XML 기반 다이어그램 시스템

김 성 근[†] · 김 영 철[††] · 윤 태 희[†††] · 유 재 우[††††]

요 약

XML 문서와 관련된 기존의 다이어그램 시스템은 대부분 특정한 목적으로 이용할 수 있도록 작성되어 있다. 또한 다이어그램 컴포넌트가 정의되어 있지 않기 때문에 자동적으로 DML 문서를 생성하기가 어렵다. 본 논문에서는 XML 문서가 실행될 수 있는 다이어그램 시스템을 설계하고 구현한다. 다이어그램 시스템은 위지웍(WYSIWYG) 개념을 이용하여 다이어그램 컴포넌트를 정의하며, 자동적으로 DML 문서를 생성할 수 있도록 설계되었다. 따라서 더욱 효율적으로 새로운 다이어그램을 개발할 수 있으며, DTD를 이용해 다이어그램에 대한 문법을 정의함으로써 XML DTD에 대한 의미적 일관성을 유지하였다. 또한 본 시스템은 다이어그램 문장에 대한 문법검사와 의미실행을 위하여 VPL(Visual Programming Language) 개념을 이용하였다. 본 시스템을 활용하면, DML 문서를 쉽게 생성할 수 있고, XML DTD 기반의 문법검사와 의미실행을 할 수 있다.

키워드 : XML, 다이어그램시스템, DML, VPL

A Diagram System based on XML

Kim Sung-Keun[†] · Kim Young-Chul[††] · Youn Tae Hee[†††] · Yoo Chae-Woo[††††]

ABSTRACT

Generally, Diagram Systems related XML document are designed for certain purpose. It is also difficult to create DML document, because there is no definition of diagram component. In this paper, we design and implement the diagram system to execute the XML document. This diagram system defines the diagram component with WYSIWIG concept ,and it is designed to generate DML document automatically. Therefore, it is possible to develop diagram efficiently and maintain consistency by definition of syntax about diagram with DTD. And this system uses the concept of VPL(Visual Programming Language) to check syntax and semantic about diagram sentence. Though this system, DML documents can be generated easily, and it can also check syntax and perform semantic.

Key Words : XML, Diagram System, DML, VPL

1. 서 론

현재의 향상된 컴퓨팅 환경에서 다이어그램은 많은 분야에 널리 사용되고 있으며 점차 중요성도 높아지고 있다. 다이어그램은 표현형식을 간략화 할 수 있고 의미전달을 명확히 할 수 있는 특징이 있어 사용자의 이해를 돕고, 그룹 성원들 간의 의사소통을 원활히 하는 장점이 있다. 이 때문에 다이어그램은 단순한 문서화에서부터 UML[1]과 같은 시스템 분석 및 설계의 도구로 사용되고 있으며 프로그래밍 도구로도 사용되고 있다. 또한, 시각 프로그래밍 환경에서나 새로운 모델링을 위한 소프트웨어를 개발할 때 다이어그램은 필수적인 요소이다. 이러한 소프트웨어 시스템에서 사용되는 다이어그램들은 응용분야에 맞는 규칙과 의미를 가지고 있으며, 이러한 요구 사항들을 만족시키기 위해서 다이어그램은 단순한 GIF나 JPEG 등의 이미지가 아닌 프로그램으로 작성되어야 한다. 다이어그램을 프로그램으로 구현하는 것은 많은 시간과 노력을 필요로 하는 작업이기 때문에 자동화된 방법이 필요하다. 또한 기존 다이어그램 소프트웨어들은 서로 다른 방법으로 구현되어서 소프트웨어들 간의 다이어그램 컴포넌트 공유가 불가능하다. 이러한 문제점을 해결하고자 XML을 이용하여 그래픽 정보를 나타내는 방법들이 연구되고 있다.[2, 3]

XML(eXtensible Markup Language)[4]는 구조적 문서를 표현하기 위한 표준화된 문서규격으로 최소의 단위는 원소(element)이고 이러한 원소들의 내부에 다른 원소를 포함하는 형태를 갖는다. XML은 확장성, 이식성 등의 특징 때문에 문서의 표현, SOAP[5], XML Query[6] 등의 프로그래밍 언어로 사용되며 VML[3], CML[7], MathML[8] 등의 특수한 목적으로 사용되고 있다.

※ 본 연구는 숭실대학교 교내연구비 지원으로 이루어졌음.
† 정 회 원 : 가톨릭상지대학 컴퓨터정보계열 조교수
†† 정 회 원 : 숭실대 정보미디어기술연구소
††† 정 회 원 : (주)로커스 테크놀로지스
†††† 정 회 원 : 숭실대학교 컴퓨터학부 교수
논문접수 : 2004년 8월 18일, 심사완료 : 2005년 4월 18일

다이어그램 컴포넌트는 정적인 형태 정보뿐만 아니라 동적인 행위 정보를 갖고 있다. 즉, 화면에서의 위치, 크기 또는 색깔 등의 정적인 정보뿐만 아니라 사용자의 속성 변경에 대한 제약 사항이나, 다이어그램 컴포넌트 자체의 의미와 같은 동적인 정보를 갖기도 한다. 그러나 기존의 XML을 이용한 그래픽 표현은 정적인 형태 정보만을 표현하기 때문에 동적인 객체의 특성을 갖는 다이어그램 컴포넌트를 표현하기에는 부족하며, 행위 속성을 정의할 수 있는 새로운 방법을 필요로 한다. 또한, 다이어그램 실행환경에서 다이어그램 컴포넌트들을 연결해서 사용할 경우 다이어그램 컴포넌트는 자신이 다른 컴포넌트와 연결될 수 있는지에 대한 정보를 가지고 있어야 한다. 즉, 잘못된 다이어그램을 그렸을 경우 사용자에게 잘못 그려진 다이어그램이라는 정보와 어느 부분이 잘못 그려진 것인지에 대한 힌트를 제공하여야 한다. 따라서 본 논문에서는 이러한 요구사항에 맞도록 XML을 이용하여 다이어그램 컴포넌트를 정의하고 다이어그램 편집기를 통하여 DML(Diagram Markup Language)을 생성해 낼 수 있는 다이어그램 시스템을 설계하고 구현하였다. DML은 다이어그램 컴포넌트의 형태정보(shape), 의미정보(semantics), 제약사항(constraints) 그리고 연결정보(connections)를 위한 원소로 구성된 마크업 언어로서, 새로운 다이어그램 개발 환경을 만들기 위해 필요한 언어이다. 본 논문에서는 DML 그래픽 편집기로 새로운 다이어그램 컴포넌트를 정의하고 편집기는 DML 문서를 자동으로 생성한다. 자동으로 생성된 DML 문서는 소스코드 변환기를 통해 Java 또는 C# 등의 언어로 변환되고 다이어그램 시스템에 등록되어 사용된다.

본 논문의 구성은 다음과 같다. 제2절에서는 관련 연구에 대해 기술하였고, 제3절에서는 다이어그램 시스템 모델에 대해서 제시하였다. 또한 제4절에서는 본 시스템의 실험 및 평가에 대해서 기술하였다. 마지막으로 제5절에서는 결론에 대해서 기술하였다.

2. 관련 연구

XML 다이어그램과 관련된 연구는 현재까지 많이 존재한다. 그러나 다이어그램에 관련되어 있는 연구는 자신의 응용프로그램에 적합하도록 만들어져 있다. 따라서 기존에 존재하는 XML 다이어그램 응용프로그램을 활용하는 것은 새로 개발하는 만큼이나 어렵다. 현재 XML 다이어그램 시스템과 관련되어 있는 연구는 대표적으로 SVG[2], VML[3], XGMML[9], DiaGen[10]이 대표적이다.

SVG(Scalable Vector Graphics)[2]는 2차원의 그래픽을 XML로 묘사할 수 있는 규격으로, 세 가지 그래픽 객체 즉 모양(shapes), 이미지(images), 텍스트(text)로 구성되어 있으며, 객체들은 서로 그룹으로 묶을 수 있고 스타일을 지정할 수도 있다. 또한 간단한 움직임을 표현하는 것뿐만 아니라 3차원 모핑(Morphing) 효과까지 줄 수 있다. SVG는 그래픽을 표현하기 위한 많은 기능들을 제공하기 때문에 웹 개발자에게 동적인 웹 페이지 개발할 수 있는 장점을 가지고 있다. 또한 SVG에서는 자바스크립트, 임베디드 SVG 등을 이용하여 애니메이션을 생성할 수 있으며, XML DOM을 이용하여 그래픽 객체의 모든 요소 및 속성에 쉽게 접근하고 조작할 수 있다.

VML(Vector Markup Language)[3]은 인터넷 익스플로러에 구현된 벡터 그래픽 구현 기술로 XML에 기반을 두고 있다. 즉, 벡터 그래픽을 그리기 위해 정의된 XML 태그를 이용해 간단하게 웹페이지에 원하는 그래픽을 그릴 수 있도록 지원한다. 따라서 간단한 벡터 그래픽을 보여주기 위해 VML을 사용하면 이미지 파일을 받을 필요 없이 텍스트 형태의 XML을 사용하므로 HTML 파일 크기를 줄일 수 있는 장점이 있다. 또한 인터넷 익스플로러에서 DHTML 코드를 이용해 VML 코드를 직접 다루므로 동적인 그래픽 효과를 줄 수 있다.

XGMML(Extensible Graph Markup and Modeling Language)[9]은 그래프를 표현하기 위하여 GML을 기반으로 하는 응용프로그램으로 원소가 그래프의 노드(node)와 에지(edge)를 나타내며, 서로 다른 그래프 저작도구나 탐색 도구들 간의 그래프 교환을 목적으로 WWWPAL시스템에서 웹사이트를 표현하기 위해 개발되었다.

DiaGen(Diagram Editor Generator)[10]은 다이어그램 컴포넌트를 정의하는 방법, 재구성(Reducing) 규칙, 문법 및 의미를 기술하여 자동으로 다이어그램 편집기를 생성한다. 또한 다이어그램 해석 과정은 크게 어휘분석 단계, 재구성 단계, 구문분석 단계, 의미분석 단계로 이루어져 있다. 어휘분석 단계(Scanning Step)는 다이어그램에서 관계성을 파악하여 SRHG(Spatial Relationship Hyper Graph)를 생성한다. 또한 재구성 단계(Reducing Step)에서는 파서가 읽을 수 있도록 SRHG를 패턴 매칭과 그래프 재작성 기법을 사용하여 하이퍼그래프(hypergraph)[11]로 변형한다. 하이퍼그래프는 유한한 개수의 노드와 하이퍼간선(hyperedge)을 갖는 방향성 그래프의 일반화된 자료구조로, 각 하이퍼간선은 이름과 노드와 연결될 수 있는 여러 개의 촉수(tentacle)를 갖고 각 하이퍼간선의 촉수는 노드를 통해 서로 연결된다. 구분 분석 단계(Parsing Step)는 이전 단계에서 생성된 하이퍼그래프가 다이어그램 언어를 정의한 하이퍼그래프 문법에 적합한지 여부를 검사하고 ASG(Abstract Syntax Graph) 생성하며, 의미 해석 단계(Semantic Analysis Step)에서는 구문분석 단계에서 생성된 ASG를 사용하여 의미를 해석한다. DiaGen은 하나의 다이어그램 정의를 위해 여러 개의 복잡한 하이퍼그래프 문법을 사용함으로써 이해와 구현하기 어렵고 확장성이 낮은 단점이 있다.

이외에도 XML 편집기와 관련된 연구는 CLIP! XML Editor[12], XML Pro v2.01[13], EXml[14], xmlspy® 2004[15] 등이 있다.

Techno2000에서 개발한 CLIP! XML Editor[12]는 트리 기반, 텍스트 기반의 편집을 지원하며 DTD 트리 보기 기능, 에러 출력 및 수정 기능을 제공하며 잘 설계된(well-for-

med) XML 문서에서 DTD를 추출, 생성해 준다. 또한 다른 XML 문서들의 엘리먼트를 불러와 재사용할 수 있어 다량의 XML 문서 저작 시 문서 작성 시간을 최소화할 수 있다.

Vervet Logic에서 개발한 XML Pro v2.01[13]은 순수 자바 애플리케이션으로 쉽고 직관적인 그래픽 사용자 인터페이스(GUL), 문서 구조를 유지하는 문서 트리 편집 뷰(View), DTD를 통한 XML 유효화 검증을 지원한다.

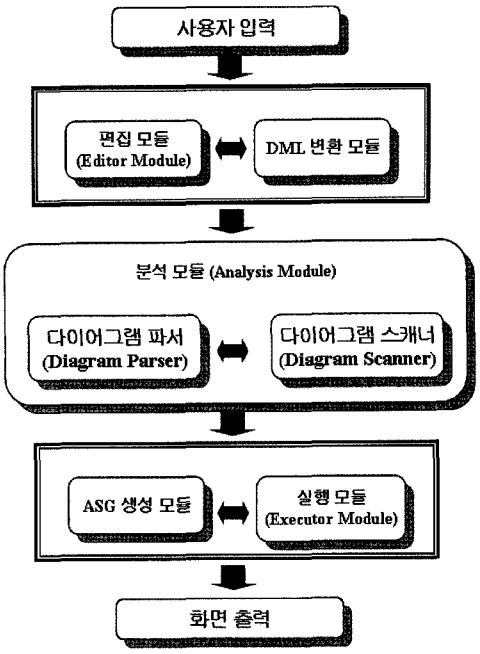
Cuesoft의 EXml[14]은 well-formed XML 문서 편집기로서 트리나 소스 뷰를 지원한다. 트리 뷰를 통해서 엘리먼트 편집이 가능하다. 다른 편집기와 마찬가지로 잘 구성된 문서(well-formed document)를 체크하는 것 이외에 특이한 사항은 XSL 이름공간(namespace)지원과 PCDATA, 주석, 속성값을 엘리먼트에 매핑한 테이블 출력, 소스뷰 상에서의 직접접적인 텍스트 편집이 가능하다는 것이다.

Altova의 xmlspy® 2004[15]는 유효한 XML 문서를 쉽게 만들 수 있도록 템플릿과 상황에 대응하는 (context-sensitive) 편집, 필요한 엘리먼트와 속성을 자동으로 채우는 기능(auto-completion)을 지원한다. 또한 현재 위치에서 어떤 엘리먼트를 사용할 수 있는지 알려주는 엔트리 헬퍼(Entry Helper)를 지원을 통한 직관적인 사용자 인터페이스를 제공하고 DTD 편집기, XSL 편집 및 디버깅 기능 등을 지원한다.

XML 다이어그램은 직관성과 간결성을 갖는 장점이 있기 때문에 현재의 컴퓨팅 환경에서 널리 사용되고 있다. 하지만 다이어그램 작성에 대한 표준화된 방법의 부재(不在)로 소프트웨어 상호간의 자료공유가 어렵고, 다이어그램 컴포넌트와 규칙을 직접 프로그래밍 해야 하기 때문에 많은 시간과 노력을 필요로 한다. 이러한 문제점을 해결하고자 본 논문에서는 표준화된 문서규약인 XML을 이용해 다이어그램 컴포넌트의 형태와 행위를 정의하는 방법, 다이어그램의 규칙과 의미를 기술하는 방법에 대해 제안하고 제안한 XML 문서가 실행될 수 있는 다이어그램 시스템의 설계 및 구현에 관하여 논의한다.

3. 다이어그램 시스템 모델

본 논문에서 개발한 다이어그램 시스템은 DML 그래픽 편집기를 제공하여 위지윅(WYSIWYG : What You See Is What You Get)방식으로 다이어그램 컴포넌트를 정의하고 자동으로 DML 문서를 생성할 수 있도록 함으로써 더욱 효율적으로 새로운 다이어그램을 개발할 수 있도록 하였다. 뿐만 아니라 DTD를 이용해 다이어그램에 대한 문법을 정의함으로써 문서의 구조를 정의하는 DTD에 대한 의미적 일관성을 유지하였으며 의미정의(Semantic Definition) XML을 이용하여 의미를 기술할 수 있도록 하였다. 또한 다이어그램 시스템에서 다이어그램 문장에 대한 문법검사와 의미실행의 방법은 VPL(Visual Programming Language)의 여러 개념들을 이용하였다. 다음 그림 1은 본 논문에서 제시한 다이어그램 시스템 전체 모델이다. (그림 1)과 같이 본



(그림 1) XML 기반 다이어그램 시스템

시스템은 향후에 확장성을 고려하여 모듈단위로 구분하여 설계하였다.

3.1 편집 모듈

편집 모듈은 DML 다이어그램을 편집할 수 있는 모듈로 이루어진다. 편집 모듈에서는 등록된 다이어그램을 보관할 수 있는 목록을 포함하여, 해당 다이어그램의 다이어그램 컴포넌트 목록, 다이어그램 캠퍼스, 편집 툴 그리고 실행 툴로 구성된다.

3.2 DML 변환 모듈

DML은 다이어그램 컴포넌트를 정의하는 XML로 다이어그램 시스템에서 다이어그램 컴포넌트 객체로 변환되어야 한다. 이러한 역할을 DML 변환 모듈이 수행한다. DML 그래픽 편집기는 사용자가 여러 개의 기본 도형을 이용해 다이어그램 컴포넌트를 정의하면 집합관계 트리 생성기를 통해 집합관계를 파악하여 트리를 생성한다. 이 트리를 순회하여 앞에서 정의한 DML 문서로 변환한다. 생성된 DML문서는 XSLT[16] 또는 DOM[17] 파서를 통해 Java, C++ 또는 C#등의 언어로 변환된다. 변환된 다이어그램 컴포넌트 객체는 다이어그램 시스템의 편집모듈에 등록되어 사용된다.

3.3 분석 모듈

분석 모듈은 크게 다이어그램 스캐너와 다이어그램 파서로 구성되어 있다. 다이어그램 스캐너는 편집모듈에서 읽은 다이어그램 컴포넌트에 대한 속성과 의미 정보를 이용하여 SRG(Spatial Relationship Graph) 자료구조를 생성한다. 다이어그램 파서는 다이어그램 문장에 대해 문법을 검사한다.

이때 만일 잘못된 문장인 경우 오류 메시지를 보내고 옳은 문장인 경우 Derived DAG를 생성한다.

3.3.1 다이어그램 스캐너

분석모듈에 속한 다이어그램 스캐너는 사용자가 편집기 모듈을 통해 입력한 다이어그램 문장에 대한 SRG를 생성하고 다이어그램 파서가 토큰을 요청할 경우 lookahead 토큰을 반환한다. 본 시스템에서 개발한 다이어그램 스캐너는 getRelationType 메소드를 호출하여 두 다이어그램 컴포넌트 사이의 관계성을 파악하고, buildSRG 메소드를 호출하여 SRG 자료구조를 생성한다. 생성된 SRG 자료구조에서 getNextNode 메소드를 호출하여 현재의 노드에 연결된 다음 lookahead 노드를 다이어그램 파서에게 반환한다.

SRG 노드는 다이어그램 컴포넌트인 경우 COMPONENT, 관계성 노드인 경우 RELATION의 종류를 갖고 실제 다이어그램 컴포넌트 객체에 대한 링크정보를 갖는다. 본 시스템에서 SRG 자료 구조를 생성하기 위한 알고리즘은 다음과 같다.

```
입력: 다이어그램 컴포넌트 집합
출력: SRG(Spatial Relationship Graph)
전체 다이어그램 컴포넌트 객체의 SRG노드를 생성한다.
for( 다이어그램 컴포넌트 집합의 각 모델 α에 대해 ) {
  for( 모델 α의 연결영역에 연결된 각 모델 β에 대해 ) {
    if( α와 β가 아직 관계성을 파악하지 않은 경우 ) {
      α와 β가 갖는 관계성의 타입과 수식을 파악한다.
        if( 관계성 수식을 계산하여 true인 경우 ) {
          관계성에 대한 SRG노드를 생성한다.
          관계성 노드와 α와 β노드를 서로 연결한다.
        }
    }
  }
}
```

전체 다이어그램 컴포넌트에서 두 개의 컴포넌트 사이의 관계성파악을 위한 시간 복잡도 T는 반복문이 내포되어 사용되었으므로 $O(n^2)$이다. 그러나 연결된 다이어그램 컴포넌트만을 고려하여 계산하면 $O(n) \leq T \leq O(n^2)$ 가 된다. 왜냐하면 O(n)은 내포된 반복문의 조건식이 참이 아닐 경우에 나타난다.

3.3.2 다이어그램 파서

다이어그램 파서는 사용자가 입력한 다이어그램 문장에 대해 문법적 오류를 검사한다. 다이어그램 파서는 ASG를 기반으로 재귀적 하향식 파싱(Recursive Descent Parsing)을 수행한다. 다이어그램 DTD는 확장된 LL(1) 문법을 이용한다. 따라서 STAR(*), PLUS(+), OPT(?)에 대한 오퍼레이션 노드는 다음 <표 1>과 같은 방법으로 처리된다.

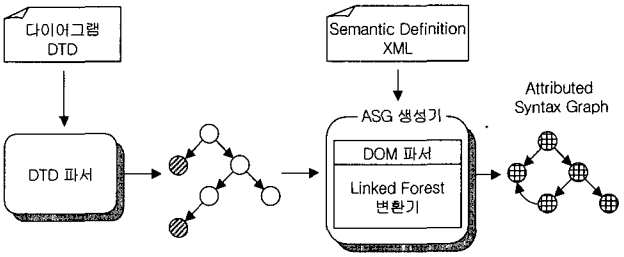
생성할 Derived DAG 노드의 종류는 다이어그램 컴포넌트와 관계성을 TERMINAL, 넌 터미널 원소를 나타내는 NONTERMINAL 그리고 문법규칙을 적용하기 위해 존재하는 터미널 노드인 CONTEXTSYMBOL로 구성된다. 그리고 사용자 입력 파라미터 값과 의미정보의 참조를 위해 SRG와 ASG노드에 대한 링크정보를 저장하고 있다.

<표 1> 오퍼레이션 노드 처리방법

C(*)의 경우	while(lookahead가 C의 FIRST인 동안) { ... }
C+의 경우	do { ... } while(lookahead가 C의 FIRST인 동안)
C?의 경우	if(lookahead가 C의 FIRST이면) { ... }

3.4 ASG 생성 모듈

ASG(Attribute Syntax Graph)는 다이어그램의 DTD를 기반으로 생성되며 DTD가 갖는 의미적 결함을 Semantic Definition XML을 이용해 보완한 구문그래프(syntax graph)로서 다이어그램 파서가 다이어그램 문장의 유효성 판단을 위한 자료구조로 활용한다. (그림 2)는 ASG 생성모듈을 보여준다.



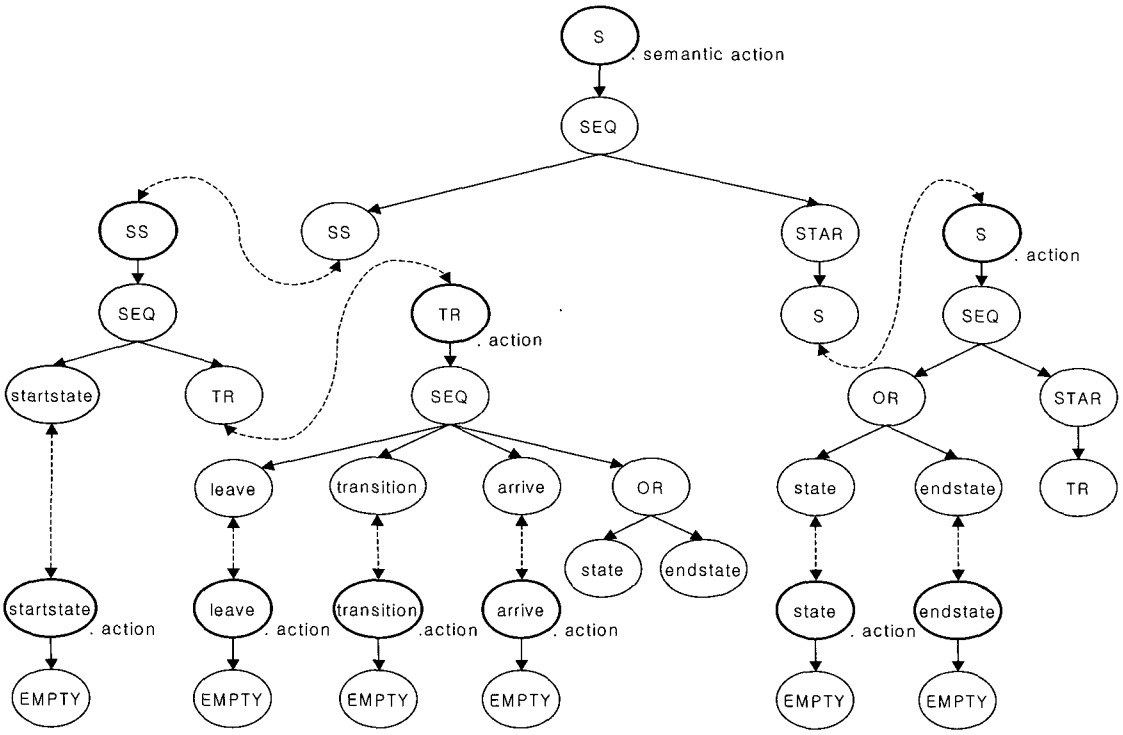
(그림 2) ASG 생성모듈

본 시스템에서 생성되는 ASG 노드의 종류는 ELEMENT, PCDATA, ANY, EMPTY 등과 같은 단말노드가 있으며, 비단말 노드로 SEQ(','), OR('|'), OPT('?'), PLUS('+'), STAR('*') 노드가 있다. 또한 의미 행위(semantic action)는 Semantic Definition XML의 해당 규칙(rule)에 정의된 의미정보를 추출하여 설정한다. 예를 들면 다음 <표 2>와 같은 상태도 DTD가 있다면, 본 시스템에서는 만들어진 상태도 ASG는 다음 (그림 3)과 같다.

(그림 3)에서 각 노드는 자신의 자식노드와 부모노드에 대한 링크정보를 갖고 각 트리의 부모노드는 자신을 참조하는 노드에 대한 링크정보를 갖고 있다. (그림 3)은 ASG생성모듈이 상태도 DTD를 파싱하여 생성한 트리이며 각 노드

<표 2> 상태도 DTD

```
<!ELEMENT STD (SS, S*)>
<!ELEMENT SS (startstate, TR)>
<!ELEMENT S ((state|endstate), TR*)>
<!ELEMENT TR (leave, transition, arrive, (state|endstate))>
<!ELEMENT startstate EMPTY>
<!ELEMENT state EMPTY>
<!ELEMENT endstate EMPTY>
<!ELEMENT transition EMPTY>
<!ELEMENT leave EMPTY>
<!ELEMENT arrive EMPTY>
```



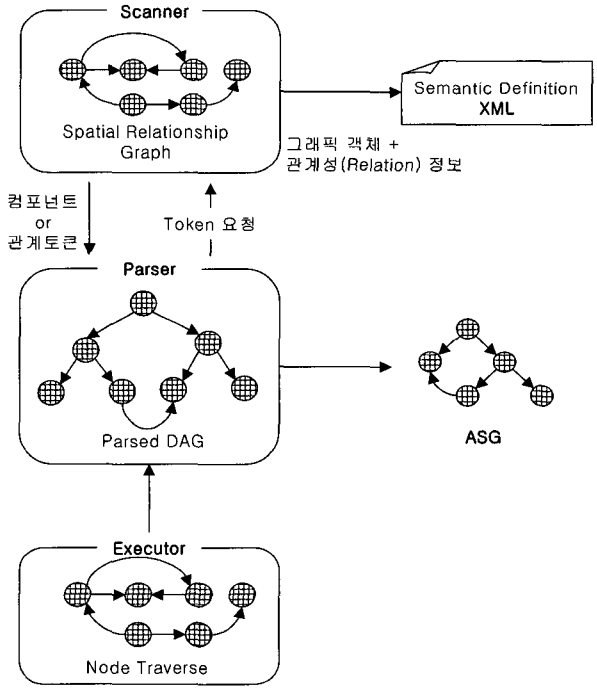
(그림 3) 상태도 ASG

에 의미 정의(Semantic Definition) XML에 정의된 노드의 의미 규칙들을 등록한 것이다. 이 자료구조는 다이어그램의 유효성 판단을 위한 재귀적 파싱(Recursive Descent Parsing) 과정과 다이어그램의 의미해석을 위한 기반 자료로 사용된다. 따라서 원활한 순회를 위해 각 노드는 자신의 자식노드와 부모노드에 대한 링크정보를 갖고 있다. DTD의 파싱 과정에 독립된 서브 트리들이 생성되고 서브 트리의 최상위 부모 노드는 자신의 위치에 대한 링크정보를 갖고 있어서 하나의 완성된 트리를 생성하게 된다. 의미규칙을 의미 정의 XML이라는 외부 XML 파일로 정의하는 또 다른 이유는 하나의 다이어그램에 여러 가지 의미규칙을 적용하기 용이하기 때문이다.

3.5 실행모듈

실행모듈은 분석모듈에서 생성된 Derived DAG를 순회하며 노드에 부여된 의미 행위를 실행한다. 다음 (그림 4)는 분석모듈과 실행모듈의 관계를 보여준다.

(그림 4)는 사용자가 생성한 다이어그램의 의미를 해석하기 위한 입력부터 실행까지의 과정이다. Parser는 ASG 자료구조를 기반으로 SRG의 입력을 이용해 Recursive Descent Parsing을 수행하여 Parsed DAG를 생성한다. 실행모듈은 생성된 Parsed DAG를 순회하면서 각 노드의 속성으로 ASG에 등록된 의미(Semantic)를 실행하며 노드 순회를 위한 컨트롤 역할을 하게 된다. 순회방법은 깊이 우선(depth-first evaluation order)을 기반으로 했으며 각 노드에 등록된 의미실행 결과에 따라 다른 노드로의 분기가 가능하다.

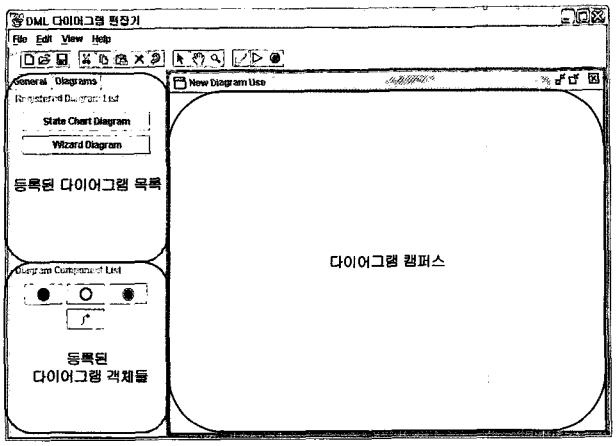


(그림 4) 다이어그램 분석 및 실행과정

4. 실험 및 평가

4.1 DML 다이어그램 편집기

다음 (그림 5)는 본 시스템에서 개발한 DML 다이어그램



(그림 5) DML 다이어그램 편집기

편집기의 초기화면을 보여준다. 그림에서와 같이 다이어그램 편집기는 새로운 다이어그램을 편집할 수 있는 영역과 이미 등록된 다이어그램을 사용할 수 있는 기능 등을 가지고 있다.

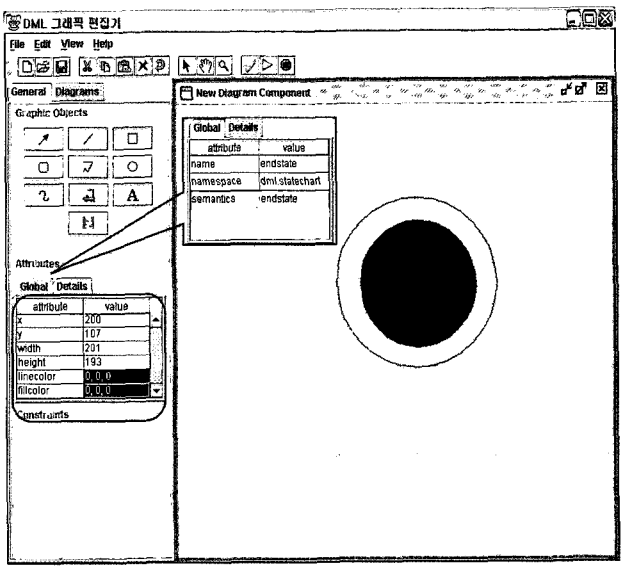
4.2 DML 그래픽 편집기와 문서 변환

DML 문서 변환은 DML 그래픽 편집기를 이용해 쉽게 변환할 수 있다. 즉, 그래픽 편집기에서 다이어그램 컴포넌트들의 형태와 속성을 정의하면 원하는 DML 문서를 얻을 수 있다. 예를 들면, 만일 다음 (그림 6)과 같이 DML 그래픽 편집기에서 다이어그램 형태와 속성을 정의하였다면, 본 시스템에서 변환된 DML 문서는 <표 3>과 같다.

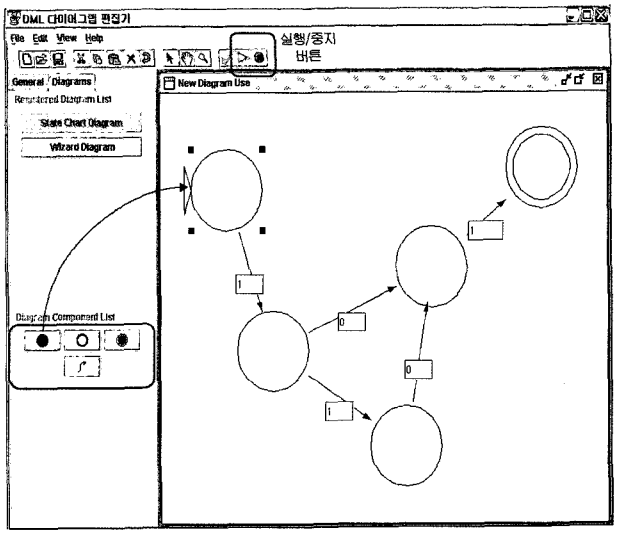
4.3 상태도

사용자는 자신이 원하는 다이어그램의 종류를 등록된 다이어그램 목록에서 선택한다. 다이어그램 컴포넌트의 종류(예를 들어, state, transition 등)를 선택해 캔버스에 Drag & Drop 또는 Click & Point를 한다. 다이어그램 문장을 모두 그린 후 실행버튼을 클릭 한다.

상태도가 모두 완성되었다면, 그 상태도의 유효성을 판단을 행한다. 다음 (그림 8)은 입력된 상태도의 유효성을 판단하는 과정을 보여준다.



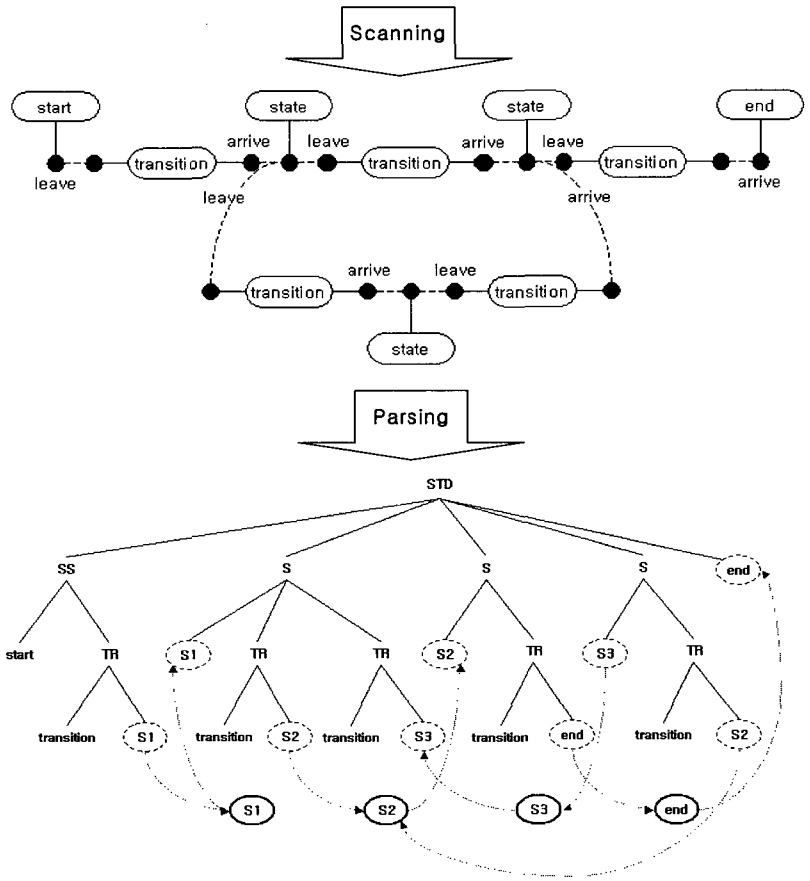
(그림 6) endstate 다이어그램 컴포넌트 정의



(그림 7) 상태도 실행화면

〈표 3〉 변환된 DML

```
<?xml version="1.0" encoding="EUC-KR"?>
<diagram name="endstate" coordx="200" coordy="200" namespace="dml.statechart">
<model>
   <appearance>
   <shape id="a" type="rect" x="0" y="0" width="200" height="200"/>
      <appearance>
      <shape id="b" type="circle" x="0" y="0" width="200" height="200"/>
         <appearance>
         <shape id="c" type="circle" x="30" y="30" width="140" height="140"
         linecolor="0,0,0" fillcolor="0,0,0"/>
         </appearance></appearance></appearance>
   <semantics name="endstate"/>
</model>
<connections><connection type="point" value="100, 100"/></connections>
</diagram>
```



(그림 8) 상태도의 유효성 판단

사용자가 그린 상태도 문장은 scanning 과정을 거쳐 SRG를 생성하고 parsing 과정을 통해 Derived DAG를 생성한다. 만일 해석 과정을 통해 올바르지 않은 다이어그램 문장의 경우 사용자에게 오류 메시지를 출력한다.

4.4 평 가

지금까지 개발된 XML 기반 다이어그램 시스템은 특정한 목적으로 하는 응용프로그램에 불과했다. 따라서 기존에 개발된 도구를 이용하여 자신의 응용프로그램을 개발하는 것은 XML 다이어그램을 새로 개발하는 것보다 더 어렵다. 따라서 XML 응용 프로그램에 맞는 범용성 다이어그램 시스템이 요구된다. 본 논문에서 제시한 시스템은 정적인 형태 정보뿐만 아니라 동적인 행위 정보를 갖고 있기 때문에 다양하게 활용할 수 있도록 개발되었다. 기존의 타 시스템에 비해 본 논문에서 제시한 시스템의 장점은 다음과 같다. 첫째, 본 시스템의 가장 큰 장점은 DML 문서를 쉽게 생성할 수 있다는 점이다. DML은 다이어그램 컴포넌트의 형태 정보, 의미 정보, 제약 사항, 연결 정보에 모든 요소를 포함하고 있기 때문에 새로운 다이어그램 개발 환경을 만들 수 있다. 둘째, 생성된 DML 문서는 XML DTD 기반의 문법검사와 의미실행을 수행한다. 이는 DTD를 이용하여 다이어그램에 대한 문법을 정의함으로써 XML DTD에 대한 의미적 일관성을 유지할 수 있다는 장점을 지닌다. 셋째, 본 시스템에서 제공되는 변환기를 이용하면, DML 문서를 Java나 C# 환경에서 이용할 수 있는 언어로 변환이 가능하다. 이 밖에도 본 시스템은 타 시스템에 비해 효율적이고 내부적인 자료 구조를 이용함으로써, 유효한(Valid) DML 문서를 만들 수 가 있으며, 다양한 인터페이스를 이용하여 시각적으로 편집할 수 있는 환경을 지원한다.

5. 결 론

본 논문에서는 현재 다이어그램 시스템을 만들기 위해 많은 시간과 노력을 들이는 문제와 서로 다른 시스템간의 자료공유의 어려움을 해결하고자 표준화된 문서형식인 XML을 적용해 다이어그램 컴포넌트를 나타내고 올바른 다이어그램 문장을 작성할 수 있도록 도와주는 시스템에 대해 제안하였다. 본 논문에서 제안한 다이어그램 시스템은 위지웍(WYSIWYG) 개념을 이용하였으며, 다이어그램 컴포넌트를 정의하여 자동적으로 DML 문서를 생성할 수 있었다. 또한 DTD를 이용하여 다이어그램에 대한 문법을 정의함으로써 XML DTD에 대한 의미적 일관성을 유지할 수도 있었다. 본 논문의 실험 부분에서는 본 시스템을 활용하면 쉽게 DML 문서를 생성할 수 있었으며, XML DTD 기반의 문법

검사와 의미실행을 할 수 있다는 것을 보여주었다.

본 논문의 향후 연구과제로는 제약사항을 직접 기술하는 현재의 불편함을 개선하여 자동화된 방법을 제공하고, 여러 개의 다이어그램 컴포넌트를 사용하면서 생기는 복잡한 화면을 자동으로 레이아웃(layout)을 지정하는 방법과 추상화(abstraction)시키는 방법에 대해 연구하여야 할 것이다.

참 고 문 헌

[1] Booch, G., Rumbaugh, J. and Jacobson, I., The Unified Modeling Language User Guide, Addison-Wesley, 1999.

[2] Scalable Vector Graphics, http://www.w3.org/Graphics/SVG

[3] Vector Markup Language, http://www.w3.org/TR/NOTE-VML, 1998.

[4] Extensible Markup Language (XML) 1.0, http://www.w3.org/TR/2000/REC-xml-20001006

[5] Simple Object Access Protocol (SOAP) 1.1, 2000. http://www.w3.org/TR/SOAP

[6] XML Query, http://www.w3.org/XML/Query

[7] Chemical Markup Language, http://www.xml-cml.org

[8] Mathematical Markup Language, http://www.w3.org/Math

[9] Extensible Graph Markup and Modeling Language, http://www.cs.rpi.edu /~puninj/XGMML

[10] Mark Minas and Oliver Köth, Generating Diagram Editors with DiaGen, in Proc. of the Int'l Workshop with Industrial Relevance, pp.433-440, Sep., 1999.

[11] Mark Minas, Concept and realization of a diagram editor generator based on hypergraph transformation, in Journal of Science of Computer Programming(SCP), 2001.

[12] CLIP! XML Editor Avalilable http://xml.t2000.co.kr

[13] XML Pro v2.0 Available http://www.vervet.com/products.php

[14] EXml v1.2 Available http://www.cuesoft.com/products/exml.asp

[15] xmlspy® 2004, Available http://www.xmlspy.com/products_ide.html

[16] XSL Transformations, http://www.w3.org/TR/xslt

[17] Document Object Model, http://www.w3.org/DOM

김 성 근

e-mail : skkim@csangji.ac.kr

1993년 숭실대학교 전자계산학과 졸업

1995년~현재 숭실대학교 컴퓨터학과 박사과정

1998년~현재 가톨릭상지대학 컴퓨터정보계열 조교수

관심분야 : 프로그래밍 환경, 에이전트 프로그래밍 언어

김 영 철

e-mail : yckim@ss.ssu.ac.kr

1990년 한남대학교 전자계산학과(학사)

1996년 숭실대학교 전자계산학과(석사)

2003년 숭실대학교 컴퓨터학과 공학박사

현 재 숭실대학교 정보미디어기술연구소

관심분야 : 프로그래밍 언어, 컴파일러, XML, 컴퓨터 통신

윤 태 희

e-mail : oinee2k@locus.com

2001년 숭실대학교 컴퓨터학부(학사)

2003년 숭실대학교 컴퓨터학과(석사)

현 재 (주)로커스 테크놀로지스

관심분야 : HCI, XML, 컴파일러

유 재 우

e-mail : cwyoo@comp.ssu.ac.kr

1976년 숭실대학교 전자계산학과(공학사)

1978년~1985년 한국과학기술원 전산학과(공학석사, 박사)

1986년~1987년, 1996년~1997년 Cornell Univ.와 Univ. of Pittsburgh의 Visiting Scientist

1983년~현재 숭실대학교 컴퓨터학부 교수

관심분야 : 컴파일러, 프로그래밍 환경, 인간과컴퓨터