High Availability On-Demand Stream 서버 설계 및 구현

김경훈, 류재상, 김성, 남지승
전남대학교 컴퓨터공학과
e-mail : pluit@mdclab.chonnam.ac.kr

Design And Implementation of HA(High-Availability) On-Demand Stream Server System

KyungHoon Kim, JaiSang Ryu, Sung Kim, Jiseung Nam
Dept. of Computer Engineering, Chonnam National University

요 약

멀티미디어 서버 시스템은 고용량이어야 하며 지속적으로 늘어나는 사용자수 뿐만 아니라 추가되는 새로운 저장 공간에 대한 우수한 확장성을 제공하여야 하는 것이 필수 불가결이다. 그러나 현재의 범용 서버 시스템은 이러한 요구사항을 충분히 반영하지 못할 뿐만 아니라 늘어나는 사용자 부하와 시스템 요구에 대한 고려, 그리고 미디어 데이터에 대한 반영이 이루어지지 못하여 점차 증가되는 사용자의 고화질 미디어 서비스 요구 사항을 충족시키지 못하고 있다. 본 논문에서는 구현된 시스템의 서비스 방식과 확장성 있는 구조가 고 대역폭 고화질 On-Demand 서버로서 효율적인 대안임을 보이며 또한 QoS 요구 보장과 효율적인 시스템 Management 정책을 제시하여 범용 서버를 멀티미디어 저장 및 Streaming 서버에 적합한 환경의 클러스터로 구성하는 방법을 제시한다.

1. 서론

멀티미디어 저장 서버는 그 용량의 방대함 뿐만 아니라 고속의 네트워크 자원, 또한 사용자 접근의 동시성 및 임의성, 자원의 분배 및 활용, 제공 컨텐츠 특성에 따른 제어 등 시스템 성능을 위한 많은 부분이 고려되어 설계 되어야 한다.[1]

기존 솔루션은 상대적으로 취약한 네트워크 자원을 최대로 활용하기 위한 스트리밍 기술 위주로 그 연구 개발이 행하여 졌으나 네트워크 인프라가 각 가정에 보편적으로 보급되고 이를 기반으로 한 새로운 고품질 서비스 요구를 수용하기 위해서는 미디어 특성을 최대로 반영하는 고성능 미디어 서버로 최적화 되어야 한다. 이것은 로컬 환경의 미디어 재생을 네트워크로 확장하는 것으로 사용자에게 로컬 환경과 같은 환경의 미디어 시청을 가능하게 해주는 강력한 서버가 필요하다.

기존의 범용 서버를 최적화된 미디어 서버로 이용하기 위해서는 하드웨어와 미디어 전송 및 관리 S/W 의 두 가지의 구성요소가 필요하다.

H/W 는 많은 용량과 빠른 전송 속도를 갖는 저장 장치가 가장 중요한 요소이며 충분한 데이터 전송을 위한 I/O 장치 역시 중요하다. 저장장치의 물리적인 성능 뿐만 아니라 효율적인 데이터 배치 및 관리 도구가 필수적인데 이것은 매우 임의적인 사용자 서비스 요구에도 불구하고 예상 가능한 다양한 변수가 존재하기 때문이다. 영화 서비스를 위한 서버의 경우 동시에 요구되는 미디어는 전체 저장 미디어의 극히 일부일 가능성이 있고 장시간 서비스를 받는 사용자의 특성에 따라 초기지연 시간을 조정하는 등 부족한 시스템 자원을 최대로 이용하게 할 수 있는 다양한 제어를 통하여 그 같은 상황에 맞추어 시스템 운영을 탄력적으로 할 수 있기 때문이다. 또한 새로운 형식의 미디어 파일 타입과 로컬 타입의 컨텐츠를 수정없이 네트워크 상에서 이용하기 위하여 거기에 맞는 제어 역시 미디어 솔루션의 중요 포함 요소가 되었다.

구현된 시스템은 기본적으로 여러대의 서버가 동시에 하나의 클라이언트를 위해 다중의 접속경로를 갖는 형태로 구성되었다. 미디어 타입과 관계없이 모든 파일 타입 지원을 위하여 기본적으로 TCP 가 전송 프로

토콜로 선택되었고 미디어 특성에 맞는 데이터 흐름을 자율적으로 조정하고 여러 서버의 제어흐름을 조정하고 관리하기 위한 제어 서버 노드가 별도로 존재한다. 이는 다른 타입의 데이터 전송 예를 들어 인터페이스용 웹 데이터, 데이터베이스 트랜잭션 데이터 등을 미디어 전송서버로부터 분리 시키기 위한 목적으로 존재한다. 또한 사용자 인터페이스로서의 웹 서버와 미디어 관리용 데이터 베이스, 각 저장 서버에 데이터를 분산 저장하고 콘텐츠를 관리 재배치 하는 관리자용 부 프로그램들로 전체 시스템이 구성된다.
이와 같은 구조로 인하여 사용자 부하를 효율적으로 각 서버 노드에 분산 시키고 새로운 노드와 저장 공간 추가 시에 생기는 중복저장의 단점을 해결하는 새로운 형식의 시스템 구조를 가지게 되었다.

2. 시스템 구조 및 서비스 방법

구현된 On-Demand 저장 서버는 하나의 컨트롤서버와 이의 제어를 받는 최소 2 대이상의 저장 노드 전송 서버들로 구성된다. 컨트롤 서버는 사용자 인터페이스를 위한 웹 서버와 DB 가 설치되며 각 노드들의 부하와 시스템 자원상태를 제공받는 프로그램 그리고 컨트롤 노드의 데이터를 분산 저장 하고 재배치, 관리하는 프로그램들도 구성된다.
저장 서버에는 클라이언트로부터 요구받은 데이터를 전송하는 프로그램이 설치된다.
이와 같은 기능적인 분배는 저장노드가 다른 특성을 갖는 데이터 전송으로 인하여 디스크와 I/O 장치를 소모하는 Overhead 감소를 위해 고려되었다.
저장 노드는 동일한 시스템 환경으로 구성하는 것이 가장 바람직하지만[2] 다른 시스템들의 사용도 관리자의 올바른 배치와 시스템 관리로 성능을 최대화 할 수 있다. 클라이언트에는 다중 접속 경로를 갖는 데이터 수신 모듈이 탑재되고 이는 기존 플레이어 코덱에 데이터를 공급한다.

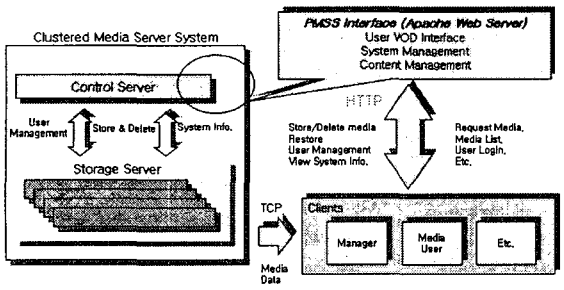

그림 1. 전체 시스템 구조

사용자는 먼저 컨트롤 노드의 웹서버를 통하여 데이터 목록을 확인하고 사용자 인증 후 서비스 요청을 한다. 컨트롤 서버는 사용자가 선택한 데이터의 저장 구조와 위치등을 표시한 메타 데이터를 사용자 클라이언트에 전송하고 이 메타 데이터를 이용하여 클라이언트는 저장 노드들과 다중 접속 경로를 설정하고 동시 수신 받은 데이터를 조합 하여 미디어를 재생하게 된다. 그림 1 은 전체 시스템 구조를 나타낸 것이다. 새로운 저장 노드가 추가 됨으로써 전체 시스템은 동시 사용자 로드를 전체 시스템에 분산한다. 또한 새로운 노드에 장착된 저장 디스크는 기존 저장 데이터의 재분산 저장을 통하여 전송 부하를 저장 노드간에 분산 할 수 있다. 극단적으로 전체 데이터의 재 분산 (Re-striping)을 통하여 전체 시스템 부하를 분산 할 수 있고 필요에 따라 앞에서 언급한 바와 같이 일부 인기 데이터에 대한 재분산만 실시하여 로드 균형을 맞출 수도 있다. 이것은 전체적으로 시스템 가용성에 중점을 둔 시스템의 설계 원칙을 반영하는 것으로써 관리자는 충분한 시스템 상황과 서비스 상황을 모니터하고 이와 같은 정책에 따라 시스템 환경을 구성한다.

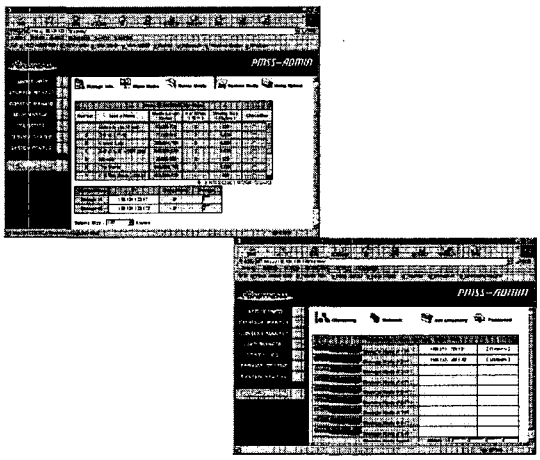

그림 2. 관리자의 시스템 확장과 데이터 저장 관리

그림 2 는 시스템 확장과 저장 공간 관리를 위한 웹 기반의 시스템 인터페이스를 나타낸 것이다
결국 시스템은 데이터의 분산 저장정책에 의하여 전체 시스템 성능에 영향을 미치는데 효율적인 관리자의 시스템 환경 구성을 위한 올바른 시스템 성능 모니터링은 이러한 관점에서 매우 중요하다.
또한 시스템의 성능 관련 모니터링은 사용자 수용에 관한 적합한 정보를 수용 제어기에 제공하여야 하는데 잘못한 정책으로 인한 사용자 수용으로 인하여 전체 사용자에게 서비스 품질을 제공할 수 없는 상황이 발생하는데 이것은 최악의 상황으로 반드시 고려되어야 하는 부분이다. 따라서 성능을 최대화 할 수 있는 충분한 시스템 모니터링의 구현과 이를 토대로 시스템 가용성을 최대로 할 수 있도록 하는 시스템 구성 정책, 또한 이러한 정책을 통하여 산출된 값을 통하여 안정된 서비스 수준의 사용자 수에 따라 시스템을 운용하고 보다 안정된 고품질 미디어 서비스를 위해서는 적합한 예비율을 산출 시스템 폭주에 대비하여야 한다. 이와 같은 고려로 인하여 시스템은 다소 불편한 관리 매커니즘을 갖는데 이것은 앞에서 설명한 부분에 대한 세세한 배려 때문이라고 할 수 있다. 개발 시스템은 관리자와 사용자 모두 웹 인터페이스를 통한 접근을 하는데 관리자는 미디어 데이터를 컨트롤

서버에 업로드 하여야 하고 이를 각 저장 서버에 스트라이핑 정책에 따라 분산 저장하여야 한다. 이와 같은 이중 데이터 저장은 컨트롤 노드에 중요한 원시 미디어 데이터를 저장 하여 여러 노드에 걸쳐 데이터를 분산 배치하는 데서 생기는 데이터 손실과 에러에 대비하는 백업 및 복구 데이터로 사용할 수 있다. 이 같은 구조 때문에 컨트롤 서버에는 전체 저장 노드의 데이터 용량만큼의 저장 공간이 필요하게 되나 사용자 데이터의 중요성에 따라 이러한 백업 대상 데이터의 보관 여부를 결정할 수 있다. 시스템의 오류에 대비한 백업 공간의 낭비는 어느 시스템에서 발생 하는 문제로서 단일 시스템으로 구성되었다 하더라도 이 같은 디스크 사용은 해결 할 수 없는 문제이다.

RAID 디스크의 물리적 복구 가능한 레벨을 적용 하거나 SAN 같은 자능형 저장장치를 채용하더라도 짧은 시간 시스템을 복구하거나 서비스 보장을 위해서는 별도의 백업 데이터가 필요하다. 구현 시스템에도 이러한 저장 장치를 채용하면 같은 효과를 기대 할 수 있으나 많은 비용을 소모하므로 효율적이지 않다. 시스템 클러스터를 통하여 RAID 의 수준의 성능을 기대 할 수 있고 거기에 시스템 확장은 추가 노드의 증설로 간단히 확장되고 이에 대한 별도의 부하 분배기가 필요하지 않는점 등이 개발 시스템이 갖는 중요한 장점이다.

사용자가 미디어 서비스를 이용하는 과정은 다음과 같다. 사용자는 컨트롤 서버에 접속하여 관리자가 데이터를 분산 저장하면서 자동으로 생성된 미디어 정보를 통해 서비스 요구 데이터를 결정하고 컨트롤 서버로부터 실제 데이터가 저장된 분산 서버들과 저장 스트라이핑 사이즈, 서버 오류로 인하여 서비스 중단시 재 전송을 맏게될 백업 시스템 서버 정보등 스트리밍 서비스를 위한 정보를 가진 메타 데이터를 전송 받는다. 전송된 메타 데이터를 통한 정보로 사용자 클라이언트는 각각의 저장서버와 연결을 설정하고 저장 정보와 분산 데이터 사이즈에 따라 적합한 버퍼링과 전송 데이터 블록을 전송 받고 이를 조합하여 미디어를 재생하게 된다. 각 저장 서버는 사용자 스트림이 정보를 컨트롤 서버에 전송함으로서 콘트롤 서버는 현재 수용되어 있는 사용자 정보 서비스 중인 미디어 정보 및 타입 등의 관리 데이터를 보유하게 된다. 이것은 기존의 스트리밍 서버가 메타 데이터를 획득한 불법 사용자로 인한 문제를 해결 할 뿐만 아니라 새로운 사용자를 수용하는 정책을 위한 입력 데이터의 역할은 한다.

그림 3 는 현재 저장 노드의 프로세서,메모리, 디스크 등의 자원 상황을 모니터링 하는 관리자 인터페이스를 나타낸 것이다.

이 같이 단순한 모니터링 뿐만 아니라 여기에 표시되는 데이터는 새로운 사용자를 수용하고 시스템을 관리하는 데 좀 더 유용하게 사용된다.

새로운 사용자 서비스 요청을 컨트롤 서버가 받게 되면 수용제어 프로그램은 현재 각 노드에서 전송되는 시스템 모니터링 정보와 서비스 요구가 발생한 데이터의 타입과 특성을 반영하여 서비스 수용 가부는 물론 한계범위 안에서 사용자 서비스 시작 시점을 결정한다.

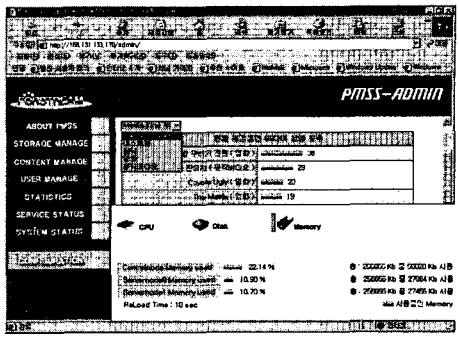

그림 3. 시스템 자원 상황 모니터링

이것은 뒤에서 설명할 VBR 타입의 시간적으로 가변성을 갖는 시스템 자원을 요구하는 사용자 클라이언트 서비스 요구를 적용하는데 필요하다.[1] 가변 데이터의 최대 데이터 전송 요구를 가능하면 중첩 수용하지 않아 동시에 전체 시스템에 걸리는 부하 요구를 줄이고자 하는 목적에 따라 이러한 정책이 반영되었다. 예를 들어 어떤 VBR 데이터는 어떤 서비스 주기 동안에는 클라이언트 서비스 품질을 보장하기 위해서 많은 디스크 대역폭을 사용하지만 어떤 주기 동안에는 매우 작은 데이터 READ 만을 필요로 할 수도 있다 결국 수용제어기는 전체 서비스 데이터의 평균 혹은 최대 요구 주기 데이터 요구량을 기준으로 이러한 사용자의 자원 사용을 예상하여야 하는데 평균데이터의 전송률의 선택은 QoS 보장을 위해 합리적인 선택은 아니다. 사용자의 서비스 요청 시점은 지극히 임의적일 뿐만 아니라 사용자 수신 프로그램의 버퍼링 만으로 이러한 문제를 완충 시키기에는 일반적인 VBR 데이터의 가변 폭이 상당히 심하기 때문에 적합하기 않다. 따라서 최대 주기를 기준으로 사용자 수용을 하여야 하는데 그러기에는 시스템 자원이 너무 많이 낭비되는 단점이 있다.

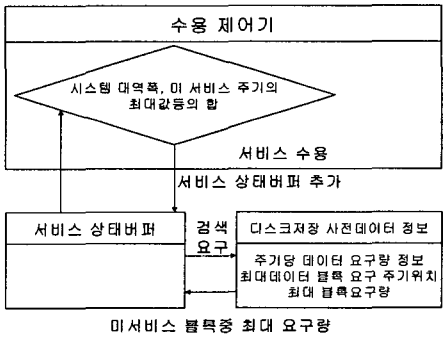

그림 4. VBR 데이터 수용제어 정책

그래서 본 시스템에서는 데이터 베이스에 이미 저장

된 저장 정보와 미디어 정보를 바탕으로 최대 주기 데이터의 시간적인 위치 정보를 새로운 사용자 수용 기준의 인자로 반영한다. 따라서 현재 수용되어 서비스 되고 있는 전체 사용자 부하는 새로운 사용자가 수용될 때마다 새로 계산된다.

그리고 새 사용자의 서비스 미디어는 처음에는 최대 주기를 기준으로 계산 수용되고 최대 서비스 주기를 이미 재생 했다면 나머지 데이터의 최대 주기가 다시 새로운 서비스 요청의 수용 기준이 되는 방식이다.

이 같은 방법은 VBR 데이터를 서비스 하는데 전체 시스템 효율을 어느 정도 개선 할 수 있으나 사용자가 미디어 재생 위치를 변경 했을 경우에는 문제가 발생할 수 있다. 시스템에서는 사용자의 서비스 위치 변경 시마다 새로운 수용제어를 행하게 하는 방법으로 이 같은 문제를 해결하고 있으나 그 결과로 약간의 계산 지연이 사용자에게 부여된다.

그리고 이러한 재생 위치 변경이 만일 수용제어기를 통하여 전체 시스템 가용 능력을 초과할 경우 사용자의 미디어 검색은 허용되지 않는다.

그림 4 는 VBR 데이터의 수용 정책을 나타낸 것이다. 사용자 클라이언트의 수신 모듈은 기존의 Windows Media Player 의 Source 필터의 형태로 구현되어 있다.[3] 따라서 미디어 플레이어가 지원하는 모든 표준 형식의 미디어 데이터를 모두 지원한다. 하지만 스트리밍 포맷 데이터의 재생은 개발 시스템의 스트리밍 정책과 부합하지 않으므로 배재 했다.

따라서 클라이언트는 웹서버를 통하여 미디어의 리스트를 확인하고 수신 모듈을 웹서버를 통해 간단히 전송 받아 설치하면 기존 미디어 플레이어를 통해 미디어 서비스를 받을 수 있다.

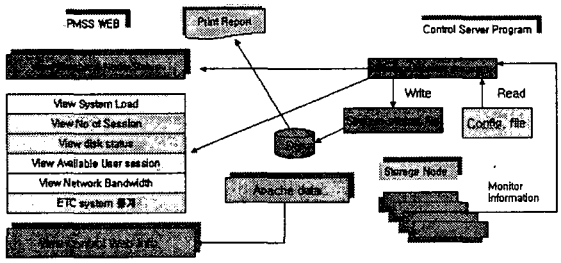

그림 5 시스템 인터페이스 및 관리 구조

그림 5 은 시스템의 서비스 인터페이스 작성과 관리 위한 구조를 그림 6 은 데이터 저장 구조를 도식화 한 것이다. 미디어 서버를 위해서는 기존 서버에 단지 미디어 데이터를 네트워크 상으로 전달하고 이 데이터를 수신하고 재생하는 프로그램 들로 기본적으로 그 역할을 수행한다. 하지만 범용 시스템들로 많은 다수의 동시 접속자에게 서비스를 제공하여야 하는 서버를 구성하기에는 좀더 미디어 서비스 측면의 시스템 구성이 필요하다. 개발 시스템은 이미 기본적인 성능이 검증된 상태로 포스트림에서 제품으로 1 차적인 버전이 출시되었으며 다양한 형태로 테스트되고 문제점이 지속적으로 개선되고 있다. 본 시스템은 분산 저장과 병렬 전송등의 특징적인 방법을 사용하여 시스템의 성능과 부하 분산을 수행하였으며 기본적으로 미디어 서비스 시 요구되는 기능에 대한 부 프로그램들로 효율적이고 고성능의 미디어 서버 시스템을 설계 구현 하였다.

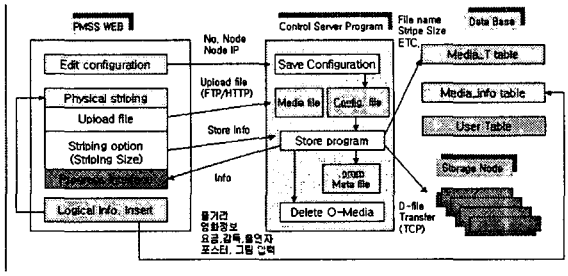

그림 6 데이터 저장 구조

3. 결론 및 향후 연구 과제

미디어 서버 뿐만 아니라 서버시스템은 그 안정적인 운영이 무척 중요시된다. 본 시스템 역시 이와 같은 안정성 보장을 위해 백업 서버 운영방안 또한 다수의 서버의 공동 작업인 서비스 형태의 보완을 위한 연구를 지속적으로 수행하고 있다. 향후 인터넷 서비스의 주종을 이룰 데이터는 고용량 고품질의 멀티미디어 데이터가 될 것이다. 네트워크의 인프라는 마치 프로세서의 발전 속도와 같이 비약적으로 발전하고 있으나 이와 같은 추세에 맞는 서비스 시스템은 상대적으로 부족하다. 많은 수요의 멀티미디어 서버가 필요하게 될 것이며 따라서 다수 사용자에게 안정적으로 서비스를 제공할 서비스 시스템 역시 지속적으로 연구되어야 한다. 우리의 시스템은 병렬 전송과 분산 저장으로 기본적으로 시스템 확장을 부하 분배기나 백업 시스템 등 의 도움 없이도 구성 할 수 있도록 개발되었다. 또한 미디어 서비스를 위한 각종 정책과 관리도구들을 활용해 미디어 서버의 성능을 최대화 할 수 있도록 고려했다. 향후 서비스의 주종을 이룰 미디어 타입에 적합한 정책을 연구하여 시스템에 반영하고 성능을 최적화 할 수 있는 관리 구성 값을 찾아내고 적용하는 연구가 지속적으로 필요하다.

참고문헌

[1] Huang-Jen Chen T.D.C Little,"Storage Allocation Policies for Time-Dependent Multimedia Data",IEEE Trans. On Knowledge and Data Engineering, Vol. 8, No. 5. pp 855-864,1996

[2] S.Moyer and V.Sunderam,"Parallel I/O as a Parallel Application." Journal of Supercomputer Application, Vol 9,No.2,1995.

[3] MicroSoft,"MicroSoft Media Service SDK",1998