제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호 (2003.5)

클러스터 VOD 시스템에서의 내장형 클라이언트 플랫폼 설계 및 구현

서동만*, 방철석*, 이좌형*, 김병길*, 박총명*, 정인범*
*강원대학교 컴퓨터정보통신공학과
e-mail : dmseo@snslab.kangwon.ac.kr

Design and Implementation of an Embedded Client platform in clustered VOD system.

Dongmahn Seo*, Cheolseok Bang*, Joahyoung Lee*,
Byounggil Kim*, Chongmyung Park*, Inbum Jung*
*Dept of Computer Information & Telecommunication Engineering
, Kangwon National University

요 약

본 연구에서는 클러스터 VOD 서버에서의 내장형 클라이언트 플랫폼을 설계 및 구현하기 위한 연구를 수행하였다. 클러스터 VOD 서버의 구성과 기능을 살펴보고, 리눅스 환경에서 서버의 기능을 지원하면서 리모콘을 통한 제어와 TV를 통한 출력이 제공되도록 클라이언트를 설계 및 구현하였다. 구현된 시스템에 대한 성능 평가를 통하여 문제점들을 분석하고 대책을 제안한다.

1. 서 론

VOD 서버에서는 보다 많은 사용자에게 비디오 데이터를 안정적으로 제공하고자 한다. 이러한 목적으로 비디오 데이터의 연속적인 전송 특징을 고려할 뿐만 아니라 사용자들의 서로 다른 요구들을 만족시키기 위하여 클러스터링 비디오 서버에 대한 많은 연구들이 있었다[1, 2, 3, 4, 5]. 이러한 VOD 서비스와 관련된 연구들 중 서버에 관한 연구는 많이 진행되어 왔지만 클라이언트에 관한 연구가 적어서 실제로 서비스하는데 문제점이 많았다.

본 연구에서는 클러스터 VOD 서버인 VODCA 서버[6]에 사용할 수 있는 내장형 클라이언트 플랫폼을 설계 및 구축하였다. 내장형으로 설계된 클라이언트 시스템은 VODCA 서버에서 제공하는 모든 기능을 지원하도록 설계되었다. 사용자는 리모콘(Remote controller)을 통하여 모든 동작을 조작할 수 있으며, 일반 TV를 통해 영화를 시청할 수 있도록 내장형 클라이언트 시스템을 설계 및 구현하였다.

* 이 논문은 2003년도 강원대학교 두뇌한국21사업에 의하여 지원되었음.

본 논문의 구성은 다음과 같다. 2장에서는 클러스터 VOD 서버인 VODCA 서버의 구조와 특징에 대하여 설명한다. 3장에서는 내장형 클라이언트의 구조와 특징에 대하여 설명하고, 4장에서는 VODCA 시스템의 구현 환경과 구현한 시스템의 동작 및 성능을 살펴본다. 5장에서는 본 논문의 결론을 맺고 향후 연구 과제를 기술한다.

2. 클러스터 VOD 서버의 구조와 특징

VODCA는 Video-On-Demand on Clustering Architecture의 약자로 클러스터 VOD 서버이다[6]. VODCA는 MPEG 1과 MPEG 2를 지원하며 일반 재생은 물론 고속 재생, 역재생, 일시 정지와 같은 VCR의 기능을 제공한다. VODCA는 그림 1에서 보는 바와 같이 VODCA 클라이언트와 HS(Head-End Server), MMS(Media Management Server)로 구성되어 진다.

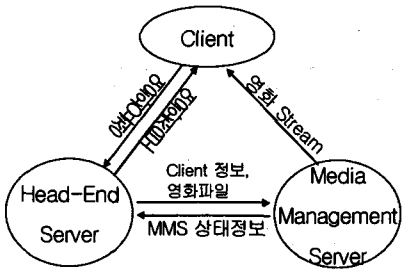

그림 1 VODCA의 구성

2.1 HS(Head-End Server)

HS는 사용자의 요청을 처리하고, MMS 노드들을 관리, 제어하는 역할을 수행한다. 또한 새로운 영화를 등록하기 위해 비디오 데이터를 분석하여 각각의 MMS 노드에 분산 저장하는 역할을 수행한다. HS는 MPEG 분석 및 스트라이핑 모듈, 자원 감시 및 관리 모듈, 연결 요청 관리 및 서비스 제어 모듈로 구성된다.

MPEG 분석 및 스트라이프 모듈은 MPEG 1과 2를 분석하고 비디오 데이터를 스트라이핑 하여 각 MMS 노드에 전송하는 역할을 수행한다. 스트라이핑의 단위는 GOP단위로 이루어진다. VCR 기능을 지원하기 위하여 기존의 영화 데이터와는 별도로 I 프레임만을 추출하여 저장하는 기능을 기지고 있다. 각 스트라이핑 단위마다 시퀀스 번호와 데이터의 크기, 종류를 나타내는 헤더를 포함하여 MMS 노드에 전달한다.

자원 감시 및 관리 모듈은 HS와 MMS 노드의 자원 상태를 전송 받아 시스템 관리자에게 통합된 화면으로 보여주는 기능을 가진다. 또한, MMS 노드의 추가, 수정 및 삭제를 가능하게 한다.

연결 요청 관리 및 서비스 제어 모듈은 클라이언트로부터 연결 요청이 들어오면 연결을 제어하고 클라이언트의 요청에 따라 서비스를 제공해 주는 역할을 수행한다.

2.2 MMS(Media Management Server)

MMS는 HS의 제어에 따라 사용자에게 비디오 데이터를 전송하여 주는 역할을 수행한다. 2초 간격으로 노드의 상태 정보를 HS에 전송하고, HS로부터 비디오 제어 명령어들을 수신하여 수행한다. MMS는 영화 관리자 모듈, MMS 자원 감시 모듈 및 영화 서비스 모듈로 구성된다.

영화 관리자 모듈은 스트라이핑 된 비디오 데이터를 HS로부터 전송 받아 자신의 디스크에 분산 저장하는 기능과 영화의 추가, 삭제 및 수정을 담당한다.

MMS 자원 감시 모듈은 자신의 자원 상태를 /proc에서 수집하여 2초 간격으로 Heartbeat을 통해 HS에 보고하는 기능을 가진다. Heartbeat는 32Bytes의 크기로 128bps 정도의 네트워크 대역폭을 사용하지만 시스템의 성능에 큰 영향을 미치지 않는다.

영화 서비스 모듈은 HS로부터 받은 명령 코드에 의해 클라이언트에게 영화를 전송하는 역할을 수행한다. 일반 재생일 경우에는 GOP 단위의 비디오 데이터를 전송하고, 고속 재생이나 역재생일 경우에는 I 프레임 단위의 비디오 데이터를 전송하며, 일시정지의 경우 영화 전송을 중지한다. 두 개의 쓰레드가 생성되어 하나의 쓰레드는 디스크에서 데이터를 읽는 역할을 수행하고 다른 하나의 쓰레드는 데이터를 전송하는 역할을 수행하며 각 쓰레드는 하나의 버퍼를 가지고 있다.

3. 내장형 클라이언트

내장형 클라이언트는 사용자 인터페이스 부분과 네트워크 수신부, 재조합 부분, 영화 재생기로 구성되어 있다.

3.1 사용자 인터페이스 부분

HS에 연결 요청을 하고 연결을 유지한다. 사용자의 요구에 따라 HS에 명령 코드를 전송하고, HS로부터 전송 받은 각종 영화 정보를 사용자에게 보여주는 역할을 수행한다. 사용자의 요청은 RS-232C 시리얼 통신을 이용하여 적외선 방식의 리모콘을 통해 입력받도록 하였다.

3.2 네트워크 수신부

MMS 노드들로부터 전송되어진 비디오 데이터를 수신하는 역할을 수행한다. 하나의 쓰레드가 다수의 MMS 노드들과 연결하여 데이터를 전송 받아 공유 메모리 영역에 저장한다.

3.3 재조합 부분

공유 메모리 영역에 저장되어 있는 비디오 데이

터의 헤더를 검사하여 순서에 맞는 시퀀스 번호를 가진 데이터를 검색하여 파이프 메커니즘을 사용해 영화 재생기에게 전송하는 역할을 수행한다.

3.4 영화 재생기

재조합 부분과 연결된 파이프로부터 비디오 데이터를 받아 디코딩 작업을 수행한다. 디코딩 된 영화 데이터는 TV 화면을 통해 사용자에게 보여준다. 고속 상영 및 역재생 동작 시에는 인코딩 과정에서 사운드 데이터를 출력하지 않도록 하여 영상 데이터만 사용자에게 출력되도록 한다.

4. 구현 방법, 시스템 동작 및 성능

4.1 구현 방법

내장형 클라이언트의 사양은 표 1과 같다. 클라이언트는 디지털 아날로그 컨버터를 이용하여 일반 가정용 TV와 연결하여 서비스를 제공하도록 하였다. 내장형 클라이언트의 구현 과정은 크게 커널과 파일 시스템 구축, 사용자 인터페이스를 위한 환경 구축, 영화 재생기로 나누어 볼 수 있다.

Board	IB790
CPU	Intel Pentium III 800Mhz
메모리	64MB SDRAM
디스크	128MB Flash Disk
운영체제	Linux Kernel 2.4.18
입력장치	RS-232C 시리얼 적외선 수신 장치, 적외선 리모트 컨트롤러
GUI	Embedded Qt 3.0.6
Movie Player	mplayer 0.18
개발 환경	RedHat 7.3, P4 1.6GHz, 256MB

표 1 클라이언트의 사양

4.1.1 커널 및 파일 시스템

RedHat 7.3이 설치된 일반 PC를 개발 환경으로 사용하였다. 클라이언트 보드에 리눅스를 이식하기 위해 PC 디스크에 별도의 파티션을 나누고 그곳에 Filesystem Hierarchy Standard에 맞게 디렉토리 구조를 생성하였다. 이 디렉토리에 구현시 사용되는 패키지 및 커널을 컴파일 하여 독립적으로 부팅 가능한 상태로 만들었다[7]. 커널 컴파일 단계에서 프레임 버퍼를 활성화하도록 설정하였다. 그 후에 타깃 디스크를 ext3 타입으로 포맷한 후 부팅에 필요한 최소한의 파일들을 선별하여 별도의 파티션에서 타깃 디스크로 복사한다[8]. 그 상태에서 리눅스 설정에 필요한 파일 및 최소한의 바이너리 파일, 라이브러리 파일들을 복사하였다. 필수 유틸리티 외에 개발과 디버깅을 위해 ftp, sshd와 같은 별도의 유틸리티를 추가로 복사하였다.

4.1.2 사용자 인터페이스

클라이언트에서는 GUI 환경의 사용자 인터페이스를 제공하기 위하여 리눅스 프레임 버퍼 상에서 동작하는 윈도우 개발 환경인 Embedded Qt 3.0을 사용하였다[9]. 개발 PC에 Qt/Embedded 3.0.6 라이브러리를 설치한 환경에서 애플리케이션을 개발하였다. VODCA 서버는 C를 기반으로 제작되었으나 Qt는 C++ 기반이기 때문에 상호간의 통신에 문제점이 발생하였다. 따라서 Qt의 라이브러리를 최소한으로 사용하고 대부분의 라이브러리를 표준 C 라이브러리를 사용하도록 함으로서 문제를 해결하였다. 특히 Qt에서는 사용자 입력 이벤트 처리를 하기 위한 함수들이 다른 기능의 함수들로 재정의 되어 있어 시그널 처리가 용이하지 않기 때문에 타이머 함수를 이용하여 0.5초마다 사용자의 입력을 검사하는 방식으로 구현하였다. 애플리케이션과 애플리케이션에 사용된 라이브러리를 타깃 디스크에 복사하여 클라이언트에 이식하였다.

4.1.3 영화 재생기

영화 재생기는 Open Source Project로 개발 중인 mplayer 0.18 버전의 소스를 이용하였다[10]. mplayer에서 시그널 핸들러를 등록하여 고속 재생과 역재생시에 오디오를 재생하지 않도록 수정하였다. mplayer에서 제공되지 않는 고속 재생과 역재생을 구현하기 위해 시그널 핸들러를 통해 데이터를 선별하여 수신하는 기능을 추가하였다. mplayer의 기존 HTTP 연결 기능을 이용하여 여러 MMS 노드로부터 데이터를 수신하고 데이터를 재조합 하는 기능을 새로 추가하였다. 또한 기존의 2단계 버퍼링 기능을 3단계 버퍼링으로 강화하였다.

클라이언트 보드에 장착된 플래시 디스크의 128MB의 용량 중 37MB만 사용하였다. 이중 개발과 디버깅을 위한 유틸리티들과 일부 라이브러리를 제거하면 약 24MB의 크기로 이식이 가능하다.

4.2 시스템 동작 및 성능

4.2.1 시스템 동작

그림 2, 3, 4, 5는 내장형 클라이언트를 사용하여 VODCA로 접속하여 영화를 선택하는 화면을 나타내고 있다. 클라이언트의 전원을 켜면 부팅 과정을 거쳐 클라이언트의 메인 화면이 나타난다. 사용자가 리모콘의 조작을 통해 장르를 선택하고 영화를 검색하여 상영을 요청하면 영화가 재생된다. 재생 도중에 리모콘 조작을 통해 일반 재생, 고속 재생, 역재생, 일시 정지간의 자유로운 전환이 가능하다.

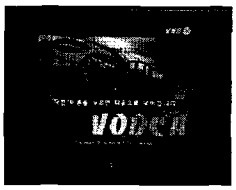

그림 2 메인 화면

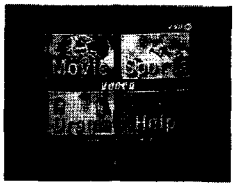

그림 3 장르 선택 화면

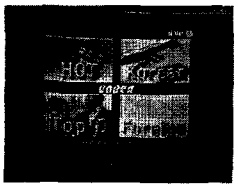

그림 4 영화 검색 화면

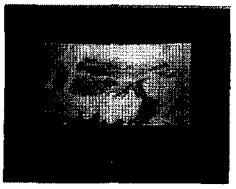

그림 5 영화 재생 화면

4.2.2 성능

클라이언트에서 VODCA 서버에 접속하여 영화를 상영할 경우 메모리는 적게 소모하지만 CPU 자원의 대부분 사용하는 것으로 측정되었다. 클라이언트에서 일반 재생에서 고속 재생이나 역재생으로 변환할 경우 1초 정도의 지연 시간이 발생되었다. 고속 재생에서 역재생으로 전환할 경우에는 최대 3초의 지연 시간이 발생하였다. 이런 결과는 사용자 명령 전달 지연과 버퍼링 문제로 분석되었다.

사용자 명령 전달 지연 문제는 GUI 모듈에서 사용자 입력을 검사하는 속도가 0.5초 간격이기 때문에 최대 0.5초의 지연이 발생되며, 클라이언트에서 HS를 거쳐 각 MMS 노드에 전달되는 네트워크 지연이 원인이었다. 버퍼링 문제는 각 MMS 노드의 버퍼를 비우고 다시 채우는 시간에 따른 지연과 클라이언트의 버퍼에 남은 비디오 데이터가 재생되면서 그 시간만큼 지연이 발생되는 것으로 분석된다. 따라서, 사용자 명령 전달 지연 문제는 사용자 입력 검사 속도를 빠르게 함으로서 개선할 수 있을 것이다. 버퍼링의 문제는 각 MMS 노드에서는 버퍼를 채우지 않고 바로 전송하는 기능을 추가하며, 또한 하나의 스트림을 지원하는 버퍼의 개수를 증가시켜 버퍼 선 인출 및 사용 버퍼 내용의 재사용을 통하여 문제의 지연 시간을 단축시킬 수 있을 것이다. 클라이언트 버퍼의 경우 버퍼 내용을 flush시키는 기능을 개선하여야 할 것으로 보인다.

4. 결론 및 향후 연구 계획

본 논문에서는 클러스터 VOD 서버에서의 내장형 클라이언트 플랫폼을 설계 및 구현하였다. 구현된 내장형 클라이언트는 VODCA 서버의 모든 기능을 지원하면서, 사용자의 편의성을 최대한 고려하여 적외선 리모콘으로 모든 기능을 제어 할 수 있다. 또한, 쉽게 TV와 연결하여 사용할 수 있도록 설계 및 구현하여 사용자의 시스템 접근 친화도를 향상시켰다.

향후에는 MPEG 4를 지원하는 서버 및 클라이언트에 관한 연구를 진행 할 것이다. 또한, 유비쿼터스 환경에서 VOD 서비스를 받을 수 있는 연구를 진행 할 계획이다.

참고문헌

[1] D. James Gemmell, Harrick M. Vin, Dilip D. Kandlur, P. Venkat Rangan, "Multimedia Storage Servers : A Tutorial and Survey" , IEEE Computer, pp . 40-49, May 1995.

[2] 배인한, 천성광, "분산 주문형 비디오 시스템을 위한 영화 할당 알고리즘의 설계 및 평가", 정보과학회논문지(A) 제25권 제6호, pp. 536-548, 1998

[3] Joseph Kee-Yin Ng, Calvin Kin-Cheung Hui, Wai Wong, "A Multi-server Design for a Distributed MPEG Video System with Streaming Support and QoS Control", IEEE RTCSA, 2000

[4] 최숙영, 유관종, "병렬 VOD 서버의 확장을 위한 스트라이핑 기법", 정보과학회논문지 : 정보통신 제28권 제3호, pp. 426-434, 2001

[5] Calvin K. Hui, Jodeph K. Ng, Wai Wong, Karl R.P.H. Leung, "The Implementation of a Multi-server Distributed MPEG Video System", IEEE RTAS, 2001

[6] 서동만, 방철석, 이좌형, 김병길, 정인범, "QoS를 지원하기 위한 리눅스 클러스터 VOD 서버의 성능 분석", 정보과학회 제30회 춘계학술발표회 논문집 게제 예정, 2003

[7] Gerard Beekmans, "Linux From Scratch Version 3.3", (http://www.linuxfromscratch.org)

[8] Tom Fawcett, "The Linux Bootdisk HOWTO" (http://www.tldp.org)

[9] "Qt/Embedded Whitepaper", trolltech, (http://trolltech.com/products/embedded/)

[10] mplayer 개발 사이트 (http://mplayerhq.hu/)