효율적인 부하 분산 정책을 위한 WMI 기반 VOD 서비스의 설계 및 구현

방한민, 박총명, 서동만, 김학수, 정인범*
강원대학교 컴퓨터정보통신공학
e-mail : hmbang@snslab.kangwon.ac.kr

Design and Implementation of WMI based VOD Service for efficient Load Balance Policy

Han-Min Bang, Chong-Myung Park, Dong-Mahn Seo,
Hark-Soo Kim, In-Bum Jung*
Program of Computer and Communications Engineering,
Kangwon National University

요 약

클러스터 VOD 시스템에서 한정된 자원을 사용하여 많은 사용자에게 안정적인 QoS를 제공하기 위해서는 클러스터 노드들 간의 효율적인 부하 분산이 필요하다. 본 연구에서는 효율적인 부하 분산 방법으로 윈도우즈 환경에서 클러스터 각 노드의 정보를 WMI를 이용하여 부하 분배 서버가 필요할 때마다 원격 질의를 통해 네트워크의 오버헤드를 줄이면서 각 클러스터 노드의 장애 상황을 파악하고 복구하는 방법에 대해 연구한다.

1. 서론

주문형 비디오 즉 VOD(Video On Demand) 서비스는 사용자의 요청에 따라 네트워크를 통해 실시간으로 사용자에게 멀티미디어 데이터를 서비스 함을 목적으로 한다.

이러한 VOD 서비스의 성능은 사용자에게 일정한 QoS(Quality of Service)를 보장 하는 최대 동시 접속자의 수로 나타낼 수 있다. 최적의 VOD 서비스를 제공하기 위해서는 VOD 서버의 특성, 저장 장치의 성능, 네트워크 대역폭 등에 대한 관련성 연구가 선행되어야 한다. 특히 VOD 서버는 사용자들에게 공급하는 스트리밍 데이터를 저장, 인출하는 역할을 하며 사용자의 요청이 많을시 성능이 쉽게 포화되는 문제를 가지고 있다. 고성능의 단일 VOD 서버는 성능을 향상시키기 위해 서버를 확장하기가 어렵다는 단점이 있는 반면, 클러스터형 VOD 서버는 새로운 노드의 추가를 용이 하게 함으로써 성능 향상을 기대할 수 있는 확장성을 갖추고 있음은 물론 가격대 성능비가 우수하다는 장점을 가지고 있다[1-4]. 이러한 클러스터형 VOD 서버에서의 각 클러스터 노드의 정확한 자원 현황 파악을 통한 효과적인 부하 분산은 VOD 서비스의 QoS를 향상 시킬 수 있는 요소가 된다.

본 연구에서는 클러스터형 VOD 서버에서의 효과적인 부하 분산을 위한 선행 연구로서, 각 클러스터 노드의 정확한 자원 현황을 분석 할 수 있는 방법을 윈도우즈 환경에서 WMI(Windows Management Instrumentation)를 통해 구현하며, 각 클러스터 노드의 장애를 파악하여 장애를 복구하는 방법에 대해 연구한다. 본 연구에서는 사용자의 영상 요청에 의해 WMI를 이용하여 분배서버가 클러스터 노드의 상태정보를 가져오는 방식을 사용한다. 클러스터 노드의 CPU, 메모리, 네트워크 대역폭의 사용량을 가져와 각 자원의 사용률 계산을 통해 부하 분산 정책을 적용함으로써 클러스터 노드간의 부하균형을 유지 하도록 한다. 또한 WMI에서 지원 하는 WQL(WMI Query Language)을 이용하여 클러스터 노드 프로세스의 장애를 파악하고, 해당 프로세스를 재가동 함으로써, 클러스터 노드의 장애를 대비를 할 수 있는 방법을 연구 한다.

- 본 연구는 교육과학기술부와 한국산업기술재단의 지역혁신 인력 양성사업으로 수행된 연구 결과임.

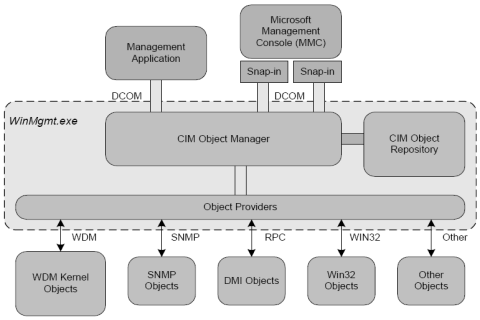
*교신저자

2. 관련 연구

2.1 WBEM(Web-Based Enterprise Management)

WBEM은 엔터프라이즈 환경에서의 관리를 통합하고자 발전된 인터넷 표준 기술들과 관리 솔루션의 집합이라 할 수 있다. WBEM은 다양한 형태의 플랫폼과, 동일 플랫폼 상의 여러 시스템 자원들을 제어하고 감시하며 관리하는 DMTF의 표준화 기술 중 하나이다. 따라서 WBEM은 서로 다른 이기종 시스템을 통합하는데 사용되며, 통합 표준이었던 SNMP나 CMIP등 기존 관리 표준 모델들을 하나의 표준으로 통합 하는것이 주 목적이다[5].

WBEM은 모든 정보를 객체화 하여 표현하기 때문에 이를 지원 하기 위한 오버헤드가 발생한다. 또한 모든 메시지는 XML로 부호화되어 표현되므로 이를 지원하기 위



(그림 1) WMI 구조

한 오버헤드가 발생한다. 하지만 WBEM은 모든 표준을 하나의 표준으로 통일한다는 주 목적처럼 효율성 보다는 일반성과 용이성에 더 큰 비중을 두고 있다.

2.2 WMI(Windows Management Instrumentation)

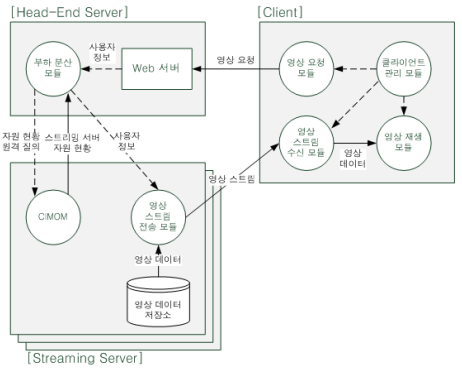
WMI 은 마이크로소프트에서 윈도우즈 환경을 위해 구현한 WBEM(Web-Based Enterprise Management) 이다. WMI 는 (그림 1) 과 같이 개체 정의 저장에 관한 표준 메커니즘과 관리 데이터 획득 및 배포에 관한 표준 프로토콜, WMI 데이터 제공자의 역할을 하는 동적 연결 라이브러리로 구성된다. WMI 제공자는 CIM(Common Inform -ation Model) 스키마에 대해 개체 장치에 대한 데이터를 공급한다[5, 6].

2.3 CIMOM(Common Information Model Object Manager)

CIMOM는 마이크로소프트사에서 구현한 WBEM 기술의 핵심 구성 요소이다. WBEM의 중심 목표는 데이터를 통일된 형태로 표현하고, 그 데이터를 CIM 개체 저장소에서 개체 지향적으로 캡슐화하는 것이며, CIM 개체 관리자는 저장소에 저장되어 있는 관리 대상 개체에 대한 수집 및 조작 지점을 제공한다. CIMOM는 이들 관리 대상 개체에 대한 정보의 수집 및 조작을 간편하게 하는 편이성을 제공한다. 참고로, CIMOM는 관리 정보에 직접 액세스하지 않고 WMI 제공자가 리소스(관리 대상 개체)로부터 정보를 수집한 다음, 관리 응용 프로그램이 사용할 수 있도록 WMI API 를 통해 제공한다. 간단히 말해서, CIMOM는 WMI 에서 CIM 기능을 제공하는 것이라 할 수 있다[5, 7].

2.4 WQL(WMI Query Language)

WQL은 이벤트 통지를 비롯한 WBEM 호환 기능들이 지원되도록 구조적 쿼리 언어(SQL)을 변형 시킨것이다. 사용자는 이벤트를 통지 받도록 등록할 때 이벤트의 유형과 그 이벤트가 전달되길 원하는 조건을 정의하는 쿼리를 지정한다. 정의된 WQL은 사용자에 의해 로컬 및 원격지의 컴퓨터에 DCOM 서비스를 통해 원격질의를 실행 할 수 있다.



(그림 2) WMI를 이용한 VOD 서비스 구조도

3. WMI 를 이용한 자원 현황 분석

3.1 WMI를 이용한 VOD 서비스의 구현

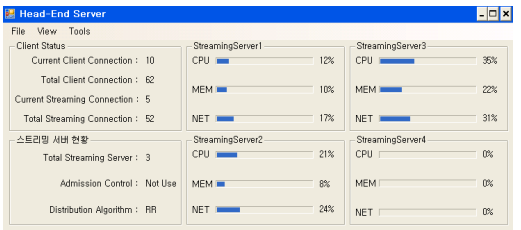
(그림 2)는 WMI를 이용하여 구현된 VOD 시스템의 구조도 이다. 구현된 시스템에서는 클라이언트 사용자의 영상의 요청에 의해 HS(Head-end Server) 는 각 클러스터 노드의 CIM에 원격 질의를 실행 한다. 원격 질의를 통한 정보를 바탕으로 3.1절에서 논의된 방법으로 각 장치의 사용률을 계산한다. 이 사용률은 부하 분산 알고리즘의 정보로 사용되게 되며, 부하 분산된 알고리즘에 의해 HS는 사용자의 연결 정보를 클러스터 노드에게 전송한다. 클러스터 노드는 HS 로부터 전송받은 정보를 바탕으로 사용에게 스트림을 전송한다.

서버의 모든 모듈은 윈도우즈 환경에서 C# 언어와 .NET Framework 3.5 버전으로 구현되었으며, 사용자의 접근성을 높이기 위해, 웹 서버를 통해 사용자의 영상에 대한 요청을 요구 받게 된다. (그림 2)는 구현된 VOD 서비스의 구조도 이다.

(그림 3)은 HS 의 실행화면으로 연결된 스트리밍 서버의 각 자원 현황을 가져와서 화면이 보여준다. 실행화면에서 보듯이 총 4대의 클러스터 노드로 구성된 환경에서 WMI를 이용하여 각 클러스터 노드들의 자원 정보를 가져와 관리자에게 제공한다.

3.2 WMI 기반의 부하 분산 서버 구현

VOD 서비스의 성능은 효율적인 부하 분산 알고리즘에



(그림 3) HS 실행 화면

의해 많은 성능의 차이를 보인다[8]. 본 연구에서는 WMI를 이용하여 HS(Head-End Server) 에서 사용자의 영상 요청의 발생에 따라 각 클러스터 노드에 원격 질의를 통해 CPU, 메모리, 네트워크 대역폭 사용량을 가져온다.

CIM과 관련된 WMI 네임스페이스는 "root\CIMV2"의 위치에 존재하게 되며, 각 CPU, 메모리, 네트워크와 관련된 클래스는"Win32_PerfFormattedData_PerfOS_Processor","Win32_PerfFormattedData_PerfOS_Memory","win32_PerfFormattedData_Tcpip_NetworkInterface" 에 존재 한다.

CPU 사용률의 경우 특정 연산에 많은 자원이 사용되는 경우가 발생하므로 특정 시점에서의 CPU 점유율만을 가지고 부하를 분배하는 것은 효율적이지 못한 결과를 가져온다. 따라서 본 논문에서는 CPU의 사용률을 2번 측정하여 단위 시간당 발생하는 사용률 간의 차로 사용률을 구한다.

$$(1-\frac{CPU_{Usage\,Current}-CPU_{Usage\,Old}}{TimeStamp_{Current}-TimeStamp_{Old}})*100$$

(수식 1) CPU 사용률 측정

(수식 1) 은 평균적인 CPU의 사용률을 계산하기 위한 수식이다. $CPU_{UsageCurrent}$ 는 현재 CPU의 점유율이며, $CPU_{UsageOld}$ 는 단위 시간 전에 측정된 CPU의 점유율이다. $TimeStamp_{Current}$ 와 $TimeStamp_{Old}$ 는 대상 컴퓨터의 100 나노초 간격의 타임 스탬프이다.

메모리의 사용률은 물리 메모리의 전체 크기와 현재 사용중인 물리 메모리의 사용량을 바탕으로 사용률을 계산한다. 물리 메모리의 전체 크기의 정보를 담고 있는 클래스는 "Win32_ComputerSystem" 이므로, 현재 메모리 사용량 정보를 관리하는 클래스가 서로 다르다. 따라서 메모리의 사용률 계산을 위해서는 2번의 원격 질의를 수행해야 한다.

메모리의 경우 스트리밍 서버에서 사용자에게 스트림을 전송할 경우 각 사용자의 수에 비례하는 결과를 보이며, CPU와 같이 짧은 시간 동안의 큰 변화가 발생 하지 않는다. 따라서 이전 값과의 비교를 통해 사용률을 구하는 것은 비효율적이다.

$$(1-\frac{Memory_{Usage}}{Memory_{Total}})*100$$

(수식 2) 메모리 사용률 측정

(수식 2)는 메모리의 사용률을 구하는 식이다. 위의 식과 같이 전체 메모리 크기와 현재 메모리 사용량을 비교함으로써 사용률을 구한다.

네트워크 대역폭의 사용률 측정은 현재 이더넷이 연결된 회선의 대역폭에 초당 사용량을 비교하여 결정한다.

네트워크에 바인드된 전체 네트워크 대역폭의 정보를 관리는 "Win32_PerfFormattedData_Tcpip_NetworkInterface" 내의 "CurrentBandWidth" 프로퍼티 이며, 초당 네트워크 대역폭 사용량 프로퍼티는 "ByteTotalPerSec" 이다. 이 정보를 바탕으로 네트워크 대역폭의 사용률을 계산한다.

$$(1-\frac{Byte\,Total\,Per\,Sec}{Current\,Band\,Width*8})*100$$

(수식 3) 네트워크 대역폭 사용률 측정

(수식 3)은 네트워크 대역폭 사용률을 계산하는 식이다. 네트워크에 바인드된 네트워크 대역폭의 크기와 초단위의 네트워크 대역폭 사용량과 비교함으로써, 매초 서비스를 위해 사용되는 네트워크 대역폭의 사용률을 계산한다.

3.3 WMI를 이용한 클러스터 노드의 장애 판단 및 복구

클러스터 노드의 자원 현황을 파악하기 위해 사용된 방법으로, 주기적으로 클러스트 노드와 정보를 교환하는 방법과 사용자의 요청이 발생하였을 경우 클러스터 노드와 정보를 교환 하는 방법 등이 있다. 전자의 경우는 주기적으로 정보를 교환함으로써, 클러스터 노드의 장애를 빠르게 파악할 수 있는 장점이 있지만, 네트워크의 오버헤드가 발생하게 된다. 후자의 경우, 네트워크의 오버헤드는 줄어들지만 장애를 파악하기 위해서는 오랜 시간이 발생할 수 있다. 따라서 본 연구에서는 WMI를 사용하여 후자의 경우와 같이 네트워크상의 오버헤드는 줄이면서 장애 판단과 장애의 복구를 효율적으로 할 수 있는 기법을 구현하였다.

윈도우즈 환경에서 동작중인 프로세스에 대한 정보 또한 WMI를 통해 정보를 얻을 수 있다. 이와 관계된 클래스는 "Win32_Process" 이다. 이 클래스의 프로퍼티로서 Caption 부분에 현재 동작중인 모든 프로세스의 네임을 가지고 있다. 따라서 부하분배 서버는 각 클러스터 노드의 장애를 파악하기 위해 원격 질의로서 "Win32_Process" 클래스의 Caption을 질의함으로써, 현재 클러스터 노드에서 동작중인 모든 프로세스에 대한 정보를 얻을 수 있으며, 이를 바탕으로 클러스터 노드의 장애를 파악한다.

또한 위의 기법으로 판단된 장애를 복구하는 방법으로 "Win32_Process" 클래스에 존재하는 Create 메소드를 사용한다 이 메소드는 인자로 주어지는 프로그램의 경로를 바탕으로 프로그램을 실행하는 기능을 한다. 따라서 부하분배 서버에서 원격질의를 통해 클러스터 노드의 장애를 판별하고, 장애를 복구 하기위해 프로세스를 재가동하는 기법을 구현하였다.

4. 결론 및 향후 연구

본 논문에서는 효율적인 부하 분산 정책을 위한 선행연구로서, 각 클러스터 노드의 자원 정보를 가져 오는 방법과 장애 판단 및 복구 기법을 설계 및 구현 하였다.

구현한 서버에서는 사용자의 영상 요청이 발생한 시각에 WMI의 원격 질의를 통해 각 스트리밍 서버 노드의 자원 현황을 파악하고 부하를 분배 하게 된다. 따라서 자

원 현황 파악을 위해 주기적인 정보 교환으로 소모되던 네트워크 자원의 오버헤드를 줄일 수 있다. 또한 위의 WMI를 이용하여 장애를 판단하고 장애를 복구하는 방법을 제시함으로써, 기존의 부하분배 서버와 각 클러스터 노드간의 주기적으로 정보교환을 통해 얻을 수 있었던 장점 또한 얻을 수 있었다.

본 연구는 효율적인 부하 분산 알고리즘 연구를 위한 선행 연구로서, 향후에는 원격 질의로 얻은 각 스트리밍 서버 노드의 자원 현황에 가중치를 적용하여, VOD 서비스의 성능을 향상 시킬 수 있는 기법을 연구할 계획이다.

참고문헌

[1] Dinkar Sitaram, Asit Dan, "Multimedia Servers: Appli-cations, Environments, and Design," Morgan Kaufmann Publishers, 2000.

[2] Joseph Kee-Tin 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.

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

[4] Jack Y.B. Lee, "Parallel Video Servers: A tutorial," IEEE Multimedia, pp. 20-28, 1998.

[5] 조희남, 안창원, 정성인, 김영호, 김지연, "오픈 소스 프로젝트 WBEM 구현물에 대한 분석," KNOM Review, Vol.7, No. 1, 2004.

[6] Open Group - http://www.opengroup.org

[7] KwonGee Kwak, "WMI Background and Overview," Microsoft, June 1999.

[8] Dongmahn Seo, Joahyoung Lee, Yoon Kim, Changyeol Choi, Hwangkyu Choi, Inbum Jung, "Load Distribution Strategies in Cluster-based Transcoding Servers for Mobile Clients," Lecture Notes in Computer Science, Vol 3983, pp. 1156-1165, May 2006.