XML 기반의 구조화된 그래픽 표현을 위한 SVG 문서 저작 시스템

배희재[O], 송병규, 김윤기, 김창수, 정회경
배재대학교 컴퓨터공학과
{lookout[O], spk77,ddoja69,hkjung}@mail.pcu.ac.kr

SVG document editing system based on XML for structured graphic representation

HeeJae Bae[0] PyungKyu Song, YoonGi Kim, ChangSu Kim, HoeKyung Jung
Dept. of Computer Engineering, PaiChai Univ

요 약

인터넷의 급속한 발전은 기존의 정적인 웹을 탈피하여 사용자들로 하여금 더욱더 동적이고 다양한 콘텐츠를 요구하는 형태로 바뀌어 가는 실정이다. 따라서 이러한 사용자들의 욕구를 충족시키기 위해 인터넷 상에서 표현의 고급화와 체계적이고 구조적인 방식의 콘텐츠를 생성하고 관리하기 위한 방법들이 논의됨에 따라 이를 충족시키기 위한 방법으로 W3C(World Wide Web Consortium)에서는 SVG(Scalable Vector Graphics)를 제안하였다. SVG는 기존의 인터넷에서 사용되던 비트맵 기반의 디스플레이 보다 훨씬 풍부하고 정교한 그래픽 표현을 제공하기 때문에 기능이나 장치 호환성의 희생 없이 완전하게 벡터 그래픽을 표현한다. 또한, 그래픽에 대한 논리적인 구조를 기술함으로써 인덱싱, 검색, 저장 또는 공유가 가능하도록 정의하고 있다.

이에 본 논문에서는 SVG에 관한 기초기술 연구 및 구조화된 SVG 문서를 사용자 중심의 편집 인터페이스를 통해 일반 사용자들이 손쉽게 그래픽 객체를 직접 저작함에 따라 복잡한 SVG 구문을 자동으로 생성하는 SVG 문서 저작 시스템을 설계 및 구현한다.

1. 서 론

오늘날 인터넷은 더욱 진보된 기술들 중 하나가 바로 인터넷 상에서의 그래픽 표현에 관한 것이라 할 수 있다. 기존의 인터넷에서는 그래픽에 대한 표현의 고급화와 보편성을 제공하는 방식을 주축으로 이루어져 왔으며, 이들을 처리하기 그래픽 저작 시스템들이 나날이 개발되어왔다. 하지만, 이러한 시스템들로 작성된 그래픽 문서는 표현의 고급화에 중점을 둔 나머지 정보의 교환과 공유라는 인터넷 기반에서의 그래픽 처리에 관한 모티브(Motive)를 고려하지 않은 채 각기 독자적인 구조로 인해 상호 호환성이나 재사용에 따른 비용 낭비 차원의 여러 문제점을 내포하고 있는 실정이다.

이에 따라 W3C에서는 인터넷 상에서 벡터 그래픽 표현의 효율적인 처리와 저장 및 공유를 가능하게 하는 SVG를 제정하였다[1,2,3,4]. SVG는 기존의 인터넷에서 사용되던 비트맵 기반의 디스플레이 보다 훨씬 풍부하고 상호작용(Interactive) 가능한 그래픽 표현을 제공하기 때문에 기능이나 장치 호환성의 희생 없이도 완전하게 벡터 그래픽을 표현하며, 정교한 구조와 비주얼한 제어와 함께 실시간 데이터로부터 고품질의 그래픽을 동적으로 창조할 수 있도록 해준다. 또한, 그래픽에 대한 논리적인 구조를 기술함으로써 인덱싱, 검색, 교환 또는 공유가 가능하도록 정의하고 있다.

이에 본 논문에서는 SVG 1.1 권고안에 대한 기초 기술 연구 및 인터넷 상에서 사용되는 그래픽 처리를 위해 일반 사용자들이 손쉽게 그래픽 객체를 생성하고 편집할 수 있도록 그래픽 저작 시스템에 관한 구조를 정의하였다. 또한 저작된 각 그래픽 객체들을 통해 SVG 문서를 생성하는 코드 변환기를 정의하여 벡터 그래픽을 XML 기반의 논리구조로 변환하는 SVG 문서 저작 시스템을 설계 및 구현한다.

2. 시스템 설계

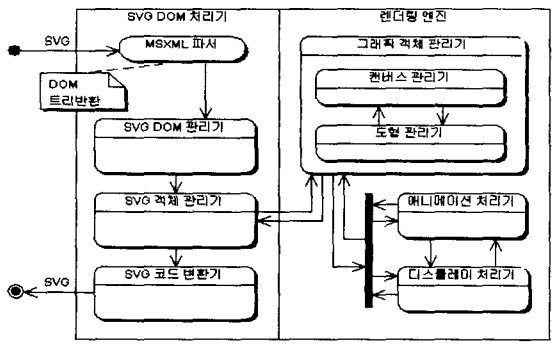

그림 1 전체 시스템 구성도

그림 1은 본 시스템의 전체적인 구성도이다. 본 시스템은 W3C에서 제안한 XML을 기반으로 작성된 SVG 문서를 입력으로 문서의 논리 구조를 처리하기 위한 SVG DOM[5] 처리기와 각각의 그래픽 객체를 사용자 뷰포트에 랜더링 시키기 위한 랜더링 엔진(Rendering Engine)으로 나뉘어 진다.[6]

2.1 SVG DOM 처리기 설계

입력된 SVG 문서는 파서 API를 통해 파싱한 결과를 메모리에 DOM 트리 형태로 저장한다. 저장된 DOM 객체는 SVG 객체 관리기를 통해 각 SVG 논리 구조에 해당하는 기본 정보를 수집하고 관리하게 된다. 객체 관리기에 수집된 정보들은 랜더링 엔진의 데이터로 입력되는 자료가 된다.

2.1.1 SVG DOM 관리

SVG의 모든 객체들은 계층적 구성을 가지고 있어 객체들에 접근하기 위한 방법으로 기존의 XML DOM을 사용할 수 있지만, SVG의 논리적인 정보를 이해하기 위해서는 기존의 DOM 만으로는 처리하기가 어려움이 있다. 따라서 본 시스템에서는 SVG 문서의 논리적인 구조상에서 필요한 정보를 추출 및 저장, 공유하기 위한 기반이 되도록 SVG DOM 관리기를 설계하였다.

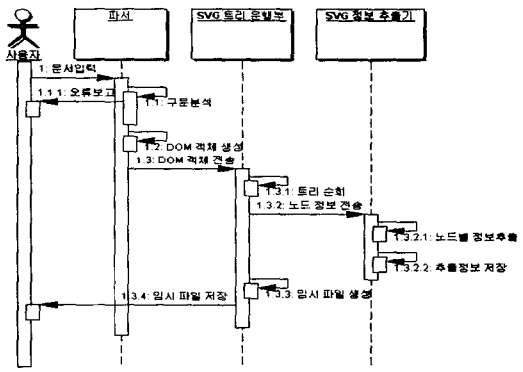

그림 2 SVG DOM 관리기 데이터 처리 과정 순차 흐름도

SVG DOM 객체 트리 운행은 입력된 DOM 트리에 대해 깊이 우선 탐색(Depth First Search : DFS)으로 처리하며 각 SVG 객체의 특징에 따라 분류 후 SVG 객체 관리기를 호출한다. SVG DOM 트리의 탐색 방법인 깊이 우선 탐색은 일반적 트리의 탐색방법에 주로 사용되며, SVG의 구조 관리에 적합한 알고리즘이다.

SVG 트리 운행부로부터 전달받은 노드 정보들은 그래픽 객체로의 변환을 위해 SVG 정보 추출기에서 각 그래픽 객체 생성을 위한 논리 정보를 추출한다. 추출된 정보들은 랜더링 엔진과의 상호작용을 위해 SVG 객체 관리기에 의해 직렬화 시킨 후 연결 리스트에 저장된다.

2.1.2 SVG 객체 관리

SVG DOM 관리기로부터 전달받은 각각의 DOM 노드 정보들은 랜더링 엔진과의 상호작용을 위해 그래픽 객체를 위한 논리 정보로 변환되는 과정을 거쳐야 한다. 논리 정보라 함은 각 그래픽 객체의 프로퍼티에 해당하는 정보를 말한다. 예를 들어 사각형 객체의 경우 SVG 노드 정보의 x, y 좌표(시작위치)와 길이, 높이, 스타일 정보, 식별자 등을 사각형 객체의 논리 정보로 취급한다.

2.2 랜더링 엔진 설계

입력된 SVG 문서에 대한 구문 분석 및 필요한 노드별 데이터 수집이 SVG DOM 처리기를 통해 완료되면 수집된 데이터를 기반으로 사용자들에게 그래픽 저작에 필요한 인터페이스를 제공하기 위해 랜더링 엔진을 설계하였다.

2.2.1 그래픽 객체 관리

입력된 SVG 문서에 대해 SVG DOM 처리기에서 필요한 정보의 수집이 완료되면 랜더링 엔진의 그래픽 객체 관리기에 수집된 정보를 저장한다. 이들 정보를 토대로 그래픽 객체 관리기는 캔버스 관리기와 도형 관리기를 호출함으로써 사용자의 뷰에 랜더링되기 위한 사전 작업을 완료하게 된다.

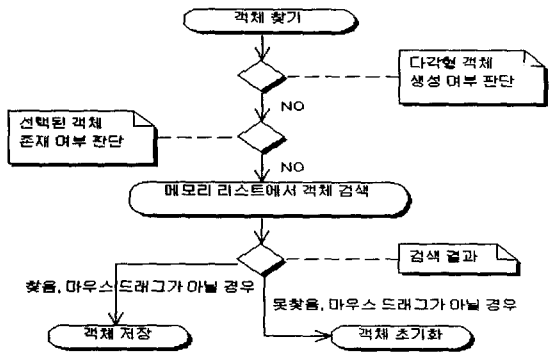

그림 3 객체 검색 과정

객체를 검색하기 위해서는 그림 3과 같이 마우스 이벤트에 의해 메모리 리스트에서 해당 도형 객체를 검색하고 검색 결과가 참이면, 객체 타입에 해당하는 툴을 반환한다. 만약 거짓이면, 새로 생성될 객체로 판단하고 메모리 리스트에 해당 툴을 추가하여 도형 객체를 생성한다. 객체를 검색하기 전에 객체 타입이 다각형이나 패스일 경우에는 검색을 중단하고 다각형 좌표를 메모리 배열에 저장하는 루틴으로 과정을 전환한다.

마우스 이벤트에 따른 객체 이동은 다른 객체들과는 달리 다각형의 객체 이동은 다른 의미를 가진다. 일반적인 객체의 이동은 위치 이동을 하지만 다각형 객체의 이동은 그림 4와 같이 ②를 생성하기 위한 ①의 과정을 의미한다.

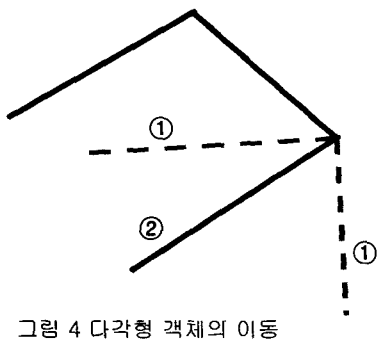

그림 4 다각형 객체의 이동

따라서 다각형의 이동은 단순히 위치 이동만을 하는 것이 아니라 이동하기 위한 기준 객체를 삭제하고 새로운 객체를 다각형의 메모리 배열에 추가하는 과정을 반복해야 한다.

2.2.2 애니메이션 처리

애니메이션 처리기는 그래픽 객체 관리기에 의해 관리된 정적인 도형 객체를 동적인 객체로 랜더링 시키기 위한 모듈로 타임 테이블(Time Table) 형태의 사용자 인터페이스를 가지며 타임 라인(Time Line)을 이용해 애니메이션 시간을 지정하도록 한다. 사용자로부터 선택된 그래픽 객체는 SVG 애니메이션을 위해 타임 라인에 객체의 애니메이션 타임을 설정하고 타임 라인 컨트롤을 변경시킨다. 변경된 정보는 클래스에서 애니메이션 규칙을 설정하고 변경된 내용을 토대로 SVG DOM 처리기의 해당 노드 정보를 변경한다

2.2.3 디스플레이 처리

그래픽 객체 관리기와 애니메이션 처리기에서 설정 및 변경된 정보를 토대로 SVG 그래픽을 랜더링 시키기 위해 디스플레이 처리기는 메모리 리스트에 저장된 모든 정보를 사용자 뷰포트에 연속적으로 랜더링 시킨다. 랜더링 순서는 메모리 리스트에 저장된 순서를 따르며 사용자 편집에 따라 랜더링 순서가 바뀌면 해당 객체의 리스트 위치를 이동(Shift) 시킨다

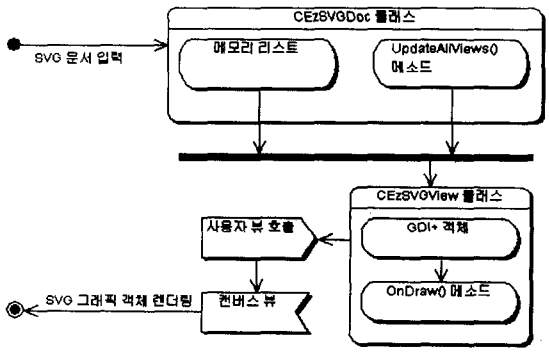

그림 5 디스플레이 처리기 처리 흐름도

그림 5는 디스플레이 처리기가 메모리 리스트에 저장된 그래픽 객체들을 GDI+ 객체를 이용하여 사용자 뷰포트에 랜더링 시키기 위한 처리 흐름을 보인다. .

3. 시스템 구현 및 고찰

그림 6은 시스템의 화면 구성을 보여준다. 전체 화면은 일반적인 그래픽 툴의 형태로 이루어지며, 자식(Child) 윈도우의 중앙에 SVG 캔버스를 위치시킴으로써 저작되는 그래픽 객체들을 한눈에 볼 수 있도록 인터페이스를 사용자 중심적으로 지향하였다. 특정 객체에 애니메이션이 추가될 경우 화면의 하단에 위치한 애니메이션 컨트롤에 등록되어 사용자가 쉽게 SVG 그래픽 객체에 애니메이션을 추가하도록 구현하였다.

SVG 미리 보기 창과 원문 보기 창은 기존의 다른 시스템과는 달리 지저분한 윈도우들을 줄이기 위해 팝업(Popup) 다이얼로그로 구현하여 사용자가 확인하고자 할 때만 보여질 수 있도록 하였다.

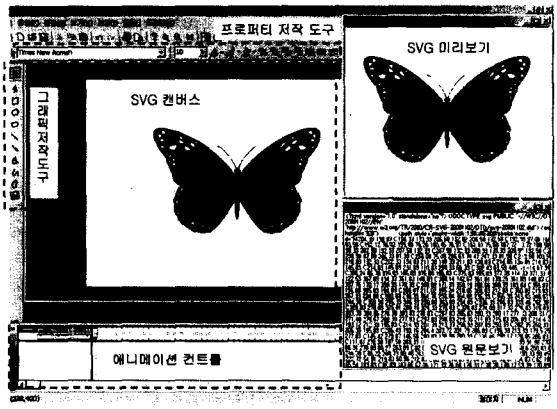

그림 6 SVG 저작 시스템의 화면 구성

4. 결 론

최근 인터넷에서 고급화된 그래픽 표현 기술 발전의 가속화에 따라 W3C에서는 구조적으로 벡터그래픽 처리를 하기 위한 SVG를 제정하였다.

본 논문에서는 SVG 1.1 문서의 입력에 따라 문서의 논리 구조를 이해하고 처리하기 위한 SVG DOM 처리기와, 생성된 SVG 그래픽 객체를 랜더링 시키고, 저작된 그래픽 객체들을 SVG 구문으로 변환해주는 SVG 문서 저작 시스템을 설계 및 구현하였다.

본 시스템의 장점으로는 웹 표준화 기구인 W3C에서 제안하는 SVG 1.1의 표현 지침을 따르고 있어 표준화에 입각한 처리 시스템이라는 점이고, 결과적으로 표준화의 변화에 빠른 대처가 가능하다. 또한 시스템의 대부분을 모듈화하여 부분적인 수정 및 대체와 이식이 가능하다는 장점이 있다.

향후 권고안으로 채택된 SVG Mobile Profile을 수용하기 위한 모바일(Mobile) 기술로의 전환이 될 수 있도록 연구해야 할 것이다.

참고문헌

[1]W3C, Scalable Vector Graphics (SVG) Version 1.1, http://www.w3.org/TR/SVG11, Jan. 14, 2003
[2]W3C, Mobile SVG Profiles: SVG Tiny and SVG Basic, http://www.w3.org/TR/SVGMobile/, Jan. 14, 2003
[3]나방현, 심규찬, 이종연 공저, "XML 그래픽 입문", 21세기사
[4]J. David Eisenberg, "SVG Essentials", February 2002, O'Reilly & Associates
[5]W3C, DOM, http://www.w3.org/DOM
[6]W3C, Synchronized Multimedia Integration Language Version 2.0, http://www.w3.org/TR/smil20, Aug. 07, 2001