Journal of the Korea Academia-Industrial cooperation Society
Vol. 16, No. 1 pp. 735-743, 2015

http://dx.doi.org/10.5762/KAIS.2015.16.1.735
ISSN 1975-4701 / eISSN 2288-4688

효율적인 서비스 시스템 개발을 위한 룰 기반의 관점 지향 기법

이우진[1], 최일우[2*]
[1]세종대학교 정보통신공학과, [2]강남대학교 교양학부

Rule-based Aspect Oriented Approach for Efficient Service System Development

Woo-Jin Lee[1], Il-Woo Choi[2*]
[1]**Dept. of Information and Communication Engineering, Sejong University**
[2]**Division of General Studies, Kangnam University**

요 약 서비스 지향 아키텍처(Service Oriented Architecture)는 엔터프라이즈 애플리케이션 개발 시 유연성을 보장하여, 비즈니스 변화에 최대한 민첩하게 대응하도록 지원한다. 그럼에도 불구하고 비즈니스와 제약조건을 하나의 서비스로 코딩하는 일반적 접근법은 비즈니스 룰의 변경을 위하여 전체 결합로직을 변경하는 등 많은 자원을 소모하게 된다. 본 논문에서는 이러한 문제점을 개선하기 위하여 서비스 시스템 개발에 관점 지향 기법(Aspect Oriented Approach)을 확장하여 적용한다. 관점 지향 기법의 기존 핵심관심(Core Concern), 횡단관심(Cross Cutting Concern)이외에 비즈니스 프로세스와 서비스에 포함된 비즈니스 룰을 룰관심(Rule Concern)이라는 새로운 관심사로 분리한다. 룰관심은 수준에 따라 프로세스 룰관심(PRA: Process Rule Aspect)과 서비스 룰관심(SRA: Service Rule Aspect)으로 구분된다. 시스템은 이러한 관심사의 분리를 통하여 핵심관심, 횡단관심 및 룰관심으로 각각 모듈화 되고 독립적으로 유지보수 되어 서비스 시스템의 적용성, 재사용성 및 유지보수성을 높일 수 있다.

Abstract The service oriented architecture assures flexibility of enterprise application development, so it supports agile reaction to business change. On the other hand, considerable effort is needed to develop a service by combining business and constraint consumes because the entire combination logic should be changed according to the change in business rule. To improve the current method, this paper applied an aspect oriented approach to service system development. In this paper, the rule concern is proposed in addition to the core concern and cross cutting concern of aspect oriented approach. The rule concern is extracted from business rules included in the business processes and services. The rule concern is classified into the process rule aspect and service rule aspect according to the level of the rule. In the proposed approach, system is modularized into the core concern, cross cutting concern and rule concern through separation of concern, and they are maintained independently. Therefore, the adaptability, reusability, and maintainability of a service system will be enhanced.

Key Words : Aspect-Oriented, Business Process, Rule Concern, Service System Development

1. 서론

서비스 지향 아키텍처의 서비스는 일반적으로 소프트웨어로 구현된 비즈니스 함수이며, 비즈니스의 각 기능과 이를 정의하는 비즈니스 룰(Business Rule)들의 모임이다[1]. 서비스들은 각기 다른 공급자들로부터 다양한(Heterogeneous) 형태로 제공되며, 이는 BPEL(Business Process Execution Language), BPML(Business Process

이 연구는 (2014년도) 강남대학교 교내연구비 지원에 의해 수행되었음.
*Corresponding Author : Il-Woo Choi (Kangnam University)
Tel: +82-31-2803-662 email: iwchoi@kangnam.ac.kr
Received July 24, 2014 Revised (1st September 29, 2014, 2nd October 27, 2014) Accepted January 8, 2015

Modeling Language)과 같은 비즈니스 프로세스 기술언어(Business Process Description Language)를 통하여 서비스들 간의 컨트롤 플로우와 데이터 플로우를 명세하며, 서비스들 간의 오케스트레이션(Orchestration)을 통하여 비즈니스 프로세스로 결합된다[2].

외부환경의 변화에 따라 비즈니스 프로세스는 계속적으로 변화하게 된다. 이러한 IT시스템과 비즈니스 프로세스의 변화는 필연적이다. 서비스 지향 아키텍처는 상호운용성, 위치투명성, 프로세스 중심 등의 특성을 가지는데 이러한 특성은 애플리케이션 개발 시 유연성을 보장하여, 비즈니스 변화에 최대한 민첩하게 대응하도록 지원한다[3].

그럼에도 불구하고 서비스내의 비즈니스 로직에 소스코드의 형태로 포함된 비즈니스 룰은 이러한 변화에 효과적으로 대응할 수 없다. 전체 비즈니스와 제약조건을 함께 묶어 코딩하는 이러한 접근법은 결과적으로 비즈니스 룰의 변경을 위하여 전체 결합로직을 변경하는 등 많은 자원을 소모하게 된다[2,4,5].

비즈니스의 변화는 일반적으로 비즈니스 룰의 변화로 나타나는데 비즈니스 룰은 전체 비즈니스 프로세스의 실행 효율을 좌우하기도 하며, 계속적인 변화는 비즈니스 룰의 관리 및 변경, 구현에 많은 시간을 소모시킨다. 이러한 비즈니스 룰의 효율적인 구현과 실행은 전체 비즈니스 프로세스에 큰 영향을 끼치므로 효율을 위하여 룰 엔진 등 다양한 방법을 활용한다[5].

관점 지향 방법은 일반적으로 시스템 도메인의 기능을 중심으로 핵심관심(Core Concern)과 여러 핵심관심에 분포되어있는 공통기능인 횡단관심(Cross Cutting Concern)으로 관심을 분리(Separation of Concern)한다[5]. 본 연구에서는 서비스 시스템의 적용성, 재사용성, 유지보수성의 향상을 위하여 시스템 도메인에 관심 지향 기법을 적용, 핵심관심과 횡단관심을 반영하여 서비스를 설계한다. 또한 새롭게 제시되는 관심사인 룰관심(Rule Concern)은 시스템에 포함된 비즈니스 룰(Business Rule)을 모듈화 한다. 룰관심은 효율적인 크로스커팅(Cross Cutting)을 위하여 프로세스 룰관심(PRA: Process Rule Aspect)과 서비스 룰관심(SRA: Service Rule Aspect)의 두 관점으로 모듈화 된다. 이러한 시도는 서비스 지향 아키텍처의 유연성 등의 장점과 더불어 관점 지향 기법의 관점에서 제공되는 느슨한 결합도를 통하여 비즈니스 룰의 효율적인 모듈화가 가능하다. 결과적으로 서비스 구현 및 서비스 오케스트레이션에 모듈성과 다이나믹한 적용성을 제공한다. 이는 서비스 뿐 아니라 비즈니스 프로세스의 적용성(Adaptability), 재사용성(Reusability) 및 유지보수성(Maintainability)을 높일 수 있다[6,7].

2. 관련연구

2.1 관점 지향 프로그래밍

관점 지향 프로그래밍(Aspect Oriented Programming)은 절차중심과 객체지향 프로그래밍에서 충분히 처리될 수 없는 문제를 다루기 위해 소개되었다. 시스템의 총체적인 레벨 체계를 절차 혹은 객체관점에서 수직적으로 분할하여 시스템의 기능과 사용자 서비스에 집중했던 기존의 종단 모듈 구조는 여러 개의 모듈에 공통적으로 적용되는 수평적인 요구사항을 모듈화 하지 못하였다.

관점 지향 프로그래밍은 기존의 수직적으로 시스템을 다루는 객체, 컴포넌트, 서비스 등의 패러다임을 핵심관심으로 정의하고, 이러한 핵심관심에 흩어져있는 공통기능을 크로스 커팅이라는 관점지향 모듈화를 통하여 횡단관심으로 모아서 각각의 관점으로 처리하는 패러다임이다.

일반적으로 횡단관심은 로그작성, 보안인증, 트랜잭션 등과 같은 시스템 공통 서비스들을 의미한다. 이러한 횡단관심 분리를 통한 시도는 다양한 재사용단위 객체, 컴포넌트, 서비스들의 재사용성과 유지보수성을 높일 수 있다[5].

2.2 프로세스 지향 결합 언어

일반적으로 웹 서비스는 비즈니스 프로세스와 그 프로세스와 연관된 파트너들 사이의 상호작용으로 기술되고 각 파트너들과의 상호작용은 웹 서비스 인터페이스인 웹 서비스 기술언어(Web Service Description Language)로 이루어진다. 웹 서비스를 위한 비즈니스 프로세스 실행 언어(BPEL, Business Process Execution Language)의 워크플로우 프로세스들은 서비스들 간의 컨트롤 플로우와 데이터 플로우의 결합 활동으로 구성되어 있다[2]. 이러한 비즈니스 프로세스 실행 언어는 협력하는 파트너와 밀접한 관련을 가지기 때문에 유사한 작업흐름을 가지는 유사한 프로세스로의 재사용이 어렵다. 웹서비스의 재사용도 중요하지만 이를 이용하는 프로세

스의 재사용역시 프로세스를 개발하는 개발자들에게는 중요한 문제로 남아있다.

기존의 레거시를 포함한 대형 엔터프라이즈 시스템은 일반적으로 비즈니스 로직과 비즈니스 룰이 결합되어 있다. 이러한 시스템의 룰 변경 및 유지보수는 시간 소모적인 작업이다. BPML과 BPEL같은 프로세스 지향 결합 언어들은 서비스들 간의 컨트롤 플로우와 데이터 플로우를 정의하여 서비스를 결합하나 계속적으로 변하는 비즈니스 룰을 효율적으로 시스템에 결합하는 방법을 고려하고 있지 못하다[4].

AO4BPEL은 BPEL에 관점 지향 개념을 추가하여 확장한 개념이다. 이러한 시도는 BPEL로 작성되는 웹 서비스 컴포지션 기능에 확장성을 제공한다. BPEL의 서비스 컴포지션 기능을 통하여 구축된 비즈니스 프로세스는 관점 지향 방법에서 제공되는 느슨한 결합도를 통하여 비즈니스 프로세스의 모듈화가 가능하며, 느슨한 결합도는 비즈니스 프로세스 코드의 재사용성을 높일 수 있다. 결과적으로 BPEL의 웹 서비스 오케스트레이션에 모듈성과 다이나믹한 적용성을 제공한다[7].

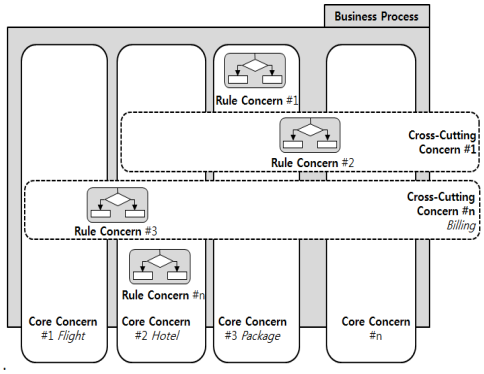
3. 룰 기반의 관점 지향 기법

3.1 룰관심

효율적인 서비스 구현 및 오케스트레이션은 비즈니스 프로세스를 수행하기 위한 흐름 제어와 서비스 호출을 처리하는 것 이외에도 새로운 서비스나 애플리케이션의 변화, 비즈니스 프로세스의 변화 등에도 능동적으로 대처하기 위해 프로세스의 재사용과 확장성 등을 고려하여야 한다. 그럼에도 불구하고 현재 서비스 구현 및 오케스트레이션은 단순히 서비스의 오퍼레이션과 입/출력 정보의 조합을 통해 이루어지고 있으며 웹 서비스를 위한 비즈니스 프로세스 실행 언어(Business Process Execution Language for Web Services, BPEL4WS)에서도 마찬가지 상황이다.

본 연구에서는 비즈니스 프로세스와 서비스에 능동적인 비즈니스 룰 적용을 위하여 관점 지향 기법을 확장하여 제시한다. 제시한 기법은 비즈니스 프로세스와 서비스에 포함된 비즈니스 룰을 기능과 분리시킴으로써 기존 방법에 비하여 적용성, 재사용성, 유지보수성을 높일 수 있다.

본 연구에서는 Fig.1과 같이 기존 관점 지향 기법의 관심사를 확장하여 핵심관심, 횡단관심 이외의 새로운 관심사인 룰관심을 정의한다.



[Fig. 1] Core Concern, Cross Cutting Concern, Rule Concern

일반적으로 핵심관심은 각 모듈이 수행해야 하는 본질적인 업무 기능을 나타내고, 횡단관심은 여러 모듈에 걸쳐 있는 공통적인 시스템의 부가적 기능을 나타낸다. 본 연구에서 추가적으로 제시하는 룰관심은 핵심관심과 횡단관심에 내포된 일부(part)의 비즈니스 룰로 존재한다. 룰관심은 효율적인 크로스커팅을 위하여 수준에 따라 프로세스 룰관심과 서비스 룰관심으로 구분된다.

프로세스 룰관심은 비즈니스 룰 중 다양한 웹 서비스들을 하나의 비즈니스 프로세스로 구성하는 비즈니스 프로세스 오케스트레이션 기능을 수행하는 룰이다. 프로세스 룰관심에 따라 비즈니스 프로세스에는 변화가 수행되어지며, 서비스 오케스트레이션에 변이가 생긴다. 일반적으로 BPEL 상에서 정의되어지므로 비즈니스 프로세스 설계 시 고려되어진다.

서비스 룰관심은 서비스 내에 구현되어 비즈니스 기능을 수행하는 룰로 개별 서비스 내에서 기능을 수행하거나 처리하는 로직에 포함된 룰을 의미한다. 일반적으로 서비스 내에 코드로 내포되어지므로 개별 서비스 설계 시 고려되어진다. 비즈니스 기능과 분리된 비즈니스 룰 즉, 룰관심은 서비스 시스템에 효율적인 적용, 재사용 및 유지보수를 지원한다.

3.1.1 프로세스 룰관심

룰관심 중 프로세스 룰관심은 “다양한 서비스들을 연

관된 일련의 비즈니스 프로세스로 구성하는 비즈니스 프로세스 오케스트레이션 기능을 수행하는 룰"로 정의한다. 하나의 서비스 내에 정의된 비즈니스 룰과는 달리 프로세스 룰은 비즈니스의 변화에 따라 비즈니스 프로세스 및 서비스 오케스트레이션에 변이가 생긴다. 이러한 특성을 가지는 프로세스 룰은 일반적으로 비즈니스 프로세스 기술 언어 즉 BPEL상에서 정의된다.

일반적인 비즈니스 프로세스는 다양한 조건들로 서비스들을 오케스트레이션하여 구성되는데 이러한 형식의 BPEL 기술은 다양한 형태의 서비스 룰의 변화에 시간소모적인 작업을 요구한다. 본 연구에서는 이러한 문제를 해결하기 위하여 프로세스 룰관심을 제시한다. 빈번한 룰과 서비스의 변화에 효율적인 대응을 하기 위하여 변화하는 프로세스 룰을 모듈화하여 독립적으로 관리하는 것은 각 서비스들의 효율적인 오케스트레이션을 제공한다.

이를 위하여 프로세스 룰관심은 AO4BPEL, B2J등을 이용하여 구현한다. AO4BPEL은 관점 지향 프로그래밍(AOP)의 관심(Aspect), 결합점(Joint-Point), 어드바이스(Advice), 교차점(Point-Cut), 직조(Weaving) 등의 장치를 BPEL의 확장을 통하여 제공한다[8,10]. AOPBPEL, B2J의 이러한 특징은 계속적으로 추가, 변경되어지는 다양한 프로세스 룰을 관점 지향 프로그래밍의 관심사를 통하여 프로세스 룰관심으로 모듈화하고 효율적으로 관리하는데 적용 가능하다.

3.1.2 서비스 룰관심

시스템 도메인에 관점 지향 기법을 적용하여 서비스를 추출하는 경우, 단일 서비스로 기능을 모듈화 하는 핵심관심과 다양한 핵심관심들 간에 공통적으로 수행되는 기능을 횡단관심으로 분리하여 모듈화 한다[9]. 이렇게 추출된 횡단관심의 경우 각각의 도메인에서 공통적인 기능을 모듈화 하였으나 다양한 각 서비스의 비즈니스 로직에 특화된 다양한 룰을 모두 내포하는 경우가 많다. 특히 횡단관심에 포함된 이러한 룰은 각각의 서비스에 따라 다양하게 기술되거나, 횡단관심을 호출하는 서비스에 따라 선택적으로 사용되거나, 빈번하게 수정하게 되는 가변점(Variation Point)으로 작용한다[7,11]. 이러한 가변점은 다양한 서비스에 적합한 룰을 선택적으로 수용할 수 있는 형태로 미리 코딩되어있거나, 다양한 서비스들의 요구사항에 따라 지속적으로 룰의 변경이 수행되어야 한다. 본 연구에서는 횡단관심에 포함된 가변점, 즉 가변적인 비즈니스 룰을 서비스 룰관심으로 분리한다.

[Table 1] Description of Concern

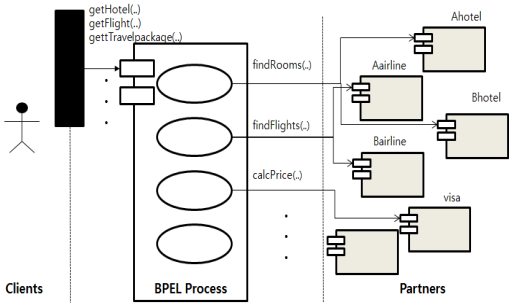
Aspect		Description
Core Concern		- Core concern is the business function that each module should perform. - A business process generally consists of several services.
Cross Cutting Concern		- Cross cutting concern is the common system function which covers several modules. - One cross cutting concern is implemented as a service.
Rule Concern	PRA (Process Rule Aspect)	- PRA is the business rule which makes a change in business process. - Service orchestration should be changed if PRA is changed. - PRA is generally related to the basic flow and the alternative flow of use case description. - PRA is implemented as a part of business process using aspect oriented business process execution language such as AO4BPEL or aspect oriented programming language such as AspectJ.
	SRA (Service Rule Aspect)	- SRA is the business rule which is included in the service. - SRA acts as the variation point of cross cutting concern. - SRA is generally related to the pre-condition of use case. The pre-condition is defined using <<include>>. - SRA is implemented in a service using aspect oriented programming language such as AspectJ.

서비스 룰관심은 "각 서비스의 비즈니스 로직과 분리된 가변적인 룰"을 의미한다. 이러한 서비스 룰관심의 추출은 기존 관점 지향 기법의 관심 분리와 동일하나 하나의 서비스에 포함된 가변적 룰의 모듈화에 관심을 둔다는 것에서만 차이를 가진다. 서비스 룰관심은 AspectJ 등의 관점 지향 언어를 통하여 서비스내에 구현된다. 각 관심사의 특징을 Table 1을 통하여 정리하였다.

3.2 룰기반 서비스 설계

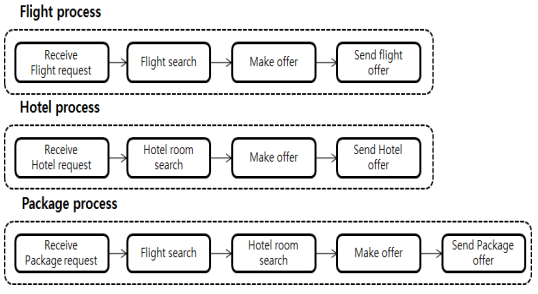
Fig.2는 룰기반 서비스 설계 기술을 위한 여행 포탈 웹 서비스 도메인의 개략적인 사례이다. 여행 포탈 서비스를 통한 클라이언트의 요구는 BPEL에 의하여 복합적인 서비스들로 오케스트레이션이 된 비즈니스 프로세스를

통하여 해당하는 파트너 서비스들을 이용하여 달성되어 진다.



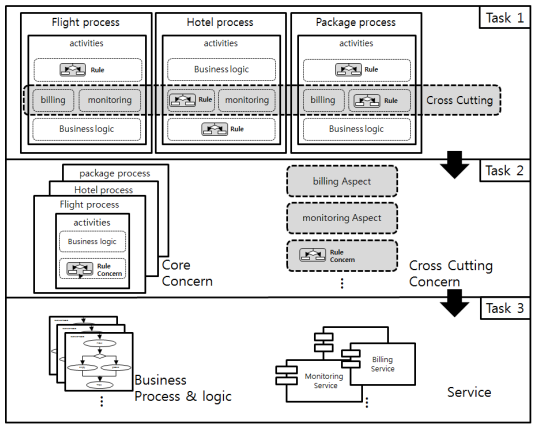
[Fig. 2] Example of travel portal as a composite web service

사례 기술을 위한 비즈니스 프로세스 시나리오의 개략은 Fig.3과 같다.



[Fig. 3] Abstract process of travel portal scenario

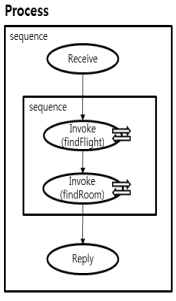
사례 도메인에 본 연구에서 제시하는 관점 지향 기법을 적용하기 위한 개략적인 절차는 Fig.4와 같다. 1, 2단계(Task)를 통하여 추출된 핵심관심과 횡단관심은 각각의 서비스로 추출되어 모듈화 한다[9]. 핵심관심에서 크로스커팅을 통하여 추출된 횡단관심들은 느슨한 결합을 유지하며 모듈화 된다. 추출된 각각의 서비스는 다양한 비즈니스 룰에 의하여 오케스트레이션 되어 하나의 비즈니스 프로세스를 구성한다. 3단계는 1, 2단계에서 추출된 서비스와 비즈니스 프로세스에 프로세스 룰관심과 서비스 룰관심을 분리하는 단계이다.



[Fig. 4] Task of Separate Concern

3.2.1 프로세스 룰관심 설계

정의한 프로세스 룰은 하나의 서비스 내에 정의된 비즈니스 룰과는 달리 비즈니스 룰의 변화에 따라 전체 비즈니스 프로세스에 변화가 수행되어 서비스 오케스트레이션에 변이가 생긴다. 이러한 특성을 가지는 프로세스 룰은 일반적으로 비즈니스 프로세스 기술 언어 즉 BPEL 상에서 정의된다.

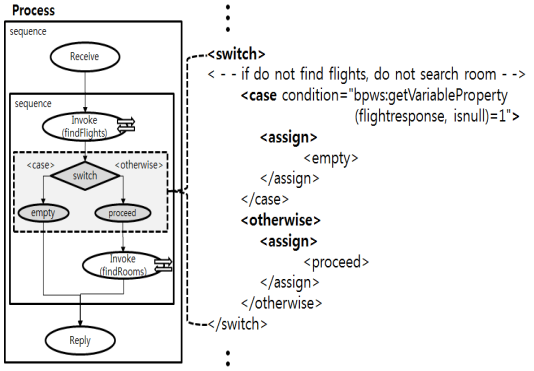



```
<processname="Package".../>
  <sequence>
    <receive  partner="client"
              operation="getTravelPackage"
              variable="request"
              createInstance="yes" .../>
    ...
    <sequence>
      <invoke     partner="airline"
                  operation="findFlight"
                  outputVariable="flightOut"/>
      <invoke     partner="hotel"
                  operation="findRoom" .../>
    </sequence>
    <assign>...</assign>
    <reply      partner="client"
                operation="getTravelPackage"
                variable="proposition" .../>
  </sequence>
</process>
```

[Fig. 5] An example of BPEL statement for Package process

Fig.5는 BPEL4WS을 이용하여 작성된 Package process의 일부이다. 서로 다른 파트너들에 의해 제공되는 서비스들은 BPEL을 통하여 오케스트레이션 되고 비즈니스 프로세스를 이루어 실행된다. 일반적으로 BPEL의 조건은 <switch>,<case>문을 통하여 수행되고 이러한 룰에 따라 서비스들의 오케스트레이션이 수행된다.

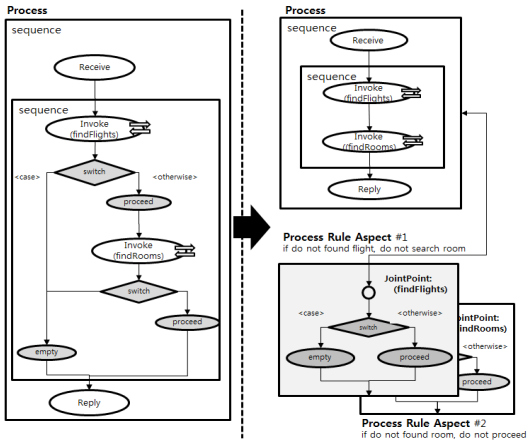
Fig.5에 “비행편을 찾지 못하면 객실을 검색하지 말아라”와 같은 새로운 룰이 적용되어지면 Fig.5에 Fig.6과 같은 BPEL문을 <switch>,<case>문을 통하여 추가하여야 한다. 이러한 룰의 변화는 BPEL문의 계속적인 수정, 변경을 요구하므로 전체적인 비즈니스 프로세스 오케스트레이션에 계속적으로 영향을 미친다.



[Fig. 6] Addition of BPEL statement according to the change of rule

본 연구에서 제시하는 관점 지향 기반 프로세스 룰관심을 적용, 관심사를 분리한 경우는 Fig.7과 같다. 각 프로세스 룰관심, 즉 서비스와 분리된 프로세스 룰은 AO4BPEL을 통하여 프로세스 실행을 위한 잘 정의된 결합점(Joint Point)을 정의하고 각 결합점에 삽입되어 실행될 수 있는 프로그램 코드인 <advice>내에 <switch>, <case>문으로 구현되고, 또한 연관된 결합점의 모임을 <pointcut>으로 식별한다. 프로세스 룰관심으로 작성된 모듈에서 교차점을 통하여 핵심관심, 횡단관심으로 분기한다. 교차점을 통하여 결합점에 어드바이스가 삽입, 직조(Weaving)된다. 그러므로 프로세스 룰의 변화가 요구되면 프로세스 룰관심만 변경을 수행하여 비즈니스 프로세스 내의 변화를 최소화 한다.

분리된 관심사는 AO4BPEL을 사용하여 비즈니스 프로세스를 이루는 서비스와 서비스들을 조합하는 프로세스 룰관심으로 분리, 구현이 가능하다. 전체 비즈니스 프로세스는 변하지 않고 분리된 프로세스 룰관심들의 변화에 의하여 서비스들은 선택적으로 활용되어지므로 룰 및 프로세스의 변화는 프로세스 룰관심에 의하여 독립적으로 수행하게 된다.




```
// Process Rule Aspect #1
// if do not found flight, do not search room

    <aspectname="ifdonotfoundflight,donotsearchroom">
    <variables>...</variables>
      <pointcutandadvicetype="around">

      <pointcutname="hotelprocurement">
       //process[@name="FullTravelPackage"]
       //invoke
       [@portType="hotelPT"and @operation="findRoom"]
      </pointcut>

      <advice>
      <switch>
      <case condition="bpws:getVariableProperty
                       (flightresponse,isnull)=1">
          <empty/>
      </case>
      <otherwise>
          <proceed>
```

[Fig. 7] An example of AO4BPEL statement for Separated Process Rule Aspects

B2J는 BPEL4WS를 지원하는 서비스 엔진 중 하나로 여타의 BPEL 엔진과는 다르게 BPEL 문서를 자바소스로 변환하여 다양한 응용을 제공한다. 현재 BPEL은 컨텍스트 정보를 분석하기 위한 직관적인 조건을 제공하지 않아 편재형(Pervasive) 컴퓨팅 환경에서 사용하기 어렵다. 이를 해결하기 위하여 개발자는 다양한 BPEL 도구들을 사용한다. 비즈니스 룰은 컨텍스트 정보를 분석하기 위하여 높은 수준의 조건을 제공한다. 그러므로 BPEL이나 AO4BPEL에 규칙을 처리하는 언어를 추가하면 BPEL의 프로세스 명세 기능과 웹 서비스 기능을 그대로 사용하면서 컨텍스트 정보를 효율적으로 처리할 수 있다.

B2J 엔진이 생성한 자바소스에 새로운 요구사항과 변화되는 룰을 효율적으로 관리하기 위한 관점 지향 프로그래밍 개념을 추가하기 위하여 AspectJ를 이용한다. 사용자는 BPEL 문서와 Fig.8과 같은 인터셉터(Interceptor) 문서를 작성하고 시스템은 이 두 문서를 입력으로 한다.

```
public class flightOutInterceptor extends AbstractInterceptor {
        public String intercept(ActionInvocation invocation) throws Exception {
        ActionContext context = invocation.getInvocationContext();

        FindFlight = (FindFlight)session.get("flight");
          if(flight==null){
                        returnAction.empty;
                        }
          returninvocation.invoke(); }}
```

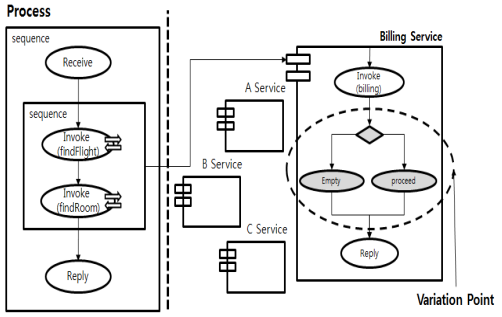
[Fig. 8] An example of Interceptor

인터셉터 문서는 BPEL 실행 루틴 중 특정 부분의 룰을 처리하기 위한 루틴을 삽입하기 위한 정보를 가지고 있다. BPEL 문서는 B2J 코디네이터를 통해 자바소스를 생성하고 인터셉터 문서는 Intercepter2AspectJ 엔진을 통하여 AspectJ 소스로 변환된다. 이렇게 생성된 두 프로그램 소스는 AspectJ 컴파일러를 통하여 컴파일 되어 실행 가능한 자바클래스 파일을 생성하고 B2J WorkerHost에서 실행된다[10,15].

인터셉터는 액션의 호출을 동적으로 가로채는 객체이다. 인터셉터는 액션 실행 전 후에 다른 실행코드를 넣을 수 있는 수단을 제공한다. 관점 지향 프로그래밍의 기능은 인터셉터와 BPEL 문서를 독립적으로 작성하여 인터셉터로 하여금 프로세스 룰을 또 다른 관심사인 프로세스 룰관심으로 모듈화하여 BPEL 문서에 새로운 요구사항을 손쉽게 추가, 변경하여 사용할 수 있게 한다. 또한 AO4BPEL에 비하여 자바언어의 구문을 활용하여 보다 직관적인 룰을 기술함이 가능하다.

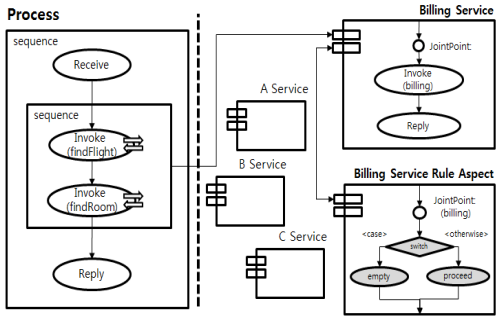
3.2.2 서비스 룰관심 설계

도메인에서 추출된 핵심관심과 횡단관심은 각각의 서비스로 모듈화 가능하다[9]. 횡단관심은 크로스커팅을 통하여 느슨한 결합을 유지하며 모듈화 되었으나 Fig.9의 Billing Service의 경우 다른 서비스들과 마찬가지로 기능과 다양한 룰이 하나의 Billing Service내에 코드로 내포되어진다. 내포되어진 비즈니스 룰은 횡단관심을 호출하는 핵심관심의 조건에 따라 지속적인 변경이 수행되는 가변점(Variation Point)이다. 가변점은 기존방법으로 서비스 코드에 내포되어진 룰은 결과적으로 룰의 변경을 위하여 전체 결합로직을 변경하는 등 많은 자원을 소모하게 된다.



[Fig. 9] An example of Variation Point (Billing Cross Cutting Concern)

이러한 문제는 Fig.10과 같이 서비스에 포함된 가변점의 비즈니스 룰을 서비스 룰관심으로 분리, 모듈화하여 해결한다. AspectJ의 인터셉터를 통하여 구현된 Billing Service Rule Aspect는 서비스 내의 룰의 변화가 요구되면 모듈화된 Billing Service Rule Aspect만 수정하므로 Billing Service 내의 변화를 최소화 할 수 있다.



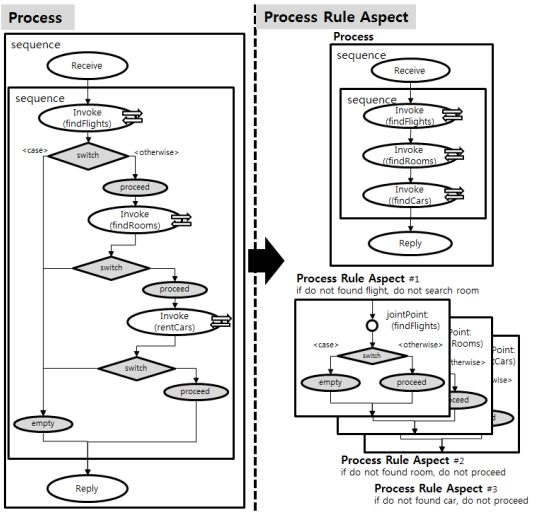
[Fig. 10] An example of Service Rule Aspect

인터셉터가 액션의 호출을 동적으로 가로채 액션 실행 전 후에 다른 실행코드를 넣을 수 있는 수단을 제공해 줌으로써 액션이 다양한 방법으로 실행될 수 있도록 한다. 인터셉터는 한개 이상의 액션에 공통적으로 재사용할 수 있는 횡단관심과 같은 일반적인 기능을 모듈화 하는 경우와 다양한 조건의 형태로 실행되는 서비스 룰관심과 같은 가변점을 캡슐화하는데 적합하다[12].

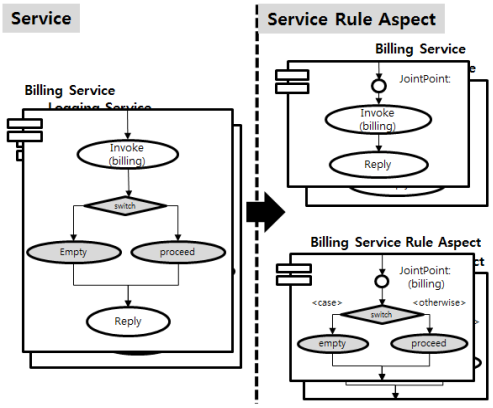
4. 평가

Fig.11과 Fig.12는 여행 포탈 웹 서비스 도메인에 사례 시나리오를 기반으로 프로세스 룰관심과 서비스 룰관심

을 분리하여 설계한 결과의 일부이다.



[Fig. 11] Separate Concern of travel portal scenario Process Rule Aspect



[Fig. 12] Separate Concern of Services Rule Aspect

시스템 도메인의 비즈니스 서비스들은 관점 지향 프로그램 관점으로 핵심관심과 횡단관심으로 추출, 모듈화가 수행되며 또한 제시한 룰관심을 통하여 프로세스 룰관심, 서비스 룰관심으로 각각 모듈화를 수행한다. 이러한 핵심관심, 횡단관심, 룰관심들은 AO4BPEL, AspectJ과 같은 관점 지향 언어와 관점 지향 비즈니스 프로세스 실행 언어의 코드로 구현되어지므로 서비스 분석 시점에서 고려하여야 한다[13].

[Table 2] The comparison of the proposed approach with existing approach

	Existing approach	Proposed approach
Architecture	SOA	SOA
Concern	Core concern, Cross cutting concern	Core concern, Cross cutting concern, Rule concern (Process rule, Service rule)
Maintainability	Good	Higher than the existing approach
Reusability	Good	Higher than the existing approach
Adaptability	Good	Higher than the existing approach
Reusable unit	Service	Service, Process, Rule
Design complexity	Average	More complex than the existing approach

Table 2를 통하여 기존 방법과 제시한 방법의 장단점을 비교, 평가하였다. 본 연구에서 제시하는 비즈니스 프로세스기반 룰관심의 분리는 비즈니스 프로세스와 서비스에 포함된 비즈니스 룰의 계속적인 변화에 적용성(Adaptability)을 증가시킨다. 프로세스 룰관심은 비즈니스의 변화에 따라 비즈니스 프로세스와 서비스 오케스트레이션에 변이가 생긴다. 비즈니스 정책이 변경되면 사용자는 해당 모듈의 AO4BPEL의 어드바이스나 AspectJ의 인터셉터로 기술된 프로세스 룰관심의 특정 룰만을 수정하면 되므로 전체적인 비즈니스 프로세스를 수정할 필요가 없어 유지보수성(Maintainability)이 향상된다. 또한 모듈화된 룰은 비즈니스 프로세스 코드의 재사용성(Reuasbility)을 높일 수 있다[14].

서비스 측면에서 보면 서비스 룰관심은 기존 서비스 형태에 비하여 AspectJ 등의 관점 지향 언어로 구현된 느슨한 결합도를 통하여 가변점의 수정 및 변경을 수행함으로써 서비스 룰의 기능 확장 및 유지보수에 유리하다. 그러나 이러한 관심사의 분리는 서비스 설계 절차를 복잡하게 하고 코드 흩어짐, 얽힘 등 설계복잡도(Design Complexity)측면의 추가적인 문제를 발생시킬 수 있다.

5. 결론

본 연구에서는 효율적인 서비스 시스템의 개발을 위한 룰 기반의 관점 지향 기법을 제시하였다.

제시한 기법은 서비스와 비즈니스 프로세스에 비즈니스 룰의 능동적인 적용을 위하여 비즈니스 프로세스에 포함된 비즈니스 룰을 프로세스 기능과 분리시키는 것으로, 관점 지향 기법을 적용하여 핵심관심, 횡단관심 이외

에 시스템 도메인에 포함된 비즈니스 룰을 룰관심이라는 새로운 관심사로 모듈화 하였다.

룰관심은 수준에 따라 프로세스 룰관심과 서비스 룰관심으로 구분된다. 프로세스 룰관심은 비즈니스 프로세스에 변화를 주는 비즈니스 룰로 서비스 오케스트레이션을 위하여 변경되며, 서비스 룰관심은 단위 서비스에 포함된 비즈니스 룰로 횡단관심 내에서 핵심관심의 다양한 조건의 수용을 위한 가변점이 된다.

시스템은 이러한 관심사의 분리를 통하여 핵심관심, 횡단관심 및 룰관심으로 각각 모듈화 되고 독립적으로 유지보수 되어 서비스 시스템의 적용성, 재사용성 및 유지보수성을 높일 수 있다.

향후연구로 새로운 관심사인 프로세스 룰관심과 서비스 룰관심을 효율적으로 식별하는 체계적인 식별 방법이 요구되어진다.

References

[1] Roy W. Schulte Yefim V. Natis, “Service Oriented Architecture,“ Gartner Group. SSA Research Note SPA-401-068. 1996.

[2] F. Curbera et. al. Business Process Execution Language for Web Services, version 1.1, May 2003

[3] W3C Web Services WG, “Web Services Architecture,” http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/.W3C, Working Group Note 11, Feb 2004.

[4] A Charfi and M. Mezini. "Hybrid Web Service Composition: Business Processes Meet Business Rules," In proceedings of the 2nd International Conference on Service Oriented Computing, November 2004

[5] M.A. Cibrán and B. Verheecke, “Dynamic Business Rules for Web Service Composition,” Proc. of the 2nd Dynamic Aspects Workshop DAW05, March 2005.

[6] M.A. Cibrán, M. D'Hondt, and V. Jonckers, “Aspect-Oriented Programming for Connecting Business Rules,” Proceedings BIS, 2003.

[7] Chankyu Park et al. “Knowledge-Based AOP Framework for Business Rule Aspects in Business Process,” ETRI Journal, Volume 29, Number 4, August 2007
DOI: http://dx.doi.org/10.4218/etrij.07.0106.0145

[8] A. Charfi and M. Mezini, “Aspect Oriented Web Service Composition with AO4BPEL,” Proc. of the European Conference on Web Services ECOWS 2004, LNCS 3250.

[9] Jacobson, Ivar, Aspect-oriented software development with use cases, Addison-Wesley, 2005.

[10] Gregor Kiczales, John Lamping, Anurag Mendhekar, Chris Maeda, Cristina Videira Lopes, Jean-Marc Loingtier, and John Irwin, “Aspect-Oriented Programming,” ECOOP, pp220-242, 1997.

[11] Adam Przybyłek, "Where the truth lies: AOP and its impact on software modularity," FASE'11/ETAPS'11, March 2011.

[12] Shaukat Ali, Tao Yue, Lionel C. Briand, "Does aspect-oriented modeling help improve the readability of UML state machines", Software and Systems Modeling (SoSyM) , Volume 13 Issue 3, July 2014.

[13] G. H. Alférez, V. Pelechano, R. Mazo, C. Salinesi, D. Diaz, "Dynamic adaptation of service compositions with variability models", Journal of Systems and Software , Volume 91, May 2014.
DOI: http://dx.doi.org/10.1016/j.jss.2013.06.034

[14] Eric Bodden, Éric Tanter, Milton Inostroza, "Join point interfaces for safe and flexible decoupling of aspects", Transactions on Software Engineering and Methodology (TOSEM), Volume 23 Issue 1, February 2014

[15] B2J,”http://www.eclipse.org/stp/b2j”

이 우 진(Woo-Jin Lee) **[정회원]**

- 2002년 2월 : 숭실대학교대학원 (컴퓨터공학석사)
- 2007년 2월 : 숭실대학교대학원 (컴퓨터공학박사)
- 2009년 3월 ~ 현재 : 세종대학교 정보통신공학과 초빙교수

<관심분야>
센서네트워크, 모바일컴퓨팅, 소프트웨어개발

최 일 우(Il-Woo Choi) **[정회원]**

- 1997년 2월 : 숭실대학교대학원 (컴퓨터공학석사)
- 2004년 2월 : 숭실대학교대학원 (컴퓨터공학박사)
- 2007년 3월 ~ 현재 : 강남대학교 교양학부 교수

<관심분야>
개발 프로세스, 재사용, SOA, USN, 모바일컴퓨팅