 분류 및 다양한 크기를 가진 서비스가 효율적으로 식별되도록 구체적인 가이드라인과 매핑 방법을 제시하였다. 또한, 온라인 도서주문 시스템 사례 및 비교 평가를 통해 그 효율성을 검증하였다. 즉, 본 논문에서 제안한 서비스 식별 방법을 적용하여 컴포넌트기반 시스템을 서비스지향 시스템으로 효율적으로 확장할 수 있다.

참 고 문 헌

[1] K. H. Bennett and J. Xu, “Software Services and Software Maintenance,” *Proceedings of the 7th European Conference on Software Maintenance and Re-engineering*, Benevento, pp. 3-12, 2003.

[2] Z. Zhang and H. Yang, “Incubating Services in Legacy Systems for Architectural Migration,” *Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC'04)*, pp. 196-203, 2004.

[3] O. Zimmerman, P. Krogdahl and C. Gee, “Elements of Service-Oriented Analysis and Design,” *IBM developer Works*, Jun., 2004.

[4] A. Arsanjani, “Service-oriented modeling and architecture,” *IBM developer Works*, Nov., 2004.

[5] Thomas Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*, Prentice Hall, 2005.

[6] F. Chen, H. Yang, C. H. Wang and W. C. Chung Chu, “Feature Analysis for Service-Oriented Reengineering,” *Proceedings of the 12th Asia-Pacific Software Engineering Conference(APSEC'05)*, pp. 201-208, 2005.

[7] Rim Samia Kaabi, Carine Souveyet and Colette Rolland, “Eliciting Service Composition in a Goal Driven Manner,” *Proceedings of the Second International Conference on Service Oriented Computing (ICSOC 2004),* Nov., pp. 308-315, 2004.

[8] Knual Mittal, “Service Oriented Unified Process(SOUP),” http://www.knual-mittal.com/html/soup.shtml, 2006.
[9] 윤홍란, M4SOD : SOA를 위한 서비스 지향 개발 방법론, 숙명여대, 박사학위논문, 2005.
[10] C. R. Turner, A. Fuggetta, L. Lavazza and A. L. Wolf, “A Conceptual Basics for Feature Engineering,” *Journal of System and Software*, Vol.49, No.1, Dec., 1999.
[11] 전병선, J2EE Enterprise System, 객체지향 CBD 개발 방법론, 영진닷컴, 2004.

최 미 숙

1990년 전북대학교 수학과 졸업 (이학사)
1994년 숙명여자대학교 컴퓨터과학과 석사과정 졸업 (이학석사)
2002년 숙명여자대학교 컴퓨터과학과 박사과정 졸업 (이학박사)
1995년～1999년 나주대학 소프트웨어 개발과 전임교수
2004년～2006년 우석대학교 컴퓨터공학과 초빙교수
2006년～현재 우석대학교 기초및자연과학연구소 연구교수
관심분야 : 소프트웨어 개발 방법론, 컴포넌트 기반 시스템, 소프트웨어 메트릭, SOA

이 서 정

1989년 숙명여자대학교 전산학과 졸업(이학사)
1991년 동대학교 대학원 전산학과 석사과정 졸업 (이학석사)
1998년 동대학교 대학원 전산학과 박사과정 졸업 (이학박사)
1998년～2003년 동덕여자대학교 강의교수
2004년 숭실대학교 정보미디어연구센터 연구교수
2005년～현재 한국해양대학교 조교수
관심분야 : 소프트웨어 개발방법론, SOA, 임베디드 소프트웨어 설계

이 종 석

1988년 서울대학교 계산통계학과 학사
1990년 서울대학교 계산통계학과 석사
2001년 서울대학교 컴퓨터공학 전공 박사
1993년～현재 우석대학교 컴퓨터교육과 교수
관심분야 : 소프트웨어 공학, 소프트웨어 복잡도, 컴포넌트 기반 시스템, SOA

양 승 원

1997년 7월～1998년 12월 ETRI 초빙 연구원
2003년 9～2004년 8월 Univ. of Guelph 초빙교수
1994년 3월～2006년 2월 우석대학교 컴퓨터공학과 교수
2006년 3월～현재 우석대학교 게임 콘텐츠학과 교수
관심분야 : SOA, 임베디드 소프트웨어 설계