가변 버퍼와 상호대화형 객체 삽입을 이용한 페트리 넷 기반의 스트림 동기화 기법

이양민°, 이재기°
°동아대학교 컴퓨터공학과

Stream Synchronization Mechanism based on the Petri Net using Insertion Interactive Object and Variable Buffer

YangMin Lee°, Jaekee Lee°
°Dept. of Computer Engineering, Dong-A Univ.
E-mail :manson@thrunet.com

요 약

네트워크를 통한 미디어 서비스에 있어 다양한 미디어 서비스의 개발과 사용자들의 요구에 대응하기 위해 사용자와의 상호대화를 할 수 있는 미디어 전달 방법과 연속적인 재생을 보장하기 위한 버퍼정책이 요구된다. 지금까지의 관련된 연구에서는 여러 가지 다양한 방법을 통하여 동기화를 달성하고 있으나 상호대화라는 측면에서는 만족할 만한 해결책을 제시하지 않고 있다. 본 논문에서는 상호대화형 객체(Interactive Object)를 각 미디어 파일에 삽입하고 객체들이 서로의 정보를 이용할 수 있는 함수를 설계하여 동기화와 상호대화성이라는 문제를 해결하며 네트워크에 대한 의존성 때문에 발생하는 불연속적인 재생을 가변 버퍼를 이용함으로써 해결하였다.

1. 서론

디지털 매체의 발달과 네트워크의 대역폭 증가에 의해 컴퓨터를 이용한 여가 시간의 활용과 다양한 서비스의 제공이 이루어지고 있다. 이러한 서비스의 대표적인 예로는 실시간 인터넷 방송, 화상회의, 원격강의 등이 있으며 각 서비스들은 분산 환경이 급격히 부각되고 원격지를 제어할 수 있는 여러 기술들이 개발됨에 따라 단일 서버로부터 미디어들을 서비스 하던 기술이 다중 시스템으로부터 서비스를 할 수 있는 형태로 발전하였다. 다양한 응용들 중 실시간 방송과 같은 서비스를 보면 현재 기존의 기법은 단일 파일 내에 비디오, 오디오, 기타 미디어들을 묶어서 전송함으로써 실시간에 하나의 미디어만 조작해내야 하는 응용에는 한계를 가진다. 현재 미디어 서비스는 각 미디어를 따로 분리한 후 제공하는 추세이며 미디어 파일을 분리해서 제공할 경우 이 미디어들 간의 동기화가 핵심 문제로 부각 된다.[1] 뿐만 아니라 실시간에 사용자의 미디어 탐색과 같은 이벤트 발생을 위한 상호대화도 미디어를 분리하여 전송할 경우 분리된 미디어 간에 동기화가 문제가 된다. 더욱이 네트워크 특성상 미디어 스트림 중 특정 부분이 지연되어 전송 될 경우 전체 미디어의 재생이 불연속이 되는 문제도 존재한다. 본 논문에서는 상호대화성과 동기화 문제를 해결하는 방법으로

미디어 간의 시간 명세 기능이 뛰어난 패트리 넷(Petri Net)을 기반[2][3]으로 하고 정밀한 동기화를 이루기 위해 상호대화형 객체라는 개념을 도입하여 스트림에 삽입함으로써 두 가지 문제점을 해결하고자 한다. 또한 전송 받는 스트림들을 저장하는 버퍼를 상호대화형 객체 내에 포함된 지연값을 이용하여 가변적으로 운영 함으로서 불연속적인 재생을 최소화 하고자 한다. 본 논문에서는 2장에서 기존의 동기화 모델에 대한 고찰과 더불어 구성하고자 하는 시스템의 개요에 관해 설명하고 3장에서 시스템 내부의 동작 원리와 가변 버퍼의 역할에 대해서 설명한다. 그리고 4장에서 제안한 시스템의 효과 및 장점을 설명하고 향후 과제를 제시함으로써 결론을 맺는다.

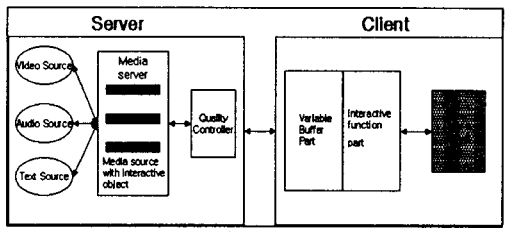
2. 가변버퍼를 이용한 상호대화형 패트리 넷

2.1 기존의 동기화 모델

미디어 동기화 기법들을 살펴보면 크게 두 가지 범주로 나눌 수 있다. 하나는 Allen이 제시한 13개의 시간 관계성[2]을 만족하게 하는 방법이며 다른 하나는 송, 수신측의 버퍼 정책[5][6]을 이용하는 방법이다. 시간 관계성을 명세하는 모델로는 모델로는 OCPN(Object Composition Petri Net)[3]이라고 하여 패트리 넷에 시간과 자원에 관한 명세를 추가하여 시간 관계에 대한 명세를 할 수 있도록 한 모델이 있으며, XOCPN(extended OCPN)[4]이라고 하여 기존의 OCPN을 확장한 모델이 있다. 또 다른 모델로는 실시간성을 최대로 강조한 모델인 RTSM(Real Time Synchronization Model)[7]이 있다. 수신측의 버퍼를 이용하는 방법으로는 버퍼 정책을[5][6]이용하여 버퍼를 유동적으로 조정함으로써 네트워크의 전송 지연으로 인한 미디어 간의 불일치를 적절히 보완하는 기법이 존재한다. 이러한 연구들의 문제점은 사용자와 미디어 스트림간에 발생하는 상호작용에 대한 처리 방법을 가지고 있지 못하다는 것이다. 다시 말하면 사용자가 미디어 스트림의 재생 중 특정 이벤트를 발생 시켰을 때 미디어 내의 동기화나 미디어 간의 동기화에 문제가 발생하게 되며 이는 미디어 내 프레임의 시간 관계성 및 표현 시간에 혼란을 발생시키게 된다는 뜻이다. 본 논문에서는 기존에 서술했던 "스트림 전송을 위한 패트리 넷 기반의 상호대화형 동기화 기법"[1]에 사용되었던 상호대화형 객체라는 아이디어를 바로 적용하도록 한다. 그리고 버퍼레벨[5]의 개념을 이용하여 네트워크 상태에 따라 적절히 버퍼의 크기를 조정함으로써 연속적인 미디어 스트림을 보장할 수 있도록 한다.

2.2 시스템 개요

제안하는 시스템의 아래의 (그림 1)에 나타나 있다. 서버에는 분리된 미디어 스트림을 저장하고 있는 미디어 데이터베이스와 미디어 스트림에 상호대화형 객체를 삽입하는 미디어 서버로 구성되며 클라이언트에는 서버로부터 전송되는 상호대화형 객체가 삽입된 미디어 스트림을 원래의 미디어와 상호대화형 객체로 분리하고 이 객체의 정보를 해석하는 상호대화형 함수 부분과 네트워크 상태에 따라 그 크기가 변하는 버퍼 부분으로 구성된다. 물론 사용자가 미디어를 재생하여 볼 수 있는 뷰어(viewer)도 클라이언트의 일부분이다.

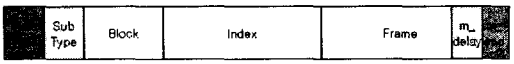


(그림 1) 시스템의 개요

2.3 상호대화형 함수(Interactive function)의 기능

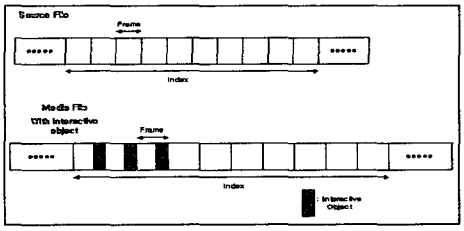
상호대화형 함수는 기존의 연구들이 가지지 못하였던 사용자와 미디어 스트림간의 상호대화를 가능하도록 하기 위해 몇 가지의 프로세서를 수행하거나 처리하는 역할을 한다. 상호대화형 함수는 크게 4개의 함수로 나눌 수 있으며 각 함수의 이름들은 매핑(Mapping), 해쉬(Hash), 동기화(Synchronization) 내부통신(Inter communication) 함수이다. 이러한 함수들의 구체적인 동작은 뒷 절에서 설명하도록 하겠다. 본 논문에서는 상호대화 및 동기화를 달성 하기 위해 기존 연구의 아이디어 였던 상호대화형 객

체(interactive object)를 그대로 사용한다. 이 객체는 상호대화와 동기화에 필요한 정보를 포함하는 객체로 아래 (그림 2)와 같은 데이터 포맷을 가지며 미디어 지연값(m_delay)필드가 새롭게 추가 되었다. 미디어 지연값을 이용하여 네트워크 상태에 따라서 지연이 심하게 되는 미디어 스트림의 패킷을 드롭할 수도 있고 버퍼에 저장할 수도 있으며 또한 버퍼의 크기 변화를 제어할 수 있다.



(그림 2) 상호대화형 객체의 포맷

(그림3)은 미디어 스트림에 상호대화형 객체를 삽입한 것으로 미디어의 종류에 따라 그 삽입 빈도를 다르게 책정할 수 있다. 즉 미디어 스트림의 크기나 동기화의 요구 수준 등에 따라 상호대화형 객체의 삽입 빈도를 조정할 수 있다는 것이다.



(그림 3) 객체를 삽입한 미디어 스트림

2.4 해쉬 함수(Hash function)

주 미디어 스트림 내에서 원하는 특정 프레임 위치를 찾아주는 함수이며 상호대화형 함수들 중 제일 처음으로 발동된다. 클라이언트는 수신되는 미디어 파일 중에서 상호대화형 객체의 인덱스와 프레임 정보를 가지고 있다가 이 정보를 이용하여 서버측에 주 미디어의 특정 위치를 전송하도록 요구할 수 있다. 서버측은 상호대화형 객체가 가지고 있는 특정 프레임 정보와 일치되는 정보를 찾기 위해 미디어 스트림의 프레임을 탐색한 후 필요한 정보를 전송하게 된다. 간단히 말해서 해쉬 함수의 기본적인 역할은 미디어 스트림 내의 특정 프레임 탐색이다.

2.5 매핑 함수(Mapping function)

해쉬 함수의 동작 후 결과가 나오게 되면 이 값은 주 미디어 스트림 내의 특정 프레임 위치이다. 이 위치 값을 이용하여 주 미디어 외의 다른 상대 미디어들 내의 적절한 위치를 찾아주는 것이 매핑 함수의 역할이다. 매핑 함수는 주 미디어의 위치가 결정되면 상호대화형 객체 내부의 값을 읽어 동기화가 이루어져야 할 상대 미디어의 특정 프레임 값을 찾아서 돌려준다.

```
Mapping_Function(i_Object)
Tag Ti; //interactive object 내부의 Tag
Ti = Search(i_Object); // 객체 검색
SendMainMediaFrom(i_Object); //주 미디어
  에서 객체 검색 후 전송
SendSubMediaFrom(Ti); //상대 미디어에서
  객체 검색 후 전송
```

(그림 4) 매핑 함수

2.6 동기화 함수(Synchronization function)

기본적인 동기화 함수의 기능은 두 가지이다.

```
Synchronization_Function
begin
    if correct frame not founded then
       begin
         if wait time > delay time then
            begin
               reduce media's quality;
            end
       call variation function
       end
    if current_skew > skew then
     begin
          wait (some Index);
          check(skew);
          if current_skew > skew then
            begin
              Inter-communication function;
            end
          else
            begin
                wait
            end;
     end
end
```

(그림 5) 동기화 함수

첫번째는 네트워크 로드의 증가로 인해 필요한 미디어의 전송이 장시간 지연될 경우 미디어 지연값을 이용하여 이것을 감지하고 미디어의 품질을 감소 시

키면서 버퍼를 변화시키는 변형(variation)함수에 미디어가 지연되고 있음을 알리는 역할을 해주는 것이며 두 번째는 예측할 수 없는 오류로 인해 미디어 간의 불일치가 한계값(skew)을 넘어설 경우 이것을 적절히 처리할 수 있는 역할을 하는 것이다. 실질적인 동기화 동작은 본 논문에서 제안하고 있는 시스템을 사용할 경우 다른 처리 없이 해쉬 함수와 매핑 함수에 의해 자동적으로 달성된다.

2.7 내부통신 함수(Inter-communication function)

이 함수는 상호대화형 객체 사이에서의 통신을 담당하고 있다. 미디어 스트림 간의 프레임 위치 정보를 서로 비교하여 스큐값을 계산하고 객체들의 값이 변경되어야 할 경우 객체 내부의 값을 수정하는데 사용된다. 동기화 함수는 내부 통신 함수의 기능을 이용하여 미디어 스트림 내에 삽입되어 있는 상호대화형 객체의 정보를 수정한다.

2.8 Quality Controller

동기화 함수의 동작 결과로 미디어의 품질을 감소시켜야 할 경우 이 품질 제어기가 동작한다. 즉 네트워크 로드에 의해 미디어의 전송 품질을 저하 시키거나 저하 시킨 품질을 상승 시키기 위한 부분으로 전송되는 패킷의 크기를 줄이거나 영상 등의 미디어에는 픽셀을 줄이는 역할을 담당하는 부분이다. 클라이언트에서 특정 시간 동안 미디어를 수신하지 못하였다는 것을 나타내는 미디어 지연값에 의해 동작하며 이 미디어 지연값은 동기화 함수에 의해 감지 된다.

2.9 가변 버퍼 (Variable Part) 변형 함수(Variation function)

네트워크의 특성상 많은 트래픽이 특정 서버나 회선에 집중될 경우 서비스 중인 미디어 스트림 중에 일부가 지연되어 전송되는 경우가 발생하게 된다. 본 논문에서 제안하는 시스템의 특성상 분리되어 전송되는 미디어 스트림 중에 일부가 지연되어 전송된다면 이 지연되는 미디어 패킷의 도착을 기다리거나 아니면 드롭시키는 경우가 발생할 수 있다. 이런 경우에 있어 무계획적인 대기 또는 드롭을 막기 위해 미디어 지연값(m_delay)를 이용하여 재생되는 시점이 조정될 수 있도록 버퍼의 크기를 유동적으로 변화시켜 불연속적인 재생을 최소화 하는 동작을 하는 부분이다. 변형 함수에는 두 가지의 프로시져가 포함 된다. 첫 번째 프로시져는 버퍼를 확장, 축소 시키는 프로시져이며 (그림 6)에 버퍼 확장 프로시져가 나와 있다.

버퍼 확장 프로시져

B = Q * T

Q : 미디어 스트림의 패킷 크기

T : 시간 B : 버퍼의 증가분

(단 T < 200ms, Q 즉 패킷의 크기는 미디어의 종류나 응용에 따라 다름)

(그림 6) 버퍼 확장 프로시져

두 번째 프로시져는 미디어의 프레젠테이션 속도를 증가, 감소 시키는 프로시져로 (그림 7)에 나와 있다.

프레젠테이션 속도 제어 프로시져

$Vp = skew * (1 /(B_{Tot} - S_c))$

Vp : 프레젠테이션 속도 (presentation speed)

B_{Tot} : 버퍼 전체 크기 (Total buffer size)

S_c: 현재 스트림 양(current stream size)

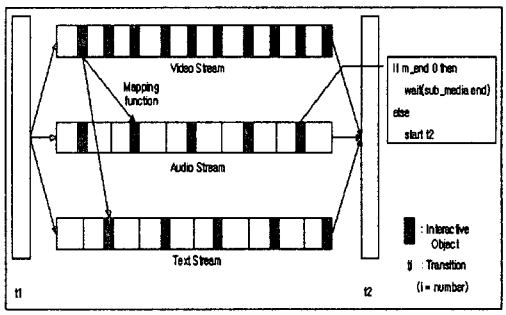
(그림 7) 프레젠테이션 속도 제어 프로시져

3. 시스템 동작 순서 및 함수들의 동작

3.1 상호대화형 객체와 시스템의 동작

스트림 간의 동기화를 위한 동작들은 대부분 서버와 클라이언트 양쪽이 서로 통신을 하여 수행하며 상호대화형 객체들도 서로의 정보를 이용하기 위해 통신을 하도록 구성되어 있다. 서버측에서 서비스할 미디어 품질을 결정하고 상호대화형 객체를 삽입하여 전송할 미디어를 준비한다. 만약 클라이언트에서 상호대화 동작을 발생하는 이벤트가 일어나면 클라이언트가 미디어 스트림을 해석하면서 가지고 있던 상호대화형 객체의 정보를 이용하여 서버측으로 요

청 정보를 요구하게 된다. 클라이언트는 요청 정보를 전송하기 위해서 해쉬 함수를 발동시키고 탐색 버튼의 위치에 알맞는 상호대화형 객체를 찾아서 서버로 전송한다. 서버는 클라이언트측의 요청 정보를 수신하고 이 정보에 따라서 해쉬 함수를 발동시킨다.



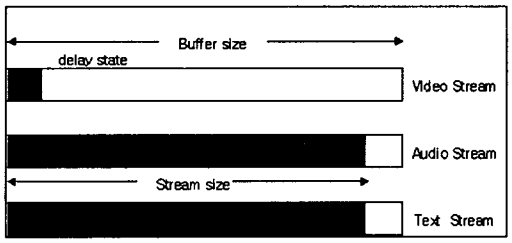
(그림 8) 각 함수들의 동작

발동된 해쉬 함수에 의해 주 미디어의 프레임이 탐색 된다. 그리고 해쉬 함수에서 탐색한 상호대화형 객체 정보를 이용하여 매핑 함수를 발동한다. 매핑 함수에 의해서 나머지 상대 미디어들의 프레임 위치가 결정되고 결정된 위치로부터 미디어 스트림들이 클라이언트로 전송된다. 마지막 동작은 이렇게 전송되어 오는 스트림들을 동기화하여 재생하는 것이다. 패트리 넷에서의 트랜지션 전환은 설정되어 있는 미디어 완료값(m_end)을 이용해서 수행한다.

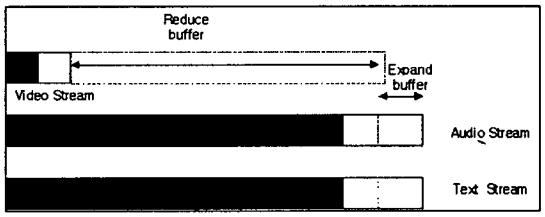
3.2 가변형 버퍼의 동작

가변형 버퍼는 제안된 시스템의 클라이언트측에 포함된다. 미디어 지연값을 이용하여 변형 함수가 버퍼의 확장 또는 축소를 계산하게 되고 이것에 의해서 실제로 각 미디어의 버퍼 크기가 유동적으로 변화된다. 버퍼의 크기를 유동적으로 변경하게 되면 미디어 스트림의 플레이 아웃(play out)시간을 의도한 대로 조정할 수가 있다. 즉 미디어 스트림은 버퍼에서 빠져 나오면서 재생되는 것이며 본 논문에서 제안한 시스템에서도 마찬가지로 동작한다. 가변형 버퍼는 다음과 같이 네트워크의 상태에 따라 버퍼에 저장되어 있는 미디어 스트림들의 양이 변경되는 상황에 적절히 대응할 수 있다. 만약 네트워크의 문제로 인해 한 미디어의 패킷이 늦게 전송되면 다른 미디어들은 이 미디어 패킷의 도착을 기다리게 되고 이 때 버퍼에 쌓이게 되는 패킷이 늘어나게 된다. 이러한 현상이 발생한 후 시간이 지나면 버퍼 오버플로우가 발생한다. 일반적으로 비교적 용량이 적은 오디오는 충분히 확보가 되었으나 그 시점에서 동기화 되어야 할 비디오 스트림이 충분히 확보되지 못하여 오디오는 플레이 아웃 상태가 될 가능성이 있고 비디오는 아직 그렇게 되지 못한 경우에 일어난다. 이 상태에서 비디오는 버퍼 언더플로우 상태가 될 확률이 있고 다른 미디어의 버퍼들은 오버플로우가 될 상황이 된다. 이러한 상태의 감지는 각 미디어 스트림에 삽입되어 있는 상호대화형 객체의 프레임 정보를 이용하여 할 수 있다. 이러한 문제를 해결하기 위해 가변형 버퍼를 사용한다. 가변형 버퍼의 기본적인 목적은 크기를 유동적으로 변경함으로써 미디어의 플레이 아웃 시간을 의도적으로 조정하는 것이다. 즉 버퍼 오버플로우가 일어날 경우 패킷들이 드롭되는 현상이 발생하므로 스큐값 200ms이내에서 버퍼를 최대한 확장하는 것이다. 물론 버퍼의 최대 임계치와 최소 임계치는 구성 시스템의 사양에 따라 차이가 있다. 그 후에 200ms 이내의 스큐값에서는 사용자가 미디어간의 불일치를 인식할 수 없으므로 이 시간 동안 버퍼를 최대한 확장하였다가 이 시간이 지나면 다른 미디어 버퍼에 있는 스트림들은 강제적으로 재생하여 버퍼를 비운다. 이 때 지연되고 있는 미디어의 패킷들은 드롭을 하게 된다. 이런 경우에는 피할 수 없는 불연속적인 재생이 일어나지만 가변형 버퍼를 두지 않는 것에 비해 많은 수의 프레임을 재생할 수 있으리라고 본다. 또한 이 시점에서 미디어의 프레젠테이션 시간을 미디어간 스큐값에 의존하여 계산함으로서 오버플로우가 될 상태에 있는 다른 미디어들의 프레젠테이션 시간을 짧게 하여 최대로 불연속을 느낄 수 없도록 재생을 한다. 비디오의 버퍼는 기다리던 패킷을 드롭하고 현재 버퍼를 최소화하여 패킷이 도착하는 즉시 플레이 아웃 할 수 있는 상태로 만들어 준다. 이 때 역시 프레젠테이션 시간을 의도적으로 짧게 만들어야 하며 이것을 이용하여 불연속적인 재생을 최소화 한다.

즉 늦게 재생될 미디어의 재생 속도를 일시적으로 증가시켜 네트워크 지연에 의한 미디어 간의 불일치를 최대한 흡수한다. 아래 (그림 9-a)는 버퍼의 크기가 변형 되기 전을 보여 주며 (그림 9-b)는 변형 함수가 동작하고 난 후의 그림이다.



(그림 9-a)



(그림 9-b)

(그림 9) 가변 버퍼의 동작

4. 결론 및 향후 과제

본 논문은 상호대화형 객체를 미디어 스트림에 첨가하여 미디어간의 동기화를 달성하며 사용자와의 상호대화성을 이룬다. 여기에 패트리 넷을 이용함으로써 트랜지션간 또는 트랜지션 내의 상호작용 및 동기화를 달성할 수 있다. 또한 가변형 버퍼 정책을 적용함으로써 네트워크 상태에 따른 미디어 스트림의 지연을 흡수하여 불연속적인 재생을 최소화하고 있다. 본 논문에서 제안한 모델은 패트리 넷을 이용한 다른 모델과 비교하여 Allen의 시간 관계를 명세할 수 있다는 점은 동일하다. 그러나 미디어 스트림에 상호대화형 객체를 삽입하기 위해 미디어를 분할해야 되고 또한 직접적으로 삽입하는데 있어서 부가적인 수행 시간과 각 함수들이 동작하는 부가 시간이 필요하다. 그러나 사용자와의 상호대화 동작이 일어난 후 동기화를 취하는 점에 있어서는 스큐값을 20ms 이하로 할 수 있을 만큼 우수하며 이러한 우수성은 제안된 함수의 기능들이 적절히 동작함으로서 이루어진다. 또한 가변형 버퍼를 이용함으로써 네트워크 상태에 따라 폐기될 수 있는 많은 패킷들을 적절히 보존할 수 있어 불연속적인 재생을 최소화 할 수 있는 장점을 가진다. 뿐만 아니라 상호대화 동작이 발생한 후에 동기화를 수행하는 점에 있어서는 상대적으로 매우 우수하다는 것을 알 수 있다. 또한 기존의 모델들은 스큐값이 불안정하며 서비스에 따른 차별적인 정책을 취할 수 없지만 본 논문에서 제안한 기법은 서비스에 따라서 다양하게 차별적인 정책을 취할 수 있는 것도 장점이다. 향후 연구 과제로는 가변형 버퍼의 최대, 최소 크기 결정 및 상호대화형 객체를 적절히 처리할 수 있는 프로세서를 결정할 수 있도록 여러 시스템에서의 실험이 필요할 것이며 상호대화형 함수 및 변형 함수의 보다 좋은 최적 알고리즘을 설계해야 할 것이다.

[참고문헌]

[1] 이양민, 이재기, " 스트림 전송을 위한 패트리 넷 기반의 상호대화형 동기화 기법" , 한국정보처리학회 추계학술발표논문집 8권 2호, pp.1517-1520, 10.12. 2001

[2] Allen, J.F, " Maintaining Knowledge About Temporal Intervals" , CACM, 11, vol. 26, pp.832-843, 1983.

[3] Naveed U. Qazi " A Synchronization and Communication Model for Distributed Multimedia Objects " Proc. Of the First ACM Conference on Multimedia, pp.147-155,Aug. 1993.

[4] Lynda Hardman Dick C. A. Bulterman, " Composition and Linking in Time-based Hypermdia" , European Union ESPRIT Chameleon project,1999

[5] 성경상, 황민구, 이기성, 이근왕, 오해석," 버퍼레벨을 이용한 적응형 멀티미디어 동기화 재생 기법, 한국정보처리학회 춘계학술발표논문집, 제8권, 제1호, pp.619-622, 2001.

[6] 이기성, 이근왕, 이종찬, 오해석, " 대기시간을 이용한 적응형 멀티미디어 동기화 기법" , 한국정보처리학회 논문지, 제7권 제2호, pp.649-655, 2000.2.

[7] 박영숙, 이승원, 정기동," 분산 멀티미디어 응용을 위한 실시간 동기화 메커니즘" , 한국정보처리학회 논문지, 제7권, 제12호, pp.3785-3793, 2000. 12