개념적 시간관계 기반의 멀티미디어 프레젠테이션 저작 시스템

(A Multimedia Presentation Authoring System based on Conceptual Temporal Relations)

노 승 진 † 장 진 희 † 성 미 영 ††
(Seung Jin Rho) (Jinhee Jang) (Mee Young Sung)

요 약 모든 개념적 시간관계는 7가지의 관계('before', 'meets', 'starts', 'finishes', 'overlaps', 'during', 'equals') 중 하나로 표현될 수 있다. 개념적 표현은 멀티미디어 저작 시스템의 자동 생성에 필요한 세부적 시간에 대해 효과적인 수단을 제공한다. 본 연구에서는 서로 다른 미디어들 간의 시간관계를 개념적으로 표현하는 사용하기 쉽고 효과적인 멀티미디어 프레젠테이션 저작 시스템을 개발하였다.

본 시스템을 구성하는 시간관계 편집기는 사용자에게 다른 편집기들로부터의 시간 정보를 간단하고 직접적인 그래픽 조작을 이용하여 프레젠테이션의 개념적 흐름을 직관적으로 표현할 수 있는 메커니즘을 제공한다. 본 시스템은 SMIL(Synchronized Multimedia Integration Language)에 기반한다. 본 시스템의 편집기들은 SMIL 객체 관리자를 통해 실시간으로 정보를 서로 교환하여 SMIL 코드를 자동 생성한다. 그리고, 본 시스템에서는 멀티미디어 프레젠테이션의 내부표현 구조로 TRN(Temporal Relation Network)을 제안한다. TRN은 프레젠테이션의 흐름을 방향 그래프 구조로 표현한 것이다. TRN의 모든 병렬관계는 하나의 동기화된 블록으로 간소화될 수 있다. 이것은 컴포넌트들 간의 재생시간을 결정하는데 유용하며, 이미 구성되어 있는 프레젠테이션 문서를 재사용 할 때 그 기본단위로 이용될 수 있다. 또한, 멀티미디어 프레젠테이션 플레이어의 스케줄러로의 응용에도 적합하다.

키워드 : 멀티미디어 프레젠테이션 저작, 개념적 시간관계, TRN(Temporal Relation Network), SMIL(Synchronized Multimedia Integration Language)

Abstract Every conceptual temporal relationship can be described using one of seven relations (before, meets, overlaps, during, starts, finishes, and equals). The conceptual representation provides an efficient means for our multimedia authoring system to automatically fill in the necessary timing details. We developed a multimedia presentation authoring system that supports a mechanism for conceptually representing the temporal relations of different media.

Among the many editors that make up our system, the temporal relation editor provides users with an intuitive mechanism for representing the conceptual flow of a presentation by simple and direct graphical manipulations. Our system is based on the SMIL(Synchronized Multimedia Integration Language). The conceptual temporal relation editor and other editors of our system exchange their information in real-time and automatically generate SMIL codes through the SMIL Object Manager. Our system uses TRN(Temporal Relation Network) as its internal multimedia presentation representation. The TRN corresponds exactly to the structure seen in the graphical representation of the presentation. A parallel relationship found in a TRN can be collapsed into a single synchronization block. This facilitates the determination of the playing time of each component and can be the basic

· 본 연구는 한국과학재단 지정 인천대학교 멀티미디어연구센터의 지원에 의한 것입니다.
† 비 회 원 : 인천대학교 컴퓨터공학과
sjrho@incheon.ac.kr
S951111@incheon.ac.kr
†† 종신회원 : 인천대학교 컴퓨터공학과 교수
mysung@incheon.ac.kr
논문접수 : 2002년 10월 1일
심사완료 : 2003년 2월 4일

unit for reusability of already prepared blocks of presentation code.

Key words : Multimedia Presentation Authoring, Conceptual Temporal Relation, TRN(Temporal Relation Network), SMIL(Synchronized Multimedia Integration Language)

1. 서 론

우리는 흔히 말하고 있는 시간과 공간을 초월한 인터넷시대에 살고 있다. 그와 더불어 멀티미디어 시대라는 말로 현대를 인용하기도 한다. 멀티미디어란 이미지, 사운드, 동영상, 텍스트 등과 같은 서로 다른 단일 미디어들의 조합된 형태를 말한다. 멀티미디어 서비스의 요구는 1990년대 중반부터 급속히 발전한 인터넷의 영향으로 인터넷상의 멀티미디어 서비스라는 좀 더 진보적인 형태로 발전하게 되었다. 단순히 로컬 컴퓨터상의 멀티미디어 서비스에서 더 나아가 네트워크를 통한 멀티미디어 서비스 형태가 급속히 확산됨으로써 기존의 애플리케이션 위주의 멀티미디어 서비스는 인터넷이라는 매체를 통해 보다 발전된 형태의 서비스를 제공하게 되었다.

인터넷의 확산과 더불어 HTML(Hyper Text Markup Language)을 비롯한 각종 태그 언어들이 정의되고 사용되면서 인터넷 이용자의 멀티미디어에 대한 요구는 날로 증가하고 있는 추세이다. 그러나 HTML은 단순히 텍스트와 이미지 중심의 문서구조를 가지고 있어 정적인 정보 위주의 서비스를 제공할 수밖에 없는 한계에 접하게 되었다. 이에 따라 DHTML(Dynamic HTML)로의 발전과 각종 플러그인 프로그램이 웹을 통해서 지원되고 있지만 이 또한 웹 상에서 프레젠테이션 기능을 원활히 표현하지 못하고 있는 실정이다. 이러한 상황에서 W3C는 1998년 6월 SMIL(Synchronized Multimedia Integration Language)이라는 표준 권고안을 제정하여, 일련의 개별적 멀티미디어 객체를 동기화된 멀티미디어 표현으로 통합할 수 있도록 하였다[1].

SMIL은 HTML과 유사하게 정의되어 배우기 쉽고 사용하기 쉬운 언어로 출발하였다. SMIL은 XML (eXtensible Markup Language) 기반의 다중매체를 위한 종합언어로서 간단한 몇 개의 태그와 속성들로 구성된다. 웹에서 멀티미디어 스트리밍 서비스를 하기 위해서는 많은 기술들을 필요로 하며 표현하는 방법도 여러 가지이다. 이러한 기술들을 SMIL로 통합하여 보다 쉽고 효율적으로 멀티미디어 컨텐츠를 제작할 수 있게 하였다. HTML이 XML로 발전해 가는 움직임에 따라 SMIL도 XML에 기반을 두어 만들어졌다[2-5]. 그에 따라, 현재 상용화되었거나 상용화 단계에 있는 SMIL 시스템들은 미국 제품으로는 Allaire HOMESITE(Macromedia), SMIL Validator(CWI), HotSausage SMIL Composer SuperTool, (LP Studio), GRiNS(Oratrix), RealSlideshow 2.0 (RealNetworks), TAG Editor 2.0(G2 release by Digital Renaissance), VEON authoring tool(VEON), MAGpie(NCAM), Fluition (Confluent Technologies Inc.) 등이 있고, 일본 제품으로는 SMIL Editor ver.1.0(Rikei), 그리고 우리나라 제품으로는 TagFree 2000 SMIL 저작 도구(다산기술)가 있다.

위와 같은 현존하는 시스템에서 제공하는 SMIL 편집 도구들은 텍스트 편집기로 태그 데이타와 내용 데이타를 혼합하여 편집하는 형식이 대부분이어서 SMIL 태그들을 잘 알지 못하는 비전문가들이 사용하기에 매우 불편하다. 그러므로 직관적이고 쉬운 사용자인터페이스를 제공하는 SMIL 기반의 동기화된 멀티미디어 데이타를 쉽게 저작할 수 있는 시스템이 절실히 요구된다.

이 논문에서는 SMIL을 지원하는 멀티미디어 프레젠테이션을 위한 개념적 시간관계 편집기의 설계 및 구현에 대해 소개한다. 2장에서는 관련연구로 멀티미디어 데이타 모델링에 대해 알아보고, 3장에서 본 멀티미디어 프레젠테이션 저작 시스템의 설계 내용에 대해 자세하게 살펴본다. 그리고 4장에서는 본 시스템의 구성에 대해 살펴보고 5장에서는 연구결과로 본 시스템을 활용한 예와 다른 SMIL 기반 멀티미디어 저작 시스템과 비교해 본다. 끝으로 6장에서는 결론을 내리고 향후 연구과제에 대해 알아본다.

2. 멀티미디어 데이타 모델링

이 장에서는 멀티미디어 데이타 모델링을 위하여 시간 정보를 표현하는 모델들에 대해서 시간 축 모델(time-line model)과 페트리 네트(Petri Nets) 기반 모델(interval-based model)을 중심으로 간단히 살펴보도록 하겠다.

멀티미디어 저작은 크게 두 가지로 구분해 볼 수 있다. 하나는 멀티미디어 문서 저작(multimedia document authoring)이고 다른 하나는 멀티미디어 프레젠테이션 저작(multimedia presentation authoring)이다[6]. 멀티미디어 문서 저작은 문서를 기본 골격으로 하고 이 문서 구조에 다른 미디어 자료를 포함(inclusion)시키는

기법이라고 할 수 있다. 한편, 멀티미디어 프레젠테이션 편집은 주로 시간적인 조합에 중점을 두어 시간 축에 따라 여러 미디어 자료를 공간적으로 위치시키는 기법이라고 할 수 있다. 그러므로 프레젠테이션 저작은 시간 관계(temporal relation)의 명세를 위한 모델이 중요시되며 공간 관계(spatial relation)는 시간 합성 모델 안의 한 요소에 대한 사항으로 볼 수 있다. 또한, 멀티미디어 데이타의 특징은 데이타 내에 시공간(spatio-temporal) 정보를 내재하고 있는 경우가 많다는 점이다. 비디오 데이타의 경우에 비디오의 각 프레임이 순차적으로 재생되어야할 뿐만 아니라 비디오 데이타와 오디오 데이타간의 동기화가 필요하다[7,8]. 따라서 이러한 시간 관계에 관한 정보를 데이타 모델에 적절히 표현해 줄 수 있어야 한다.

2.1 시간 축 모델

시간 축(time-line) 모델에서 모든 사건(event)들은 하나의 절대적인 시간 축 상에 표현된다[9]. 이때 사건이란 시간 축 상에서 각 미디어들이 상연되어지는 시작 시점 또는 끝 시점을 의미한다. 따라서 각 미디어들의 동기화 정보는 절대적인 시간 축 상에 표현된 사건들의 시점에 의해 표현되며 미디어들은 해당 시점에서 화면사이에 적절히 표시된다. 예를 들어, 여러 개의 슬라이드와 각 슬라이드에 해당되는 내용을 동시에 보여주는 시나리오를 Time-line 모델로 표현하면 그림 1과 같다.

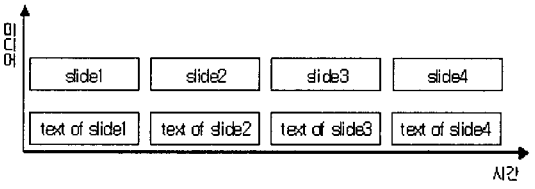

그림 1 타임라인(Time-line 모델)

2.2 페트리 네트 기반 모델

OCPN(Object Composition Petri Nets) 모델은 Allen이 정의한 13가지의 시간 관계[10]에 의해 시간 정보를 표현한다[11,12]. 이 모델은 페트리 네트(Petri Net)[13]를 이용하여 미디어들 사이의 동기화 정보를 나타내며, 미디어 사이의 시간 관계를 이진 관계 이상의 다 차원 관계로 표현할 수 있도록 페트리 네트 개념을 확장한 모델이다. 그림 2는 OCPN에 의해 표현된 시나리오의 예를 나타낸다.

그림 2에서 t 는 해당 프로세스의 시간간격(intervals)을

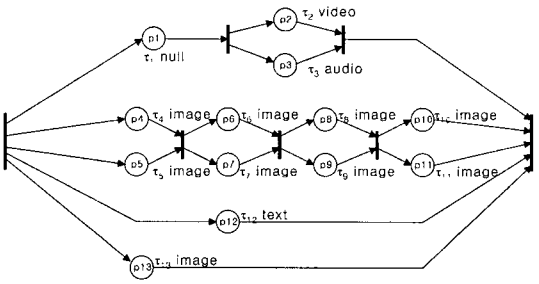

그림 2 OCPN(Object Composition Petri Nets) 모델

의미하며, 그 옆은 각 프로세스 객체들의 타입(type)을 명시함으로써 프레젠테이션을 모델링하게 된다.

그림 2에서 전체 재생시간 τ_T는 다음과 같다.

$$\tau_T = \tau_1 + \tau_2 = \tau_1 + \tau_3 = \tau_4 + \tau_6 + \tau_8 + \tau_{10}$$
$$= \tau_5 + \tau_7 + \tau_9 + \tau_{11} = \tau_{12} = \tau_{13}$$

OCPN 모델은 지연객체를 포함한 멀티미디어 데이타를 프로세스(process) 단위로 추상화하여 프레젠테이션을 효과적으로 표현하지만 미디어들 간의 시간관계 표현에서는 직관적이지 못하다는 결점을 가지고 있다.

3. 멀티미디어 프레젠테이션 저작 시스템

지금부터 이 연구에서 제안하는 개념적 시간관계 기반의 멀티미디어 프레젠테이션 저작 시스템에 대하여 자세히 살펴보도록 하자.

3.1 개념적 시간관계 표현

두 미디어간의 시간관계는 Allen이 밝힌 13가지 상황으로 표현될 수 있다[10]. Allen이 밝힌 13가지 시간관계는 서로 역관계(예를 들면, A before B와 B before A의 관계)에 있는 시간관계를 제거하면 7가지의 시간관계로 압축할 수 있으며 이 연구에서 다루는 모든 시간관계는 Allen의 meet, before, overlaps, during, starts, finishes, equals의 7가지 상황 중 하나에 적용된다.

이 연구의 목적은 이러한 구체적인 시간관계들을 기본 골격으로 하여 사용자에게 보다 직관적인 인터페이스를 제공하는 것에 있다. 이 연구에서 고유하게 제안하는 개념적 시간관계 기반 편집기에서는 미디어 객체의 논리적인 흐름을 이들 7가지 시간 관계로 표현하는데, 이들 시간관계를 아이콘과 화살표로 구성된 작업 네트워크(activity network) 형태로 보여 줌으로써 사용자에게 직관적인 프레젠테이션의 흐름을 제공한다[14].

멀티미디어 프레젠테이션에서는 시간에 따른 미디어 객체들의 행동을 기술하는 것에 초점이 맞추어져 있으므로 이 시스템은 미디어 객체들 간의 개념적인 시간

행위에 대해 사용자가 선택할 수 있도록 설계되었다. 그림 3은 이 시스템에서 사용하는 7가지의 개념적 시간관계의 그래픽 표현을 보여주고 있다. 그림에서 볼 수 있듯이 병렬관계('starts', 'finishes', 'overlaps', 'during')들은 실질적인 미디어 객체가 아닌 지연 객체(delay object)가 추가되어 'equals' 관계를 형성함을 알 수 있다.

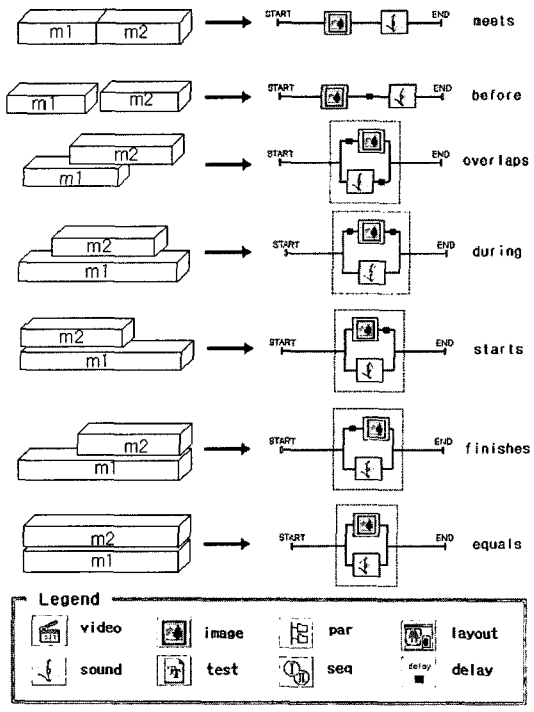

그림 3 시간관계에 따른 그래픽 표현

미디어객체들 간의 개념적 병렬관계는 필요한 경우 지연 객체를 삽입하여 'equals' 관계로 일반화하였다. 또한, 개념적 병렬관계는 하나의 객체로 통합 표현될 수도 있는데 이 시스템에서는 그것을 블록(block) 표현으로 단순화하였다. 블록 표현과 블록이 확장된 표현 그리고 이에 해당하는 시간 축 기반 표현은 5장의 그림 10에서 보여주는 예제에 잘 나타나 있다. 또한, 이미 작성된 SMIL 코드를 재사용 할 수 있도록 SMIL 사양을 확장할 경우, 이러한 블록 표현은 코드 재사용 메카니즘의 한 단위로 이용될 수 있다[15].

3.2 공간관계 표현

음성 미디어를 제외한 미디어들(이미지, 비디오, 텍스트, 텍스트 스트림, 애니메이션 등)은 시간 관계뿐 아니라 공간상의 배치로 표현되어야 한다. 공간 배치를 위한 인터페이스에서는 드래그&드롭 방식으로 객체를 배치고 속성 인터페이스에서 속성 값을 변경하여 해당 미디어의 공간 배치에 대한 세세한 값들을 정의할 수 있다. QuickTime이나 RealPlayer 등 현재까지 개발된 SMIL 플레이어들은 각 미디어들이 같은 시간에 재생되는 같은 공간상의 오버랩핑 즉, 같은 공간 같은 시간에 재생되는 것을 지원하지 않으므로 같은 공간 안의 재생일 경우에는 순차적인 재생만을 지원한다. 그러므로, 이 연구에서는 공간상의 오버랩핑은 고려하지 않았다.

3.3 TRN(Temporal Relation Network)과 DOM (Document Object Model)

멀티미디어 프레젠테이션 편집은 사용자와의 상호 작용에 따라 구성된다. 즉, 사용자의 입력에 시스템은 즉각적인 반응과 적당한 피드백(feedback)을 제공하여야 하는데 이를 위해서 편집 시스템에서는 프레젠테이션에 대한 표현을 별도의 내부적인 자료구조를 이용해 유지하고 관리하고 있어야 한다. 여기서 우리는 멀티미디어 문서의 저작 시스템의 내부 자료구조로 TRN(Temporal Relation Network)을 제안한다. 이 논문에서 제안하는 TRN은 DAG(Directed Acyclic Graph) 구조이며 사이클이 없는 AOV(Activity On Vertex) 네트워크 형태로 시스템 내부에 존재하면서 멀티미디어 문서가 포함하게 되는 각 미디어 객체들의 정보를 가지고 미디어들 간의 시간관계 및 재생시간을 표현하게 된다.

표준화된 문서객체 모델 DOM(Document Object Model)은 트리(tree)구조를 이용해 문서 컴포넌트들의 논리적 관계를 묘사하며 SMIL 문서도 DOM으로 표현된다. 그러나, 트리 구조는 SMIL로 표현된 미디어 객체들의 개념적인 시간관계들을 묘사하기에 적절하지 않다. 그러므로, 이 연구에서는 SMIL 문서 내의 미디어 객체들간의 개념적인 시간관계를 효과적으로 서술할 수 있는 별도의 데이타 구조와 관련 메커니즘을 제안한다.

이 시스템의 저작을 위한 시각적인 인터페이스는 시스템 내부에서 사용되어지는 TRN 구조 그대로를 그래픽하게 표현한 형태이다. 즉, SMIL 문서에 표현된 미디어 객체들 간의 시간관계와 그래픽하게 표현된 개념적 시간 관계는 동일하다.

TRN은 새로운 프레젠테이션을 만들 때마다 저작자가 그래픽 인터페이스에 표현한 대로 구성되면서 생성된다. 이미 생성되어진 SMIL 문서의 경우는 파싱(parsing)을 통해서 문서의 내용을 DOM 구조로 변환시킨 후 자동으로 TRN 구조로 변환시킨다. 그리고 사용자 인터페이스는 이렇게 생성된 TRN의 내용을 편집기로 출력하게 되며 사용자가 편집을 하게 되면 하나의

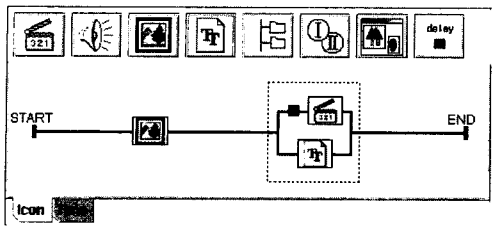

그림 4 그래픽 인터페이스

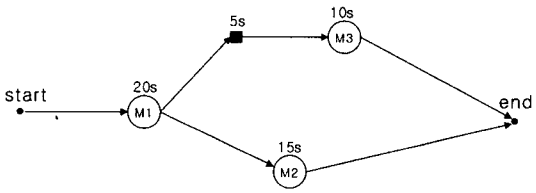

그림 5 사용자의 입력에 의해 생성된 TRN 구조

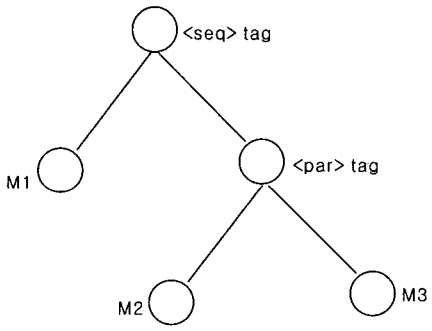

그림 6 TRN에 의해 생성된 DOM 구조

편집 작업이 일어날 때마다 TRN은 동적으로 변화된다. 이렇게 변화된 TRN의 정보는 DOM으로 다시 변환되고 이에 따라 SMIL 문서도 갱신된다. 이렇게 시스템은 TRN과 DOM 사이의 상호 작용을 통해 SMIL 문서를 자동 생성한다.

그림 5는 사용자가 그림 4에서와 같이 프레젠테이션을 저작할 때 생성되는 TRN의 예를 보여주고 있다. 각 노드는 하나 이상의 화살표를 통해 해당 노드의 작업이 종료된 후의 방향을 표현한다.

저작이 진행되는 동안 시스템은 그림 6과 같은 DOM 자료구조와 SMIL 문서를 동적으로 생성한다. 다음은 생성된 SMIL 문서의 일부분으로서 SMIL의 주요한 시간관계 태그는 <seq>과 <par>이다. <seq>는 시간적으로 순차적(sequential)인 관계를 표현하고 <par>는 시간적으로 병렬인 관계를 표현한다.

```
<seq>
  <img id="M1" src="media1.gif" region="i_region"
    dur="20s"/>
  <par>
    <text id="M2" src="text.txt" region="t_region"
      dur="15s"/>
    <video id="M3" src="video.mpg" region=
      "v_region" begin="5s" dur="10s"/>
  </par>
</seq>
```

3.4 멀티미디어 프레젠테이션 모델(Model of Multimedia Presentation)

이 절에서는 본 연구에서 제안하는 멀티미디어 프레젠테이션 모델에 대해서 살펴보도록 하겠다. 이미 멀티미디어 프레젠테이션 표현의 핵심이 TRN이 있다고 설명해 왔다. TRN이 프레젠테이션의 시간적 흐름을 명세한다는 것을 기억하자. 그리고, 프레젠테이션 영역에서의 컴포넌트들의 배치(layout)는 프레젠테이션의 공간적 관점이 되고 프레젠테이션 공간의 정보를 표현하기 위해 둘 이상의 산술적 수치들을 가진다.

본 시스템에서는 멀티미디어 프레젠테이션을 다음과 같이 정의한다.

$$MP=(TRN,\ LA)$$

여기서 MP는 멀티미디어 프레젠테이션, TRN은 Temporal Relation Network, LA는 공간 배치 속성들의 집합이다.

멀티미디어 프레젠테이션 MP를 위한 TRN은 다음과 같이 정의한다.

$$TRN(MP)=<\ O,\ R\ >$$

O는 멀티미디어 객체들의 집합이며, 지연객체(delay object)를 포함한다. 또한 R은 미디어 객체 집합 O에서 미디어 객체들간의 시간관계를 말한다.

또한 O는 다음과 같이 정의한다.

$$O=\{n(u_i)\ |\ ui\in DUR\}$$

$n(u_i)$는 u_i 객체를 나타내는 노드이며 해당 작업을 의미한다. DUR은 프레젠테이션에서 0(zero) 이상의 값을 가진 객체들의 재생시간 집합이다.

DUR의 정의는 다음과 같다.

$$DUR=\ \underset{m_i\in M}{\mathrm{Y}}\{m_i.d\}\ \underset{d_i\in D}{\mathrm{Y}}\{d_i.d\}\ \underset{o_i\in O}{\mathrm{Y}}\{o_i.d\}$$

$m_i.d$는 미디어 객체들의 집합 M에서의 m_i의 재생시간이며, $d_i.d$는 지연객체 d_i의 재생지연시간이다. 따라서 집합 O는 재귀적(recursive)으로 정의됨으로써 미디어 객체들만을 포함하는 것이 아니라 압축된 형태의 블록 표현도 가능하게 한다.

시간관계 R은 다음과 같이 정의한다 :

$$R=\{r(o_i, o_j) | o_i \in O, o_j \in O, r \in T\}$$

o_i와 o_j는 O집합의 원소이며, r은 이들 원소사이의 시간관계가 된다. T는 다음과 같은 원소들로 구성된다.

$$T=\{meets, before, starts, finishes, during, overlaps, equals\}$$

다음은 멀티미디어 프레젠테이션 MP의 LA에 대한 정의이다.

$$LA(MP)=\{l_i(p_i, s_i) | p_i \in P, s_i \in S\}$$

l_i는 배치(layout) 객체이며 p_i와 s_i로 구성된다. p_i는 위치(position)를 나타내는 속성 값 {*top, left*}으로 구성되며, s_i는 크기(size)를 나타내는 속성 값 {*width, height*}로 표현된다.

P의 정의는 다음과 같다:

$$P = \bigcup_{l_i \in L} \{(l_i.top, l_i.left) | 0 \le l_i.top, 0 \le l_i.left\}$$

S의 정의는 다음과 같다:

$$S = \bigcup_{l_i \in L} \{(l_i.width, l_i.height) | 0 \le l_i.width, 0 \le l_i.height\}$$

요약하면, 멀티미디어 프레젠테이션은 TRN과 배치속성(layout attributes)들로 구성되고 TRN은 미디어 객체 사이의 시간관계들의 집합과 미디어 객체의 재생시간에 의해 결정된다. 또한, 미디어 객체 사이의 시간관계는 7가지 시간관계(meet, before, overlaps, during, starts, finishes, equals)들로 정의될 수 있다.

TRN은 멀티미디어 프레젠테이션을 저작을 위한 여러 내부연산과 연계되어 있다. 멀티미디어 프레젠테이션 저작을 위해 본 시스템이 제공하고 있는 기본적인 내부연산에는 *InsertObject*(o_i, $M_{clip\text{-}start}$, d), *RemoveObject* (o_i), *ModifyObject*(o_i, $M_{clip\text{-}start}$, d), *InsertRelation*(r, o_i, o_i), *ModifyRelation*(r, o_i, o_j), *RemoveRelation*(o_i, o_j), *InsertDelay*(o_i, d_i, d), *RemoveDelay*(d_i), *Modify Delay*(o_i, d_i, d), *InsertLayout*(l_i, p_i, s_i), *Remove Layout*(l_i), *ModifyLayoutPosition*(l_i, p_top, p_left), *ModifyLayoutDimension*(l_i, *width*, *height*) 등이 있다. 여기서, o_i는 블록 객체를 포함한 멀티미디어 객체를, $M_{clip\text{-}start}$는 미디어 객체에서 재생이 시작되는 시점을, 그리고 d는 재생 시간(duration)을 의미한다. 또한, l_i는 레이아웃 객체를 말하고 d_i는 지연 객체를 말한다. 그리고, p_i와 s_i는 각각 layout의 위치와 크기를 뜻한다.

3.5 TRN(Temporal Relation Network) 구성 알고리즘

이 절에서는 본 시스템에서 TRN을 구성하는 과정에 대해 살펴보겠다. 이미 만들어진 기존의 SMIL 문서를 파싱을 통해 DOM 자료구조를 얻어오는 과정에 대해서는 생략하도록 하겠다. 표 1은 DOM 문서로부터 TRN을 생성하기 위한 DOM 분석 모듈이다.

표 1 Building_TRN 알고리즘

```
bulid_TRN(Node node)
//DOM문서의 body node로부터 TRN을 생성한다.

begin
 while(node != NULL) // buliding 종료조건
 // 단말노드가 없을 때까지 순회한다.
 begin
 if(node_type == PAR)
 begin
 insertPar(node); //par 블록 처리 메소드
 node ::= getNextSibling(node);
 // 이웃한 노드를 얻어온다
 end
 else if(node_type == SEQ)
 begin
 //현재노드의 자식 노드를 얻어온다
 node ::= getChildNode(node);
 end
 else
 begin
 insertSeqNode(node); // node를 TRN에 삽입
 node ::= getNextSibling(node);
 if(node == NULL)
 begin
 node ::= getParentNode(node);
 node ::= getNextSibling(node);
 end
 end
 end
end
```

모듈 building_TRN은 DOM 자료구조의 Node 타입의 node를 매개변수(arguments)로 하여 최종 단말노드(terminal node)가 널(NULL)일 때까지 반복하여 실행하게 된다. TRN을 구성하기 위해 가장 먼저 입력받는 것이 <body> 태그의 node이다. 즉, building_TRN은 <body> 태그의 내용만으로 TRN을 구성하게 된다. 각 노드는 각 노드의 고유한 타입(node_type)으로 요소이름(element name)을 사용하게 되며, 해당 노드 타입에 따라 별도의 추가적인 TRN 구성 알고리즘으로 분기된다. 노드의 타입에 따라 <par> 태그는 insertPar()로 이동하며, <seq> 태그는 자식노드를 순회하게 된다. 또한, <par> 또는 <seq> 태그가 없는 경우에는 디폴트로 <seq> 태그가 적용되므로 이웃하는 노드를 순회하게 된다. 위에서 알 수 있듯이 building_TRN 모듈은 실제로 body node에 대한 순회를 담당하고 있으며, insertPar() 모듈과 insertSeqNode() 모듈이 해당 node를 실제로 TRN에 삽입하는 모듈이다. TRN을 구축하는 이 세 가지 모듈들은 DOM 문서의 트리 구조를 순차적 또는 직접적으로 순회할 수 있는 메소드들

(getNextSibling, getChildNode, getChildNodes, get ParentNode 등)을 포함하고 있다.

표 2의 모듈 insertSeqNode()에서는 매개변수로 전달받은 노드의 속성들을 이용해 미디어 객체를 생성하고 필요에 따라 지연 객체를 추가적으로 생성하여 TRN에 삽입하고 지연 객체의 재생 시간(do.dur)에 따라 순차적 시간관계인 'meets'와 'before' 중 하나의 시간관계를 설정하게 된다.

표 2 insertSeqNode 알고리즘

```
insertSeqNode(Node node)
//노드의 속성 값을 객체에 저장, 해당객체를 TRN에 삽입
DelayObject do; //node의 begin속성을 위한 Delay 객체
MediaObject mo;
..... // node의 속성 값으로 미디어 객체 mo 생성

if (node_begin_value > 0)
 DelayObject do = createDelayObject(node);
begin
 if (do.dur > 0)
 begin
 insertDelay(); // Delay 객체 do를 삽입
 insertObject(); // Media 객체 mo를 삽입
 insertRelation(); // 'before' relation 설정
 end
 else
 begin
 insertObject();
 insertRelation(); // 'meets' relation 설정
 end
end
```

표 3에 요약한 insertPar() 모듈은 <par> 태그를 구현하는 알고리즘으로서 각 노드에 대해 재귀적(recursive)으로 동작하여 객체들을 TRN에 삽입하는데 아래와 같은 네 부분으로 구성되어 있다.

① par 노드의 자식 노드 수를 알아야 한다(par 노드 내의 자식 노드 수에 따라, 둘 이상의 자식 노드가 있다면 각 미디어 노드들은 par 블록과 시간관계를 형성하기 때문이다).

② par 노드의 전체 시간을 산출하여야 한다(par node 내의 각 자식 노드가 par 블록과의 시간관계를 형성하기 위해서 반드시 par 블록 전체의 재생시간을 계산하여야 한다. 이 때는 각 미디어의 지연시간(delay time)과 재생시간(playing time)을 합산하여 그 중 최대값을 취한다).

③ par 블록과 par 노드 내의 미디어 객체들의 관계를 설정한다(위의 과정을 통해 얻어진 시간 정보를 토대로 각 미디어 객체들을 TRN에 삽입하고, par 블록과의 시간관계를 설정하게 된다).

④ 만약 자식 노드의 수가 둘 이하일 경우엔, 블록의 전체시간과 각 미디어의 시간 정보를 가지고 두 미디어 사이의 시간관계를 산출하고, 각 미디어 객체를 TRN에 삽입하여 시간관계를 설정한 후 insertPar()를 종료한다.

표 3 insertPar 알고리즘

```
insertPar(Node node) // par 블록 삽입

begin
 node_count ::= getChildNodes(node).length;
 node ::= getFirstChildNode(node);
 // par 블록의 전체 시간 산출
 for node_count // par 블록의 자식노드의 수
 begin
 DurTime ::= node_begin_value + node_duration;
 if (MaxTime < DurTime)
 MaxTime ::= DurTime;
 node ::= getNextSibling(node);
 end

 node ::= getFirstNode(node);

 while (node != NULL)
 begin
 if (node_type == PAR)
 begin
 insertPar(node);
 node :: = getNextSibling(node);
 end
 else if (node_type == SEQ)
 node ::= getChildNode(node);
 else
 begin
 if (node_count > 2) // par 블록과의 관계 설정
 begin
 if (node_begin==0 && node_dur==MaxTime)
 .....
 // media 객체 삽입, 'equals' relation 설정;
 else if (node_begin==0&&node_dur<MaxTime )
 .....
 // media, delay 객체 삽입, 'starts' relation 설정;
 else if (node_begin + node_dur == MaxTime)
 .....
 // delay, media 객체 삽입, 'finishes' relation 설정;
 else if (node_begin + node_dur < MaxTime)
 .....
 // delay, media 객체 삽입, 'during' relation 설정;
 end
 else
 begin
 MediaObject mo1 ::=
 createObject(node_Attributes);
 MediaObjectmo2 ::=
 getNextSibling(node).node_Attribute;
 .....
 // 두 노드의 속성 정보를 통해 mo1, mo2 객체생성
 .....
 // mo1과 mo2의 시간 정보를 통해 시간관계 추출
 insertObject(); // 두 미디어 객체의 삽입
 insertRelation(); // 두 객체간의 시간관계 설정
 insertPar().exit //insertPar 종료
 end
 end

 node ::= getNextSibling(node);

 end while
end
```

4. 시스템 구성

이 시스템에서는 이미지, 비디오, 오디오, 텍스트, 텍스트 스트림 그리고 애니메이션과 같은 미디어 객체들에 대해 시간적 구성과 공간적 구성을 위한 다양한 편집 기능들을 제공한다. 다음은 시스템을 구성하고 있는 두 개의 주요 부분이다.

- 편집 시스템(Editing System)
- SMIL 객체 관리자(SOM; SMIL Object Manager)

그림 7은 전체 시스템 구조를 요약해서 보여주고 있다.

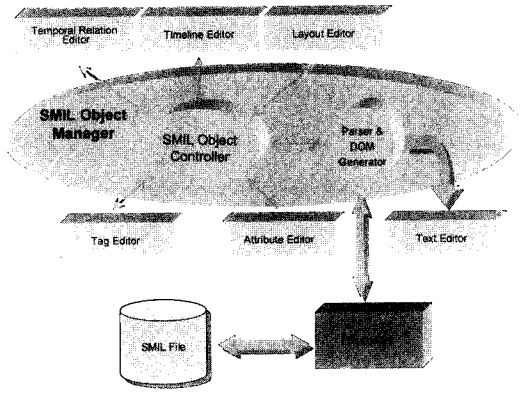

그림 7 전체 시스템 구성도

이 시스템은 J2SE(Java 2 Standard Edition)로 구현되었으며 또한, SMIL이 XML 기반의 언어로서 XML과 동일한 문법체계를 가지고 있으므로 XML 파서를 이용하여 구문을 분석할 수 있다. 이 연구에서는 SUN사(社)의 XML파서인 JAXP1.1를 사용하였다.

4.1 편집 시스템(Editing System)

편집 시스템은 개념적 시간관계(Conceptual Temporal Relation) 편집기, 타임라인(Time-line) 편집기, 배치(Layout) 편집기, 속성(Attribute) 편집기 그리고 텍스트(Text) 편집기로 구성되어 있다. 이들 편집기들 사이의 정보는 SOM을 통해 실시간으로 상호 교환됨으로써 효율적인 편집환경을 제공한다. 그림 8은 편집 시스템의 사용자 인터페이스를 보여주고 있다. 각 편집기들의 기능을 정리하면 다음과 같다.

- 시간관계 편집기(Temporal Relation Editor) : 프레젠테이션의 개념적 흐름을 아이콘을 이용해 추가, 수정, 삭제 등의 편집을 할 수 있도록 지원하며 시간적 흐름을 직관적으로 표현해 준다.
- 타임라인 편집기(Time-line Editor) : 각 미디어들의 시간을 절대시간 축을 이용해 표현하며 시작시간, 재생시간, 종료시간 등의 세부적인 편집을 지원한다.
- 배치 편집기(Layout Editor) : 미디어 객체들이 표현되어질 공간 정보에 대한 편집을 지원하며, 드래그 & 드롭(drag & drop) 기능을 지원하여 공간 편집을 쉽게 해준다.
- 태그 편집기(Tag Editor) : 멀티미디어 문서에서 사용하고 있는 SMIL 태그를 표현해 주는 편집기로 새로운 요소를 추가, 삭제, 또는 선택하는 기능을 제공한다.
- 속성 편집기(Attribute Editor) : 각 편집기에서 표현하지 못하는 미디어들의 세부적인 속성들에 대한 편집을 지원하여 해당 미디어 타입에 따른 정밀한 속성의 설정이 가능하다.
- 텍스트 편집기(Text Editor) : 각 편집기를 통해 교환되는 정보가 SOM을 통해 직접 텍스트 편집기로 갱신되므로 사용자로 하여금 자신의 입력에 대한 결과를 즉시 확인할 수 있게 하는 기능을 제공한다.

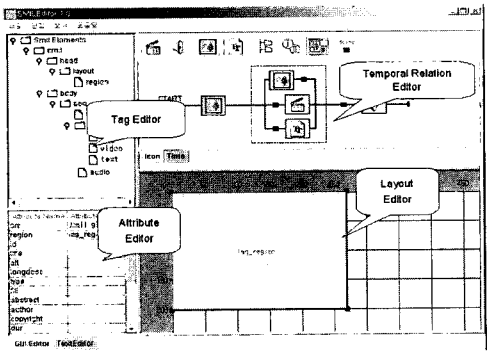

그림 8 전체 시스템

4.2 SOM(SMIL Object Manager)

그림 9는 이 시스템의 주요 부분인 SOM의 내부구조를 보여주고 있다. SOM은 프레젠테이션의 일관성을 위해 각 편집기로부터 정보를 수집해 실시간으로 다른 편집기에게 정보를 전달하는 책임을 지고 있으며 아래와 같은 두 부분으로 구성되어 있다.

- SMIL 객체 제어기(SMIL Object Controller; SOC) : 각 편집 인터페이스의 정보교환과 일관성 유지를 위해서 각 편집기는 SOC를 통해 정보를 교환한다. 즉, SOC는 모든 편집 인터페이스로부터의 정보를 수집하고 분석하는 일과 TRN에 대한 처리를 담당한다. SOM 중에서 TRN에 대한 연산을 담

당하는 부분으로 필요에 따라 적절한 지연 객체(delay object)를 삽입하는 기능 등을 수행한다. 또한, 이렇게 생성되거나 갱신되는 TRN의 정보를 DOM 생성기(DOM Generator)에 전달하는 역할도 한다.

- 파서와 DOM 생성기(Parser & DOM Generator): 이미 만들어진 SMIL 문서의 유효성을 파싱(parsing)을 통해 판단하여, DOM 생성기에서는 문서의 내용을 DOM 구조로 변환한다. 또한, 새로운 SMIL 문서를 저작하였거나 문서의 내용을 수정하였을 때는 SOC로부터 전달되는 정보에 맞추어 자동으로 생성되는 SMIL 문서에 대한 문법 검사도 수행한다.

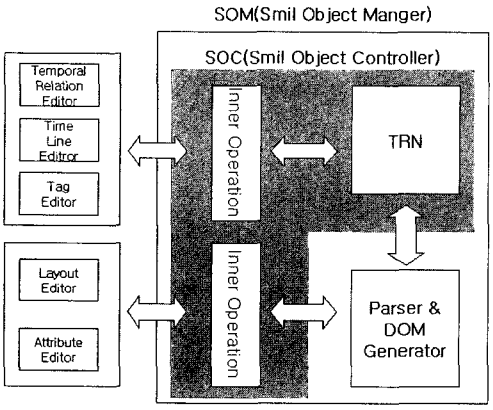

그림 9 SOM(SMIL Object Manager)의 내부구조

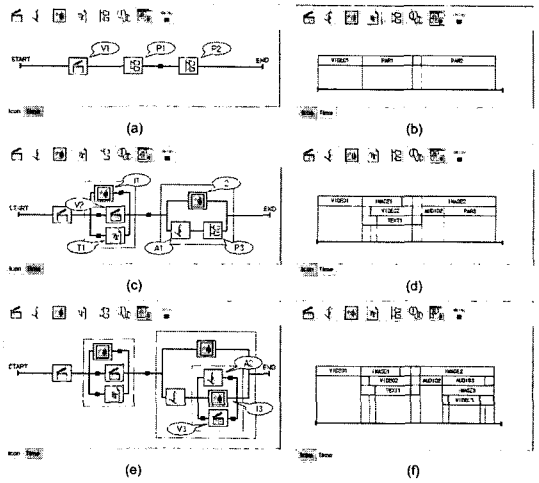

그림 10 멀티미디어 프레젠테이션 저작 예제

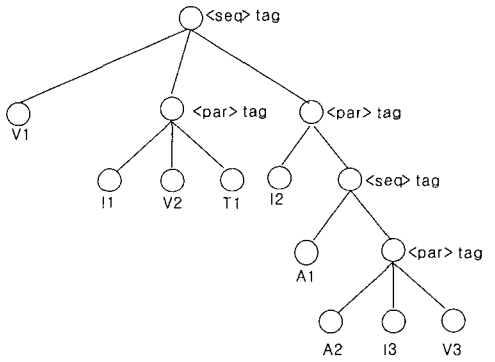

그림 11 예제에 따라 생성된 DOM 구조

5. 연구결과 적용 및 분석

여기서는 이 논문에서 제안하는 시스템을 활용하여 멀티미디어 문서를 작성한 활용 예 하나를 살펴보고 다른 저작 시스템들과 본 저작 시스템을 비교 분석한다.

5.1 적용 예

그림 10은 멀티미디어 프레젠테이션의 그래픽 표현을 보여주고 있다. 그리고 그림 11은 프레젠테이션의 저작이 완료되었을 때의 DOM 구조를 도식화한 것이다. 3장에서 살펴보았듯이 이 시스템은 사용자와의 상호 작용을 통해 실시간으로 DOM 구조 및 멀티미디어 문서를 갱신한다.

먼저, 저작자는 그림 10의 (a)에서처럼 프레젠테이션의 전체적인 뼈대를 구성한다. 첫 번째 객체인 비디오 객체는 V1이며, 첫 번째 블록 객체는 P1이고, P1과 V1의 시간관계는 'meets'로 설정되었다. 또한 P1과 두 번째 병렬 블록 P2는 'before' 관계로 설정되었다. 이 시스템에서는 새로운 객체가 삽입될 때마다 사용자에게 이전 객체와의 시간관계 설정을 요구하게 된다. 사용자가 두 객체간의 시간관계를 설정하지 않을 경우, 시스템은 기본적으로 'meets'관계로 두 객체간의 시간관계를 설정하게 된다. 또한 지연객체(delay object)를 미디어 객체들 간에 삽입하거나 삭제하는 방법 등으로 미디어들 사이의 시간관계를 수정할 수도 있다. 그림 10의 (b)는 (a)를 타임라인(Time-line) 형태로 표현한 것이다.

첫 번째 블록은 이미지 I1, 비디오 V2 그리고 텍스트 데이타 T1으로 구성하였다. 두 번째 블록은 이미지 I2, 오디오 A1 그리고 세 번째 병렬 블록을 포함하고 있다.

첫 번째 블록에서 저작자는 I1과 V2사이의 시간관계를 'overlaps'로 설정하였다. 이때 I2의 재생시간을 20초로 설정하였다. V2와 I1이 'overlaps' 관계를 가지고 있기 때문에 두 미디어 중 나중에 재생되는 미디어(V2)는

병렬 블록의 시작으로부터 일정시간의 지연을 가져야 하며, 사용자가 지정하게 되어있다. 또한 V2의 파일 재생시간(암시적 재생시간)은 25초이다. 이 예제에서는 5초로 설정하였다. 그리고 사용자가 V2와 텍스트 T1 사이의 시간관계를 'finishes'로 설정하였다. 'finishes' 관계에 있는 두 미디어들이 파일 재생시간을 가지고 있는 미디어들이라면 종료시간에 맞게 지연시간(delay time)이 설정되어 해당 미디어의 시작시간이 계산되지만, 텍스트와 이미지와 같은 미디어들은 파일 재생시간을 가지고 있지 않으므로 사용자가 임의의 지연시간을 설정해 주거나, 재생시간을 설정해 주어야 시작시간에 대한 계산이 이루어진다. 이 예에서는 T1의 재생시간을 20초로 설정하였다.

(c)의 두 번째 병렬 블록 P2의 내부는 이미지 I2와 오디오 A1, 그리고 세 번째 병렬 블록 P3로 구성하였고, I2의 재생시간은 40초이고, I2와 A1사이의 시간관계는 'starts'로 설정하였다. A1과 병렬블록 P3와는 순차적 관계인 'meets'로 정의하였으며, I2와 P3의 관계를 'finishes'로 설정하였다. 이때 A1과 P3가 'meets'이고, A1과 I2가 'starts', I2와 P3가 'finishes'이므로 순차관계에 있는 A1과 P3 전체는 I2와 'equals'의 관계가 설정된다. 따라서, 세 번째 병렬블록 P3의 재생시간은 이미지 I3의 재생시간과 A1의 재생시간의 차이가 된다.

(e)는 세 번째 블록 P3의 구성을 보여주고 있다. 사용자는 P3의 내부에 오디오 A2, 이미지 I3 그리고 비디오 V3로 구성하였다. A2와 I3 사이의 시간관계를 'starts'로 설정하고, I3의 재생시간을 20초로 정의하였다. 따라서 재생시간(playing time)이 15초로 20초보다 짧은 A2 뒤에는 지연(*delay*)을 의미하는 지연객체(*delay object*)가 삽입되었다. 이와 같은 경우 실제로 A2 뒤에 오는 지연객체는 실제 코드에는 적용되지 않고 시간관계 편집기에 표시만 된다. 그리고 저작자는 V3를 추가하고, I3와의 시간관계를 'during'으로 설정하였다. 두 미디어 간의 시간관계가 'during'일 경우 시스템은 사용자로부터 두 가지 값을 요구하게 된다. 먼저 I3의 재생시작 후 얼마 뒤에 V3가 재생을 시작할 지에 대한 지연시간(delay time)과 비디오의 재생시간(playing time)이다. 만약 V3에 대한 지연시간과 재생시간의 합이 이미지 I3의 재생시간보다 클 경우, 'during' 관계가 성립하지 않으므로 이러한 경우에는 미디어 객체의 'clip-begin'이나 'clip-end' 속성을 이용하여 V3의 재생시간을 수정하여야 한다. 이 예제에서는 V3의 지연시간(*delay time*)을 3초, 재생시간(playing time)을 12초로 설정하였다. 그림 7의 (f)는 (e)를 타임라인(Time-line) 형태로 표현한 것이다.

위에서 언급했듯이 미디어 파일의 파일재생시간(native duration)과 사용자가 요구하는 미디어의 재생시간(playing time)과 차이가 날 경우, 사용자는 'clip-begin'이나 'clip-end' 속성을 설정해 주어야 하며, 이는 속성 편집기(Attribute Editor)에서 편집을 지원하고 있다. 또한 미디어들의 repeat이라는 속성을 이용해서 하나의 미디어나 병렬블록 등의 반복 재생을 지정할 수도 있다.

다음은 이 예제를 위와 같은 과정을 통해 작업한 내용 중 미디어의 시간적 요소에 관련된 SMIL 문서의 <body> 태그의 내용들을 보여주고 있다.

```
<body>
 <seq>
 <video id="V1" region="video_region" src="intro.mpg" dur="20s" />
 <par>
    <img id="I1" region="img_region" src="choice.gif" dur="20s" />
    <video id="V2" region="video_region" src="move.mpg" begin="5s" dur="25s" />
    <text id="T1" region="text_region" src="testword.txt" begin="10s" dur="30s" />
 </par>
 <par begin="5s">
   <img id="I2" region="img_region" src="wind.gif" dur="40s" />
   <seq>
     <audio id="A1" src="wind.wav" dur="20s"/>
   <par>
     <audio id="A2" src="explain.wav" dur="15s"/>
     <img id="I3" region="img2_region" src="moon.gif" dur="20s" />
     <video id="V3" region="video_region" begin="3s" dur="12s" />
   </par>
  </seq>
 </par>
</seq>
</body>
```

5.2 다른 시스템과의 비교

다음 표는 본 저작도구의 성능 분석을 위하여 이미 상용화되었거나 상용화 예정인 SMIL 저작도구 중 SMIL Composer(Sausage Software)[16], GRiNS (Oratrix)[17], SMIL Editor ver.1.0(Rikei, 일본)[18], TagFree 2000 SMIL Editor(다산기술, 국내)[19]와 이 시스템을 비교해보았다.

표 4에서 나타난 대로 본 연구의 저작 시스템은 사용자에게 다양한 인터페이스를 제공하며 미디어 객체들의 시간적, 공간적, 논리적 편집을 가능하게 함으로써 좀더 빠르고 다양한 SMIL 프레젠테이션을 저작할 수 있게 해 주는 장점이 있다.

6. 결론 및 향후 연구

본 연구에서는 사용자에게 미디어 객체들 사이의 개념적 시간관계를 그래픽 인터페이스를 통한 직접 조작으로 수정하고 편집할 수 있는 SMIL 기반의 멀티미디어 저작 시스템을 설계하고 구현하였다. 또한, 멀티미디

표 4 다른 SMIL 저작도구와의 비교

	SMIL Composer (Sausage Software)	GRiNS (Oratrix)	SMIL Editor ver.1.0 (Rikei, 일본)	TagFree 2000 SMIL Editor (다산기술, 국내)	본 연구의 SMIL Editor
사용자 인터페이스	다양한 윈도우로 구성	타임라인만 제공	버튼으로 구성, 사용불편	다양한 윈도우로 구성	다양한 윈도우로 구성 (타임라인,아이콘,레이아웃,객체트리, 속성 등)
레이아웃 편집기능 (공간 편집)	전용 편집기 제공	없음	없음	전용 편집기 제공	전용 편집기 제공
시간적 편집기능	객체들의 시간관계가 직관적이지 않음	타임라인만 제공	없음	타임라인만 제공	타임라인 편집기 제공
논리적 편집기능	없음	없음	없음	직관적이지 않음	각 객체를 아이콘으로 나타내어 논리적으로 구성
실시간 코드 생성&확인	없음	없음	없음	직접 코딩	상태변화에 따라 실시간으로 수정
미리보기 기능	없음	자체 재생기 내장	미디어 편집도구 지원	없음	외부 재생기와 연동
사용환경	윈도우즈 계열	윈도우즈 계열	윈도우즈 계열	윈도우즈 계열	윈도우즈 및 유닉스 계열

어의 동기화와 관련된 개념적 시간 관계의 편집을 지원하는 직관적이고 효과적인 시간관계 기반의 인터페이스를 제안하였다. 아울러, 멀티미디어 프레젠테이션의 내부 표현으로 이 연구에서 새롭게 제안하는 TRN 구조를 사용할 것과 SMIL 문서로부터 TRN 구조를 자동으로 생성하는 알고리즘을 제안하였다.

본 시스템에서는 SMIL 객체 관리자(SOM; SMIL Object Manager)와 사용하기 쉬운 여러 편집기들을 이용하여 멀티미디어 프레젠테이션의 내용을 효율적으로 저작하거나 변경할 수 있다. 특히, 본 시스템은 멀티미디어 프레젠테이션을 생성할 때 사용자가 모든 미디어 객체의 시작 시간이나 재생 시간을 구체적으로 명시하지 않아도 미디어들 간의 시간관계와 지연 시간만을 설정함에 의해 프레젠테이션을 구성할 수 있게 해주는 장점을 가지고 있다. 또한, 사용자 인터페이스를 시스템 내부 자료구조인 TRN(Temporal Relation Network)과 동일한 형태로 제공함으로써 사용자의 요구에 따른 프레젠테이션의 흐름을 직관적으로 표현할 수 있게 해준다.

본 시스템은 표준 DOM 구조를 따르고 있기 때문에 멀티미디어 문서의 내부적 구조가 DOM 구조를 사용하는 일부 멀티미디어 시스템에 쉽게 적용될 수 있을 것이다. 따라서, MPEG-4 XMT(eXtensible MPEG-4 Textual Format)과 같은 형식을 지원하는 다양한 종류의 멀티미디어 프레젠테이션 저작도구에 응용될 수 있을 것이다[15].

앞으로는 이 논문에서 제안하는 미디어 사이의 개념적 시간관계 표현에서 SMIL 2.0 Recommendation에 정의된 이벤트 모델과 확장된 사용자 상호작용 메커니즘을 지원하는 연구를 계속할 것이다. 또한, 웹을 이용하는 공동 저작 메커니즘을 결합하여 원거리에 위치한 사용자들이 네트워크를 통해 멀티미디어 프레젠테이션을 공동으로 저작할 수 있는 시스템에 대한 연구를 진행할 것이다.

참 고 문 헌

[1] W3C, Synchronized Multimedia Integration Language (SMIL) 1.0 Specification, http://www.w3.org/TR/REC-smil/

[2] Rousseau, F., and Duda, A., "Synchronized Multimedia for the WWW," Computer Networks and ISDN Systems, Vol 30, No.11, April, 1998.

[3] W3C, W3C Issues SMIL as a Proposed Recommendation, http://www.w3.org/Press/1998/SMIL-PR

[4] W3C, W3c Synchronized Multimedia Working Group, http://www.w3c.org/AudioVideo

[5] W3C, Extensible Markup Language(XML), http://www.w3c.org/XML

[6] 성미영, 윤자천, "멀티미디어 프레젠테이션을 위한 시·공간 합성의 시각화", HCI '95 학술대회 발표논문집, 한국정보과학회 인간과 컴퓨터 상호작용 연구회지 4권 1호, 한국정보과학회, pp. 152-161, 1995.2.16-17.

[7] Ralf Steinmetz, Klara nahrstedt, "Multimedia : Computing, Communications & Applications," Prentice-Hall Inc, 1995.

[8] B. Prabhakarasn and S.V. Reghavan, "Synchronization Models For Multimedia Presentation With

User Participation," Proceedings on ACM Multimedia 93, Anaheim, California, 1993, pp. 157-166, August 1-6, 1993.
[9] N. U. Qazi, M. Woo, and A. Gahfoor, "Synchronization and Communication Model for Distributed Multimedia Objects," Proceedings on ACM Multimedia 93, Anaheim, California, 1993, pp. 147-155, August 1-6, 1993.
[10] James F., Allen, "Maintaining Knowledge about Temporal Intervals," communications of the ACM, pp. 832-843, November, 1983.
[11] Thomas. D. C. Little, Arif Ghafoor, "Synchronization and Storage Models for Multimedia Objects," IEEE Journal on Selected Areas on Communication, Vol. 13, No. 3, pp. 413-427, April, 1990.
[12] Thomas. D. C Little, Arif Ghafoor, "Multimedia Object Models for Synchronization and Databases," Proceedings of IEEE International Conference on Data Engineering, pp. 20-27, April, 1990.
[13] M. A Holliday and M. k. Vernon, "A generalized timed Petri net model for performance analysis," in Proceedings International Conference Timed Petri Nets, Torino, pp. 181-190, Italy, July, 1985.
[14] 노승진, 장진희, 성미영, "SMIL 저작도구를 위한 아이콘 기반의 동기화 표현 기법", 2001년도 한국정보과학회 봄 학술발표논문집(II), Vol. 28, No. 1, pp. 403-405, 수원 경희대학교, 2001.4.27-28.
[15] Mee Young Sung, Seung Jin Rho, Jin Hee Jang, "A SMIL-based Multimedia Presentation Authoring System and Some Remarks on Future Extension of SMIL," Proceedings of Packet Video 2002, Pittsburgh, Pennsylvania, USA, 11 pages, April 24-26, 2002, http://www.pv02.org
[16] http://www.rikei.co.jp/en/index.php3
[17] http://www.sausage.com/
[18] http://www.oratrix.com/GRiNS/
[19] http://www.tagfree.com/

노 승 진

2000년 인천대학교 전자계산학과 졸업(공학사). 2000년~2002년 인천대학교 컴퓨터공학과 대학원 졸업(공학석사). 관심분야는 멀티미디어 협동 컴퓨팅(multimedia collaborative computing), 네트워크, SMIL (Synchronized Multimedia Integration Language)

장 진 희

2002년 인천대학교 전자계산학과 졸업(공학사). 관심분야는 네트워크 게임 기술, SMIL(Synchronized Multimedia Integration Language)

성 미 영

1978년~1982년 서울대학교 식품영양학과 졸업(계산통계학과 계산학전공 부전공). 1985년~1987년 프랑스 INSA de Lyon 컴퓨터공학과 졸업(공학석사). 1987년~1990년 프랑스 INSA de Lyon 컴퓨터공학과 졸업(공학박사). 1990년~1993년 한국전자통신연구소 인공지능연구실 선임연구원. 1993년~현재 인천대학교 컴퓨터공학과 교수. 관심분야는 멀티미디어, 협동 컴퓨팅(multimedia collaborative computing), 가상현실