업무규칙을 포함한 능동문서

남철기* 배재학**
*한국 오라클(주), **울산대학교 컴퓨터·정보통신공학부
cholki.nam@oracle.com, jhjbae@ulsan.ac.kr

Active Documents with Business Rules

Chul-Ki Nam* Jae-Hak J. Bae**
*Oracle Korea,
**School of Computer Engineering and Information Technology
University of Ulsan

요 약

업무규칙은 선언적인 방법으로 업무정책을 표현하고 업무조항을 정의한다. 또한 업무프로세스를 정의하고 프로세스가 수행되는 과정에 제약을 가한다. 본 논문에서는 능동문서 처리 프레임워크를 구현하였다. 즉, 사무문서에 함축되어 있는 업무규칙을 어플리케이션 로직과 분리하여 생각하였고, 업무규칙은 Prolog로 명시적으로 표현하였다. 이 Prolog 규칙은 논리프로그램 추론엔진에서 수행가능하고, 이종의 규칙기반 어플리케이션에서 사용할 수 있도록 XML 문서로 변환하였다. 이렇게 문서의 데이터와 규칙을 저장한 XML 문서는 규칙기반의 시스템과 워크플로우 관리시스템(WFMS) 환경에서 처리된다.

1. 서 론

과거의 어플리케이션은 업무 요구사항을 충족시키기가 어려웠고 유연하지 못했다. 요구사항을 만족시키기 위해 어플리케이션을 수정하는 것은 많은 비용과 시간을 소모했다. 오늘날의 어플리케이션은 급변하는 업무 요구사항을 만족시키기 위해 유연해야 한다. 이러한 요구사항은 업무 전문가가 업무 규칙을 정의하고 수정함으로써 해결될 수 있다. 선언적으로 정의된 업무 규칙은 프로그래밍 언어로 복잡한 구현 없이 즉시 수행가능 해야 한다.

업무 규칙에 대한 여러 정의가 있지만, 일반적으로 다음과 같이 정의한다. "업무가 성취되는 방법이며, 조직 내에서의 업무의 상태와 프로세스에 대한 제약(restriction)과 지침(guideline)이다[3, 5]." 이러한 규칙은 업무의 함축된 의미(semantics)를 캡슐화하고, 데이터베이스가 데이터와 어플리케이션을 분리시켰듯이 업무 규칙을 어플리케이션 로직으로부터 분리시킬 수 있다.

XML(eXtensible Markup Language)은 과거 전자문서의 논리적 정보와 물리적 정보가 같이 표현되던 구조에서 문서의 논리적 구조(XML), 물리적 구조(XSL), 문서의 연결(Xlink, Xpointer)을 분리하려는 요구와 시도에 의해 생겨나게 되었다. XML은 이상적인 전자문서의 요건을 갖추고 있으며 현재 모든 형태의 데이터와 문서를 통합, 저장, 처리할 수 있는 프레임워크[1]를 제공한다.

본 논문에서는 XML로 업무 문서의 논리적 구조를 정의하고 문서에 함축되어진 업무 규칙과 문서작성 규칙을 명시적으로 표현하였다. 즉, 문서의 각 항목간의 상호의존관계, 무결성 제약조건, 업무 프로세스를 규칙표현에 효과적이고 논리프로그래밍 언어인 Prolog로 선언적으로 정의하였다. 그리고, 업무 규칙을 변환기(Prolog2XML)를 이용하여 수행가능하고 이종의 규칙기반 시스템과 규칙을 공유하기 위해 ERML(Executable Rule Markup Language)로 변환하였다[2]. 변환된 ERML은 변환기(XML2Prolog, XML2Workflow)를 각각 이용하여 논리프로그램 추론엔진과 워크플로우 관리시스템(WFMS) 환경에서 처리된다.

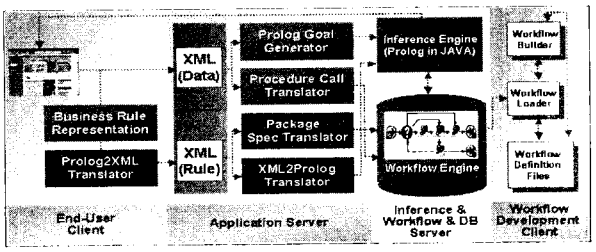
본 논문과 관련된 대표적인 연구는 다음과 같다: (1) RuleML[9]은 웹상의 지식(Knowledge on the web)을 규칙으로 정의 하며, (2) IBM의 *CommonRules*[4]는 CLP(Courteous Logic Program)를 규칙 표현의 언어로 사용하였으며, 규칙 기반의 어플리케이션을 개발하기 위한 규칙기반의 프레임워크이다. 본 논문에서는 CommonRules와는 다르게 1) 규칙표현 언어로 Prolog를 사용하였으며 2) 업무문서에 데이터와 규칙을 함께 포함시켜 처리 하였다. 또한, 3) 업무 프로세스를 자동화 하기 위해 상용 워크플로우 시스템과 연동하였다.

2. 능동문서 처리 프레임워크

본 장에서는 능동문서(active documents) 처리를 위한 프레임워크[7]의 구성요소와 역할에 대해 기술한다. 사무문서가 함축하고 있는 묵시적 또는 명시적인 업무 규칙을 Prolog 규칙으로 표현하여 웹환경에서 논리프로그램 추론엔진과 워크플로우 시스템에서 문서를 처리한다. 문서내의 상호관계와 제약조건, 다른 규칙으로부터 유도된 규칙은 Prolog 추론엔진에서 처리한다. 그리고 업무 프로세스는 워크플로우 시스템에서 업무 규칙의 제어를 통해 문서를 처리한다.

2.1 시스템 구조

아래 [그림 1]은 문서처리 프레임워크의 전체 구조를 나타내며 크게 4계층으로 나누어진다.



[그림 1] 능동문서 처리 프레임워크

첫 번째 계층은 업무 전문가가 문서를 생성하고 문서에 함축되어 있는 업무 규칙을 Prolog 규칙으로 표현하는 부분이고, 두 번째는 사용자가 문서(폼)에 입력한 내용을 변환기(Prolog2XML)를 통해 규칙과 데이터를 포함하는 XML 문서로 생성한다. 그리고 문서의 업무 규칙을 검증하고 업무 프로세스를 수행하기 위해 변환기들을 통해 변환하는 부분이다. 세 번째는 생성된 Prolog Goal을 실행/추론엔진에서 실행하고, 또한, 워크플로우를 생성하고 시작하는 부분이다. 마지막으로 네 번째 계층은 오라클 워크플로우 빌더로 업무 프로세스를 만들고 워크플로우 엔진으로 업무 프로세스를 로드하는 부분이다.

2.2 구성요소

(1) 변환기 및 생성기

1) Prolog2XML Translator: 업무 규칙을 표현한 Prolog 규칙을 ERML로 변환한다.

2) XML2Prolog Translator: Prolog2XML변환기로 생성된 ERML로부터 추론엔진에서 사용하기 위한 원래의 Prolog 규칙으로 변환한다.

3) Prolog Goal Generator: 데이터가 저장되어 있는 XML문서로부터 실행/추론 엔진에서 수행할 Goal을 생성한다.

4) Procedure Call Translator: 데이터가 저장되어 있는 XML문서로부터 워크플로우 엔진API를 이용해 워크플로우를 생성하고 시작하기 위한 프로시저를 호출한다.

(2) 엔진

1) Inference Engine: SWI-Prolog[10]를 사용한다.

2) Workflow Engine: 오라클 워크플로우 시스템[8]을 사용하며 업무 프로세스를 수행하고 자동화하며 사무 문서를 승인 또는 거절한다.

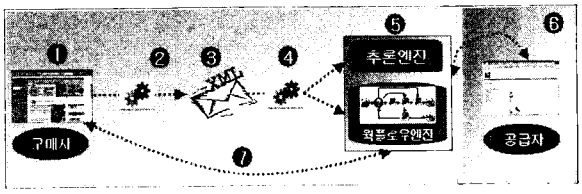
3. 실 험

3.1 실험 도구

능동문서 처리시스템을 구현하기 위해 (1)Oracle9i Application Server 1.0.2.2 (2)Oracle8i Enterprise Edition Release3 (3)Oracle Workflow Server Release 2.6 (4)Oracle Workflow Builder 2.6 (5)SWI-Prolog 4.0.9 (6)JPL(Java Interface to Prolog)[6]을 사용하였다.

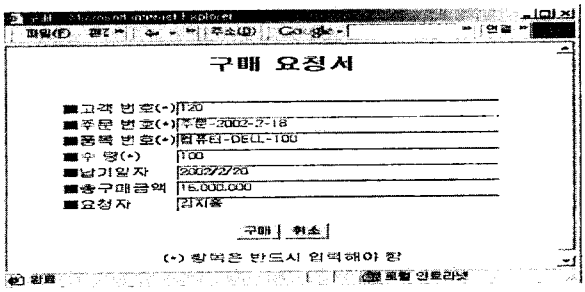
3.2 실험 시나리오 및 결과

능동문서의 처리를 실험하기 위해 [그림 2]와 같은 간결한 시나리오를 작성하였다.



[그림 2] 실험 시나리오

①구매자가 PC 100대를 구매하기 위해 구매 B2B 사이트에 접속한후 [그림 3]같은 구매요청서에 내역을 기록한다.



[그림 3] 구매 요청서(능동문서의 예)

구매요청서는 [표 1]과 같이 구매요청과 관련된 업무 규칙을 함축하고 있다. [표 1]에 의해 고객은 우수고객이며 15%의 할인을 받고 납기일이 주문날짜로부터 10일이 된다.

[표 1] 구매 업무 규칙

(규칙 1)	고객번호가 100이상 200이하이면 우수고객이다.
(규칙 2)	우수고객이면서 총 구매금액이 1000만원 이상이면 15%를 할인해 준다.
(규칙 3)	우수고객이면 구매 금액의 10%를 할인해준다.
(규칙 4)	모든 구매주문은 반드시 공급업체 승인자의 결재(승인,거절)가 필요하다.
(규칙 5)	우수고객 또는 구매금액이 1000만원 이상이면 납기일이 10일이고 구매금액이 1000이하이면 납기일이 20일이다.
(규칙 6)	(*) 항목은 반드시 입력하여야 한다.

(규칙 4)는 결재가 필요한 부분이므로 워크플로우와 연동되는 규칙이며, 나머지 규칙은 Prolog 인터프리터에서 추론 가능한 규칙들이다. [표 1]의 (규칙 2)와 (규칙 4)를 Prolog 규칙으로 표현하면 다음과 같다.

(규칙 2)

```
discount(CustomerNumber, OrderNumber, OrderDesc, ItemNumber,
  Quantity, DeliverDate, TotalAmount, Requestor,'15 0 percent')
:-
premiumCustomer(CustomerNumber), TotalAmount > 1000.
```

(규칙 4)

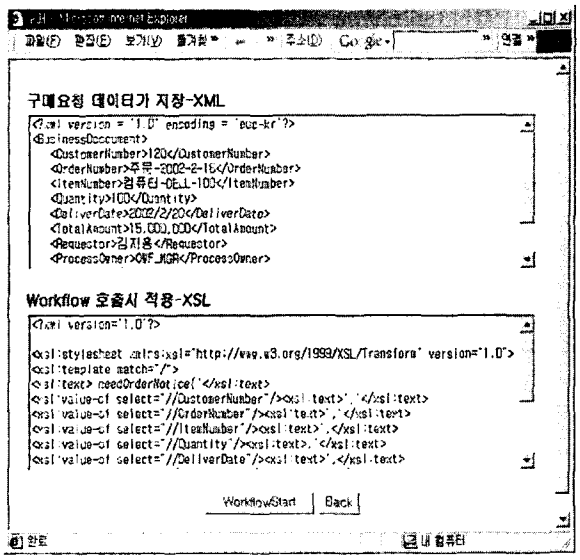
```
needOrderNotice(CustomerNumber, OrderNumber, OrderDesc,
  ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor)
:-
purchaseOrder(CustomerNumber, OrderNumber,ItemNumber, Quantity).
```

②,③단계는 사용자가 입력한 데이터가 XML문서로 저장되고 Prolog로 표현한 규칙을 변환기(Prolog2XML)를 통해 ERML(Executable Rule Markup Language)로 변환하는 과정이다. (규칙 2)를 변환한 결과는 [그림 4]와 같다.

```
<?xml version="1.0" encoding="euc-kr" ?>
- <RuleSet>
  - <hn>
    - <relationship>
        <relator>:-</relator>
      - <relationship>
          <relator>discount</relator>
          <var>CustomerNumber</var>
          <var>OrderNumber</var>
          <var>OrderDesc</var>
          <var>ItemNumber</var>
          <var>Quantity</var>
          <var>DeliverDate</var>
          <var>TotalAmount</var>
          <var>Requestor</var>
          <atom>15.0 percent</atom>
        </relationship>
        <relationship>
          <relator>,</relator>
        - <relationship>
            <relator>premiumCustomer</relator>
            <var>CustomerNumber</var>
          </relationship>
        - <relationship>
            <relator>></relator>
            <var>TotalAmount</var>
            <number>1000</number>
          </relationship>
        </relationship>
      </relationship>
    </hn>
</RuleSet>
```

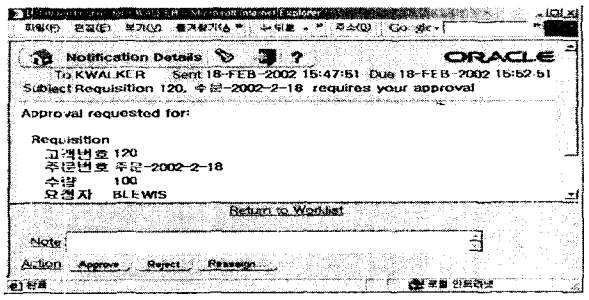
[그림 4] ERML로 표현된 능동문서의 업무규칙

④단계는 문서처리 및 검증을 위해 [그림1] Application Server계층에 있는 변환기 및 생성기가 각각 추론엔진, 워크플로우 엔진에서 필요로 하는 입력값을 생성한다. 아래 [그림5]는 추론엔진을 통해 워크플로우를 생성 및 시작하기 위한 오라클 프로시저를 호출하는 화면이다.



[그림 5] (규칙 4)수행을 위한 워크플로우 시작

⑤워크플로우 엔진 API를 이용해 작성해둔 오라클 저장 프러시저가 수행되면서 워크플로우가 생성되고 시작된다. ⑥공급업체 승인자가 구매관리 사이트에 접속하면 [그림 6]에서 처럼 자신의 작업리스트에서 구매자가 주문한 내역을 확인할 수 있으며 승인 및 거절을 할 수 있다. ⑦구매자는 자신의 작업리스트에서 공급업체의 승인 또는 거절 내역을 확인할 수 있다.



[그림 6] 공급업체 승인자의 통지 상세내역

4. 결 론

본 논문에서는 사무 문서에 함축되어 있는 업무 규칙을 Prolog로 표현하여 XML형태로 문서에 함께 포함시켰다. 그 결과, 문서처리 시스템은 사무 문서를 효과적으로 처리 할 수 있다. 즉, 기존의 문서처리 시스템은 문서 작성 규칙과 함께 서로 다른 업무 규칙을 지닌 문서를 처리하기 위한 로직을 포함하고 있었다. 하지만, 본 논문에서는 업무 규칙을 어플리케이션 로직과 분리하여서 문서처리 시스템은 업무 규칙 추론/실행엔진으로 단지 Prolog 인터프리터만 있으면 가능하다. 또한, 선언적(declarative) 의미로 업무 규칙을 기술하여서 이해하기가 쉽고, 비-프로그래머인 구매관리자, 마케팅관리자 같은 업무 전문가가 쉽게 규칙을 정의하고 수정할 수 있다. 변환기(Prolog2XML)를 통해 생성된 규칙을 포함하는 XML문서는 업무 프로세스를 자동화하기 위해 상용 워크플로우 관리 시스템과의 연동도 가능함을 보였다.

5. 참고 문헌

[1] 남철기, 김혜경, 배재학. 응용프로그램을 위한 일관된 XML뷰 제공에 관한 연구, 한국정보처리학회 춘계학술발표대회 논문집, 제8권 제1호, pp.1105~1108, 2001.

[2] 남철기, 양재군, 배재학. 검증규칙을 포함한 XML문서, 한국 정보과학회 추계학술발표대회 논문집, 제28권 제2호, pp.709~711, 2001.

[3] Bell, J., Brooks, D. Re-Engineering Case Study - Analysis of Business Rules and Recommendations for Treatment of Rules in a Relational Database Environment, Bellevue Golden, US West Information Technologies Group, 1990.

[4] Common Rules, http://www.alphaworks.ibm.com/tech/commonrules.

[5] Kuldar Taveter and Gerd Wagner. Agent-Oriented Enterprise Modeling Based on Business Rules,Proc. of 20th Int. Conf. November 2001.

[6] JPL, http://sourceforge.net/projects/jpl/.

[7] Nam, C.-K. and Bae, J.-H. J. A Framework for Processing Active Documents. (to be published)

[8] Oracle Workflow, http://www-apps.us.oracle.com/atg/wf/.

[9] RuleML Initiative, http://www.dfki.uni-kl.de/ruleml/.

[10] SWI-Prolog, http://www.swi-prolog.org/.