XML 기반 대본 작성 및 연습 시스템 구현

김 신 우[†] · 신 기 호[††] · 박 성 은[†] · 이 용 규[†††]

요 약

지금까지 연극의 대본을 XML을 이용하여 문서화하기 위한 연구는 있었으나, 시나리오 작가가 대본을 XML 형식으로 작성하도록 도와주는 문법 지향적인 편집 도구나 XML 형식의 대본을 활용하여 배우들이 연습할 수 있도록 지원하는 대본 연습 시스템에 관한 연구는 찾아볼 수 없었다. 따라서, 본 논문에서는 시나리오 작가가 웹에서 대본을 쉽게 작성할 수 있는 XML 기반의 문법 지향적 대본 편집기를 개발한다. 또한, 배우들이 동기화 쇼 기능을 이용하여 함께 연습할 수 있는 웹 기반 대본 연습 시스템을 구현하며, 이를 위해 필요한 클라이언트들간의 이벤트 동기화 모델을 제안하고, 성능 평가를 통하여 어떤 구현 기술이 가장 효율적인지를 보인다. 따라서, 이 시스템을 활용하면 시나리오 작가가 쉽게 XML 형식의 표준화된 대본을 작성하여 교환할 수 있을 뿐만 아니라, 이를 이용하여 배우들이 웹을 통해서 함께 연습할 수 있다는 장점이 있다.

Implementation of a Scenario Editing and Practicing System Based on XML

Shin Woo Kim[†] · Ki Ho Shin[††] · Sung Eun Park[†] · Yong Kyu Lee[†††]

ABSTRACT

In order to represent a play script using XML, a DTD has been defined and used on the web. However, it is not easy to make XML scripts using text editors for a writer who is not familiar with XML. Moreover, editors do not help actors/actresses use XML scripts for practicing plays. In this research, we develop a syntax-directed XML editor designed for writing XML scripts easily. Also, we implement a system that help actors/actresses practice plays together with multi-clients. Play events are synchronized by a time server synchronizing clocks of participants. We have evaluated the performance of various event implementation techniques through experiments and have used an appropriate technique for synchronizing play events. By using the system, a play writer can easily make an XML script and actors/actresses practice the play using the script.

키워드 : XML, XML 스키마(XML Schema), 동기화(Synchronization), 대본(Script)

1. 서 론

XML(eXtensible Markup Language) 형식의 대본을 이용하면 시나리오 작가들이 웹에서 표준화된 문서를 공유할 수 있다는 장점이 있다. 따라서, 희곡을 XML 형식으로 표현하기 위하여 셰익스피어 DTD(Document Type Definition)[1]가 정의되었으며, 햄릿, 오델로 등의 대본이 작성되어 활용되고 있다. 그러나, 작가가 XML 형식의 대본을 직접 작성하려면 XML 편집기나 일반 편집기를 사용해야 하는데, XML 편집기는 사용하기가 복잡하고, 일반 편집기는 XML 태그와 XML 문법에 대해서 자세히 알아야하는 단점이 있다.

따라서, 본 논문에서는 XML에 익숙하지 않은 시나리오 작가도 웹에서 XML 스키마[2] 구조에 맞는 XML 형식의 대본을 쉽게 작성하고 확인할 수 있는 문법 지향적인 대본 편집기(Syntax-Directed Script Editor)를 개발한다.

또한, 뉴스 앵커들이 보도자료를 프롬프터(Prompter)를 이용하여 연습하거나 방송하는 것처럼, 대본 작성 시스템에 의해 만들어진 XML 형식의 대본을 이용하여 배우들이 초기단계에서 함께 연습하거나 외울 수 있는 대본 연습 시스템이 필요하지만, 지금까지는 이러한 시스템을 찾아볼 수 없었다. 그러므로, 본 논문에서는 배우들에게도 연습하는데 도움을 주기 위해서 대본 연습 시스템을 개발한다. 이 시스템은 웹을 통해 연습이 가능하며, 연습 도중에 감독 또는 배우가 연습을 중지하거나 재시작 할 수 있고, 스크롤 바를 이용하여 원하는 위치로부터 대본 연습을 다시 시작할 수 있는 기능을 제공한다.

이를 위해서는 먼저 서버와 클라이언트들간의 시각 동기화(Time Synchronization)가 필요한데, 우리는 분산 시스템에서의 시각 동기화를 위해 제안된 버클리 알고리즘[3, 4]을 이용한다. 또한, 서버와 클라이언트들간의 이벤트 동기화 구현을 위해 선택할 수 있는 다양한 구현 방법들에 대하여 성능을 평가하고, 이들 중에서 성능실험 결과를 통해

※ 이 연구는 동국대학교 연구년 지원에 의하여 이루어졌음.
† 준 회 원 : 동국대학교 대학원 컴퓨터공학과
†† 준 회 원 : 어필텔레콤 S/W 연구원
††† 종신회원 : 동국대학교 컴퓨터멀티미디어공학과 교수
논문접수 : 2002년 4월 18일, 심사완료 : 2003년 2월 12일

수행 속도가 좋은 서블릿(Servlet)을 이용하여 대본 연습 시스템을 구현한다.

2. 관련 연구

본 절에서는 시스템 구현과 관련된 XML 기술 및 시각 동기화 알고리즘에 대해서 설명한다.

2.1 XML 관련 기술

본 논문에서 구현한 시스템은 다음과 같은 최신 표준 XML 기술로 개발되므로 기존 시스템이 갖는 표준화와 확장성의 문제를 해결할 수 있다. 다음은 XML과 관련된 기술이다.

2.1.1 XML 스키마

XML 문서의 형식과 데이터형을 정의하기 위해 제안된 DTD(Document Type Definition)가 어려운 문법을 사용하고, 수치 정보를 정확히 나타낼 수 없으므로 이러한 문제를 해결하고자 XML 스키마(XML Schema)가 제안되었다[2]. XML 스키마는 DTD와 같은 역할을 하지만, XML 형식으로 표현하기 때문에 익히기 쉽고, 확장성을 갖는 장점이 있다. 또한 데이터형도 다양해서 날짜, 숫자, 시간 등의 데이터 형태로 나타낼 수 있다.

2.1.2 DOM

XML DOM(Document Object Model)은 XML 문서를 효율적으로 관리하기 위해 트리 구조 형태로 파싱한 객체에 대한 인터페이스 표준이다[5]. DOM은 XML 문서를 객체로 사용 가능하게 만들어 XML 문서로부터 자료를 읽거나 탐색, 수정, 추가, 삭제 등이 가능하다. 이를 가능하게 하기 위해서는 일반적으로 DOM API를 사용하는데, W3C에서 이 인터페이스 표준을 정한다.

2.1.3 XSL

XSL(eXtensible Stylesheet Language)은 XML 데이터나 문서가 웹 브라우저에서 문서의 외양이 사용자에게 어떻게 보일 것인지를 기술하는 언어이다[6]. XSL은 문서의 구조를 바꿔주는 XSLT와 문서의 출력형식을 정의하는 XSLFO로 구성된다.

2.1.4 SOAP

SOAP(Simple Object Access Protocol)은 XML과 HTTP를 기반으로 네트워크 상에 존재하는 각종 컴포넌트간의 호출을 효율적으로 할 수 있게 하는 통신 프로토콜로서 다음과 같은 특징이 있다[7, 8]. 첫째, XML 기반이므로 이 기종의 시스템간에 객체를 공유하여 사용할 수 있다. 둘째, 인터넷 표준인 HTTP를 사용하므로 널리 사용될 수 있다. 따라서 이는 독자적인 기술로 개발되어 상호 운용성에 문제가 있는 기존 분산객체 시스템의 문제를 해결할 수 있어 현재 많은 관심을 받고 있다.

2.2 시각 동기화 알고리즘

대본 연습 시스템을 구현하기 위해서는 서버가 단방향으로 클라이언트에게 메시지를 전송하는 푸시[9] 방식이 아닌, 분산 시스템에서의 시각 동기화 기법[4, 10-14]을 이용한 서버와 클라이언트들간의 이벤트 동기화가 요구된다. 소프트웨어적으로 시각을 동기화하는 대표적 알고리즘으로는 크리스챤 알고리즘[4, 15]과 버클리 알고리즘[3, 4]이 있다.

2.2.1 크리스챤 알고리즘

크리스챤 알고리즘에서는 클라이언트가 시각 서버(Time Server)에게 시각을 문의하는 메시지를 보내면 시각 서버는 단순히 자신의 시각을 문의한 클라이언트에게 알려준다. 그러므로, 시각 서버는 단지 질문에 답하기만 하는 수동적인 성향을 보여준다.

2.2.2 버클리 알고리즘

버클리 알고리즘에서는 시각 서버가 동기화를 필요로 한 모든 클라이언트들에게 그들의 시각을 문의하고, 각 클라이언트는 각자의 시각을 시각 서버에게 통보한다. 모든 클라이언트들로부터 각자의 시각을 받은 시각 서버는 자신의 시각과 비교하여 메시지 전달 시간을 고려한 시각의 차이를 계산하여 클라이언트에게 알려주며, 각 클라이언트는 서버로부터 받은 시각 차이를 이용하여 자신의 시각을 조정한다. 그러므로, 시각 서버가 직접 클라이언트의 시각을 문의하고 조정해 주는 능동적인 성향을 보여준다.

3. XML 기반 대본 작성 시스템 설계

본 절에서는 XML 기반의 대본 작성 시스템을 설계한다.

3.1 대본용 XML 스키마와 인스턴스

기존의 셰익스피어 DTD[1]는 데이터형을 표현하는데 제약점이 많고 엄격한 문법을 요구하므로, 다양한 데이터형을 제공하고 확장성이 좋은 XML 스키마[2]를 정의하는 것이 바람직하다. 따라서, 본 논문에서는 기존의 셰익스피어 대본 DTD 전체를 XML 스키마로 변환하였으며, 여기에서 사용되는 엘리먼트(Element)들의 일부는 <표 1>과 같다.

<표 1> XML 셰익스피어 스키마의 일부 엘리먼트들

엘리먼트	설 명	엘리먼트	설 명
TITLE	대본의 제목	SCNDESCR	시나리오 설명
PREFACE	대본의 서문	PLAYSUBT	대본의 부제목
ACTORS	등장 배우들	ACT	막, 장
SPEECH	대사	SPEAKER	화자
LINE	대사의 라인	SUBLINE	지문

(그림 1)은 XML 셰익스피어 스키마의 전체적인 구조이며, 최상위 엘리먼트는 <PLAY>이고, 서브 엘리먼트로는 <표 1>에서 설명된 엘리먼트들이 사용된다.

```
<?xml version = "1.0"?>
<xs : schema xmlns : xs = "http://www.w3.org/2001/XMLSchema">
<xs : element name = "PLAY">
    <xs : complexType>
        <xs : sequence>
            <xs : element ref = "TITLE"/>
                <xs : element ref = "PREFACE"/>
                <xs : element ref = "ACTORS"/>
                <xs : element maxOccurs = "unbounded"
                              ref = "ACT"/>
                ...
            </xs : sequence>
        </xs : complexType>
    </xs : element>
    ...
```

(그림 1) XML 셰익스피어 스키마

엘리먼트 중에서 막·장을 나타내는 <ACT> 엘리먼트는 대본에서 가장 기본이 되는 요소로 (그림 1)에서와 같이 min Occurs와 maxOccurs 속성을 통해 반복 횟수를 지정할 수 있다. 또한, 스키마에 정의된 <ACT> 엘리먼트의 구조는 (그림 2)와 같으며, <SUBTITLE>, <STAGEDIR>, <SCENE>, <SPEECH> 등의 서브엘리먼트들로 구성된다.

```
<xs : element name = "ACT">
    <xs : complexType>
        <xs : sequence>
            <xs : element ref = "SUBTITLE"/>
            <xs : element minOccurs = "0" ref = "STAGEDIR"/>
            <xs : element maxOccurs = "unbounded"
                          ref = "SCENE"/>
            <xs : element maxOccurs = "unbounded"
                          ref = "SPEECH"/>
        </xs : sequence>
    </xs : complexType>
</xs : element>
```

(그림 2) <ACT> 엘리먼트에 대한 XML 스키마

(그림 1)과 (그림 2)에서 정의한 XML 셰익스피어 스키마에 유효한 XML 기반의 시나리오는 (그림 3)에서 볼 수 있다.

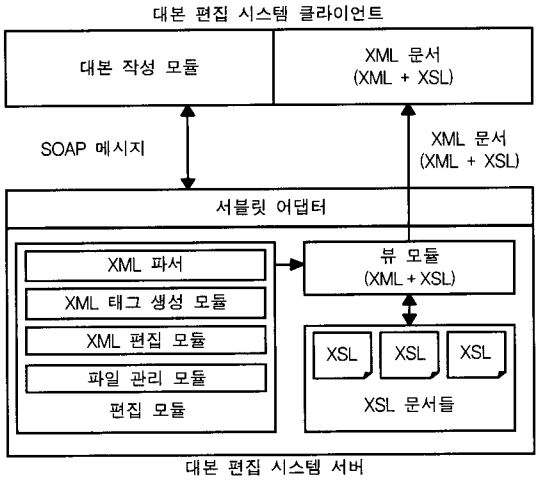
```
<?xml version = "1.0" encoding = "euc-kr" ?>
  <PLAY>
    <ACT>
    <TITLE> ACT I </TITLE>
    <SPEECH>
      <SPEAKER> BERNARDO </SPEAKER>
      <LINE> Who's there? </LINE>
    </SPEECH>
    <SPEECH>
      <SPEAKER> FRANCISCO </SPEAKER>
      <LINE> Nay, answer me : stand, and unfold yourself.
      </LINE>
    </SPEECH>
    <SPEECH>
      <SPEAKER> BERNARDO </SPEAKER>
      <LINE> Long live the king! </LINE>
  </SPEECH>
    ...
  </ACT>
    ...
</PLAY>
```

(그림 3) XML 셰익스피어 인스턴스(햄릿의 일부)

<ACT> 엘리먼트는 XML 스키마에서 정의한 것처럼 횟수에 관계없이 반복하여 사용되고, 다른 엘리먼트들도 XML 정의에 따라 시나리오에서 사용된다.

3.2 대본 작성 시스템 구조

(그림 4)는 대본 작성 시스템에 대한 구성도이다. 클라이언트 모듈에는 왼쪽의 대본 작성 모듈과 오른쪽의 뷰 모듈이 존재한다. 대본 작성 모듈은 작가가 XML 스키마 구조에 맞는 대본을 작성할 수 있는 부분이며, 뷰 모듈은 XML 형식의 대본에 맞게 설계된 XSL을 적용하여 작가에게 대본을 보여주는 역할을 한다.



(그림 4) 대본 작성 시스템 구성도

서버 모듈에서는 XML 형식의 대본을 처리하는 편집 모듈과 뷰 모듈 그리고 기능별로 구분된 XSL(Extensible Stylesheet Language) 문서들이 존재한다. 편집 모듈에는 XML 파서, XML 태그 생성 모듈, XML 편집 모듈 그리고 파일 관리 모듈로 구성되어 있다. XML 파서는 XML 문서를 분석해서 내용을 추출하는 부분이고, XML 태그 생성 모듈은 작가가 작성하는 대본에서 대사 및 지문 등에 해당하는 태그를 생성하여 붙여주는 부분이다. XML 편집 모듈은 대본을 새로 작성하고 내용을 수정하거나 삭제할 수 있는 부분이며, 파일 관리 모듈은 각 시나리오 파일을 관리하는 부분이다.

따라서, 클라이언트 모듈에서 작가가 시스템 화면을 통해서 대본을 작성하기 위해 배우를 선택하고 대사 등을 적어 서버로 전송하면, 서버는 전송된 메시지를 분석하고 각각에 알맞은 태그를 생성하여 붙이며 XML 형식의 대본을 생성하거나 수정하여 해당 시나리오 파일에 저장한다. XML 문서는 XSL을 이용하여 뷰 모듈을 통해 화면에 대본을 보여준다.

4. XML 기반 다자간 대본 연습 시스템 설계

본 절에서는 XML 기반의 다자간 대본 연습 시스템을 설계한다.

4.1 클라이언트간 동기화

대본 연습을 시작하기 위해서는 모든 클라이언트들의 시각을 동기화 하는 것이 필요한데, 본 논문에서는 이를 위해서 능동적인 버클리 알고리즘[3, 4]을 이용한다. 대본 연습을 하기 위해서는 연습을 시작하고, 중지하고, 원하는 위치부터 재시작하는 기능들이 요구되는데, 이때 발생할 수 있는 이벤트의 종류는 <표 2>와 같다.

〈표 2〉 이벤트의 종류

이벤트 명	설 명
시각 동기화	서버와 모든 클라이언트들의 시각을 맞춤
배역 선택	연습 시작 전에 배역을 선택
연습 시작	연습 시작을 요청
연습 중지	연습 중지를 요청
연습 재시작	중지된 지점부터 연습 시작
스크롤 바 재시작	스크롤 바를 움직이기 시작하는 순간에 연습을 중지하고, 스크롤 바의 이동이 끝난 위치에서 연습을 재시작하는 것임

(그림 5)는 <표 2>에서 설명한 이벤트 중에서 가장 동기화 기능이 복잡한 스크롤바 재시작에 대한 처리 절차이다. 사용자가 자신이 원하는 위치로 이동하기 위해서 스크롤 바를 마우스로 누르면, 클라이언트가 서버로 중지 메시지를 보내고, 서버는 다시 다른 클라이언트들에게 중지 메시지를 전달한다. 그리고 사용자가 스크롤 바의 위치 이동을 끝내는 순간 이동된 위치가 담긴 스크롤바 메시지가 서버를 통해 모든 클라이언트들에게 전달된다.

모든 클라이언트들은 자신의 스레드를 유지하고 있다가 서버로부터 중지 메시지를 전달받으면, 메시지 분석모듈 함수를 통해 연습을 중지시킨다. 이어서 스레드의 이동된 위치 값을 가진 스크롤바 메시지가 도착하면 해당 위치값을 설정하고, 그 위치부터 정해진 시각에 연습을 다시 시작하도록 한다.

/* 사용자가 스크롤바를 눌렀을 때 */

/* 클라이언트 */
1 : 클라이언트는 SOAP을 이용하여 중지 메시지를 만든다. ;
2 : 클라이언트는 서버에게 중지 메시지를 보낸다. ;

/* 서버 */
3 : 웹 서버는 중지 메시지를 받는다. ;
4 : 웹 서버는 중지 메시지를 분석한다. ;
5 : 웹 서버는 중지 메시지를 소켓서버에게 보낸다. ;
6 : 소켓서버는 모든 클라이언트들에게 중지 신호를 보낸다. ;

/* 클라이언트 */
7 : 클라이언트는 중지 신호를 받는다. ;
8 : 클라이언트는 연습을 중지한다. ;

/* 사용자가 스크롤바의 이동을 끝냈을 때 */

/* 클라이언트 */
9 : 클라이언트는 서버에게 지정된 위치를 포함하고 있는 재시작 메시지를 보낸다. ;

/* 서버 */
10 : 웹 서버는 재시작 메시지를 받는다. ;
11 : 웹 서버는 재시작 메시지를 분석한다. ;
12 : 웹 서버는 재시작 시간을 계산한다. ;
13 : 웹 서버는 소켓 서버에게 재시작 시간과 이동된 위치를 포함하고 있는 재시작 메시지를 보낸다. ;
14 : 소켓 서버는 모든 클라이언트들에게 받은 메시지를 보낸다. ;

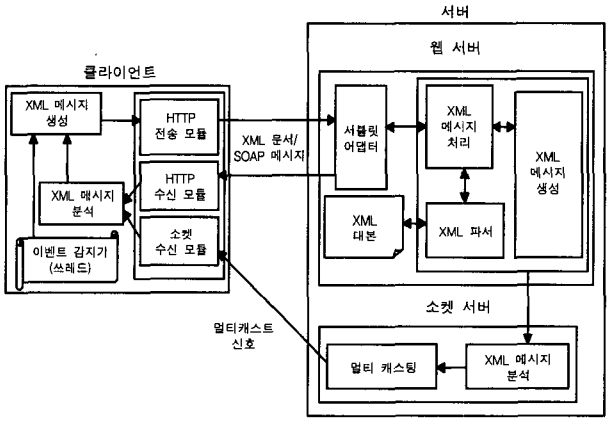
/* 클라이언트 */
15 : 클라이언트는 재시작 메시지를 받는다. ;
16 : 클라이언트는 정해진 시각에 이동된 위치에서 연습을 재시작한다. ;

(그림 5) 스크롤바를 이용한 연습 재시작 절차

이러한 시스템을 구현하기 위해서는 XML 기술과 함께 서버는 서블릿, JSP(Java Server Page), ASP(Active Server Page), Perl 등을 사용할 수 있고[16, 17], 클라이언트는 애플릿을 사용할 수 있다.

4.2 다자간 대본 연습 시스템 구조

다자간 대본 연습 시스템의 구성도는 (그림 6)과 같다. 먼저, 클라이언트 부분은 사용자가 발생시킨 이벤트에 대한 XML 메시지를 SOAP(Simple Object Access Protocol) 형식으로 서버에게 보내는 역할을 한다. 그림의 왼쪽을 보면 클라이언트 부분은 XML 메시지 생성 모듈, XML 메시지 분석 모듈, 메시지 전송 및 수신 모듈 그리고 이벤트 감지기로 구성되어 있다. XML 메시지 생성 모듈은 연습을 시작하거나 중단하는 등의 이벤트에 대해서 메시지를 생성하는 부분이고, HTTP 모듈과 소켓 모듈은 생성된 메시지를 서버와 송·수신하는 역할을 하는 부분이며, XML 메시지 분석 모듈은 서버로부터 받은 메시지를 분석하여 그에 맞는 일을 처리하는 부분이다. 또한, 이벤트 감지기는 대본 연습에 발생할 수 있는 모든 이벤트들을 감지할 수 있다.



(그림 6) 다자간 대본 연습 시스템 구성도

서버 부분은 클라이언트로부터 받은 메시지를 분석하여 해당 작업을 수행하고 다른 여러 클라이언트에게 응답 메시지를 전송하는 역할을 한다. 그림의 오른쪽을 보면 서버

부분은 크게 웹 서버와 소켓 서버로 나눌 수 있다. 웹 서버는 서블릿 어댑터, XML 파서, XML 메시지 처리 모듈 그리고 XML 메시지 생성 모듈로 구성되어 있다. 서블릿 어댑터는 클라이언트와 메시지를 주고받는 부분이며, XML 파서는 대본으로부터 내용을 분석하는 부분이다. XML 메시지 처리 모듈은 클라이언트로부터 받은 메시지를 분석하여 연습을 시작하거나 중단하는 등에 맞는 일을 XML 파서와 서로 연계하여 처리하는 부분이고, XML 메시지 생성 모듈은 XML 메시지 처리 모듈로부터 메시지 처리 후 결과에 대해서 클라이언트에게 전달할 응답 메시지를 생성하는 부분이다. 소켓 서버는 XML 메시지 분석 모듈과 멀티 캐스팅 모듈로 구성되어 있다. XML 메시지 분석 모듈은 웹 서버로부터 받은 메시지를 분석하는 부분이고, 멀티캐스팅 모듈은 분석된 메시지를 여러 클라이언트들에게로 전송하는 부분이다. 이 시스템에서 SOAP[7]은 클라이언트에서 서버로 연습의 시작이나 중단 등과 같은 이벤트 메시지를 전송할 때 사용하고, 소켓은 서버에서 모든 클라이언트들에게 연습의 시작이나 중단 등을 알리는 멀티캐스트(Multicast) 신호를 보낼 때 사용한다.

5. 성능 실험

본 절에서는 서버와 클라이언트들간의 이벤트 동기화 모델을 구현할 수 있는 구현 기술에 대해서 성능 실험을 한다. 즉, 대본 연습 시스템에서 처음 연습을 시작하기 위해 서버와 모든 클라이언트들의 시각을 동기화하는 시간과 모든 클라이언트들이 연습을 준비하기 위해서 시스템을 초기화하는 시간을 측정한다. 그리고, 한 클라이언트가 서버에게 연습을 요청하여 연습이 시작하게 될 때까지 걸리는 시간을 측정한다.

5.1 실험 환경

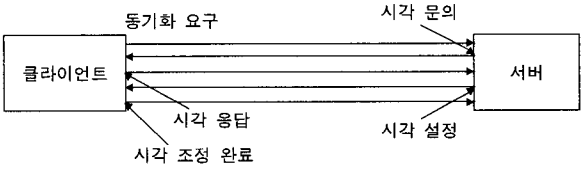
성능 실험을 위해 서버는 펜티엄Ⅳ 1.5GHz, 클라이언트는 펜티엄Ⅱ 233MHz를 사용하였다. 또한 모두 윈도우즈 2000을 운영체제로 사용하였고, 100Mbps 랜카드를 사용하였다. 그리고, 서버는 서블릿, JSP, ASP, Perl 등을 사용하여 구현하였고, 클라이언트는 애플릿을 사용하여 구현하였다.

5.2 성능 실험 모델

본 절에서는 3가지 실험 모델을 설정하여 성능 실험을 한다.

5.2.1 연습 시스템 동기화 모델

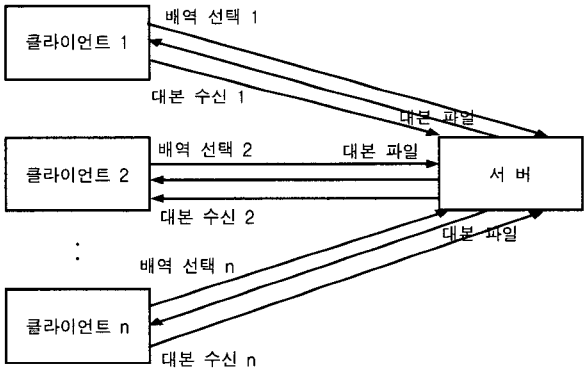
(그림 7)은 대본 연습 시스템을 시작하기에 앞서 서버와 모든 클라이언트들의 시각을 맞추기 위한 시각 동기화 모델을 나타낸다. 처음에 클라이언트에서 시각 동기화를 서버에 요청하고 그 다음은 버클리 알고리즘을 이용한다. 즉, 서버는 모든 클라이언트들에게 그들의 시각을 문의하고, 각 클라이언트는 서버의 요구에 대해 각자의 시각을 서버에게 통보하며 서버는 각 클라이언트로부터 받은 시각을 자신의 시각과 비교하여 메시지 전달 시간을 고려한 시각의 차이를 클라이언트에게 알려준다. 각 클라이언트는 서버로부터 받은 시각 차이를 이용하여 자신의 시각을 조정한다. 본 논문에서는 이 알고리즘에서 클라이언트의 시각 응답 및 서버의 조정을 3회 반복하여 평균을 계산한 후, 그 결과를 클라이언트의 시각 동기화에 이용한다. 마지막으로, 조정을 끝낸 클라이언트는 서버에게 자신의 시각 조정이 끝났음을 알린다.



(그림 7) 연습 시스템 동기화 모델

5.2.2 연습 시스템 초기화 모델

(그림 8)은 연습을 시작하기 위한 초기화 모델을 보여준다. 대본 연습을 시작하기 위해 각각의 클라이언트는 배역을 선택하고 대본 받을 준비가 다 되었다는 메시지를 서버에게 보낸다. 이때 배역 선택을 마지막으로 한 클라이언트의 시각부터 모든 클라이언트들이 대본을 서버로부터 받고 다시 서버에게 대본을 받았다는 메시지를 보내어 서버가 마지막으로 대본 수신 메시지를 받을 때까지 걸린 시간을 측정한다. 예를 들어, 그림의 클라이언트 1부터 n까지 차례대로 배역 선택 메시지를 서버에게 보냈다면, 마지막으로 클라이언트 n이 배역 선택 메시지를 보낸 시각부터 서버가 모든 클라이언트들에게 대본 파일을 보내서 전 클라이언트들이 다 받아 대본 수신 메시지를 서버에게 보내고, 서버에서 마지막 대본 수신 메시지를 받을 때까지의 시간을 측정하는 것이다.

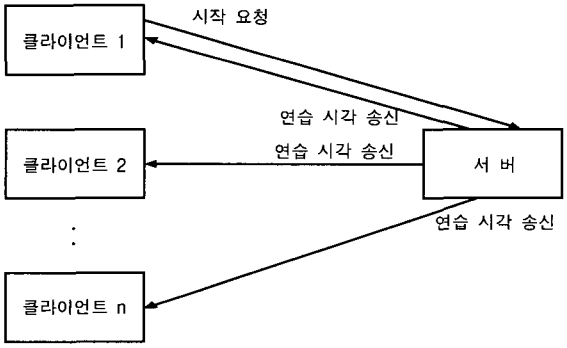


(그림 8) 연습 시스템 초기화 모델

5.2.3 연습 시스템 시작 모델

대본 연습을 시작할 준비가 다 되어 있으면, 클라이언트는 서버에게 연습 시작을 요청한다. 서버가 클라이언트로부터 연습 시작을 요청 받으면, 서버는 연습을 시작할 시각을

계산하여 모든 클라이언트들에게 알려준다. 이때 대본에서의 위치값이 없으면 대본의 처음부터 시작하고, 위치값이 있으면 그 위치가 가리키고 있는 대본의 위치부터 연습을 시작한다. (그림 9)는 처음 연습 시작을 요청한 시각부터 실제 연습이 시작되는 시각까지의 시간을 측정하는 모델이다. 예를 들면, 클라이언트 1이 연습 시작을 서버에게 요청했다면 이때의 시각부터 서버는 연습 시작할 시각을 계산하여 모든 클라이언트들에게 시작할 연습 시각을 알려주고 실제 연습이 시작하는 시각까지의 시간을 측정한다. 연습 중지와 연습 재시작 이벤트의 모델도 시작 모델과 유사하다.



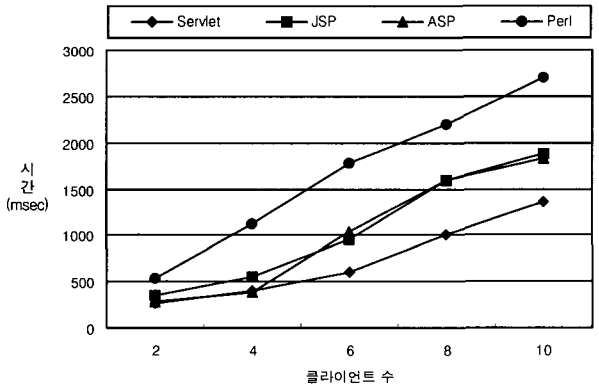
(그림 9) 연습 시스템 시작 모델

5.3 실험 결과 분석

본 절에서는 앞 절에서 설명한 각 모델을 기반으로 서블릿, JSP, ASP, Perl에 대한 실험을 통해 성능을 분석한다. 각 실험에서 측정하는 시간은 자체적으로 지원하는 시간 함수를 이용하여 측정하였다.

5.3.1 동기화 시간

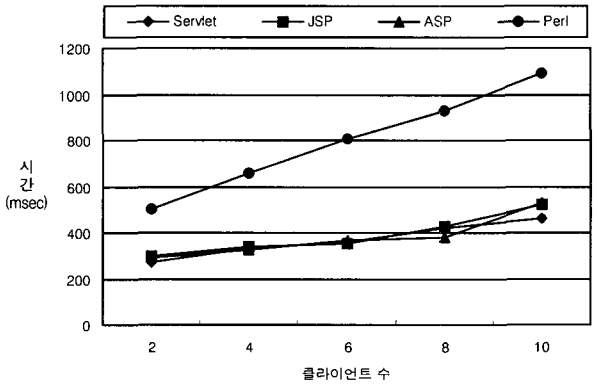
연습 시스템 동기화 모델을 기반으로 서버와 클라이언트들간에 시각을 동기화하는데 걸리는 시간을 측정하였다. 그 결과는 (그림 10)과 같으며, 서블릿이 제일 좋은 성능을 보이고 ASP와 JSP, Perl 순으로 동기화하는데 점점 시간이 많이 소요됨을 볼 수 있다.



(그림 10) 동기화 시간

5.3.2 초기화 시간

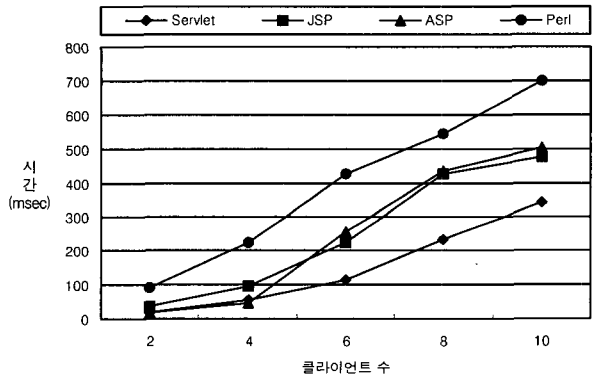
(그림 11)은 연습 시스템 초기화 모델을 기반으로, 배역 선택을 마지막으로 한 클라이언트의 시각부터 모든 클라이언트들이 대본을·서버로부터 받고 다시 서버에게 대본 수신 메시지를 보내어 서버가 마지막 대본 수신 메시지를 받는데 까지 걸린 시간을 측정한 결과를 보여준다. Perl을 제외한 나머지들은 소요되는 시간이 거의 비슷함을 볼 수 있다.



(그림 11) 초기화 시간

5.3.3 시작 이벤트 처리 시간

(그림 12)는 연습 시스템 시작 모델을 기반으로, 한 클라이언트가 서버에게 연습을 요청하는 시각부터 시작 응답을 서버로부터 받아 실제 연습을 시작하는 시각까지의 시간을 측정한 결과를 보여준다. 서블릿이 제일 좋은 성능을 보이고 ASP와 JSP, Perl 순으로 점점 시간이 많이 소요됨을 볼 수 있다. 연습 중지와 연습 재시작의 성능도 실험 결과 연습 시작 이벤트와 동일한 결과를 얻었다.



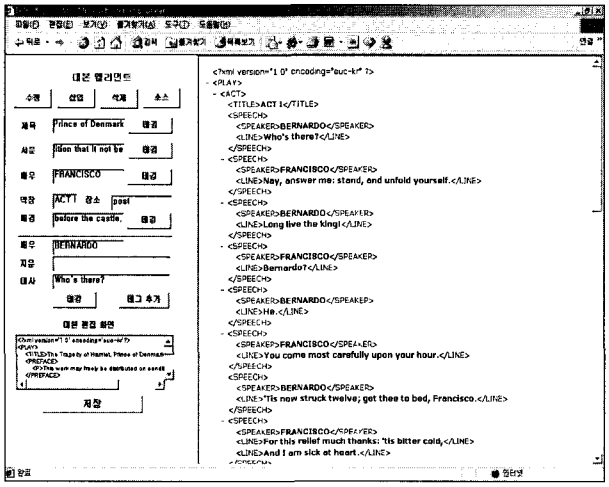
(그림 12) 시작 이벤트 처리 시간

6. 대본 작성 및 연습 시스템 구현

XML 기반 대본 작성 및 연습 시스템 구현에 대해 설명한다.

6.1 XML 기반 대본 작성 시스템 구현

먼저, 대본 작성 시스템의 서버 부분을 구현하기 위해서 성능 실험 결과가 좋은 서블릿[18], XML 스키마[2], XSL[6], SOAP[7], MSXML DOM(Document Object Model)[19]을 이용하고, 클라이언트를 구현하기 위해서는 애플릿과 자바스크립트를 이용한다. (그림 13)에서 작가가 대본을 작성하기 위해서는 제목, 배우, 대사 등의 입력창에 원하는 데이터를 입력하여 버튼을 누르면 된다. 이때, 각 데이터에 맞는 엘리먼트들이 생성되며, 애플릿을 이용하여 XML 스키마 구조에 맞는 XML 기반의 대본을 작성하게 된다. 그리고, MSXML의 DOM 객체와 자바스크립트를 사용하여 오른쪽 프레임에 작성중인 대본을 화면에 보여준다.

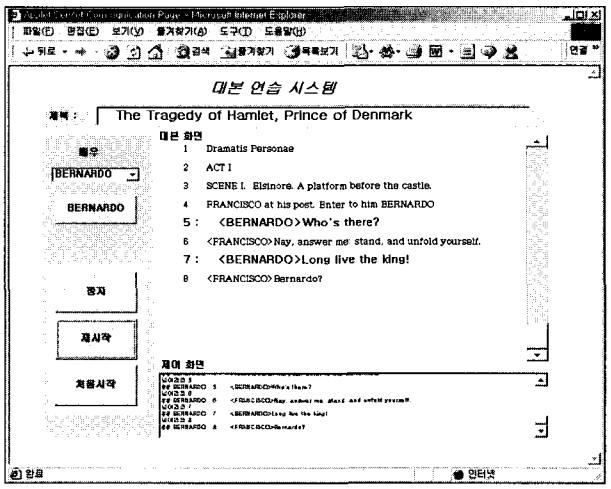


(그림 13) 대본 작성 화면

또한, 대본 쇼 중간에 대본을 수정하고 싶을 경우에 사용자는 편집 모드로 전환할 수 있다. 자바스크립트를 이용하여 XML 문서에 XSL을 적용하면 사용자는 애플릿으로 구현된 편집 기능을 이용하여 대본을 수정할 수 있다. 수정할 내용의 데이터가 서버의 서블릿으로 전송되면, 내용을 수정하고 응답 메시지를 애플릿에게 다시 보낸다. 애플릿은 자바스크립트 함수를 호출하여 서버에 있는 수정된 대본을 다시 로드한다. 대본을 삭제하고자 할 때도 수정과 유사한 방법으로 한다.

6.2 XML 기반 다자간 대본 연습 시스템 구현

다자간 대본 연습 시스템의 서버 부분을 구현하기 위해서 서블릿, XML 스키마, XSL, SOAP, MSXML DOM, 소켓을 사용하고, 클라이언트 부분을 구현하기 위해서는 애플릿을 사용한다. (그림 14)는 대본 연습 화면을 보여준다. 왼쪽 창에는 배역을 선택하고, 연습을 시작, 중지, 재시작을 할 수 있는 단추가 있다. 오른쪽 창에서는 대본의 내용이 보이며, 대본은 시간이 지나면서 천천히 위로 올라간다. 각자의 대사는 색상이나 글씨의 굵기를 이용하여 다른 배역의 대사와 구별되게 보여준다. 연습은 스크롤바를 이용하여 중지되거나 재시작 될 수 있으며, 이러한 이벤트가 발생할 경우에는 모든 클라이언트의 연습도 동시에 중지되거나 재시작 된다. 또한 연습의 속도도 연습 시작시 정해 줄 수 있다.



(그림 14) 대본 연습 화면

6.3 대본 작성 및 연습 시스템의 기능

본 논문에서 구현한 대본 편집 및 다자간 연습 시스템의 기능은 다음과 같다.

첫째, 기존의 대본 DTD를 다양한 데이터형을 제공하고 확장성이 좋은 대본 스키마로 변환하였으며, 이를 이용하여 웹 기반에서 XML에 관한 지식이 없는 시나리오 작가가 쉽게 XML 형식의 표준화된 대본을 작성할 수 있다. 둘째, 웹을 통해 작가나 배우들이 공동으로 대본을 작성, 수정 및 삭제 할 수 있으며, 쇼 기능을 사용하여 작성한 대본을 바로 확인할 수 있다. 셋째, 이벤트 동기화 기법을 사용하여 멀티캐스트가 가능하며, 서로 떨어져 있는 배우들이 이 시스템을 통해서 실제 연극처럼 함께 연습을 할 수 있다. 넷째, 웹페이지 스크롤링 동기화 기능을 제공하여, 대본 연습을 하고 있는 배우들 중에 한 배우가 대본 연습을 중단시키거나, 대본의 원하는 위치부터 연습을 재시작 시키는 등의 작업을 수행한다면 다른 배우들에게서도 같은 결과가 동시에 일어나도록 한다.

7. 결론 및 향후 연구

본 논문에서는 XML에 관한 지식이 없는 시나리오 작가도 웹에서 XML 스키마 구조에 맞는 XML 형식의 대본을 쉽게 작성하고 확인할 수 있는 문법 지향적인 대본 편집기를 개발하였다. 그리고, 이를 이용하여 작성된 대본을 배우들이 함께 외우고 연습하는데 도움을 주는 대본 연습 시스템을 구현하였다. 이를 위해 필요한 서버와 클라이언트들간의 이벤트 동기화 모델을 제안하고, 성능 평가를 통하여 구

현에 이용된 서블릿이 빠른 수행 속도를 보임을 확인하였다.

따라서, 이 시스템은 웹을 통해 시나리오 작가가 쉽게 XML 형식의 표준화된 대본을 작성하여 교환할 수 있을 뿐만 아니라, 쇼 기능을 사용하여 작성한 대본을 바로 확인 및 수정할 수 있으며, 서로 떨어져 있는 배우들이 웹을 통하여 실제 연극처럼 함께 연습할 수 있다는 장점이 있다.

향후에는 확장된 대본 구성 요소를 스키마에 적용할 수 있도록 스키마 구조를 동적으로 변경하는 기능을 연구하고, 여러 저작자들이 동시에 대본을 편집할 수 있도록 동시성 제어 시스템을 구현하며, 배우들이 상대역의 연기에 맞춰 연습할 수 있도록 실시간 상호작용 시스템에 관한 연구가 요구된다.

참 고 문 헌

[1] Jon Bosak, Shakespeare DTD, http://www.oasis-open.org/cover/bosakShakespeare200.html, 1999.

[2] XML Schema Spec., http://www.w3.org/XML/Schema.

[3] Riccardo Gusella and Stefano Zatti, "The Accuracy of the Clock Synchronization Achieved by TEMPO in Berkeley UNIX 4.3BSD," Vol.15, No.7, pp.847-853, 1989.

[4] Andrew S. Tanenbaum, "Modern Operating Systems," Prentice-Hall, 1992.

[5] DOM Spec., http://www.w3.org/DOM.

[6] XSL Spec., http://www.w3.org/style/XSL.

[7] SOAP 1.2 Spec., http://www.w3.org/TR/SOAP.

[8] Kent Sharkey and Scott Seely, "SOAP Cross Platform Web Service Development Using XML," Prentice Hall, 2001.

[9] Manfred Hauswirth and Mehdi Jazayeri, "A Component and Communication Model for Push Systems," Proc. of the 7th European Engineering Conference, Toulouse, France, pp. 20-38, September, 1999.

[10] George Coulouris, Dollimore and Kindberg Tim, "Distributed Systems Concepts & Design," Addison Wesley, 1994.

[11] Doreen L. Galli, "Distributed Operating Systems Concepts & Practice," Prentice Hall, 2000.

[12] Silberschatz Abraham and Galvin P. Baer, "Operating Systems Concepts," Addison Wesley, 1998.

[13] 김문석, 성미영, "동기적 웹 브라우저 공유를 지원하는 협동 작업 시스템", 정보처리학회논문지, 제8권 제3호, pp.283-288, 2001.

[14] 이점숙 외 2명, "웹 기반의 실시간 원격강의를 위한 서버와 클라이언트간의 웹 브라우저 동기화", 정보처리학회논문지, 제8권 제1호, pp.70-74, 2001.

[15] Flaviu Cristian, "Probabilistic Clock Synchronization," Distributed Computing, Vol.3, No.3, pp.146-158, 1989.

[16] Richard M. Haefel and David Chappell, "Java Message Service," O'Reilly, 2000.

[17] Ralf D. Schimkat, Stefan Muller and Wolfgang Kuchlin, "A Lightweight, Message-Oriented Application Server for the WWW," Proc. of the 2000 ACM Symposium on Applied Computing, Como, Italy, pp.934-941, March, 2000.

[18] Servlet Document, http://java.sun.com/products/servlet/2.3/javadoc/index.html.

[19] MSXML 4.0 Document, http://msdn.microsoft.com/library/.

김 신 우

e-mail : purian@dgu.edu

1997년 동국대학교 컴퓨터공학과(학사)

2000년 동국대학교 컴퓨터공학과(석사)

2000년~현재 동국대학교 컴퓨터공학과 (박사과정)

관심분야 : XML 및 웹, 스토리지 시스템, 데이터베이스

신 기 호

e-mail : khshin@appeal.co.kr

1997년 목원대학교 컴퓨터공학과(학사)

2002년 동국대학교 컴퓨터공학과(석사)

2002년~현재 어필텔레콤 S/W 연구원

관심분야 : XML 및 웹, 스토리지 시스템, 데이터베이스

박 성 은

e-mail : pse76@dgu.due

2000년 동국대학교 컴퓨터공학과(학사)

2002년 동국대학교 컴퓨터공학과(석사)

2002년~현재 동국대학교 컴퓨터공학과 (박사과정)

관심분야 : XML 및 웹, 스토리지 시스템, 데이터베이스

이 용 규

e-mail : yklee@dgu.edu

1986년 동국대학교 전자계산학과(학사)

1988년 한국과학기술원 전산학과(석사)

1996년 Syracuse University(전산학박사)

1978년~1983년 정보통신부 국가공무원

1988년~1993년 한국국방연구원 선임연구원

1996년~1997년 한국통신 선임연구원

1997년~현재 동국대학교 컴퓨터멀티미디어공학과 교수

관심분야 : XML 및 웹, 스토리지 시스템, 데이터베이스