BPEL4WS을 이용한 동적이고 재사용가능한 웹 서비스 통합 모델

Dynamic and Reuseable Composition Model of Web Services using BPEL4WS

김 운 용*
Woon-Yong Kim

요 약

웹 서비스는 웹 표준인 SOAP과 WSDL 그리고 UDDI 기반으로 인터넷상의 분산된 어플리케이션을 통합하기 위한 효율적인 방법으로 널리 인식되고 있으며 최근에는 이 서비스들의 효율적인 통합을 위한 웹 서비스 표준 프로세스 통합모델 언어로 BPEL4WS가 제시되었다. 현재 웹 서비스 통합에 대한 연구는 주로 서비스들 간의 호환성 문제 및 프로세스 식별과 추적 그리고 자동화 문제 등을 다루고 있지만 통합된 비즈니스 프로세스 재사용이나 확장성 그리고 서비스들의 동적 할당을 통한 활용 기법에 대한 연구 가 부족하다. 이에 본 논문에서는 BPEL4WS을 이용하여 웹 서비스들을 통합할 때 통합 구조의 확장성과 재사용성을 증가시키는 기법과 웹 서비스 동적 활용방법을 제시한다. 또한 제시된 통합 구조의 효율적인 활용을 위해 필요한 웹 서비스 통합 프레임 워크를 제시한다.

Abstract

Web services are being recognized as the efficient way to integrate the distributed applications on the web based on the web standards such as SOAP, WSDL and UDDI. Recently, BPEL4WS is suggested as a standard integration model language for web services to integrate them efficiently. The recent studies of web service integration are focused on the comparability among the services, process identification and tracing, and automation. However, little or no effort has been dedicated to the reusability, extensibility on the business process integrated services and how to allocate the services dynamically. So, in this paper, we propose how to increase the reusability and extensibility of integrated structure when services are integrated using BPEL4WS, and how to allocate and use the services dynamically. Also, we propose the framework of the composition of web services for the efficient use of suggested integrated structure.

☞ Keyword : web services, business process, service composition, service reusability, BPEL4WS

1. 서 론

웹의 급속한 보급으로 사용자들은 보다 더 많은 서비스들을 쉽게 얻을 수 있고 서비스 공급자들은 보다 빠르고 편리하게 서비스들을 공급할 수 있게 되었다. 또한 B2B나 B2C 등과 같은 비즈니스 서비스들 간의 공유를 통한 분산 컴퓨팅 환경의 시대를 열어가고 있다. 이러한 분산 컴퓨팅 환경에서 서비스들의 통합을 위한 해결책으로 웹 서비스가 제시되었다. HTTP와 XML을 기반으로 한 웹 서비스는 웹 서비스 통신 프로토콜인 SOAP(Simple Object Access Protocol)과 웹 서비스에 대한 정보 기술서인 WSDL(Web Service Description Language) 그리고 웹 서비스 정보 공유 저장소 표준인 UDDI(Universal Discovery Description Integration)을 포함하고 있으며 인터넷상의 어플리케이션 통합을 위한 효율적인 방법으로 인식되고 있다[6]. 또한 서비스들의 통합 문제를 다루기 위해 BPEL4WS가 최근에 웹 서비

* 정 회 원 : MAXSoft Co.,LTD. 기술연구소 책임 연구원
wykim@kw.ac.kr(제 1저자)
[2004/08/14 투고 - 2004/09/13 심사 - 2004/09/04 심사완료]

스 표준으로 채택되었다[4]. 웹 서비스는 표준명세서를 기반으로 느슨하게 결합한 구조를 가지며 플랫폼에 독립적인 모델을 통해 상호작용 하여 구현 모델에 추상적인 관점을 제공하고 분산된 환경의 서비스들을 통합한다. 이 과정에서 서비스들의 통합은 웹 서비스에서 중요한 연구 분야가 되고 있다. 그러나 현재 웹 서비스 통합에 대한 연구는 주로 서비스들 간의 호환성 문제[13] 및 프로세스 추적 및 보안[7,2], 패턴분석[10] 그리고 자동화 문제[9] 등을 다루고 있으며 통합된 비즈니스 프로세스 재사용이나 확장성 그리고 서비스들의 동적 할당 기법에 대한 연구 가 부족하다.

일반적으로 웹 서비스 통합 구조는 비즈니스 프로세스와 그 프로세스와 연관된 파트너들 사이의 상호작용으로 기술되고 각 파트너들과의 상호작용은 웹 서비스 인터페이스(WSDL)를 통해 이루어진다. 최근에 표준으로 채택된 BPEL4WS가 이러한 통합을 위해 제시된 대표적인 언어이다[4]. 그러나 이러한 BPEL4WS는 협력하는 파트너와 밀접한 관련을 가지기 때문에 유사한 작업흐름을 가지는 유사한 프로세스의 재사용이 어렵다. 웹 서비스의 재사용 또한 중요하지만 이를 이용하는 프로세스의 재사용 역시 프로세스를 개발하는 개발자들에게는 중요한 문제로 남아있다. 예로 주문 프로세스는 도서 주문, 차량 주문 그리고 전자제품 주문 프로세스 등으로 다양한 프로세스 형태로 표현 될 수 있으며 이때 이들이 유사한 작업 흐름을 가질지라도 다른 파트너 관계에 의해 각기 다른 통합 모델을 만들어 내야하는 단점을 가진다. 이러한 이유는 BPEL4WS에서 비즈니스 프로세스가 파트너의 서비스와 연관된 인터페이스(WSDL)에 직접적으로 종속되기 때문이다. 이러한 문제는 비즈니스 프로세스의 효율적인 재사용과 확장성을 저하시킨다.

이러한 문제점을 해결하기 위해 본 논문에서는 BPEL4WS 환경에서 웹 서비스 통합 시 비즈니스 프로세스의 재사용성과 확장성을 증가시키는 기법을 제시하고 서비스들의 동적 활용 기법을 다룬다. 이를 위해 먼저 비즈니스 프로세스의 의존성 분석을 통해 작업 흐름과 WSDL의존 정보를 식별하고 이를 기반으로 재사용과 서비스 동적할당 정책을 수행하기위한 표현 언어를(DAL4WS: Dynamic Assignment Language for Web Services)를 제시한다. 또한 이 구조의 효율적인 활용을 위해 요구되는 프레임워크를 보인다. 이를 통해 비즈니스 프로세스는 재사용가능한 구조를 가지고 동적인 접근을 통해 효율적인 활용성을 제공할 것이다.

본 논문의 구성은 다음과 같다. 먼저 2장에서 서비스 통합 관련 연구를 보이고 3장에서 재사용과 확장 가능한 서비스 통합 구조를 제시한다. 그리고 4장에서는 이 서비스 통합 구조에서 웹 서비스의 동적 활용을 위한 프레임 워크 구조와 실행 과정을 보이고 마지막 5장에서 결론과 향후 과제에 대해 이야기한다.

2. 관련 연구

2.1 웹 서비스 통합

주로 웹 서비스의 통합은 객체지향 언어나 비즈니스 프로세스 모델링 언어를 사용하여 이루어진다. 이들 중 비즈니스 프로세스 모델링 언어는 서비스 지향 아키텍처(SOA: Service Oriented Architecture)환경을 가장 잘 수용할 수 있는 언어로 인식되고 있으며 정형화된 형태와 서비스 관점의 통합을 보다 자연스럽게 해결하는 특징을 가진다. 이러한 비즈니스 프로세스 모델링 언어로 WSCI[12], BPML[5], BPEL4WS[4] 그리고 BPSS[11]등 다양한 언어가 소개되었다. 웹 서비스 통합에 대한 연구는 주로 호환성[13] 및 통합에 따른 프로세스 추적[7] 그리고 보안 문제[2]들에 집중되고 있다. 그리고 웹 서비스 통합의 효율적인 구조 생성을 위한 연구로는 서비스 통합에서 발생되는 다양한 패턴들의 분석[10,8]방법과 웹 서비스 통합 설계 과정에서 재사용성을 증가시

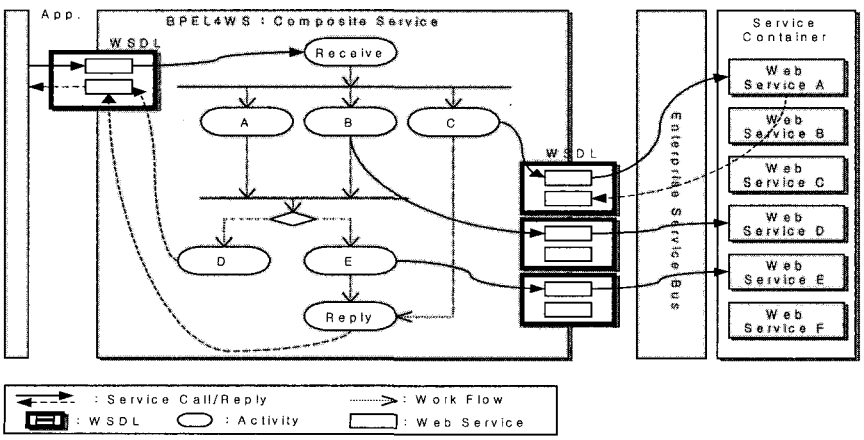
키기 위한 도구들을 제시하고 있다[3]. 이 도구는 설계과정에서 비즈니스 프로세스활동을 객체로 구성하고 새로운 비즈니스 프로세스를 생성 시 재사용 할 수 있는 기법을 제시하고 있으며 특정 프로그램 언어를 통해 비즈니스 프로세스활동을 클래스화하고 이를 객체 지향적인 접근방법으로 재사용하고 있다. 반면에서 본 논문에서는 XML기반으로 BPEL4WS의 파트너 종속적인 부분을 DAL4WS을 통해 그룹화하고 동적으로 할당하는 과정을 통해 비즈니스 프로세스를 재사용하고 있으며 설계과정보다는 런 타임 환경에서 비즈니스 프로세스의 구체적인 정보를 재사용함으로써 동적 비즈니스 프로세스 환경을 제공하는데 그 목적을 가진다.

본 논문에서는 다음과 같은 특징을 제공한다.

- 웹 서비스 표준인 BPEL4WS을 기반으로 재사용과 확장성을 고려한 동적 활용 정책을 제시한다.
- 웹 서비스 통합 시 재사용가능한 작업 흐름과 서비스 요소를 분리하고 관련 서비스의 그룹화를 통해 비즈니스 프로세스와 웹 서비스 그룹의 동적인 재사용과 확장성을 제공한다.

2.2 BPEL4WS

최근에 웹서비스 표준으로 채택된 BPEL4WS는 IBM의 WSFL(Web Services Flow Language)[14]와 Microsoft의 XLANG (Web Services for Business Process Design)[15]의 장점들을 조합하여 구성한 언어이다. 이 언어는 XML 기반 명세언어로 두 가지 타입의 프로세스 모델링을 지원한다. 하나는 추상적 프로세스 모델링으로 내부적인 실제 행위 관점보다는 집단간의 메시지 교환 행동에 대한 비즈니스 프로토콜을 모델링 한다. 다른 하나는 수행 가능한 프로세스(executable process) 모델링으로 비즈니스 상호작용에서 프로세스를 구성하는 많은 행위(activities)사이의 수행 흐름을 모델링 한다. 이것은 또한 프로세스 안에 파트너들(partners)을 포함하고 있으며 이들 파트너들 사이에서 메시지 교환이 이루어진다. 본 논문에서는 이 수행 가능한 프로세스 모델을 이용한다. 일반적으로 BPEL4WS는 작업 흐름으로 표현되며 이들 흐름의 단위는 활동(activity)으로 나타낸다. 이 활동단위는 크게 기본 활동(basic activity)과 구조적 활동(structured activity)으로 나누어지며 기본 활동은 서비스 호출과 할당, 프로세스 종료와 같은 단일 활동에 대한 표현을 제공하고 구조적 활동은 기본활동의 흐름인 수행 순서, 조건처리, 반복처리, 병렬처리 등의 작업을 담당한다. 아래 그림 1은 BPEL4WS를 이용한 웹 서비스 통합 구조와 구성요소들 간의 관계를 보여준다.



〈그림 1〉 BPEL4WS을 이용한 웹 서비스 통합 구조

그림 1에서 보는 것처럼 BPEL4WS는 웹 서비스를 제공하는 파트너들과 협력하는 비즈니스 프로세스이다. 이때 파트너들과의 협력은 서비스 인터페이스(WSDL)에 의존하여 이루어진다.

3. 재사용과 확장 가능한 서비스 통합 구조

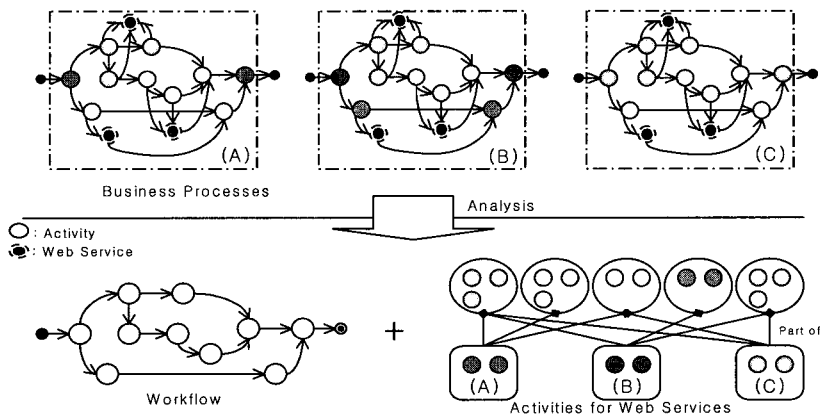
3.1 재사용과 확장 가능한 웹 서비스 통합

효율적인 웹 서비스 통합은 비즈니스 프로세스를 수행하기위한 흐름제어와 서비스 호출을 처리하는 것 이외에도 새로운 서비스나 어플리케이션 변화에 능동적으로 대처하기위해 통합 구조의 재사용과 확장성 등을 고려해야한다. 이러한 사실에도 불구하고 현재 웹 서비스 통합은 단순히 서비스의 오퍼레이션과 입/출력 정보의 조합을 통해 이루어지고 있으며 BPEL4WS에서도 비슷한 상황이다. 그러나 유사한 비즈니스 프로세스를 가진 웹 서비스들의 통합은 프로세스와 서비스 의존정보의 분할과 이들의 동적인 합병을 통해 비즈니스 프로세스의 재사용과 확장성을 제공할 수 있으며 웹 서비스의 변화에 빠르게 수용할 수 있을 것이다. 예를 들어 설명하기 전에 파트너들과 협력을 위해 사용되는 서비스와 이를 통합하는 서비스와의 구분을 위해 이 논문에서는 통합 서비스를 비즈니스 프로세스라는 용어를 사용하여 표현한다. 예를 위해 주문 비즈니스 프로세스를 고려해 보자. 이 주문 비즈니스 프로세스는 도서주문 비즈니스 프로세스와 차량주문 비즈니스 프로세스 그리고 가전제품주문 비즈니스 프로세스 등으로 구체화 될 수 있고 이들 구체적 비즈니스 프로세스는 유사한 작업 흐름을 가지지만 서로 다른 파트너의 웹 서비스를 이용한다고 가정하자.

이때 각각의 구체적 비즈니스 프로세스를 위해 그림 2의 상위에 표현되는 것처럼 각각의 목적에 필요한 파트너들과 협력하여 독립적인 비즈니스 프로세스를 만들어 낼 수 있다. 그러나 이들 각각의 비즈니스 프로세스는 유사한 작업 흐름을 가지고 있다. 이러한 사실은 그림 2의 하단에서 보이는 것처럼 특정 서비스에 독립적인 프로세스와 서비스에 의존하는 정보를 분리하고 이들을 동적으로 결합함으로써 프로세스의 재사용을 가능하게 하고 확장성을 증가 시킬 수 있다. BPEL4WS에서 비즈니스 프로세스는 웹 서비스 파트너들과 협력하기위해 활동(activity)을 이용한다. 또한 이 활동들 간의 정보 교환을 위한 자원을 유지한다.

이러한 특징을 통해 우리는 다음과 같은 분석 결과를 이끌어낸다.

먼저 비즈니스 프로세스(BP)는 다음과 같은 식으로 정의된다.



〈그림 2〉 비즈니프로세스와 구성요소 관계

$BP=<P,A,R>, P$: $PartnerLinks, A$: $Activities$, R: $Resources$

비즈니스 프로세스는 파트너 링크, 활동 그리고 이들 활동들 간의 정보 교환을 위한 자원들로 구성된다. 이때 파트너 링크는 $\forall p \in P, p=<rde, S>$로 표현되고 여기에서 role은 활동과 관계를 나타내는 역할과 구체적인 서비스 S로 표현할 수 있다. 또한 활동은 다음과 같이 표현된다.

$\forall a \in A, a=<w,op,I,O>, w: work, op: operation, I: input, O: output$

각각의 활동 a는 파트너와 협력할 작업(w)과 이 작업에서 구체적으로 호출되는 오퍼레이션(op) 그리고 입(I)/출(O)력 정보로 표현될 수 있다. 여기에서 두 비즈니스 프로세스의 재사용성과 확장성을 고려하기위해 각 활동들 간의 유사성을 평가할 필요가 있다. 이 논문에서는 두 활동들 간의 유사성을 표현하기위해 $\cong$과 같은 표기 식을 이용한다.

$\mathrm{if}(a'.w = a.w), where(\exists a' \in BP'.A, \exists a \in BP.A)$
$BP.A' \cong BP.A$

두 비즈니스 프로세스 활동의 유사성은 구체적인 작업을 제외한 협력 작업(w)의 비교를 통해 얻어낼 수 있다. 즉 다른 비즈니스 프로세스 활동의 작업과 동일한 작업이 동일하다면 이 두 활동은 유사한 활동이 된다. 결국 다음과 같은 식을 만족한다.

$BP' \cong BP, \mathrm{if}(BP(BP.A \cong BP'.A))$

다른 비즈니스에 속한 활동들의 추상적인 작업들이 해당 비즈니스의 활동들의 작업과 유사할 경우 두 비즈니스 프로세스는 유사한 구조를 가지고 있다. 결국 두 비즈니스 프로세스는 재사용과 확장이 가능한 구조를 가지고 있다.

그림 2에서 3개의 비즈니스 프로세스는 파트너들의 구체적인 연결 정보를 제외한 1개의 비즈니스 프로세스와 그룹화 된 구체적 비즈니스 행위 정보들로 나누었다. 이 정보는 비즈니스 프로세스와 연관된 파트너의 관점에서 활동과 자원들로 이루어지고 실제 비즈니스 프로세스 실행 과정에서 동적으로 연결되어 활용 된다. 다음 절에서 이들 구체적인 행위 정보를 관리하고 동적인 활용을 제공하기위한 표현 언어를 제시한다.

3.2 구체적 행위 정보관리와 동적할당을 위한 표현 언어 스키마 구조 (DAL4WS: Dynamic Assignment Language for Web Services)

DAL4WS는 비즈니스 프로세스 실행 과정에서 구체적 행위와 자원에 대한 정보를 유지 관리하고 동적으로 할당하기위해 사용된다. 이 스키마는 <processContent>의 루트 요소를 가지고 <groups>와 <wsdlContents> 자식 요소를 가진다. <groups>는 비즈니스 프로세스에서 사용되는 모든 활동들의 그룹정보를 포함하고 있으며 <wsdlContents> 정보는 비즈니스 프로세스가 구체화 될 때 사용되어지는 WSDL 정보를 포함하고 있다. 또한 <groups>의 자식요소로 <group>을 포함하고 있으며 이 <group>은 구체화 될 파트너 관련 정보를 포함하는 재사용가능한 단위이다. <group>은 name과 abstract 속성을 가진다. 여기에서 name은 파트너 관련 프로세스 이름이 되고 abstract 속성은 비즈니스 프로세스로 구체화 될지 다른 비즈니스 프로세스의 부분으로 활용될지를 나타낸다. 이<group>속성은 5가지의 자식요소인 <composite>, <partnerLinks>, <variable>, <correlationSet>, <activities>를 가진다. <composite>는 이 그룹에 포함되어질 다른 그룹 정보를 포함한다. 이것은 그룹에 포함된 다른 모든 그룹의 모든 내용이 이곳에서 사용된다는 것을 의미한다. <partnerLinks>, <variables>, <correlationSets>은 이 그룹에서 사용되는 파트너와 연관된 데이터 정보를 나타낸다. <activities>에서는 비즈니스 프로세스를 구성하는 활동에 대한 구체적인 부분을 담고 있다.

이것 역시 이 그룹에서 사용되는 파트너 정보를 통해 구성된다. 그림 3은 이 DAL4WS의 스키마일부를 보여주고 있다.

```
<element name="processContent" type="processContentType" />
- <complexType name="processContentType">
  <attribute name="name" type="string" />
  - <sequence>
    <element name="groups" type="groupsType" />
    <element name="wsdlContents" type="wsdlContentsType" />
  </sequence>
</complexType>
- <complexType name="groupsType">
  <element name="group" type="groupType" minOccurs="1" maxOccurs="unbounded" />
</complexType>
- <complexType name="groupType">
  <attribute name="name" type="string" />
  <attribute name="abstract" type="tns:boolean" default="no" />
  - <sequence>
    <element name="composite" type="compositeType" minOccurs="0" />
    <element name="partnerLinks" type="parnerLinksType" minOccurs="0" />
    <element name="variables" type="variablesType" minOccrus="0" />
    <element name="correlationSets" type="correlationSetsType" minOccurs="0" />
    <element name="activities" type="activitiesType" />
  </sequence>
</complexType>
- <complexType name="compositeType">
  - <element name="group" minOccurs="0" maxOccurs="unbounded">
    - <complexType>
      <attribute name="name" type="QName" />
      </complexType>
    </element>
  </complexType>
- <complexType name="partnerLinksType">
  - <element name="partnerLink" minOccurs="0">
    - <complexType>
      <attribute name="name" type="QName" />
      <attribute name="partnerLinkType" type="QName" />
      <attribute name="myRole" type="string" minOccurs="0" />
      <attribute name="partnerRole" type="string" minOccurs="0" />
      </complexType>
    </element>
  </complexType>
- <complexType name="variablesType">
  - <element name="variable" minOccurs="0">
    - <complexType>
      <attribute name="name" type="QName" />
      <attribute name="messageType" type="QName" />
      </complexType>
    </element>
  </complexType>
```

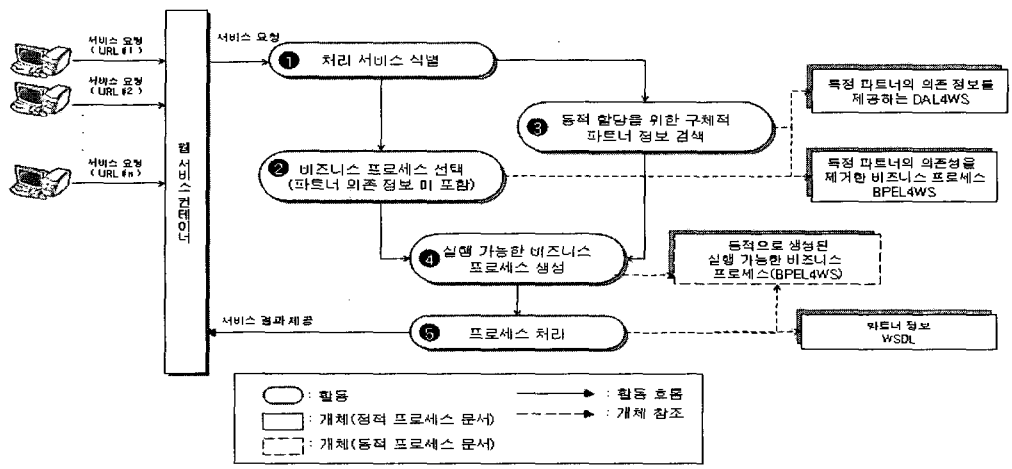
```
- <complexType name="correlationSetsType">
  - <element name="correlation" minOccurs="0">
    - <complexType>
      <attribute name="name" type="QName" />
      <attribute name="messageType" type="QName" />
      </complexType>
    </element>
  </complexType>
- <complexType name="activitiesType">
  - <element name="activity" minOccurs="0" maxOccurs="unbounded">
    - <complexType>
      <attribute name="name" type="QName" />
      <attribute name="portType" type="QName" />
      <attribute name="operation" type="QName" />
      - <element name="variable" minOccurs="0">
        - <complexType>
          <attribute name="name" type="QName" />
          <attribute name="part" type="string" minOccurs="0" />
          <attribute name="query" type="XQuery" minOccurs="0" />
          </complexType>
        </element>
      </complexType>
    </element>
  </complexType>
- <complexType name="wsdlContentsType" minOccurs="0">
  - <element name="wsdlContent" minOccurs="0" maxOccurs="unbounded">
    - <complexType>
      <attribute name="group" type="QName" />
      - <element name="wsdl" minOccurs="0" maxOccurs="unbounded">
        - <complexType>
          <attribute name="name" type="string" />
          <attribute name="namespace" type="anyURI" />
          <attribute name="location" type="anyURI" />
          </complexType>
        </element>
      </complexType>
    </element>
  </complexType>
```

〈그림 3〉 DAL4WS 스키마 구조

3.3 DAL4WS을 이용한 동적인 웹 서비스통합 기법

DAL4WS을 이용한 동적 웹 서비스 통합은 특정 파트너 정보를 포함하지 않는 BPEL4WS와 특정 파트너 정보를 제공하는 DAL4WS와의 동적 통합을 통해 웹 서비스가 통합되어진다. DAL4WS는 비즈니스 프로세스를 수행에 필요한 공통 그룹과 개별적 행위로 분류되어 구성되어지고 이들은 특정 서비스 요청에 의해 BPEL4WS에 구체적인 파트너 정보를 동적으로 제공한다. 이러한 동적 통합 흐름은 그림 4와 같이 이루어진다.

그림 4에서 보는 것처럼 서비스 요청자는 특정 비즈니스 처리를 위한 서비스를 요청한다. 이때 시스템은 처리 서비스를 식별하고 해당 비즈니스의 BPEL4WS을 선택한다. 이때 이 BPEL4WS는



〈그림 4〉 DAL4WS을 이용한 동적 웹 서비스 통합 흐름

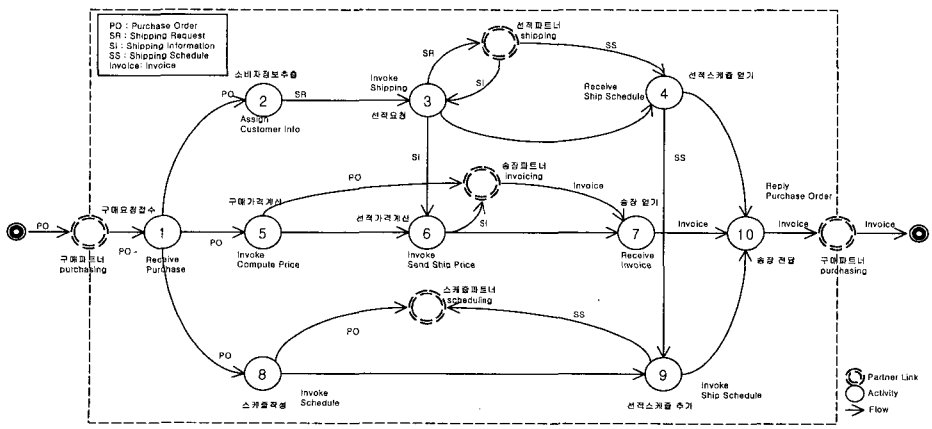
특정 파트너의 구체적인 정보를 포함하고 있지 않고 있다. 또한 시스템은 이 서비스 요청에 필요한 구체적인 파트너 정보를 DAL4WS을 통해 식별한다. 이들 파트너 정보는 4단계에서 통합되어 실행 가능한 비즈니스 프로세스(BPEL4WS)를 동적으로 생성한다. 이 BPEL4WS는 비즈니스 프로세스 처리에 필요한 구체적인 파트너 정보를 포함하는 프로세스이다. 마지막으로 5단계서 동적으로 생성된 비즈니스 프로세스는 파트너 정보와 협력하여 요청자에게 서비스를 제공한다. 이러한 관점은 DAL4WS을 통해 웹 서비스 통합 관점에서 비즈니스 프로세스의 재사용가능한 작업 흐름과 서비스 요소를 분리하고 관련 서비스의 그룹화를 통해 비즈니스 프로세스와 서비스 그룹의 재사용과 확장성을 제공할 것이다.

3.4 적용 예 : 구매처리 비즈니스 프로세스

구매처리 비즈니스 프로세스는 도서주문, 자동차주문 그리고 가전제품 주문 등의 프로세스로 구체화 될 수 있고 이들은 유사한 작업 흐름을 가진다고 가정한다. 이때 이들 공통의 비즈니스 프로세스는 다음과 같은 형태로 표현된다.

이 구매 프로세스는 자신의 비즈니스 프로세스에 대한 WSDL과 3개 파트너인 선적, 송장, 스케줄처리의 WSDL과 연결된다. 구매 요청을 받은 비즈니스 프로세스는 소비자 정보 추출과 구매가격 계산 스케줄 작업을 동시에 수행하며 각각의 파트너들과 협력하여 구매에 대한 송장을 제공하는 프로세스이다. 이 비즈니스 프로세스의 재사용과 확장성을 위해 파트너에 종속되는 행위정보와 데이터 부분을 제거한다. 제거 후 작성된 BPEL4WS용 비즈니스 프로세스는 그림 6에서 보여준다.

이 BPEL4WS는 파트너의 구체적인 WSDL 관련 정보를 제거하여 비즈니스 프로세스 활동을 기술하고 있다. 이 비즈니스 프로세스가 구체화되기 위해서는 연관된 파트너 정보와의 동적인 할당을 통해 이루어진다. 이제 도서, 자동차, 가전제품에 대한 주문을 고려해 보자. 이 들 3가지의 구체적 비즈니스 프로세스가 그림 6에서 제시한 구매처리 비즈니스 프로세스를 공유하기위해서는 이 정보가 유지 관리되어야 한다. 이것은 이전 절에서 제시한 DAL4WS을 이용하여 기술된다. 이 스키마는 파트너 정보를 기반으로 필요한 서브 프로세스를 그룹화하고 이들을 조합함으로써 구체화된 프로세스 정보를 관리하고 동적인 프로세스 할당 정책에 이용된다. 그림 7에서 구매처리 비즈니스에 대한 구체화 정보의 그룹 구조를 보여준다. 이 예에서 선적관련 프로세스는 3개의 구체적 비즈니스 프로세스에 공유되고 스케줄과 송장관련 프로세스는 비



〈그림 5〉 구매처리 비즈니스 프로세스

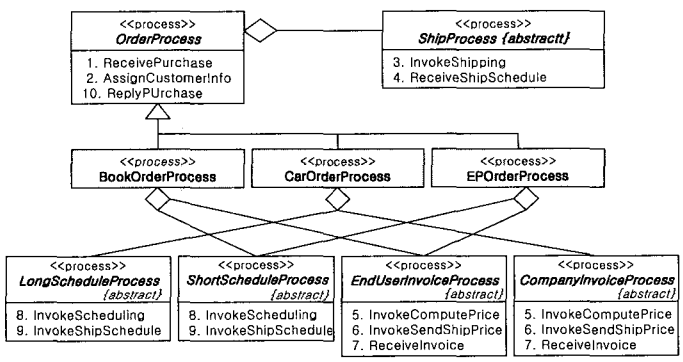
```
<?xml version="1.0" encoding="UTF-8" ?>
- <process name="OrderProcess" targetNamespace="http://docom.kw.ac.kr/services/bp/OrderProcess"
    xmlns="http://schema.xmlsoap.org/ws/2003/03/business-process/"
    xmlns:tns="http://docom.kw.ac.kr/services/bp/OrderProcess">
  - <partnerLinks>
      <partnerLink name="purchasing" />
      <partnerLink name="invoicing" />
      <partnerLink name="shipping" />
      <partnerLink name="scheduling" />
    </partnerLinks>
  - <variables>
      <variable name="PO" />
      <variable name="SR" />
      <variable name="SI" />
      <variable name="SS" />
      <variable name="Invoice" />
    </variables>
  - <sequence>
      <receive name="ReceivePruchase" partnerLink="purchasing" variable="PO" />
    - <flow>
      - <links>
          <link name="ship-to-invoice" />
          <link name="ship-to-scheduling" />
        </links>
      - <sequence>
        - <assign name="AssignCustomerInfo">
          - <copy>
              <from variable="PO" />
              <to variable="SR" />
            </copy>
          </assign>
        - <invoke name="InvokeShipping" partnerLink="shipping" inputVariable="SR" outputVariable="SI">
            <source linkName="ship-to-invoice" />
          </invoke>
        - <receive name="ReceiveShipSchedule" partnerLink="shipping" variable="SS">
            <source linkName="ship-to-scheduling" />
          </receive>
        </sequence>
      - <sequence>
          <invoke name="InvokeComputePrice" partnerLink="invoicing" inputVariable="PO" />
        - <invoke name="InvokeSendShipPrice" partnerLink="invoicing" inputVariable="SI">
            <target linkName="ship-to-invoice" />
          </invoke>
          <receive name="ReceiveInvoice" partnerLink="invoicing" variable="Invoice" />
        </sequence>
      - <sequence>
          <invoke name="InvokeSchedule" partnerLink="scheduling" inputVariable="PO" />
        - <invoke name="InvokeShipSchedule" partnerLink="scheduling" inputVariable="SS">
            <target linkName="ship-to-scheduling" />
          </invoke>
        </sequence>
      </flow>
      <reply name="ReplyPurchase" partnerLink="purchasing" variable="Invoice" />
    </sequence>
</process>
```

〈그림 6〉 구매처리에 대한 비즈니스 프로세스(BPEL4WS)

즈니스 특성을 고려하여 조합된다고 가정한다. 이것은 파트너 관련 활동(Activity)들이 목적에 의해 그룹화 되고 이들이 특정 목적의 비즈니스 프로세스에서 재사용된다는 것을 의미한다.

그림 7의 비즈니스 그룹과 파트너 관련정보를 DAL4WS을 표현하면 그림 8과 같다.

그림 8에서 도서주문 프로세스(BookOrderProcess)는 ShipProcess, EndUserInvoiceProcess 그리고 ShortScheduleProcess의 서브 프로세스를 포함하고 있으며 구체화될 비즈니스 프로세스에 관련된 정보를 제공한다. 다음 장에서 이들 비즈니스 프로세스 구체화에 필요한 프레임워크를 제시한다.



〈그림 7〉 구체화를 위한 비즈니스 프로세스 그룹

```
<?xml version="1.0" encoding="UTF-8" ?>
- <processContent name="bp:OrderProcess" xmlns="http://docom.kw.ac.kr/schemas/DAL4WSSchema" xmlns:bp="http://docom.kw.ac.kr/services/bp/OrderProcess">
  - <groups>
    - <group name="ShipProcess" abstract="yes">
      - <partnerLinks>
          <partnerLink name="bp:shipping" partnerLinkType="sns:shippingLT" myRole="shippingRequester" partnerRole="shippingService" />
        </partnerLinks>
      - <variables>
          <variable name="bp:SR" messageType="sns:ShippingRequestMessage" />
          <variable name="bp:SI" messageType="sns:ShippingInfoMessage" />
          <variable name="bp:SS" messageType="sns:ShippingScheduleMessage" />
        </variables>
        <correlationSets />
      - <activities>
          <activity name="bp:InvokeShipping" portType="sns:shippingPT" operation="requestShipping" />
          <activity name="bp:ReceiveShipSchedule" portType="ons:shippingCallbackPT" operation="sendSchedule" />
        </activities>
      </group>
    + <group name="EndUserInvoiceProcess" abstract="yes">
    + <group name="CompanyInvoiceProcess" abstract="yes">
    + <group name="ShortScheduleProcess" abstract="yes">
    + <group name="LongScheduleProcess" abstract="yes">
    - <group name="BookOrderProcess" abstract="no" xmlns:ons="http://docom.kw.ac.kr/services/wsdl/BookOrderService" xmlns:sns="http://docom.kw.ac.kr/services/wsdl/ShipService"
        xmlns:ins="http://docom.kw.ac.kr/services/wsdl/EndUserInvoiceService" xmlns:cns="http://docom.kw.ac.kr/services/wsdl/ShortScheduleService">
      - <composite>
          <group name="tns:ShipProcess" />
          <group name="tns:EndUserInvoiceProcess" />
          <group name="tns:ShortScheduleProcess" />
        </composite>
      - <partnerLinks>
          <partnerLink name="bp:purchasing" partnerLinkType="ons:BookOrderLT" myRole="pruchaseService" />
        </partnerLinks>
      - <variables>
          <variable name="bp:PO" messageType="ons:BookOrderMessage" />
        </variables>
        <correlationSets />
      - <activities>
          <activity name="bp:ReceivePruchase" portType="ons:purchaseBookOrderPT" operation="sendBookPurchaseOrder" />
        - <activity name="bp:AssingCustomerInfo">
            <variable name="bp:PO" part="customerInfo" />
            <variable name="bp:SR" part="customerInfo" />
          </activity>
          <activity name="bp:ReplyPurchase" portType="ons:purchaseBookOrderPT" operation="sendBookPurchaseOrder" />
        </activities>
      </group>
    + <group name="CarOrderProcess" abstract="no" xmlns:ons="http://docom.kw.ac.kr/services/wsdl/CarOrderService" xmlns:sns="http://docom.kw.ac.kr/services/wsdl/ShipService"
        xmlns:ins="http://docom.kw.ac.kr/services/wsdl/CompanyInvoiceService" xmlns:cns="http://docom.kw.ac.kr/services/wsdl/LongScheduleService">
    + <group name="ElectronicOrderProcess" abstract="no" xmlns:ons="http://docom.kw.ac.kr/services/wsdl/ElectronicOrderService"
        xmlns:sns="http://docom.kw.ac.kr/services/wsdl/ShipService" xmlns:ins="http://docom.kw.ac.kr/services/wsdl/EndUserInvoiceService"
        xmlns:cns="http://docom.kw.ac.kr/services/wsdl/LongScheduleService">
  </groups>
- <wsdlContents>
  - <wsdlContent group="tns:BookOrderProcess">
      <wsdl name="ons" namespace="http://docom.kw.ac.kr/services/wsdl/BookOrderService" location="wsdl/BookOrderService.wsdl" />
      <wsdl name="sns" namespace="http://docom.kw.ac.kr/services/wsdl/ShipService" location="wsdl/ShipService.wsdl" />
      <wsdl name="ins" namespace="http://docom.kw.ac.kr/services/wsdl/EndUserInvoiceService" location="wsdl/EndUserInvoiceService.wsdl" />
      <wsdl name="cns" namespace="http://docom.kw.ac.kr/services/wsdl/ShortScheduleService" location="wsdl/ShortScheduleService.wsdl" />
    </wsdlContent>
  + <wsdlContent group="tns:CarOrderProcess">
  + <wsdlContent group="tns:ElectronicOrderProcess">
  </wsdlContents>
</processContent>
```

〈그림 8〉 비즈니스 프로세스 구체화 정보 (DAL4WS)

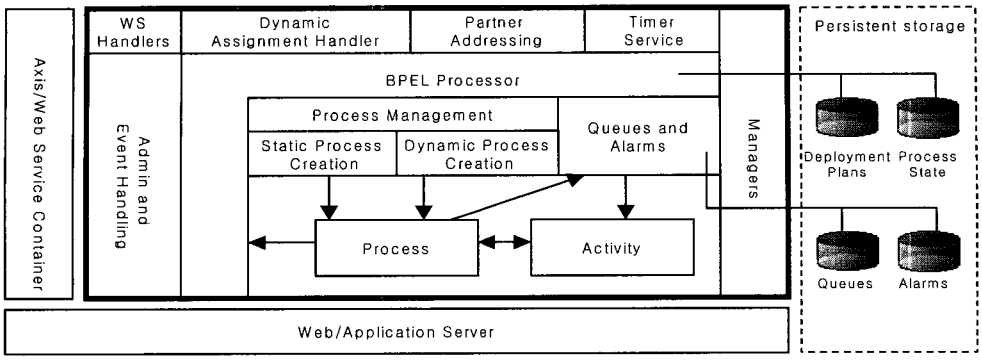
4. 비즈니스 프로세스의 동적할당을 위한 프레임 워크 구조

이장에서는 위에서 기술한 비즈니스 프로세스 정보를 이용하여 동적으로 비즈니스 프로세스를 구체화 하기위한 프레임워크 구조를 제시한다. 본 논문에서는 이러한 환경을 제공하기위해 Active Endpoints 사에서 오픈 소스 프로젝트로 진행하고 있는 ActiveBPEL[1] 엔진을 확장하여 구성한다. 이 엔진은 BPEL4WS 1.1 명세에 의해 생성된 비즈니스 프로세스를 처리할 수 있는 수행 가능한 환경으로 Java을 통해 구현하였다. 아래의 그림 9는 비즈니스 프로세스 동적할당 정책을 포함하는 BPEL4WS 엔진 프레임워크를 보여준다.

이 엔진 프레임 워크는 BPEL4WS와 DAL4-WS 정보를 유지하고 사용자의 요청에 대한 동적 할당 처리를 담당하는 동적할당 핸들러(Dynamic Assignment Handler)를 포함하고 있다. 이 핸들러는 파트너의 구체화 정보를 모두 포함하는 BPEL4WS문서를 다루기 위해 정적 프로세스 생성 모듈(Static Process Creation)을 이용하여 처리하고, 동적 비즈니스 할당과 처리를 위해 동적 프로세스 생성 모듈(Dynamic Process Creation)을 이용한다. 이를 통해 비즈니스 프로세스의 재사용과 동적인 사용을 가능하게 한다.

5. 결론 및 향후 과제

인터넷과 웹 관련 서비스의 대중화는 기업 비즈니스 환경의 급속한 변화를 이끌고 있다. 이것



〈그림 9〉 비즈니스 프로세스 동적 할당을 제공하는 BPEL4WS 엔진 프레임워크

은 데이터/객체 중심의 처리에서 서비스 중심의 변화이며 이 서비스는 SOA환경을 이끄는 중요한 요소로써 인식되고 있다. 그 결과 이들 서비스들의 효율적인 통합 문제가 크게 부각되고 있다. 서비스 통합은 상호운영성 문제 이외에도 다양한 관점에 대한 고려가 필요하다. 본 논문에서는 BPEL4WS 환경에서 서비스 통합의 효율성을 증가시키기 위해 비즈니스 프로세스의 동적 활용 기법을 제시하였다. 이것은 비즈니스 프로세스의 재사용과 확장성을 증가시키기 위해 공통 프로세스와 파트너 의존 정보의 분할을 통해 이루어지며 특정 서비스 요청 시 동적으로 합병되어 수행된다. 이를 위해 먼저 비즈니스 프로세스의 의존성 관계를 분석하고 이를 기반으로 웹 서비스의 동적 할당 언어(DAL4WS)를 제시하였다. 또한 BPEL4WS 환경에서 비즈니스 프로세스의 동적 수행을 제공하는 프레임워크를 보였다. 현재 공통 프로세스와 특정 서비스에 의존적인 정보가 동적으로 결합되어 수행되는 프레임워크는 구현되지 않고 있으며 본 논문에서는 ActiveBPEL 엔진[1] 모듈에 동적 메커니즘을 포함하여 프레임 워크를 설계하였고 이를 4장에서 보였다. 이러한 비즈니스 프로세스의 동적 재사용과 활용을 위해 추가적으로 비즈니스 프로세스 도메인 영역에 따른 공통 요소 인식과 분류 체계 식별에 대한 고려가 필요하고 웹 서비스의 특성을 인식하고 판단할 수 있는 방법이 요구 된다.

향후 연구 내용으로는 위에서 제시한 공통요소 인식과 분류 체계방법과 동적으로 생성되는 비즈니스 프로세스의 효율적인 관리를 위한 동적 비즈니스 프로세스의 모니터링 및 테스팅 기법이 요구된다.

참 고 문 헌

[1] ActiveBPEL "The Open Source BPEL Engine", www.activebpel.org, 2004 http://www.activebpel.org/info-/intro.html

[2] B. Atkinson, G. Della-Libera, S. Hada, M. Hondo, P. HallamBaker, J. Klein, B. LaMacchia, P.Leach, J. Manferdellie, H. Maruyama, A. Nadalin, N. Nagaratnam, H. Prafullchandra, J. Shewchuk, and D. Simon. "Web Services Security" IBM Corp., Microsoft Corp., VeriSign, Inc., 2002

[3] B. Orriens, Jian Yang, M.P. Papazoglou "ServIceCom: a tool for service composition reuse and specialization" Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on, 10-12 Dec. 2003

[4] BPEL4WS, "Specification:Business Process Execution Language for Web Service Version 1.1" 05 May 2003, http://www106.

ibm.com/developerworks/library/ws-bpel/

[5] BPML, "Business Process Modeling Language", Accessed November 2003 from www.bpmi.org, 2003

[6] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana. "Unraveling the Web Services Web:An Introduction to SOAP, WSDL, and UDDI." IEEE Internet Computing, March 2002

[7] J. Mendling, M. Strembeck, G. Stermsek, and G. Neumann,"An Approach to Extract RBAC Models from BPEL4WS Processes", the Second InternatIonal Workshop on Distributed and Mobile Collaboration, July 2004,

[8] Kuo-Ming Chao, M. Younas, N. Griffiths, I. Awan, R. Anane, C-F Tsai, "Analysis of grid service composition with BPEL4WS", Advanced Information Networking and Applications, 2004. AINA 2004. 18th International Conference on, Volume 1, 29-31 March 2004

[9] M. Paolucci, T. Kawmura, T. Payne, and K. Sycara. "Semantic Matching of Web Services Capabilities", In The First International Semantic Web Conference, 2002

[10] P. Wohed, W.M.P. van der Aalst, M. Dumas, and A.H.M. ter Hofstede. "PatternBased Analysis of BPEL4WS", QUT Technical report, FIT-TR-2002-04, Queensland University of Technology, Brisbane, 2002

[11] U1N/CEFACT and OASIS, "ebxml business process specification schema" Accessed November 2002 form www.ebxml.org/specs/ebBPSS.pdf, 2001

[12] WSCI, "Web Service Choreography Interface", Tech Report by Italio, SAP, BEA, Sun Microsystems. 2003

[13] WS-I Basic Profile "Web Service Interoperability Basic Profile Version1.0 Final" 16 April 2004, http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html

[14] WSFL, "Web Service Flow Language (WSFL 1.0)", http://www-306.ibm.com/software/solutions/webservi-ces/pdf/WSFL.pdf

[15] XLANG, "Web Services for Business Process Design", http://www.ebpml.org/xlang.htm

◐ 저 자 소 개 ◑

김 운 용(Woon-Yong Kim)

1996년 광운대학교 독학사 컴퓨터과학과 졸업(학사)
1999년 광운대학교 정보통신대학원 전자계산학과 졸업(석사)
2003년 광운대학교 대학원 컴퓨터과학과 졸업(박사)
2003년~2004년 광운대학교 정보통신연구원
2004년~현재 MAXSoft Co.,LTD. 기술연구소 책임 연구원
관심분야 : 분산컴퓨팅, 웹 서비스, 객체지향프로그래밍, 디자인 패턴
E-mail : wykim@kw.ac.kr