韓國 컴퓨터情報學會 論文誌
第14卷 第1號, 2009. 1.

2009-14-1-4-2

웹 서버 연동의 실시간 디지털 정보 디스플레이 시스템

이 세 훈*

Realtime Digital Information Display System based on Web Server

SeHoon Lee *

요 약

이 논문에서는 인터넷상의 실시간 날씨 정보, 실시간 뉴스, 생활 정보 같은 콘텐츠나 특정 홈페이지 또는 자체에서 제작한 각종 홍보 및 광고용 플래시 및 동영상 콘텐츠를 자동 실행하는 DID(Digital Information Display)서비스를 설계 및 구현하였다. 제안된 DID 시스템은 클라이언트/서버 구조를 갖으며, 서버는 웹서버로부터 스케쥴 정보를 받아 클라이언트에게 관련 정보와 데이터를 전송하여, 일시적인 네트워크 장애를 극복할 수 있다. 또한 웹페이지 필터링 기능으로 특정 페이지의 부분적인 정보 추출을 하여 실시간으로 서비스 해줌으로써 DID서비스의 광고 효과를 증대할 수 있다.

Abstract

In this paper, we designed and implemented realtime DID(digital information display) system based on web server that displayed multimedia contents. The contents are weather, news information on the internet web sites and public relations or advertisements data on local systems. The DID system has client/server architecture that the server send to client that schedule informations and multimedia contents received form web server and the client displayed the contents though scheduled information. Therefore the systems overcome network fault for the mean time. Also, the system has realtime services of web page filtering function that extract the partial information of specific web pages.

▸ Keyword : DID(Digital Information Display), 디지털게시판(Digital Display Board), 실시간웹정보(Realtime Web Information)

• 제1저자 : 이세훈
• 투고일 : 2008. 11. 12, 심사일 : 2008. 11. 20, 게재확정일 : 2008. 12. 29.
* 인하공업전문대학 컴퓨터시스템과 교수
※ 이 논문은 2007학년도 인하공업전문대학 교내연구비지원에 의하여 연구되었음.

Ⅰ. 서 론

최근에는 관공서 혹은 금융 기관 등의 건물에는 LED 광고판을 대신하여 대형 LCD/PDP TV가 설치되어 있는 것을 볼 수 있다. 그리고 그 대형 디스플레이어들은 단순한 프로그램 방영이 아닌 다양한 콘텐츠 들을 디스플레이 하고 있으며 관공서에는 행사 홍보물이나 금융기관에서는 금융정보 등 것을 볼 수 있다[1,2]. 이것이 DID, 디지털 게시판이며 대형 PDP TV는 처리장치의 탑재로 더 이상 TV로서 만의 개념이 아닌 하나의 홍보기기로 탈바꿈 하여 그 시장이 형성되게 되었다. 외국에서는 DID 대신 디지털 영상장치(Digital Signage)라는 용어를 사용하고 있다[3,4,5].

미래를 배경으로 한 영화에서도 IT를 이용한 영상 기법들을 쉽게 발견할 수 있다. 길을 걷거나 버스, 또는 지하철을 타더라도 언제 어디서든 눈이 가는 도시의 곳곳에서 정보 및 광고와 접할 수 있는 세계를 보여주고 있는 것이다. 이러한 영화 속 상상이 언제부터인가 디지털 영상장치를 통해 실제로 현실 세계에서 이뤄지고 있다.[6,7,8,9,10].

디지털 영상장치는 소프트웨어, 하드웨어, 콘텐츠, 네트워크 등 다양한 정보기술이 복합적으로 이뤄져 정보와 광고를 전달하는 디지털 영상장치라고 할 수 있다. DID는 하드웨어의 측면을 강조한 용어라고 할 수 있으나, 여기서는 DID라는 용어를 디지털 영상장치와 같은 의미로 사용한다.

이러한 DID의 환경구축 방식을 보자면 초기의 DID는 커다란 PDP TV와 VTR등의 매체 재생기를 통하여 미리 저장되어 있는 홍보영상 등을 디스플레이 해주는 것뿐이었다. 기존의 TV와 VTR의 개념에서 크게 못 벗어난 방식으로 크게 특이한 점은 없었다. 그 다음 나온 것이 임베디드 개념으로 출시되어 처리장치와 저장장치가 내장되어 있는 DID 보드로 외부장치 없이 동작 가능한 일체형 DID 였다. 처리장치로 사진과 텍스트, 동영상 등 이종매체를 적절하게 디스플레이 하는 프로그램으로 홍보효과를 살렸으며 USB나 네트워크를 이용하여 재생 콘텐츠의 주기적인 업데이트가 가능하게 되어 하나의 기기로서 산업이 형성되었으며 새로운 세대의 홍보매체가 되었다[1,2,3].

이 논문의 DID 시스템은 임베디드 적인 요소의 DID에 웹 기술을 적용하여 저장매체의 디스플레이 뿐만 아니라 실시간적으로 뉴스 등의 데이터 처리와 영상처리 기술이 가능하다. 본 논문의 DID 시스템은 웹으로부터 실시간으로 얻어오는 뉴스나 날씨, 도서 등의 콘텐츠를 통해 단순 반복적인 내용의 전달을 넘어 고객들에게 필요한 생활 정보로 눈길을 끌어들여 홍보효과를 높일 수 있다. 또한 기존의 DID 시스템의 경우 콘텐츠의 업데이트 시에 DID용으로 콘텐츠를 새로 제작해야 하는 불편함과 자원의 낭비가 있지만 본 DID 시스템의 경우 해당 관공서의 홈페이지의 내용을 그대로 디스플레이 할 수 있으므로 따로 제작하여 일일이 업데이트 할 필요가 없다.

Ⅱ. DID 시스템 설계 및 구현

2.1 시스템 구성

전체 시스템 구성도는 그림1 과 같다. 서버와 클라이언트는 소켓을 통해 연결하게 되고, 서버에서는 작성한 스케줄 파일 및, 콘텐츠 파일을 전송하게 된다.

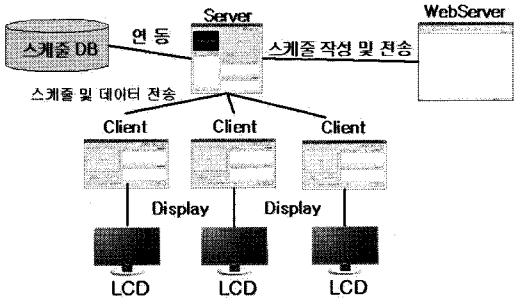

그림 1. 시스템 구성도
Fig. 1. The DID system configuration

또한 클라이언트에게 디스플레이 제어 메시지를 보내, 클라이언트에 연결되어 있는 LCD에 콘텐츠를 디스플레이 및 Stop 할 수 있다. 서버는 스케줄 DB와 연동하여, 미리 작성해놓은 스케줄 파일을 불러올 수 있고, 스케줄을 새로 작성하여 저장 할 수 있다.

웹서버과 연동하여, 원격으로 웹상에서 스케줄을 작성하고, 서버 시스템으로 전송할 경우 서버 시스템 은 이를 받아서 연결된 클라이언트 시스템으로 전송하게 되고, 클라이언트 시스템에서는 전송받은 스케줄 파일을 LCD 에 디스플레이 하게 된다.

2.2 서버 시스템

2.2.1 서버 시스템의 기능

서버 시스템 에서는 동영상, 그림, 웹페이지 등의 Mdeia 콘텐츠와 자막으로 사용할 Text 콘텐츠의 스케줄을 작성하는 기능이 있다. 이 스케줄은 XML 파일 형식으로 저장이 되며, 클라

이언트 시스템으로 스케줄을 전송 시에 XML 파일과 Media 및 Text 콘텐츠를 동시에 전송하게 된다. 또한 스케줄이 전송되면 클라이언트 시스템은 전송받은 스케줄로 변환이 된다.

서버 시스템은 Display, Stop 같은 명령어를 통해 클라이언트 시스템에 연결된 패널을 제어할 수 있다.

2.2.2 서버 시스템의 구조

DID 서버 시스템은 그림2에서 보는바와 같이 구성되어 있다. 기본구조는 'CI_Board_ServerDlg' 라는 프레임워에 하위의 프레임들이 연결되어 있는 구조이다. 그림 2는 서버 시스템의 Class Diagram 구조이다.

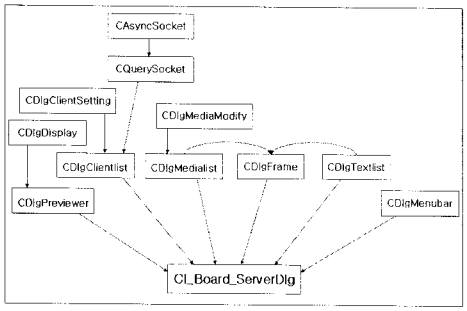

그림 2. Class Diagram of Server System
Fig. 2. 서버 시스템의 클래스 다이어그램

CI_Board_ServerDlg는 그림2와 같이 메인다이얼로그 등의 하위 속성들을 각각의 화면 크기에 맞춰 생성하며 자식 다이얼로그로 운영. 각 다이얼로그에서 오는 메시지들을 처리한다. 스케줄 관리가 주이다. 최상단에 위치한 다이얼로그로 CDlgMenubar는 스케줄에 관련된 버튼과 ["New", "Open", "Path", "Save", "SaveAs", "MapInfo", "Exit"]라는 메뉴를 가지고 있다.

리스트에 추가된 리스트를 더블클릭하면 이곳에서 미리보기 기능으로, 시연이 되며, CDlgPreviewer가 담당한다. CDlgClientlist는 서버에 의해 제어되는 클라이언트의 리스트들을 나타내며, 클라이언트와의 소켓통신을 담당한다.

디스플레이 될 영상의 형태를 설정하는 CDlgFrame은 상, 하의 자막을 추가할 수 있으며, 총 7개의 타입으로 미디어 화면을 하나 또는 두 개로 나눌 수 있다.

CDlgFrame에서 선택되는 프레임에 해당되는 미디어의 리스트들을 나타내는 CDlgMedialist은 콘텐츠들(사진, 동영상, 웹페이지 등)을 담당한다. 또한, 텍스트의 리스트들과 자막 처리는 CDlgTextlist에서 처리한다.

그림 3은 DID 서버 시스템의 설정 화면으로 프레임의 설정 및 선택은 체크박스와 콤보박스에 해당되는 멤버변수를 두어, 각 컨트롤이 수행될 때마다 값을 줌과 동시에 각 프레임에 해당되는 버튼을 Visible에 변화를 두어 화면에 디스플레이 되도록 한다. 스케줄의 저장시에는 메인다이얼로그인 CI_Board_ServerDlg에서 멤버변수의 값을 참조한다. 각 프레임의 형태는 버튼형식으로 되어 사용자의 클릭이 있을 경우 UM_SELECT_FRAME이라는 메시지와 함께 선택된 프레임값을 CI_Board_ServerDlg에 보낸다. CI_Board_ServerDlg에서는 해당 메시지 핸들러 함수를 호출하여, 전달된 프레임값에 따라 CDlgMedialist 혹은 CDlgTextlist의 함수를 호출하여 프레임이 선택되었음을 나타낸다.

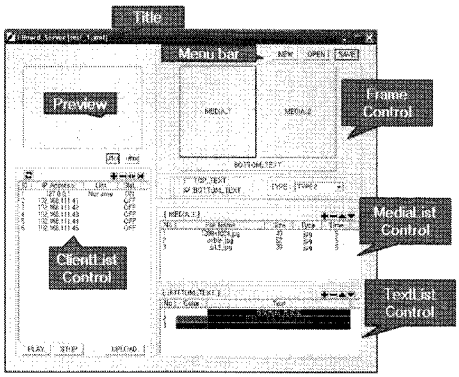

그림 3. DID 서버 시스템 설정
Fig. 3. DID server system control

파일의 추가 및 수정시에는 CDlgMediaModify라는 팝업 다이얼로그를 이용하여 파일의 이름 및 수행시간에 대한 정보를 얻어와 리스트에 표시한다. 이때 파일일 경우 파일다이얼로그를 띄워 해당 파일의 파일 이름, 경로를 얻어온다. 이 때 만약 파일의 경로가 현 작업경로와 다르면 경고메시지를 띄우고 추가하지 않는다. 작업경로는 CI_Board_ServerDlg에 의해 업데이트된다. 만약 파일이 동영상일 경우 DirectX가 제공하는 함수를 이용하여 동영상의 플레이 시간을 얻어온다. 인터넷상의 웹페이지 일 경우에는 URL값을 입력받는다.

리스트의 목록들은 MediaList라는 객체와 연동된다.

MediaList객체는 해당 미디어리스트의 이름(Media_1 or Media_2)과 리스트의 개수, 그리고 미디어의 크기, 시간, 타입, 이름에 각각의 배열클래스를 가지고 있어 값을 저장한다. CDlgMedialist의 MediaList 멤버 변수는 CI_Board_ServerDlg의 MediaList 멤버 변수와 연동되어 스케줄 저장 시 반영된다.

자막의 추가 및 수정시에는 CDlgTextAdd라는 팝업 다이얼

로그를 이용하여 자막의 내용, 색에 대한 값을 얻어와 리스트에 표시한다. 리스트의 목록들은 TextList라는 객체와 연동된다. 해당 텍스트리스트의 이름과 리스트의 개수, 그리고 텍스트의 색, 글귀에 각각의 배열클래스를 가지고 있어 값을 저장한다.

클라이언트의 추가, 수정에서 클라이언트의 추가 및 수정 이벤트시에 클라이언트세팅이라는 팝업 다이얼로그를 띄워 클라이언트의 ID와 IP 주소 값을 얻어온다. 만일 리스트 상에 동일 ID 혹은 IP 주소가 있다면, 경고메시지를 띄우고 추가 작업을 하지 않는다. 클라이언트 리스트의 관리에서는 클라이언트의 리스트만 관리할 뿐 아니라 클라이언트와의 통신도 관리한다. 이는 멤버 변수를 두어 리스트에 클라이언트에 추가될 때 마다 소켓을 새로 생성하여 객체배열에 추가한다. 추가시에 클라이언트 리스트의 핸들러 윈도우의 포인트를 전해 배열에 추가된 소켓들의 상태를 알 수 있게 한다.

소켓은 CAsyncSocket을 상속받은 CQuerySocket을 이용한다. 클라이언트 연결 이벤트 시에 CObArray에서 해당 소켓을 찾아 클라이언트와 연결시킨다.

소켓으로부터 메시지가 오거나 소켓이 끊길때 등의 이벤트에 발생하는 CQuerySocket의 멤버 함수인 OnRecieve 및 OnClose등의 멤버 함수에서는 UM_SOCKET_RECEIVE, UM_SOCKET_CLOSE의 메시지를 소켓이 생성될때에 주어진 핸들러윈도우 멤버 변수를 이용하여 소켓의 상태를 CDlgClientlist에 전한다.

표 1. 메시지 패킷
Table 1. Message packet

1Byte	1Byte	1Byte	N Byte	1Byte
ID	FUNC	SIZE	DATA	CHECK SUM

메시지 패킷은 표1 과 같다. 첫 번째 1Byte는 클라이언트 시스템 의 ID를 의미한다. FUNC 바이트는 서버 시스템에서 클라이언트 시스템 으로 보내는 메시지의 종류를 의미한다. 그리고 마지막에 삽입되는 1Byte는 오류처리 코드로서 메시지가 잘 수신이 되었는지를 검사하는 패리티 비트이다.

스케줄의 관리는 스케줄이 저장되거나 저장되어 있던 스케줄이 열린다면 타이틀바에 표시한다. 스케줄에 관한 각 다이얼로그에 변화가 있을 시에 UM_CHANGED_SCHEDULE 메시지를 서버 다이어로그로 보내 해당 메시지 핸들러 함수에서 타이틀 바에 현재 스케줄의 변화가 있음을 나타내준다. CDlgMenubar의 메뉴중 "Path"를 선택하면, 작업경로의 확인 및 변경이 가능하다. 기본 작업경로는 프로젝트 내 "Define.h"에 정의 되어 있는 "C:\i_Board_WorkingFile\"이다.

만약 작업경로의 변경시에는 UM_PATH_SETTING 메시지가 CI_Board_ServerDlg로 전달되어 메인 다이얼로그의 작업경로 및 다른 다이얼로그의 작업경로가 갱신된다. 동시에 만약 현재 작업중이던 스케줄이 있다면 저장 여부를 확인하고 종료 후 새로운 스케줄로 변경한다.

2.3 클라이언트 시스템

서버에 문제가 있을경우를 대비해서 클라이언트 시스템에서도 스케줄을 작성 및 저장 할 수 있다. 스케줄 파일은 XML 형식으로 저장되며 서버 시스템에서 작성된 스케줄 파일의 형식과 같다.

서버 시스템으로부터 스케줄 및 콘텐츠 파일을 수신받고, 이 스케줄의 형식에 맞게 연결된 패널에 디스플레이 하게 된다.

그림 4에서 보는 바와 마찬가지로 클라이언트 시스템 은 기본 CI_Board_ClientDlg 프레임 위에 하위 프레임들이 연결되어 있는 구조이다.

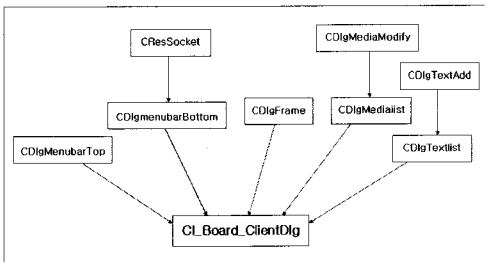

그림 4. 클라이언트 시스템의 클래스 다이어그램
Fig. 4. Class Diagram of Client System

CI_Board_ClientDlg는 메인다이얼로그로서 하위 필요 속성들을 각각의 화면 크기에 맞춰 생성하며 자식 다이얼로그로 운영한다. 각 다이얼로그에서 오는 메시지들을 처리한다. 스케줄 관리가 주이다.

최상단에 위치한 다이얼로그인 CDlgMenubarTop은 스케줄에 관련된 버튼과 파일에 관련된 메뉴를 가지고 있다. 메뉴바 하단은 주로 서버와의 통신에 관한 작업을 주로 한다. 현재 클라이언트의 ID와 연결된 서버의 IP주소 등을 나타낸다. 서버와 연결이 되어 있지 않은 상태라면 ID는 디폴트 값으로 1을 가지며 IP주소에는 "DISCONNECTED" 라는 글귀로 현재 연결 상태가 아님을 나타낸다. 그 외의 나머지는 서버쪽과 유사하다.

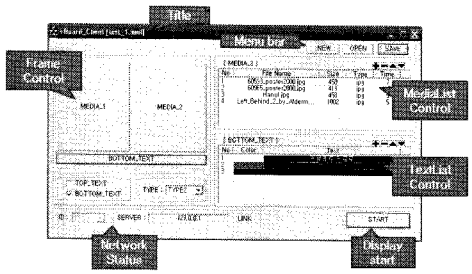

그림 5. 클라이언트 시스템 화면
Fig. 5. Screen of Client System

CDlgMenubarBottom에서는 서버와의 통신을 담당한다. 다이얼로그 생성 시부터 CResSocket이라는 CAsyncSocket을 상속받은 소켓을 Listen하여 서버로의 접속에 대기한다. CResSocket 역시 서버의 소켓을 설정하는 것과 같이 CDlgMenubar Bottom의 핸들러 윈두에 대한 값을 받아 멤버 변수로 두어 서버로부터의 Accept가 있을시에 CDlgMenubarBottom에 UM_SOCKET_ACCEPT이라는 메시지를 전달해 연결이 되었음을 알린다.

메시지를 받은 메뉴바 하단은 해당 핸들러 함수를 호출하여 CChildSock이라는 소켓을 CResSocket에서 Accept한 값을 넘겨 서버와의 통신을 담당하게 한다. 이 CChildSock 역시 CDlgMenubarBottomd의 핸들러 윈도우 값을 넘겨주어 메시지가 들어오거나 파일이 수신 될 시에 알 수 있게 한다. 서버로부터 Upload 메시지가 온다면 파일을 받을 준비를 하고, 리스트 파일을 우선 전송 받는다. 또 리스트 파일에 해당되는 콘텐츠 파일(사진, 동영상, html 등)을 다운 받고 전송이 완료되면 파일들을 검사 후에 새로 전송된 스케줄로 전환한다. 서버로 부터의 Start 명령 혹은 클라이언트 Application의 Start 버튼의 이벤트가 발생 하면 CI_Board_ClientDlg의 자식 다이얼로그인CDlgDisplay를 이용하여 현재 열려있는 스케줄에 해당하는 파일들을 디스플레이 한다.

동영상은 DirectX9.0의 API들을 활용하여 재생을 하며, html 파일 혹은 인터넷상의 웹페이지는 CWebBrowser2을 이용한다. Display 역시 MediaList 객체와 TextList 객체를 이용하여 실연시킬 객체를 참고한다.

2.4 스케줄 DB

다음은 스케줄 파일을 저장하고 있는 XML 파일 및 XML 파일의 스키마를 설명하겠다.

리스트 1. test.xml 파일
List 1. test..xml file

```
〈?xml version="1.0" encoding="euc-kr" ?〉
 〈schedule name="test.xml"
        path="C:\i_Board_WorkingFile\"〉
  〈frame text="0" type="1" /〉
 〈media position="3" fileCount="1"〉
  〈file_1 mediaType="URL"
    fileName="http://www.naver.com"
 fileSize="0" duration="5" /〉
  〈/media〉
  〈/schedule〉
```

다음은 기본적으로 1번 media 타입으로 미디어는 naver를 디스플레이 할 수 있는 스케줄DB를 보여주고 있다. 루트 앨리먼트인 schedule 앨리먼트의 name 속성은 스케줄 파일의 이름, path 속성은 스케줄 파일이 저장되어 있는 경로를 나타내고 있다.

frame 앨리먼트는 기본적으로 Text Contents 와 Media Contents의 수를 나타낸다. 0일 경우에는 Top Text와 Bottom Text가 설정되어있지 않았을 경우 경우를, 1일 경우는 Top Text 만 설정 되었을 경우, 2일 경우는 Bottom Text 만 설정되어있을 경우, 3일 경우에는 Top Text 와 Bottom Text 가 둘 다 설정 되어있을 경우를 나타낸다. Type 속성은 어플리케이션의 1~7번 타입을 각각 나타낸다.

media 앨리먼트는 동영상 및 그림파일을 나타내는 앨리먼트이다. position 속성은 각각의 콘텐츠가 가지고 있는 고유 위치 값이고 fileCount 속성은, 콘텐츠의 개수를 의미한다. file_1 앨리먼트는 Media 콘텐츠 파일의 정보를 가지고 있는 앨리먼트로 MediaType 속성은 Media Content 의 타입이고, fileName 속성은 파일의 이름을, fileSize 속성은 파일의 크기를, duration 속성은 파일의 재생시간을 각각 나타낸다.

리스트 2. 스키마
List 2. Scheme

```
〈?xml version="1.0" encoding="euc-kr"?〉
〈xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSche
ma"〉
〈xsd:element name="schedule"
                    type="ctSchedule"/〉
 〈xsd:complexType name="ctSchedule" 〉
  〈xsd:sequence〉
   〈xsd:element name="frame"〉
    〈xsd:complexType〉
     〈xsd:attribute name ="text"
```

```
                type ="stFrame_text"/>
    <xsd:attribute name ="type"
                type ="stFrame_type"/>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="message"
               type="ctMessage"
               minOccurs="0"
               maxOccurs="2"/>
  <xsd:element name="media"
        type="ctMedia" minOccurs="0"
        maxOccurs="2"/>
 </xsd:sequence>
  <xsd:attribute name="name"
                 type="xsd:string"/>
  <xsd:attribute name="path"
                 type="xsd:string"/>
</xsd:complexType>
```

루트 앨리먼트인 schema의 하위 앨리먼트로는 각각 frame, message, media 가 올 수 있다. 각각의 스케줄 설정에 따라, message 앨리먼트와 media 앨리먼트는 최소 0개에서 2개까지 올 수 있다.

리스트3은 frame 앨리먼트의 설정 부분을 보고 있다. text 속성의 값은 0~3, Type 속성의 값을 1~7 사이의 값을 설정 되어 있어야 한다.

리스트 3. 스키마 (frame 앨리먼트 부분)
List 3. Scheme(part of frame element)

```
<xsd:simpleType name="stFrame_text">
 <xsd:restriction base="xsd:int">
  <xsd:minInclusive value="0" />
  <xsd:maxInclusive value="3" />
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="stFrame_type">
 <xsd:restriction base="xsd:int">
  <xsd:minInclusive value="1" />
  <xsd:maxInclusive value="7" />
 </xsd:restriction>
</xsd:simpleType>
```

리스트 4는 message 앨리먼트의 스키마를 나타내고 있다. topLine, bottomLine 앨리먼트는 최소 0개에서 최대 무제한으로 나타낼 수 있고, (실질적으로 Text Contents 의 개수는 제한되어 있다.) color_r, color_g, color_b 속성 값은 0~255 의 값으로 제한되어있다.

리스트 4. 스키마 (message 앨리먼트 부분)
List 4. Scheme(part of message element)

```
<xsd:complexType name="ctMessage">
<xsd:sequence>
  <xsd:element name="topLine"
               type="stLine"
               maxOccurs="unbounded"
               minOccurs="0"/>
 <xsd:element name="bottomLine"
               type="stLine"
               maxOccurs="unbounded"
               minOccurs="0" />
 </xsd:sequence>
 <xsd:attribute name="position"
               type="xsd:int"/>
 <xsd:attribute name="lineCount"
               type="xsd:int"/>
</xsd:complexType>
 <xsd:complexType name="stLine">
 <xsd:simpleContent>
  <xsd:extension base="xsd:string" >
   <xsd:attribute name="text"
                 type="xsd:string"/>
   <xsd:attribute name="color_r"
                 type="stColor"/>
   <xsd:attribute name="color_g"
                 type="stColor"/>
   <xsd:attribute name="color_b"
                 type="stColor"/>
  </xsd:extension>
 </xsd:simpleContent>
</xsd:complexType>

<xsd:simpleType name="stColor">
 <xsd:restriction base="xsd:int">
  <xsd:minInclusive value="0" />
  <xsd:maxInclusive value="255" />
 </xsd:restriction>
</xsd:simpleType>
```

마지막으로 media 앨리먼트 부분의 스키마 이다. file_1, file_2 앨리먼트의 속성인 mediaType 의 값은 jpg, png, gif, bmp, avi, mpeg, wmv, Indeo, QuickTime, html, URL로 제한되어 있다. 이는 지원 가능한 그림 및 동영상 파일의 확장자를 의미한다.

리스트 5. 스키마 (media 엘리먼트 부분)
List 5. Scheme(part of media element)

```
<xsd:complexType name="stline">
 <xsd:simpleContent>
  <xsd:extension base="xsd:string" >
   <xsd:attribute name="mediaType"
                  type="stMediaType"/>
   <xsd:attribute name="fileName"
                  type="xsd:string"/>
   <xsd:attribute name="fileSize"
                  type="xsd:int"/>
   <xsd:attribute name="duration"
                  type="xsd:int"/>
  </xsd:extension>
 </xsd:simpleContent>
</xsd:complexType>
<xsd:simpleType name="stMediaType">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="jpg"/>
  <xsd:enumeration value="png"/>
  <xsd:enumeration value="gif"/>
  <xsd:enumeration value="bmp"/>
  <xsd:enumeration value="avi"/>
  <xsd:enumeration value="mpeg"/>
  <xsd:enumeration value="Indeo"/>
  <xsd:enumeration value="QuickTime"/>
  <xsd:enumeration value="html"/>
  <xsd:enumeration value="URL"/>
  <xsd:enumeration value="wmv"/>
 </xsd:restriction>
</xsd:simpleType>
```

2.5 웹 페이지 필터링

서버에서 운용되는 JSP 프로그램으로 DID 기기에서 디스플레이 할 수 있는 웹페이지가 생성되며 인터넷으로 서비스되는 특정 홈페이지의 내용을 읽어올 수 있으므로 그 데이터를 일정한 틀에 맞추어 생성하여 따로 업데이트 작업이 필요없게 한다. 이 JSP 프로그램의 웹페이지 데이터 수집 부분에서는 뉴스, 날씨, 광고용 플래시, 공지 사항 등 수집하고자 하는 데이터가 존재하는 인터넷상의 웹페이지의 URL로 접속하여 웹페이지의 소스를 읽어 온다.

이렇게 수집한 웹페이지의 소스 중에서 DID에서 디스플레이 할 부분만 추려내어 정렬시킨 후 각 사용처에 맞는 텍스트와 디자인을 더하여 DID의 영상에 맞게 편집한다.

리스트 6. JSP 프로그램의 핵심코드
List 6. Core code of JSP program

```
URLurl=newURL("http://www.WebPage.com/Test1.php
");
URLConnection urlC = url.openConnection();
urlC.setUseCaches(false);
urlC.connect();
InputStreamReader  isr = null;
BufferedReader br =  null;
isr=newInputStreamReader(urlC.getInputStream())
br = new BufferedReader(isr);
String temp;
while((temp=br.readLine()) != null){
  Filtering Logic;
}
```

리스트 6은 JSP파일의 웹 페이지 소스를 필터링하는 부분의 핵심코드이다. URL 클래스를 이용 해당 웹페이지에 접속하여 웹페이지의 소스를 한 줄씩 읽어 들이면서 각각의 필터링 로직에 따라 DID에서 서비스 해줄 콘텐츠를 위한 소스만을 추려내어 출력해 준다.

2.6 디스플레이

영상처리 부분에서는 일종의 클라이언트로 동작을 하며 DID 기기에서 네트워크로 서버에 있는 JSP파일에 접속하여 웹페이지를 디스플레이 한다.

우선 DID 프로그램에서는 웹 브라우저 객체를 얻어와 접속한 페이지를 디스플레이 할 환경을 구축한다. 또한 DID기기의 모니터 부분의 크기 정보를 얻어와 디스플레이 할 콘텐츠에 따라 적절히 화면을 분할한다. 분할한 화면마다의 크기에 맞춰 앞서 웹 브라우저 객체를 이용 서버의 JSP 파일에 접속하여 출력되는 웹페이지를 디스플레이 해준다.

이 동작을 Timer를 이용하여 정해둔 시간 간격마다 화면을 바꿔간다. 물론 콘텐츠에 따라 화면 분할을 다르게 할 수도 있다. 리스트 7은 이러한 영상처리 프로그램의 핵심코드를 나타낸다.

리스트 7. 영상처리 프로그램의 핵심코드
List 7. Core code of Video processing program

```
CRect rect;
GetDesktopWindow()->GetWindowRect(&rect);
ShowWindow(SW_SHOWMAXIMIZED);
```

```
SetTimer(interval time);
page=1;
OnTimer(){
  switch(page){
    case 0:
      WebBrowser.MoveWindow(coordinates, size);
      WebBrowser.Navigate("URL of 0_JSP");
      break;
          ...
    case N:
      WebBrowser.MoveWindow(coordinates, size);
      WebBrowser.Navigate("URL of N_JSP");
      break;
  }
  page++;
  if(page==N);
    page = 0;
}
```

III. 실험 및 평가

3.1 실험 개요

이 장에서는 설계한 DID의 서비스에 대한 실험을 한다. 실험 환경으로는 서버는 Intel P4 3Ghz, 메모리 1GB인 환경을 구축하였다. 서버와 클라이언트 프로그램은 MS Visual C++를 사용해 MFC로 개발하였다.

시스템을 테스트하기 위해서는 몇 가지의 환경이 갖추어져 있어야 한다. 우선 Media 콘텐츠의 재생을 위해 DirectX SDK에 포함되어있는 Direct Show가 설치되어 있어야 한다. 이 Direct Show는 Mircosoft Windows 플랫폼에 있어서의 미디어 스트리밍 아키텍처로 멀티미디어 스트림의 고품질인 캡춰와 재생을 실연한다. 지원가능 포맷으로는 ASF,AVI,DV,MPEG, MP3,WMA/WMV,WAV가 있다.

비디오 및 오디오 가속화 하드웨어가 사용 가능한 경우에는 하드웨어를 자동적으로 검출해 사용하지만, 가속화 하드웨가 없는 시스템도 지원하고 있다. 테스트를 위해 DirectX 9.0c 버전을 사용하였다.

Direct Show 이외에도 MS-XML가 설치되어있어야 하는데, 이 프로그램은 XML 관련 프로그램으로 XML 파서라고도 한다. 이 MS-XML프로그램이 스케줄 문서를 분석하고 Save 및 Open을 위해 사용 되고 있다.

3.2 실험

DID에서 디스플레이 할 콘텐츠는 임시로 포털 사이트 네이버의 각종 뉴스 서비스와 날씨 서비스 및 몇가지의 그림 파일, 동영상 파일을 이용하였다.

그림 6,7에서는 네이버에서 서비스 하고 있는 웹페이지를 나타내고 있다. DID는 웹페이지 필더링 기술을 적용하여, 그림에서 보는바와 같이 빨간색 네모칸의 부분만을 실시간으로 가져와, 디스플레이 하게 된다.

이로써 본 DID 프로그램은 정상적으로 웹과 연동하여 실시간 데이터의 정확한 디스플레이가 됨을 있음을 알 수 있다.

그림 6. 네이버 날씨 서비스 페이지
Fig. 6. Naver Weather Page

그림 7. 웹페이지 (네이버 날씨 서비스)
Fig. 7. Web Pages(Naver)

그림 8은 실질적으로 LCD를 이용한 디스플레이 화면을 보여주고 있다. 디스플레이의 타입은 Type2를 적용하여, 윈

쪽에는 동영상을, 오른쪽에는 이미지 파일을 재생하고 있다. 화면 하단에는 Bottom Text 콘텐츠를 추가하여, 텍스트를 화면에 디스플레이 한다.

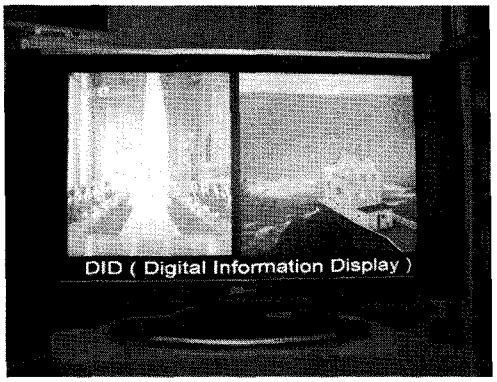

그림 8. 다중 비디오 디스플레이 실제 화면
Fig. 8. Real pictures of multiple video display

IV. 결론

현대 생활에서의 광고의 중요성은 이루 말할 수 없다. 하지만 현란하며 동적인 매체들이 가득한 현대 사회에 현대인들은 더 이상 포스터나 LED전광판 매체는 눈길을 끌기 힘들다. 이런 상황과 함께 대형 LCD패널 가격의 하락과 기술의 발달로 디지털 영상 게시판산업이 점차 활성화 되고 있다.

이 논문에서는 인터넷 상의 콘텐츠를 활용하여 뉴스 및 날씨, 기타 플래시등 다양한 매체를 실연할 수 있음을 보였다. 이러한 웹페이지 매체의 디스플레이는 DID가 서비스 될 장소에서 일반인들이 관심을 가질 수 있는 뉴스 및 날씨, 다양한 콘텐츠의 정보를 통하여 시선을 끌어들여 홍보효과를 높일 수 있는 시너지 효과를 볼 수 있을 것이라 생각한다.

제안 시스템은 DID에 재생되는 매체들이 인터넷에서 서비스되는 웹페이지의 소스를 그대로 읽어와 페이지를 생성하여 디스플레이 하는 방식이다. 이러한 방식으로 관공서 등이나 특정 기업의 경우 자사의 홈페이지내의 일정 영역을 사용하여 디스플레이 해주는 공지사항이나 홍보물들 같은 경우 한 번의 커스터마이징 작업으로 따로 내용의 업데이트 작업 없이 그대로 사용할 수 있다. 이는 하나의 콘텐츠로 온·오프라인을 동시에 디스플레이 할 수 있으므로 대단히 경제적이라고 보며 앞으로 DID 산업에서의 그 활용성을 기대해 본다.

참고문헌

[1] Digital Signage Expo, *http://www.digitalsignageexpo.eu/index.htm, 2008.*

[2] Society Information Display, *http://www.sid.org, 2008.*

[3] webpavement, *http://www.webpavement.com, 2008.*

[4] Stephen S. Intille, *"Change Blind Information Display for Ubiquitous Computing Environments,"* MIT Home of the Future Consortium, UbiComp, 2002.

[5] Gheorghe Berbecel, "Digital Image Display: Algorithms and Implementation, Wiley Sid Series in Display Technology," 2003.

[6] Isenzo, Mediacastor: Digital Signage Software, *http://www.isenzo.com, 2008.*

[7] Bob Brittan, *"Deploying Digital Signage in Your Company.(Contact Center Technology): An article from: Customer Interaction Solutions,"* Apr. 2007.

[8] Chad Munce, *"Digital Signage Comes into Focus. (Technology Use by Retail Industry) : An Article From: Digital Imaging Digest,"* Aug. 2005.

[9] Screen Digest Ltd. *"Digital Signage in Europe: Opportunities for Digital Out-of-home Advertising,"* Aug. 2008.

[10] 김기백, 최종호, "DirectShow 기반 UCC 컨텐츠 제작 시스템," 한국컴퓨터정보학회 논문지, 제 13권, 제 2호, 127~134쪽, 2008년 3월.

저 자 소 개

이 세 훈

1985: 인하대학교 전자계산학과 졸업
1987: 인하대학교 대학원 전자계산학과 졸업
1996: 인하대학교 대학원 전자계산공학과 졸업(공학박사)
1987~1990: 해병대 분석장교
1991~1993: (주)비트컴퓨터 연구소
2001~2002: NJIT, USA visiting scholar
1993~현재: 인하공업전문대학 교수
관심분야 : 임베디드시스템, 유비쿼터스 컴퓨팅, 상황인식서비스