웹 서비스 기반의 협상 시스템을 위한 프레임워크의 설계

이종혁*, 국윤규*, 김운용*, 최영근*
*광운대학교 컴퓨터과학과
e-mail:shineljh@kw.ac.kr

Design of Framework for Negotiation System based on Web Service

Jong-Hyuk Lee*, Youn-Gyu Kook*, Woon-Yong Kim*,
Young-Geun Choi*
*Dept of Computer Science, Kwangwoon University

요 약

협상은 일상생활, 기업 및 정부 비즈니스 활동에 널리 사용되며 점차 그 영역을 넓혀가고 있다. 기존의 협상 시스템들은 독립적인 애플리케이션 형태로 구축이 되어 있어, 협상 프로세스를 기업 내 또는 기업간의 다른 프로세스들과 연동하여 사용하기 힘들다. 또한 구매 또는 판매를 담당하는 에이전트나 애플리케이션 작성을 특정 플랫폼에서만 작성해야 한다. 본 논문에서는 이러한 단점을 보완하기 위해 웹 서비스 기반으로 협상 프레임워크를 설계하여, SCM 등과 같은 더 큰 프로세스와 쉽게 연동이 될 수 있도록 하고 에이전트나 애플리케이션 개발 자체도 특정한 언어나 플랫폼에 구애받지 않게 작성할 수 있도록 하였다.

1. 서론

협상은 일상생활, 기업 및 정부 비즈니스 활동에서 널리 사용되고 있으며, 점차 그 영역을 넓혀가고 있다. 협상은 크게 입찰, 경매, 흥정의 세 가지 형태이다. 입찰은 가장 단순한 형태로 구매자가 원하는 물품 또는 서비스를 요청하면 다수의 판매자들이 좋은 조건으로 경쟁하게 되고 구매자는 가장 좋은 조건을 제시한 판매자를 선택하는 형태이다. 반대로 경매는 판매자가 물품 또는 서비스를 올려놓을 때, 다수의 구매자들이 경쟁하여 판매자가 가장 좋은 조건을 제시한 구매자에게 판매하는 형태이다. 끝으로 흥정은 가장 복잡한 형태로 협상 참여자들이 서로의 합의를 찾을 때까지 제안과 반대 제안을 주고받는 형태이며, 본 논문에서는 흥정 형태에 초점을 맞추었다[1][2][3].

기존의 협상 시스템[1][2][3][4]에서는 독립적인 애플리케이션 형태로 구축 되어 있어, 협상 프로세스를 기업 내 또는 기업간의 다른 프로세스들과 연동하여 사용하기 힘들다. 또한 구매 또는 판매를 담당하는 에이전트나 애플리케이션 작성을 특정 플랫폼에서만 작성해야 한다.

본 논문에서는 이러한 문제를 해결하기 위해 웹 서비스를 기반으로 하는 협상 시스템을 위한 프레임워크를 제안한다. 웹 서비스는 상호운용성이 우수하여 용이한 시스템 통합이 가능하도록 해주며,, 플랫폼에 상관없이 서비스를 개발할 수 있는 특징을 가진다[5]. 따라서 제안된 프레임워크는 SCM(Supply Chain Management) 등과 같은 더 큰 프로세스와 쉽게 연동이 될 수 있도록 하고 에이전트나 애플리케이션 개발 자체도 특정한 언어나 플랫폼에 구애받지 않고 작성할 수 있도록 하였다.

본 프레임워크에서는 구매자와 판매자 서비스 개발을 위한 WSDL 템플릿을 제공하여, 구매자와 판매자가 에이전트나 애플리케이션을 작성할 때 플랫폼에 영향 받지 않고 작성할 수 있도록 하였고 협상 프로세스 자체를 BPEL[6] 기반의 웹 서비스로 작성하여 기업 내 또는 기업간 시스템과의 연동을 용이하게 하였다. 또한 기존의 BPEL 모니터링 시스템

[7]를 통해 협상 과정을 쉽게 파악할 수 있도록 한다.

2장에서는 웹 서비스와 BPEL의 특성을 살펴보고, 3장에서는 본 논문에서 제안하는 프레임워크를 소개한다. 끝으로 4장에서는 결론 및 향후 연구 과제를 보인다.

2. 관련연구

웹 서비스는 XML 기반의 표준들이 기반이 되어 플랫폼에 독립적으로 인터넷상의 분산 애플리케이션들을 통합할 수 있도록 해 준다. 웹 서비스는 WSDL을 사용하여 자신의 서비스를 표현하고, 표준 인터넷 전송 프로토콜을 사용하여 SOAP 메시지를 다른 시스템들과 교환하게 된다. 따라서 어떠한 프로그램 언어로 구현할 수 있고, 이기종의 시스템 간 통합이 용이하다[5].

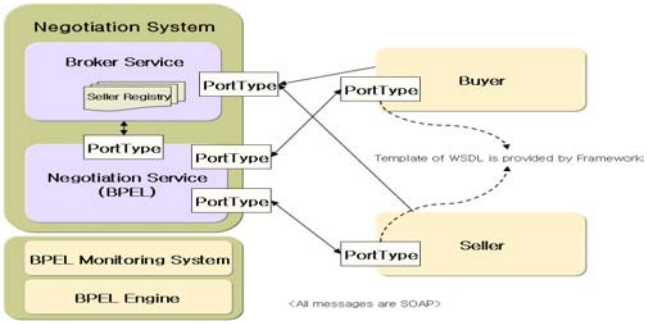
BPEL4WS(Business Process Execution Language for Web Service, shortly BPEL)은 비즈니스 프로세스의 상호 관계를 기술하는 XML 기반의 언어로서, 기업 내부뿐만 아니라 기업간 상 호 업무 흐름을 기술한다. BPEL을 사용함으로서, 사내 다른 조직이나 파트너 회사간의 메시지 교환을 용이하게 할 수 있게 되고, 동일한 비즈니스 프로세스를 상호 운용할 수 있게 된다. 또한 BPEL로 작성된 프로세스도 하나의 WSDL을 갖는 독립적인 웹 서비스이기 때문에 또 다른 시스템에 쉽게 운용할 수 있게 된다[6].

본 논문에서는 WSDL 템플릿을 구매자 및 판매자에게 제공하여 구매자와 판매자가 플랫폼에 독립적인 환경을 제공하고, BPEL로 협상을 진행하여 확장성을 높인다.

3. 협상 프레임워크

3. 1. 개요

협상 시스템은 (그림 1)과 같이 구성되며, [7]의 BPEL 모니터링 시스템 기반으로 동작하여, 협상 진행 상황을 실시간으로 파악할 수 있도록 한다.. BPEL 모니터링 시스템은 수행중인 BPEL 프로세스의 진행 상황 및 파트너의 상태를 파악할 수 있도록 한 시스템으로서 협상 시스템과 연동하여 BPEL 프로세스로 진행되는 협상 프로세스를 모니터링 할 수 있도록 한다[7].

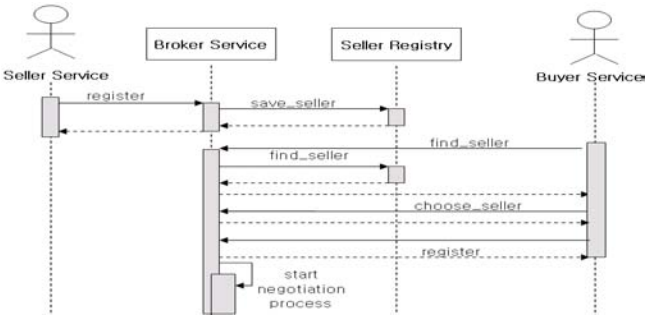


(그림 1) 협상 시스템의 구조

프레임워크는 협상 시스템을 위해 Broker Service, Negotiation Service, WSDL Template을 제공한다. 따라서 협상 시스템 구현을 하기 위해 물품이나 서비스에 대한 도메인을 정의하고 그에 따른 판매자 및 구매자를 제공된 WSDL Template에 맞추어 생성하면 된다.

Broker Service는 판매자와 구매자를 요청을 받아 협상을 시작하게 해주는 역할을 담당하고, Seller Registry로 판매자들의 목록을 관리하며 구매자는 Broker Service를 통해 협상을 시작하게 된다. 협상이 시작되면 협상은 BPEL 기반의 Negotiation Service에 의해 수행된다. 협상에 필요한 메시지들은 미리 정의가 되어 있으며 이를 위해 구매자와 판매자에게 WSDL 템플릿을 제공한다. 따라서 구매자와 판매자는 WSDL 템플릿을 기반으로 웹 서비스를 생성하게 되며 자세한 내용은 3.2절에서 다룬다.

판매자는 자신의 서비스를 Negotiation Service에 자신의 정보와 함께 등록하면 Negotiation Service는 Seller Register에 판매자를 등록한다. 구매자는 Negotiation Service에 요청하여 판매자 목록을 얻어온 후, 적절한 판매자와 협상을 시작한다. (그림 2)는 협상 프로세스가 생성되기 전까지의 과정을 보인 것이며, 협상 프로세스에 대한 세부적인 내용은 3.3절에서 다룬다.



(그림 2) Prenegotiation

3. 2. WSDL 템플릿

WSDL 템플릿은 BPEL로 작성된 협상 프로세스

에 필요한 판매자와 구매자의 인터페이스 등이 포함된다 하지만 협상 물품 또는 서비스가 협상 도메인마다 다르고 서비스의 물리적인 위치도 다르기 때문에 물품 또는 서비스 정보와 물리적인 위치를 기술하는 부분은 사용자가 작성하게 된다.

판매자와 구매자는 프레임워크에서 제공하는 WSDL 템플릿을 받아 협상 도메인 정보와 물리적인 위치를 추가해 WSDL을 완성한 후, WSDL 기반으로 웹 서비스를 작성한다. 따라서 판매자와 구매자는 언어 및 플랫폼에 독립적으로 작성된다.

WSDL의 구조는 크게 (표 1)의 엘리먼트들로 이루어져 있다. 하지만, 판매자 및 구매자는 협상에 고려될 속성들은 협상 도메인마다 달라지기 때문에, <types> 엘리먼트는 도메인 정의자에 의해 완성된다. 또한 <service> 엘리먼트는 서비스들의 물리적인 위치에 따라 달라지기 때문에 실제 서비스 구현 측에서 작성하게 된다. <message>, <portType>, <binging> 엘리먼트는 WSDL 템플릿에서 제공하며, 협상에 필요한 프로토콜을 위한 프로시저들이 들어간다.

Elements	Description
<types>	WSDL 문서 내에서 사용할 타입을 기술하며, 협상 도메인마다 달라짐 (템플릿에서 제공하지 않음)
<message>	협상에 필요한 input, output 메시지를 기술
<portType>	협상에 필요한 프로시저들이 들어감
<binding>	호출에 사용되는 프로토콜을 기술
<service>	웹 서비스 시스템의 endpoint 정보를 기술하며, 제공하는 서비스마다 달라짐 (템플릿에서 제공하지 않음)

(표 1) WSDL 템플릿의 구조

(그림 3)은 완성된 WSDL 문서이다. 점선으로 표시된 부분은 WSDL 템플릿에서 제공하지 않는 부분며, <types>과 <service> 엘리먼트는 각각 도메인 정의자와 구매자 서비스 개발자가 작성한 것이다.

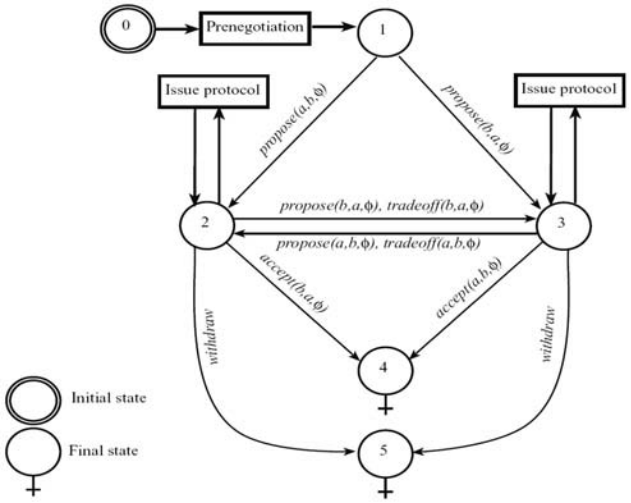
```
- <wsdl:types>
  - <schema targetNamespace="pllab.kw.ac.kr" xmlns="http://www.w3.org/2001/XMLSchema">
      <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
    - <complexType name="Product">
      - <sequence>
          <element name="price" type="xsd:int" />
          <element name="state" nillable="true" type="xsd:string" />
        </sequence>                                          도메인 정의
      </complexType>
    </schema>
  </wsdl:types>
+ <wsdl:message name="acceptRequest">
  <wsdl:message name="proposeResponse" />
+ <wsdl:message name="withdrawRequest">
+ <wsdl:message name="withdrawResponse">
- <wsdl:message name="proposeRequest">
    <wsdl:part name="in0" type="tns1:Product" />
  </wsdl:message>
+ <wsdl:message name="acceptResponse">
- <wsdl:portType name="BuyerService">
  + <wsdl:operation name="accept" parameterOrder="in0">
  + <wsdl:operation name="propose" parameterOrder="in0">
  + <wsdl:operation name="withdraw" parameterOrder="in0">
  </wsdl:portType>
+ <wsdl:binding name="BuyerServiceSoapBinding" type="impl:BuyerService">
- <wsdl:service name="BuyerServiceService">
  - <wsdl:port binding="impl:BuyerServiceSoapBinding" name="BuyerService">
      <wsdlsoap:address location="http://docom5.kw.ac.kr:8080/axis/services/BuyerService" />
    </wsdl:port>                                             서비스의 물리적 위치
  </wsdl:service>
```

(그림 3) 구매자의 WSDL 문서

3. 3. Negotiation Process

판매자와 구매자는 Negotiation Service에 의해 간접적으로 통신하게 되며, Negotiation Service는 BPEL 기반의 웹 서비스로 구성된다. 제안된 프레임워크는 모두 웹 서비스로 작성되기 때문에 다른 시스템과 쉽게 상호 운용할 수 있게 된다.

협상 과정은 [2]에서 제시된 프로토콜을 따르게 되며 (그림 5)와 같이 이루어진다. Prenegotiation 과정을 거쳐 판매자와 구매자가 협상을 하게 되면, accept나 withdraw가 발생할 때까지 계속 propose를 주고받는 형태이다.



(그림 4) 협상 프로토콜의 구조

협상이 진행되면 진행 상황에 대해 파악할 수 있어야 하는데 본 프레임워크에서는 [7]의 BPEL 모니터링 시스템과 연동하여 협상 진행 상황을 파악할 수 있도록 한다.

```
<process name="NegotiationService" ...>
<partnerLinks>
  <partnerLink name="broker"
partnerLinkType="tns:NegotiationService"
myRole="NegotiationServiceProvider" partnerRole="BrokerService"/>
  <partnerLink name="BuyerService"
partnerLinkType="BuyerService:BuyerServiceLink"
myRole="NegotiationService" partnerRole="BuyerServiceProvider" />
  <partnerLink name="SellerService"
partnerLinkType="SellerService:SellerServiceLink"
myRole="NegotiationService"partnerRole="SellerServiceProvider" />
</partnerLinks>
<variables>
<!-- List of variables used within this Negotiation process  -->
</variables>
<sequence name="main">
<!-- Due to the limited space, we take off <assign> element  -->
<receive name="receiveInput" partnerLink="broker" .../>
<invoke name="askOffer" partner="seller" outputVariable="product" ...
/>
<invoke name="proposeToBuyer" partner= "buyer"...>
<receive name= "responseFromBuyer"partner="buyer" ...>
<switch>
  <case
condition="bpws:getContainerProperty('responseFromBuyer',
'ns:isComplete')=true" >
    <invoke name= "completeMessageToSeller" partner="seller" .../>
```

```
  </case>
  <otherwise>
    <while condition
="bpws:getContainerProperty('responseFromBuyer',
'ns:isComplete')=false and
bpws:getContainerProperty('responseFromSeller',
'ns:isComplete')=false" >
      <invoke name="proposeToSeller" partner= "seller"...>
      <receive name= "responseFromSeller"partner="seller" ...>
      <switch>
        <case condition=
"bpws:getContainerProperty('responseFromSeller',
'ns:isComplete')=true">
          <invoke name= "completeMessageToBuyer" partner=
"buyer" />
        </case>
        <otherwise>
          <invoke name="proposeToBuyer" partner= "buyer"...>
          <receive name= "responseFromBuyer"partner="buyer" ...>

          <switch>
            <case
condition="bpws:getContainerProperty('responseFromBuyer',
'ns:isComplete')=true" >
              <invoke name= "completeMessageToSeller"
partner="seller" .../>
            </case>
          </switch>
        </otherwise>
      </switch>
    </while>
  </otherwise>
</switch>
<invoke name="callbackBroker" partnerLink="broker"
inputVariable="returnBroker"/>
</sequence>
</process>
```

(그림 5) 협상 프로세스

(그림 5)는 협상 프로세스를 나타내는 BPEL 문서이다. 지면상 일부만 보이도록 한다. 처음 판매자에게 초기 조건을 얻어와(동기) 구매자에게 보내고(비동기) 구매자는 승인, 거부, 다른 제안을 하게 된다. 승인이나 거부를 하게 되면 프로세스는 BrokerService에게 알리고 종료를 하게 되며, 이 때 승인을 하게 되면 BrokerService는 SellerRepository에서 판매자를 없애게 된다. 구매자가 다른 제안을 하게 되면 판매자는 이 제안을 받아(비동기) 승인, 거부, 다른 제안을 하게 되고, 둘 중 하나가 승인이나 거부를 할 때까지 서로 제안을 교환받게 되는 형태이다. 이 때 처음 초기 조건을 얻어오는 메시지와 승인과 거부 메시지는 즉시 처리될 수 때문에 동기적인 방식으로 처리하고, 제안 메시지를 처리하는 부분은 비동기적으로 처리한다.

4. 결론 및 향후 연구 과제

본 논문에서는 협상 시스템을 위한 웹 서비스 기반의 프레임워크를 제안하고, WSDL 템플릿과 BPEL 기반의 협상 서비스에 대해 기술하였다.

WSDL 템플릿을 통해 구매자 및 판매자는 플랫폼에 영향 받지 않고 애플리케이션이나 에이전트를 개발하게 된다. 또한 기존의 BPEL 모니터링 시스템과 쉽게 연동할 수 있도록 하여 협상 진행 과정을 쉽게 파악할 수 있도록 하였고, 협상 서비스 자체도 BPEL 기반의 웹 서비스로 제공하여 다른 시스템에 쉽게 상호 운용할 수 있도록 하였다.

향후 프레임워크의 구현과 자동화된 협상 시스템을 위한 구매자 서비스와 판매자 서비스에 대한 에이전트에 대한 연구가 이루어져야 할 것이다.

참고문헌

[1] Haifei Li; Jun-Jang Jeng; Jen-Yao Chung, “Commitment-based approach to categorizing, organizing and executing negotiation processes”, E-Commerce 2003 IEEE p12-15, 2003. 6

[2] Faratin, P., Sierra, C., Jenninings, N.R. and Buckle, P., "Designing Responsive and Deliberative Automated Negotiators", Proc. AAAI Wrokshop on Negotiation, Orlando, FL, pp. 12-18, 1999.

[3] Stanley Y. W. Su, Chunbo Huang, Joachim Hammer, "A Replicable Web-Based Negotiation Service For E-Commerce", 33rd Hawaii International Conference on System Sciences, 2000

[4] Chavez, A., and Maes, P., "Kasbah : An Agent Marketplace for Buying and Selling Goods", 1st international Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, 1996

[5] Web Services. http://www.w3.org/2002/ws

[6] BPEL4WS, http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/

[7] 이종혁, 국윤규, 김운용, 최영근, “BPEL4WS 기반의 비즈니스 프로세스 관리를 위한 통합 모니터링 시스템”, 정보과학회, 2004. 10