웹 멀티미디어 시나리오 상영기 설계

임재걸°
동 국 대 학 교 컴 퓨 터 학 과
yim@wonhyo.dongguk.ac.kr

A Design of a Web Multimedia Scenario Player

Jaegeol Yim°
Dept. of Computer Science, Dongguk University

요 약

홈페이지는 하이퍼미디어를 사용하여 작성하는 것이 보통이다. 본 연구의 최종 목적은 이를 확장하여 멀티미디어 시나리오를 작성하여 홈페이지에 삽입할 수 있는 기능을 제공하는 것이다. 이를 위하여, 본 논문은 시나리오 표현 방식을 제안하고, 이렇게 표현된 시나리오를 웹브라우저에서 해석하여 상영하는 프로그램의 설계를 소개한다. 인터넷이 아닌 PC에서 시나리오를 작성하는 환경을 제공하고 작성된 시나리오를 상영하여 주는 멀티미디어 저작 도구는 이미 많이 나와 있지만, 홈페이지에 멀티미디어 시나리오를 삽입하고 이를 해석하여 상영하여 주는 소프트웨어 개발 사례는 별로 없다. 본 연구는 웹페이지에 멀티미디어 시나리오를 손쉽게 삽입할 수 있는 소프트웨어 개발의 계기가 될 것으로 본다.

1. 서론

멀티미디어 저작 도구를 이용하여 시나리오를 작성함으로써 멀티미디어 타이틀을 작성하는 것은 이미 보편적인 일이다. 멀티미디어 저작 도구로 툴 북, 디렉터, 새빛 등이 많이 사용된다. 멀티미디어 타이들은 특히 교육 분야에서 많이 사용된다. 다양한 멀티미디어를 이용하여 교육 내용을 흥미 있게 전달하고, 성취도를 검사하고 feed back하는 과정을 시나리오에 포함함으로써 피교육자가 스스로 모든 것을 완벽하게 학습할 수 있도록 돕는다.

이러한 멀티미디어 시나리오가 홈페이지에 삽입된다면, 당연히 사용자에게 도움을 줄 것이다. 그러나, 홈페이지에 멀티미디어 시나리오를 삽입하기 위하여 사용할 수 있는 소프트웨어 시스템은 발견하기 어렵다. 따라서, 이러한 소프트웨어의 개발이 매우 시급하게 요구되는 실정이다.

홈페이지에 멀티미디어 시나리오 삽입을 가능하게 하려면 시나리오 제작 환경을 제공하고, 제작된 시나리오를 해석하여 상영해 주어야 한다. 본 논문에서는 우선 시나리오를 해석하고 상영하여 주는 소프트웨어 개발 방법을 소개한다.

2. 관련연구

본 논문과 관련된 연구는 멀티미디어 저작 도구 분야의 연구[1]와 인터넷에서의 멀티미디어 시나리오 모델링과 관련된 연구를 들 수 있다[2-4]. [1]은 멀티미디어 시나리오를 페트리 넷으로 표현하고, 이렇게 표현된 시나리오를 해석하여 상영하는 소프트웨어 개발이 소개되어 있다.

또한 [2-4]에는 멀티미디어 동기화를 모델하는 페트리 넷 방법들이 소개되었다. [2]에는 OCPN이라는 페트리넷이 소개되어 있으며 [3]에는 TSPN이라는 페트리 넷이 소개되었고, [4]에는 Fuzzy 이론을 도입한 페트리 넷이 소개되었다. TSPN을 소개하면 다음과 같다.

분산 비동기 시스템에서는 통신, 처리, 출력에 따른 시간적 비결정성이 필연적이다. 그럼에도 불구하고 OCPN은 nominal 경과 시간만을 표기함으로 허용 가능한 시간적 jitter의 표현이 불가능하다.

시간을 transition에 사상하여 놓은 페트리넷을 Time Petri net이라고 하며, 이때 nominal 시간 대신 시간 interval을 사용하면 jitter를 표현하는 것이 가능하다. 그런데 Time Petri net은 fire가 늦게 도착하는 토큰에 의하여 주도됨으로 멀티미디어 시나리오를 모델링할 때, Time Petri net으로는 표현이 불가능한 경우가 존재한다

유향간선에 Time intervals을 매핑한 페트리 넷을 TAPN이라 한다. TAPN의 경우에는 transition의 입력 유향간선들의 시간 interval들의 intersection 중에 fire가 가능하다. 그러나 멀티미디어 시나리오의 경우에는 intersection이 empty가 되는 경우가 발생한다.

이러한 기존의 모델들의 단점을 보완하고자 제안된 것이 TSPN이다. TSPN의 정의는 다음과 같다.

TSPN = (P, T, B, F, M_0, IM, SYN) where:

(P, T, B, F, M_0)은 Petri Net.

2절에 소개된 페트리 넷의 A는 B와 F의 합집합이다. A의 원소 중 P×T 순서쌍의 집합을 B라 하고, T×P 순서쌍의 집합을 F라 한다.

A를 transition의 입력 간선의 집합이라 하자, A = $\{a_i=(p_i, t_i) \mid B(p_i, t_i) > 0\}$. 그러면 interval mapping 함수 IM은 다음과 같이 정의된다.

IM: $A \rightarrow Q^+ \times Q^+ \times (Q^+ \cup +\infty)$, $a_i \rightarrow (x_i^s, n_i^s, y_i^s)$, 이 tuple의 내용은 각각 x_i^s=최소, n_i^s=nominal, y_i^s=최대 정적(static) 활성화 기간이다.

SYN: T → {and, weak-and, or, strong-or, master, or-master, and-master, weak-master, strong-master}.

일반적으로 Petri net에서 transition 격발은 마킹을 변화시킨다. 마킹은 place에 놓인 토큰의 상태임으로, TSPN에서의 격발은 상태 변화로 정의된다. TSPN에서의 상태는 (M, I) 쌍으로 정의된다. M은 마킹, I는 enabled 된 간선의 temporal validity interval의 list로 정의된다. 이 list의 entry 수는 마킹 M에서 enabled 된 간선의 수와 같다.

현재 상태를 S라 하고, transition t가 현재 상태에서 θ 시간이 흐른 다음 격발 가능하다고 가정하자. 그러면 θ에서 t를 격발하여 얻는 다음 상태 S'=(M', I')은 다음과 같이 계산하여 얻는다.

1. M'는 일반적인 Petri net의 경우처럼 다음과 같이 얻는다.

$$M'(p) = M(p) - B(t,p) + F(t,p).$$

2. I'는 다음과 같은 세 단계에 의하여 구한다.

1) t의 격발에 의하여 disabled되는 모든 간선 a_i에 대하여 $IM(a_i)$를 I에서 삭제한다.

2) enable 상태를 유지하는 모든 간선 a_k의 IM 값을 θ로 다음과 같이 갱신한다. $IM(a_k) = (x_k, n_k, y_k)$ 라면 이것을 $(\max(0, x_k-\theta), n_k-\theta, \max(0, y_k-\theta))$로 대치한다.

3) 격발 결과 enabled 된 간선의 IM 값으로 I'를 만든다.

TSPN이 멀티미디어 데이터간의 동기화 현상을 모델하기에 적당하기는 하지만, 본 논문의 목적인 멀티미디어 시나리오를 표현하기에는 부적절하다. 왜냐하면 활성화 시간의 MAX 시간 이전에 전송된 데이터가 전혀 없을 만큼 전송 속도가 느릴 경우에는 아무것도 보여 주지 못하는 경우가 발생할 수 있기 때문이다.

활성화 시간의 MAX를 무한대로 놓으면 이러한 현상을 방지할 수는 있다. 그러나 이렇게 하면 시나리오 진행이 가장 늦게 도착하는 데이터에 의하여 결정됨으로 사용자에게 지루함을 주게된다. 인터넷은 속도가 불규칙하고 사용자가 전 세계에 퍼져 있기 때문에 사용자와 서버간의 거리에 따른 전송 속도의 편차가 크다. 따라서, TSPN은 시나리오 표현에 부적절하다.

3. 페트리넷 모형

TSPN은 미디어간 동기화를 모델하기 위하여 제안된 페트리넷이다. 이에 반하여, 본 논문이 제안하는 Internet Multimedia Petri net (IMPN)의 목적은 인터넷에서 상영될 수 있는 멀티미디어 시나리오를 표현하는 것이다. 따라서, 기존의 Petri net에 상영될 멀티미디어 데이터를 표현하는 기능을 추가시켜야 하고, 시간의 경과에 따른 시나리오 진행을 표현할 수 있어야 한다. TSPN의 경우와 마찬가지로 출력시간은 간선에 사상하고, 동기화 정책을 transition에 표현한다. 단 TSPN은 jitter를 고려한 동기화 정책을 나타내는 것이 목적인데 반하여 본 논문의 목적은 시나리오를 표현하는 것이 목적임으로 동기화 정책을 달리해야 한다. 또한, 출력될 멀티미디어 데이터를 IMPN의 place에 사상하여 나타내고, 통신 속도를 고려하여 간선에 사상된 활성화 기간을 동적으로 변화시킨다. 이러한 기능이 가미된 IMPN의 정의는 다음과 같다.

IMPN = (P, T, B, F, M, SYN, MT, IM, S_POS, W_POS, RECT, URL, F_MT, F_S_POS, F_W_POS, F_RECT, F_URL), 모든 요소는 TSPN의 해당 요소와 같다. TSPN에 나타나지 않는 요소들의 정의는 다음과 같다.

MT = {MT_start, MT_end, MT_au, MT_txt, MT_mpg, MT_jpg}. 멀티미디어 데이터 유형의 집합.

S_POS = {(X_pos, Y_pos)} X_pos와 Y_pos는 음이 아닌 정수로 시나리오 작성시 스크린 상의 위치.

W_POS = {(X_pos, Y_pos)}. 윈도우의 위치.

RECT = {(Width, Height)} Width와 Height는 음이 아닌 정수로 윈도우의 크기.

URL 인터넷 주소와 파일 이름.

F_MT, F_S_POS, F_W_POS, F_RECT, F_URL은 각각 place에 연관된 MT, S_POS, W_POS, RECT, URL를 찾아 주는 함수임.

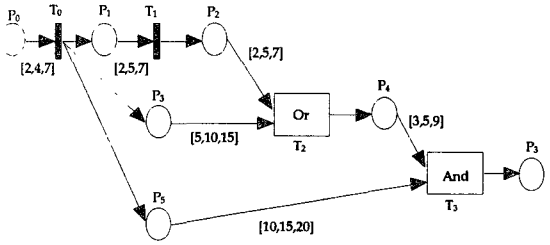
IMPN의 상태는 TSPN의 경우와 마찬가지이고, 격발 방법도 기본적으로 마찬가지다. 다만 네트웍의 속도를 고려하여 출력 간선의 IM에 전송비율을 반영하여 새로운 IM'을 만들고, 이를 바탕으로 다음 트랜지션을 격발한다. 예를 들어, 그림1을 고려하여 보자. t_0가 격발하면 $S_1=(\{p_1, p_3, p_5\}, \{[2,5,7], [5,10,15], [10,15,20]\})$이 된다. S_1에서 격발가능한 transition은 t_1이다. t_1이 시각 7에서 격발하면 p_1의 파일 크기와 실제 전송된 버퍼의 크기를 비교하여 전송비율을 구한다. 그리고, 전송율을 반영하여 I를 구한다. 전송율이 0.5라고 한다면, t_1 격발 결과 얻은 새로운 상태 S_2의 I는 다음과 같다. $IM(p_2, t_2)=[2,5,7]$에 전송율의 역수를 곱하여 [4, 10, 14]를 구한다. I의 다른 원소에 대해서도 전송율을 반영한다. 즉, $IM(p_3, t_2)=[5,10,15]$에서 [10, 20, 30]을, $IM(p_5, p_6)=[10,15,20]$에서 [20,30,40]을 얻는다. 그리고 여기에 격발시간을 반영하면, 다음과 같은 상태 S_2를 얻는다.

$S_2=(\{p_2, p_3, p_5\}, \{[4,10,14], [3,13,27], [13,23,33]\})$.

전송율을 고려하지 않았다면 $IM(p_3,t_2)$의 [5, 10, 15]로부터, 여기에 격발시간을 반영한 [0,3,8]이 I의 원소가

되었을 것이다. 상태 S_2를 기점으로 하여 t_2는 이제 8시간이 아닌 27시간 이내에 격발하여야 한다. 만일 t_2의 격발이 8시간 이전에 수행되면 p_2와 p_3의 데이터가 지정시간 내에 모두 출력 완료된 것임으로 IM(p_5, p_6)를 다시 [10,15,20]으로 고치고, t_1의 격발시간과 t_2의 격발시간을 여기에서 빼 줌으로써 이 시간 간격에 해당하는 I의 원소를 구해야 한다. 즉, t_2의 격발은 27시간까지 기다려 주지만 전송율을 구할 때는 원래의 IM 값을 기준으로 한다.

따라서, t_2가 격발될 시점의 전송율도 [0,3,8]과 [2,5,7]의 max, 즉 8시간에 p_2와 p_3의 파일 크기와 버퍼에 있는 데이터의 크기를 비교하여 각각 전송율을 구한 다음 전송율이 작은 것을 선택하여, 이를 이용하여 IM 값을 갱신한다.



(그림 1) IMPN 예

4. 구현

시나리오 상영기는 시나리오를 해석하여 재생한다. 따라서, 상영기의 입력은 시나리오 명세다. 시나리오 명세는 시나리오 작성기에서 작성하여 준다. 시나리오 명세에는 페트리넷의 incidence 행렬이 포함된다. 이 행렬의 행과 열의 수가 명세의 첫 부분에 쓰인다. 그리고, B-행렬과 F-행렬이 이 뒤를 따른다. 다음에는 각 place와 transition에 연합된 정보들이 수록된다. 다음은 시나리오 명세의 일부를 예로 보인 것이다.

```
6, 15          : 행(transition)과 열(place)의 수
B-Matrix
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0    : B Matrix의 내용
0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0    : B Matrix의 내용
.....
F-Matrix
0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0    : F Matrix의 내용
0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0    : F Matrix의 내용
.....
P1             : place의 Identification
MT_start       : place의 Media Type
2,4,6          : 출력시간(min, nominal, max)
50,100         : place의 Screen Pos
100,100                : 미디어의 출력 위치
200,200                : 출력될 크기
gate.mpg       : 출력될 미디어 데이터

T1             : transition의 Identification
AND            : transition의 동기화 전략
null           : 동기화에서 제외할 place가 없음
100,100                : transition의 Screen Pos

P2             : place의 Identification
MT_mpg         : place의 Media Type
17,19,21               : 출력시간(min, nominal, max)
150,50         : place의 Screen Pos
...
```

시나리오의 진행을 통제하는 알고리즘은 다음과 같다.

```
control_flow(i)
S=(M,I) 생성
while(True) {
  토큰이 놓인 place의 데이터 형태가 END이면 종료
  토큰이 놓인 place마다 thread 생성, 해당 데이터 출력;
  enable된 transition ti의 I(ti)에 연합된 데이터 출력이 모
        두 종료하면 ti를 격발하고 I를 조절하여 새로운
        S를 생성;
  enable된 transition ti의 동기화 정책에 의한 '격발 가능
        시간'이 종료하면 ti를 격발한다. ti의 입력 arc의
        IM을 I에서 삭제하고, O(ti)의 출력 arc의 IM을
        전송율로 조절한 결과를 I에 첨가함. 그리고 I의
        다른 원소에도 전송율과 경과시간을 반영하여
        새로운 S를 생성;
}
```

place 객체에 다음과 같은 변수를 둔다:

```
class Place  {
  String Identification; // place의 식별자
  String Media_Type; // Media의 type
  int  Delay[3] = new int[3];  // Media의 지연시간
  Point Screen_Position; // 모델링시의 위치
  Point Window_Position; // 윈도우 출력시 위치
  Point Window_Rectangle;  // 윈도우의 크기
  String Media_Url;  // 인터넷 주소와 파일 이름
}
```

Place class에서 제공하는 method는 다음과 같다.

```
public String F_MT()
public int[] IM()
public Point F_S_POS()
public Point F_W_POS()
public String F_URL()
```

F_MT()는 place에 해당하는 Media Type을 리턴하고, IM()은 place의 temporal validity interval을 리턴하고, F_S_POS()은 Screen_Position을, F_W_POS()는 Window_Position을, F_URL()은 Media_Url을 리턴하는 method들이다.

Transition class에는 다음과 같은 변수가 필요하다.

```
class Transition  {
   String Identification;    // transition의 식별자
   String Syn;               // transition의 진행정책
   String EX_p;  // 진행정책에서 제외되는 place
   Point Screen_Position;  // 모델에서의 위치
}
```

Transition class에서 제공하는 method는 다음과 같다.

```
public Point F_S_POS()
public String F_Syn()
public String F_Syn_ID()
```

F_S_POS()은 Screen_Position을, F_Syn은 동기

화 정책을, F_Ex_p()는 진행정책에서 제외되는 place들을 리턴하는 method들이다.

기본적으로 자바에서는 동영상을 재생할 수 있는 class를 지원하지 않는다. 이를 해결하기 위해 나온 것이 JMF(Java Media Framework)이다. 이 JMF는 각종 멀티미디어 파일이나 스트림을 다룰 수 API이다. JMF를 이용하면 웹 브라우저에서 어떠한 플러그인 없이도 음악파일이나 동영상을 재생할 수 있다. 비디오 출력을 위한 MediaPlayer class의 골격은 다음과 같다:

```
class MediaPlayer extends Thread implements ControllerListener{
      .....
    public MediaPlayer(Applet a, String mediaFile) {
       a.setLayout(null);
       URL url = null;
       URL codeBase = a.getDocumentBase();
        ....
              player = Manager.createPlayer(url);
       ....
           player.addControllerListener(this);
           player.realize();
       ...
       }
    }
    public void run() {
       if (play_start)
              player.start();
    }
  public void Play_Destroy() {
       ....
  }
    public synchronized void controllerUpdate(ControllerEvent event) {
       ...
       if (event instanceof RealizeCompleteEvent) {
              .....
              // 사용자 인터페이스를 화면에 표시
           player.prefetch();
       } else if (event instanceof EndOfMediaEvent) {
           player.setMediaTime(new Time(0));
           // 미디어의 끝에 다달았을 경우
       } else if (event instanceof PrefetchCompleteEvent) {
           play_start =  true;
           player.synStart();
       } else if (event instanceof ControllerErrorEvent) {
           player = null;
           Fatal(((ControllerErrorEvent)event).getMessage());
        } else if (event instanceof PrefetchCompleteEvent) {
              .....
       }
```

소리 출력을 위한 AudioOut class의 골격은 다음과 같다:

```
class AudioOut extends Thread {
  AudioClip audio;
  public AudioOut(Applet a, String audio_file) {
    audio      =      a.getAudioClip(a.getDocumentBase(),
                    audio_file);
  }
  public void run() {
     audio.play();
  }
}
```

이미지 출력을 위한 class의 골격은 다음과 같다:

```
class ImageDisplay extends Thread {
      public ImageDisplay(Applet a, ImageObserver o,
                          String file, Point pos) {
           g = a.getGraphics();
           img = a.getImage(a.getDocumentBase(),file);
           this.img_obs = o;
      }
      .....
}
```

5. 결론

본 논문은 웹브라우저에서 멀티미디어 시나리오를 해석하고 재생할 수 있는 상영기의 설계를 소개하였다. 이를 바탕으로 홈페이지에 멀티미디어 시나리오를 작성하여 삽입할 수 있는 시스템 개발의 계기가 될 것이다.

6. 참고 문헌

[1] 임재걸, 이계영, 한승협, "페트리넷으로 표현된 멀티미디어 시나리오의 재생기 구현", 정보과학회, 1998년 추계학술발표대회.

[2] T. Little and A. Ghafoor, "Synchronization and Storage Models for Multimedia Objects," IEEE J ournal on Selected Areas in Communications, vo l. 8, No. 3, April 1990, pp. 413-427.

[3] M. Diaz and P. Senac, "Time Stream Petri Nets A Model for Timed Multimedia Information," A pplication and Theory of Petri Nets 1994, Spring er-Verlag, Berlin, pp. 219-238.

[4] Y. Zhou and T. Murata, "Fuzzy-Timing Petri N et Model for Distributed Multimedia Synchroniza tion,"