CML을 이용한 웹 기반 차트출력시스템

Web-based Chart Generating System Using CML

윤 현 님* 김 양 우**
Hyun-Nim Yoon Yang-Woo Kim

요 약

차트는 여러 종류의 정보를 시각적으로 표현하기 때문에 정보 전달력이 높다. 이런 이유로 웹 개발자들도 정보를 표현할 때 차트를 자주 활용한다. 그러나 차트를 활용하기 위해서는 차트를 작성하는 전용 프로그램이 필요하며, 작성된 차트도 일반적으로 래스터 방식의 이미지로 표현되기 때문에 차트정보를 공유하기 어렵다. 이러한 래스터 방식의 이미지는 이미지의 크기가 고정되기 때문에 이미지의 크기를 변경하면 이미지가 왜곡된다는 문제점이 발생한다. 본 논문에서는 차트정보의 호환성과 차트정보의 공유 문제를 해결하기 위해 CML(Chart Markup Language)을 이용한 웹 기반의 차트출력시스템을 제안한다. 본 논문에서 제안한 차트출력시스템은 XML, Text, HTML 문서를 분석하여 차트정보를 추출한 후, 추출한 차트정보를 CML로 변환하고, 변환된 CML 문서를 벡터 방식을 이용하여 차트를 웹브라우저에 출력한다. 벡터 방식은 이미지를 표현할 때 벡터를 활용하기 때문에 이미지의 크기가 변경되더라도 이미지가 깨지거나 왜곡되는 현상이 없다는 장점이 있다. 본 논문에서 제안한 CML은 차트정보를 표현하기 위해 정의한 XML 기반의 차트생성언어이다. 웹에서 차트를 표현할 때 CML을 활용하면 차트정보를 쉽게 공유할 수 있으며, 차트정보 변환이 수월하다는 장점이 있다.

Abstract

Charts can propagate information more effectively by visualizing various types of information. Due to this reason, many web developers often use charts when they display information on the web. However, using a chart requires a special dedicated software to install and it is difficult to share the finished charts since they are usually in raster format. The raster format images are fixed in size, therefore, experience image distortion problem when their sizes are changed. In this paper, we propose a web-based chart generating system using CML(Chart Markup Language) to solve the current compatibility and sharing problems. The proposed chart generating system first analyzes XML, Text, or HTML documents to extract chart information in the documents, and then it converts collected chart information to CML. The converted CML documents are displayed on the web browser using a vector method. The vector method has an advantage of protecting its images from distortion even when image sizes are scaled. The CML, proposed in this paper, is a chart markup language based on XML for modeling charts information. If CML is used for presenting a chart in a web, it makes easier to share and to convert the chart information.

☞ keywords : CML, Chart Generating System, XML, Chart

1. 서론 및 관련연구

웹과 사용자의 상호작용이 증가함에 따라 사용자들은 웹에서 정보를 효과적으로 표현할 수 있어야 한다[1][2]. 정보를 효과적으로 표현하기 위한 가장 좋은 수단은 그래픽 정보를 이용한 차트이다. 차트는 복잡하고 다양한 정보를 시각화시키고 축약시키기 때문에 정보전달 효과가 높고 정보에 대한 이해가 쉽다는 장점이 있다.

이와 같은 장점으로 현재 웹에서는 차트가 많이 활용되고 있다. 현재 차트 작성에 많이 사용되고 있는 대표적인 프로그램에는 마이크로소프트사의 엑셀과 MathWorks사의 MATLAB이 있다[3][4]. 엑셀과 MATLAB은 차트를 표현할 수 있는 대표적인 프로그램이지만, 이 프로그램들의 결과물들은 서로

* 정 회 원 : 동국대학교 정보통신공학과 박사과정
yhnim@dongguk.edu
** 종신회원 : 동국대학교 정보통신공학과 교수
ywkim@dongguk.edu(교신저자 및 책임저자)
[2008/08/24 투고 - 2008/08/28 심사 - 2008/09/22 심사완료]

호환되지 않고 프로그램 종속적이라는 단점이 있다. 또한 웹에서 차트를 활용할 때 각 프로그램에서 구현한 차트는 이미지로 저장하여 웹페이지에 사용해야 한다. 이 경우, 이미지를 웹페이지에 삽입하면 파일 전송속도의 저하를 초래할 뿐만 아니라 정보를 공유하기도 어렵다는 단점이 있다[5][6].

프로그램에서 구현한 차트를 이미지로 저장하는 경우와 같이 웹에서 주로 활용되는 그래픽 표현 방식은 래스터(raster) 방식이다. 래스터 방식은 이미지를 이진파일 형태로 저장하며, 래스터 방식의 이미지는 이미지 크기가 고정되고 픽셀에 색채정보가 설정되어 있으므로 이미지의 크기가 변화되면 이미지의 원상이 왜곡되어 이미지가 확대되었을 경우에는 해상도가 떨어진다는 단점이 있다. 또한 래스터 방식은 사용자가 파일 내용을 읽을 수 없고 편집이 불가능하다는 단점이 있다. 래스터 방식의 문제점을 해결하기 위해 나온 것이 벡터 방식이다. 벡터 방식은 이미지를 표현할 때 벡터를 활용하기 때문에 이미지의 크기가 변경되더라도 이미지가 깨지거나 왜곡되는 현상이 없으며, 이미지의 크기에 관계없이 일정한 파일 크기를 갖는다는 장점이 있다.

웹상에서 차트를 작성하기 위한 시스템은 없지만, 그래픽 처리를 수행하기 위해 XML을 사용하여 벡터 방식으로 그래픽을 표현하는 마크업언어로는 VML(Vector Markup Language)[7][8][9][10][11], PGML(Precision Graphics Markup Language)[12][13], SVG(Scalable Vector Graphic)[14][15]가 있다.

VML은 웹브라우저에서 그래픽을 구현할 수 있도록 W3C에서 제안한 XML 기반의 그래픽 마크업 언어이다[7]. VML은 벡터 방식으로 그래픽 정보를 처리하기 때문에 래스터 방식보다 그래픽 처리에 있어서 훨씬 더 유연하다는 것이 가장 큰 장점이다. 그러므로 이미지의 크기 변화에도 해상도가 영향을 받지 않으며, 이미지의 크기가 변화해도 파일의 크기가 변화하지 않는다는 장점이 있다. VML은 웹상에서 구조적 데이터 표현을 위해 XML을 기반으로 하므로 XML의 특징인 개방성, 단순성, 확장성을 지닌다[16]. 또한 VML은 HTML 페이지 안에 삽입되어 전달되므로 외부 파일을 참조하는 방식보다 빠르게 그래픽을 처리할 수 있으며, CSS 2.0 뿐만 아니라 DOM(Document Object Model)과 같은 다른 W3C 표준을 지원한다[17][18]. 그러나 VML은 그래픽 요소를 표현하는데 필요한 속성의 사용이 복잡하고 어렵다는 단점이 있다[11].

PGML은 어도브(Adobe)사의 포스트스크립트 및 PDF(Portable Document Format)형식을 확장한 것으로 포스트스크립트 명령을 XML 요소와 속성으로 변환한 것이다[12]. PGML은 대부분의 그래픽에 대한 요구를 만족시킬 수 있는 강력한 기능을 가진 언어이다. 또한 PGML은 간단한 애니메이션 컴포넌트와 복잡한 애니메이션의 동적인 처리를 위한 객체모델을 포함하고 있으며, 기존의 웹 표준화 규격인 HTML, XML, XSL, CSS, DOM, RDF와 호환 가능하다는 장점이 있다. PGML은 웹페이지 내의 텍스트나 이미지의 배치 및 글꼴, 색상 등을 제어하는 벡터 그래픽 방식의 언어이다. 이러한 PGML의 구성요소는 패스(path), 텍스트, 래스터 이미지이다. 패스는 선을 그리는 기능 또는 이미지, 택스트 객체를 이동 및 배치하는 기능을 수행한다. 텍스트는 문서에 삽입될 문자열을 의미한다. 래스터 이미지는 PGML에 의해서 웹페이지에 배치될 이미지이다. PGML은 벡터 그래픽 방식을 취하고 있으며 특별한 플러그인이나 뷰어가 없어도 웹브라우저에 출력할 수 있지만, 래스터 방식의 이미지 파일을 활용하기 때문에 이미지 파일을 편집할 수 없다는 단점이 있다.

W3C는 XML에서 그래픽을 구사하기 위해 SVG 규격을 발표했다[14]. SVG는 벡터 방식을 기반으로 서로 다른 데이터베이스간의 그래픽 표현 데이터를 전달하는 기능, LAN 이용 또는 웹 이용 애플리케이션을 위해 관련 그래픽 및 텍스트를 간략하게 결합하는 기능, 다른 데이터 타입간의 변환 기능 등이 있다. 그러나 SVG는 벡터 방식을 기반으로 함으로 그래픽 표현에 장점이 있지만, 그래픽 개체의 형식을 지원하는 뷰어가 필요하므로 범용 웹브라우저에서 디스플레이가 되지 않는다는 단점이 있다[15].

McKeown과 Grimson은 웹에서 차트를 동적으로 출력하는 동시에 이미지의 크기 변화에 영향을 받지 않도록 하기 위해서 XML 기반의 SVG를 활용하는 방법을 제안하였다[19]. XML을 기반으로 함으로 플랫폼에 독립적이며, 벡터 방식을 채택하여 그래픽을 처리한다는 장점이 있으나, SVG는 웹브라우저에서 독립적으로 실행될 수 없다는 단점을 여전히 해결하지 못하였다.

Sagar는 수식, 차트, 벡터 방식 이미지 등을 표현하기 위한 브라우저를 구현하였다[20]. 수식과 벡터 방식 이미지를 표현하기 위해서 XHTML, CSS와 XML 기반의 MathML, SVG를 활용하였다. 그러나 이런 수식과 벡터 방식 이미지들을 웹브라우저에서 출력하기 위해서는 별도의 플러그인 소프트웨어가 필요하다는 단점이 있다.

본 논문에서는 차트 표현의 호환성과 범용성 문제와 차트정보의 공유성 문제를 해결하고 사용의 편리성을 위해 웹 기반의 차트출력시스템을 제안한다. 본 논문에서 제안한 차트출력시스템은 그래픽 처리를 위한 XML 기반 마크업언어의 복잡성에서 탈피하고, 차트의 표현 및 출력을 수월하게 하기 위하여 차트 마크업언어(CML, Chart Markup Language)를 정의한다. CML은 전문 개발자뿐만 아니라 일반 웹디자이너도 쉽게 활용할 수 있다는 장점이 있다. 본 논문에서 제안한 차트출력시스템은 사용자의 편의성을 위해 XML, Text, 또는 HTML 파일에서 차트정보를 자동으로 추출하여 분석하고 CML 형태로 변환하도록 설계되었다. 그리고 제안한 차트출력시스템은 별도의 차트 작성을 위한 프로그램이 필요하지 않으며, 웹 표준에 근거하므로 플랫폼에 관계없이 웹브라우저에서 차트를 표현할 수 있다는 장점이 있다. CML을 이용한 차트는 일반적으로 활용되는 이진 데이터의 차트와는 달리 데이터의 변환, 수정, 공유가 용이하다는 장점이 있다. 본 논문에서 제안한 차트출력시스템은 웹 프로그래밍을 모르는 사용자들도 웹에서 차트를 생성할 수 있도록 도와주는 웹기반 차트출력시스템으로 활용할 수 있으며, 기존의 텍스트 문서를 입력받아 웹에서 차트를 생성하는 웹기반 차트생성기로도 활용할 수 있다. 또한 본 논문에서 제안한 차트출력시스템은 텍스트 문서에서 차트정보를 추출하여 XML로 변환하는 기능을 가지고 있기 때문에 XML 변환기로도 응용하여 활용할 수 있다는 장점이 있다.

2. CML 구조

차트 표현을 위한 마크업언어를 정의하기 위해서는 차트를 표현하기 위해 요구되는 조건들을 먼저 파악해야 하고, 이 정보들을 활용하여 마크업언어의 DTD(Document Type Definition)를 작성해야 한다. 차트 표현에 필요한 요소들에는 차트의 종류, 차트를 그리기 위한 기본 영역, 차트 제목, 차트의 값 축 제목, 차트의 항목 축 제목, 차트의 값 축 표현, 눈금단위 표현을 위한 최대값과 최소값 정보, 차트의 항목 축 표현, 항목 명과 항목의 개수 표현 정보, 범례 표현, 데이터 처리, 데이터 영역, 효과적인 차트 표현을 위한 색깔 정보 등이 있다. 본 논문에서 제안한 CML의 DTD는 이와 같은 차트정보를 바탕으로 정의하였다.

2.1 CML DTD의 정의

XML을 이용한 차트의 표현을 위하여 본 논문에서는 CML을 제안한다. CML은 차트의 정보를 표현하기 위한 다양한 논리적 정보를 포함하고 있으며, 일반 사용자들도 쉽게 사용할 수 있을 정도의 간결한 구조를 제공하고 있다.

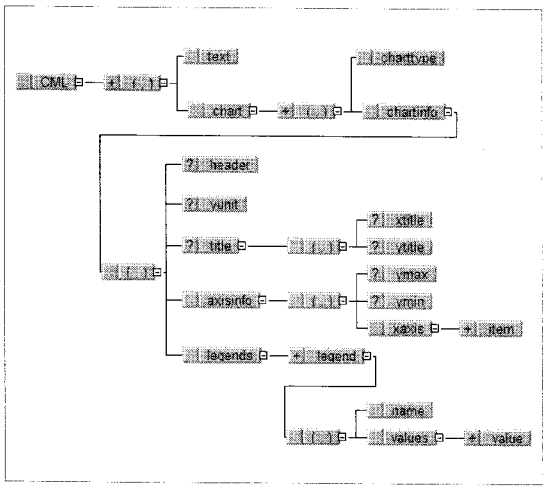
DTD는 XML 문서의 내용 구조 모델을 설명하는 것으로 그 요소의 속성과 서로의 관계를 구조화하는 방법을 말해준다[21]. 이로써 그 애플리케이션을 위해 작성한 XML 문서의 내용과 구조를 검사하는 처리를 완벽하게 제어할 수 있다. CML DTD에서 표현하고 있는 중요 차트정보를 정리하면 표 2와 같다.

(표 2) CML DTD에서 표현하고 있는 중요 차트정보

요소	속성	기능 설명
text	없음	차트에 대한 설명 또는 관련 내용
chart	없음	차트의 표현
charttype	shape, planetype	차트 종류와 기본평면 유형
chartinfo	없음	차트 표현을 위한 요소
header	headerposition	차트의 제목
yunit	없음	데이터 단위 표현
title	없음	막대 차트의 축 제목 정보
xtitle	position	막대 차트의 항목 축 제목
ytitle	position	막대 차트의 값 축 제목
axisinfo	없음	막대 차트의 축 표현 정보
ymax	없음	막대 차트의 값 축 표현을 위한 최대값
ymin	없음	막대 차트의 값 축 표현을 위한 최소값
xaxis	없음	막대 차트의 항목 축 표현 정보
item	없음	항목 값 표현, 하나 이상 출현
legends	없음	범례 정보
legend	color	항목에 대한 개별적인 범례 표현, 하나 이상 출현
name	없음	범례의 이름 정보
values	없음	범례에 대한 항목의 데이터 정보
value	없음	각 항목의 범례에 대한 데이터 값 표현

표 2에서 <text> 요소는 차트에 대한 정보나 설명 또는 관련 내용 등을 작성할 수 있는 요소이고, <chart> 요소는 실질적인 차트의 내용을 표현하기 위해 필요한 요소들을 포함한다. CML 문서에서 <text>와 <chart> 요소는 일반적인 문서에서 글과 차트가 번갈아 작성되는 것처럼 여러 번 출현할 수 있다. <charttype> 요소는 차트의 유형을 표현하며, "shape" 속성과 "planetype" 속성을 갖는다. 차트의 모양을 나타내는 "shape" 속성에는 원형(circle), 수평 막대(hbar), 수직 막대(vbar), 선형(line)과 같은 정보를 포함한다. "planetype" 속성은 차트의 바탕 영역에 대한 설정으로 "1"은 눈금선을 표시하며, "2"는 눈금선을 표시하지 않는다. <chartinfo> 요소는 하나의 차트를 표현하는데 필요한 모든 요소를 포함한다. 포함하는 요소는 <header>, <yunit>, <title>, <axisinfo>, <values>, <legends> 요소이다. <header> 요소는 차트의 제목을 표현하고, 차트 제목의 위치를 속성으로 갖는다. <yunit> 요소는 각 차트에서 나타내고 있는 데이터 단위의 시각적인 효과를 위해 표현한다. <title> 요소는 차트의 항목축(x축)의 제목과 값 축(y축)의 제목을 표현하기 위한 <xtitle>와 <ytitle> 요소를 포함한다. <xtitle>와 <ytitle>는 제목의 위치를 속성으로 갖는다. <axisinfo> 요소는 <ymax>와 <ymin> 그리고 <xaxis> 요소를 포함하고 있다. <ymax> 요소는 값 축의 최대값을 표현하며, <ymin> 요소는 값 축의 최소값을 표현한다. <xaxis> 요소는 막대 차트의 x축인 항목 축의 각 항목과 원 차트의 각 항목을 표현하기 위한 요소를 포함한다. 차트에 나타나는 각 항목은 <item> 요소로 표시한다. 차트에서는 하나의 항목에 대하여 여러 가지 사항들을 비교할 수 있어야 한다. 예를 들어 TV 항목에 대하여 판매량, 재고량, 입고 요구량 등을 비교할 수 있다. 이와 같이 차트에서 한 항목에 대한 여러 가지 비교 사항들은 범례를 통하여 구별될 수 있다. <legends> 요소는 여러 개의 범례 정보를 포함하는 요소이다. 그리고 <legends> 요소에는 하나의 범례를 의미하는 <legend> 요소가 하나 이상 포함된다. <legend> 요소의 속성으로는 범례의 색상이 있다. 각 범례를 나타내는 <legend> 요소에는 각 항목에 대한 값을 입력할 수 있어야 한다. DTD에서는 이를 위하여 <legend> 태그 하위에 <values> 태그를 삽입하였다. <values> 요소는 범례 내에서 각 항목에 대한 막대의 높이를 표현하는데 사용되는 데이터를 표현한다. 또한 시각적 효과를 위해 색깔 속성을 포함한다. 각각의 데이터 값에 대해서는 <value> 요소를 사용하여 표현한다.

그림 1은 표 2에 나타난 차트정보를 기반으로 작성한 CML DTD의 트리 구조를 보인 것이다. 그림 1에 표현된 DTD의 트리 구조에서 알 수 있듯이 CML 문서는 여러 개의 <text> 요소와 <chart> 요소로 구성될 수 있다. 그러므로 CML 문서에는 여러 개의 차트가 출현할 수 있으며, 차트에 대한 설명이나 관련 글이 차트의 전후에 출현할 수 있다. 하나의 차트는 차트의 유형을 결정하는 <charttype>과

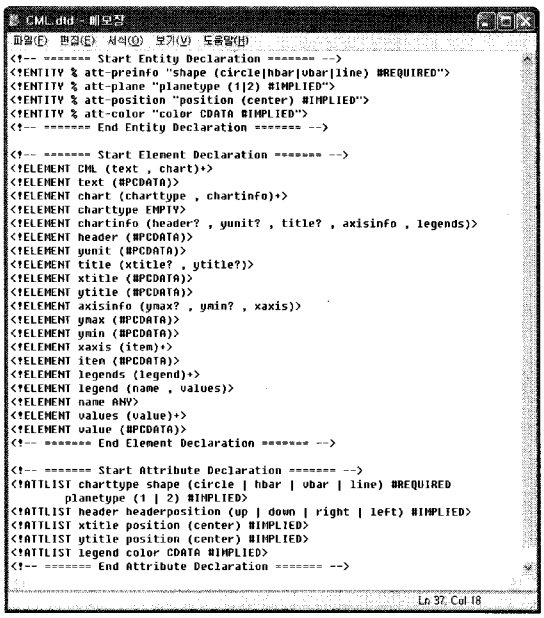


(그림 1) CML DTD의 트리구조

차트 안에 표현될 값을 포함하는 <chartinfo> 요소로 이루어짐을 그림에서 확인할 수 있다. 그리고 <chartinfo> 요소는 다시 <header>, <yunit>, <title>, <axisinfo>, <legends> 요소를 자식 요소로 갖는 구조로 되어 있다. <axisinfo> 요소는 <ymax>, <ymin>, <xaxis> 요소를 자식 요소로 가지며 <xaxis> 요소 아래에는 항목을 표시하는 <item> 요소가 하나 이상 나타난다. 범례를 나타내는 <legends> 요소들은 자식 요소로 하나 이상의 <legend> 요소를 갖는다. 그리고 <legend> 요소는 자식 요소로 범례의 이름을 표시하는 <name> 요소와 값을 표시하는 <values> 요소를 갖는다. <values> 요소에는 자식 요소로 항목의 개수만큼 <value> 요소가 포함되며, 각 항목의 범례에 대한 데이터 값을 저장한다. <item>, <legend>, <value> 항목은 한 번 이상 발생하는 것을 기본으로 한다. 이것은 지정된 개수의 데이터를 표현하는 것이 아니라 가변적인 데이터를 표현할 수 있도록 하기 위함이다. 그림 2는 CML DTD를 보인 것이다.

2.2 CML 문서의 작성 예제

본 절에서는 표 3에 나타난 가전제품 정보를 이용하여 CML 문서의 작성 예를 설명한다. 상품으로



(그림 2) CML DTD

는 TV, Aud, Cam이 있으며, 각 상품에 대한 정보로는 판매량(sold), 재고량(stock), 가격(price)이 존재한다. 이와 같은 경우에 각각 가전제품인 TV, Aud, Cam은 항목이 되고, 판매량, 재고량, 가격은 항목에 대한 비교 정보가 되어 범례에 사용된다고 가정하자.

(표 3) 가전제품 정보

	TV	Aud	Cam
Sold	30	25	40
Stock	35	20	25
Price	250	150	200

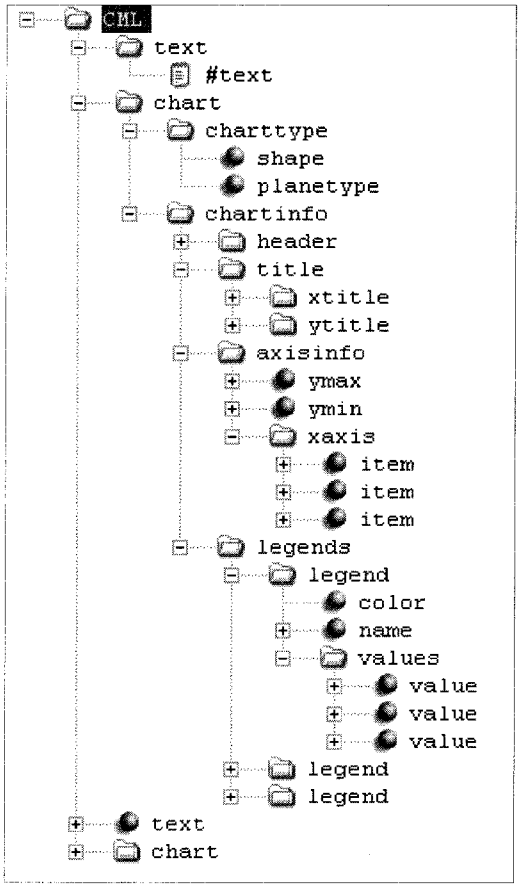
그림 3은 표 3에서 각 가전제품별 판매량, 재고량, 가격을 이용하여 작성한 수직 막대 차트의 CML 문서이다. 그림 3에서 ①은 <text> 요소로써 차트에 대한 설명이나 일반적인 텍스트를 표시하는 곳이다. ②는 차트의 종류와 차트를 그리기 위한 기본영역의 유형을 표현하는 부분이다. ③은 차트의 제목을 표현하는 부분이다. ④는 막대 차트의 X축인 항목 축의 제목을 표현하는 부분이다. ⑤는 막대 차트의 Y축인 값 축의 데이터 제목을 나타내

는 부분이다. ⑥과 ⑦은 Y축의 최대값과 최소값을 표현하는 부분이다. 그림 3에서는 최대값을 250, 최소값을 0으로 표현하였다. ⑧은 각 항목을 표현하는 부분이다. ⑨는 범례 집합을 표현하는 부분이며, ⑩은 개별적인 범례 하나를 표현한다. ⑩에서 속성으로 범례 색상에 대한 정보를 갖는다. 이 색상 값이 차트에서 막대 그래프의 색상이 된다. ⑪은 범례의 이름을 표현하는 부분이고 ⑫는 해당 범례에서 각 항목에 대한 데이터를 기술한다. <value> 요소 안의 각 값들은 <item> 요소의 항목들에 대한 값이다.

```
<CML>
<text>      This is multi-table Test 1. </text> —①
<chart>
<charttype shape="vbar" planetype="1"/> —②
<chartinfo>
<header headerposition="up">  1-1 Home Appliance
</header> —③
<title><xtitle position="center">Item</xtitle> —④
      <ytitle position="center">Price</ytitle> —⑤
</title>
<axisinfo>
    <ymax>250</ymax> —⑥
    <ymin>0</ymin> —⑦
    <xaxis> —⑧
        <item>TV</item>
        <item>Aud</item>
        <item>Cam</item>
    </xaxis>
</axisinfo>
<legends> —⑨
    <legend color="Aqua"> —⑩
        <name> Sold </name> —⑪
        <values> —⑫
            <value>30</value>
            <value>25</value>
            <value>40</value>
        </values>
    </legend>
    <legend color="Aqua">
        ... ... ...
    </legend>
</legends>
</chartinfo>
</chart>
</CML>
```

(그림 3) CML 문서의 예

그림 4는 그림 3에서 작성한 CML 문서의 논리적인 구조를 보인 것이다. 그림 4에서 폴더 형태는 속성을 갖는 요소를 의미하며, 원 형태는 속성을 갖지 않는 요소를 의미한다. 각각의 요소들은 포함 관계를 갖는 구조로 되어 있다.

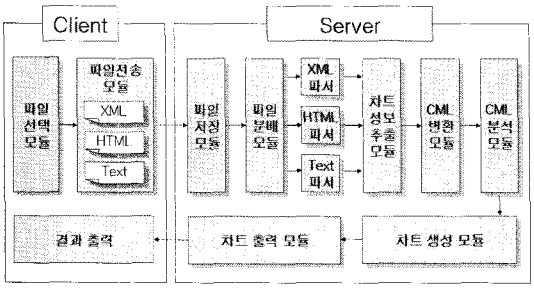


(그림 4) CML 문서의 논리적 구조

3. 차트출력시스템

3.1 시스템 구조

차트출력시스템은 크게 클라이언트 영역과 서버 영역으로 나뉘며, 차트출력시스템의 시스템 구조도는 그림 5와 같다.



(그림 5) 차트출력시스템 구조도

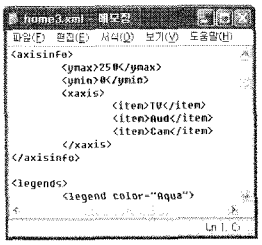
클라이언트는 차트정보가 존재하는 파일을 선택하여 서버로 전송하고, 서버로부터 받은 결과를 웹 브라우저에 출력하는 역할을 수행한다. 차트정보는 수치형 데이터가 행과 열의 이름 정보와 함께 테이블 형태로 표현되어 있는 것을 나타낸다. 클라이언트에 입력된 파일에서 차트정보만을 추출하여 차트로 변환한다. 클라이언트에서 서버로 파일을 전송할 때는 어떤 종류의 차트로 표현할 것인지 인터페이스에서 선택할 수 있으며, 선택하지 않으면 기본적으로 수직 막대 차트로 출력된다. 클라이언트에서 선택 가능한 파일의 종류는 XML, Text, HTML이다.

서버는 클라이언트로부터 받은 파일을 분석하여 차트정보를 추출하고, 그 정보를 CML로 변환하여 차트를 출력할 수 있는 그래프로 표현한다. 그리고 그 결과를 클라이언트로 전송한다. 서버는 파일 저장, 파일 분배, 파서, 차트정보 추출, CML 변환, CML 분석, 차트 생성 및 출력 모듈로 구성된다.

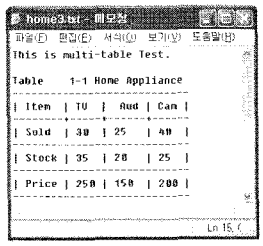
파일 저장 모듈은 클라이언트로부터 전송된 파일을 서버에 저장하고, 전송된 파일을 파일 분배 모듈에 전달한다. 파일 분배 모듈은 전송된 파일을 분석하여 각 파일에 해당하는 파서에게 파일을 전달해주는 모듈이다. 전송된 파일의 태그나 구조를 이용하여 파일의 종류를 파악하고, 분석이 종료된 파일을 파일의 형식에 맞는 파서에 전송한다. 파일 분배 모듈에서 처리할 수 있는 파일의 종류는 XML, Text, HTML 파일이고, 파서의 종류로는 XML 파서, Text 파서, HTML 파서가 있다.

차트출력시스템에서 활용하는 파서는 XML, Text, HTML 문서에서 차트정보가 나타난 부분을 검사 및 확인할 수 있어야 하므로 파일의 형식적 특성에 맞게 파서를 설계하였다. 각각의 파서는 전송된 파일에 대하여 파일의 특성에 맞게 파싱을 수행한다. 특히 차트정보가 출현하는 부분을 파서가 검사하므로 차트정보 추출 모듈은 빠르게 차트정보를 추출할 수 있다. XML, HTML 파일은 각각의 태그와 태그의 값에 대하여 파싱을 수행한다. 파싱은 자체적으로 제작한 파서를 사용한다.

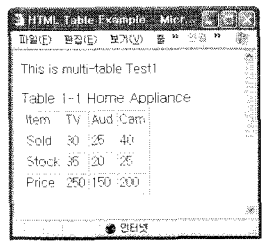
그림 6은 XML 파일, Text 파일, HTML 파일의 예제를 보인 것이다. 그림 6의 (a)는 XML 확장자를 갖는 파일을 예제로 보인 것이고, (b)는 Text 파일의 예제를 보인 것이다. Text 파서는 테이블의 형태를 분석하여 테이블의 정보를 추출한다. (c)는 HTML 파일의 테이블 예제를 보인 것이다. XML 또는 HTML 파일의 파서는 요소나 태그를 분석함으로써 테이블 정보를 추출하고 차트로 변환한다.



(a) XML 파일의 예



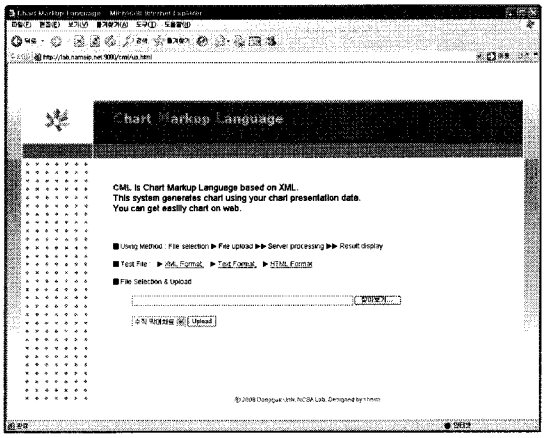
(b) Text 파일의 예



(c) HTML 파일의 예

(그림 6) 입력 파일의 예

차트정보 추출 모듈은 파싱된 XML, Text, HTML 정보에서 차트정보를 표현하는 부분을 검색하여 추출하는 모듈이다. 차트정보는 테이블 형태로 구성되므로 문서 내에서 테이블을 구성하는 요소들 또는 테이블 관련 태그들을 검사한다. 그리고 그 테이블 형태가 수치형 데이터, 행 이름 정보, 열 이름 정보 등 차트정보 형태를 만족하는가를 검사한다. Text 파일의 경우 문서에 나타나는 특정한 형식의 테이블 표현을 차트정보로 활용한다. 만약 테이블 표현을 만족하는 부분이 존재하면 그 영역을 텍스트 문서의 차트정보로 활용한다. 예를 들면 HTML 파일에서는 "table" 태그 부분을 추출 및 분석하여 차트정보로 활용한다. Text 파일에서는 표 시작 단어와 표의 테두리 선과 같은 테이블 구조를 이용하여 차트정보를 추출한다. 추출된 차트정보는 CML 변환 모듈에 전달된다. CML 변환 모듈은 각 파일에서 추출된 차트정보를 CML 형식의 문서로 변환하는 모듈이다. 차트정보에는 차트의 제목, 차트의 X축에 표현되는 항목들에 대한 정보, 범례, 데이터 값 등이 포함된다. 그리고 CML 문서는 차트정보를 구조적으로 표현하여 저장할 수 있도록 되어 있다. CML 변환 모듈은 추출한 정보를 CML 문서의 해당하는 차트정보 항목에 삽입하여 차트를 표현하는 CML 문서를 작성한다. CML 분석 모듈은 DOM을 이용하여 CML로 표현된 문서를 분석하고 DOM 트리를 생성한다. 그리고 DOM 트리의 CML 태그와 값 정보를 읽어서 차트 생성 모듈에 전달한다. 차트 생성 모듈은 CML 분석 모듈에서 생성된 차트정보를 이용하여 웹브라우저에서 차트를 출력할 수 있도록 그래픽으로 차트를 생성하는 모듈이다. 차트 출력 모듈은 차트 생성 모듈에서 생성한 차트의 그래픽 정보를 이용하여 웹브라우저에 차트를 출력하는 모듈이다. 그림 7은 차트출력시스템의 사용자 인터페이스이다. 클라이언트에서 차트 문서를 전송할 때는 차트정보를 어떤 차트로 표현할 것인지 결정하기 위해서 차트의 종류를 선택하도록 한다. 만약 종류를 선택하지 않는다면 수직 막대차트를 기본 차트로 사용한다.

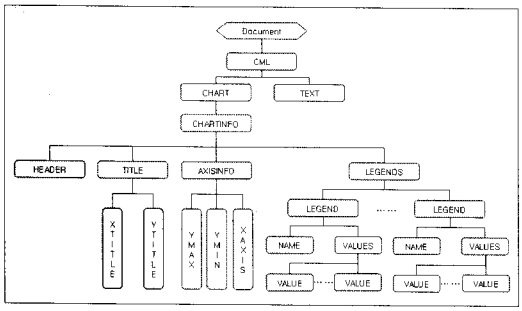


(그림 7) 차트출력시스템의 사용자 인터페이스

3.2 CML 분석 및 차트 생성

본 논문에서 제안한 차트출력시스템은 차트 생성을 위해 CML 문서를 분석하고 변환하는 과정에는 DOM을 이용한다. DOM은 HTML과 XML 문서를 다루기 위한 트리 기반 API로 플랫폼 및 언어 중립적인 인터페이스이다. DOM을 이용하면 XML 문서를 객체화하여 이를 DOM 트리 형태로 접근할 수 있다. CML 문서의 차트 표현 정보를 출력하기 위해서는 DOM을 이용하여 변환하는 과정이 필요하다. 변환 과정에서 CML 문서는 DOM 인터페이스를 통하여 DOM 라이브러리에 전달된다. DOM 라이브러리는 입력받은 CML 문서의 구조에 따라 DOM 트리를 생성한다. 차트정보를 VML로 표현할 때는 DOM 트리의 노드를 순회하며 노드에 포함된 차트의 내용을 VML로 표현한다.

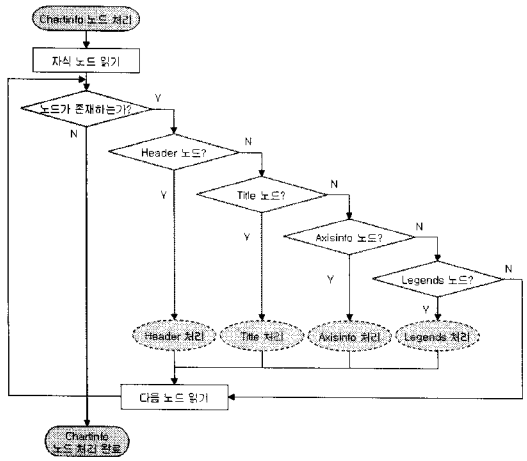
그림 8은 CML 문서를 DOM 스크립트를 통해 DOM 트리로 변환한 결과이다. DOM 트리는 CML 문서의 논리적 구조에 따라 계층 구조로 표현된다. 그림에서 원형 사각형은 노드를 나타내며, 육각형의 'Document'는 하나의 CML 문서 객체를 의미한다.



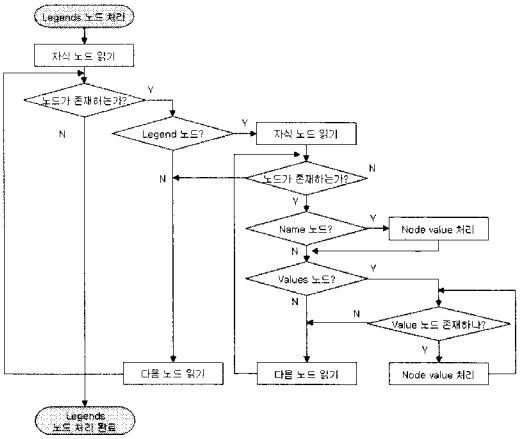
(그림 8) CML 문서에 대한 DOM 트리

그림 9는 DOM 트리를 순회하며, 각 노드에 대한 정보를 추출하는 작업 중에서 <chartinfo> 노드를 처리하기 위한 흐름도이다. <chartinfo> 노드를 처리할 때는 DOM 트리에서 <chartinfo> 노드의 자식 노드들을 순차적으로 읽으면서 해당하는 내용을 처리하도록 한다. 예를 들어 <header> 노드이면 "Header 처리" 부분을 수행하고, 더 이상 처리할 노드가 없으면 <chartinfo> 노드의 처리를 완료한다.

그림 10은 그림 9의 <chartinfo> 노드를 처리하기 위한 흐름도 중에서 "Legends 처리" 부분의 흐름도를 보인 것이다. <legends> 노드 처리에서는 자식 노드로 등록된 각 <legend> 노드에 대하여 <legend> 노드 처리를 수행한다. <legend> 노드 처리는 자식 노드들을 검사하며, <name> 노드로부터 범례의 이



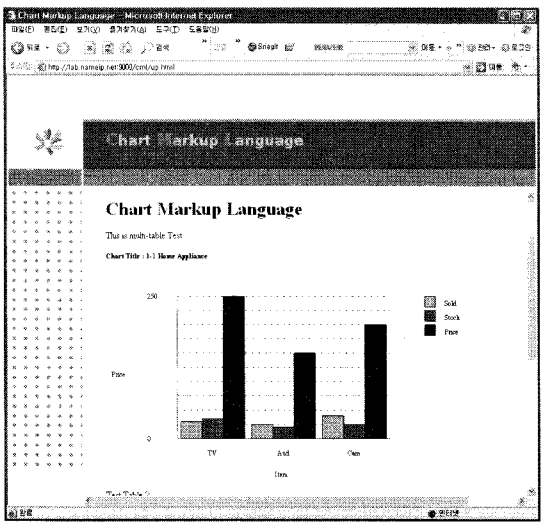
(그림 9) 〈Chartinfo〉 노드를 처리하기 위한 흐름도



(그림 10) 〈Legends〉 노드를 처리하기 위한 흐름도

름 정보를 추출한다. 그리고 <values> 노드를 만나면 <values> 노드의 자식으로 등록된 <value> 노드들의 값들을 추출한다. <value> 노드는 <item> 노드의 수와 동일해야 한다. 각 <legend> 노드에 대하여 처리를 마치면 <legends> 노드 처리를 종료한다.

본 논문에서 제안한 차트출력시스템에서 최종적으로 생성된 차트는 별도의 출력 프로그램 없이 웹 브라우저에서 출력할 수 있으며, 차트 데이터를 이용하여 생성된 차트를 웹브라우저에 출력한 결과를 그림 11에 나타내었다.



(그림 11) 차트를 출력한 결과

4. 결론 및 향후연구

차트는 정보를 시각화하기 때문에 정보전달 효과가 매우 높아 웹에서 높은 활용빈도를 나타내고 있다. 그러나 차트를 웹에서 직접 생성할 수 없기 때문에 사용자들은 차트 제작을 위한 전용 프로그램으로 차트를 생성해야 한다. 이와 같은 이유로 웹에서 차트를 활용할 경우에는 이미지로 변환하여 활용해야 하며, 이러한 경우에 대다수 활용되는 방식은 래스터 방식이다. 래스터 방식의 이미지는 크기가 변경될 경우 원상 이미지의 왜곡이 발생하고, 차트정보를 공유하기도 어렵다는 문제가 있다. 이러한 래스터 방식의 문제점을 해결하기 위해 벡터 방식이 활용되고 있으나, 대다수가 활용하기 어렵고 생성된 이미지를 출력하기 위해 전용 프로그램이 필요하다는 문제점이 있다.

본 논문에서 제안한 차트출력시스템을 활용하면 이러한 문제점을 해결하고 일반 사용자도 웹에서 차트를 쉽게 생성할 수 있다. 본 논문에서 제안한 차트출력시스템은 클라이언트에서 XML, Text, HTML 문서를 입력받아 문서분석을 수행하여 텍스트 기반 차트정보를 추출하고, 추출된 차트정보를 이용하여 웹상에 차트를 출력하는 구조로 이루어져 있다. 차트출력시스템의 구조는 크게 클라이언트 부분과 서버 부분으로 나뉜다. 클라이언트 부분은 파일을 선택하고 전송하는 역할을 담당하고, 서버 부분은 정보를 분석하여 차트정보를 추출하고 웹브라우저에 차트를 출력하는 역할을 담당한다. 본 논문에서 제안한 차트출력시스템은 웹에서 차트를 표현할 때 차트의 호환과 공유를 위해 CML을 정의하여 사용하였으며, CML은 차트정보를 표현하기 위해 만들어진 마크업언어로 XML 기반의 차트생성언어이다. 본 논문에서 제안한 CML은 XML을 기반으로 하고 범용성이 뛰어나며 변환이 용이하다는 장점이 있다. 또한 CML 문서 형태로 차트정보를 저장하고 있기 때문에 차트내의 정보 활용성이 높다는 장점이 있다.

본 논문에서 제안한 차트출력시스템은 벡터 방식으로 차트를 처리하기 때문에 이미지의 크기가 변경되더라도 이미지가 왜곡되지 않는다는 장점이 있으며, 이미지의 크기가 변경되더라도 해상도가 우수하고 파일의 크기가 작다는 장점이 있다. 또한 차트출력시스템은 별도의 전용 프로그램이나 호환기 없이도 웹브라우저에 차트를 출력할 수 있다는 장점이 있다. 본 논문에서 제안한 차트출력시스템은 웹 프로그래밍을 모르는 사용자들도 웹에서 차트를 생성할 수 있도록 도와주는 웹기반 차트출력시스템으로 활용할 수 있으며, 기존에 존재하는 텍스트 문서를 입력받아 웹에서 차트를 생성하는 웹기반 차트생성기로도 활용할 수 있다는 장점이 있다. 또한 본 논문에서 제안한 차트출력시스템은 텍스트 문서에서 차트정보를 추출하여 XML로 변환하는 기능을 가지고 있기 때문에 차트정보에 대한 XML 변환기로도 응용하여 활용할 수 있다는 장점이 있다.

향후연구로는 다양한 차트의 표현을 위해서 표현 가능한 차트 유형 범위의 확대가 필요하고, 차트간의 데이터 교환이 가능한 동적인 차트의 표현에 대한 연구가 필요하다. 또한 웹브라우저에서 사용자와의 상호작용을 통하여 직접 차트를 편집하고 수정할 수 있는 편집 환경에 대한 연구가 필요하다.

참 고 문 헌

[1] P. L. Thomas and D. F. Brailsford, "Enhancing Composite Digital Documents using XML-based Standoff Markup," Proceedings of the 2005 ACM symposium on Document engineering, 2005.

[2] S. Ronnau, J. Scheffczyk, and U. M. Borghoff, "Towards XML Version Control of Office Documents," Proceedings of the 2005 ACM symposium on Document engineering, 2005.

[3] C. Stinson and M. Dodge, Microsoft Office Excel 2003 : Inside Out, Microsoft Press, 2003.

[4] D. C. Hanselman and B. L. Littlefield, Mastering MATLAB 7, Prentice Hall, 2004.

[5] S. Abiteboul, P. Buneman, and D. Suciu, "Data on the Web," Morgan Kaufmann Publishers, San Francisco, CA, 2000.

[6] F. Bes and C. Roisin, "A Presentation Language for Controlling the Formatting Process in Multimedia Presentations," Proceedings of the 2002 ACM symposium on Document engineering, 2002.

[7] B. Mathews, "Vector Markup Language(VML)," http://www.w3.org/TR/1998/NOTE-VML-19980513, 1998.

[8] C. F. Goldfarb and P. Prescod, The XML Handbook, Prentice Hall, 1998.

[9] A. Zisman, "An Overview of XML," Journal of Computing & Control Engineering, 2000.

[10] S. Holxner, XML Complete, McGraw-Hill, 1998.

[11] I. Zaslavsky, "A New Technology for Interactive Online Mapping with Vector Markup and XML," Cartographic Perspectives, 2000.

[12] N. Al-Shamma, R. Ayers, R. Cohn, J. Ferraiolo, M. Newell, R. K. deBry, K. McCluskey, and J. Evans, "Precision Graphics Markup Language(PGML)," http://www.w3.org/TR/NOTE-PGML-19980410, 1998.

[13] P. Topping, "Mathematics on the Web : Math ML and MathType," http://www.mathtype.com/features/white_papers/mt_mathml. html, January 21, 1999.

[14] R. H. Landau, D. Vediner, P. Wattanakasiwich, and K. R. Kyle, "Future Scientific Digital Documents with MathML, XML, and SVG," Computing in Science and Engineering, 2002.

[15] J. C. Mong and D. F. Brailsford, "Using SVG as the Rendering Model for Structured and Graphically Complex Web Material," Proceedings of the 2003 ACM symposium on Document engineering, 2003.

[16] S. Widergren, A. deVos, and Z. Jun, "XML for Data Exchange," Power Engineering Society Summer Meeting, IEEE, 1999.

[17] L. Wood and V. Apparao, "Document Object Model(DOM) Level 1 Specification," http://www.w3.org/TR-REC DOM Level-1, 1999.

[18] A. L. Hors, P. L. Hegaret, L. Wood, G. Licol, J. Robie, M. Champion, and S. Byrne, "Document Object Model (DOM) Level 3 Core Specification," http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/, 2004.

[19] J. McKeown and J. Grimson, "SVG; Putting XML in the Picture Introduction," In Proceedings of XML Europe 2000, 2000.

[20] M. S. Sagar, "An SVG Browser for XML Languages," In Proceedings of Theory and Practice of Computer Graphics, 2003.

[21] G. J. Bex, F. Neven, and J. V. Bussche, "DTDs versus XML Schema: A Practical Study," Proceedings of the 7th International Workshop on the Web and Databases, 2004.

◐ 저 자 소 개 ◑

윤 현 님(Hyun-Nim Yoon)
1996년 상명대학교 수학교육학과 졸업
2000년 동국대학교 교육대학원 컴퓨터교육학과 졸업
2001~현재 동국대학교 대학원 정보통신공학과 박사과정
2001~현재 한국폴리텍여자대학 디지털정보과 교수
관심분야 : 가상교육, XML, 분산 그리드 컴퓨팅 시스템
E-mail : yhnim@dongguk.edu

김 양 우(Yang-Woo Kim)
1984년 연세대학교 전자공학과(공학사)
1986년 Syracuse Univ. 컴퓨터공학전공(공학석사)
1992년 Syracuse Univ. 컴퓨터공학전공(공학박사)
1992년~1996년 한국전자통신연구원 선임연구원
1996년~현재 동국대학교 정보통신공학과 교수
관심분야 : 분산 그리드 컴퓨팅 시스템, 컴퓨터구조, 가상교육
E-mail : ywkim@dongguk.edu