ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템 설계 및 구현☆

Design and Implementation of MongoDB-based Unstructured Log Processing System over Cloud Computing Environment

김 명 진[1] 한 승 호[1] 최 운[1] 이 한 구[1*]
Myoungjin Kim Seungho Han Yun Cui Hanku Lee

요 약

컴퓨터 시스템 운용 간에 발생하는 많은 정보들이 기록되는 로그데이터는 컴퓨터 시스템 운용 점검, 프로세스의 최적화, 사용자 최적화 맞춤형 제공 등 다방면으로 활용되고 있다. 본 논문에서는 다양한 종류의 로그데이터들 중에서 은행에서 발생하는 대용량의 로그데이터를 처리하기 위한 클라우드 환경 하에서의 MongoDB 기반 비정형 로그 처리시스템을 제안한다. 은행업무간 발생하는 대부분의 로그데이터는 고객의 업무처리 프로세스 간에 발생하며, 고객 업무 프로세스 처리에 따른 로그데이터를 수집, 저장, 분류, 분석하기 위해서는 별도로 로그데이터를 처리하는 시스템을 구축해야만 한다. 하지만 기존 컴퓨팅환경 하에서는 폭발적으로 증가하는 대용량 비정형 로그데이터 처리를 위한 유연한 스토리지 확장성 기능, 저장된 비정형 로그데이터를 분류, 분석 처리할 수 있는 기능을 구현하기가 매우 어렵다. 이에 따라 본 논문에서는 클라우드 컴퓨팅 기술을 도입하여 기존 컴퓨팅 인프라 환경의 분석 도구 및 관리체계에서 처리하기 어려웠던 비정형 로그데이터를 처리하기 위한 클라우드 환경기반의 로그데이터 처리시스템을 제안하고 구현하였다. 제안한 본 시스템은 IaaS(Infrastructure as a Service) 클라우드 환경을 도입하여 컴퓨팅 자원의 유연한 확장성을 제공하며 실제로, 로그데이터가 장기간 축적되거나 급격하게 증가하는 상황에서 스토리지, 메모리 등의 자원을 신속성 있고 유연하게 확장을 할 수 있는 기능을 포함한다. 또한, 축적된 비정형 로그데이터의 실시간 분석이 요구되어질 때 기존의 분석도구의 처리한계를 극복하기 위해 본 시스템은 하둡 (Hadoop) 기반의 분석모듈을 도입함으로써 대용량의 로그데이터를 빠르고 신뢰성 있게 병렬 분산 처리할 수 있는 기능을 제공한다. 게다가, HDFS(Hadoop Distributed File System)을 도입함으로써 축적된 로그데이터를 블록단위로 복제본을 생성하여 저장관리하기 때문에 본 시스템은 시스템 장애와 같은 상황에서 시스템이 멈추지 않고 작동할 수 있는 자동복구 기능을 제공한다. 마지막으로, 본 시스템은 NoSQL 기반의 MongoDB를 이용하여 분산 데이터베이스를 구축함으로써 효율적으로 비정형로그데이터를 처리하는 기능을 제공한다. MySQL과 같은 관계형 데이터베이스는 복잡한 스키마 구조를 가지고 있기 때문에 비정형 로그데이터를 처리하기에 적합하지 않은 구조를 가지고 있다. 또한, 관계형 데이터베이스의 엄격한 스키마 구조는 장기간 데이터가 축적되거나, 데이터가 급격하게 증가할 때 저장된 데이터를 분할하여 여러 노드에 분산시키는 노드 확장이 어렵다는 문제점을 가지고 있다. NoSQL은 관계형 데이터베이스에서 제공하는 복잡한 연산을 지원하지는 않지만 데이터가 빠르게 증가할 때 노드 분산을 통한 데이터베이스 확장이 매우 용이하며 비정형 데이터를 처리하는데 매우 적합한 구조를 가지고 있는 비관계형 데이터베이스이다. NoSQL의 데이터 모델은 주로 키-값(Key-Value), 컬럼지향(Column-oriented), 문서지향(Document-Oriented)형태로 구분되며, 제안한 시스템은 스키마 구조가 자유로운 문서지향(Document-Oriented) 데이터 모델의 대표 격인 MongoDB를 도입하였다. 본 시스템에 MongoDB를 도입한 이유는 유연한 스키마 구조에 따른 비정형 로그데이터 처리의 용이성뿐만 아니라, 급격한 데이터 증가에 따른 유연한 노드 확장, 스토리지 확장을 자동적으로 수행하는 오토샤딩 (AutoSharding) 기능을 제공하기 때문이다. 본 논문에서 제안하는 시스템은 크게 로그 수집기 모듈, 로그 그래프생성 모듈, MongoDB 모듈, Hadoop기반 분석 모듈, MySQL 모듈로 구성되어져 있다. 로그 수집기 모듈은 각 은행에서 고객의 업무 프로세스 시작부터 종료 시점까지 발생하는 로그데이터가 클라우드 서버로 전송될 때 로그데이터 종류에 따라 데이터를 수집하고 분류하여 MongoDB 모듈과 MySQL 모듈로 분배하는 기능을 수행한다. 로그 그래프생성 모듈은 수집된 로그데이터를 분석시점, 분석종류에 따라 MongoDB 모듈, Hadoop기반 분석 모듈, MySQL 모듈에 의해서 분석되어진 결과를 사용자에게 웹 인터페이스 형태로 제공하는 역할을 한다. 실시간적 로그데이터분석이 필요한 로그데이터는 MySQL 모듈로 저장이 되어 로그 그래프생성 모듈을 통하여 실시간 로그데이터 정보를 제공한다. 실시간 분석이 아닌 단위시간당 누적된 로그데이

[1] Department of Internet and Multimedia Engineering, Konkuk University, Seoul, 143-701, Korea
* Corresponding author (hlee@konkuk.ac.kr)
[Received 31 July 2013, Reviewed 9 August 2013, Accepted 4 October 2013]
☆ "본 연구는 미래창조과학부 및 정보통신산업진흥원의 대학IT 연구센터 육성·지원사업의 연구결과로 수행되었음"(NIPA-2013-H0301-13-3006)
☆ 본 논문은 2013년도 인터넷정보학회 춘계학술발표대회 우수 논문 추천에 따라 확장 및 수정된 논문임.

터의 경우 MongoDB 모듈에 저장이 되고, 다양한 분석사항에 따라 사용자에게 그래프화해서 제공된다. MongoDB 모듈에 누적된 로그 데이터는 Hadoop기반 분석모듈을 통해서 병렬 분산 처리 작업이 수행된다. 성능 평가를 위하여 로그데이터 삽입, 쿼리 성능에 대해서 MySQL만을 적용한 로그데이터 처리시스템과 제안한 시스템을 비교 평가하였으며 그 성능의 우수성을 검증하였다. 또한, MongoDB의 청크 크기별 로그데이터 삽입 성능평가를 통해 최적화된 청크 크기를 확인하였다.

☞ 주제어 : 클라우드 컴퓨팅, NoSQL, MongoDB, 비정형 데이터, 은행처리 시스템

ABSTRACT

Log data, which record the multitude of information created when operating computer systems, are utilized in many processes, from carrying out computer system inspection and process optimization to providing customized user optimization. In this paper, we propose a MongoDB-based unstructured log processing system in a cloud environment for processing the massive amount of log data of banks. Most of the log data generated during banking operations come from handling a client's business. Therefore, in order to gather, store, categorize, and analyze the log data generated while processing the client's business, a separate log data processing system needs to be established. However, the realization of flexible storage expansion functions for processing a massive amount of unstructured log data and executing a considerable number of functions to categorize and analyze the stored unstructured log data is difficult in existing computer environments. Thus, in this study, we use cloud computing technology to realize a cloud-based log data processing system for processing unstructured log data that are difficult to process using the existing computing infrastructure's analysis tools and management system. The proposed system uses the IaaS (Infrastructure as a Service) cloud environment to provide a flexible expansion of computing resources and includes the ability to flexibly expand resources such as storage space and memory under conditions such as extended storage or rapid increase in log data. Moreover, to overcome the processing limits of the existing analysis tool when a real-time analysis of the aggregated unstructured log data is required, the proposed system includes a Hadoop-based analysis module for quick and reliable parallel-distributed processing of the massive amount of log data. Furthermore, because the HDFS (Hadoop Distributed File System) stores data by generating copies of the block units of the aggregated log data, the proposed system offers automatic restore functions for the system to continually operate after it recovers from a malfunction. Finally, by establishing a distributed database using the NoSQL-based Mongo DB, the proposed system provides methods of effectively processing unstructured log data. Relational databases such as the MySQL databases have complex schemas that are inappropriate for processing unstructured log data. Further, strict schemas like those of relational databases cannot expand nodes in the case wherein the stored data are distributed to various nodes when the amount of data rapidly increases. NoSQL does not provide the complex computations that relational databases may provide but can easily expand the database through node dispersion when the amount of data increases rapidly; it is a non-relational database with an appropriate structure for processing unstructured data. The data models of the NoSQL are usually classified as Key-Value, column-oriented, and document-oriented types. Of these, the representative document-oriented data model, MongoDB, which has a free schema structure, is used in the proposed system. MongoDB is introduced to the proposed system because it makes it easy to process unstructured log data through a flexible schema structure, facilitates flexible node expansion when the amount of data is rapidly increasing, and provides an Auto-Sharding function that automatically expands storage. The proposed system is composed of a log collector module, a log graph generator module, a MongoDB module, a Hadoop-based analysis module, and a MySQL module. When the log data generated over the entire client business process of each bank are sent to the cloud server, the log collector module collects and classifies data according to the type of log data and distributes it to the MongoDB module and the MySQL module. The log graph generator module generates the results of the log analysis of the MongoDB module, Hadoop-based analysis module, and the MySQL module per analysis time and type of the aggregated log data, and provides them to the user through a web interface. Log data that require a real-time log data analysis are stored in the MySQL module and provided real-time by the log graph generator module. The aggregated log data per unit time are stored in the MongoDB module and plotted in a graph according to the user's various analysis conditions. The aggregated log data in the MongoDB module are parallel-distributed and processed by the Hadoop-based analysis module. A comparative evaluation is carried out against a log data processing system that uses only MySQL for inserting log data and estimating query performance; this evaluation proves the proposed system's superiority. Moreover, an optimal chunk size is confirmed through the log data insert performance evaluation of MongoDB for various chunk sizes.

☞ keyword : Cloud Computing, NoSQL, MongoDB, Unstructured Data, Banking System

1. 서 론

다양한 분야에서 생성되는 로그데이터에는 컴퓨터 시스템 운용 간에 발생하는 많은 정보가 기록되며 축적된 로그데이터는 컴퓨터 시스템운용, 프로세스의 최적화, 사용자의 최적화 맞춤형 서비스 제공 등 다방면으로 활용되고 있다 [1][2]. 본 논문에서는 다양한 종류의 로그데이터 중 은행에서 발생하는 대용량의 로그데이터를 처리하

기 위하여 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템을 제안한다. 은행 업무에서 발생하는 대부분의 로그데이터는 고객의 업무 처리 프로세스 간에 발생하며, 고객 업무 프로세스 처리에 따른 로그데이터를 수집, 저장, 분류, 분석하기 위해서 별도로 로그데이터를 처리하는 시스템이 요구된다.

기존 컴퓨팅 환경을 이용하면 급격하게 증가하는 비정형 로그데이터 처리를 위한 스토리지의 유연한 확장이 어렵고, 장기간 축적되는 비정형 로그데이터를 분류, 분석 처리 기능을 제공하는 시스템을 구현하기가 매우 어렵다. 이에 본 논문에서는 클라우드 컴퓨팅 기술을 도입하여 기존 컴퓨팅 인프라 환경의 분석 도구와 관리 체계에서 처리하기 어려웠던 비정형 로그데이터를 처리하기 위한 클라우드 환경 기반의 로그데이터 처리시스템을 제안하고 구현하였다.

최근 클라우드 컴퓨팅의 등장으로 그동안 기존 컴퓨팅 환경에서 처리하기 어려웠던 대용량 '빅데이터' 처리, 분석 문제를 해결할 수 있게 되었다[4]. 특히 장기간 축적된 대용량 데이터와 실시간 단위로 빠르게 증가하는 데이터들이 클라우드 컴퓨팅 기술에 의하여 수집, 분류, 분석될 수 있기 때문에 다양한 분야에서 클라우드 컴퓨팅 기술이 적극적으로 연구되고 있다[5].

제안한 본 시스템은 IaaS(Infrastructure as a Service) 클라우드 환경을 도입하여 컴퓨팅 자원의 유연한 확장성을 제공하며 실제로, 로그데이터가 장기간 축적되거나 급격하게 증가하는 상황에서 스토리지, 메모리 등의 자원을 신속성 있고 유연하게 확장을 할 수 있는 기능을 포함한다. 또한, 축적된 비정형 로그데이터의 실시간 분석이 요구 될 때 기존 인프라 환경에서의 분석도구의 처리의 한계를 극복하기 위해 본 시스템은 하둡(Hadoop) 기반의 분석 모듈을 도입함으로써 대용량 로그데이터를 빠르고 신뢰성 있게 병렬 분산 처리할 수 있는 기능을 제공한다. 게다가, 본 시스템은 HDFS(Hadoop Distributed File System)을 도입함으로써 장기간 축적된 로그데이터를 블록단위로 복제본을 생성하여 저장 관리하여 시스템 장애와 같은 상황에서 시스템이 멈추지 않고 작동할 수 있도록 자동 복구 기능을 제공한다.

은행 업무 관련 로그처리 시스템 관점에서 제안한 본 시스템은 은행 업무 전반에서 발생하는 고객의 업무 대기시간, 업무 처리 시간 등 비정형 로그데이터를 수집, 분류, 분석하기 위하여 각 은행에서 클라우드 서버로 로그정보를 전송하며 웹 모니터링 시스템을 통하여 사용자에게 분석된 결과를 보여주는 기능을 가지고 있다. 비정형 데이터의 처리, 분석을 진행하기 위하여 시스템은 기존의 관계형 데이터베이스 대신 비관계형 데이터베이스인 MongoDB를 적용하였다. 지금까지 MySQL과 같은 관계형 데이터베이스를 이용하여 많은 데이터가 저장되고 관리되어왔다. MySQL와 같은 관계형 데이터베이스들은 미리 정의된 스키마를 가지고 정형화된 데이터들을 저장하였다. 하지만 엄격한 스키마 구조를 가지는 관계형 데이터베이스는 형식이 다른 스키마의 데이터베이스와 통합이 어려우며, 로그데이터와 같은 비정형 데이터 처리에 적합한 구조를 가지고 있지 않으며 노드의 분산을 통한 확장이 어렵다는 문제점 가지고 있다. 이러한 구조상의 한계를 극복하기 위해 최근 많은 연구자에 의해서 스키마 구조가 유연한 분산형 데이터베이스 연구가 활발히 연구되고 있다[3][6][7].

MongoDB는 문서지향(Document-oriented) 데이터 모델로 스키마가 자유롭기 때문에 비정형 데이터를 처리하는 시스템에 적용하기 용이한 구조로 가지고 있다. 로그데이터가 장기간 수집되어 축적되거나, 로그데이터가 급격하게 증가하여 스토리지의 저장 공간 초과에 따른 문제를 해결하는 방법으로 데이터를 분할하여 노드 분산을 통한 해결방법이 중요하다. 많은 데이터베이스에서 노드 분산을 통한 데이터 분할을 위하여 샤딩(Sharding) 기능을 제공하지만 MySQL과 같은 기존의 관계형 데이터베이스는 엄격한 스키마 구조를 가지고 있어 데이터 분할이 어렵고, 수동으로 노드를 추가하고 분산시켜야 하는 관리의 어려움이 있다. MongoDB는 오토샤딩(AutoSharding) 기능을 제공하여 사용자 또는 운영자가 손쉽게 데이터를 작은 청크 단위로 분할하여 여러 샤드로 분산하여 저장할 수 있는 관리의 용이성을 제공한다.

본 시스템은 로그 수집기 모듈, 로그 그래프 생성 모듈, MongoDB모듈, Hadoop기반 분석 모듈, MySQL 모듈로 구성된다. 로그 수집기 모듈은 은행 업무 전반에 발생하는 로그데이터 중 고객의 업무 프로세스 시작부터 종료 시점까지 발생하는 모든 로그데이터가 클라우드 서버로 전송될 때, 로그의 종류에 따라 MongoDB 모듈과 MySQL 모듈로 분배한다. 로그 그래프 생성 모듈은 수집된 로그의 분석 시점, 분석 종류에 따라 MongoDB 모듈, Hadoop 기반의 분석 모듈, MySQL 모듈에 의해서 분석된 결과를 사용자에게 웹 인터페이스 형태로 제공한다. 실시간적 로그데이터 분석이 필요한 로그데이터는MySQL 모듈에 저장이 되어 로그 그래프 생성 모듈을 통하여 실시간 로그데이터 정보를 제공한다. 실시간 분석이 아닌 단위 시간당 누적된 로그데이터는 MongoDB 모듈에 저장되

고 로그 그래프 생성 모듈을 통해 사용자에게 제공된다. MongoDB에 장기간 누적된 로그데이터는 Hadoop 기반의 분석 모듈을 통하여 병렬 분산 처리 되어 그래프 생성 모듈을 통하여 사용자에게 제공된다.

구현된 시스템의 우수성을 검증하기 위하여 본 논문에서는 MongoDB , MySQL이 각각 적용된 로그 처리시스템에 데이터 삽입(Insert)및 쿼리(Query) 처리 시간 측정을 통한 성능평가를 수행하였다. 또한, MongoDB의 청크(Chunk) 크기별 로그데이터 삽입 성능 평가를 통해 최적화된 청크(Chunk) 크기를 확인하였다.

본 논문의 구성은 다음과 같다. 2장 관련 연구에서는 관계형 데이터베이스 모델, 비관계형 데이터베이스 모델인 NoSQL과 본 시스템에 적용된 MongoDB에 대하여 기술한다. 3장에서는 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템 모델과 프로토타입을 기술하고 4장에서는 성능 평가를 수행하여 본 시스템의 우수성을 검증하고, 마지막으로 5장의에서는 논문의 결론과 차후 연구 과제에 관해 기술한다.

2. 관련 연구

NoSQL(Not only SQL)은 비관계형 데이터베이스이다. 미리 정의된 스키마를 가지고 있는 관계형 데이터베이스는 많은 분야에서 사용되어 왔지만 비정형 데이터를 처리함에 있어서 한계가 있다. NoSQL 데이터베이스는 유연한 스키마 구조와 데이터의 증가에 따른 노드 분산을 통하여 데이터베이스 구조를 쉽게 확장할 수 있기 때문에 많은 분산 시스템 개발자들에게 주목을 받고 있다[6][7][8][9][10].

본 절에서는 관계형 데이터베이스의 데이터 모델과 NoSQL(Not only SQL)의 데이터 모델을 비교 설명하고, 키-값(Key-Value), 컬럼지향(Column-oriented), 문서지향(Document-oriented) 특징에 대해 알아보고 특징을 대표하는 데이터베이스에 대해서 설명한다. 특히, 본 논문에서 활용한 MongoDB에 대한 일반적인 내용과 구조적 특징에 대해서도 설명한다.

2.1 NoSQL과 관계형 데이터베이스의 비교

원자성(Atomicity), 일관성(Consistency), 고립성(Isolation), 영구성(Durability)의 특성을 가지는 관계형 데이터베이스는 트랜잭션(Transaction) 기능을 통하여 데이터베이스의 완전성(integrity)을 유지한다. 또한 관계형 데이터베이스는 미리 정의된 스키마 구조를 가지고 있다. 그러므로 관계형 데이터베이스에 데이터를 저장할 때 정규화 과정을 거치게 되기 때문에 데이터의 중복성을 최소화를 할 수 있다. 이외에 관계형 데이터베이스는 조인(Join) 연산과 같은 복잡한 쿼리(Query) 연산을 할 수 있는 장점이 있다. 그러나 관계형 데이터베이스는 많은 분야에서 사용되고 있지만 관계형 데이터베이스의 미리 정의된 엄격한 스키마 구조로 데이터를 저장하려면 정형화 과정이 필요하기 때문에 비정형 데이터를 처리하는데 매우 비효율적인 구조를 가지고 있다. 또한 데이터가 빠르게 증가할 때 노드 분산을 통한 확장이 어렵다는 한계점을 가지고 있다[6].

NoSQL(Not only SQL)은 비관계형 데이터베이스로 기존 관계형 데이터베이스에서 제공하는 조인(Join) 연산과 같은 복잡한 연산을 지원하지 않지만 데이터가 빠르게 증가할 때 쉽게 노드 분산을 통한 확장의 용이성을 제공하며, 비정형 데이터를 처리하는데 적합한 구조를 가지고 있기 때문에 관계형 데이터베이스의 한계를 극복하기 위한 대안으로 주목받고 있다[6][7][8][9][10][11].

NoSQL(Not only SQL)은 키-값(Key-Value), 컬럼지향(Column-oriented) 모델, 문서지향(Document-oriented) 모델 등 다양한 데이터 모델들을 가지고 있다.

2.1.1 키-값(Key-Value) 문서지향 모델

키-값(Key-Value) 데이터 모델은 Key와 Value 형태로 데이터를 저장하는 단순한 구조로 Key를 이용하여 Value 값을 검색한다[5][6][7]. 키-값(Key-Value) 모델을 가지는 데이터베이스에는 Redis, Tokyo Cabinet, Tokyo Tyrant, Memcached 등이 있다.

2.1.2 컬럼지향(Column-oriented) 지향 모델

컬럼지향(Column-oriented) 데이터 모델은 테이블 구조를 가지고 있다. 컬림(Column) 단위로 데이터를 분리해서 데이터를 저장하기 때문에 데이터가 빠르게 증가해도 처리 속도가 빠르다는 장점을 가지고 있다[5][6][7]. 컬럼지향(Column-oriented) 데이터 모델을 가지는 데이터베이스로는 Cassandra, Hbase, Hyper표 등이 있다.

2.1.3 문서지향(Document-oriented) 지향 모델

문서지향(Document-Oriented) 데이터 모델은 스키마가 자유로운 문서(document) 저장 구조를 가지고 있기 때문

에 스키마를 미리 정의할 필요가 없다는 특징을 가지고 있다[5][6][7]. 문서지향(Document-Oriented) 데이터 모델은 JSON 또는 XML 포맷으로 데이터를 저장한다. 문서지향(Document-Oriented) 데이터 모델을 가지는 데이터베이스에는 Mongodb, CouchDB, SimpleDB 등이 있다.

2.2 MongoDB

MongoDB는 10gen에서 운영되어지고 있는 C++기반의 오픈소스 NoSQL(Not only SQL) 데이터베이스 프로젝트이다. MongoDB는 OS X, Linux, Windows, Solaris, Freebsd 등 다양한 운영체제를 지원한다[13][14].

MongoDB는 문서지향(Document-Oriented) 데이터 모델로 스키마가 자유로운 구조를 가지고 있기 때문에 로그 데이터와 같은 비정형 데이터를 처리하기에 적합하다. MongoDB에서 문서(Document)는 데이터 저장 구조의 기본단위이다. 문서는 숫자, 날짜, 배열, 내장 문서 등 다양한 자료형을 제공하고 키-값(Key-Value)구조로 구성되며 하나 이상 다수 키-값(Key-Value)의 집합으로 이루어진다. 문서는 BSON(Binary JSON)을 사용하여 JSON (JavaScript Object Notation)의 형태로 키-값(Key-Value)구조로 데이터를 저장한다. 컬렉션(Collection)은 문서의 모음으로 구조가 다른 문서들을 저장할 수 있으며 스키마가 자유롭다. 다양한 문서의 컬렉션(Collection)을 통합하여 데이터베이스(database)가 이루어진다.

MongoDB는 BSON(Binary JSON)를 이용하여 바이너리 데이터를 저장할 수 있고 시스템 자원의 낭비를 방지를 하기 위하여 BSON(Binary JSON)의 크기를 16MB로 제한한다. 또한 16MB 이상의 바이너리 데이터 저장소 기능을 위하여 서브 컬렉션(Collection)으로 GridFS 분산파일시스템을 제공한다. GridFS는 바이너리 데이터를 작은 청크 조각으로 나누어 분산 저장한다[13][14].

MongoDB는 시스템 장애에 대비하기 위하여 Master-Slave와 Replica-Set 두 가지 복제 정책을 제공한다. 두 복제정책을 통하여 MongoDB에서는 데이터 읽기작업을 분산함으로써 시스템의 안정성을 보장한다. Master-Slave 구조는 Master 노드에서 Slave 노드로 데이터를 백업하여 시스템의 신뢰성 및 성능을 향상시킨다. 그러나 Master- Slave 구조에서 Master 노드에 장애가 생기면 모든 Node를 사용하지 못한다는 단점이 있다. MongDB는 이런 단점을 보완하기 위하여 Replica-Set 복제방법을 제공한다. Replica-Set 복제방법은 Master 노드에서 장애가 발생하면 다른 Slave 노드에서 새로운 Master 노드를 선출하여 시스템의 신뢰성을 보장한다[13][14].

현존하는 많은 데이터베이스는 데이터가 급격히 증가할 때 여러 노드로 데이터를 확장하는 샤딩(Sharding) 기능을 제공한다. 그러나 대부분 데이터베이스에서 제공하는 샤딩(Sharding) 기능은 사용자 및 운영자는 수동으로 노드를 추가해야하고 분할해야 하는 등 관리상의 어려움이 가지고 있다. MongoDB에서는 오토샤딩(AutoSharding) 기능을 제공하여 사용자 및 운영자가 손쉽게 데이터를 작은 청크 단위로 분할하여 여러 샤드로 분산하고 저장할 수 있는 장점을 제공한다[13][14].

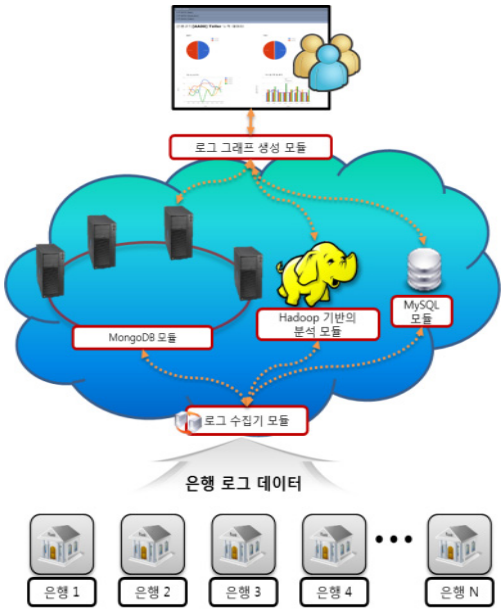
MongoDB는 Springer, CERN, The Guaardian, The New York Times, CNN Turk, Foursquare, Viber, Naver Japan, Github 등 다양한 시스템에서 사용되고 있다[14].

3. 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템

최근 대용량 '빅데이터'를 처리하는 위한 방법으로 클라우드 컴퓨팅 기술이 주목받고 있다. 본 장에서는 클라우드 컴퓨팅 환경 및 기술을 본 시스템에 도입한 배경 및 본 논문에서 제안한 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템의 아키텍처를 살펴본다. 또한, 본 시스템의 처리 대상인 로그데이터의 매개변수에 대하여 설명한 후 시스템의 프로토타입을 설명한다.

3.1 클라우드 컴퓨팅 시스템의 도입 배경

본 논문에서 제안한 시스템은 IaaS(Infrastructure as a Service) 클라우드 환경에서 시스템을 구축함으로써 기존 컴퓨팅 인프라에서 별도의 수집, 분류, 분석을 위한 시스템을 필요로 하지 않는다. 이외에 장기간 축적된 비정형 로그데이터를 저장하기 위한 스토리지 및 시스템 자원의 유연한 확장성을 제공함으로써 시스템의 자원 관리가 쉽다는 장점을 가지고 있다. 기존 컴퓨팅 인프라스트럭처의 분석체계로는 축적된 비정형 데이터에 대한 분석이 어렵다는 문제점이 있다. 클라우드 컴퓨팅 기술은 상기 문제를 해결하기 위한 최적의 방법으로 선택되고 있으며 '빅데이터' 처리 분야에서도 많은 주목을 받고 있다. 또한 클라우드 컴퓨팅 기술로 활용되고 있는 Hadoop 기반의 분산처리기술은 장기간 축적되어 처리하지 못했던 대용량 비정형 데이터를 처리하는 핵심 플랫폼으로 자리 잡고 있다. 그러므로 Hadoop기반의 분석 모듈을 도입한 본 시



(그림 1) 클라우드 컴퓨팅 환경에서 MongoDB 기반 비정형 로그 처리 시스템 전체 구조도

(Figure 1) The overall architecture of MongoDB-based unstructured log processing system over cloud computing environment

스템은 병렬, 분산처리방식으로 대용량의 비정형 로그데이터를 빠르게 분석한다. 또한 본 시스템은 HDFS(Hadoop Distributed File System)[15][16]을 사용함으로써 비정형 로그데이터는 NameNode에 의하여 블록 단위로 관리되고 복제본을 생성하여 분산, 저장 및 관리되기 때문에 시스템 장애와 같은 상황에서 제안한 시스템이 멈추지 않도록 복제본을 활용한 자동 복구 기능을 제공한다.

3.2 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템 아키텍처

그림 1은 본 논문에서 제안한 클라우드 환경에서 MongoDB 기반의 은행 비정형 로그 처리 시스템의 구조도를 보여주고 있다. 제안 시스템은 클라우드 환경에서 로그 수집기 모듈, 로그 그래프생성 모듈, MySQL 모듈, MongoDB 모듈, Hadoop기반 분석 모듈로 구성되었다.

3.2.1 로그 수집기 모듈

로그 수집기 모듈은 각 은행에서 고객의 업무 프로세스 시작부터 종료 시점까지 발생하는 로그데이터를 클라우드 서버에서 수집하는 역할을 한다. 로그 수집기 모듈은 각 은행에서 고객의 업무 프로세스 시작부터 종료 시점까지 발생하는 로그데이터가 클라우드 서버로 전송될 때 로그데이터 종류에 따라 데이터를 수집하고 분류하여 MongoDB 모듈과 MySQL 모듈로 분배하는 기능을 수행한다. 실시간 데이터 분석이 필요한 로그데이터는 MySQL 모듈로 전송하며, 단위 시간별로 분석이 필요한 로그데이터는 MongoDB 모듈로 전송된다. 특히 로그 수집기 모듈은 장기간 축적된 대용량 로그데이터를 Hadoop 기반의 분석 모듈로 전송하고, 로그 수집기 모듈로부터 전송된 로그데이터는 Hadoop 기반의 분석 모듈에 의하여 처리된다.

3.2.2 로그 그래프 생성 모듈

로그 그래프 생성 모듈은 웹 인터페이스를 이용하여 사용자에게 분석된 정보를 시각화하기 위해 사용된다. 로그 수집기 모듈에 의하여 수집된 로그데이터는 분석 시점, 로그데이터 종류에 따라 MongoDB 모듈, Hadoop기반의 분석 모듈, MySQL 모듈에 의하여 분석 및 처리되고, 로그 그래프 생성 모듈에 의하여 웹 인터페이스 형태로 사용자에게 분석된 내용이 제공된다.

3.2.3 MySQL 모듈

MySQL 모듈은 각 은행에서 은행원이 현재 업무처리를 진행하고 있는 프로세스의 번호와 업무 프로세스를 진행하기 위해 현재 대기하고 있는 인원수를 관리 하는 로그데이터를 저장하고 처리하는 역할을 수행한다. 즉 MySQL 모듈은 고객의 업무 프로세스 시작 시점부터 종료 시점까지 실시간으로 발생하는 로그데이터를 처리한다.

3.2.4 MongoDB 모듈

다중 노드로 구성된 MongoDB는 Relica-Set 정책을 적용하여 로그데이터의 복제본을 생성 및 저장하고, 로그데이터의 증가에 따라 자동으로 샤딩(Sharding)하고 분산하여 저장한다. 제안한 시스템에서 MongoDB 모듈은 은행 프로세스가 종료되면 MySQL 모듈에 저장된 로그데이터와 해당 로그데이터에 연관된 대기시간, 프로세스 처리시간, 프로세스 처리에 투입된 은행원의 번호 등의 정보들을 통합하여 저장한다. MongoDB 모듈은 전체 시스템에서 발생하는 모든 로그데이터를 저장, 관리한다. 또한 사

용자의 쿼리 요구에 따라 해당 로그데이터를 추출하여 Hadoop 기반의 분석 모듈에 전달하여 분석 작업을 진행하게 된다.

3.2.5 Hadoop 기반 분석 모듈

Hadoop 기반 분석 모듈은 MongoDB에 누적되어 있는 대량의 로그데이터를 사용자의 분석 요구에 따라 빠르고 신뢰성 있게 분산, 병렬처리하고 그 결과를 그래프 생성 모듈로 전송한다. 제안한 모듈에서 제공하는 MapReduce을 이용하여 사용자는 MongoDB에 저장된 로그데이터에서 사용자의 요구에 맞게 시간별, 날짜별, 월별, 연도별, 지점별 등 쿼리로 로그데이터를 추출하여 고객 은행업무 처리시간, 대기시간 등 통계 결과를 활용하기 위한 분석 작업을 진행할 수 있다. 이외에 Hadoop기반의 분석 모듈은 HDFS (Hadoop Distributed File System)을 이용하여 로그데이터를 블록단위로 복제본을 생성하여 저장한다. HDFS에 있는 NameNode는 생성된 복제본을 유기적으로 관리하여 시스템의 치명적인 오류 및 예기치 못한 시스템 손상이 있을 때 다른 노드에 저장된 복제본을 이용하여 자동으로 시스템 복구 작업을 진행한다. 이런 시스템적 특징에 의거해서 제안한 모듈은 전체 시스템의 안정성을 유지하고, 로그데이터의 안전한 처리 분석을 위한 신뢰성을 보장한다.

3.3 로그데이터 매개변수

본 논문에서 제안한 시스템은 고객의 업무 프로세스 단계 간에서 발생한 로그데이터의 수집, 분류, 분석을 목적으로 한다. 제안한 시스템의 각 모듈별 데이터 통신에서 사용되는 로그데이터 정보에 대한 정확성, 일치성을 확보하기 위하여 로그데이터 매개변수는 사전 정의되어야 한다. 표 1은 사전 정의된 로그데이터 파라미터를 보여주고 있다.

로그데이터의 매개변수는 bank_code, teller, job, number, generator_time, generator_wait_time, teller_start_time, teller_end_time으로 정의하였다. bank_code는 로그데이터가 생성된 은행의 번호를 알려주는 매개변수이고 teller는 현재 고객 업무 프로세스를 진행하고 있는 은행원의 번호를 지정하는 매개변수이다. job은 은행업무 종류를 구별하기 위하여 사용되는 매개변수로 은행업무의 종류에는 일반 업무 N과 기타업무 F가 정의되어 있다. number는 대기 순번 시스템으로부터 생성된 번호를 구분하기 위한 매개변수이다. 이상의 모든 매개변수에 의하여

(표 1) 로그데이터 파라미터 정의
(Table 1) The definition of log data parameter

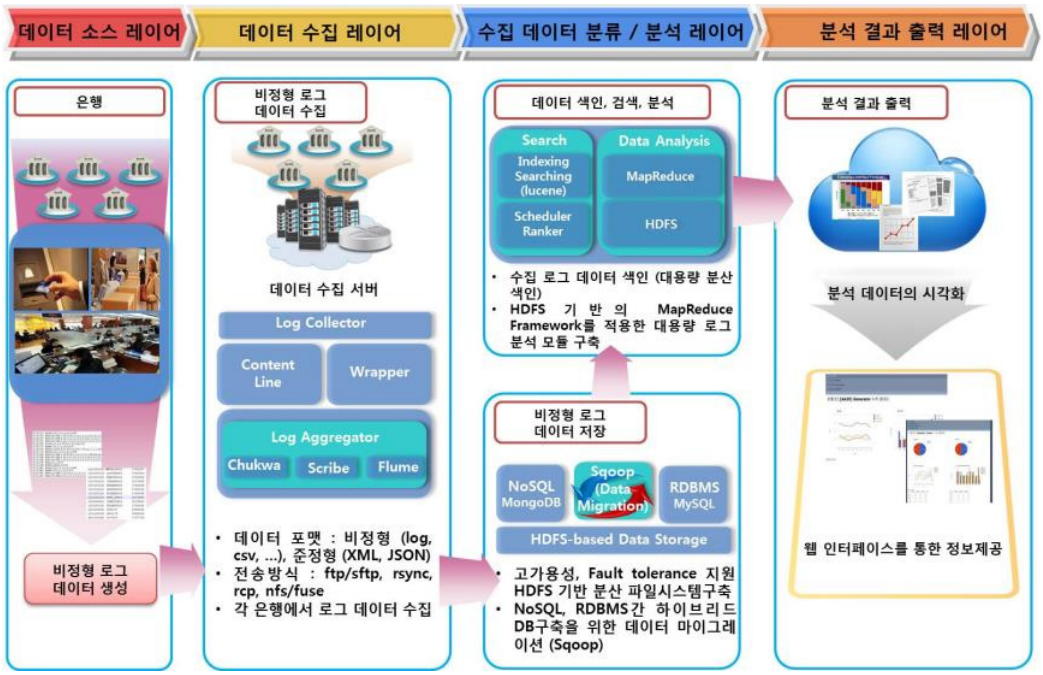
Type	Example
bank_code	AA01
teller	t001
job	N, F
number	43
generator_time	09:10
generator_wait_time	20
teller_start_time	09:30
teller_end_time	09:40

실시간으로 은행업무 프로세스가 진행 중인 로그데이터는 MySQL모듈에 저장되고, 은행 업무 프로세스가 종료된 로그데이터는 향후 분석처리로 사용하기 위하여 MongoDB모듈에 저장된다.

generator_time은 은행 업무를 진행하기 위해 생성된 대기 번호를 발생한 시간을 구분하여, generator_wait_time은 번호가 생성된 후 업무 프로세스가 시작되기 전까지의 시간을 표시하기 위해 정의한 매개변수이다. teller_start_time은 고객의 은행 업무 프로세스가 시작되는 시점의 시간을 기록하는 매개변수이며, teller_end_time은 고객의 은행 업무 프로세스가 종료된 시점의 시간을 기록하는 매개변수이다. generator_ time, generator_wait_time, teller_start_time과 teller_end_time의 매개변수를 활용한 로그데이터는 MongoDB 모듈에 저장되고 은행 업무 효율성 분석 자료로 활용하게 된다.

3.4 클라우드 환경에서 MongoDB 기반의 비정형 로그처리 서비스 모델 및 워크플로우

본 논문에서는 다양한 종류의 비정형 로그데이터들 중에서 은행에서 발생하는 대용량의 로그데이터를 처리하기 위한 클라우드 환경 하에서의 로그처리 시스템을 제안하였다. 본 시스템은 은행 업무 처리 프로세스 간에 발생하는 대용량 로그데이터의 수집부터 저장, 분류, 분석하는 기능을 포함하고 있다. 본 절에서는 구체적인 로그처리 서비스의 설명을 위해 MonogDB 기반의 비정형 로그처리 서비스 모델과 본 시스템에서의 주요 기능과 태스크를 포함하는 서비스모델 워크플로우를 설명한다.



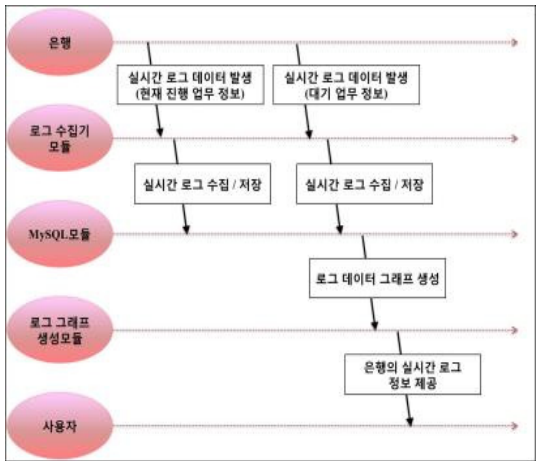
(그림 2) MongoDB 기반 비정형 로그 처리 서비스 모델
(Figure 2) MongoDB-based unstructured log processing service model

3.4.1 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리서비스 모델

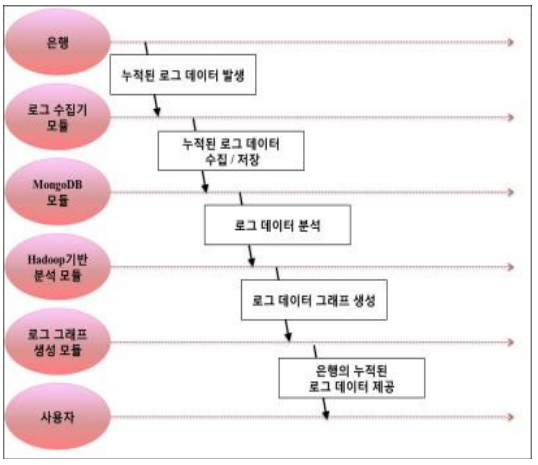
그림 2는 본 논문에서 제안하는 MongoDB 기반의 비정형 로그처리 서비스 모델을 보여준다. 제안 모델은 크게 데이터 소스 레이어, 데이터 수집 레이어, 수집 데이터 분류/ 분석 레이어, 분석 결과 출력 레이어로 구성되어 있다.

본 시스템 제안이전의 로그처리는 은행 지점별로 존재하는 로컬서버에서 처리 후 중앙 서버로 처리 결과를 전송하는 형태로 구성되어져 있다. 그러나 기존 컴퓨팅환경 하에서는 폭발적으로 증가하는 대용량 비정형 로그데이터 처리를 위한 유연한 스토리지 확장성 기능, 저장된 비정형 로그데이터를 분류, 분석 처리할 수 있는 기능을 구현하기가 매우 어려운 단점을 가지고 있다. 이러한 단점을 보완하기 위해 본 서비스 모델은 클라우드 컴퓨팅 기술과 다양한 오픈소스를 도입함으로써 컴퓨팅 자원의 유연한 확장성을 제공하며 실제로, 로그데이터가 장기간 축적되거나 급격하게 증가하는 상황에서 스토리지, 메모리 등의 자원을 신속성 있고 유연하게 확장을 할 수 있는 기능을 제공할 수 있다.

본 서비스 모델에서 로그처리의 대상은 무수한 은행지점에서 업무 프로세스 처리의 결과로 생성된 비정형 형태의 로그 데이터이다. 데이터 소스 레이어에서 발생된 로그데이터들은 해당지점의 로그처리 서버에 의해서 로그분석이 수행 되는 것이 아니고 클라우드 IaaS 환경에서 생성된 가상 머신이나 클라우드 중앙서버로 전송되어진다. 이때 확장성, 로그수집의 효율성을 유지하기 위해 데이터 수집 레이어에서는 Hadoop Eco System에서 제공하는 Chukwa, Scribe, Flume 등의 수집기 모듈을 이용해서 로그를 수집한다. 수집 데이터 분류 / 분석 레이어에서는 수집된 데이터는 HDFS (Hadoop Distributed File System)에 분산 저장되어지며 HDFS의 분산 저장 정책을 수용함으로써 데이터 손실, 시스템 실패 등 문제 발생 시 자동 복구 기능을 지원 받을 수 있다. 또한 저장된 비정형 데이터와 정형 데이터와의 동시 수집, 처리 시에 효율적인 데이터 이주를 위해 해당 레이어에서는 Sqoop를 이용하여 필요시에 MongoDB의 데이터를 MySQL로 MySQL의 데이터를 MongoDB로 손쉽게 데이터를 이주시킬 수 있다. 본 레이어에서의 가장 핵심적인 서비스는 Hadoop의 MapReduce기반의 비정형 로그분석 서비스이다. 저장된



(그림 3) 실시간 로그 데이터 처리 흐름도
(Figure 3) Diagram of real time log data processing workflow



(그림 4) 누적 로그 데이터 처리 흐름도
(Figure 4) Diagram of accumulated log data processing workflow

로그는 Hadoop의 MapReduce 기반의 분석 모듈을 통하여 대용량 데이터를 빠르고 신뢰성 있게 병렬 분산 처리 할 수 있는 환경을 제공한다. 개발자 및 로그처리 분석가는 단순히 맵리듀스 알고리즘에 맞게 분석 프로그램만 개발하고, 비정형 로그데이터의 입출력과 분산 병렬 처리 등 기반 작업은 MapReduce 프레임워크가 자동으로 처리해 주는 환경을 제공한다. 마지막 분석 결과 출력 레이어에서는 분석된 데이터를 시각화 하여 웹 인터페이스 환경을 통하여 사용자에게 분석된 로그 데이터 정보를 제공해 주는 역할을 한다.

3.4.2 실시간 로그 정보 제공을 위한 서비스모델 워크플로우

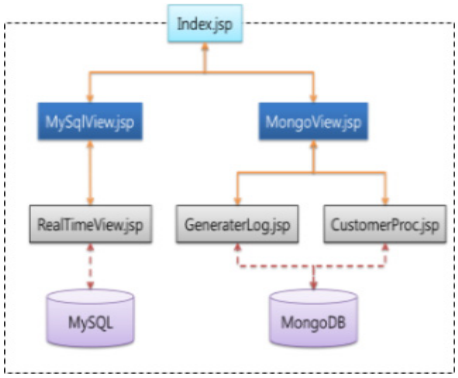
그림 3은 실시간 로그 정보 제공 프로세스 워크플로우 다이어그램을 보여준다. 은행에서 고객의 업무 프로세스가 시작되면 실시간 로그 데이터(현재 진행 업무 정보, 대기 업무 정보)가 발생한다. 발생된 로그 데이터는 로그 수집기 모듈을 통하여 수집되어 사용자에게 정보를 제공하기 위하여 MySQL 모듈에 저장이 된다. 저장된 실시간 로그 정보는 로그 그래프 생성 모듈을 통하여 그래프로 웹 인터페이스 환경에서 사용자에게 현재 은행의 실시간 로그 정보를 제공한다.

3.4.3 누적된 로그 정보 제공을 위한 서비스모델 워크플로우

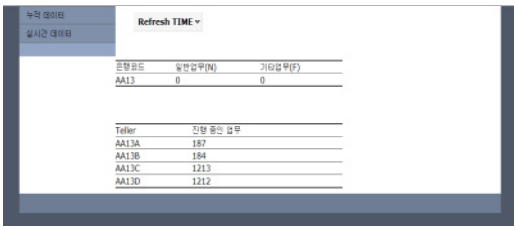
그림 4는 고객의 업무 프로세스 종료 후 생성된 누적된 로그 데이터 정보 제공 프로세스 워크플로우 다이어그램을 보여준다. 고객의 업무 프로세스가 종료되면 표 1의 매개변수 기반의 업무 간 누적된 로그 데이터가 발생한다. 발생된 로그 데이터는 로그 수집기 모듈을 통하여 수집된 로그 데이터는 정형화 과정을 거치지 않고 문서(Document)의 형태로 MongoDB에 저장된다. 저장된 누적 로그 데이터는 사용자의 요구조건 (시간별, 날짜별, 월별, 연도별, 지점별)에 따라 Hadoop기반의 분석 모듈을 통하여 빠르게 병렬 분석 처리되어 로그 그래프 생성 모듈을 통하여 웹 인터페이스 환경에서 사용자에게 누적된 로그 데이터 정보를 제공한다.

3.5 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템 구현

본 시스템은 Ubuntu Linux 10.04 LTS 64bit 환경에서 구현되었다. 로그 수집기 모듈은 Java로 구현되었으며, 로그 그래프 생성 모듈은 Tomcat7 환경에서 JSP와 Google Chart API를 이용하여 구현되었다. 실시간 데이터 처리를 위해서 MySQL을 단일노드위에 설치하였으며 사용된 버전은 5.1이다. 누적된 정보를 처리를 위하여 MongoDB는 4노드 환경위에 설치되어졌다. MongoDB와 각 모듈을 연결하기 위하여 10gen Mongo Driver가 사용되었고, MySQL과 각 모듈을 연결하기 위하여 Oracle MySQL Driver가 사용되었다. 은행에서 업무 프로세스 과정을 시뮬레이션하기 위하여 은행코드 생성, 업무처리 은행원 코



(그림 5) 사용자 웹 인터페이스 구성도
(Figure 5) The structure of web interface

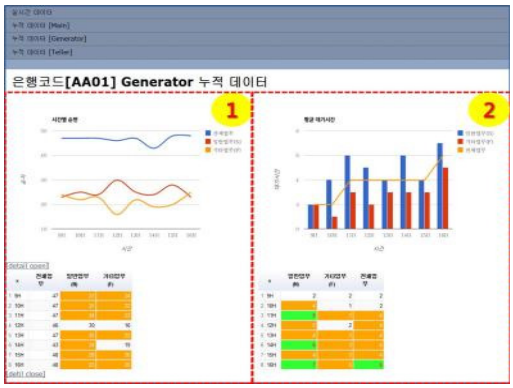


(그림 6) 실시간 로그 정보 제공을 위한 웹 기반 데쉬보드
(Figure 6) Web-based dashboard for real time log information

드번호 생성, 대기 순번 생성 등의 시뮬레이션 기능을 Java를 이용해서 구현하였다.

3.6 MongoDB 기반의 비정형 로그 처리 시스템 프로토타입

그림 5는 본 논문에서 제안한 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템에서 사용자에게 제공하는 웹 인터페이스의 구성도를 보여주고 있다. 그래프 생성 모듈은 RealTimeView.jsp을 이용하여 MySQL 모듈에 저장된 로그데이터 정보에 대한 그래프를 생성하고, 생성된 그래프를 MySqlView.jsp에 전달하여 웹 인터페이스로 사용자에게 보여준다. 또한 GeneraterLog.jsp를 이용하여 일정 시간당 대기번호 생성 개수와 업무처리를 위한 평균 대기시간에 대한 그래프를 생성하고, Customer Proc.jsp를 이용하여 은행원의 시간당 업무처리 개수, 업무처리에 소비한 평균시간 및 일간 은행원의 업무처리 효율에 대한 정보를 그래프로 표현한다. GeneraterLog.jsp와 CustomerProc.jsp에서 생성된 그래프는 MongoView.jsp



(그림 7) 대기순번 누적 로그 정보 제공을 위한 웹 기반 데쉬보드
(Figure 7) Web-based dashboard for accumulated log data information of waiting number

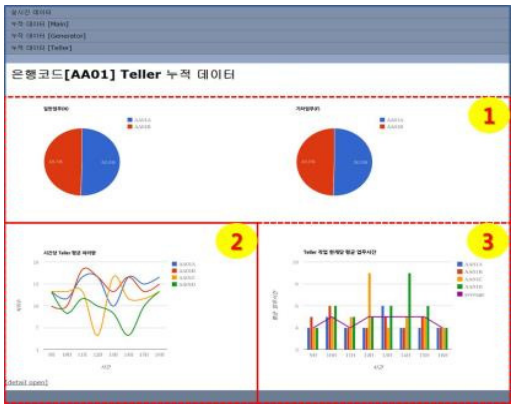
에 전달되어 웹 인터페이스로 사용자에게 표현된다.

3.6.1 실시간 로그 정보 제공 웹페이지

그림 6은 각 은행에서 실시간으로 고객의 업무 프로세스에 의하여 생성되는 실시간 로그 정보를 보여주고 있다. 사용자가 MySqlView.jsp를 접속하게 되면, 실시간 로그 정보 제공 모듈은 현재 MySQL에 저장되어 있는 실시간 정보를 추출하고 그래프 생성 모듈을 통하여 사용자가 선택한 은행의 현재 진행 중인 업무 번호, 대기 업무 숫자 등 실시간 정보를 제공하게 된다.

3.6.2 대기 순번 누적 로그 정보 제공 웹페이지

그림 7은 각 은행에서 고객의 업무 프로세스가 종료된 후 발생한 누적 로그데이터를 그래프로 표현한 웹페이지를 보여주고 있다. 사용자가 MongoView.jsp에 접속하여 사용자가 원하는 은행을 클릭하면 해당 은행에서 일정 시간당 생성되는 대기번호 개수와 은행업무처리를 위한 고객의 평균 대기시간 정보를 시각화된 그래프로 확인할 수 있다. 그림7 ①은 특정은행에서 은행 업무시간 범위(오전 9시부터 오후 4시까지)내에 일반 업무 및 기타업무를 처리하기 위해 생성된 대기번호 개수를 그래프와 표로 표현한 웹페이지를 보여주고 있다. 그림 7 ②는 특정 은행에서 은행 업무시간 범위(오전 9시부터 오후 4시까지)내에 일반 업무 및 기타업무를 처리하기 위해 기다리는 고객의 평균 대기시간을 그래프와 표로 표현한 결과를 보여주고 있다.



(그림 8) 단위 시간당 누적 로그 정보 제공을 위한 웹 기반 데쉬보드

(Figure 8) Web-based dashboard for accumulated log information per unit time

3.6.3 단위시간당 누적된 누적 로그 정보 제공 모듈

그림 8은 사용자가 CustomerProc.jsp에 접속하여 특정 은행에서 일일 은행원의 업무처리 개수, 시간당 은행원의 업무처리 개수와 은행원이 업무 하나당 처리에 소요되는 시간에 대한 수치를 반영하고 통계하여 개별 은행원의 은행업무 처리 효율 그래프 제공하는 웹페이지를 보여주고 있다. 그림 8 ①은 일반 업무와 기타업무를 처리하는 은행원의 업무처리 개수에 의한 비율을 표시한 그래프이다. 그림 8 ②는 시간당 은행원들의 업무처리량을 그래프로 표현하였으며 그림 8 ③은 시간당 은행원들이 배정된 한 개 은행 업무를 처리하는데 소모한 평균 업무 처리시간을 표현한 그래프이다.

4. 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템의 성능평가

본 장에서는 논문에서 구현한 클라우드 환경에서 MongoDB 기반의 비정형 로그 처리 시스템의 시뮬레이션에 대한 성능평가 결과에 대해서 기술한다. 은행에서 고객의 업무 프로세스를 시뮬레이션 하여 생성된 로그데이터를 바탕으로 MySQL을 적용한 로그데이터 처리 시스템과 제안한 MongoDB기반의 로그데이터 처리 시스템의 로그데이터 삽입과 쿼리 성능에 대한 비교 평가를 진행하였다. 또한 MongoDB 청크(Chunk) 크기별 로그데이터 삽입 성능 평가를 통해 최적화된 로그데이터의 청크(Chunk) 크기를 확인하였다.

4.1 성능평가 환경

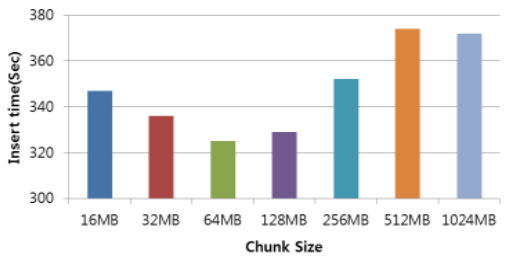
성능 평가의 환경은 Ubuntu Linux 10.04 LTS 64bit 환경에서 대표적인 관계형 데이터베이스 MySQL 5.1 버전을 적용한 로그 처리 시스템과 MongoDB 2.4.5버전을 적용한 비정형 로그 처리 시스템의 성능을 비교, 평가하였다. 성능평가 툴은 Java로 작성하였고, 로그데이터 생성 시뮬레이션, 로그데이터 삽입 및 쿼리, 최적화된 청크(Chunk)크기를 추출하는 기능에 대한 비교 평가를 진행하였다.

4.1.1. 실험을 위한 데이터 셋

삽입 성능평가를 위하여 사용된 데이터 셋은 클라우드 환경에서 MongoDB기반의 로그 처리 시스템의 시뮬레이션을 통하여 얻어진 로그데이터 셋으로 MongoDB에 생성된 문서(Document) 한건 (208bytes)을 기준으로, 2452960건(500MB), 4905920건(1GB), 9811840건(2GB)의 3가지 종류가 사용되었다.

4.2 삽입, 쿼리 성능평가

표 2는 MongoDB 기반의 로그처리 시스템과 MySQL 기반의 로그 처리시스템의 로그데이터 삽입, 쿼리 처리 시간을 보여준다. 비정형 은행 로그데이터를 처리하는 데 있어 삽입 성능은 MySQL대비 MongoDB를 적용한 우리의 시스템이 2GB 데이터 셋을 처리하는데 최대 8배 빠른 성능을 보여주고 있다. 또한 쿼리 처리 성능평가를 위하여 생성된 로그 데이터에서 은행별, 지점별, 시간별 등에 대한 데이터를 추출할 수 있는 1000건의 쿼리처리문 성능평가 결과 MySQL 대비 3만 배 정도의 빠른 처리 성능을 보여주고 있다. 본 논문에서 제안된 시스템이 MySQL 대비 수준 높은 성능향상을 보이는 이유는 다음과 같다. 로그데이터와 같은 비정형 데이터를 처리할 때 스키마가 자유로운 NoSQL MongoDB를 본 시스템에 적용하였기 때문에 기존 로그 데이터 처리 시 필요했던 정형화 단계를 거치지 않고 데이터를 저장하고 처리함으로써 기존 MySQL 기반의 로그 처리 시스템보다 로그 데이터를 처리하는 기능에서의 성능개선을 가져왔다. 또한, MongoDB는 Memory Mapping 기술을 사용하기 때문에 쿼리 속도가 매우 빠르다. 또한 쓰기 연산을 할 때 우선 메모리에 저장한 다음 파일시스템에 저장하는 구조를 가지고 있기 때문에 쓰기 속도가 빠르다. 마지막으로,



(그림 9) 청크 크기에 따른 삽입 성능평가 결과
(Figure 9) The result of Insert performance versus chunk size

MongoDB는 MySQL과 달리 트랜잭션 절차를 수행하지 않기 때문에 MySQL 대비 성능 처리 속도가 빠르다.

(표 2) 성능 평가 결과
(Table 2) The result of performance evaluation

DataSet	MongoDB		MySQL	
	Insert	Query	Insert	Query
500MB	181sec	0.03sec	829sec	230sec
1GB	301sec	0.04sec	1617sec	459sec
2GB	608sec	0.04sec	4855sec	1377sec

4.3 청크(Chunk) 크기에 따른 삽입 성능평가

본 논문에서 사용된 MongoDB에서는 분산 DB 수행을 위해 입력되어진 원본데이터를 고정된 크기의 청크 (기본 64MB)로 분할해서 구성된 노드에 분산 저장한다. 청크(Chunk) 크기는 사용자에 의해서 변경 가능하며, 데이터의 크기, 종류, 시스템 토폴로지 형태 등에 따라서 청크 사이즈 변경을 통해 시스템 성능향상을 가지고 올 수 있다. 이에 제안한 MongoDB 기반의 비정형 은행 로그 데이터 시스템의 최적화된 청크 사이즈를 확인하기 위하여 7개의 청크(Chunk) 크기에 따른 성능평가를 실시했다. 성능평가에 사용된 데이터 셋은 1GB (로그데이터 4905920건) 데이터 셋을 사용하였으며, 평가방식은 데이터 셋의 삽입시간에 대한 수행 시간 측정을 통해서 성능평가를 수행하였다.

그림 9는 청크(Chunk) 크기별 로그데이터 삽입 처리 시간을 보여준다. 성능평가 결과, 청크 크기가 16MB 일 때 347초, 32M 336초, 64M 325초, 128MB 329초, 256MB 352초, 512MB 374초, 1024MB 372초의 처리속도를 보여주고 있다. 성능평가 결과에 의하면 MongoDB의 청크(Chunk) 크기가 64MB일 때 가장 빠른 로그데이터 삽입처리 성능 효율을 보여주었고, 청크(Chunk) 크기가 증가할수록 실제로 로그 데이터 삽입 처리 시간이 증가한 것을 확인하였다. 또한 반대로 64MB 보다 청크(Chunk)크기가 작아질수록 로그데이터 삽입 처리 시간 또한 증가한 것을 확인하였다. MongoDB는 각 분산노드에서 오토샤딩(Sharding)을 통하여 로그데이터의 부하분산 하여 각 노드의 시스템 자원을 효율적으로 사용하는 기능을 제공한다. 오토샤딩 과정에서 기본단위로 청크(Chunk)를 이용하여 로그데이터를 분산시킨다. 이때 청크의 크기가 증가함에 따라 로그데이터를 전송하는데 소비되는 시간이 길어지고 시스템 자원이 소모되며 네트워크 트래픽이 증가된다. 또한 청크의 크기가 작아지면 자동 데이터 분산 작업에서 발생되는 관리비용이 커지게 된다. 성능평가 결과에 의하면 은행데이터와 같은 비정형 로그데이터 처리시의 최적화된 청크크기가 64MB로 확인되어졌다.

5. 결론 및 향후 연구 방안

본 논문에서는 MySQL과 같은 관계형 데이터베이스를 통하여 비정형 은행 로그데이터 처리에 한계성을 확인하였고, 이러한 한계점을 극복하기 위하여 클라우드 환경에서 MongoDB를 활용한 비정형 로그 처리 시스템을 제안하고 구현하였다. 성능평가영역에서 제안한 시스템은 MySQL 기반 로그처리 시스템 대비 삽입성능 최대 8배, 쿼리 성능 최대 3만 배의 성능을 보여주었다. 또한 일정 시간당 급격히 증가되는 비정형 은행 로그데이터를 MongoDB에 삽입할 때 최적화된 성능을 낼 수 있는 청크(Chunk) 크기를 찾기 위한 실험 결과, 64MB가 가장 최적화된 크기라는 것을 검증하였다. 현재 제안된 시스템은 업무 프로세스에서 발생하는 로그데이터를 대상으로 수집, 분류, 분석하는 서비스를 제공한다. 향후 연구 목표로는 은행 업무에서 발생하는 다양한 전표 및 고객의 집중 방문시간 등에 대한 실시간 분석처리를 위하여 Hadoop 에코 시스템을 이용한 실시간 로그처리 시스템에 대한 연구를 진행할 것이다.

참 고 문 헌(Reference)

[1] Woogryul Jeon, Jeeyeon Kim, Youngsook Lee, Dongho Won, "Analysis of Threats and Countermeasures

on Mobile Smartphone," Journal of The Korea Society of Computer and Information, Vol. 16, No. 2, pp.153-163, 2011.

[2] Taebok Yoon, Seunghoon Lee, KwangHo Yoonm, Jee-Hyong Lee, "Design and Application of Multi Concept Keyword Model based on Web-using information", Review of Korean Society for internet Information, Vol. 10, No. 5, pp.95-1105, 2009. 10.

[3] U-Chang Park, "A Database Schema Integration Method Using XML Schema", Review of Korean Society for internet Information, Vol. 3, No. 2, pp.39-56, 2002. 04.

[4] Hyung-Woo Lee, "Android based Mobile Device Rooting Attack Detection and Malicious Application Event Monitoring," Review of Korean Society for Internet Information, Vol. 13, No. 1, pp.30-38, 2012.

[5] Jae-woo Park, Sung-tae Moon, Gi-Wook Son, In-Kyoung Kim, Kyoung-Soo Han, Eul-Gyu Im, Il-Gon Kim, "An Automatic Malware Classification System using String List and APIs," Journal of Security Engineering, Vol.8, No.5, pp.611-626, 2011.

[6] Neal Leavitt, "Will NoSQL Databases Live Up to Their Promise?", Computer, Vol. 43, No. 2, pp.12-14, 2010. 02.

[7] Jing Han, Haihong E., Guan Le, Jian Du, "Survey on NoSQL database", Pervasive Computing and Applications (ICPCA) 2011 6th International Conference on, pp.363-366, 2011. 10.

[8] Robin Hecht, Stefan Jablonski, "NoSQL evaluation: A use case oriented survey", Cloud and Service Computing(CSC) 2011 International Conference on, pp.336-341, 2011. 12.

[9] Jaroslav Pokorny, "NoSQL databases:a step to database scalability in web environment", iiWAS '11 Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services, pp.278-283, 2011. 12.

[10] Michael Stonebraker, "SQL databases v. NoSQL databases", Communications of the ACM , Vol. 53, No. 4, pp.10-11, 2010. 04.

[11] Zhu Wei-ping, Li Ming-Xin, Chen Huan, "Using MongoDB to implement textbook management system instead of MySQL", Communication Software and Networks (ICCSN) 2011 IEEE 3rd International Conference on, pp.303-305, 2011. 05.

[12] Santo Lombardo, Elisabetta Di Nitto, Danio Ardagna, "Issues in Handling complex Data Structures with NoSQL databases", Symbolic and Numeric Algorithms for Scientific Computing (SYNASC) 2012 14th International Symposium on, pp.443-448, 2012. 09.

[13] Kristina Chodorow, Michael Dirolf, "MongoDB : the definitive guide", 2010. 09.

[14] MongoDB , http://www.mongodb.org/

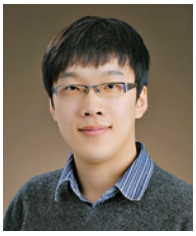
[15] Jeffry Dean, Sanjay Ghemawat, "MapReduce: simplified data processing on large clusters", Communications of the ACM - 50th anniversary issue: 1958 - 2008, Vol. 51, No .1, pp.107-113, 2008. 01.

[16] Konstatin Shavchko, Hairong Kuang, Sanjay Radia, Robert Chansler, "The Hadoop Distributed File System", Mass Storage Systems and Technologies (MSST) 2010 IEEE 26th Symposium on, pp.1-10, 2010. 05.

◐ 저 자 소 개 ◑

김 명 진(Myoungjin Kim)
2009년 건국대학교 일반대학원 컴퓨터공학과(공학석사)
2010년~2011년 건국대학교 소셜미디어 클라우드 연구센터 전문연구원
2013년 건국대학교 일반대학원 인터넷미디어공학과(박사수료)
관심분야 : 클라우드 컴퓨팅, 분산컴퓨팅, 멀티미디어 분산처리
E-mail : tough105@konkuk.ac.kr

한 승 호(Seungho Han)
2012년 배제대학교 컴퓨터공학과(공학학사)
2013년 건국대학교 일반대학원 인터넷미디어공학과 재학(공학석사)
관심분야 : 클라우드 컴퓨팅, 분산컴퓨팅, 멀티미디어 분산처리
E-mail : shhan87@konkuk.ac.kr



최 운(Yun Cui)
2008년 중국 동북사범대학 컴퓨터기술과학학과(공학학사)
2010년 건국대학교 일반대학원 컴퓨터공학과(공학석사)
2012년 건국대학교 일반대학원 인터넷미디어공학과(박사수료)
관심분야 : 홈 네트워크, 클라우드 컴퓨팅, 분산컴퓨팅
E-mail : ilycy@konkuk.ac.kr

이 한 구(Hanku Lee)
1993년 서강대학교 수학학과(이학학사)
1996년 미국 Iowa State University 컴퓨터과학학과(공학학사)
1998년 미국 Syracuse University 컴퓨터과학학과(공학석사)
2003년 미국 Florida State University 컴퓨터과학학과(공학박사)
2004년~현재 건국대학교 인터넷미디어공학과 교수
2007년~현재 KISTI 슈퍼컴퓨팅센터 자문위원
2007년~현재 한국정보과학회 컴퓨터시스템연구회 운영위원
2010년~현재 건국대학교 소셜미디어 클라우드 연구센터 센터장
관심분야 : 클라우드 컴퓨팅, 실시간 시스템, 분산컴퓨팅, 컴파일러
E-mail : hlee@konkuk.ac.kr