http://dx.doi.org/10.5392/JKCA.2012.12.02.030

클라우드 데이터 서비스를 위한 대용량 데이터 처리 분산 파일 아키텍처 설계

Distributed File Systems Architectures of the Large Data for Cloud Data Services

이병엽*, 박준호**, 유재수**
배재대학교 전자상거래학과*, 충북대학교 전자정보대학 정보통신공학과**

Byoung-Yup Lee(bylee@pcu.ac.kr)*, Junho Park(junhopark@chungbuk.ac.kr)**,
Jaesoo Yoo(yjs@chungbuk.ac.kr)**

요약

최근 클라우드 컴퓨팅 시장에 진출했거나 진출을 선언한 글로벌 IT 기업들을 이미 보유하고 있는 하드웨어, 소프트웨어 기반 기술들을 활용하거나 상호 협력을 통해 다양한 클라우드 서비스들을 제공함으로써 불특정 다수를 대상으로 급격하게 성장하고 있는 클라우드 컴퓨팅 시장에서 자신들의 영역을 지속적으로 확장해 나가고 있다. 분산 파일 시스템은 데이터의 저장과 관리뿐만 아니라 상위 계층 서비스가 요구하는 충분한 성능과 안정성을 보장해주기 위한 클라우드 컴퓨팅의 핵심 기술 중의 하나이다. 본 논문 에서는 클라우드 컴퓨팅을 위해 분산 파일 시스템이 갖추어야 할 사항들과 클라우드 컴퓨팅에서 활용 가능한 오픈 소스 기반의 하둡 분산 파일 시스템, 메모리 데이터베이스 기술, 고가용성 데이터베이스 시스템을 소개하고 현재 클라우드 컴퓨팅 시장에서 활용되고 있는 분산 파일 시스템의 동향을 통한 다양한 분산처리 기술을 참고하여 대용량 분산 데이터 처리 아키텍처를 구현하였다.

■ 중심어 : | 클라우드 | 클러스터 | DBMS | Grid | 대용량데이터 |

Abstract

In these day, some of IT venders already were going to cloud computing market, as well they are going to expand their territory for the cloud computing market through that based on their hardware and software technology, making collaboration between hardware and software vender. Distributed file system is very mainly technology for the cloud computing that must be protect performance and safety for high levels service requests as well data store. This paper introduced distributed file system for cloud computing and how to use this theory such as memory database, Hadoop file system, high availability database system. now In the market, this paper define a very large distributed processing architect as a reference by kind of distributed file systems through using technology in cloud computing market.

■ keyword : | Cloud | Cluster | DBMS | Grid | VLDB |

I. 서 론

클라우드 컴퓨팅 서비스는 2006년 아마존이 컴퓨팅 환경을 서비스로 제공하는 EC2 서비스와 스토리지를 서비스로 제공하는 S3 서비스를 시작하였다. 아마존의 클라우드 서비스는 크게 중소기업, 개발자들을 대상으로 한

* 본 연구는 2012년 교육과학기술부로부터의 지원(지역거점연구단육성사업/충북BIT연구중심대학육성사업단)과 2009년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업의 결과임.(No. 2009-0089128)
접수번호 : #111207-002
접수일자 : 2011년 12월 07일
심사완료일 : 2011년 12월 29일
교신저자 : 유재수, e-mail : yjs@chungbuk.ac.kr

스토리지 서비스 S3, 웹호스팅 서비스 ECS, 웹서비스 AWS로 분류된다. 이후 구글의 AppEngine, 마이크로소프트의 Azure등 많은 클라우드 컴퓨팅 서비스가 출시되고 있다. 이러한 클라우드 서비스를 제공하는 환경은 모두 공통적으로 분산 컴퓨팅 플랫폼을 기본 인프라로서 활용하고 있다. 따라서 분산 컴퓨팅 플랫폼을 얼마나 잘 구축하고 운영하는가가 클라우드 컴퓨팅 서비스 제공자의 성공을 뒷받침할 가장 큰 능력이라 할 수 있다[14]. 최근 클라우드 컴퓨팅 시장에 진출했거나 진출을 선언한 Google, IBM, Microsoft, Oracle 등과 같은 글로벌 IT 기업들은 그 동안의 하드웨어, 소프트웨어의 기술력을 바탕으로 클라우드 컴퓨팅을 제공하는 데 필요한 IT 인프라 및 소프트웨어의 서비스 형태를 지속적으로 확충해 나가고 있으며 제반 기술들을 개발하고 향상시키기 위해 막대한 개발 투자를 쏟아 붓고 있다[14]. 또한 공개 소프트웨어의 도입이나 기업 간의 전략적인 협력을 통해 부족한 부분들을 상호 보완하면서 다양한 클라우드 서비스들을 제공함으로써 급격하게 성장하고 있는 클라우드 컴퓨팅 시장에서 자신들의 영역을 확장하기 위해 끊임없이 노력하고 있다. 최근 클라우드 컴퓨팅 시장에 뛰어든 기업들에게 있어서 클라이언트 수준의 서비스에서 소요 및 수요 되어야 하는 대용량의 데이터를 저장하고 안정적인 관리를 하기 위해 지불해야 하는 비용을 최대한 줄이는 것이 가장 큰 이슈로 부각되고 있으며, 점점 다양해지고 있는 사용자들의 요구를 충족시킬 수 있는 클라우드 서비스들을 찾아내고 편리하게 사용할 수 있는 환경을 제공하여 보다 많은 사용자들을 확보하기 위한 경쟁이 치열해지고 있다. 클라우드 컴퓨팅을 위해 IT 인프라를 저렴하게 구축하고 효율적으로 활용할 수 있는 기술들에 대한 연구가 활발히 진행되고 있으며, 막대한 양의 데이터를 수많은 서버들에 분산 저장하고 관리하는 분산 파일 시스템 기술은 그러한 기술 중의 하나로 인식 될 수밖에 없다. 클라우드 컴퓨팅에서 요구하는 분산 파일 시스템은 단순히 데이터의 저장과 관리만을 하는 것이 아니라 하드웨어의 장애에 유연하게 대처하여 서비스가 중단되지 않도록 하는 고가용성의 시스템 아키텍처가 필요하며 또한 적절한 병렬 처리를 통해 서비스가 요구하는 성능도 클라이언트를 만족시킬 수 있어야 한다. 현재 수많은 분산 파일 시스템들이 존재하지만 최근까지 클라우드 컴퓨팅에 활용되고 있는 분산 파일 시스템은 그리 많지 않다. 대표적으로 Google의 Google 파일 시스템[2]과 Apache의 Hadoop 분산 파일 시스템[3]이 클라우드 컴퓨팅에서 가장 잘 알려진 분산 파일 시스템이다.

클라우드 컴퓨팅을 위해 분산 파일 시스템이 갖추어야 할 사항들은 실로 다양하다. 이러한 사항들을 도출하기 위해 클라우드 컴퓨팅의 대표적인 특징 들을 분산 파일 시스템 측면에서 살펴보면 다음과 같이 몇 가지로 크게 요약해 볼 수 있다[14].

첫째, 클라우드 컴퓨팅을 위한 분산 파일 시스템들이 다루는 데이터와 서버의 규모는 기존의 분산 파일 시스템들과는 비교가 되지 않을 정도로 거대하다는 점이다. 대부분의 핵심적인 상황들은 여기에서 발생하며 비용적인 측면에서의 효율성, 지속적으로 증가하는 데이터의 수용, 빈번하게 발생하는 고장에 대한 대처, 관리의 편리성과 같은 사항들이 해당된다[15].

둘째, 점점 다양해지고 있는 사용자들의 요구를 충족시키기 위해 제공되는 클라우드 서비스들이 분산 파일 시스템에게 만족할 만한 데이터의 입출력과 처리 성능을 요구한다는 점이다. 여기에는 대용량 파일에 대한 신속한 입출력 성능, 네트워크 위상 구조 인식을 통한 데이터 최적 배치, 효과적인 캐시의 사용, 순간적으로 집중되는 부하의 유연한 대처와 같은 사항들이 해당된다.

셋째, 클라우드 컴퓨팅을 활용하고자 하는 기업들뿐만 아니라 일반 사용자들은 데이터에 대한 보안문제를 가장 불안해하고 있으며, 확실하고 안전한 보안 체계를 요구한다는 점이다. 여기에는 데이터에 대한 암호화, 데이터 영역에 대한 사용자간 엄격한 접근 제어, 사용자 데이터에 대한 관리자의 접근 제한 같은 사항들이 해당된다.

넷째, 사용자가 저장한 데이터에 오류가 발생하지 않도록 방지하고, 저장 공간을 최적으로 사용하기 위한 방법들을 요구한다는 점이다. 여기에는 데이터 오류 감지 및 복구, 데이터 중복 제거와 같은 사항들이 해당된다[14][15]. 본 논문의 목적은 최근 IT 메가트랜드로 이슈화가 되고 있는 클라우드 서비스 구축을 위한 기술적인 측면에서의 서비스 성능의 극대화를 위한 파일시스템들을 제언 하였다. 특징적으로 데이터베이스의 데이터 서

비스의 확장 적으로 비정형데이터의 원활한 서비스를 위한 오픈소스 기반의 하둡 시스템과 메모리 데이터베이스를 기반으로 한 새로운 아키텍처를 제시 하였다.

따라서 본 논문의 구성은 다음과 같다. I장 서론에서는 클라우드 기술 발달의 배경을 소개하고, Ⅱ장 본론 에서는 클라우드 컴퓨팅을 이해하기 위한 일반적인 서비스 내용을 소개와 클라우드 컴퓨팅의 분산데이터 처리 기술 동향을 대표적인 클라우드 서비스 기업들을 분석 기술하였다. 또한 클라우드 컴퓨팅의 핵심 기술로 고가용성 서비스를 가능케 하는 그리드(Grid) 기술 구성들을 소프트웨어 아키텍처 측면에서 제시 하였다. 마지막 결론 Ⅲ장에서는 본 논문을 통해 정리된 고가용성 클라우드 기술을 토대로 결론 및 향후 과제를 제시하는 것으로 결론을 내고자 한다.

II. 본론

1. 클라우드 컴퓨팅의 이해

클라우드 컴퓨팅이란 인터넷 기술을 활용하여 '가상화된 IT 자원을 서비스'로 제공하는 컴퓨팅으로, 사용자는 IT 자원(소프트웨어, 스토리지, 서버, 네트워크)을 필요한 만큼 빌려서 사용하고, 서비스 부하에 따라서 실시간 확장성을 지원받으며, 사용한 만큼 비용을 지불하는 컴퓨팅을 말한다[1]. 따라서 안정적인 대용량의 클라우드 데이터 서비스를 위해서는 반드시 필요한 사항들이 선행되어야 한다.

1.1 저비용 분산파일 시스템

클라우드 컴퓨팅에서 제공하는 서비스는 제한적인 것은 아니지만, SaaS, PaaS, IaaS 세 가지를 가장 대표적인 서비스로 분류한다. 애플리케이션을 서비스 대상으로 하는 SaaS는 클라우드 컴퓨팅 서비스 사업자가 인터넷을 통해 소프트웨어를 제공하고, 사용자가 인터넷상에서 이에 원격 접속해 해당 소프트웨어를 활용하는 모델이다[11-13]. 클라우드 컴퓨팅에서 다루어지는 데이터는 꾸준히 증가하고 있으며 이러한 데이터를 저장하고 관리하기 위해 막대한 IT 인프라가 요구된다. 만일 고가의 서버와 고속의 네트워크를 활용하여 이러한 인프라를 구축하고 유지할 수 있다면 두말할 나위 없이 좋겠지만 비용을 고려했을 때 현실적으로 쉽지 않은 것이 사실이다. 따라서 클라우드 컴퓨팅 시장에 뛰어든 기업에게 있어서 이러한 인프라를 구축하고 유지하는 데 드는 비용을 최소화하는 것은 가장 중요한 경쟁력 중의 하나이다. 이를 위해 저렴한 서버들과 네트워크를 활용하여 비용을 대폭적으로 줄이면 서도 비슷한 성능을 낼 수 있는 분산 파일 시스템 기술들을 개발하고 적용해야 한다.

1.2 표준화된 확장성

클라우드 컴퓨팅 기업들은 꾸준히 증가하는 데이터의 저장과 관리를 위해 초기 구축된 IT 인프라를 지속적으로 확장해야 한다. 이러한 확장에 있어서 분산 파일 시스템은 논리적으로 공간 확장을 무한히 할 수 있어야 하며 공간을 확장하거나 축소할 때 서비스의 중단이 발생하지 않도록 해야 한다. 간단히 말하면, 서비스의 중단 없이 공간의 관리가 가능해야 한다는 것이다. 이를 위해 분산 파일 시스템은 모든 서버들의 공간을 사용자들에게 단일 공간으로 보이도록 가상 화를 수행하며, 이러한 가상화를 통해 서비스의 중단 없이 자유롭게 공간을 관리하고 사용자들에게는 단일 저장소로 사용할 수 있도록 한다.

1.3 고가용성

앞서 언급하였던 저비용 측면에서 저가의 장비들을 대규모로 활용하여 시스템을 구축함으로써 고장이 빈번하게 발생할 수밖에 없다. 물론 고가의 장비들은 저가의 장비들에 비해 고장이 좀 더 적게 발생 하기는 하지만 고장이 발생한다는 사실 자체는 변하지 않는다. 중요한 점은 이러한 고장으로 인해 시스템 전체 또는 서비스를 중단시키는 상황이 발생하지 않아야 한다는 것이다. 분산 파일 시스템은 모니터링을 통하여 고장이 발생한 상황을 인지하고 적절하게 대처함으로써 서비스가 중단되는 상황을 방지해야 한다. 또한 고장이 발생한 서버나 디스크로 인해데이터가 유실되는 것을 방지하기 위해 데이터의 중복 저장과 같은 안정적인 데이터 보관 방법이 갖춰져야 한다.

1.4 고성능

기존의 PC 환경에서 사용하던 수많은 응용들이 클라우드 서비스를 통해 클라우드 컴퓨팅 속으로 들어오고 있다. 이러한 응용들의 접근 패턴은 분산 파일 시스템의 설계에 있어서 성능과 밀접한 관련을 갖는다. 예를 들면 사진, 동영상과 같은 대용량의 데이터들을 다루는 클라우드 서비스들이 최근 많은 관심을 끌고 있는데, 분산 파일 시스템의 입장에서 바라볼 때 이러한 응용들은 대용량의 데이터를 순차적으로 입출력하는 패턴을 가지며, 대부분 저장된 데이터를 읽어가는 연산을 수행한다. 이러한 패턴에 최적화된 분산 파일 시스템은 이러한 패턴에 있어서는 최상의 성능을 나타내겠지만, 저용량의 임의적인 입출력 패턴의 경우에는 현저하게 성능이 저하될 수 도 있다. 결과적으로 분산 파일 시스템이 모든 접근패턴에 최고의 성능을 보장할 수 있으면 좋겠지만 사실상 그것은 상당히 어려우며 분산 파일 시스템 상위에서 접근 패턴 자체를 분산 파일 시스템이 지원하는 최적의 접근 패턴으로 변형하여 최상의 성능을 얻어내거나 적절한 다른 분산 파일 시스템을 활용하는 것이 보다 현실적인 대안이 될 수 있다. 분산 파일 시스템은 서버, 스위치, 랙 등의 네트워크 위상 구조를 인식하고 이를 이용하여 최적으로 데이터를 배치함으로써 클라이언트의 요청을 빠르게 처리할 수 있어야 한다. 또한 메모리 캐시를 효율적으로 활용하여 디스크 입출력을 최소화함으로써 성능을 향상시킬 수 있다. 앞서 예로 들었던 클라우드 서비스에서 수많은 사용자들이 관심을 가지고 짧은 기간 안에 접근하게 되는 hot 데이터라 불리는 특정 데이터는 과도한 접근으로 인해 서비스가 불가능한 상태에 빠질 수도 있는데, 이러한 것을 방지하기 위한 방법이 갖춰져야 한다.

1.5 편의성

클라우드 컴퓨팅을 활용하는 기업들과 일반 사용자들은 클라우드 서비스를 통해 제공되는 기능들만을 사용하므로 분산 파일 시스템에 대해 알 필요가 전혀 없지만 분산 파일 시스템을 활용하여 클라우드 서비스를 제공하고자 하는 기업의 관리자는 현재 분산 파일 시스템에 의해 관리되고 있는 모든 사항들을 모니터링하고 분석해야 하며 분산 파일 시스템의 상세한 사항을 이해하고 있어야 한다. 소규모의 시스템의 경우는 관리자가 명령 라인을 통해 분산 파일 시스템을 관리하고 여러 상황들을 확인할 수 있지만, 대규모의 클라우드 컴퓨팅 환경에서는 이러한 작업이 불편하고 어려울 수밖에 없다. 따라서 클라우드 컴퓨팅을 위한 분산 파일 시스템은 관리자에게 웹 또는 GUI 형태의 관리 도구를 통해 분산 파일 시스템이 관리하는 모든 서버나 네트워크의 상황을 일목요연하게 보여주고 손쉽게 관리할 수 있는 기능을 제공해야 한다.

1.6 보안성

클라우드 서비스를 이용하는 사용자들이 가장 우려하는 사항이 바로 보안성이다. 우려를 하고 있는 사항은 크게 두 가지로 볼 수 있는데, 첫째는 관리자나 해당 기업의 관계자에 의한 정보 유출이다. 이 유형의 경우는 클라우드 서비스를 이용하는 사용자들 의 데이터가 사용자의 동의 없이 상업적으로 이용되거나 유출되는 것이다. 또한 사용자의 동의 없이 삭제된 데이터가 보관되는 것도 포함된다. 둘째는 취약한 보안 허점으로 악의적인 해킹에 의해 사용자의 데이터가 손실되거나 유출되는 것이다. 이러한 사용자들의 우려를 불식시키기 위해서는 철저한 데이터 관리와 더불어 제도적인 사항이 뒷받침되어야 하며, 전문적인 보안관련 기업들과의 협력을 통해 지속적으로 보안을 강화해야 한다. 분산 파일 시스템의 입장에서는 철저한 사용자 인증과 접근 허용 권한 관리가 필수적이다.

1.7 데이터 무결성

데이터를 저장하다 보면 하드웨어적인 오류나 알 수 없는 원인에 의해 의도된 대로 데이터가 저장되지 않고 오류가 발생하는 경우가 종종 생긴다. 분산파일 시스템은 이러한 상황으로부터 사용자의 데이터를 보호하기 위해 오류를 검출하고 복구할 수 있어야 한다. 클라우드 컴퓨팅에서 다루어지는 데이터의 양은 너무나도 방대하여 기존의 파일 시스템 검사 방법으로는 모든 데이터를 검사한다는 것이 사실상 불가능하다. 따라서 분산 파일 시스템의 전체 데이터를 빠르고 효과적으로 검사할 수 있는 방법이 갖춰져야 한다.

2. 클라우드 컴퓨팅 분산 파일 시스템 기술

현재까지 무수히 많은 분산 파일 시스템들이 존재하고 있으며 여전히 진화를 거듭하고 있다. 최근 클라우드 컴퓨팅이 부각되면서 기존의 분산 파일 시스템 기술들이 다양한 형태로 활용되고 있으며, 적절한 변형을 통해 클라우드 컴퓨팅으로의 접근을 꾀하고 있다. 이 장에서 살펴볼 클라우드 컴퓨팅에 활용되고 있거나 활용이 가능한 분산 파일 시스템들은 앞서 살펴보았던 요구 사항들을 반영하고 있다.

2.1 해외 클라우드 서비스 업체 동향

2.1.1 구글 파일 시스템

구글은 Google AppEngine을 통해 어플리케이션 개발을 용이하게 하는 플랫폼 서비스 및 개발을 위한 호스팅 공간을 제공, 구글 Earth 등 각종 어플리케이션 서비스를 제공 중이다. 최근에는 안드로이드 폰과 모바일 클라우드 서비스를 앞세워 글로벌 모바일 서비스 시장 공략에 노력하고 있다. 클라우드 컴퓨팅 기술에 기반을 둔 음성 및 위치인식 서비스를 향후 모바일 전략의 핵심으로 삼고 모바일 시장공략을 강화할 예정이다. 이를 위해 구글은 스마트폰이 데이터망에 연결 통로 역할을 할 수 있도록 카메라가 눈의 역할, 마이크가 귀의 역할, 터치센서가 촉각의 역할을 하도록 할 계획이다. 특히 높은 인식률을 위해 알고리즘을 개발하기보다 클라우드 컴퓨팅망의 방대한 데이터를 수집하고 연결하는데 더 많은 역량을 쏟아 정확한 정보를 찾아낼 수 있도록 할 계획이다.

Google은 클라우드 서비스를 제공하기 위해 자체 개발한 분산 파일 시스템인 Google 파일 시스템(이하 GFS)을 사용하고 있다[6][7]. GFS는 여러 클라이언트에 의해 접근되는 단일 마스터와 여러 청크(chunk) 서버들로 구성되며, 저가의 리눅스 서버 상에서 사용자 수준 프로세스로 구동된다. 마스터는 모든 파일 시스템 메타데이터를 관리하고 시스템 전반적인 동작을 제어하는 역할을 수행하며, 청크 서버들은 실제 데이터를 보관하고 클라이언트의 입출력 요청을 처리하는 역할을 수행한다. 이처럼 GFS는 메타데이터와 데이터의 흐름을 분리한 비대칭형 구조를 통해 확장성과 고성능을 보장하고 빈번하게 발생하는 장애에 적절히 대처함으로써 안정성을 확보한다. GFS에서 파일들은 청크라 불리는 64 메가바이트 크기의 단위로 나뉘어 관리되며 각 청크들은 여러 청크 서버에 분산되어 저장된다[8][9]. 또한 각 청크에 대한 다수의 사본을 여러 청크 서버에 분산하여 저장함으로써 장애로 인한 데이터의 손실을 방지한다. GFS는 이렇게 청크라 불리는 큰 단위로 데이터를 관리함으로써 마스터가 관리해야 하는 메타데이터 의 수를 줄이고 메타데이터를 교환하기 위해 필요로 하는 네트워크 부하를 감소시킨다[8].

2.1.2 아마존 분산 파일 시스템

Amazon S3 파일 시스템은 Amazon 웹 서비스에 의해 제공되는 온라인 스토리지 웹 서비스인 Amazon S3을 위한 분산 파일 시스템이다. 현재 Amazon S3 파일 시스템의 구조는 Amazon의 특허를 통해 살펴볼 수 있다[5]. S3 파일 시스템은 5 기가바이트 크기의 객체를 저장할 수 있으며 각 객체 당 최대 2 킬로바이트 크기의 메타데이터를 수반한다[15]. [그림 1]에서 처럼 각 객체는 버켓(bucket)들로 구성되며 각 버켓과 객체는 ACL를 통해 접근이 관리된다. 아마존 분산파일 시스템의 스토리지 모델은 사용자의 데이터를 저장하는 객체들과 객체들의 집합인 버켓들로 구성된다.

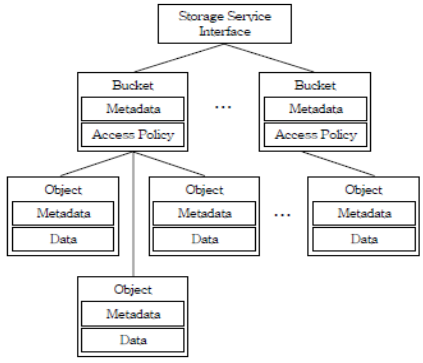

그림 1. 아마존의 스토리지 모델

사용자는 스토리지 서비스를 위해 제공되는 스토리지 서비스 인터페이스를 사용하여 버켓에 접근 할 수 있다. 스토리지 서비스 인터페이스는 우리가 일반적으로 사용하는 웹 주소 표현 방식을 그대로 사용한다. 버켓은 메타

데이터와 접근 정책으로 이루어진다. 버켓에 포함된 메타데이터는 버켓의 생성 시간, 생성자의 ID, 버켓이 객체들을 포함하고 있는지에 대한 정보, 버켓에 포함된 객체들의 총 크기, 버켓에 대한 사용자의 접근 기록, 버켓과 관련된 청구 기록 등이다. 객체는 메타데이터와 데이터로 이루어진다. 객체에 포함된 메타데이터는 객체의 생성 시간, 객체의 크기, 객체에 저장된 데이터의 타입, 객체의 사용과 이력 정보, 접근 정책 등이다. 각 객체들은 스토리지 서비스 시스템에서 키(key)와 로케이터(locator)에 의해 구별된다. 키는 버켓 단위로 유일한 값을 가지는 반면 로케이터는 모든 버켓에 대해 유일한 값을 갖는다.

3. 클라우드 컴퓨팅 분산 파일 기술 구성

3.1 데이터 분산처리 기술의 활용

안정성과 성능향상을 위해 HW Vendor에서 제시하고 있는 솔루션은 단적으로 볼 때 MPP(Massively parallel Processing)와 SMP(Symmetric multi processing) 머신으로 이어져 오면서, 업무에 따라 Scale-up 과 Scale-out 형태로 확장성이 진화해 왔다고 볼 수 있다. Scale-up이란 사용하고 있는 머신을 보다 큰 상위 기종으로 업그레이드를 하거나 기존 운영서버에 각종 리소스, 즉 CPU나 Memory확장 및 빠른 접근 속도를 주장하는 하드디스크를 추가로 장착하여 확장하는 방식이다. 이러한 Scale-up방식을 채택함으로써 얻을 수 있는 장점으로는 시스템 구조나 개발 및 운영에 변화가 없으며, 현재 사용하고 있는 Application이나 Data Source, Middleware등에 대한 수정이 전혀 필요 없다. 가장 손쉽게 당면하는 성능의 문제점을 해결할 수 있는 방법으로 자원이 추가되는 동안 서버만 잠시 다운시켜 빠른 시간 내에 확장을 할 수 있다는 장점이 있다. 그러나 단점으로는 확장 시 요구되는 비용이 상대적으로 크며, 지속적으로 확장을 하는데 에 한계가 있다는 점이다. 또한 SOPF(Single Point of Failure)에 대한 높은 장애 비용을 감안해야 한다는 점 또한 무시 할 수 없다.

데이터의 트랜잭션의 성능을 높이기 위해 고려가 되고 있는 최근의 동향으로, 데이터베이스 서버를 미들티어, 즉 메모리상에 올려놓고 트랜잭션을 처리함으로 Disk I/O에 대한 비용을 줄이고 메모리 I/O를 통해 처리토록 하는 방안으로 고려가 되고 있는 영역이 메모리 DBMS이다. 메모리 데이터베이스는 Disk 기반으로 트랜잭션을 처리할 때 발생되는 Disk I/O 부분을 메모리상에 데이터베이스를 둠으로 해서 I/O극대화로 성능 문제를 해결하고자 하는 것이다. 주로 Scale-up을 위해 고려되는 솔루션이며, Disk기반의 DBMS와 비교를 해 볼 때 나름대로의 장단점이 있다고 볼 수 있다. 따라서 클라우드 컴퓨팅 분산 파일시스템의 극대화를 위해 메모리 데이터베이스의 활용에 대한 기술을 구성하고자 한다.

3.1.1 메모리 분산 파일 시스템

데이터 소스로부터 읽혀온 데이터를 여러 미들티어 서버의 메모리상에 캐시 형태로 상주 시키면서, 어떤 노드를 통해 데이터를 요구하더라고 그리드로 연결된 노드상의 메모리에서 데이터를 찾아 빠르게 서비스하는 구조로 [그림 2]와 같이 제언 될 수 있다.

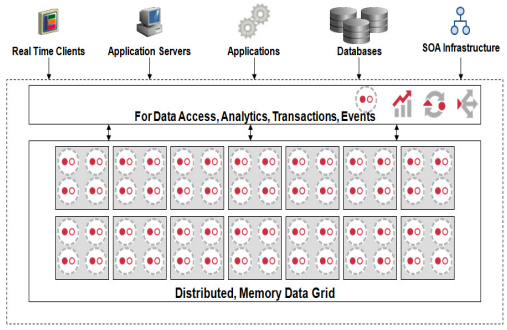

그림 2. 메모리 분산 파일 구조

위와 같은 메모리 구조의 특징을 살펴보면, 첫째 데이터 분산이 투명하게 처리가 되며, 각 노드의 메모리를 로드 분산을 자동 처리하는 장점이 있다. 또한 병렬처리 방식으로 데이터를 로딩 할 수 있으며, 데이터를 조회, 데이터 처리가 가능하며, 수정이나 입력 등 트랜잭션을 위한 처리일 경우 Queue에 보관하여 원하는 시기에 데이터베이스에 저장시킬 수 있다. 클러스터링 되어 있는 각 노드에 검색되어질 데이터를 Primary 데이터로 정의하며, 각 노드마다 Primary 데이터를 메모리에 저장하고 이 Primary 데이터에 대한 백업을 클러스터내의 다른 노

드들에게 분산되어 백업이 된다[그림 3]. 그러므로 사용자는 어떤 노드를 통해 접속하더라도 동일한 논리적인 데이터 뷰를 볼 수가 있게 된다. 두 번째는 클러스터 상의 여러 미들티어 노드들에 대해 가용상태를 서로 확인하여, 가용 상태가 불안정할 때는 나머지 노드들이 자체 상태를 진단하여, 불안정한 노드가 발생되었을 경우 클러스터 상에서 해당 노드를 제외시키며 남은 노드들이 불안정한 노드 상의 캐시에 있었던 데이터를 남은 노드들로 자동 재 분산시킴으로 Fault tolerance를 할 수 있는 아키텍처를 가지고 있다.

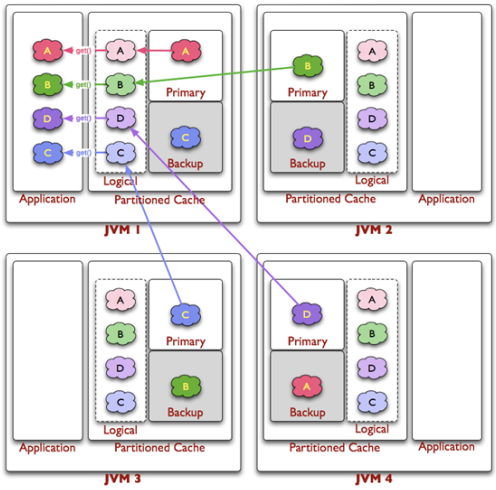

그림 3. 메모리 분산 파일 가용성 구조

3.1.2 데이터베이스 분산 시스템

초기의 그리드는 방대한 양의 계산이나 데이터를 필요로 하는 계산과학 분야에서 먼저 주목을 받기 시작했다. 기존의 계산 및 데이터 관리가 각 지역에 있는 슈퍼컴퓨터나 클러스터, 대용량 저장장치 등을 이용해서 이뤄지던 것을 그리드가 등장하면서 지금까지는 처리할 수 없었던 대용량의 계산 및 데이터를 처리할 수 있게 되었고, 처리 시간을 단축하거나 비용을 절감하는 등의 효과를 거두고 있다. 이후 대형 IT 기업들이 그리드 기술에 관심을 가지기 시작했고, SOA(Service Oriented Architecture)에 기초한 그리드 서비스의 개념이 자리 잡게 되면서 데이터베이스 그리드 컴퓨팅 기술은 차세대 비즈니스 모델을 위한 핵심인프라 기술로 인식되고 있다. 데이터베이스의 그리드 환경의 고 가용성을 구축해야 하는 크리티컬한 비즈니스의 IT환경의 요구에 따라 O사는 과거 OPS(Oracle Parallel Server)를 지원하였으며 현재 버전 에서는 각 노드간 캐시의 일치성을 보장하기 위한 서버간의 통신 방식을 디스크를 이용한 방식에서 초고속 인터커넥트를 이용한 캐시 퓨전(Cache Fusion)으로 변경하면서 고가용성 구현의 완성도를 높였다. 현재 O사의 10g에서부터 그리드 컴퓨팅을 지원하는 더욱 발전된 RAC(real application cluster)구조를 상용화 하였다. O사의 RAC는 다중 노드를 지원하는 공유 디스크 구조를 사용하여 한층 강화된 HA 솔루션을 데이터베이스에 제공 한다.

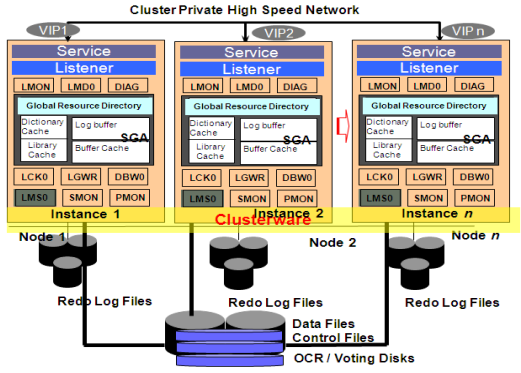

그림 4. 데이터베이스 Grid 구현 아키텍처

RAC는 동일 데이터베이스 또는 스토리지를 여러 인스턴스에서 동시에 액세스할 수 있는 장점을 가지고 있으며, 시스템 확장 즉 유기적으로 인스턴스 노드의 추가가 가능하기 때문에 탁월한 로드밸런싱 및 향상된 성능을 구현 할 수 있다. 또한 RAC 구조는 모든 노드가 동일한 데이터베이스를 액세스하기 때문에 한 인스턴스에서 장애가 발생해도 데이터베이스에 대한 액세스가 손실되지 않는 장점을 가지고 있다.

3.1.3 Hadoop 분산 파일 시스템

Hadoop 분산 파일 시스템(이하 HDFS)은 신뢰성과 확장성을 갖춘 분산 컴퓨팅을 위한 오픈 소스소프트웨어 개발 프로젝트인 Hadoop에서 분산 컴퓨팅 프레임워크

를 지원하기 위해 개발된 분산 파일 시스템이다. HDFS는 현재 Amazon, IBM, Yahoo등과 같은 글로벌 IT 기업들의 클라우드 컴퓨팅 플랫폼의 기반이 되는 분산 파일 시스템으로 가장 널리 활용이 되고 있다[4]. HDFS는 GFS를 본보기로 삼아 개발된 분산 파일 시스템으로 플랫폼간의 이식성을 보장하기 위해 자바를 사용하여 구현되었으며 공개 소프트웨어이다. HDFS는 이름만 다를 뿐 구글의 파일 시스템과 동일한 구조와 기능을 제공한다. 네임노드는 구글 파일시스템의 마스터와 동일하며 데이터 노드는 구글 파일시스템의 청크 서버와 동일하다. 또한 HDFS 에서 블록이라 불리는 파일 관리 단위는 구글 파일시스템의 청크와 동일하다.

(1) 클라이언트

HDFS 클라이언트는 HDFS 파일을 생성할 때 우선 자신의 로컬 저장 영역에 한 블록 크기의 임시 파일을 생성하고 그 파일에 먼저 데이터를 기록한다.

HDFS 클라이언트에서 HDFS 파일이 생성되는 과정을 살펴보면 다음과 같다.

- 한 블록 크기의 임시 파일이 데이터로 모두 채워지거나 더 이상 기록될 데이터가 없으면 클라이언트는 네임노드에게 메타데이터에 반영해 줄 것을 요청하고 사본이 위치할 데이터노드들의 목록을 얻어온다.
- 얻어온 데이터노드들의 목록에서 자신과 가장 근접한 첫 번째 데이터노드로 임시 파일과 나머지 데이터 노드들의 리스트를 전송한다.
- 첫 번째 데이터노드는 4 킬로바이트 단위로 데이터를 받아 자신의 로컬 파일 시스템에 파일로 저장하여 곧바로 다음 데이터노드로 데이터를 전송한다.
- 이러한 파이프라인 형태로 임시 파일의 모든 데이터를 목록에 포함된 데이터노드들로 전송하고 모든 데이터노드들이 정상적으로 그 데이터를 저장했는지 확인한다.
- 클라이언트는 모든 데이터노드들에 데이터가 정상적으로 저장된 것이 확인되면 네임노드에게 생성 완료를 알린다.

(2) 네임노드

간결하게 설계된 메타데이터 구조를 기반으로 모든 메타데이터를 메모리상에 유지하는 네임노드는 메타데이터를 자신의 로컬 파일 시스템을 사용하여 두 개의 파일로 저장한다. 첫 번째 파일은 네임스페이스, 접근 제어 정보, 파일과 블록들 간의 사상 정보, 블록들의 위치 정보와 같은 파일 시스템 메타데이터를 저장하는 FsImage라 불리는 파일이며, 두 번째 파일은 FsImage 파일이 기록된 시점 이후에 발생하는 모든 변경 사항을 기록하는 Edit-Log라 불리는 트랜잭션 로그 파일이다. 네임노드는 기동 시에 메타데이터를 메모리상에 구축하기 위해 FsImage 와 Edit-Log 파일을 사용하며, 구축이 완료된 후에는 현재 구축된 메타데이터를 FsImage에 저장하고 Edit-Log 파일에 체크포인트를 기록한다. 사본을 어떤 데이터노드에 배치할 것인가와 클라이언트에게 어떤 데이터노드로부터 데이터를 읽도록 추천할 것인가를 결정하는 정책은 안정성과 성능을 높이는 데 매우 밀접한 관련을 가진다. HDFS는 랙 인지(rack awareness)라 불리는 네트워크 위상 인지를 활용하여 기본적으로 3개의 사본을 유지하는 상황에서 동일한 랙의 서로 다른 서버에 한 개씩, 다른 랙에 있는 서버에 나머지 한 개를 배치하며, 이러한 배치 정책과 더불어 읽기 요청을 한 클라이언트에게 가장 가까운 데이터노드를 추천한다. 네임노드는 주기적으로 모든 데이터노드들로부터 자신의 상태를 나타내는 heartbeat과 자신이 관리하고 있는 블록들의 목록인 block report를 받는다. 네임노드는 heartbeat을 통해 장애가 발생한 데이터노드를 감지하고 그 데이터노드가 포함하고 있는 블록들에 대해 다른 데이터노드에 있는 사본을 활용하여 또 다른 데이터노드로 사본을 복제하는 방법을 통해 안정성을 확보한다. 또한 block report와 메타데이터로 유지하고 있는 블록들의 정보를 비교하여 일치하지 않는 블록들을 삭제하도록 데이터 노드에게 알려준다. 네임노드는 파일이나 디렉터리에 대한 삭제 요청에 대해 곧바로 삭제하지 않고 임시 디렉터리로 옮겨서 삭제된 것처럼 보이도록 하고 설정된 일정 시간이 지나면 실제로 삭제를 수행한다. 이러한 지연 삭제는 실제로 삭제되지 않은 데이터의 복구를 빠르게 할 수 있으며, 곧바로 데이터를 삭제할 때 발생하는 부하를 줄일 수

있는 장점을 지닌다.

(3) 데이터노드

HDFS 파일들은 블록의 단위로 서로 다른 여러 데이터 노드들에 분산되어 저장되며, 해당 데이터노드들은 로컬 파일 시스템의 파일로 블록들을 저장하고 관리한다. 데이터노드는 디렉터리 당 최적의 파일수를 고려하여 서브디렉토리들을 구성하고 블록에 해당되는 파일들을 적절한 위치에 저장한다. 또한 데이터의 무결성을 보장하기 위해 블록에 해당되는 파일을 저장할 때 그 파일에 대한 체크섬(checksum)을 별도의 숨겨진 파일로 저장한다. 클라이언트가 어떤 블록에 대한 읽기를 요청하면 데이터노드는 블록에 해당되는 파일과 숨겨진 체크섬 파일을 함께 전달한다.

3.2 분산처리 기술을 활용한 대용량 데이터 처리 아키텍처

앞서 살펴본 메모리데이터베이스의 기술, OLTP의 성능을 극대화 하기위한 클러스터링 기술, 비정형데이터의 RDBMS의 한계를 극복하기 위한 형태로의 수평적 확장성을 가지고 있고, RDBMS와는 다른 파일간의 릴레이션이 없는 Hadoop의 파일 시스템의 구조를 이용하여 [그림 5]와 같은 클라우드 데이터 서비스를 위한 대용량 파일 처리 시스템 아키텍처를 구현하였다. 클라우드를 데이터 서비스를 위한 가장 중요한 가용성에 대한 부분의 일정부분의 요구량과 트래픽에 대한 I/O를 적절하게 대응해야 하는 클라우드의 서비스의 특성상 위 아키텍처는 고가용성 서비스를 위한 OLTP 데이터베이스 서비스는 clustering 구조를 통해 데이터베이스 서버간의 상호 로드밸랜싱을 할 수 있는 구조로 설계되었으며, OLTP 및 어플리케이션의 데이터 서비스의 성능 극대화를 위한 메모리 데이터베이스 기술을 활용하였다. 또한 비정형화된, 데이터의 스키마가 필요 없는 대용량의 데이터를 처리하기 위해 오픈소스 기반으로 구글의 클라우드 서비스에서 대용량 데이터 처리로 검증된 대용량 분산 파일 처리 시스템인 Hadoop파일 시스템을 채택하여 아키텍처를 구현하였다.

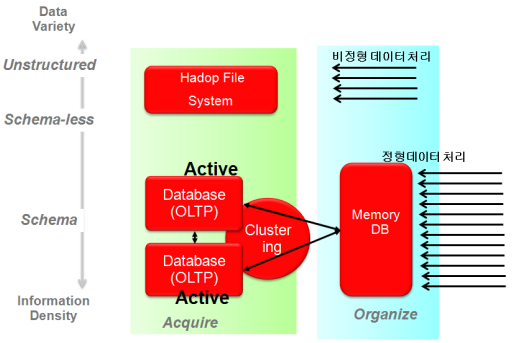

그림 5. 대용량 분산 파일 처리 시스템 아키텍처

Ⅲ. 결론 및 향후 연구

본 논문에서는 클라우드 컴퓨팅과 관련된 파일시스템들의 동향을 통해 최근 급격하게 성장하고 있는 대용량 데이터의 관리 기술 및 저장의 메커니즘을 이해하고 불특정 다수의 클라우드 서비스의 모델에 따른 데이터의 성능과 안정성을 보장해주기 위한 분산파일 시스템이 갖추어야 할 사항들을 살펴보았다. 클라우드의 가장 중요한 고가용성의 부분들을 고려하여, 데이터베이스의 고가용성 솔루션 및 또한 특징적으로 비정형 데이터의 처리를 위한 하둡 분산파일 시스템과, 메모리 데이터를 통한 시스템 퍼포먼스 향상 등을 조합한 새로운 데이터 아키텍처를 제시 하였다. 해외 클라우드 서비스의 선두 주자인 구글과, 아마존 역시 클라우드 서비스의 활용되고 있는 분산 파일 시스템들은 거의 유사한 구조와 비슷한 기능들을 갖추고 있는 것을 알 수 있고, 이는 클라우드 컴퓨팅에서 분산 파일 시스템은 비용적인 측면에서의 효율성, 지속적으로 증가하는 데이터의 안정적인 처리, 가용성에 대한 대처, 관리의 용이성, 데이터의 최적 배치 및 효과적인 캐시의 사용, 부하 집중에 대한 로드 밸랜싱, 데이터 보안등과 같이 상호 보완 되어야 할 사항들이 무수히 많다. 최근 다양한 해외 벤더들은 이러한 클라우드 서비스의 안정적인 서비스 및 대용량의 데이터들을 처리할 수 있는 오픈소스 기반의 데이터 분산처리 기술들을 선택하여 메모리데이터베이스 기반의 어플라이언스 제품들을 출시하기 시작 하였다. 본 논문은 국내외의 안정적인 클라우드 서비스를 준비하는 많은 기업들이 갖추어

야 할 주요 파일 시스템의 메커니즘을 이해하고 향후 클라우드 컴퓨팅을 위한 대용량 분산 파일 시스템으로 활용하기를 기대 한다.

참 고 문 헌

[1] Cloud computing, http://en.wikipedia.org/wiki/Colud_computing.

[2] Sanjay Ghemawat, H. Gobioff, and Shun-Tak Leung, "The Google File System," In Proc. of ACM Symp. on Operating Systems Principles, pp.20-40, 2003.

[3] HDFS, http://hadoop.apache.org/core/docs

[4] http://wiki.apache.org/hadoop/PoweredBy

[5] Distributed Storage System with Web Services Client Interface, US Patent No. 2007.

[6] http://www.pnfs.com

[7] http://en.wikipedia.org/wiki/PNFS

[8] pNFS BOF, http://www.pnfs.com/docs/sc08_pnfs_bof_slides.pdf

[9] http://sourceforge.net/projects/kosmosfs

[10] FUSE, http://fuse.sourceforge.net

[11] Dave Thomas, "Enabling Application Agility Software as a Service, Cloud Computing and Dynamic Languages," Journal of Object Technology, Vol.7, No.4, May-June 2008.

[12] 세일즈포스닷컴, "Salesforce 마케팅," http://salesforce.com

[13] KIPA, "SaaS 대표주자, Salesforce.com의 성장세 분석," 11월, 2007년.

[14] 민영수, 진기성, 김홍연, 김영균, 클라우드 컴퓨팅을 위한 분산 파일 시스템 기술 동향, 전자통신 동향분석, 2009.

[15] 김창수, 김학영,남궁한, 클라우드 서비스를 위한 대규모 클러스터 관리 기술 개발, 전자통신 동향분석, 2009.

저 자 소 개

이 병 엽(Byoung-Yup Lee) 종신회원

- 1991년 2월 : 한국과학기술원 전산학과(공학사)
- 1993년 2월 : 한국과학기술원 전산학과(공학석사)
- 1997년 2월 : 한국과학기술원 경영정보공학(공학박사)
- 1993년 1월 ~ 2003년 2월 : 대우정보시스템 차장
- 2003년 3월 ~ 현재 : 배재대학교 전자상거래학과 부교수

<관심분야> : XML, 지능정보시스템, 데이터베이스시스템, 전자상거래학

박 준 호(Junho Park) 정회원

- 2008년 2월 : 충북대학교 정보통신공학과(공학사)
- 2010년 2월 : 충북대학교 정보통신공학과(공학석사)
- 2010년 3월 ~ 현재 : 충북대학교 정보통신공학과(박사과정)

<관심분야> : 분산 데이터베이스 시스템, 센서 네트워크, RFID, 차세대 웹, U-Leaning(LMS, LCMS)

유 재 수(Jaesoo Yoo) 종신회원

- 1989년 : 전북대학교 컴퓨터공학과(공학사)
- 1991년 : 한국과학기술원 전산학과(공학석사)
- 1995년 : 한국과학기술원 전산학과(공학박사)
- 1995년 ~ 1996년 8월 : 목포대학교 전산통계학과 전임강사
- 1996년 8월 ~ 현재 : 충북대학교 전기전자컴퓨터공학부 교수

<관심분야> : 데이터베이스 시스템, XML, 멀티미디어 데이터베이스, 분산 객체 컴퓨팅 등