스트림 전송을 위한 패트리 넷 기반의 상호대화형 동기화 기법

이양민*, 이재기*
*동아대학교 컴퓨터공학과
e-mail : manson@thrunet.com

Interactive Synchornization Mechanism based on the Petri Net for the Stream Transmission

Yang-Min Lee* · Jae-Kee Lee*
*Dept. of Computer Engineering, Dong-A University

요 약

과거의 컴퓨터를 이용한 미디어 서비스는 사용자에게 단순히 비디오, 오디오, 텍스트 등의 미디어를 일방적으로 전달하였으나 현재의 서비스는 사용자와의 상호대화 및 필요한 미디어 만을 선택해서 전달할 수 있는 방식을 요구한다. 이러한 응용을 위해서 각 미디어 파일들을 분리하여 전달하는 방식이 필요하며 동기화와 더불어 상호대화성 이라는 두 가지 문제를 해결해야 한다. 지금까지의 관련 연구에서는 시간축, 패트리 넷(Petri Net), 버퍼 조작 등의 방법을 통하여 동기화를 달성하고 있으나 상호대화라는 측면에서는 만족할 만한 해결책을 제시하지 않고 있다. 본 논문에서는 패트리 넷 모델을 이용하고 상호대화형 객체(Interactive Object)를 각 미디어 파일에 삽입하여 이 객체들이 서로의 정보를 이용할 수 있는 함수를 설계함으로서 동기화와 상호대화성이라는 문제를 해결하였다.

1. 서론

네트워크의 속도가 증가와 디지털 정보 전달 매체의 발달에 따라 인터넷과 컴퓨터를 이용한 실시간 방송, VOD, 원격 화상회의, 원격 강의 등의 다양한 응용들이 시도 되고 있다. 또한 분산 환경이 급격히 부각되고 원격지를 제어할 수 있는 여러 통신 기술들이 발달하여 단일 서버로부터 미디어들을 서비스 받던 기술이 다중 시스템으로부터 서비스를 받을 수 있는 형태의 서비스로 발달하게 되었다. 특히 실시간 방송과 같은 서비스를 보면 현재 기존의 방송들은 단일 파일 내에 비디오와 오디오 및 기타 미디어들을 묶어서 방송함으로써 실시간에 하나의 미디어만 조작해내야 하는 응용에는 한계가 있다. 예를 들면 영화를 시청하고 있던 중에 한글 음성을 영어 음성으로 바꾸고자 하는 등의 서비스에서는 단일 미디어 파일로는 적절히 서비스 할 수가 없는 게 사실이다. 이러한 특정 서비스를 제공하기 위해서 미디어 파일들을 분리하여 제공하는 것이 현재의 추세이며 각 미디어 파일을 분리해서 제공할 경우 이 미디어들 간의 동기화가 핵심 문제로 부각되게 된다. 뿐만 아니라 실시간에 사용자의 미디어 파일 탐색이나 특정 이벤트 발생을 위한 상호대화도 파일들을 분리하여 전송할 경우 문제가 된다. 이 논문에서는 이러한 상호대화성과 동기화 문제를 해결하기 위해 패트리 넷(Petri Net)과 상호대화형 객체(interactive object)를 이용하였다.

2. 상호대화형 패트리 넷

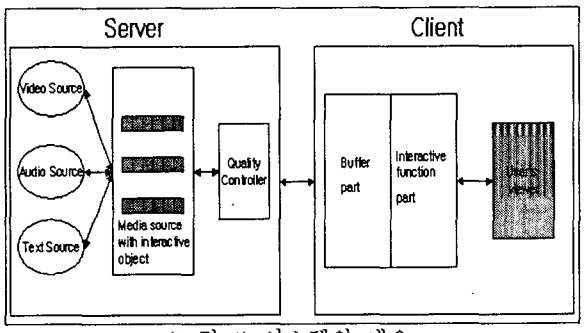
2.1. 기존의 동기화 모델

기존의 동기화 모델들을 살펴보면 일반적으로 Allen 이 제시한 13 개의 시간 관계성[1]을 표현하여 만족하게 하는 경우이거나 수신측의 버퍼 정책을 이용하여 동기화를 수행하는 종류이다. 시간 관계성을 명세할 수 있는 모델로는 OCPN(Object Composition Petri Net)[2]과 XOCPN(extended OCPN)[3] 등이 있다. OCPN 은 패트리 넷에 시간과 자원에 관한 명세를 추가하여 시간 관계에 대한 명세를 할 수 있도록 하며

XOCPN 은 OCPN 모델을 확장한 것으로 동기화 객체를 삽입하여 각 미디어들 간의 동기화를 이 객체가 수행하도록 한다. 동기화 방법에는 단순히 시간 축[4]을 기반으로 하는 방법도 존재하는데 이 모델의 경우에는 다중 미디어에 대한 동기화를 명세할 수 없다는 문제점이 있어서 시간 축에 따라 미디어들을 순차적으로 나열할 때 사용된다. 수신측의 버퍼를 이용하는 방법으로는 버퍼 레벨[5][6]을 이용하여 수신측의 버퍼를 유동적으로 조정함으로써 네트워크의 전송 지연으로 인한 미디어 간의 불일치를 적절히 보완하는 방법을 사용하고 있다. 기존의 연구들을 살펴 보면 시간 관계성을 명세한 연구들은 실시간성이나 미디어들의 동기화[7]에 관한 내용이 충실한 것에 반해 사용자의 입력 즉 상호대화가 일어날 경우 그 시간의 명세가 무너지는 단점을 가지고 있고 또한 XOCPN 같은 모델들은 모든 미디어의 종류를 같은 방법으로 취급하여 미디어의 종류에 따른 특성을 무시하는 경향이 있다. 이러한 경우에는 네트워크 전송지연에 의해 동기화에 문제가 발생할 수도 있고 사용자와의 대화식 응용에 적합하지 않을 수도 있다. 이에 본 논문에서는 패트리 넷에 다음과 같은 함수와 정의를 추가하여 동기화와 상호 대화성을 충족 시켰다.

2.2. 시스템 개요

구축해야 하는 시스템은 다음 (그림 1)과 같다. 서버측에는 미디어 데이터베이스나 미디어 원본을 가지고 있는 시스템과 미디어 원본에 상호대화형 객체를 삽입하는 미디어 서버로 구성된다. 미디어 서버에는 유동적으로 서비스 되는 미디어 파일의 품질을 조정할 수 있는 기능과 상호대화형 객체를 삽입할 수 있는 기능이 구축된다. 클라이언트측에는 객체가 삽입된 미디어 스트림(Media Stream)을 해석할 수 있는 기능과 미디어 스트림을 저장 할 버퍼가 구축된다.



(그림 1) 시스템의 개요

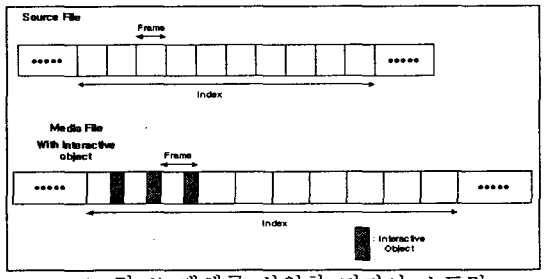
2.3. 상호대화형 함수(Interactive function)의 기능

내부에 각기 다른 기능을 하는 4 개의 함수가 포함되어 있다. 매핑(Mapping), 해쉬(Hash), 동기화(Synchronization), 내부통신(Inter-communication) 함수들이 하부 함수인데 이러한 함수들의 기능은 2.4, 2.5, 2.6, 2.7 절에서 각각 설명하도록 하겠다. 이러한 함수들의 내부 동작 외에도 상호대화형 함수가 가져야 하는 몇 가지 기능이 있다. 첫 번째로 객체의 삽입 방법과 빈도를 결정하는 프로시져가 포함되고 둘째 인덱스(index), 프레임(frame) 값의 설정 및 제어를 할 수 있는 프로시져가 포함된다. 마지막으로 패트리 넷에서의 트랜지션의 전환과 그 사이에서의 동기화 및 상호대화를 위한 프로시져가 포함된다. 본 논문에서는 상호대화 및 동기화를 달성 하기 위해서 필요한 정보를 포함하는 객체를 생성하였는데 상호대화형 객체(interaction object)로 다음과 같은 데이터 포멧을 갖는다.

```
<Main_Type>γ
<Index><Frame>{<Sub_type>}*<m_end>
<Sub_type>γ<m_type><Block>
<m_type>γ video|sound|text
<Block>γ<Index><Frame>
<Index>γ<number>
<Frame>γ<number>
<m_end>γ<number>
```

(그림 2) 상호대화형 객체의 포멧

아래의 (그림 3)은 미디어 스트림에 상호대화형 객체를 삽입한 것으로 미디어의 종류에 따라 그 삽입 빈도를 다르게 책정할 수 있다. 객체의 삽입 빈도는 전체 시스템의 수행 속도와도 관련이 있다.



(그림 3) 객체를 삽입한 미디어 스트림

2.4. 해쉬 함수(Hash function)

주 미디어의 프레임 위치를 찾아주는 함수로서 상호대화형 함수의 하부 함수 중 제일 처음으로 발동된다. 클라이언트측은 수신되는 미디어 파일 중에서 상호대화형 객체의 인덱스와 프레임 정보를 가지고 있다가 이 정보를 이용하여 서버측에 주 미디어의 특정 위치를 전송하도록 요구할 수 있다.

```
Hash_Function(bool i_signal)
  begin
    if i_signal then
      begin
        Search(i_Object) ;
        Invoke Mapping_Function ;
      end
    else
      begin
          wait(i_Signal);
      end
  end
```

(그림 4) 해쉬 함수

2.5. 매핑 함수(Mapping function)

주 미디어 외의 상대 미디어에 특정 부분 즉 인덱스와 프레임을 찾아주는 함수로 주 미디어 내의 상호대화형 객체의 값을 읽어 타 미디어의 특정 상호대화형 객체를 찾아주는 함수이다.

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

(그림 5) 매핑 함수

2.6. 동기화 함수(Synchronization function)

실질적으로 동기화에 대한 부분은 이 논문에서 제안된 방법을 사용하면 자동으로 이루어진다. 단 네트워크 로드의 증가로 인해 필요한 미디어의 전송이 장시간 지연될 경우 처리해주는 프로시져와 예측할 수 없는 오류로 인해 미디어간의 불일치가 한계값(skew)을 넘어설 경우 처리할 수 있는 프로시져를 가지고 있다.

```
Synchronization_Function
begin
    if correct frame not founded then
        begin
            if wait time > delay time then
                begin
                    reduce media's quality;
                end
        end
    if object has incorrect value then
        begin
            Inter-communication 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
```

(그림 6) 동기화 함수

2.7. 내부 통신 함수(Inter-communication function)

해쉬 함수가 상호대화형 객체를 찾거나 또는 동기화 함수에 의해 객체들의 값이 변경되는 경우에 사용되는 함수로 객체 간의 통신을 담당하는 기능을 가진다.

2.8. Quality Controller

네트워크 로드에 의해 미디어의 전송 품질을 저하시키거나 저하 시킨 품질을 상승 시키기 위한 부분으로 전송되는 패킷의 크기를 줄이거나 영상 등의 미디어에는 픽셀을 줄이는 역할을 담당하는 부분이다. 클라이언트측에서 특정 시간 동안 미디어를 수신하지 못하였다는 것을 나타내는 미디어 지연(media_delay)값에 의해 동작한다.

3. 상호대화성 및 동기화의 달성

3.1. 상호대화성(Interaction)

상호대화성은 상호대화형 객체를 이용해서 이루어진다. 클라이언트측에서 수신한 주 미디어 스트림의 상호대화형 객체의 값을 기억하고 있다가 사용자의 상호작용이 발생하면 객체의 정보를 기준으로 서버측에 주 미디어의 필요한 인덱스와 프레임을 요청하게 된다. 서버측에서 적절한 인덱스와 프레임을 포함한 객체를 찾게 되면 이 객체를 중심으로 상대 미디어들의 특정 인덱스와 프레임을 포함하고 있는 객체를 찾기 위해 매핑 함수를 이용한다. 그리고 주 미디어와 상대 미디어들의 전송이 이루어진다. 클라이언트 측에서는 이 미디어 스트림들을 받아서 다시 재생을 한다. 상호대화형 기법에서 가장 문제가 되는 부분이 상호대화형 객체의 삽입 방법과 그 빈도의 문제이다. 과도한 삽입은 본래 소스 미디어 파일에 비해 과도한 오버헤드를 가질 수 있으며 이로 인해 본래의 미디어 파일과 상호대화형 객체를 분석하는 부분에서의 지연이 발생할 수 있다. 삽입 빈도가 너무 낮은 경우에는 비교적 높은 동기화 수준과 상호대화성 수준을 유지할 수 없기 때문에 적절한 삽입의 빈도를 찾는 것이 제일 중요하다.

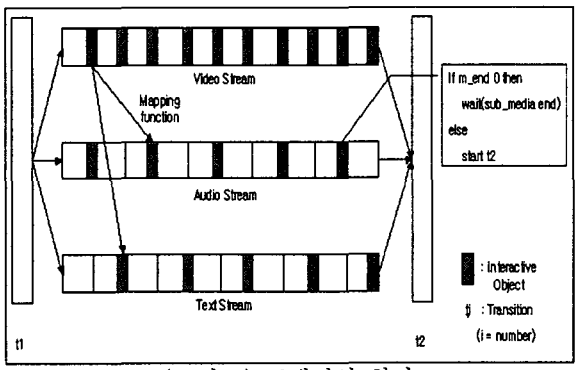
3.2. 동기화(Synchronization)

인덱스와 프레임을 이용한 매핑 함수가 미디어 간의 동기화를 자동적으로 달성한다. 이렇게 사전에 동기화가 이루어 때문에 동기화 함수에서 검증을 해야 하는 부분은 네트워크의 과부하에 의한 주 미디어의 지연과 예측할 수 없는 에러로 인한 상호대화형 개체의 잘못된 설정이다. 주 미디어가 과도하게 지연될 경우에는 주 미디어로부터 다른 상대 미디어의 동기화와 상호대화성이 모두 이루어질 수 없기 때문에 심각한 문제가 발생될 수 있다. 이러한 경우 주 미디어의 지연 한계 값(delay value)을 책정하여 이 값을 넘을 경우에는 주 미디어의 전송 품질을 저하 시키거나 주 미디어 이외의 다른 상대 미디어의 전송 자체를 중단하는 방법을 사용할 수 있다. 또한 예측할 수 없는 에러에 의해 상호대화형 객체의 값이 잘못된 값을 가질 경우 내부 통신 함수를 이용해서 이들의 값을 변경시키거나 새로 설정할 수 있다. 이러한 기능들을 통해서 동기화는 자연스럽게 달성할 수 있다.

3.3. 패트리 넷에서 트랜지션간의 처리

기존의 패트리 넷에서는 수행이 되어져야 하는 작업들이 모두 완료되었을 때 하나의 작업을 나타내는 트랜지션(*transition*)이 다음 트랜지션으로 이동할 수 있다. 그러나 본 논문에서의 트랜지션 전환은 미디어 서비스의 종류에 따라 달라질 수 있다. 즉 주 미디어

가 무엇인가에 따라 기본 방식으로 주 미디어의 재생 완료 시점에서 상대 미디어의 상호대화형 객체를 확인하여 이 객체에 반드시 모든 재생을 끝마치도록 하는 미디어 완료값(m_end)이 없으면 다음 트랜지션으로 전환 하는 방식을 사용할 수도 있고 경우에 따라서는 상대 미디어를 무시하고 주 미디어의 재생이 완료되면 바로 다음 트랜지션을 수행하도록 할 수도 있다.



(그림 7) 트랜지션 처리

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. 본 논문에서 제안한 모델은 패트리 넷을 이용한 다른 모델과 비교하여 Allen 의 시간 관계를 명세할 수 있다는 점은 동일하다. 그러나 사용자와의 상호대화성 측면에서는 제안된 함수의 기능들이 적절히 동작함으로서 훨씬 우수한 성능을 보일 수 있다. 결국 이러한 상호대화성을 지원하기 위한 객체의 삽입으로 인한 로드가 전체 시스템에 끼치는 영향력을 최소로 할 수 있는 삽입 빈도수를 실험을 통해 결정하는 것이 필요하다. 차후에는 이러한 객체 삽입 빈도수의 결정을 위해 상호대화형 객체를 삽입하고 난 후에 부가 처리 시간이 어느 정도 요구되는지 실험을 통해 측정해야 한다. 또한 원활한 상호대화성 및 동기화가 충분히 달성되는지에 대해 정량적으로 증명할 수 있는 시뮬레이션이 요구된다. 이러한 실험을 위해 버퍼 레벨에서 스트림을 제어할 수 있는 미디어 재생기 프로그래밍과 처리 시 전체 시간을 측정할 수 있는 프로그램을 구축할 것이다.

참고문헌

[1] Allen, J.F, "Maintaining Knowledge About Temporal Intrvals", CACM, 11, vol. 26, pp.832-843, 1983.
[2] Naveed U. Qazi, Arif Ghafoor, "A Synchronization and Communication Model for Distributed Multimedia Objects, "Proc. Of the First ACM Conference on Multimedia, pp.147-155, Aug. 1993.
[3] Little, Thomas D. C., and Arif Ghafoor, "Synchronization and Storage Model for Multimedia Objects", IEEE Journal on Selected Areas in Communication, vol.8, No.3, pp.413-427, April 1990.
[4] Lynda Hardman, Jacco van Ossenbruggen, Dick C. A. Bulterman, "Composition and Linking in Time-based Hypermdia", European Union ESPRIT Chameleon project, 12. 1999.
[5] 성경상, 황민구, 이기성, 이근왕, 오해석, "버퍼레벨을 이용한 적응형 멀티미디어 동기화 재생 기법", 한국정보처리학회 춘계학술발표논문집, 제 8 권, 제 1 호, pp.619-622, 2001.
[6] 이기성, 이근왕, 이종찬, 오해석, "대기시간을 이용한 적응형 멀티미디어 동기화 기법", 한국정보처리학회 논문지, 제 7 권 제 2 호, pp.649-655, 2000.2.
[7] 박영숙, 이승원, 정기동, "분산 멀티미디어 응용을 위한 실시간 동기화 메커니즘", 한국정보처리학회 논문지, 제 7 권, 제 12 호, pp.3785-3793, 2000.12.