Apache Hadoop 기반
대용량 데이터 분석 시스템 구축

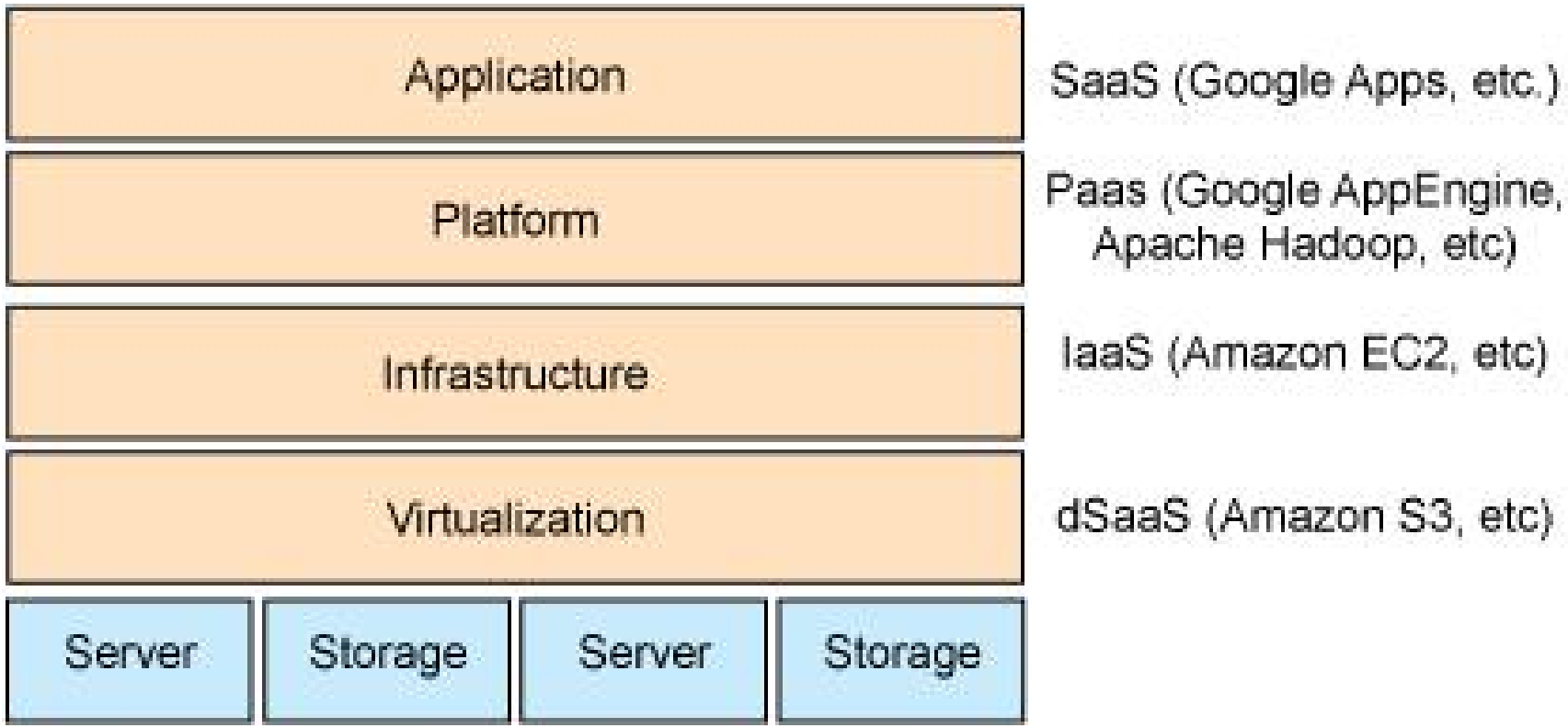
김병곤

fharenheit@gmail.com

소개

- 한국자바개발자협의회(JCO) 회장
- JBoss User Group 대표
- 한국스마트개발자협회 부회장
- 지경부/NIPA 소프트웨어 마에스트로 멘토
- 대용량 분산 컴퓨팅 Technical Architect
- 오프라인 Hadoop 교육 및 온라인 Java EE 교육
- 오픈 소스 Open Flamingo 설립(http://www.openflamingo.org)
- Java Application Performance Tuning 전문가
- 다수 책 집필 및 번역
  - JBoss Application Server5, EJB 2/3
  - Oreilly RESTful Java 번역 중

Cloud Computing과 Apache Hadoop



대용량 데이터의 세계

- 우리는 대용량 데이터 속에서 살고 있다!
  - 뉴욕증권거래소 : 1일에 1테라 바이트의 거래 데이터생성
  - Facebook : 100억장의 사진, 수 페타바이트의 스토리지
  - 통신사 : 시간당 10G 이상의 통화 데이터, 1일240G 생성, 월 생성 데이터의 크기 200T 이상

우리는 엄청나게 큰 로그 데이터를 어떻게 처리해야 할까?

Hadoop의 패러다임의 전환

로직이 데이터에 접근하지 말고

데이터가 있는 곳에 로직을 옮겨라!

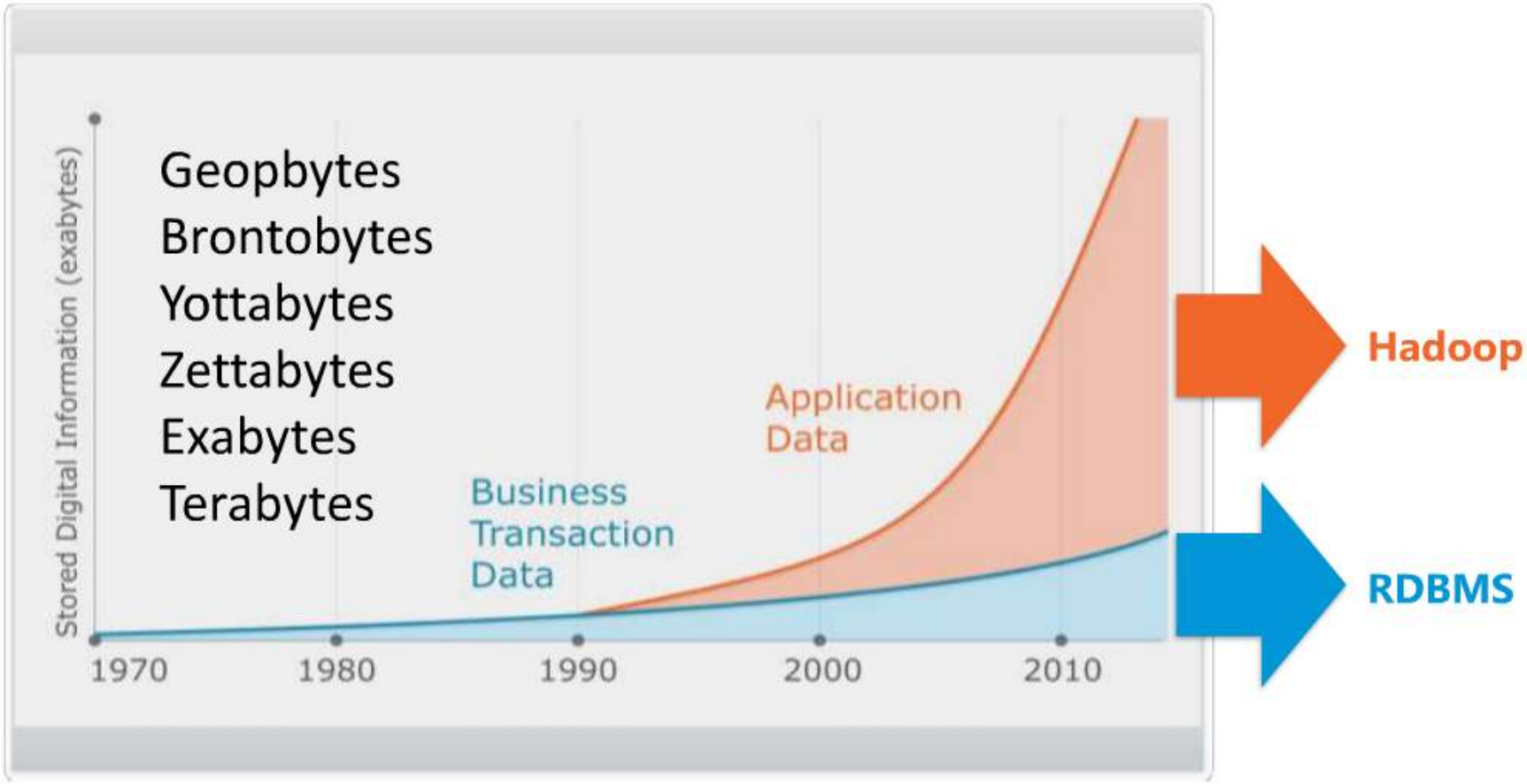
왜 대용량에 Apache Hadoop이 적합한가?

- 애플리케이션/트랜잭션 로그 정보는 매우 크다.
  - 대용량 파일을 저장할 수 있는 분산 파일 시스템을 제공한다.
- I/O 집중적이면서 CPU도 많이 사용한다.
  - 멀티 노드로 부하를 분산시켜 처리한다.
- 데이터베이스는 하드웨어 추가 시 성능 향상이 linear하지 않다.
  - 장비를 증가시킬 수록 성능이 linear에 가깝게 향상된다.
- 데이터베이스는 소프트웨어와 하드웨어가 비싸다.
  - Apache Hadoop은 무료이다.
  - Intel Core 머신과 리눅스는 싸다.

Hadoop의 다양한 응용 분야

- ETL(Extract, Transform, Load)
- Data Warehouse
- Storage for Log Aggregator
- Distributed Data Storage (예; CDN)
- Spam Filtering
- Biometric
- Online Content Optimization
- Parallel Image, Movie Clip Processing
- Machine Learning
- Science
- Search Engine

데이터 처리에 있어서 Hadoop, RDMBS의 위치



데이터베이스와 Hadoop 비교

	Traditional RDBMS	MapReduce
Data size	Gigabytes	Petabytes
Access	Interactive and batch	Batch
Updates	Read and write many times	Write once, read many times
Structure	Static schema	Dynamic schema
Integrity	High	Low
Scaling	Nonlinear	Linear

국내/해외 Hadoop Business 상황

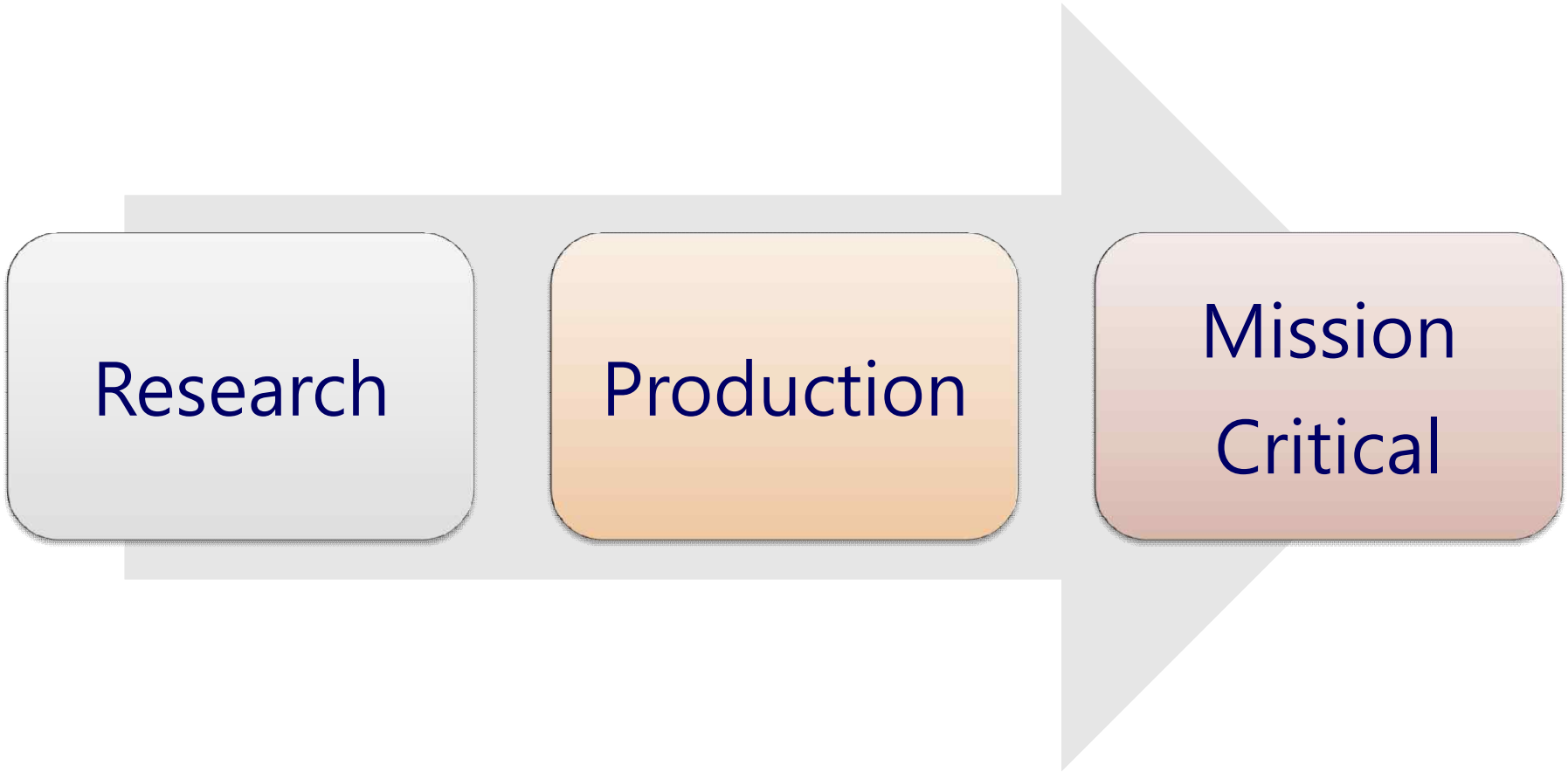
해외

Ready for Business

국내

Research, Ready for Business (일부)

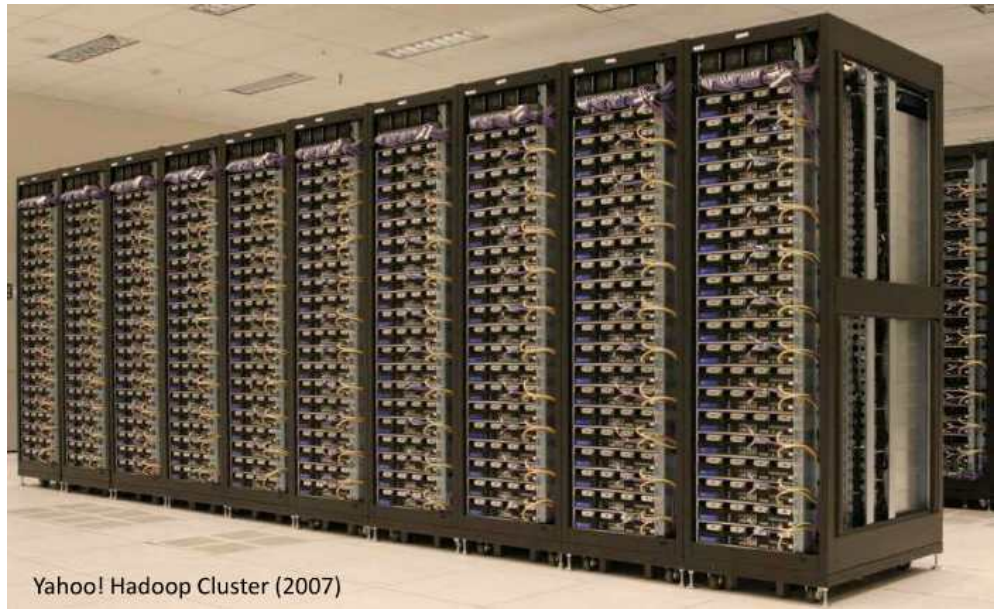
일반적인 Hadoop 적용 단계



복잡하면서 크리티컬한 비즈니스 문제

- Modeling true risk
- Customer churn analysis
- Recommendation engine
- Ad targeting
- PoS transaction analysis
- Analyzing network data to predict failure
- Threat analysis
- Trade surveillance
- Search quality
- Data “sandbox”

Hadoop Cluster



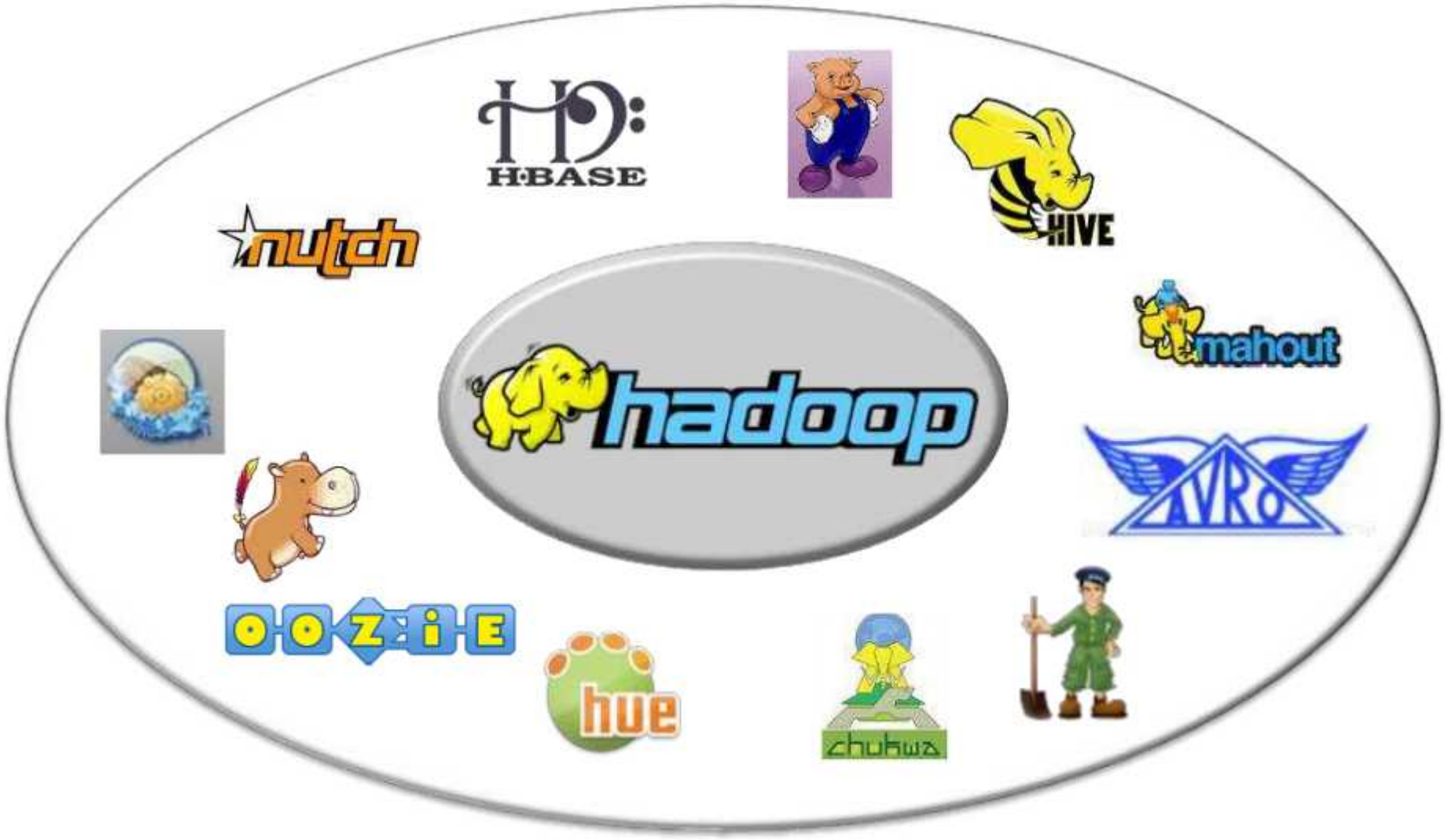
ucloud 유클라우드

amazon web services™

Hadoop Cluster 구축 시 필요한 기본 구성

- 1G~10G 스위치
- 노드 대수
  - Pilot : 최소 4~6대
  - Development : 10대 이하
  - Production : 최소 10대~20대
- 노드 스펙
  - 2 CPU(4 Core Per CPU) Xeons 2.5GHz 이상
  - 4 * 1TB SATA HDD or 4 * 2TB SATA HDD
  - 16G RAM
  - Dual 1G Ethernet
- Ubuntu Linux Server 10.04 64bit or CentOS
- Sun Java SDK 1.6.0_23 64bit
- Apache Hadoop 0.20.2

Hadoop Echosystem



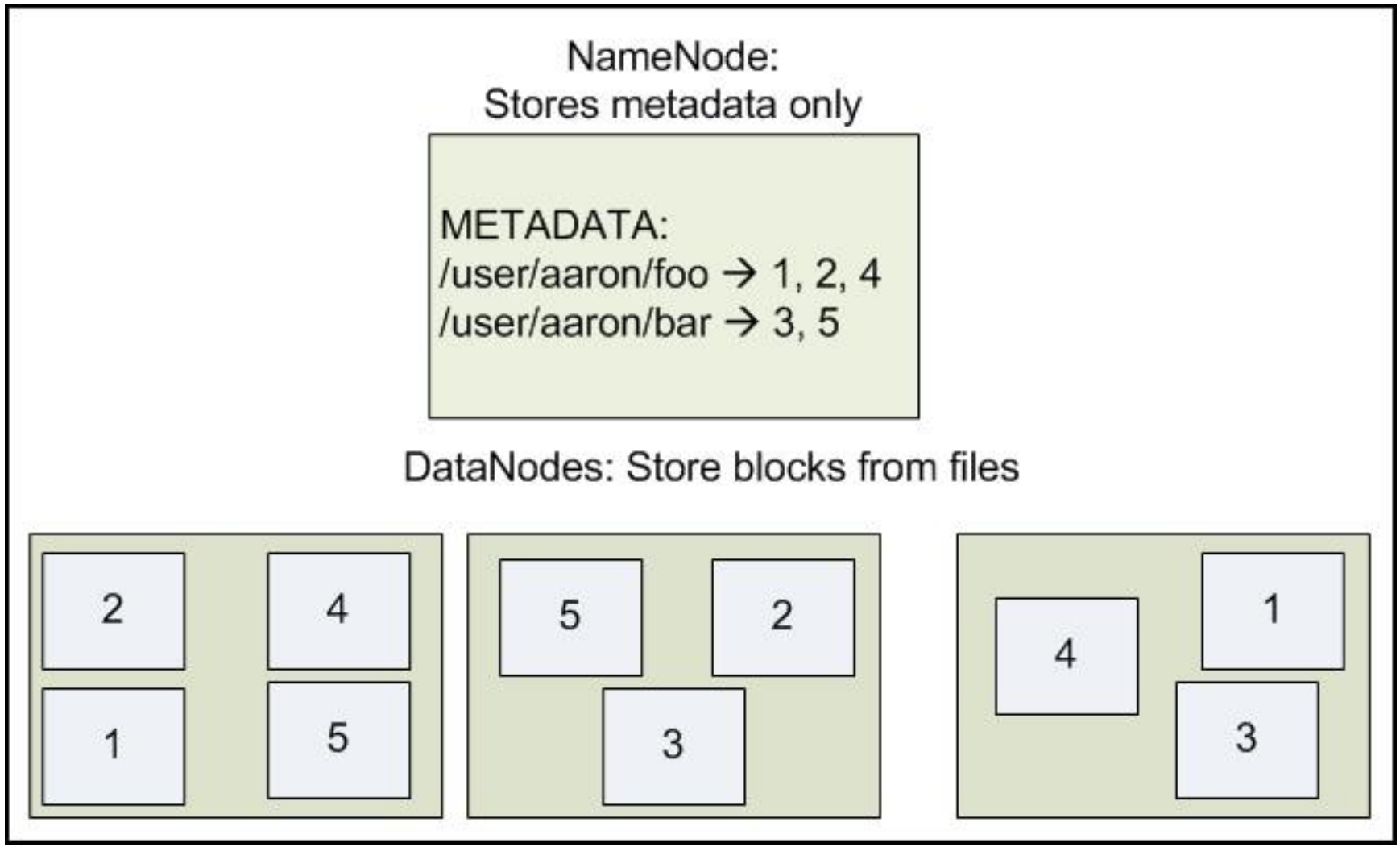
Hadoop 배포판

Apache Hadoop 기초

- **File System** : HDFS(Hadoop Distributed File System)
  - 파일을 64M 단위로 나누어 장비에 나누어서 저장하는 방식
  - 사용자는 하나의 파일로 보이나 실제로는 나누어져 있음
  - 2003년 Google이 논문으로 Google File System을 발표

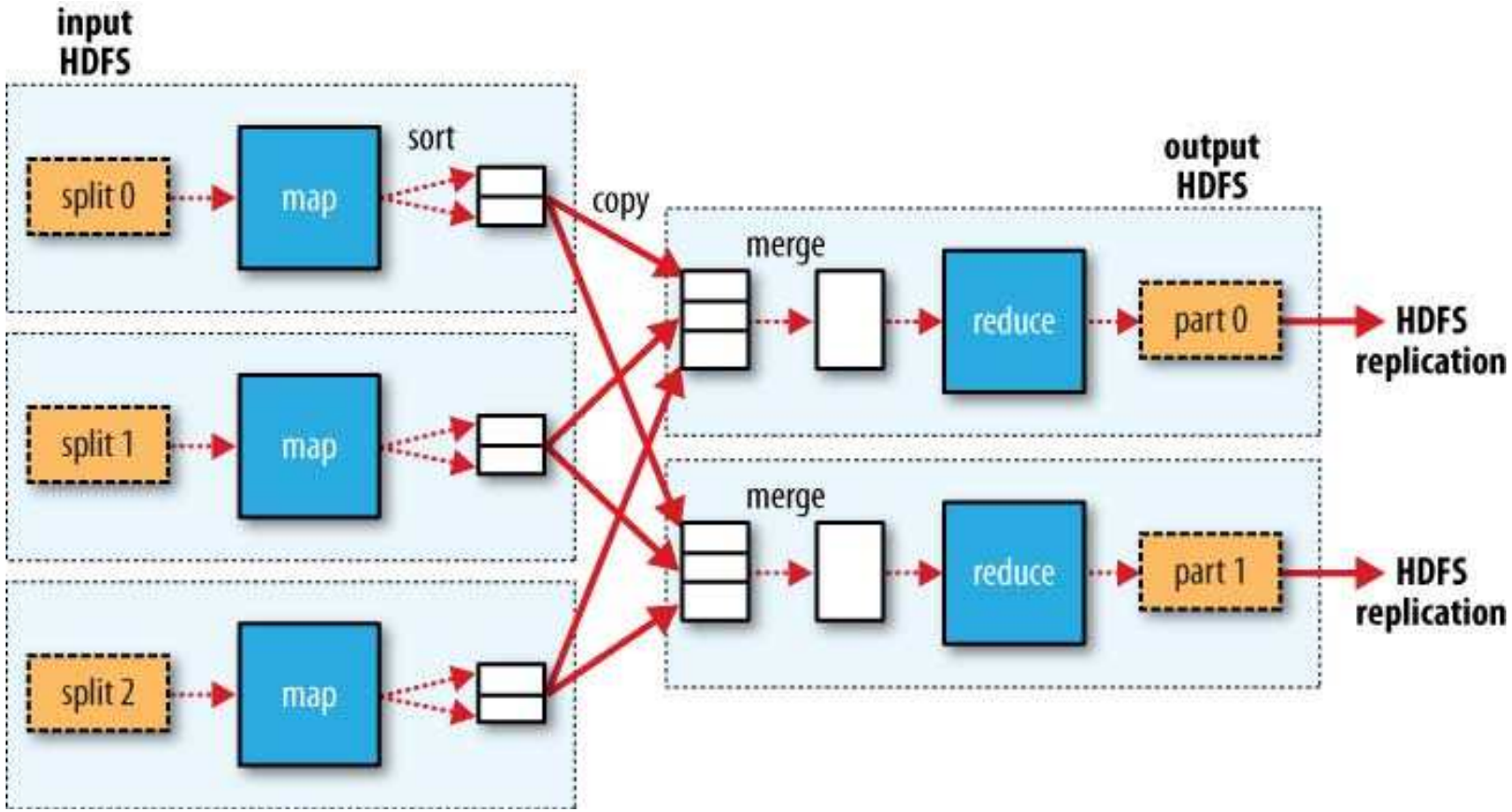
- **프로그래밍 모델**(MapReduce) (2004년 Google이 논문 발표)
  - HDFS의 파일을 이용하여 처리하는 방법을 제공
  - Parallelization, Distribution, Fault-Tolerance …

파일 시스템 : HDFS


NameNode:
Stores metadata only
METADATA:
/user/aaron/foo → 1, 2, 4
/user/aaron/bar → 3, 5
DataNodes: Store blocks from files
2
4
1
5
5
2
3
4
1
3

프로그래밍 모델 : MapReduce

- HDFS의 파일을 처리하기 위한 프로그래밍 모델

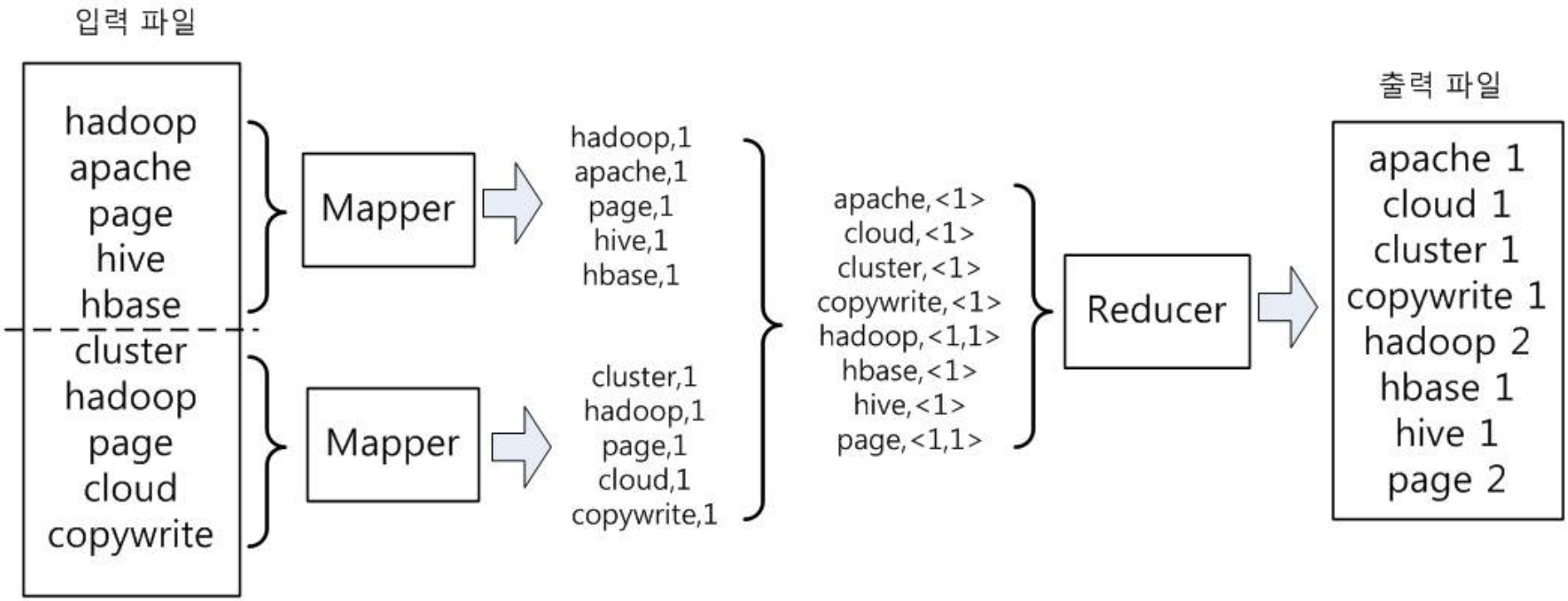


WordCount

- Hadoop의 MapReduce Framework 동작을 이해하는 핵심 예제
- 각각의 ROW에 하나의 Word가 있을 때 Word의 개수를 알아내는 예제

입력 파일(Mapper의 Input)	출력 파일(Reduce Output)
hadoop	apache 1
apache	cloud 1
page	cluster 1
hive	copywrite 1
hbase	hadoop 2
cluster	hbase 1
hadoop	hive 1
page	page 2
cloud	
copywrite	

WordCount


입력 파일
hadoop
apache
page
hive
hbase
cluster
hadoop
page
cloud
copywrite
Mapper
Mapper
hadoop,1
apache,1
page,1
hive,1
hbase,1
cluster,1
hadoop,1
page,1
cloud,1
copywrite,1
apache,<1>
cloud,<1>
cluster,<1>
copywrite,<1>
hadoop,<1,1>
hbase,<1>
hive,<1>
page,<1,1>
Reducer
출력 파일
apache 1
cloud 1
cluster 1
copywrite 1
hadoop 2
hbase 1
hive 1
page 2

WordCount의 Mapper

- 파일의 1ROW를 읽어서 Word, 1을 출력하는 Mapper

```
public class WordCountMapper
                    extends Mapper<Object, Text, Text, IntWritable> {

  private final static IntWritable one = new IntWritable(1);

  public void map(Object key, Text word, Context context)
                          throws IOException, InterruptedException {
    context.write(word, one);
  }

}
```

WordCount의 Reducer

- Mapper의 출력 중에서 같은 Key로 묶어서 처리하는 Reducer

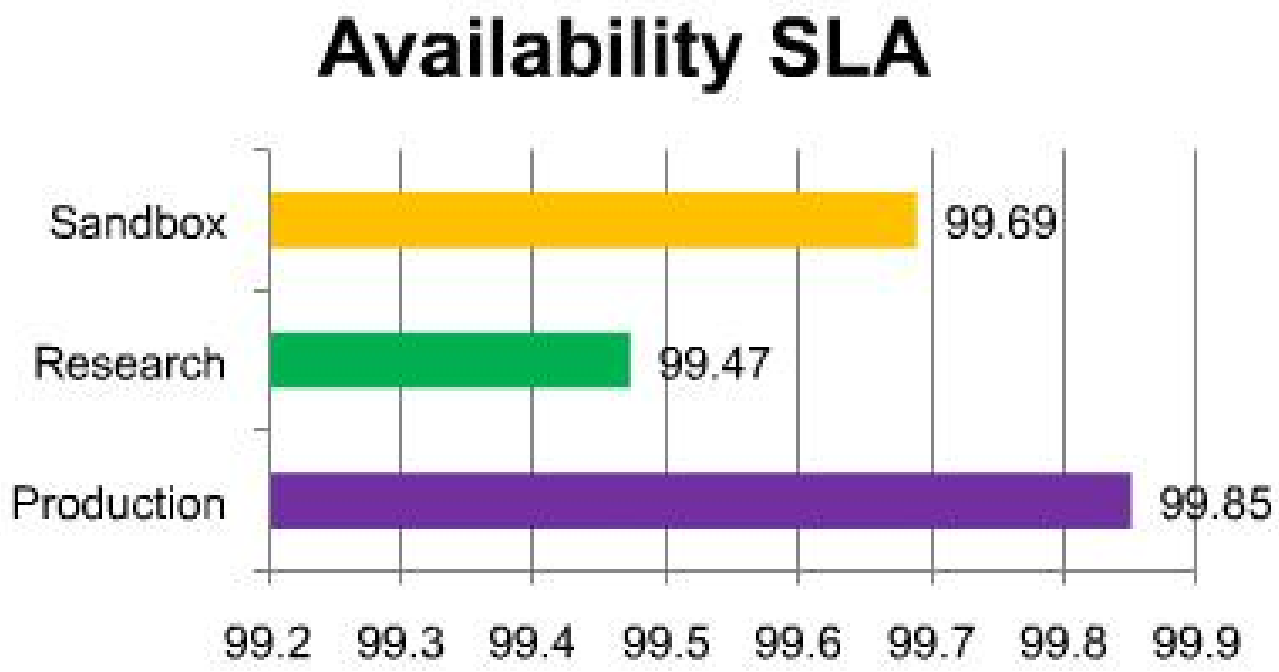
```
public class WordCountReducer
                    extends Reducer<Text, IntWritable, Text, IntWritable> {

  private IntWritable count = new IntWritable(); // for Performance

  public void reduce(Text word, Iterable<IntWritable> values,
                     Context context) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
      sum += val.get();
    }
    count.set(sum);
    context.write(word, count); // Result: Word Count (ex; hadoop 3)
  }
}
```

Database와 Hadoop의 추천 알고리즘 성능 비교

구분	Oracle 기반 머신	Hadoop 기반 머신
CPU	100%	70%
Core	80 Core	Intel 8 Core * 20 = 160 Core
처리 시간	1시간	34분
기간	1개월	1개월
상품수	120,000,000	
사용자수(T)	1,300,000	
장비 비용	6억 이상 고가 High End Server	300만원 * 20 = 6,000만원
라이선스 비용	예) Core 당 700만원 * 80 = 56,000만원	0

Hadoop의 Availability

- Hadoop은 기반 인프라 성격을 가지고 있어서 Availability가 매우 중요
- 현재 Hadoop은 Enterprise 수준의 SLA를 제공하지는 않음



Hadoop이 현재 가지고 있는 문제점

- Namenode 이중화 문제
  - Fail시 HDFS에 접근이 불가능해지는 문제
- Job Tracker 이중화 문제
  - Fail시 Hadoop Job 자체가 동작하지 않는 문제
- 작은 파일이 많은 경우
  - Namenode의 메모리 소비가 과도한 문제
- 노드가 다운된 경우
  - 급격한 성능 저하 발생
- MapReduce Job의 Availability
- Local File System 오류로 인한 노드 Fail시
  - 일부 파일을 읽지 못하는 문제 발생

국내 업계의 향후 전망

- CRM을 비롯한 많은 시스템을 관리하는 관리자들이 지속적으로 Hadoop에 관심을 보일 것으로 예상
- 많은 인프라 시스템에서 대용량 부분이 Hadoop으로 이전 될 것
  - 대용량 데이터를 갖고 있지 않더라도 Hadoop을 통해 수집/관리/처리 하려는 시도들이 나타나고 있음
  - 대용량 데이터를 보관하고 처리할 능력을 갖추었다면 본격적인 데이터 분석을 시도할 수 있게 됨
- ETL, Data Warehouse, Machine Learning 분야가 Hadoop을 중심으로 성장할 것
- ETL, Miner, Data Warehouse 등의 제품에서 Hadoop과 접목을 시도하는 제품이 나타날 것
- 이미 도입한 회사는 Mission Critical로 넘어갈 것

Hadoop 도입 시 고려할 사항

- 기존 오픈 소스와는 다른 유형의 오픈 소스
  - 일반적인 오픈 소스는 Framework, Library 정도 수준이지만 Hadoop은 인프라 성격을 가진 오픈 소스
- 오픈 소스 친화적인 엔지니어 확보 중요
  - 소스코드까지 고치려는 시도를 할 수 있는 엔지니어
- Linux 친화적이면서 System Engineer 성향을 가진 엔지니어 필요
- 적용하기 까지 많은 테스트와 최적화 필요하므로 인내 필요
  - 데이터를 다루는 작업은 기존까지 개발자의 몫이 아니었고
  - 지루한 작업에 매우 많은 염증을 느낄 수 있음