인터넷상의 동영상 메일을 재생하기 위한 실시간 연출 기법 연구

임 영 환† · 이 선 혜†† · 임 명 수††

요 약

본 논문은 기존의 텍스트 위주였던 인터넷 메일을 한 단계 넘어 동영상, 음향, 그래픽 등 다양한 멀티미디어 데이터를 이용한 프리젠테이션 메일을 제안한다. 이를 개발함에 있어 가장 큰 문제점은 프리젠테이션 메일을 구성하는 요소가 일반적으로 거대한 용량을 가지는 멀티미디어 데이터라는 것이다. 거대한 용량의 멀티미디어 데이터는 전송 및 저장에 있어서 많은 문제점을 갖게 되는데, 이를 극복하기 위해 실제 데이터 부분과 제어 데이터를 분리시켜 제어 프로그램만을 전달하고 실제 데이터는 보낸 사람의 컴퓨터에 있거나 고속 접속이 가능한 원격지 서버에 저장되어 실제 연출할 때에 서버로부터 전송하는 방식을 취한다. 이러한 상황에서 제어 프로그램에 포함된 하이퍼프리젠테이션을 인터넷상에서 실시간으로 재생하기 위한 버퍼관리 및 쓰레드 스케줄링 기법을 제안하였다. 그리고 또 다른 문제는 멀티미디어 연출을 제어하는 방법은 일반 사람들이 사용할 수 있을 정도로 쉬워야 한다는 점이다. 본 논문은 제어 프로그램을 제작하는 도구로써 VIP(Visual Interface Player)를 이용하였으며 그것을 LAN기반에서 구현하고 실험한 결과를 제시하였다.

A Study on a Real Time Presentation Method for Playing of a Multimedia Mail on Internet

Young-Hwan Lim† · Seon-Hye Lee†† · Myung-Su Lim††

ABSTRACT

In this paper, a multimedia mail including video, sound, graphic data has been proposed as the next generation mail of the text based mail. In order to develop the multimedia mail, the most outstanding problem is the fact that the multimedia data are too huge to send them to the receiving end directly. The fact of big data may cause many problems in both transferring and storing the data of the multimedia mail. Our main idea is to separate between a control program for the multimedia presentation and multimedia data. Since the size of a control program is as small as a plain text mail, it has no problem to send it attached to the internet mail to the receiver directly. Instead, the big multimedia data themselves may remain on the sender's computer or be sent to a designated server so that the data may be transferred to the receiver only when the receiver activates the play of the multimedia mail. In this scheme, our research focus is placed on the buffer management and the thread scheduling for the real time play of the multimedia mail on internet. Another problem is to provide an easy way of editing a multimedia presentation for an ordinary people having no programming knowledge. For the purposed, VIP(Visual Interface Player) has been used and the results of multimedia mail implemented on LAN has been described.

† 종신회원 : 숭실대학교 컴퓨터학부 교수
†† 준 회 원 : 숭실대학교 대학원 전자계산학과
논문접수 : 1999년 1월 15일, 심사완료 : 1999년 2월 11일

1. 서 론

초고속 정보통신망의 궁극적인 목표는 사용자가 원하는 서비스를 언제, 어디서나 원하는 형태로 제공받을 수 있게 하자는 것이다. 이를 위해서 초고속정보통신망의 미들웨어가 절대적으로 필요하다는 사실은 재론의 여지가 없다. 그러나 중요한 것은 그 시대에 맞는 서비스를 제공할 수 있느냐 하는 문제이다. 우선 인터넷이 일반화되어 있고 통신망의 전송능력도 점차 향상되어 가고 있음으로 일차적으로 완벽한 오디오/비디오 중심의 멀티미디어 서비스를 어떻게 인터넷에서 제공할 수 있는가 하는 것이 관건이라고 할 수 있겠다. 따라서 지금의 인터넷 속도를 고려하여 프리젠테이션 제어 프로그램은 저속도의 인터넷으로 보내고 실제 멀티미디어 데이터는 고속 접속이 가능한 서버에 둠으로 실제 연출할 때 실시간 연출이 가능하도록 하는 것이 바람직하다. 지금까지 개발된 것은 독립형 시스템에서 프리젠테이션을 연출하는 방법에 중점을 맞추어 왔다. 그러나 분산 환경에서의 멀티미디어 데이터는 시간적인 요구가 매우 엄격하고 그 양이 매우 방대하기 때문에 통신망을 통하여 전송하는데 시간이 많이 소요될 뿐만 아니라 받는 측에서 저장할 기억장치의 문제에 이르기까지 여러 문제가 발생한다[3,7]. 따라서 실제 데이터는 통신망에서 공유할 수 있는 곳에 저장해놓고 제어 프로그램만 교환하고 또한 제어 프로그램이 필요로 하는 시점에 필요한 양만큼 전송해 올 수 있도록 실제 데이터의 교환 없이 제어 프로그램만의 교환으로 연출될 수 있는 인터넷상에서의 프리젠테이션 메일에 대한 연구가 의미 있는 일이다.

본 논문은 기존의 텍스트 위주였던 인터넷 메일을 한 단계 넘어 동영상, 음향, 그래픽 등 다양한 멀티미디어 데이터를 이용한 프리젠테이션 메일을 제안한다. 이를 개발함에 있어 가장 큰 문제점은 일반 사람들이 자신이 원하는 멀티미디어 연출을 쉽게 작성할 수 있어야 한다는 점이다. 그리고 또 다른 문제는 프리젠테이션 메일을 구성하는 요소가 일반적으로 거대한 용량을 가지는 멀티미디어 데이터라는 것이다. 거대한 용량의 멀티미디어 데이터는 전송 및 저장에 있어서 많은 문제점을 갖게 되지만 멀티미디어 메일에 있어서는 해결가능성이 있는 것이 그 데이터가 저장되어 있다는 점과 연출하려고 하는 저작자의 의도를 제어프로그램을 분석함으로 미리 알 수 있다는 점이다. 이점을 이용하여 멀티미디어 메일은 실제 데이터 부분과 제어 데이터를 분리시켜 제어 프로그램만을 전달하고 실제 데이터는 보낸 사람의 컴퓨터에 있거나 고속 접속이 가능한 원격지 서버에 저장되어 실제 연출할 때에 서버로부터 전송하는 방식을 취한다.

이러한 상황에서 제어 프로그램에 포함된 하이퍼프리젠테이션을 인터넷상에서 실시간으로 재생하기 위하여 실행 전에 제어 프로그램을 분석하여 미리 가져와야 할 데이터 량을 결정하는 기법을 제안하였다. 그리고 멀티미디어 연출을 쉽게 제작하기 위하여 본 논문은 제어 프로그램을 제작하는 도구로써 VIP(Visual Interface Player)를 이용하였으며 그것을 LAN기반에서 구현하고 실험한 결과를 제시하였다.

다음의 2장에서는 프리젠테이션 메일 개념을 소개하고 그것을 실제 개발하는데 핵심이 되는 문제점을 제시하였다. 그리고 3장에 인터넷상에서 수신자가 받은 메일을 실시간으로 재생하기 기법을 제안하였다. 그리고 실험한 결과를 4장에 보여주고 결론을 맺었다.

2. 프리젠테이션 메일 및 문제점

프리젠테이션 메일이란, 현재 널리 사용되고 있는 텍스트 위주의 메일을 넘어서 동영상, 음향, 그래픽 등의 멀티미디어 데이터들을 이용한 메일이다. 즉, 영상과 음성을 이용해 메일을 구성함으로써 텍스트만을 통한 메일 보다 더욱 감각적인 메일이 되는 것이다. 그러나 이는 방대한 양의 멀티미디어 데이터를 필요로 하게 되고 전송에 소요되는 시간과 수신측 저장용량의 한계에 부딪히게 된다[8,9]. 이를 극복하기 위하여 실제 멀티미디어 데이터와 제어 프로그램을 분리하여 메일을 전송할 때는 연출제어 프로그램만 보내소 실제 멀티미디어 데이터는 송신자 측에 남겨 두거나 고속 접속이 가능한 원격지 서버에 전송한다. 그리고 나서 수신자가 프리젠테이션메일을 재생할 시점에 인터넷을 통하여 사용자측이나 서버에 있는 멀티미디어 데이터를 가져와서 재생하자는 것이다. 이렇게 하면 수신자는 대용량의 멀티미디어 데이터를 저장할 필요 없이 연출제어 프로그램만 저장하면 된다.

이렇게 하는데는 우선 크게 두 개의 문제가 발생한다. 첫째 메일은 일반인이 사용해야 하기 때문에 전문적인 교육을 받지 않은 사람도 사용할 수 있을 정도로

쉬워야 한다는 점이다. 그리고 두 번째 문제는 매우 기술적인 문제로 수신자가 연출 프로그램을 재생할 때 연출의 QoS(Quality of Service)를 만족하도록 실시간으로 데이터를 가져와서 연출할 수 있어야 한다. 그러나 인터넷을 사용할 때 실시간으로 원격지의 멀티미디어 데이터를 가져온다는 것은 불가능함으로 수신측의 컴퓨터에서 연출에 필요한 데이터를 미리 가져오거나 실시간 연출을 위한 스케줄링을 해야한다. 본 논문은 이러한 문제를 해결하기 위한 방안을 제시하고 실제 구현한 결과를 보여 주고자 한다.

2.1 VIP를 이용한 프리젠테이션 메일의 개념

오디오나 비디오 등이 포함된 멀티미디어 데이터를 이용하여 연출하고자 하는 바를 기술하는 방법은 여러 방법이 있다. 멀티미디어 연출을 위한 시간적 동기화를 명시하는 방법으로는 연출할 미디어간의 관계(Relation)를 가지고 연출의 시간적 우선 순위를 명시하는 간격 기반 명시방법(Interval-based Specification)[11] 과 광역시간대에 대한 미디어연출 시간을 명시하는 시간축 기반 명시방법(Time Axes-based Specification)[12], 전체 프리젠테이션의 구조를 명시하는 제어흐름 기반 명시방법(Control Flow-based Specification)[13], 페트리 네트(Petri Net)를 이용하여 미디어간의 사건(event)관계를 명시하는 사건 기반 명시방법(Event-based Specification)[14], 그리고 미디어간의 동기 시나리오를 문서적으로 기술하는 스크립(Script) 방법으로 SMIL(Synchronized Multimedia Integration Language)[15] 등이 있다. 그러나 이러한 방법들은 미디어간의 시간적인 관계를 이론적으로 나타내는데는 충분하나 어느 한 동기화 명시방법이 실제 연출상황을 전부 명시할 수 없을 뿐만 아니라 어느 한 프리젠테이션을 연출하는 도중 다른 프리젠테이션을 연출하게 하는 하이퍼프리젠테이션을 명시하는 방법을 제공하지 못한다. 이러한 문제점을 해결하기 위하여 하이퍼프리젠테이션을 포함한 연출 동기화 명시기법이 참고문헌[16]에서 제안된바 있다. 그러나 이것도 API(Application Programming Interface) 형태나 스크립 언어 형태로 제공되기 때문에 일반사람들이 메일을 보내기 위하여 멀티미디어 연출을 기술하기에는 너무 어렵다. 그래서 프로그램 하지 않고도 원하는 연출을 표현할 수 있는 비주얼 아이콘 프로그램 도구가 VIP(Visual Interface Player)[17]이다.

2.2 프리젠테이션 메일의 예

예를 들면 다음과 같은 시나리오의 뉴스를 메일로 보내고자할 때 우선 VIP를 이용하여 오디오나 비디오 데이터의 연출 관계를 시나리오에 맞게 표현한 후 그 파일을 보통사용하고 있는 인터넷 메일에 첨부해서 수신자 측에 보낸다. 그러면 실제 크기가 작은 VIP프로그램만 메일에 첨부되어 전송되고 데이터 량이 방대한 실제 멀티미디어 데이터는 송신자 측에 남아 있게 됨으로 수신자 측에 부담을 주지 않고 전송이 된다. 한편 수신자는 자기가 원하는 시간에 메일을 검사하여 첨부된 프리젠테이션 메일을 보고 싶을 때 재생명령을 하면 그때 송신자 측의 컴퓨터에서 실제 멀티미디어 데이터를 가져와서 VIP에 명시된 데로 재생하고 가져온 데이터는 버리게 된다.

2.2.1 시나리오의 예

다음 시나리오는 뉴스를 제작하여 메일로 보내려는 것으로 순서는 다음과 같다.

배경음악이 흐르면서 뉴스 앵커의 시작 멘트
의원 선거 전체에 대한 앵커의 설명
    후보 A의 연설 장면 (설명도중 사용자의 선택에 의해 후보 A의 상세 내용 볼 수 있음)
    후보 B의 연설 정면 (설명도중 사용자의 선택에 의해 후보 B의 상세 내용 볼 수 있음)
의원 선거에 대한 마무리
날씨에 대한 앵커의 소개
    뉴스로 기상캐스터의 날씨전반 소개
        지역별 날씨소개 후 해상 날씨소개

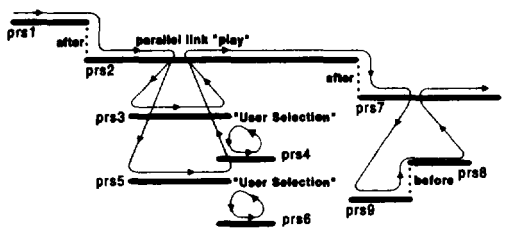
위의 시나리오를 표현하는 방법에는 두 가지가 있다. 첫 번째는 하이퍼프리젠테이션을 지원하는 MuX[18]의 API를 이용하여 구현하는 것이고 두 번째는 본 논문에서 제시하고자 하는 VIP(Visual Interface Player)를 이용하는 것이다. 먼저 API를 이용하는 경우를 먼저 살펴보자. API를 이용해 프로그래밍 하는 경우에 프로그래머는 일단 프로그래밍 언어(C/C++)와 API를 사용할 줄 알아야 한다. 또한 하이퍼프리젠테이션을 이루는 각 객체에 대해 일관성을 유지하기 위해 많은 신경을 써야 한다. 이렇게 하여 만들어진 결과를 확인하기 위해서는 컴파일의 과정을 거쳐 실행파일로 만든 후 실행해야 한다. 이같은 여러 불편한 점들을 개

선하여 하이퍼프리젠테이션을 더욱 편리하게 작성하는 방법이 바로 VIP를 이용하는 것이다. VIP에서는 각 객체를 나타내는 아이콘들이 있는데 프로그램은 단지 이들을 시나리오에 맞게 연결하고 시간 설정 등과 같은 몇 가지 추가적인 설정만 해주면 된다. 아이콘을 이용하므로 일관성 유지가 매우 쉽고 프로그래밍 언어를 모르는 사람도 구현 할 수 있는 등의 많은 장점이 있다.

2.2.2 VIP를 이용한 뉴스 시나리오 프로그램

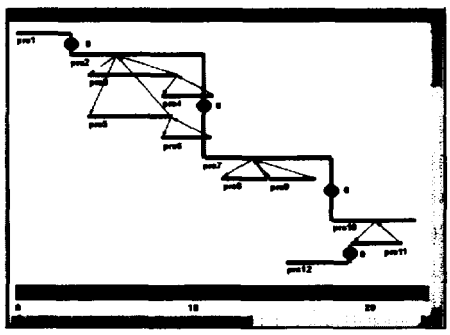
이러한 토막 토막의 기사들이 다음과 같이 파일로 저장되어 있다고 하자. 실제로 이러한 프리젠테이션을 만드는 것도 VIP를 이용하여 전혀 프로그램하지 않고 쉽게 만들 수 있다. VIP에 대한 상세한 내용은 참고문헌[17]을 참고하기 바란다.

<prs1>은 뉴스 앵커의 시작 멘트와 배경 음악으로 구성된 프리젠테이션이며, <prs2>는 의원 선거에 대한 보도이다. <prs3>과 <prs5>는 두 명의 의원후보 A와 B의 연설 장면을 담고 있다. 사용자는 동시에 화면으로 나타나는 두 연설 장면을 보며 그 중 하나를 선택하여 해당 후보의 자세한 정보를 얻을 수 있다. <prs4>와 <prs6>이 각각 후보 A와 B에 대한 자세한 소개를 담은 파일이다. 의원 선거에 대한 기사가 끝나면 계속해서 내일의 날씨가 진행된다. <prs7>은 기상 캐스터의 진행 멘트와 배경 음악, 그리고 화면으로 구성된 프리젠테이션이다. <prs8>은 각 지역별 날씨, <prs9>는 해상의 날씨로 구성되어 있으며, 이러한 프리젠테이션들을 개념적으로 표현한 뉴스진행 시나리오가 (그림 2-1)에 나와 있다.



(그림 2-1) 뉴스진행 시나리오
(Fig. 2-1) News Scenario

이것을 메일로 보내기 위하여 VIP로 나타낸다면 다음 (그림 2-2)에 있는 것과 같다.



(그림 2-2) 뉴스 시나리오의 VIP 프로그램
(Fig. 2-2) VIP Program of News Scenario

2.2.3 프리젠테이션 메일의 전송

이렇게 하여 원하는 연출프로그램이 작성되었으면 이 VIP 프로그램을 파일에 저장한다. 그리고 원하는 수신자에게 인터넷에서 사용되고 있는 메일을 이용하여 첨부 화일로 보내면 된다. 이 때 실제 전송되는 데이터는 VIP 프로그램만 전송되고 연출에 사용되는 대용량의 실제 데이터는 전송되지 않는다. 이렇게 하면 VIP프로그램의 크기가 작기 때문에 저속도의 인터넷에서도 문제가 없이 전송될 수 있게 된다.

2.2.4 수신자 측의 프리젠테이션 메일의 재생

수신자는 통상적으로 메일을 보다가 첨부된 프리젠테이션 메일이 있으면 그것을 저장하거나 "OPEN"하여 재생해 볼 수 있다. 이것을 위하여 VIP프로그램 재생기가 필요한데 처음이라면 VIP재생기를 인터넷에서 다운로드 받으면 된다. 첨부된 프리젠테이션 파일을 "OPEN"하면 (그림 2-2)와 같은 VIP프로그램을 볼 수 있고 이것을 "PLAY"하게 되면 VIP프로그램 재생기가 송신자 측의 컴퓨터에서 필요한 멀티미디어 데이터를 통신망을 통하여 가지고 와서 재생하게 된다.

2.3 프리젠테이션 메일의 재생을 위한 실시간 연출 문제

위의 예에서 보듯이 기술적으로 가장 큰 문제점은 수신자 측의 VIP프로그램 재생기가 원격지의 방대한 데이터를 가져와서 사용자가 만족하는 수준으로 재생할 수 있는가 하는 문제이다. 프리젠테이션 메일의 요구 사항인 실제 멀티미디어 데이터와 제어프로그램의 분리하기 위해서 실행 시 원격지 서버에 존재하는 실제 멀티미디어 데이터를 실시간으로 전송해와야 한다. 그러나 통신망의 상태에 따른 초기 지연, 끊김, 링크

진입 시 지연, 링크 복귀 시 지연 등이 발생하게 된다. 이를 해결하는 방법으로 통신망의 속도를 대폭 늘려서 수신자 측의 VIP 재생기가 원하는 시간만큼 실시간으로 데이터를 가져올 수 있다면 문제가 없다. 그러나 이러한 초고속 통신망 환경을 아직 요원한 실정임으로 인터넷과 같은 저속망에서는 이런 해결방법은 불가능하다. 다른 방법은 이미 연출할 VIP 프로그램을 갖고 있고 멀티미디어 데이터가 저장되어 있는 것이기 때문에 수신자 측의 컴퓨터가 가지고 있는 메모리 버퍼와 재생기의 쓰레드들을 조정함으로 초기 지연은 발생할 수 있지만 실시간으로 재생할 수 있는 방법이 존재할 수 있다. 본 논문에서는 이 문제를 해결하여 하이퍼프리젠테이션이 포함된 프리젠테이션 메일의 실시간 연출 기법을 제안하였다.

3. 인터넷상에서 프리젠테이션 메일의 실시간 재생을 위한 연출기법

인터넷상에서 분산 멀티미디어 데이터의 실시간 전송이라는 문제는 많은 문제점을 남긴다. 큰 용량의 멀티미디어 데이터와 제한된 통신망의 전송속도는 많은 논란이 되어 왔으나 통신망의 전송양은 제한되어 있으며 사용자의 QoS(Quality of Service)를 만족시키기 위한 필요한 멀티미디어 데이터 량 또한 고정적이다[1]. 이에 가장 기본적인 방법으로 버퍼를 사용한 프리페치 기법과 스케일링 기법을 들 수 있다.

프리페치 기법의 기본 원리는 실시간으로 실행되기 이전에 통신망의 상태에 따라 데이터를 미리 실행 컴퓨터로 전송 받아 흐름의 끊김을 막는 것으로 데이터를 미리 전송 받으므로 흐름의 끊김은 막을 수 있으나 초기 지연이 발생하는 단점을 갖는다. 그에 따른 연속 미디어 서버의 버퍼 공간 활용 등 프리페치에 대한 다양한 연구가 이루어져 있다[2,3,4,6].

이와 같이 멀티미디어의 실시간 연출에 대한 그 연구가 다양하게 이루어져 있으나 그 제어는 단순한 멀티미디어 스트림을 실행하는데 국한되어 있으며 엄격한 시간 제어와 하이퍼링크를 포함한 흐름제어를 요구하는 하이퍼프리젠테이션이나 프리젠테이션 등을 실행하기 위한 방식을 포함하진 않았다. 따라서 하이퍼프리젠테이션이 포함된 프리젠테이션 메일을 위한 새로운 실시간 연출 기법이 필요하다.

3.1 제안된 하이퍼링크를 이용한 실시간 연출 기법

하이퍼링크 제어를 위한 가장 기본적인 방식은 기존의 프리페치 방식이다. 프리페치는 실행 전에 데이터를 내 컴퓨터에 미리 가져 옮으로써 통신망의 상태 등에 구애받지 않고 자연스러운 연속 미디어의 실행을 목적으로 한다. 이 때 무작위 적인 데이터 프리페치가 아니라 좀더 효율적인 방식으로 프리페치하기 위해 많은 연구가 이루어져 있으며[6,10] 본 논문에서 또한 프리젠테이션 메일의 특징에 적절한 하이퍼링크를 이용한 실시간 연출 기법을 제안하였다.

기본적인 문제는 어느 한 프리젠테이션을 수행할 때 그와 하이퍼링크로 연결된 다른 프리젠테이션의 데이터를 언제 그리고 얼마나 미리 가져와야 하는가에 있다. 왜냐하면 미리 재생될 데이터가 준비되어 있지 않으면 하이퍼링크에 연결된 프리젠테이션으로 넘어가는데 시간이 많이 지연되기 때문에 재생이 부자연스럽고 또 인터넷과 같은 저속망에서 지터와 지연이 발생하여 자연스러운 재생이 불가능하기 때문이다. 그리고 미디 데이터를 가져오는 방식도 문제인 것이 멀티미디어 데이터는 매우 방대하기 때문에 데이터 전체를 미리 가져오기에는 수신자 측의 메모리 용량이 부족하기에 때문이다. 따라서 프리젠테이션 속도와 전송속도 등을 고려하여 사용자의 요구를 만족하는 재생이 되도록 버퍼의 크기와 프리페치할 데이터 량 그리고 재생하는 중에 프리페치하는 방식 등을 새롭게 제안하고자 한다.

하이퍼링크의 속성이 "ACTIVE", "INACTIVE", "USER INTERACTION" 등으로 되어 있어 "ACTIVE"한 상태이면 하이퍼링크에 연결된 프리젠테이션을 항상 수행하게 되지만 다른 경우는 조건에 맞는 "EVENT"나 사용자의 입력이 있어야만 수행된다. 따라서 정적 프리페치 방식과 동적 프리페치 방식을 혼합한 형태로써 기존의 프리페치 방식에 하이퍼링크의 특징을 이용하여 멀티미디어 스트림들의 전환 및 제어가 복잡한 프리젠테이션과 하이퍼프리젠테이션의 수행에 적절하도록 하였다.

[정의 3-1] 정적 프리페치(static prefetch)

프리페치의 시점이 프리젠테이션 메일의 재생이 실행되기 전에 이루어지는 것을 정적 프리페치 방식이라 정의한다. 따라서 이는 프리젠테이션 메일이 실행되기 전에 초기 지연을 야기하지만 링크를 따른 흐름 전환

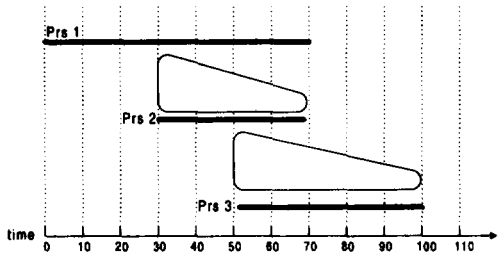
시 지연과 흐름의 끊김을 제거하여 사용자 QoS에 만족되는 자연스러운 연출을 보장한다. 여기서는 하이퍼링크의 기준이 되는 프리젠테이션이 여기에 해당되며 그 내부의 시작 시점에서 실행되는 스트림들이 포함된다. 즉, 동적인 프리페치가 불가능한 스트림들을 포함한다.

[정의 3-2] 동적 프리페치(dynamic prefetch)

프리페치의 시작 시점이 프리젠테이션 메일이 실행되는 도중에 이루어지는 것을 동적 프리페치 방식이라 정의한다. 이는 프리젠테이션과 하이퍼프리젠테이션의 복잡한 흐름 전환을 이용한 것으로 하이퍼링크에 연결된 프리젠테이션이나 프리젠테이션 내부에 다른 스트림들과의 시간적 관계에 있어 늦게 시작되는 스트림들이 여기에 해당된다. 즉, 프리젠테이션의 실행 중간부터 시작되는 스트림들이나 하이퍼링크로 연결되어 있는 스트림들이 포함된다.

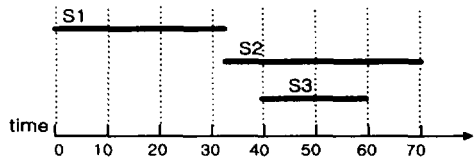
[예제 3-1] 프리젠테이션 메일

예를 들어 다음과 같은 프리젠테이션 메일을 수신자가 재생하고자 한다고 하자

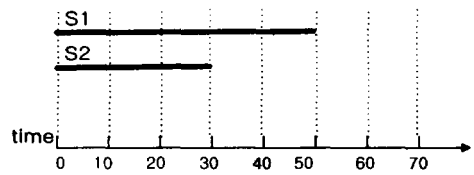


(그림 3-1) 프리젠테이션 메일 구성도
(Fig. 3-1) Configuration of a Presentation Mail

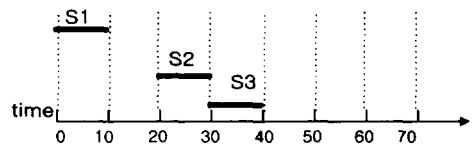
위의 (그림 3-1)과 같은 프리젠테이션 메일이 있고 메일을 구성하는 각각의 프리젠테이션이 (그림 3-2), (그림 3-3), (그림 3-4)와 같다고 하자. (그림 3-1)이 의미하는 것은 실행될 메일이 3개의 프리젠테이션 Prs1, Prs2, Prs3으로 이루어져 있으며 그 각각의 start_time과 end_time은 하단의 time 축에 나타나 있다. 이 하이퍼프리젠테이션은 Prs1이 30초 동안 실행된 후 하이퍼링크를 따라 Prs2를 수행하고 Prs2가 30초부터 50초까지 20초 동안 실행되며, 다시 하이퍼링크를 따라 Prs3이 50초부터 100초까지 50초가 수행된 후 다시 Prs2로 Prs1로 되돌아오는 구성임을 알 수 있다. 여기서 (그림 3-2)는 프리젠테이션 Prs1은 2개의 스트림으로 구성되어 있는데 스트림 S1이 0에서 30까지 실행되고 그 후에 스트림 S2가 70까지 수행되는 것을 나타내고 있다.



(그림 3-2) 프리젠테이션Prs1 내부 구성도
(Fig. 3-2) The Internal Configuration of Prs1



(그림 3-3) 프리젠테이션Prs2 내부 구성도
(Fig. 3-3) The Internal Configuration of Prs2



(그림 3-4) 프리젠테이션Prs3 내부 구성도
(Fig. 3-4) The Internal Configuration of Prs3

(그림 3-1)과 같은 메일이 재생되면 하이퍼링크에 연결된 Prs2와 Prs3은 메일이 실행되고 한참 후에 시작됨을 알 수 있다. 이때 프리젠테이션 Prs1을 구성하는 스트림 Prs1.S1이 실행되는 동안 Prs1.S2를 동적으로 전송 받을 수 있으며 Prs3.S1이 실행되는 동안 Prs3.S2를 동적으로 전송 받을 수 있음을 직관적으로 알 수 있다. 이와 같이 실행 중에 프리페치를 시작해도 가능한 스트림들에 동적 프리페치 방식을 적용한다. 이와 반대로 실행 시작 시점에 있는 Prs1을 구성하는 스트림들 중 시작 시점에 있는 S1은 메일이 실행되자마자 시작되는 스트림이다. 이 때 S1은 동적 프리페치가 불가능하므로 메일이 시작되기 전 정적 프리페치 방식을 통해 프리페치해야 함을 알 수 있다. 또한 Prs2.S1, Prs2.S2 , Prs3,S1에도 정적 프리페치를 적용

할 수 있다.

여기서 (그림 3-1)의 구성처럼 프리젠테이션들이 하이퍼링크로 연결되어 있음을 이용하여 동적 프리페치가 불가능한 Prs1에서의 Prs1.S1, Prs2에서 Prs2.S1과 Prs2.S2, 그리고 Prs3에서 Prs3.S1에 대해 그 상위 프리젠테이션이 존재한다면 그 상위 프리젠테이션을 실행하는 동안 동적 프리페치가 가능하도록 하는 것이다. 다시 말해 프리젠테이션 Prs1이 실행되는 동안 Prs2.S1과 Prs2.S2를 동적으로 프리페치하고 Prs2가 실행되는 동안 Prs3.S1을 동적으로 프리페치하려는 것이다. 결국 전체 프리젠테이션 메일이 시작되는 시점에 있는 프리젠테이션 Prs1의 시작 시점에 있는 Prs1.S1만은 동적 프리페치가 불가능함을 알 수 있다. 따라서 Prs1의 S1은 정적 프리페치 방식을 통해 실행되기 전에 전송해와야 한다. 위의 예제는 하이퍼링크를 이용한 실시간 연출 기법의 기본 원리로 동적 프리페치 방식과 정적 프리페치 방식을 하이퍼링크를 이용하여 적절히 혼합하였다.

현재 프리젠테이션 메일이 제공하는 하이퍼링크에는 그 속성에 따라 Active 하이퍼링크, Inactive 하이퍼링크, User Interaction 하이퍼링크 세 가지가 있다. Active 속성은 링크된 시점에서 무조건 분기하는 것으로 하나의 프리젠테이션이 실행되는 도중 링크를 만나면 실행 중인 프리젠테이션을 멈추고 링크를 따라 흐름이 바뀌었다가 연결된 프리젠테이션이 끝나면 스스로 상위 프리젠테이션으로 되돌아온다. Inactive는 Active와 반대로 무조건 무시되는 링크이다. 즉, 실행하지 않는다. 마지막으로 User Interaction 하이퍼링크는 웹 브라우저의 하이퍼텍스트처럼 사용자의 인터랙션이 주어지면 링크를 따라 제어가 넘어갔다가 되돌아오는 형태이다. 여기서 Inactive링크인 경우는 실행되지 않음으로 프리페치하지 않으며 Active링크인 경우는 다음에 제시된 Dynamic-Static 혼합 기법에 의해 프리페치된다.

어느 한 프리젠테이션을 재생하기 위하여 프리페치해야할 데이터는 통신망의 전송속도와 재생에 소요되는 데이터 소비율 그리고 프리젠테이션 전체 재생시간과 관계로 다음과 같이 구할 수 있다.

[방식 3-1] 프리페치할 데이터 량의 결정 방식

어느 한 프리젠테이션의 프리페치할 데이터 량은 다음과 같이 결정된다.

$$prefetch\_data(a,b) = n\,(a - b) \qquad \text{식 (1)}$$

a : 초당 재생되는 데이터 량*(bytes per sec)*

b : 초당 통신망을 통해 전송되는 데이터 량*(bytes per sec)*

n : 프리젠테이션에서 지정해준 실행 시간*(sec)*

여기서 x 만큼 프리페치되어 있을 때 시간별 보유 데이터 량 변화는 다음 표와 같다.

〈표 3-1〉 보유 데이터 량 변화표
〈Table 3-1〉 The Change of Prefetched Data

시간(sec)	1	2	3	. . .	N
데이터 량 (byte)	$x-a+b$	$x-2a+2b$	$x-3a+3b$	. . .	$x-na+nb$

n초 후에는 실행이 종료되고 보유한 데이터 량이 0이 된다. 따라서 조건 $x - na + nb = 0$을 만족하는 프리페치할 데이터 량 x의 값은 $n\,(a - b)$이 된다.

프리젠테이션 메일의 VIP 프로그램을 분석하면 prefetch(a, b) 계산에 필요한 값 n, a, b를 미리 계산할 수 있기 때문에 이 방식은 프리젠테이션이 수행되기 전에 미리 가져와야 할 데이터의 량을 계산하는 방식으로 사용될 수 있다. 그러나 이 방식의 문제점은 통신망의 전송속도가 시간에 따라 변화한다는 점을 간과한 것이다. 따라서 일정한 시간 간격으로 통신망의 전송속도와 소요량을 다시 계산하여 프리페치할 데이터를 조정하는 통신망 적응형 프리페치 데이터 량을 결정하는 방식도 필요하다.

[방식 3-2] 적응형 프리페치 데이터 량 결정 방식

프리젠테이션 p를 재생하기 위하여 프리페치할 데이터 량은 다음과 같은 식에 의해 결정된다.

$$prefetch\_data(p) = \sum_{i=1,pt} (t_i - t_{i-1})(transfer\_data(t_i) - play\_data(t_i)) \qquad \text{식 (2)}$$

t_i : 다시 계산하기 위한 시간 간격으로 t_0 는 시작시간이고 pt 는 프리젠테이션 p가 끝나는 시점임

play_data(t) : t 시점에 초당 재생에 소요되는 데이터 량

transfer_data(t) : t 시점에 초당 통신망을 통해 미리 전송 받아야 하는 데이터 량

presentation_time(p) : 프리젠테이션 p가 연출 프로그램에 지정된 수행 시간*(sec)*

이 방식은 프리젠테이션이 시작하기 전에는 활용하기 곤란하고 실제 연출이 시작해서 데이터가 소비되면

서 동시에 새롭게 원격지에서 데이터를 가져와야 할 경우 통신망의 상태를 반영하는 수단으로 활용될 수 있다. 일반적으로 prefetch_data(p)를 계산하는데 필요한 *play_data*(t_i)는 크게 변화가 없지만 *transfer_data*(t_i)는 통신망의 부하정도에 따라 심각한 영향을 받는다. 이것을 계산하는 방식은 여러 방법이 있을 수 있으나 실제 상황에 맞게 하기 위하여 ping과 같은 방법을 이용하여 전송속도를 측정하는 방법이 가장 의미 있을 것이다.

[방식 3-3] 프리페치를 시작할 시점(*t*)결정

프리젠테이션을 시작하기 전 미리 데이터를 가져와야 할 시점 t는 다음과 같이 결정된다

$$t = start\_time - x / b \qquad \text{식 (3)}$$

start_time : *stream*의 실행 시작 시간*(sec)*

x : 프리페치할 데이터 량*(byte)*

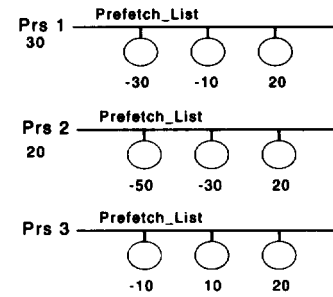
b : 초당 통신망을 통해 전송되는 데이터 량*(bytes per sec)*

프리젠테이션의 실행 시작 시간과 실행 시간에 따라 t값이 변화되며 그 값이 0보다 작으면 정적 방식에 의해 실행되기 전에 프리페치가 이루어지며, 그 값이 0보다 클 때 동적 방식으로 실행 중에 프리페치가 이루어진다.

[예제 3-2] 프리페치 데이터

예제 [3-1]에 있는 (그림 3-1)의 프리젠테이션 메일의 초당 재생되는 데이터 량 a는 14,000*byte*이며, 56*Kbps*의 모뎀인 경우 초당 전송되는 데이터 량 b는 7,000 *byte*라 하고 Prs1이 0초부터 30초까지 수행되는 스트림1과 10초부터 30초까지 수행되는 스트림2, 40초에서 60초까지 수행되는 스트림3을 갖는다고 가정하면, 기본 원리 식 (1)과 식 (3)을 이용하여 다음의 프리페치 시점(t) 값을 결정할 수 있다.

스트림1의 프리페치 데이터 량 (x)은 식 (1) x = n (a - b)에 의해 30 * (14,000 - 7,000) = 210,000 byte 이며, 그 프리페치 시점(t)은 식 (3) t = start_time - x / b에 의해 0 - (210,000 / 7,000) = -30이 된다. 이와 같은 방식으로 (그림 3-1, 3-2, 3-3)의 프리젠테이션을 이루는 각 스트림들의 프리페치 시점을 구하여 프리페치 리스트를 구성하면 다음과 같은 구성을 이룬다.

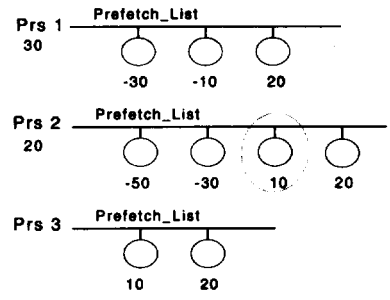


(그림 3-5) 프리페치 리스트 구성 1
(Fig. 3-5) The Configuration 1 of Prefetch List

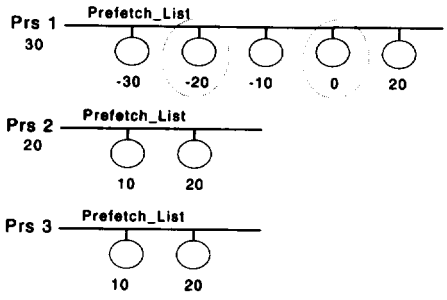
이 때 프리페치 리스트에 프리페치 시작 시점(t)에 대한 정보를 볼 수 있으며 그 값이 0보다 작을 때는 프리젠테이션이 실행된 이후에 프리페치할 수 없음을 알 수 있다. 따라서 자신의 상위 프리젠테이션에서 프리페치해 주어야 한다. 위의 그림에서 보면 Prs3에 첫 번째 리스트인 10을 자신의 상위 프리젠테이션인 Prs2로 옮겨주어야 하며 Prs2의 프리페치 리스트로 옮겨질 때는 프리페치 시작 시점 값을 다음과 같이 재 계산한다.

링크될 시점(50) - 상위 프리젠테이션의 시작 시점(30) + 프리페치 시점(-10)

따라서 50 - 30 + (-10)으로 Prs2의 프리페치 리스트에 프리페치 시점 10을 추가하면 (그림 3-6)과 같은 구조가 되며, Prs2의 50과 30을 다시 계산하면30 - 0 + (-50) = 20과 30 - 0 + (-30) = 0이 된다. 이들을 Prs1의 프리페치 리스트로 이동시키면 (그림 3-7)과 같은 동적 리스트를 구성하게 된다.

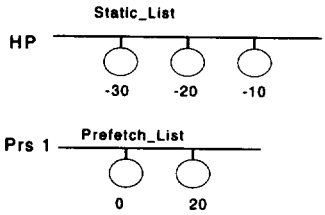


(그림 3-6) 프리페치 리스트 구성 2
(Fig. 3-6) The Configuration 2 of Prefetch List



(그림 3-7) 동적 리스트 구성 3
(Fig. 3-7) The Configuration 3 of Dynamic List

이제 더 이상 상위 프리젠테이션이 존재하지 않는다. 이때 최상위 프리젠테이션 Prs1의 마이너스 값을 갖는 리스트는 실행 시 동적 프리페치가 불가능하므로 이에 대해서는 정적 프리페치 방식을 적용해야 한다. 따라서 다음과 같이 Static_List에 실행 전에 프리페치할 리스트를 작성한다.



(그림 3-8) 정적 리스트 구성
(Fig. 3-8) The Configuration of Static List

위의 예제와 같이 하나의 하이퍼프리젠테이션은 프리젠테이션들로 구성되어 있으며 위와 같은 방법에 의하여 정적인 프리페치 리스트인 static list 와 실행 중 프리페치해야하는 dynamic list를 구할 수 있다. 따라서 실행 전에 static_list에 있는 스트림의 데이터를 미리 가져와야 하기 때문에 그만큼 초기 지연을 갖게 된다. 위의 방법을 일반화하면 다음의 알고리즘으로 나타낼 수 있다.

[절차 3-1] 하이퍼링크로 연결된 상위 프리젠테이션과 결합하는 절차

입력으로 (그림 3-5)와 같이 각 프리젠테이션의 프리페치 리스트는 정해졌다고 가정하고 프리젠테이션들 간에 설정된 하이퍼링크를 검색하여 자신이 실행 중 프리페치할 수 없는 데이터 즉, prefetch_time(t)가 0보다 작은 prefetch_object의 prefetch_time(t)을 재계산하여 자신의 상위 프리젠테이션으로 이동시키는 알고리즘으로 동적 프리페치 방식을 지원하기 위함이다

Step 0 : hyper link list 의 다음 hyper_link를 가져온다. 존재하지 않으면 종료
Step 1 : hyper_link의 sub_prs의 prefetch_list에서 다음 prefetch object를 가져온다. prefetch object가 존재하지 않으면 Step 8 수행.
Step 2 : prefetch object의 prefetch time이 0보다 작으면 계속. 0보다 크면 Step 7 수행.
Step 4 : prefetch object의 prefetch time은 prefetch object의 prefetch time + base prs의 link time - base prs의 start time으로 재 계산한다.
Step 5 : base prs의 prefetch list에 prefetch object를 prefetch time이 작은 순서대로 삽입한다.
Step 6 : sub prs의 prefetch list에서 prefetch object를 삭제한다.
Step 7 : Step 1 수행.
Step 8 : base_prs의 상위 프리젠테이션이 존재하는지 찾는다. 즉, hyper_link의 sub_prs에 base_prs와 같은 hyper_link의 base_prs를 가져온다.
Step 9 : 상위 프리젠테이션이 존재하면 다시 Step 1 수행.
Step 10 : Step 0으로 돌아간다.
Step 11 : 종료

위의 알고리즘을 통해 최상위 프리젠테이션에만 0보다 작은 prefetch_time(t)를 갖는 prefetch_object가 존재하게 되며 이들에 대해서는 동적 프리페치가 불가능하다. 따라서 정적 프리페치 방식을 적용하도록 static_list로 그들을 이동시킨다

[절차 3-5] 정적 방식을 위한 static list를 찾는 절차

Step 0 : presentation list에서 presentation 객체를 가져온다. 없으면 종료
Step 1 : presentation object의 prefetch list에서 prefetch object를 가져온다. 없으면 step 0 수행
Step 2 : prefetch object의 prefetch time이 0보다 작으면 계속, 0보다 크면 Step 5 수행.
Step 3 : static list에 prefetch time이 작은 순서로 prefetch object를 삽입한다.
Step 4 : prefetch list에서 prefetch object를 삭제한다.
Step 5 : Step 0으로 돌아간다.
Step 6 : 종료

3.2 User Interaction 하이퍼링크

그러면 User Interaction 링크에 대해 좀더 생각해 보기로 하자. 사용자가 클릭했을 때 흐름이 넘어가는 User Interaction링크에서는 사용자가 클릭할 수도 클

릭하지 않을 수도 있음을 알 수 있다. 이 때 사용자가 클릭하지 않는 하이퍼링크에 연결된 모든 프리젠테이션들을 프리페치한다는 것은 효율적이지 못한 방식이다. 따라서 실제 웹 상에서도 모든 하이퍼텍스트를 모두 클릭해 보지는 않는 것과 같이 사용자가 모든 하이퍼링크를 클릭하지만은 않는다는 사실에 중점을 두고, 한 프리젠테이션에 연결된 하이퍼링크의 클릭 여부를 예측하여 클릭될 확률이 높은 링크에 대한 데이터만을 프리페치할 수 있도록 기술적인 요소를 추가할 필요가 있다.

1) 제어 프로그램에서 사용자가 클릭하는 횟수를 카운트하여 링크 프리페치 여부를 결정한다.
2) 링크 프리페치의 초기 카운트 값은 메일 제작자가 부여할 수 있다.

각 프리젠테이션은 카운트를 가지고 있으며 메일 제작자가 프리젠테이션의 데이터 특성이나 제작 의도에 맞게 초기 카운트 값을 부여한다. 이는 메일 제작자의 의도를 실시간 연출 기법에 반영하기 위함이다. 그리고 프리젠테이션 메일이 광고 메일에 응용된다고 할 때 메일을 받은 여러 가입자들이 광고를 보면서 자주 클릭하는 User Interaction 하이퍼링크에 대한 프리젠테이션은 반드시 프리페치하고 거의 클릭하지 않는 링크에 대해서는 프리페치하지 않음으로써 사용자의 의도를 반영함과 동시에 불필요한 프리페치의 오버헤드를 줄이는 것이다. 여기서 먼저 비용효과함수 (cost effective function)를 정의하자

[정의 3-3] 비용효과 함수

$$cef(w, t) = w * t \qquad \text{식 (4)}$$

w : *weight*
t : *prefetch delay time*

이 식에서 중요한 것은 w 와 t 이다. 이 변수도 또한 변화할 수 있는 것이기 때문에 다음과 같이 사용자의 빈도수와 기준을 설정하여 프리페치할 것인지 아닌지를 결정하였다. 사용횟수에 따른 weight조정하는 방식은 다음과 같다.

```
If (max_user_rate >= pivot_rate )
  then weight = user_rate
  else weight = init_rate
```

user_rate : 전체 메일 가입자 수에 대한 클릭수의 백분율(%)
max_user_rate : 현재 시점에서 최대 백분율(%)
init_rate : 초기 제작자가 부여한 사용 율 (%)
pivot_rate : 기준이 되는 값으로 여기서는 50%를 사용한다.

그리고 다음과 같이 threshold를 설정하여 다음과 같이 프리페치여부를 결정한다.

```
if(cef(w, t) >= threshold)
  then prefetch
  else not prefetch
```

threshold : *cef(pivot_rate, pivot_delay_time)*
pivot_delay_time : 사용자가 참을 수 있는 정도의 delay로 여기서는 10초를 사용한다.

하이퍼링크를 검색하여 User Interaction 링크인 경우는 위의 식 (4)의 cost effective function에 의해 그 링크에 대한 프리페치 여부를 결정한다. 이는 링크된 프리젠테이션의 prefetch delay time에 사용자의 User Interaction 링크의 클릭 확률을 가중치로 준다. 그 가중치 값은 전체 가입자에 대한 클릭한 횟수의 백분율의 최대 값(max_user_rate)이 pivot_rate를 넘게 되면 가입자의 반 이상이 메일을 확인한 결과이다. 따라서 user_rate의 값이 정확성을 갖게 되므로 max_user_rate가 pivot_rate를 넘게 되면 user_rate의 값이 weight가 되고 그렇지 않은 경우에 대해서는 아직 가입자들의 대다수가 메일을 확인하지 않았으므로 user_rate의 정확성이 떨어진다. 따라서 이 때에는 제작자의 의도에 따라 init_rate의 값이 weight가 된다. 여기서 pivot_rate를 50%로 잡았으나 통신망의 상태가 나쁜 경우엔 그 기준을 더 높임으로서 프리페치에 대한 오버헤드를 줄일 수 있다. 이 방식을 알고리즘으로 나타내면 다음과 같다.

[절차 3-3] User Interaction 링크의 프리페치 결정 절차

Step 0	: hyper link list에서 다음 hyper link를 가져온다.hyper link가 존재하면 계속, 존재하지 않으면 Step 6 수행
Step 1	: hyper_link의 속성이 User Interaction이면 계속, 다른 속성이면 Step 5 수행
Step 2	: hyper link의 user_max_rate가 pivotrate 보다 크면 weight = user_rate, pivot rate 보다 작으면 weight = init rate

Step 3 : threshold는 cef(pivot_rate, pivot delay time)에 의해 계산한다.
Step 4 : cef(weight, prs_delay_time)이 threshold 값보다 작으면 prefetch list에서 삭제한다. 즉, prefetch하지 않는다.
Step 5 : Step 0으로 돌아간다.
Step 6 : 종료

위의 결정 알고리즘에 의해 프리페치하지 않을 링크에 대한 프리젠테이션을 프리페치 리스트에서 제거한다. 이는 저속의 통신망에서 모든 프리젠테이션의 프리페치로 인한 무한정한 초기 지연을 줄이고자 하는 목적으로 사용자가 클릭할 확률이 극히 낮은 프리젠테이션을 포기함으로써 통신망의 이득을 얻고자 함이다.

하이퍼링크에 연결된 프리젠테이션들에 대한 프리페치 시점을 최대한 동적 프리페치 방식으로 해결하고 실시간으로 해결이 불가능한 데이터에 대해 정적 프리페치 방식을 사용하였다. 이와 같이 동적 프리페치 방식과 정적 프리페치 방식을 혼합하여 데이터 흐름 제어가 복잡한 프리젠테이션 및 하이퍼프리젠테이션을 위한 하이퍼링크 제어 알고리즘을 구현하였다.

4. 실험 결과

4.1 구현 환경

본 논문의 구현을 위해 다음과 같은 환경에서 실험하였다.

1) Hardware : CombiStation(Pentium 100Mhz, A/V Board)
2) OS : Windows NT 4.0 Server
3) Language : Visual C++ 2.0, Visual C++ 5.0

4.2 구현 결과

본 연구의 기본이 된 기존의 MuX와 VIP는 독립형 서버에 초점을 두어 구성되었으므로 실제 데이터가 원격지 서버에 독립적으로 존재할 때 원격지 데이터를 제어할 수 없었다. 따라서 멀티미디어 프리젠테이션 메일을 위한 원격지 데이터 제어 기능을 추가하였으며, 그 성능을 향상시키기 위한 하이퍼링크를 이용한 실시간 연출 기법을 제안하였다. 이 방식을 통해 원격지 데이터를 통신망의 낮은 전송률에 의한 데이터의 끊김을 해결하였고 링크 진입, 복귀 시 지연을 해결하였다.

4.3 실험 데이터

다음 실험을 위해 버퍼에 데이터를 쓰는 WriteFrame thread에서 각 LAN, ISDN, PSTN 환경을 맞추기 위해 그 쓰는 속도에 delay를 주었다. WriteFrame은 한 번에 1102 byte씩 버퍼에 데이터를 쓴다. 이때 1102 byte를 네트웍 전송속도로 나눈 값만큼의 인위적 delay를 주어 전송 속도를 제한하였다.

다음 <표 4-1>의 결과는 프리젠테이션 메일 (그림 3-1)의 구성을 갖는 프리젠테이션을 사용하였다. 이들의 결과 값은 그들 프리젠테이션의 구성에 따라 변화되며 통신망의 전송속도가 떨어짐에 따라 그 초기 지연 시간이 늘어남을 알 수 있다. 다음은 기존의 하이퍼프리젠테이션이 실행 시 직접 네트웍을 통해 접속해서 데이터를 전송 받는 방식에서 발생했던 문제점인 끊김과 링크 진입 시 지연, 링크 복귀 시 지연을 해결하였다.

〈표 4-1〉 실험 결과 표
〈Table 4-1〉 Experimental Results

구분 Network	사용율 (%)	전송속도 (Kbps)	예측 delay (sec)	초기 delay (sec)	Prefetch 량(KB)
LAN (10Mbps)	100	10240	0.1	0.1	128
	80	8192	0.1	0.1	102.4
	50	5120	0.1	0.1	64
ISDN (64Kbps)	30	3072	0.1	0.1	38.4
	100	64	9.2	52.2	73.6
	80	51.2	27.67	72.6	177
	50	32	85.89	134.3	343.56
PSTN (28.8Kbps)	30	19.2	214.85	243.1	515.64
	100	28.8	105.79	152.2	380.8
	80	23.04	159.6	197.7	459.6
	50	14.4	320.8	333.8	577.38
	30	8.64	607.45	575.9	655.97

5. 결 론

지금까지 프리젠테이션 메일은 개인간의 통신 수단으로는 그리 효율적이지 못하다. 간단한 의사를 전달하기 위해 사용자들은 보다 다루기 쉬운 텍스트를 택할 것이다. 그러나 앞서 제시한 광고 프리젠테이션 메일과 같은 전문적 분야나 영상이나 음성을 전달하는데는 매우 효과적이다. 앞으로 인터넷 사용자는 증가추세에 있으며 전송 속도 또한 크게 향상 될 것이다. 따라서 프리젠테이션 메일을 통하여 자신의 의견을 확실하게 전달하 수 있는 수단으로 곧 등장할 것으로 예상된다.

본 논문에서 제안한 방법을 사용할 때 일반 사람들이 멀티미디어 메일을 사용할 수 있는 수준까지 개발하는데는 기술적으로 그렇게 어렵지 않다는 것을 보여주었다. 앞으로 저속통신망의 전송능력을 고려한 쓰레드 스케줄링 기법이 보완된다면 곧 실용화가 가능하리라고 본다.

참 고 문 헌

[1] Andrew S. Tanenbaum, Computer Networks, 3rd Edition Prentice Hall, 1996.

[2] A. Dan and D. Sitarma, "Buffer Management Policy for an On-Demand Video Server," IBM Research Report, RC 19347, Yorktown Heights, NY. 1993.

[3] B. Ozden, R. Rastogi, and A. Siberschatz, "Demand Paging for Video-on-Demand Servers," in Proc. of IEEE International Conference on Multimedia Computing and Systems, pp.264-272, May 1995.

[4] Cosmos Nicolau, "An Architecture for Real-Time Multimedia Communication Systems," IEEE J. Select. Area Communication, Vol.8, No.3, pp.391-400, April 1990.

[5] Changdong Liu and Myung J. Lee, "Multipoint Multimedia Teleconference System with Adaptive Synchronization," IEEE J. Select. Area Communication, Vol.14, No.7, pp.1422-1435, September 1996.

[6] B. Ozden and R. S. Yu, "Consumption-Based Buffer Management for Maximizing System Throughput of a Continuous Media Data," in Proc. of IEEE International Conference on Multimedia Computing and Systems, Jun 1996.

[7] David P. Anderson and George Homsy "A Continuous Media I/O Server and Its Synchronization Mechanism," IEEE Computer, Special Issue on Multimedia Information System, pp.51-57, October 1991.

[8] Francois Fluckiger, Understanding Networked Multimedia Application and Technology, Prentice Hall, 1995.

[9] Ralf Steinmetz and Klara Nahrstedt, Multimedia : Computing, Communications and Applications, Prentice Hall, 1995.

[10] Y. S. Ryu and K. Koh, "A Dynamic Buffer Management Technique for a Video-on-Demand Server," in Proc of IPSJ International Symposium on Multimedia Systems, pp.216-223, Feb. 1996.

[11] T. Wahl and K. Rothecprmel, "Representing Time in Multimedia Systems," Proceedings of International Conference on Multimedia Computing and Systems, May 1994.

[12] International Standards Organization, Hypermedia/Time-based Document Structuring Language(HyTime), ISO/IEC, 1992

[13] M. Salmony and D. Shepherd, Extending OSI to Support Synchronization Required by Multimedia Applications, Computer Communications, 13:399-406, September 1990.

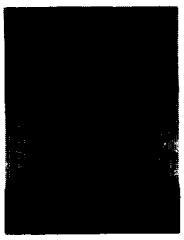
[14] W. Appelt, HyperODA, ISO/IEC/JTC1/SC18/WG3, 1989.

[15] SMIL 참고문헌 http://whatis.com/smil.htm

[16] 임영환, 김두현, 궁상환 "분산 컴퓨팅 환경에서 하이퍼 프리젠테이션을 위한 통합동기화 기법", 한국정보처리학회 논문지 제5권 제6호, 1998. 6.

[17] 임영환, 임명수, 이선혜, 우시연, "하이퍼 프리젠테이션을 위한 아이콘프로그래밍 도구", 한국정보처리학회 추계학술발표 논문집, 제4권 제2호, 1997.

[18] 김두현, 임영환, 분산멀티미디어 시스템을 위한 멀티미디어 처리 모델의 객체지향, 클라이언트-서버 구조, 한국정보처리학회논문지, 제3권 제1호, pp.9-32, 1996. 1.



임 영 환

e-mail : yhlim@computing.soongsil.ac.kr

1977년 경북대학교 수학과 졸업(학사)

1979년 한국과학원 전산학과 졸업(석사)

1985년 Northwestern University 전산학과(박사)

1979년~1996년 한국전자통신연구소 책임연구원

1996년~현재 숭실대학교 컴퓨터학부 부교수

관심분야 : 멀티미디어

이 선 혜

1997년 숭실대학교 전자계산학과졸업(학사)
1999년 숭실대학교 대학원 전자계산학과 졸업(석사)
관심분야 : 멀티미디어

임 명 수

1997년 숭실대학교 전자계산학과졸업(학사)
1997년~현재 숭실대학교 대학원 전자계산학과 재학중
관심분야 : 멀티미디어