XSL-fo를 적용한 XML 문서 표현 시스템의 설계 및 구현

*이형문° **강치원 *정회경
배 재 대 학 교 컴 퓨 터 공 학 과
*{darkness,hkjung}@mail.paichai.ac.kr, **cwkang@kdn.com

Design and Implementation of XML Document presentation that apply to XSL-fo

Hyoung-Moon Lee° Hoe-kyung Jung
Dept. of Computer Engineering, Paichai University

요 약

인터넷 표준 문서인 XML(Extensible Markup Language)이 구조적인 내용만을 갖고 있기 때문에 문서를 보여주기 위한 표현 정보를 포함하는 스타일 시트(style sheets)가 필요하다. 이를 위해 W3C(World Wide Web Consortium)에서는 XML 문서의 구조적인 접근을 위한 XPath(XML Path Language)와 문서의 내용구조를 변환 하기 위한 XSLT(XSL Transformations), 그리고 포맷팅 정보를 위한 XSL-fo(XSL Formatting objects)로 구성되는 XSL(Extensible Stylesheet Language)를 제안하였다. 본 논문에서는 XML 문서와 XSL 스타일 시트를 이용하여 XML 문서의 데이터를 변환하고 XSL-fo 정보를 이용하여 결과 FO(Formatting Objects)를 생성한다. 또한 트리로 구성된 결과 FO 에서 페이지 정보와 FOT(Formatting Objects Tree)를 추출하는 FOT 생성부를 두었고, 디스플레이 관리기와 포맷팅 모듈 객체 관리기를 두어 FOT를 포맷팅을 위한 모듈로 저장 하고 디스플레이 해주는 포맷팅 처리 시스템을 설계 및 구현하였다.

1. 서론

컴퓨터 관련 분야의 기술진보는 문서의 전자화를 가속화 시켰다. 그러나 이러한 전자문서 처리를 위한 시스템들은 각기 독자적인 문서구조와 표현정보를 포함하기 때문에 시스템들 간의 문서 공유에 상당한 어려움이 발생한다. 이런 문제점은 문서의 표준화를 가속시켰고 문서의 내용과 표현정보는 분류되기 시작했다. 이와 더블어 인터넷의 발전은 공유에 입각하여 새로운 표준인 XML(Extensible Markup Language) 1.0 이 1998년 2월 10일로 W3C(World Wide Web Consortium)에 의해 제정되었다.

XML 은 문서의 내용과 구조 정보에 관한 데이터기술을 위한 언어로써, 문서를 표현하고 처리하기 위한 부가 기술들을 필요로 한다. 때문에 1999년 11월 16일 XSLT(XSL Transformations) 1.0과 XPath(XML Path Language) 를 표준으로 권고하였고, XSL-fo 를 2000년 3월 27일 현재 개발중이다.

본 논문에서는 XML 문서의 내용정보와 구조정보를 XSL(Extensible Stylesheet Language) 스타일 시트의 FO(Formatting Objects)를 적용해 표현하는 포맷팅 시스템을 설계 및 구현하였다.

본 논문의 구성은 다음과 같다.

2 장에서 본 시스템의 관련연구 인 XML과 XSL 의 기본 개념에 대해 살펴보고, 3 장에서는 시스템의 설계 부분으로 FOT 생성부와 포맷터로 나누어 설명한다. 4장에서는 본 시스템의 구현을, 5장에서는 결론 및 고찰, 향후과제를 제시한다.

2. 관련 연구

2.1 XML 개념

XML은 인터넷상에서 구조화된 문서의 교환, 그리고 응용프로그램에서 보다 쉽게 처리하기 위해 1996년 W3C에서 제안되어 SGML(Standard Generalized Markup Language) 의 부분집합으로 1998년 표준으로 제정되었다. 이는 단순히 고정되어 있는 마크업 언어인 HTML(Hyper Text Markup Language)을 뛰어넘는 인터넷상에서 사용될 수 있는 메타 언어로서의 유연함을 보인다. XML의 최종 목표는 현재 웹(Web)상에서 HTML이 처리되는 것처럼 일반적인 SGML의 처리가 가능하도록 하는 것이다.

2.2 XSL 개념

XSL 은 크게 3개의 구성요소로 되어 있다. 변환을 위한 구성 요소인 XSLT와 XML 문서의 구조에 기초하여 접근하는 XPath , 마지막으로 렌더링(rendering)을 위한 XSL-fo 로 나뉘어 진다.

2.2.1 XSL-fo

DSSSL-O(DSSSL Online Application Profile)의 부분 집합인 DSSSL-O 핵심 흐름 객체들(Core Flow Objects)과 HTML/CSS(Cascading Style Sheets)의 핵심 흐름 객체들을 포함하는 포맷팅 모델을 제시한다.

포맷팅 모델은 크게 블록수준(block level)과 인라인수준(inline level)으로 나뉘고 테이블, 리스트 등의 포맷팅 객체 들은 이에 종속되는 모델링을 제시한다. 블록 수준 포맷팅 객체는 영역이라는 개념아래 처리되는 특정한 공간이다. 인라인 수준 포맷팅 객체는 영역을 채우기 위한 특정 공간의 연속이다. 즉, 블록 수준은 단락 정도로 해석하고 인라인 수준은 라인 단위로의 해석이 가능하다.

3. 시스템 설계

3.1 시스템 구성

본 시스템의 구성은 그림 1 과 같다.

XML 문서와 XSL 스타일 시트는 파서를 통해 W3C 표준 인터페이스인 DOM(Document Object Model) 으로 구성된다. 파서 API 를 사용해 XML DOM 객체와 XSL DOM 객체를 XSLT/XPath 처리기에 의한 변환 과정을 거쳐 새로운 DOM 객체인 결과 FO 트리로 생성한다. FOT 생성기에서는 결과 FO 트리인 DOM 객체 트리를 운행하며 포맷터에서 제공하는 인터페이스 클래스를 사용, 새로운 트리인 FOT를 생성하여 포맷터로 전송한다. 포맷터에서는 FOT를 운행하며 포맷팅 모듈에 맞게 배분한다. 각 포맷팅 모듈은 영역 및 속성정보 등의 계산후 포맷팅 모듈 객체를 생성하여 포맷터에게 템플리트 리스트(template list) 형태로 관리된다.

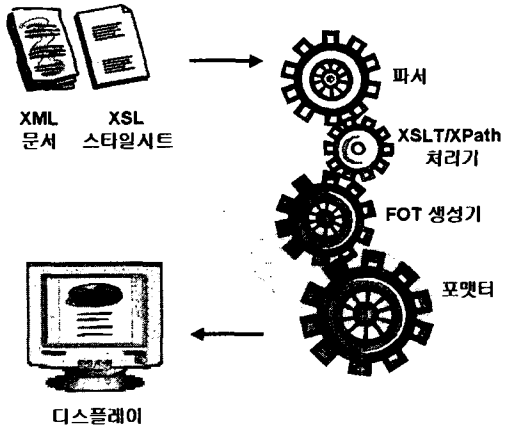

그림 1 시스템 구성도

3.2 FOT 생성부

FOT 생성부에서는 결과 FO 트리를 탐색하여 페이지 정보와 일반 포맷팅 정보로 분류하고, 각각 페이지 관리기와 FOT 생성기로 보내어 처리한다. XSL-fo의 페이지 정보는 포맷팅을 위한 객체들과 나뉘어지고 참조되므로 본 시스템에서는 페이지 정보를 관리하 는 페이지 관리기, 포맷팅 객체 트리를 생성하는 FOT 생성기로 구성하여 설계하였다.

3.2.1 페이지 관리기

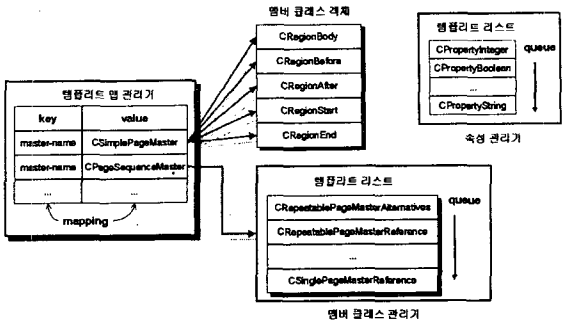

그림 2 페이지 관리기 구성도

결과 FO 트리에서 추출한 페이지 관련 포맷팅 객체들은 각각 클래스(class)로 매핑(mapping)되도록 설계하여 그림 2 에서 보이는 페이지 관리기에서 객체로써 관리된다. 페이지 관리기의 주 관리기인 템플리트 맵 관리기는 페이지 이름을 키(key)값으로 하여 단일 페이지 객체나 연속 페이지 객체와 매핑하여 관리한다.

단일 페이지 클래스인 CSimplePageMaster 는 XSL-fo 스펙에 기술된 5개의 영역과 매핑한 클래스의 객체를 멤버로 포함하고 있다. 또한 연속 페이지 클래스인 CPageSequenceMaster 는 템플리트 리스트 관리기를 포함하여 연속 페이지의 구성을 위한 페이지 참조 클래스들을 순차적으로 관리하도록 설계하였다. 각각의 참조 클래스들은 단일 페이지 객체들을 참조하므로 페이지 맵 관리기와 리스트 관리기는 매우 유기적으로 상호 운용된다.

3.2.2 FOT 생성기

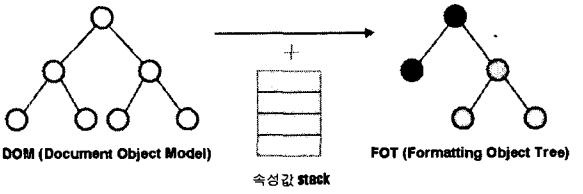

그림 3 FOT 생성 과정

FOT 생성기는 결과 FO 트리 중 페이지 관리기에서 추출한 페이지 관련 객체들을 제외한 모든 객체들을 탐색하여 그림 3 에서 보이는 과정을 포맷터에서 필요한 트리로 변환한다. 그림 3 은 결과 FO 트리를 깊이 우선 순위 방법으로 탐색하면서 포맷터에서 제공하는 인터페이스 클래스의 객체를 생성하여 새로운 트리인 FOT 를 생성하는 과정이다. 결과 FO 트리를 탐색하는 과정에서 속성값 정보는 스택에 저장 했다가 트리 객체중 하위 노드가 상속하는 경우 스택 값을 FOT 노드에 추가한다. FOT 각 노드들을 구성하는 객체들은 그림 2 의 속성 관리기를 포함하므로 입력된 속성값이나 상속 값만을 관리하도록 설계하였다.

3.3 포맷터 설계

FOT 생성부로부터 넘어온 각 FOT 노드들은 대응되는 포맷팅 모듈을 통해 영역 및 폰트 정보 등의 포맷팅 객체의 정보를 계산하여 포맷팅 한다. 포맷터에서는 계산결과를 블록

이나 그림 같은 단순 모듈과 테이블이나 리스트와 같은 노드집합의 모듈로 구분 지어 하나의 포맷팅 모듈 객체를 생성한다. 포맷팅 모듈 객체는 그림 5 에서 보이는 템플리트 배열 관리기에서 관리되며, 실제 화면을 관리하는 디스플레이 모듈과 연계하여 운용되며, 디스 플레이 모듈에서는 그림 4 에서 보이는 영역관리와 위치 관리를 담당한다.

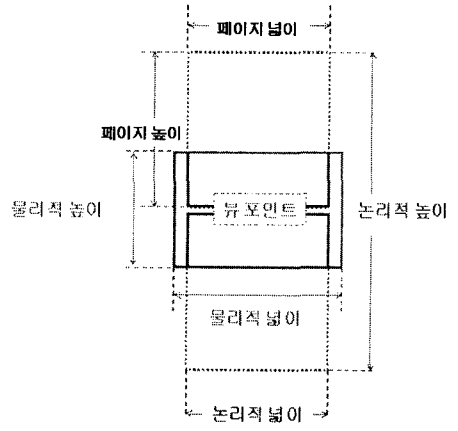

그림 4 디스플레이 관리모듈의 처리 개념도

그림 4 는 디스플레이 관리모듈의 처리에 대한 것이다. 전체 영역인 논리 영역의 정보를 가지고 있는 각 포맷팅 모듈 객체들은 디스플레이 관리기를 통해 현재 보이는 뷰 포인트 즉, 물리적인 영역에 해당하는 객체들만 선별하여 디스플레이 한다. 그림 4 에서처럼 뷰 포인트 영역중 2 개 이상의 페이지를 거치게 되는 경우는 포맷팅 모듈 객체 내부에 포함된 모듈인 영역 처리 모듈에서 검사하여 페이지 정보관리기와 연계되어 처리된다.

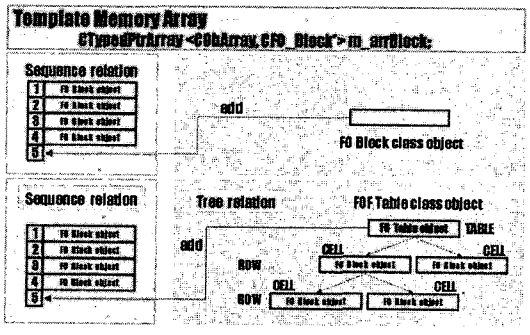

그림 5 포맷팅 모듈 객체관리 개념도

그림 5 는 템플리트 배열 관리기와 각 포맷팅 모듈 객체의 관계를 나타낸다. 블록과 같은 단순 모듈의 객체와 테이블과 같은 노드 집합 모듈의 객체가 템플리트 배열 관리기에 추가 되어 관리되는 그림이다. 블록 클래스는 내부 처리 모듈로 영역 처리 모듈, 폰트처리 모듈, 인라인 처리 모듈 그리고 단락 처리 모듈등을 포함한다. 또한 테이블 클래스는 블록 클래스를 상속받기 때문에 기본 모듈은 모듈 블록 클래스와 동일 하다. 그러나 노드집합의 처리를 위해 하위로 트리 형태의 블록 객체들을 관리하는 추가 모듈을 포함한다.

4. 구현

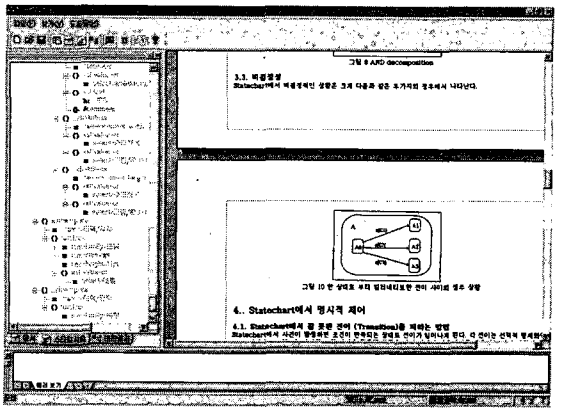

그림 6 포맷터를 내장한 브라우저

본 시스템은 WindowNT 4.0 환경에서 Visual C++ 6.0 으로 개발하였다. 파서 및 XSLT/XPath 처리기는 Microsoft 사의 MSXML 파서를 사용하였다. 그림 6 은 시스템의 전체 결과를 나타낸다. 왼쪽의 트리는 XML 문서, XSL 스타일 시트, 변환 결과 FO Tree 를 보여준다. 아래 쪽의 에러창은 XSL 스타일 시트 작성시 XSL-fo 스펙에 기준하여 fo 요소나 속성 등의 오류에 대해 알려주는 역할을 한다. 마지막으로 오른쪽 포맷팅 창은 실제 포맷터에서 화면에 디스플레이 하는 화면을 보여준다.

5. 결론

점차 가속화 되는 전자화의 물결속에 차세대 언어라고 불리는 XML 과 이를 뒷바침 해주는 여러 관련된 표준들은 XML 의 방향을 제시해주고 있다. 본 논문에서는 XML문서와 XSL 스타일 시트를 입력으로 생성된 결과 FO 트리를 변환한 FOT를 포맷팅하여 브라우징 해주는 처리 시스템을 설계 및 구현하였다. 이로써 XML 에 관련 기술들을 이용하여 새로운 비즈니스 모델 및 시스템 모델을 구성하기 위해 필요한 기반 기술을 확보하였다. 따라서 본 논문에서의 연구는 XML 관련 기술의 발전 뿐만 아니라 새로운 시스템 구성시에 유용한 기반 기술을 제공할 수 있다.
본 시스템은 링크 기능 처리에 관한 향후 연구 및 XML 문서의 구조와 데이터를 변경하기 위한 XSLT 편집기에 관한 연구가 절실히 필요하며, 향후 연구는 유연한 관계 속에 함께 진행되어야 할 필요가 있다.

6. 참고문헌

[1]. "Professional Style Sheets for HTML and XML", Frank Boumphrey, 1998 Worx Press
[2]. W3C, Extensible Markup Language (XML) Version 1.0, http://www.w3.org/TR/REC-xml, Feb. 10, 1998
[3]. W3C, Extensible Stylesheet Language (XSL) Version 1.0, http://www.w3.org/TR/xsl/, March 27 ,2000
[4]. MS, MSDN Online XML Developer Center, http://msdn.microsoft.com/xml/default.asp