XML기반의 전자문서 처리방안에 관한 연구

A Study on XML-Based Electronic Documents

남철기[1], 장길상[2]

Abstract

Recently, due to the development of internet based information technology, organizations is utilizing electronic documents as important media for processing business work and sharing information exchange. These electronic documents are most form like documents and are important user interfaces for business applications. But, presently web-based electronic documents are lack of the characteristics that are taken in documents, such as document writing rules and its workflow. This paper deals with XML-based business document that include the structure, data, and logic and proposes the framework for validating the data and logic included within business documents using Prolog.

1. 서론

인터넷의 활성화와 인터넷 기반 시스템의 증가로 인해 전자문서의 중요성이 커지고 있다. 특히, 폼 형식의 전자문서는 사용자 인터페이스로써 널리 사용된다. 폼 문서의 구성요소는 크게 문서형식(structure), 내용(content), 의미(semantics)로 나눌 수 있다[1]. 현재 폼 작성을 위한 마크업 언어로는 HTML(HyperText Markup Language) 및 XML(Extensible Markup Language)를 주로 사용한다. 최근 주목을 받고 있는 XML은 과거 전자문서의 논리적 정보와 물리적 정보가 같이 표현되던 구조에서 문서의 논리적 구조(XML), 물리적 구조(XSL), 문서의 연결(Xlink, Xpointer)를 분리하려는 요구와 시도에 의해 생겨나게 되었다. XML은 이상적인 전자문서의 요건을 갖추고 있으며, 현재 모든 형태의 데이터와 문서를 통합, 저장, 처리할 수 있는 프레임웍을 제공한다[2]. XML문서가 전자문서의 표준으로 되어가는 것은 문서의 내용, 표현, 구조가 분리되어 있어서 재사용성이 우수하고 데이터교환이 쉽기 때문이다. XML의 이러한 사상을 폼에 적용한 대표적인 연구로는 XFA(XML Forms Architecture)를 들 수 있는데, 이것은 폼 형식의 전자문서를 위한 XML 어플리케이션이다[3]. XFA특징은 다음과 같다: (1) 데이터와 문서에 대해 다양한 보기(화면, 출력, 웹, 출판 등), (2) 문서형태의 정의와 데이터

1) 한국오라클(주)
2) 동국대학교 정보산업학과

를 함께 표현 또는 분리, (3)폼 객체와 데이터의 상대적, 절대적 위치, (4) 여러 개의 스크립팅, 계산 엔진, (5) 전자서명, (6) 상호 운용성을 위하여 XML 데이터 정의를 사용한다. 하지만, [그림 1]과 같이 현재 주로 사용되고 있는 마크업 언어인 HTML과 XML은 문서내용에 대한 검증과 문서가 함축하고 있는 비즈니스 규칙은 처리하기 어렵다.

기 능	HTML	Java	HTML+Java	XML+XSL
문서형식, 데이터, 로직을 저장				
작은 파일 크기	●			●
개발시간 단축	●			●
데이터 검증		●	●	
내부로직(비즈니스 규칙)				

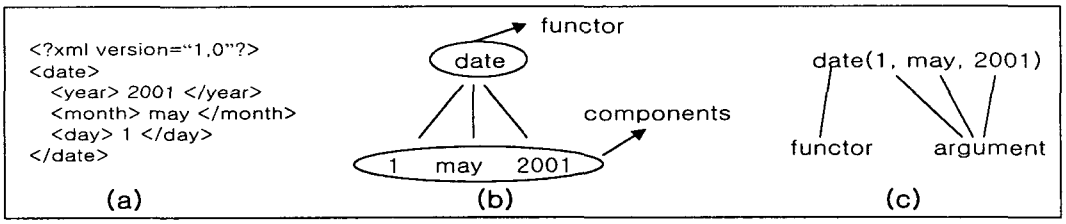
•**HTML**: HyperText Markup Language
•**XML**:Extensible Markup Language
•**XSL**: Extensible Stylesheet Language

[그림 1] HTML, Java, XML의 기능

이를 개선하기 위하여, 본 논문에서는 규칙 표현에 적합한 논리 프로그래밍 언어인 Prolog를 이용하여 폼 형식의 전자문서가 갖추어야 할 중요한 기능인 내부 로직(비즈니스 규칙)과 사용자가 입력한 데이터 검증을 위한 전자문서 처리 프레임워크를 제시한다. 또한, 비즈니스 프로세스의 자동화를 지원하기 위하여 WFMS(Workflow Management System)과 통합 방안도 제시 하였다.

2. XML과 Prolog와의 관계

Prolog와 XML과의 상호 관계를 살펴보면 XML 문서의 논리적 구조 자체는 구조적인 트리로 나타난다. 이러한 트리는 Prolog의 structure 객체로 매핑될 수 있다[4].
date를 표현하기 위해 [그림 2]에서 (a)는 XML, (b)는 트리, (c)는 트리를 Prolog로 표현한 것을 나타낸다.



[그림 2] Prolog와 XML과의 관계 표현

XML 문서는 엘리먼트로 구성되어 있고 각각의 엘리먼트는 <태그> ... </태그>의 형식이다. Prolog 프로그램을 XML형식으로 변환하기 위해서는 Prolog에서 사용하는 Herbrand Term(constants, variables, structures)과 Horn Clause(fact, rule)를 XML 문서의 엘리먼트로 변환하기 위한 규칙이 필요한데 [그림 3]과 같다.

Herbrand Term	Element	Horn Clause	Element
Constant(atom)	<atom>	Predicate	<relator>
Constant(number)	<number>	Relation Symbol	<relator>
Logic variables	<var>	Fact	<hn> <relationship>
Structures	<struc>	Rule	<hn> <relationship>

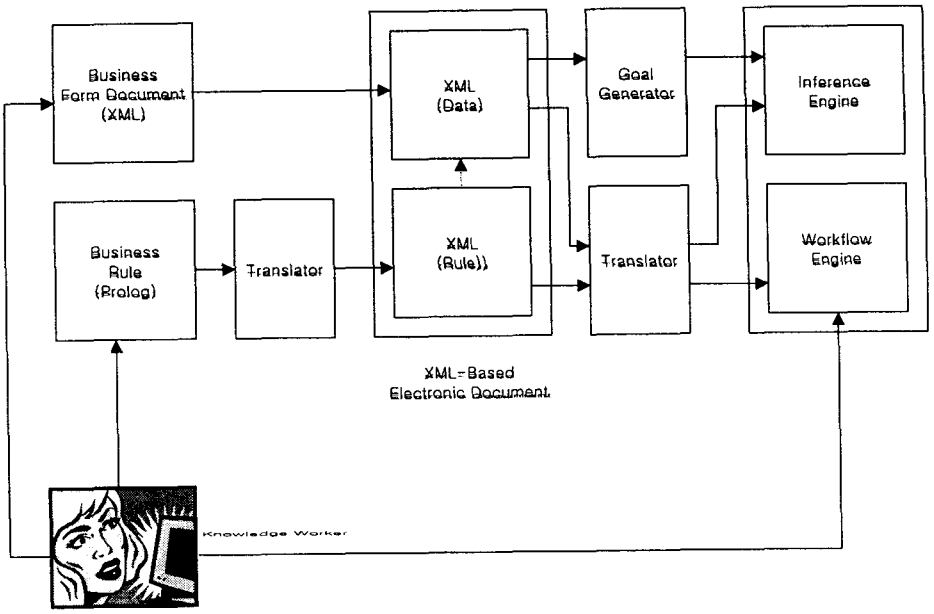
[그림 3] 엘리먼트 변환 규칙

3. XML기반의 비즈니스 문서 처리를 위한 프레임워크

비즈니스 규칙에 대한 여러 정의가 있지만, 일반적으로 다음과 같이 정의한다. "업무가 성취되는 방법이며, 조직 내에서의 업무의 상태와 프로세스에 대한 제약(restriction)과 지침(guideline)이다[5]." 이러한 규칙은 비즈니스의 함축된 의미(semantics)를 캡슐화하고, 데이터베이스가 데이터와 어플리케이션을 분리시켰듯이, 업무 규칙을 어플리케이션 로직으로부터 분리시킬 수 있다. 수세기 동안 조직은 문서를 사용해 왔다. 이들 문서의 대부분은 양식(form) 문서이다[1]. 본 논문에서의 비즈니스 문서는 양식 문서이고, 비즈니스 규칙을 포함하고 있으며, [그림 4]와 같이 문서의 내용, 형식, 로직(비즈니스 규칙), 표현을 가지고 있는 문서로 정의한다.

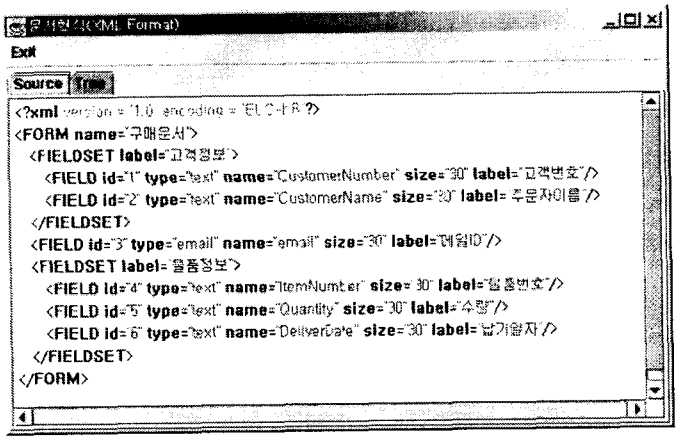
[그림 4] 비즈니스 문서

[그림 4]에서 정의된 비즈니스 문서를 처리하기 위한 프레임워크는 다음의 [그림 5]와 같다. 제시된 프레임워크는 지식관리자가 먼저 규칙 작성기(Rule Editor)를 이용하여 업무관련 비즈니스 규칙을 작성하고, 업무관련 워크플로우를 워크플로우 빌더(Workflow Builder)로 정의한다. 다음으로, 지식관리자가 폼 기반의 전자문서를 사용하여 업무관련 데이터를 입력하면, 제시된 프레임워크에 따라서 XML 기반의 비즈니스 문서 처리 작업이 진행된다.



[그림 5] 비즈니스 문서 처리 프레임워크

이러한 비즈니스 문서를 처리하는 과정을 설명하기 위하여, 조직 내에서 가장 중요한 업무중의 하나인 구매업무를 예로 들어 설명한다. 실험을 위하여 [그림 6]와 같이 XML로 문서형식을 정의 한다. 문서의 형식을 간결하게 위하여, 구매문서는 고객번호, 주문자이름, 메일ID, 물품번호, 수량, 납기일자로 이루어져 있다.



[그림 6] 문서 형식 정의

또한, 이 문서는 다음과 같이 문서의 비즈니스 규칙과 데이터 검증규칙과 같은 문서 내부로직을 포함하고 있다.

■ 비즈니스 규칙

(규칙 1) 고객번호가 100이상 200이하이면 우수고객이다.

(규칙 2) 우수고객이면서 총 구매금액이 1000만원 이상이면, 15%를 할인해 준다.

(규칙 3) 우수고객 또는 구매금액이 1000만원 이상이면 납기일이 10일이고, 구매금액이 1000이하이면 납기일이 20일이다.

■ 데이터 검증 규칙

(규칙 4) 메일ID는 반드시 입력한다.

위의 규칙들은 규칙표현에 접합하고 표현된 규칙은 구현 없이 즉시 수행되게 하기 위하여 Prolog를 사용한다. (규칙 2)와 (규칙 4)를 Prolog로 표현하면 다음과 같다.

(규칙 2)

```
discount(CustomerNumber, CustomerName, Email, ItemNumber,
   Quantity, DeliverDate, '15.0 percent')
:- premiumCustomer(CustomerNumber), TotalAmount > 1000.
```

(규칙 4)

```
requiredField(CustomerNumber, CustomerName, Email, ItemNumber,
    Quantity, DeliverDate, '메일ID를 입력해야 합니다')
:- var(Email).
```

Prolog로 표현된 규칙은 XML기반의 비즈니스 문서를 정의하기 위하여, Prolog로 만들어진 변환기를 통해 XML문서로 변환된다. (규칙 2)를 [그림 3]의 엘리먼트 변환규칙에 기반해서 변환하면 다음과 같다.

```
<?xml version="1.0" encoding="euc-kr" ?>
- <RuleSet>
  - <hn>
    - <relationship>
        <relator>:-</relator>
      - <relationship>
          <relator>discount</relator>
          <var>CustomerNumber</var>
          <var>CustomerName</var>
          <var>Email</var>
          <var>ItemNumber</var>
          <var>Quantity</var>
          <var>DeliverDate</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 7] XML로 표현된 Prolog규칙

4. 결론

일반적으로 폼 문서는 데이터 구조, 사용자 인터페이스, 내부 로직을 가지고 있다. 하지만, 웹 기반의 폼 문서를 작성하는데 가장 많이 사용하는 HTML 문서는 데이터 구조와 문서의 내부 로직을 처리하기가 어렵다. 또한, 문서 및 데이터에 대한 통합 프레임워크를 제공하는 XML도 문서의 내부 로직을 처리하기 위해서는 스크립트언어나 프로그래밍언어를 사용해야 한다. 본 논문에서는 비즈니스 문서에 함축되어 있는 업무 규칙을 Prolog로 표현하여 일관성 있게 XML로 문서처리가 가능함을 보였다. 또한, Prolog를 이용하여 선언적(declarative)으로 업무 규칙을 기술하여서 이해하기가 쉽고, 비-프로그래머인 구매관리자, 마케팅관리자 같은 업무 전문가가 쉽게 규칙을 정의하고 수정할 수 있다. 비즈니스 문서에 대한 본 논문의 접근방법을 통하여, 급변하는 비즈니스 요구사항을 만족시키기 위하여 쉽게 비즈니스 규칙을 수정할 수 있으며, 유연한 어플리케이션 개발 환경을 제공할 수 있다.

참고문헌

[1] Barclay T. Blair, John Boyer, XFDL: Creating Electronic Commerce Transaction Records Using XML,http://www8.org/w8-papers/4d-electronic/xfdl/xfdl.html.
[2] Extensible Markup Language (XML), http://www.w3.org/XML/.
[3] XML Forms Architecture (XFA), http://www.oasis-open.org/cover/xfa.html.
[4] Harold Boley, Relationships between Logic Programming and XML, Proc. 14th Workshop Logische Programmierung, W"urzburg, Jan. 2000.
[5] Kuldar Taveter and Gerd Wagner. Agent-Oriented Enterprise Modeling Based onBusiness Rules,Proc. of 20th Int. Conf. November 2001.