특집 07

빅데이터 기반 기술을 활용한 분산 처리 및 실시간 처리 방안

김병곤 (㈜클라우다인)

목 차 »

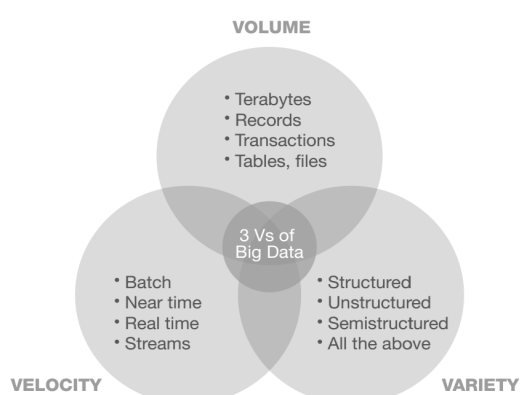
1. 서 론

2011년 하반기부터 IT 시장은 빅데이터라는 기술분야에 매우 심취해 있다고 해도 과언이 아니다. 어떤 이들은 버즈 워드라고 하며 어떤 이들은 차별화된 비즈니스를 위해서는 반드시 해야 한다고 한다. 하지만 그 안을 가만히 들어다 보면 단순히 좋다고, 해야 한다고 권하기에는 빅데이터는 매우 기술적이면서 높은 난이도를 가진 복합적인 분야라고 할 수 있다. 본 "빅데이터 기반 기술을 활용한 분산 처리 및 실시간 처리 방안"에서는 빅데이터의 기본적인 개념을 알아보고 빅데이터의 기반 기술을 통해 어떻게 시스템을 구현할 수 있는지 알아보고자 한다.

2. 빅데이터의 정의

빅데이터는 크기의 큼을 의미하는 단어인 "big"으로 인하여 일반적으로 대용량의 데이터를 처리하는 기술로만 이해하기 쉬우나 실제로는 다음의 (그림 1)과 같이 volume, velocity, variety와 같은 다양한 속성으로 정의를 해야만 빅데이터를 제대로 정의할 수 있다.

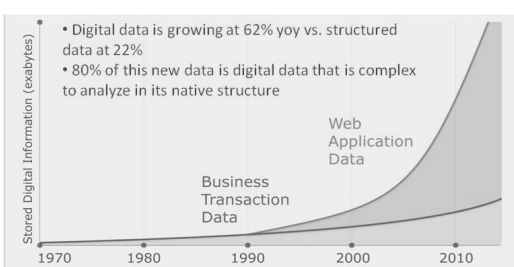
최근 다양한 디바이스와 서비스가 늘어남에 따라서 급격하게 서비스의 사용량이 증가하였고 이로 인하여 로그 파일의 크기가 급격하게 늘어나고 있다. 흔히 빅데이터에서 로그 파일의 크기는



(그림 1) 빅데이터의 세 가지 속성[1]

데이터베이스의 Giga Bytes 수준을 넘어 최소 Tera Bytes 이상을 의미하는데 국내 시장에서는 기껏 해봐야 수십 Giga Bytes 수준도 매우 큰 데이터로 인식하고 있는 실정이다. 국내 시장에서 빅데이터가 어려운 가장 큰 이유 중에 하나가 바로 데이터의 규모가 작다는 것에 있다고 볼 수 있다.

최근까지 데이터 크기의 증가 추세를 (그림 2)를 통해 확인해보면 비즈니스 트랜잭션 데이터는 일정한 수준으로 증가하였지만 애플리케이션 데이터는 최근 몇 년 동안 급격하게 늘어날 것을 확인할 수 있다. 시간이 지날수록 클라우드와 스마트 디바이스로 인하여 애플리케이션의 증가 추세는 더욱더 급격할 것이라고 누구나 충분히 예상할 수 있다. 과거에는 애플리케이션 데이터는 거의 쓸모 없는 데이터로 인식하여 모으지 않고 그대로 버리는 것이 당연시 되었고 더군다나 애플리케이션 데이터는 비정형이면서 크기가 매우 크기 때문에 높은 비용을 들여서 분석해야 할 대상이 아닌 것으로 인식되어 왔다. 하지만 최근 하드웨어의 가격이 급격하게 떨어지고, 성능은 매우 높아지면서 이러한 하드웨어를 활용하여 분석하고자 하는 요구가 발생하였고, 데이터를 처리하는 Apache Hadoop 이라는 오픈소스 소프트웨어가 출현하면서 이러한 요구사항을 구현할 수 있게 되어 빅데이터 시장은 급격하게 사람들의 관심을 끌게 되었다.



(그림 2) 데이터의 유형별 데이터의 증가 추이[2]

빅데이터는 애플리케이션 데이터처럼 크기뿐만 아니라 그 형식이 매우 다양한 특징도 있는데 일례로 애플리케이션 데이터가 그러하다. 시스템의 CPU, Memory, Network을 모니터링하는 NMON의 경우 실제로 파일에 기록하는 로그 파일의 구조는 상당히 복잡하면서 정형화 되어 있지 않은 구조를 가지고 있다. 다음은 NMON 로그의 일부분으로 특정 장비의 시스템 정보가 하나로 취합되어 있지 않고 다수의 라인으로 구성되어 상당히 처리하기 힘든 구조를 가지고 있다. 빅데이터는 기본적으로 이러한 비정형 데이터를 다루기 때문에 대부분의 작업이 문자열 처리 작업으로 구성되어 있다고 볼 수 있다.

```
TOP,+PID,Time,%CPU,%Usr,%Sys,Size,ResSet,Res
    Text,ResData,ShdLib,MinorFault,MajorFault,Co
    mmand
BBBP,000,/etc/release
BBBP,001,/etc/release,"CentOS release 5.7 (Final)"
BBBP,002,lsb_release
BBBP,003,lsb_release,"LSB Version:  :core-4.0-amd
    64:core-4.0-ia32:core-4.0-noarch:graphics-4.0-amd
    64:graphics-4.0-ia32:graphics-4.0-noarch:printing
    -4.0-amd64:printing-4.0-ia32:printing-4.0-noarch"
BBBP,004,lsb_release,"Distributor ID:  CentOS"
BBBP,005,lsb_release,"Description:  CentOS release
    5.7 (Final)"
BBBP,006,lsb_release,"Release:  5.7"
BBBP,007,lsb_release,"Codename:  Final"
```

빅데이터는 크기의 다양함, 구조의 다양함 이외에도 데이터의 생성되고 수집되고 처리되는 속도도 매우 상이한데 어떤 데이터는 실시간으로 수집되지만 모아서 분석해야만 가능한 데이터가 있는가 하면, 어떤 데이터는 실시간으로 즉시 분

석해서 사용할 수 있는 데이터도 있고 어떤 데이터는 대용량 배치 분석과 실시간 분석을 모두 해야만 제대로 된 결과를 얻을 수 있다. 예를 들면 위치 정보는 즉시 사용할 수도 있지만 모아서 분석하면 특정한 위치를 추정할 수 있다(예; 집의 위치, 회사의 위치).

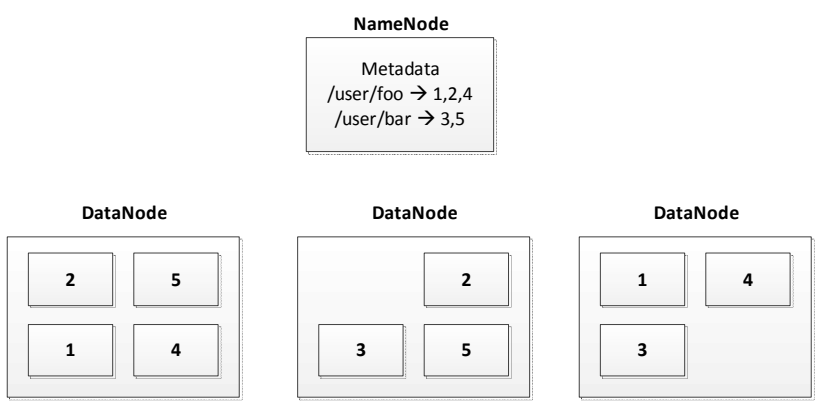
3. 대용량 데이터 처리 방법 설계

대용량 데이터를 처리하는 핵심은 Apache Hadoop으로써 Hadoop은 파일을 분산 저장하기 위한 Hadoop Distributed File System(HDFS)와 분산 되어 있는 파일을 처리하기 위한 MapReduce로 구성되어 있다. 데이터의 크기가 점점 증가하면서 기업은 데이터를 저장할 공간에 대한 비용을 줄이기 원했고 그 대안으로 NAS 보다는 파일을 분산 처리할 수 있는 Hadoop을 선택하게 되었다(일정양의 데이터를 단순 저장의 용도로는 NAS가 더 적합하다). Hadoop의 HDFS는 (그림 3)과 같이 하나의 파일을 여러 장비에 나누어서 저장한다.

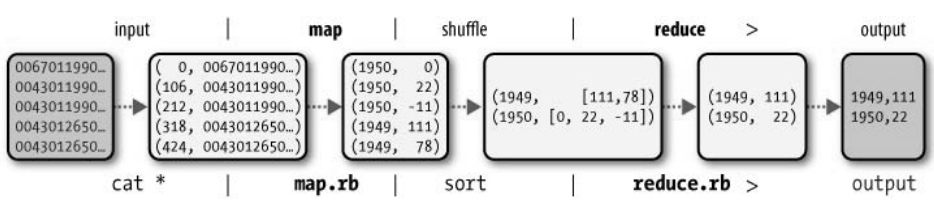
저장되어 있는 파일은 MapReduce 메커니즘에 따라서 데이터를 처리하는데 (그림 4)와 같이 입력 파일을 읽어서 key와 value 형식으로 map에서 처리한 후 reduce에서는 동일한 key로 value로 취합하여 처리하게 된다. MapReduce는 map과 reduce 간에 이러한 방식으로 데이터를 전달하며 원하는 결과를 얻게 된다.

분산 파일 시스템과 MapReduce를 결합하면 Apache Hadoop은 (그림 5)와 같이 입력 파일을 단위 크기(split)으로 나누어서 다수의 장비에서 map이 실행되고 그 결과를 다시 reduce에서 모아서 처리하는 방식으로 동작한다.

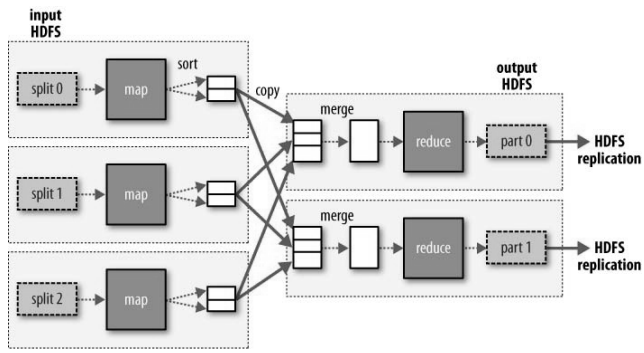
이러한 Apache Hadoop 매커니즘을 이용하여



(그림 3) Apache Hadoop 분산 파일 시스템



(그림 4) MapReduce 데이터 처리 흐름[3]



(그림 5) Hadoop MapReduce 데이터 처리 흐름[4]

〈표 1〉 처리하고자 하는 입출력 파일의 내용

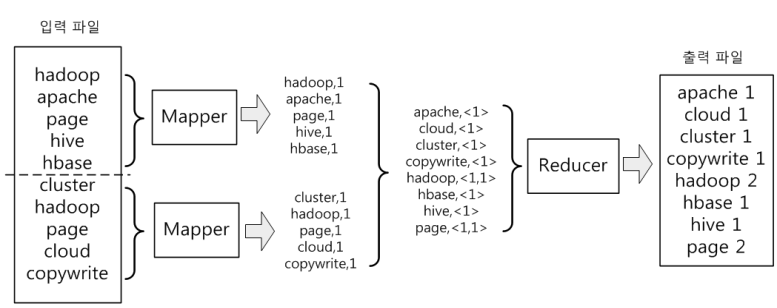
입력 파일	출력 파일
hadoop page hive hbase hadoop page copywrite	copywrite 1 hadoop 2 hbase 1 hive 1 page 2

네이버의 검색어 상위 10개를 추출하는 것을 대용량 기반으로 구현할 수 있다.

다음은 Apache Hadoop의 MapReduce로 로그 파일을 처리하는 과정을 설명한 데이터 처리 흐름 도식이다.

이러한 흐름을 가진 MapReduce 프로그램은 다음과 같이 작성할 수 있다. MapReduce는 기본으로 파일의 1개 라인을 읽어서 mapper로 넘겨주게 되므로 1개 라인에서 단어를 추출해서 취합하는 과정을 거치면 최종적으로 각 단어별 검색어 등수를 계산해 낼 수 있게 된다.

```
TOP,+PID,Time,%CPU,%Usr,%Sys,Size,ResSet,Res
    Text,ResData,ShdLib,MinorFault,MajorFault,Co
    mmand
BBBP,000,/etc/release
BBBP,001,/etc/release,"CentOS release 5.7 (Final)"
BBBP,002,lsb_release
BBBP,003,lsb_release,"LSB Version:  :core-4.0-amd
    64:core-4.0-ia32:core-4.0-noarch:graphics-4.0-amd
    64:graphics-4.0-ia32:graphics-4.0-noarch:printing
    -4.0-amd64:printing-4.0-ia32:printing-4.0-noarch"
BBBP,004,lsb_release,"Distributor ID:  CentOS"
BBBP,005,lsb_release,"Description:  CentOS release
    5.7 (Final)"
BBBP,006,lsb_release,"Release:  5.7"
BBBP,007,lsb_release,"Codename:  Final"
```



(그림 6) 단어의 개수를 합산하는 MapReduce 프로그램의 데이터 처리 흐름

최근 Apache Hadoop을 이용하여 데이터를 분석 및 가공하는 사례가 늘어나고 있으며 기존에 사용하던 분석 모듈을 MapReduce 기반으로 재개발하려는 움직임이 곳곳에서 감지되고 있다. 주의해야 할 것은 MapReduce가 기존과 처리 흐름에 있어서 많은 차별성을 보이고 대용량 데이터를 다루는 등 그 특성이 다양하기 때문에 그 부분을 충분히 고려해서 진행해야 한다는 점이다. 하지만 현재까지의 움직임은 전체 구축비용이 과거의 시스템 구축 비용보다 저렴하다는 이유로 전체 개발 일정과 비용까지도 저비용으로 진행하려는 움직임은 매우 우려할 만한 상황이다. 과거에도 그 비용이 많이 필요했다면 현재에서 그 비용은 여전히 많이 필요하다. 비용적인 문제도 중요하지만 무엇보다 데이터의 가치와 그것을 통해서 무엇을 얻고자 하는 것인지를 충분히 판단한 후에 적용한다면 좋은 결과를 얻을 것이라 기대한다.

4. 실시간 데이터 처리 방법 설계

빅데이터의 세 번째 속성인 velocity는 데이터가 생성되는 속도, 데이터의 수집 속도, 데이터의 전달 속도를 의미하며 이 분야는 전통적으로 Complex Event Processing(CEP) 및 Event Driven Architecture(EDA)라는 기술 영역으로 구분하고 있으나 최근 실시간 빅데이터의 처리를 위한 기반 기술로써 인정하는 추세이다.

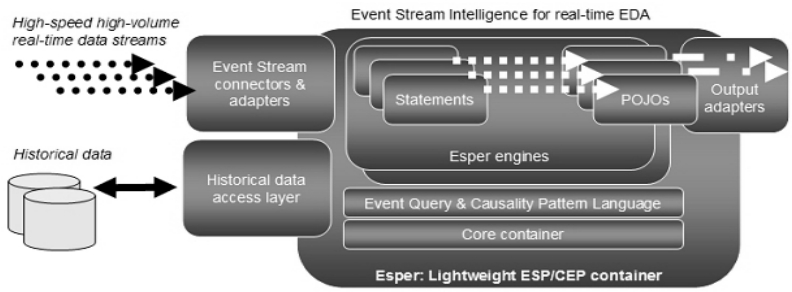
전통적으로 CEP 또는 EDA를 기반으로 한 애플리케이션은 prediction, observation, dissemination, behavior, active diagnostic등에 활용하며 금융권의 경우 증권사에서 트레이딩 시스템을 구현할 때 사용하고, 제조업의 경우 제조 공정에서 발생하는 정보를 수집할 때 사용한다. 기존의 CEP 또는 EDA와 빅데이터의 큰 차이점이라면 바로 대용량의 데이터를 통해 분석한 결과와 결합할 수 있다는 것이다. 기존의 이벤트 처리 시스템은 이벤트를 탐지하는데 사용하는 규칙의 품질이 빅데이터를 통해서 분석한 규칙보다 상대적으로 현저하게 떨어진다는 문제가 있으나 빅데이터는 대용량의 데이터를 통해서 매우 정교하고 다양한 규칙을 생성해 낼 수 있는 장점을 가지고 있어서 기존의 CEP 또는 EDA의 시스템을 더욱더 강력하게 만들 수 있게 된다.

빅데이터의 개념에서 실시간 처리와 배치 처리를 결합한 사례는 이미 해외 및 국내에서도 이제 충분히 발견할 수 있다. 사용자의 시스템 사용 정보 또는 이벤트 정보를 토대로 Apache Hadoop의 MapReduce를 이용하여 사용자의 유형 및 성향을 수십에서 수백 가지를 찾아내고(예; 30대, 강남에서 출퇴근, 미혼 여성, 급여 수준, 좋아하는 음식, 좋아하는 물건 등등) 실시간으로 수집되는 위치 정보 및 클릭 스트림을 사용자의 성향과 결합하여 실시간 개인화를 구현한다거나 광고를 하는 사례가 여기에 속한다.

CEP는 일반적으로 이벤트를 다양한 프로토콜을 통해서 수신하는 input adapter, 이벤트를 수신하여 특정 조건에 맞는 이벤트를 찾아내는 engine, 처리한 이벤트를 저장 또는 타 시스템으로 연계하기 위한 output adapter로 구성되어 있

〈표 2〉 CEP 및 EDA에서 사용하는 용어

용어	정의
이벤트	실제로 발생한 사건, 일, 메시지 상태의 변경 특정한 액션 또는 상태의 변화를 통해 발생하는 변경이 불가능한 과거의 기록
이벤트 스트림	시간의 순서대로 연속되는 이벤트의 흐름 시작과 끝이 없는 이벤트의 연속된 흐름
실시간의 특징	현저하게 낮은 수준의 지연 일정한 응답속도 예측 가능한 성능

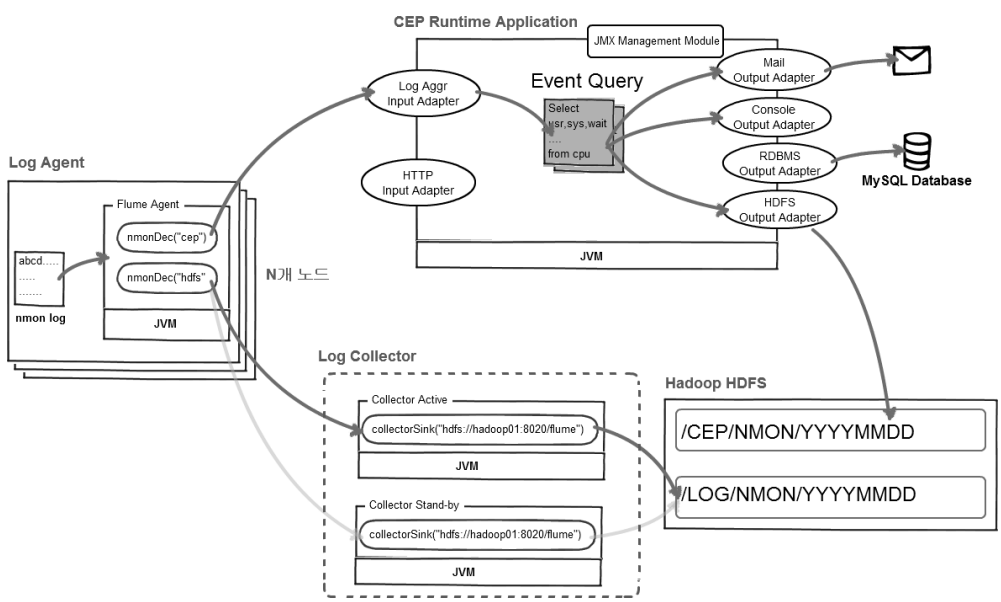


(그림 7) OpenSource Esper CEP의 아키텍처[5]

다. 다음은 OpenSource CEP인 Esper의 이벤트 처리 아키텍처이다.

CEP가 실시간 이벤트를 처리하는 데이터의 처리 속도에 중점을 둔다면 데이터의 생성 및 수집에 초점을 두는 log aggregator는 빅데이터 Platform의 일부분으로써 각 장비에서 발생하는 로그를 실시간으로 수집하여 하나로 취합하는 일을 한다. 과거에는 데이터를 수집하는 프로그램을 직접 작성하였으나 Cloud 및 빅데이터의 출현으로 많은 양의 장비에서 보다 안정적으로 데이터를 수집하기 위해서 별도의 로그 수집기를 사용하는 것이 일반적인 추세이다. 다음은 CEP, Log Aggregator, Hadoop을 이용하여 실시간으로 데이터를 수집하여 이벤트 처리 및 배치 처리를 하기 위한 시스템 구성도이다.

Log Aggregator는 원격의 장비에서 로그를 수집하는 agent와 agent가 보내온 로그를 수집하는 collector로 구성되어 있는 것이 일반적인 구성이며 agent는 비정형 데이터를 변환하는 기능, 로그의 유형에 따라서 collector를 선택하여 전송하는 기능, collector에 문제 발생시 failover하는 기능 등을 포함하고 있다. 위 그림에서는 시스템 상에



(그림 8) 로그 수집 및 실시간 이벤트 처리

서 발생하는 모니터링 정보(CPU, Memory, Network Usage)를 생성하는 nmon의 로그를 수집하여 실시간으로 CEP와 log collector에 전달하는 역할을 하고 있다. CEP는 log agent가 보낸 시스템의 이벤트를 수신하여 event query에 맞는 이벤트를 찾아서 email, RDBMS, 분산 파일 시스템 등에 기록하는 역할을 한다.

CEP가 동작하기 위해서는 가장 먼저 이벤트를 정의해야 한다. 예를 들어 주가의 tick 정보는 다음과 같이 표현할 수 있다.

이렇게 정의한 이벤트는 다음의 event query를 통해서 처리할 수 있다. 다음의 예제는 Apple 사의 tick이 평균 6이상, 2건 발생한 경우 실시간으로 수집하는 예제이다.

하지만 이 방법은 기존에도 전통적으로 쉽게 구현할 수 있었던 실시간 이벤트 처리 시스템이었다. 만약 빅데이터의 대용량 배치 처리와 실시간 처리를 결합한다면 “급여가 20M 이상이며 나이가 30~35세 이상이고 집이 강남이면서 취미가 쇼핑인 여성이 최근 20분내 시청 근처에 있었던 고객을 대상으로 실시간 광고를 실시”하겠다는 요구사항을 구현할 수 있게 된다. 여기에서 이 여성이 해당 조건에 맞는지 알아내는 것은 빅데이터의 배치 처리를 통해 데이터를 분석하여 추정 데이터를 생성했기 때문에 가능한 것이다.

빅데이터에 있어서 실시간 처리는 대용량의 데이터를 분석하여 나온 결과를 실시간 이벤트와 결합하여 매우 빠르고 적극적이며 지능적인 서비스를 구현할 수 있다는 점에서 아주 중요한 기술로 보고 있다. 하지만 높은 수준의 기술적 이해도, 높은 수준의 시스템 구축 능력, 데이터 분석 능력이 결합되어야만 가능한 것이 바로 실시간 빅데이터 처리 기술이기 때문에 국내에서는 아직 적극적으로 도입한 곳은 거의 없는 상황이다.

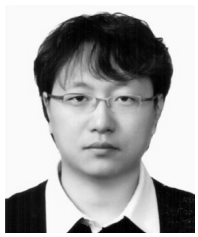
5. 결 론

지금까지 빅데이터의 기반 기술을 활용하여 어떻게 분산 처리를 하고 실시간으로 처리할 수 있는지 알아보았다. 빅데이터는 앞서 언급한 것처럼 높은 수준의 기술적 이해도와 시스템 구축 능력, 데이터 분석 능력이 결합되어야만 가능하다. 기술적으로 잘 이해하고 보다 지능적이면서 좋은 서비스 품질을 만들어 내기 위해서 데이터 분석가, 시스템 엔지니어, 개발자가 잘 조화를 이루어 기술과 비즈니스를 융합한다면 충분히 선도하는 기술 만들어 낼 수 있을 것이라 기대한다.

참 고 문 헌

[1] Philip Russom, Big Data Analytics, pp 6, TDWI Research, 2011.
[2] Ravi Kalakota, What is a “Hadoop”? Explaining Big Data to the C-Suite, http://practicalanalytics.wordpress.com
[3] Tom White, Hadoop Definitive Guide, pp 20, Oreilly, 2011.
[4] Tom White, Hadoop Definitive Guide, pp 30, Oreilly, 2011.
[5] Esper Tech, Esper Architecture, http://www.espertech.com/images/products_esper.jpeg

저 자 약 력



김 병 곤

이메일 : fharenheit@gmail.com

- 2002년 서경대학교 컴퓨터공학과 (학사)
- 2001년 ROCOZEN 객체기술연구소 연구원
- 2005년 KBTech 신호제어연구소 연구원
- 2007년 NSYSTEM 과장
- 2009년~현재 ISO 15504 (SPICE) 심사위원
- 2010년~현재 한국자바개발자협의회 6대 회장
- 2011년~현재 ㈜클라우다인 대표이사
- 2011년~현재 스마트개발자협회 부회장
- 2011년 제12회 SW산업인의날 지식경제부 장관 표창
- 2012년~현재 IT전문가협회 정회원
- 전문분야 : Big Data, OpenSource Infrastructure, Platform, Application Profiling, Autonomous Distributed System
- 집필도서 : JBoss Application Server 5, Enterprise JavaBeans 3, Enterprise JavaBeans 2