Inspire…Educate…Transform.

Engineering Big Data

YARN, BSP, Spark SQL Variants

Dr. Sreerama KV Murthy
CEO, Quadratyx

Aug 30, 2015

Wake-Up Quiz

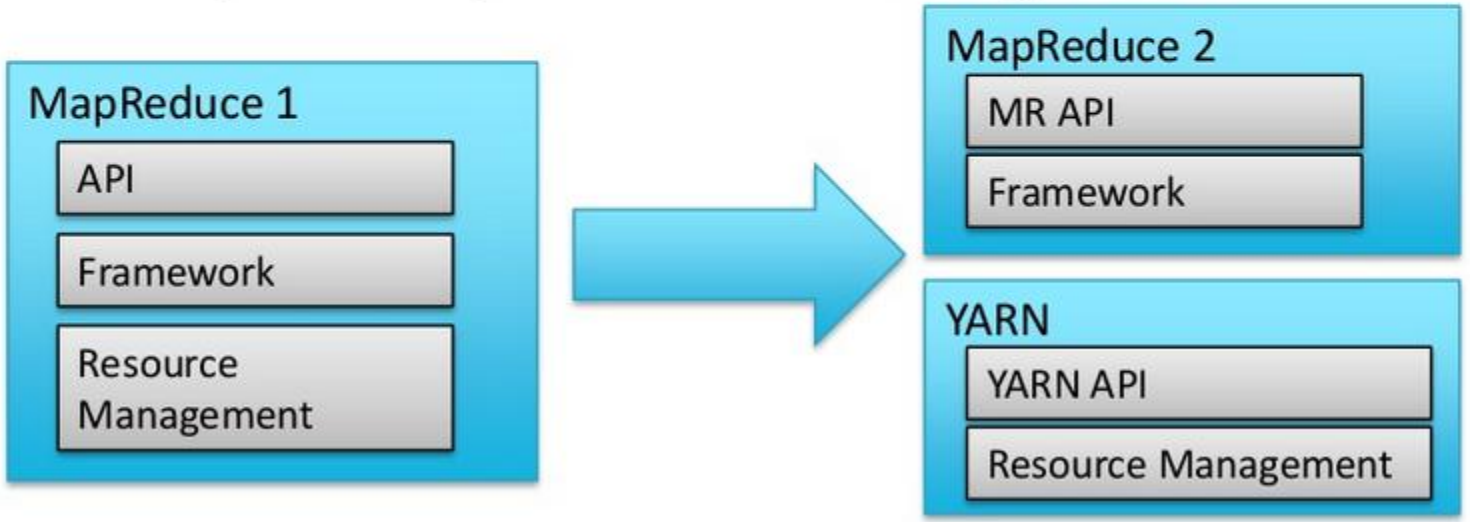
YARN, MR2

Yet Another Resource Negotiator

- YARN provides the daemons and APIs necessary to develop generic distributed applications of any kind.
- YARN handles and schedules resource requests (such as memory and CPU) from applications, and supervises their execution.
- YARN can run applications that do not follow the Map Reduce model.
- MR2 is modeled as “just another” client application on YARN.

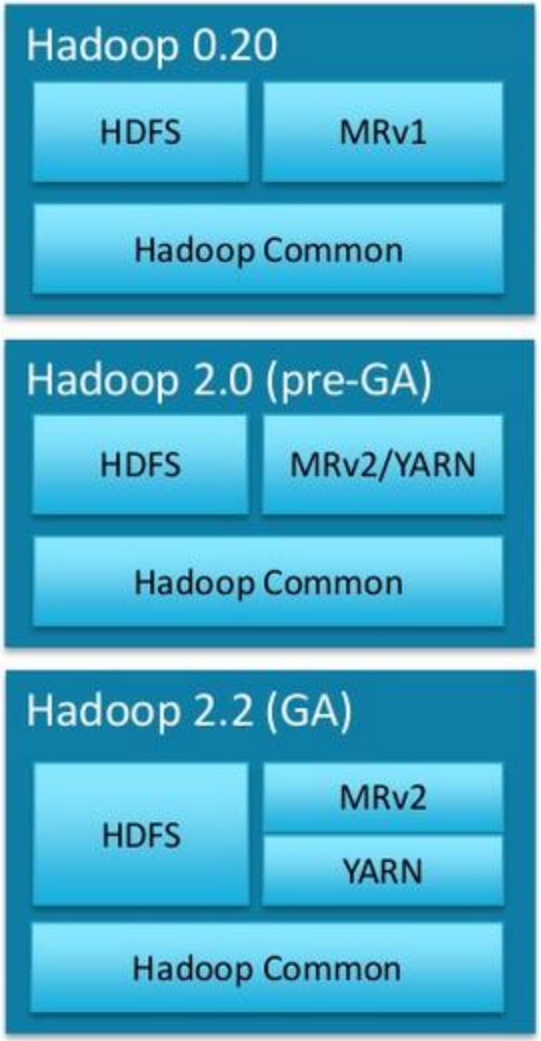
MR1, MR2, Yarn

- **MapReduce 1 ("Classic") has three main components**
  - API – for user-level programming of MR applications
  - Framework – runtime services for running Map and Reduce processes, shuffling and sorting, etc.
  - Resource management – infrastructure to monitor nodes, allocate resources, and schedule jobs
- **MapReduce 2 ("NextGen") moves Resource Management into YARN**

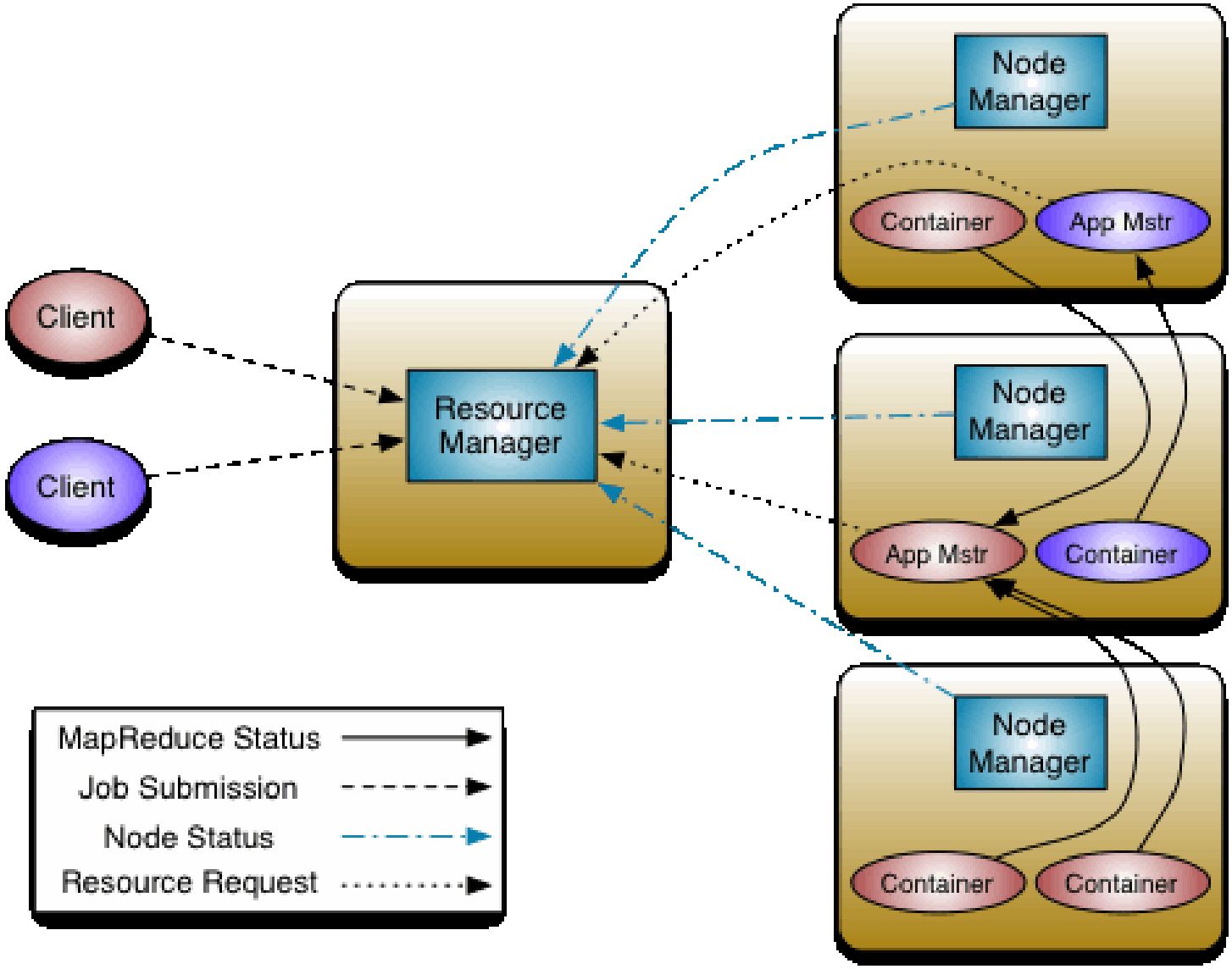


MR2 History

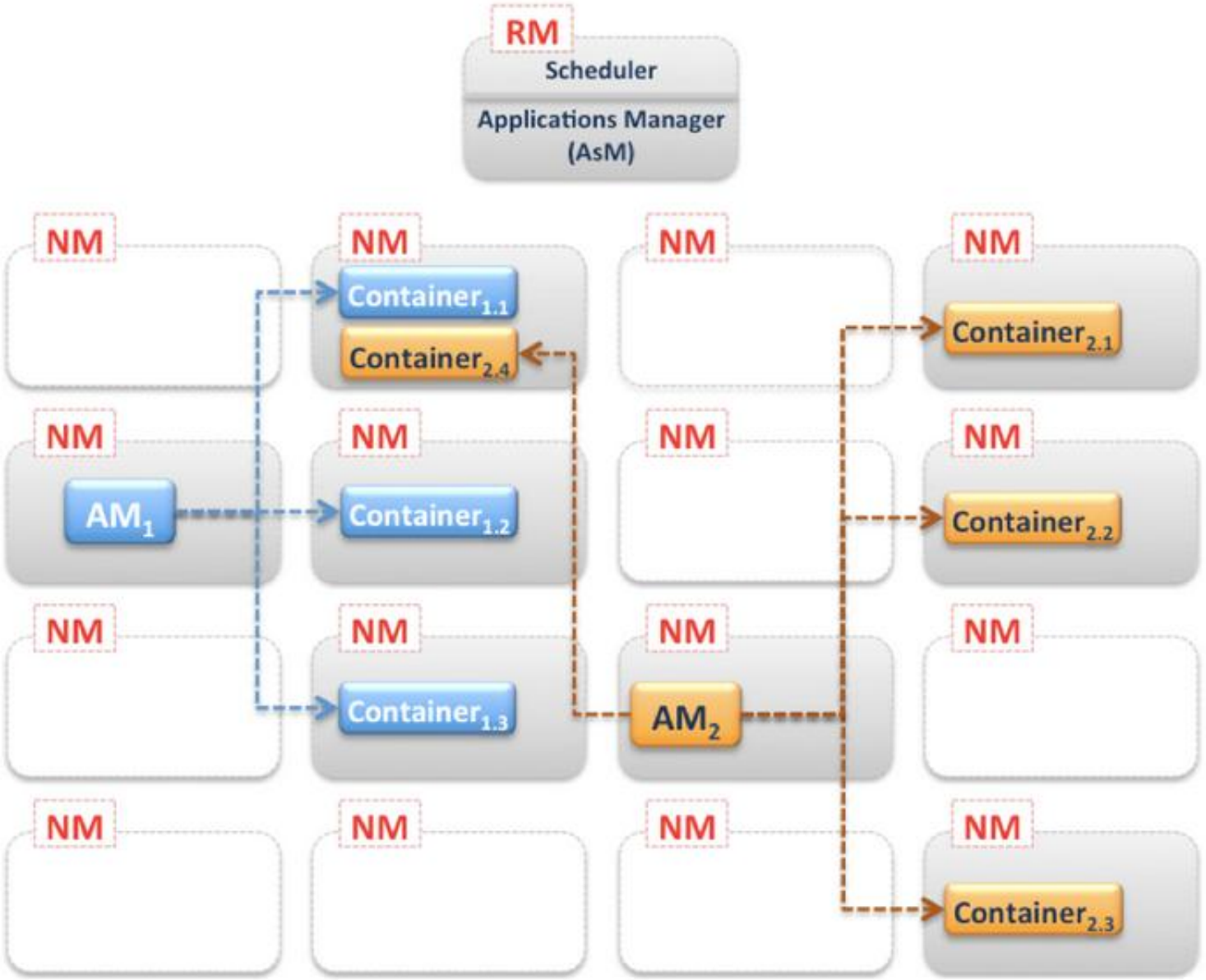
- **Originally architected at Yahoo in 2008**
- **"Alpha" in Hadoop 2 pre-GA**
  - Included in CDH 4
- **YARN promoted to Apache Hadoop sub-project**
  - summer 2013
- **"Production ready" in Hadoop 2 GA**
  - Included in CDH5 (Beta in Oct 2013)



MRv2, YARN: JobTracker Redefined



A YARN CLUSTER

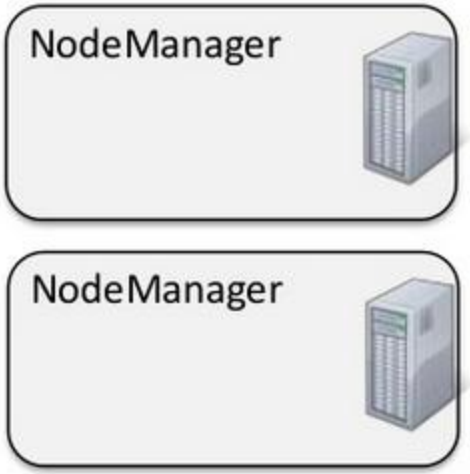


http://blog.cloudera.com/blog/2012/02/mapreduce-2-0-in-hadoop-0-23/

YARN daemons

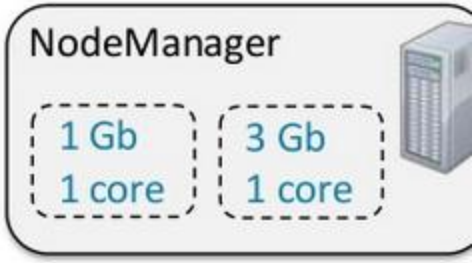
- **Resource Manager (RM)**
  - Runs on master node
  - Global resource scheduler
  - Arbitrates system resources between competing applications

- **Node Manager (NM)**
  - Runs on slave nodes
  - Communicates with RM



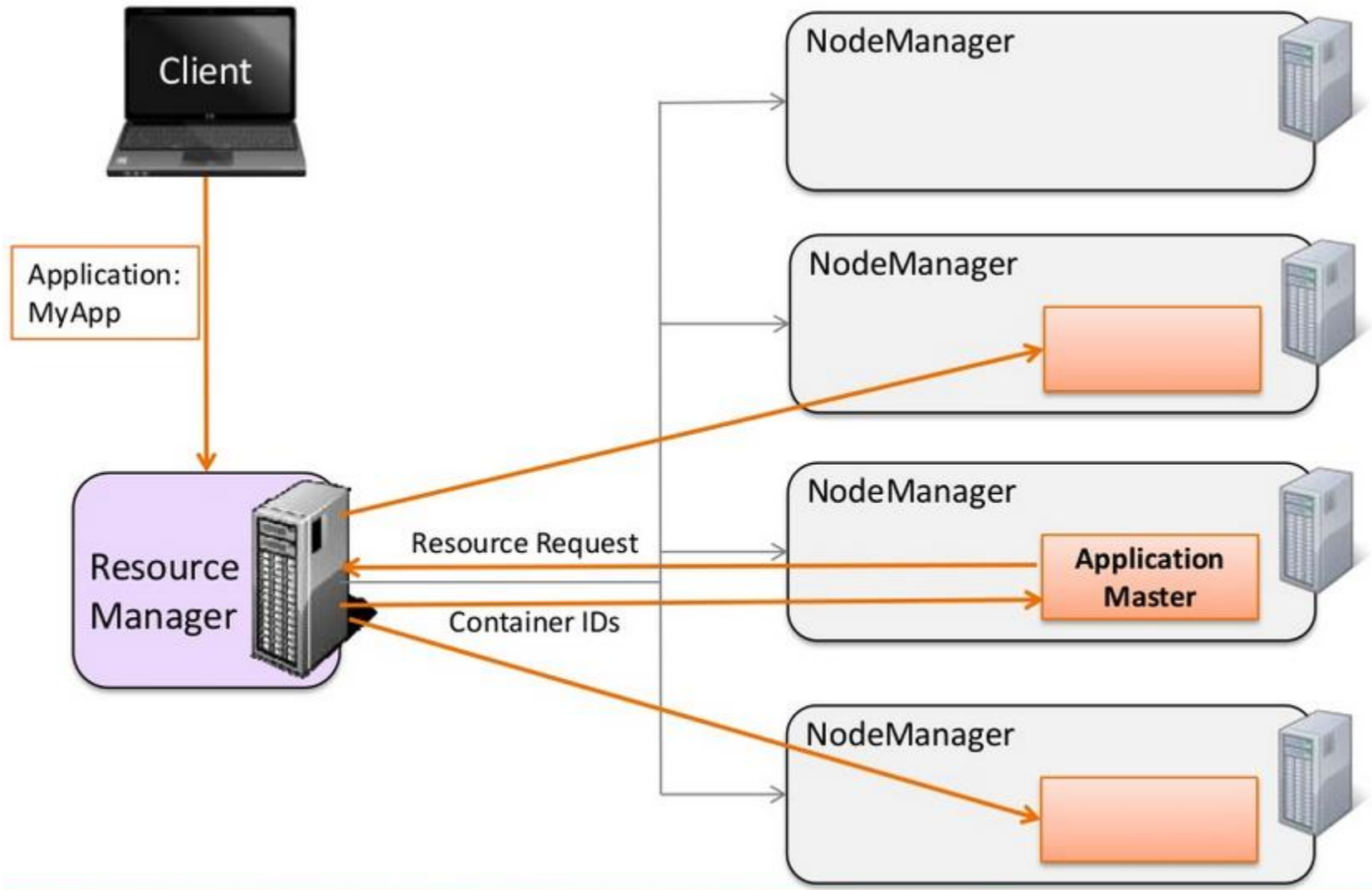
YARN daemons (contd.)

- **Containers**
  - Created by the RM upon request
  - Allocate a certain amount of resources (memory, CPU) on a slave node
  - Applications run in one or more containers

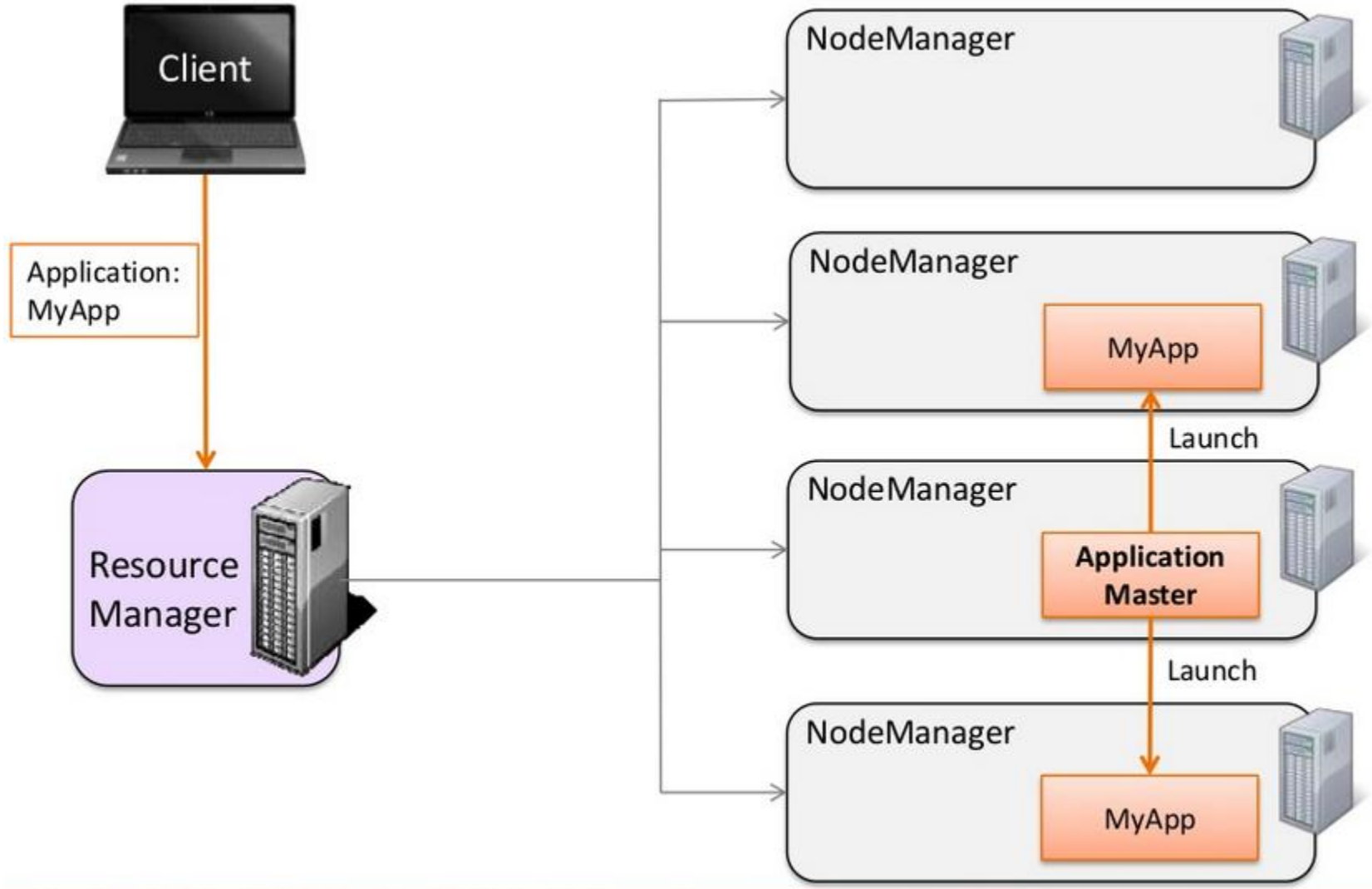


- **Application Master (AM)**
  - One per application
  - Framework/application specific
  - Runs in a container
  - Requests more containers to run application tasks

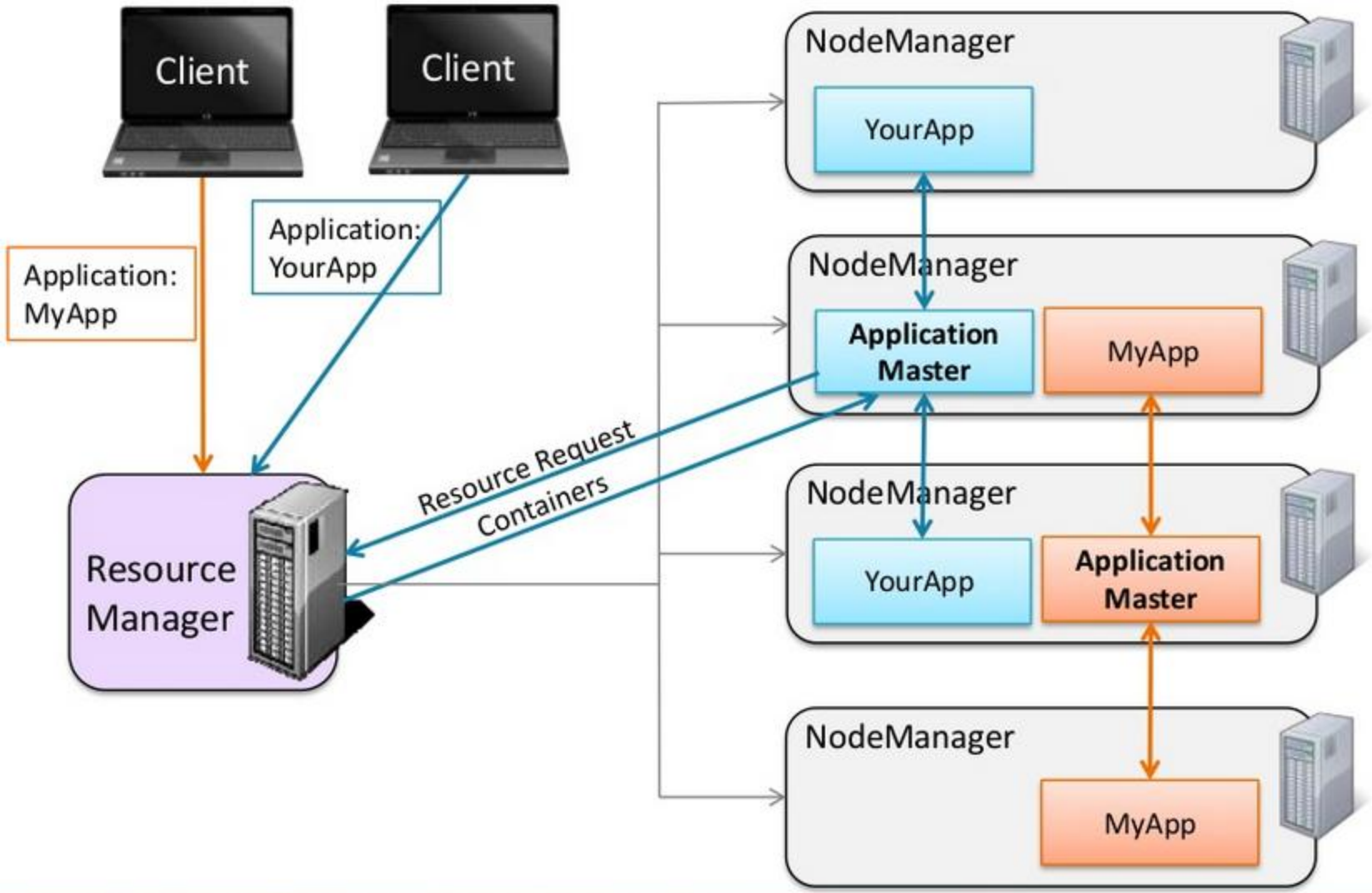
Running a YARN application



Running Apps on YARN - 2



Running Apps on YARN - 3



Role of a Resource Manager

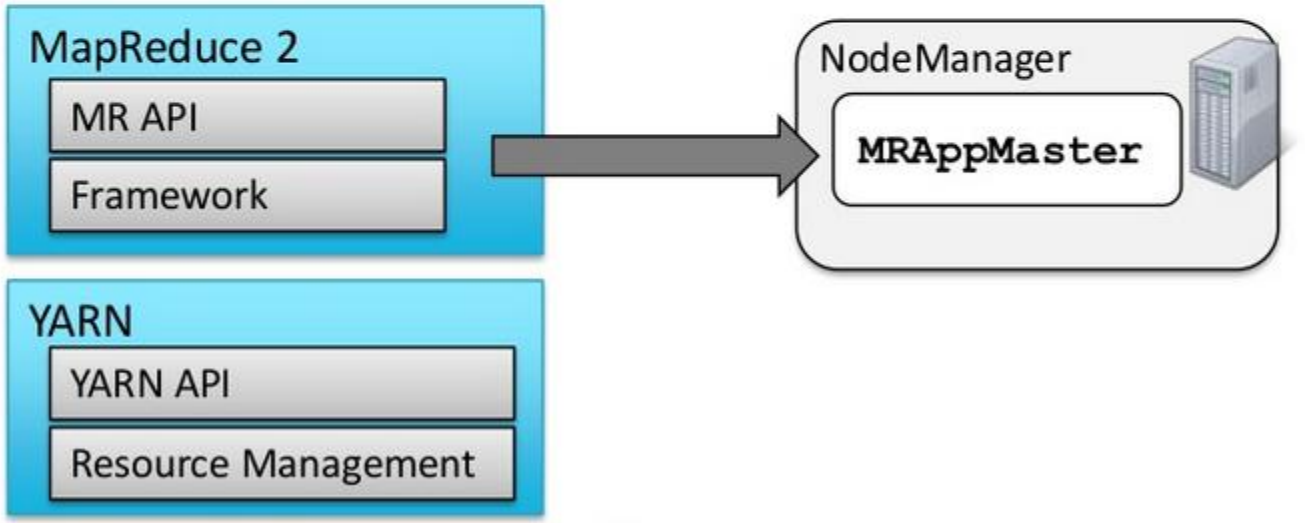
- **What it does**
  - Manages nodes
    - Tracks heartbeats from NodeManagers
  - Manages containers
    - Handles AM requests for resources
    - De-allocates containers when they expire or the application completes
  - Manages ApplicationMasters
    - Creates a container for AMs and tracks heartbeats
  - Manages security
    - Supports Kerberos

Role of a Node Manager

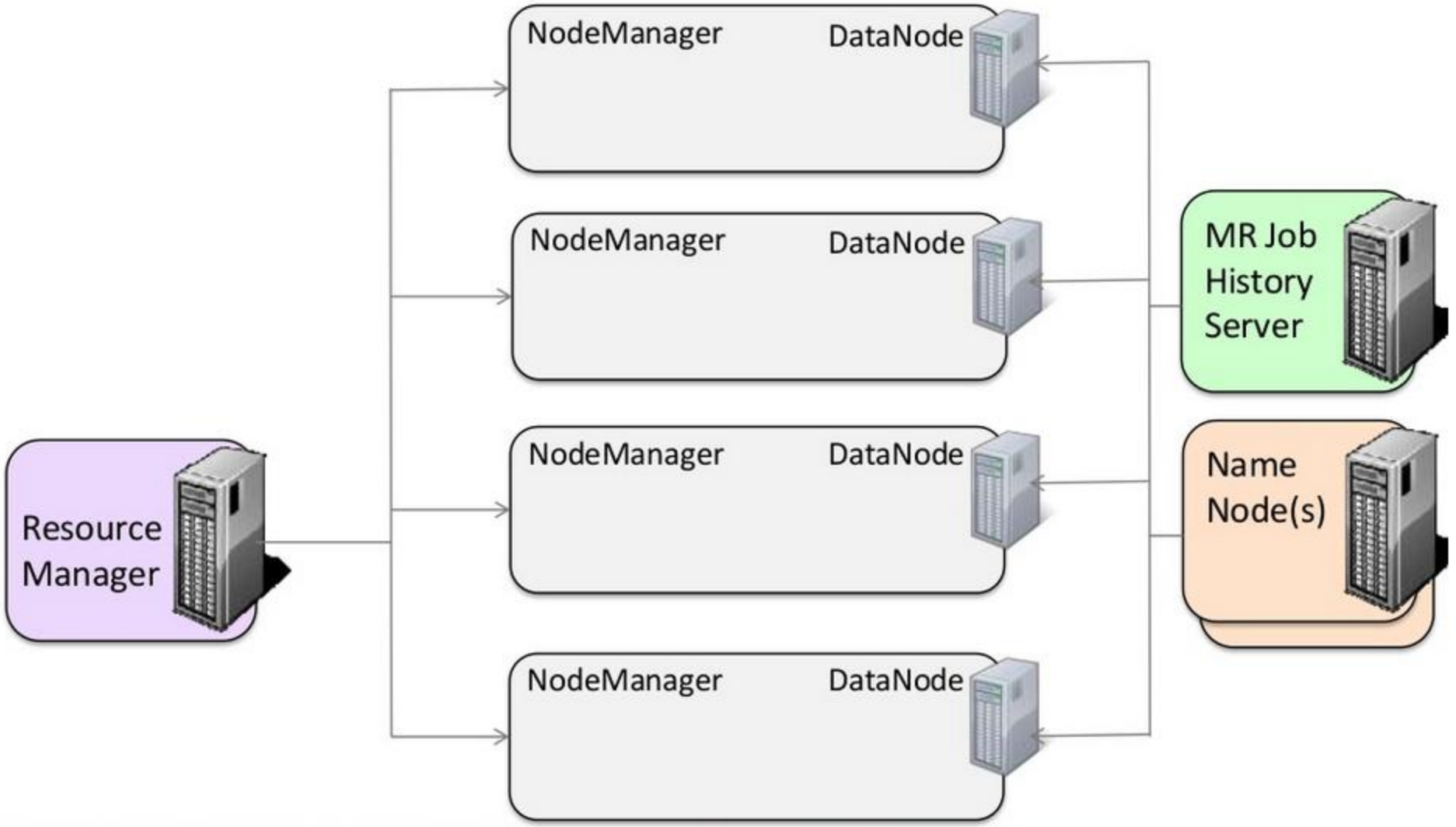
- **What it does**
  - Communicates with the RM
    - Registers and provides info on node resources
    - Sends heartbeats and container status
  - Manages processes in containers
    - Launches AMs on request from the RM
    - Launches application processes on request from AM
    - Monitors resource usage by containers; kills run-away processes
  - Provides logging services to applications
    - Aggregates logs for an application and saves them to HDFS
  - Runs auxiliary services
  - Maintains node level security via ACLs

MR2 uses YARN

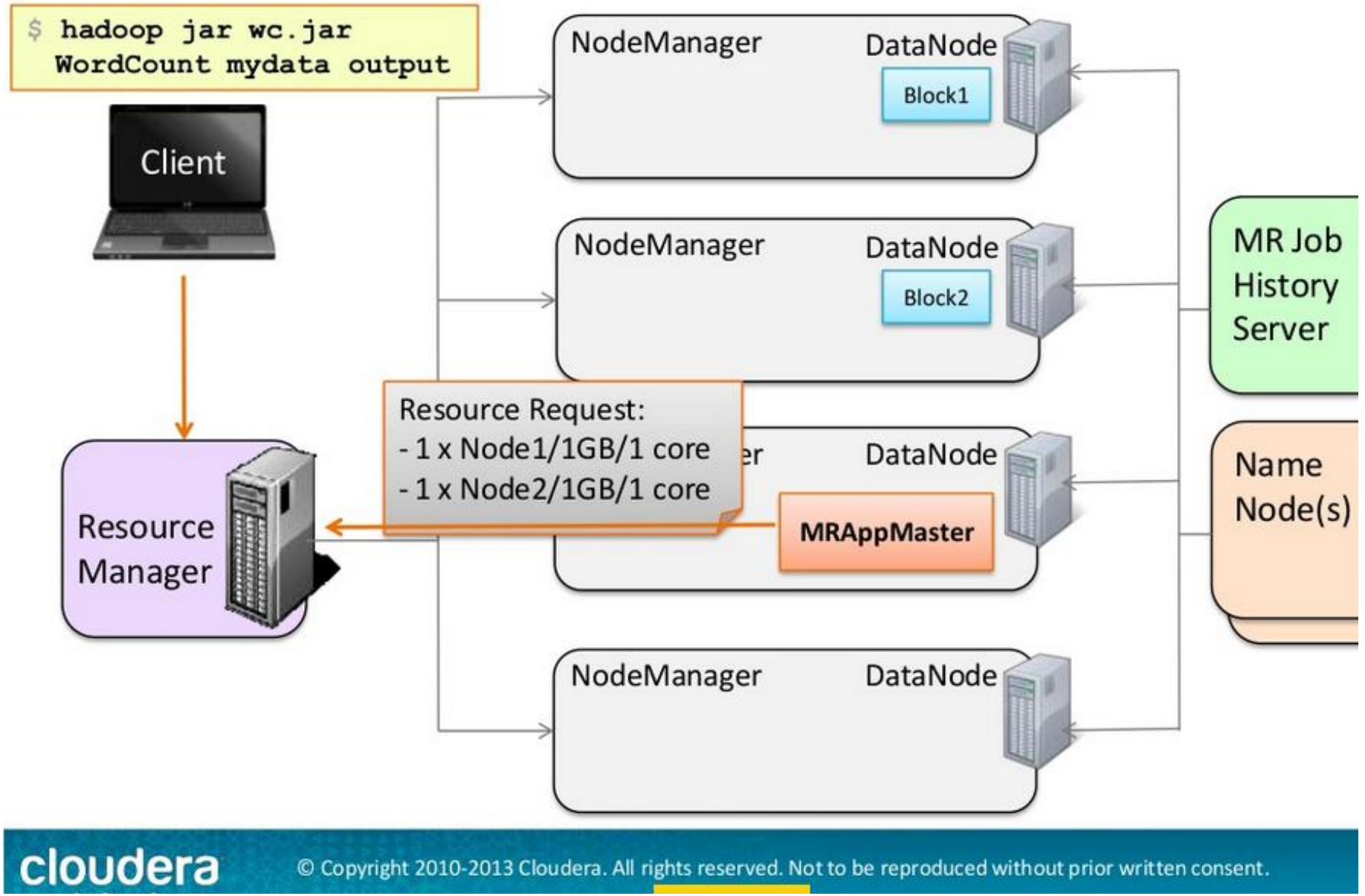
- **YARN does not know or care what kind of application it is running**
  - Could be MR or something else (e.g. Impala)
- **MR2 uses YARN**
  - Hadoop includes a MapReduce ApplicationMaster (MRAppMaster) to manage MR jobs
  - Each MapReduce job is an a new instance of an application



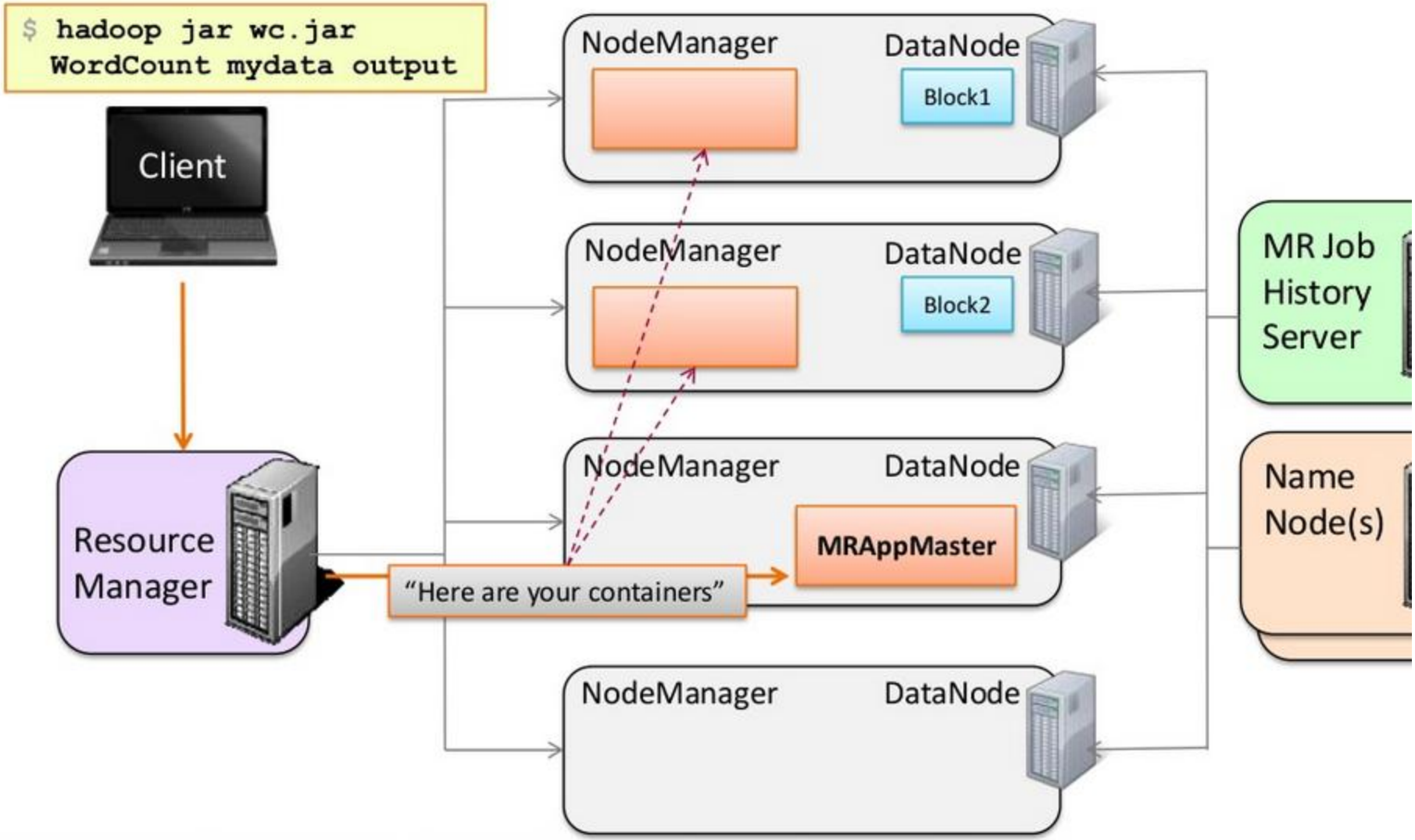
MR2 uses YARN - 2



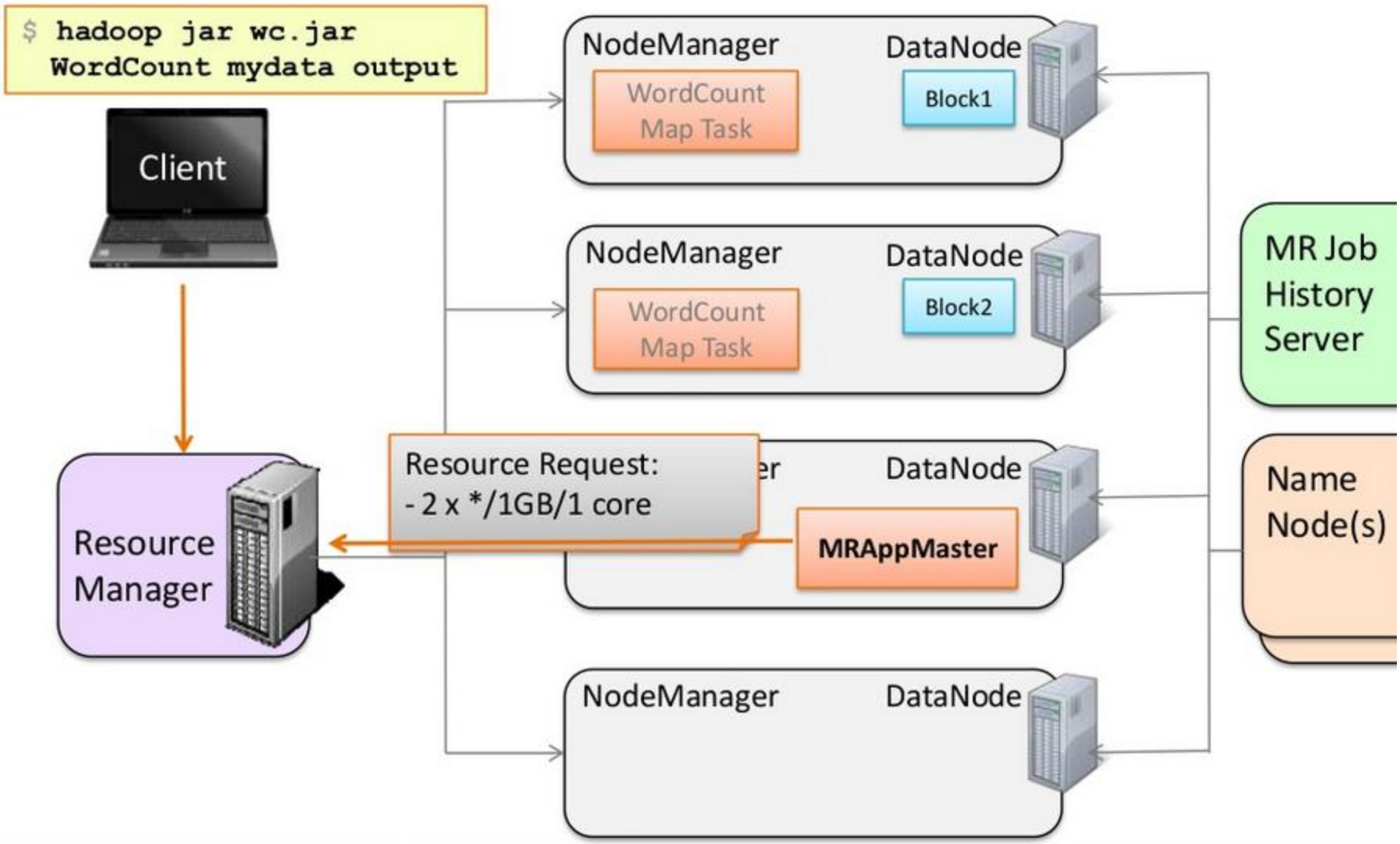
MR2 uses YARN - 3



MR2 uses YARN - 4

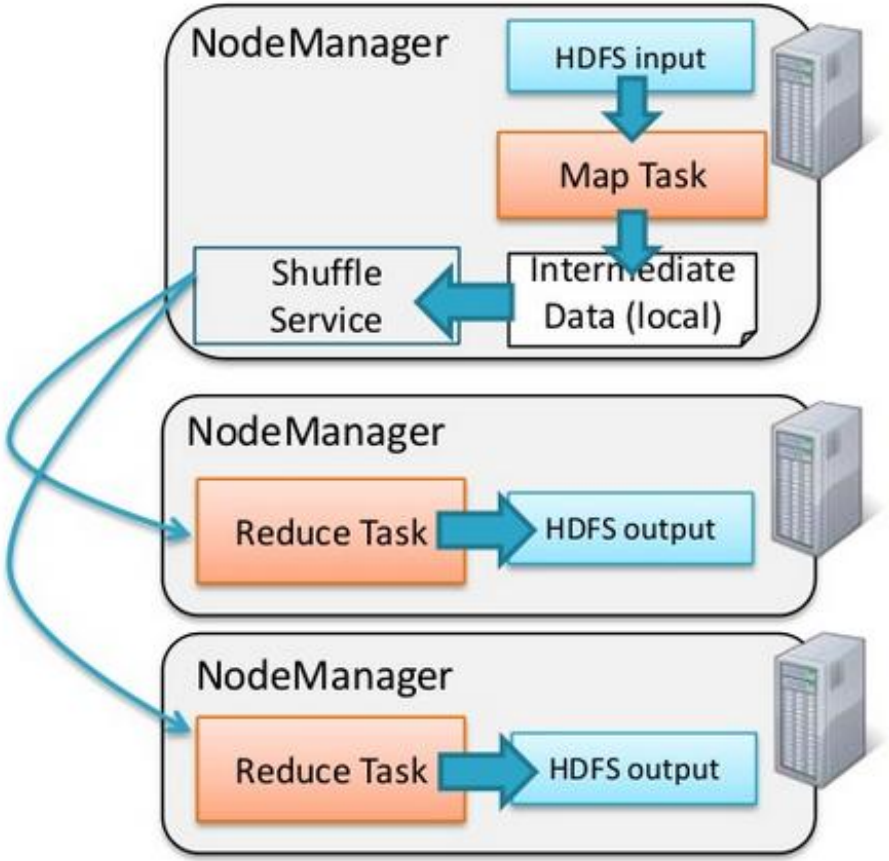


MR2 uses YARN - 5



MR2 Shuffle on YARN

- **In YARN, Shuffle is run as an auxiliary service**
  - Runs in the NodeManager JVM as a persistent service

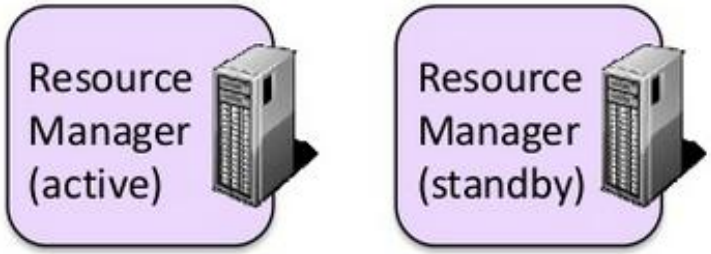


MR2 - YARN Fault Tolerance

- **Any of the following can fail**
  - Task (Container) – Handled just like in MRv1
    - MRAppMaster will re-attempt tasks that complete with exceptions or stop responding (4 times by default)
    - Applications with too many failed tasks are considered failed
  - Application Master
    - If application fails or if AM stops sending heartbeats, RM will re-attempt the whole application (2 times by default)
    - MRAppMaster optional setting: Job recovery
      - if false, all tasks will re-run
      - if true, MRAppMaster retrieves state of tasks when it restarts; only incomplete tasks will be re-run

MR2 – YARN Fault tolerance

- **Any of the following can fail**
  - NodeManager
    - If NM stops sending heartbeats to RM, it is removed from list of active nodes
    - Tasks on the node will be treated as failed by MRAppMaster
    - If the App Master node fails, it will be treated as a failed application
  - ResourceManager
    - No applications or tasks can be launched if RM is unavailable
    - Can be configured with High Availability

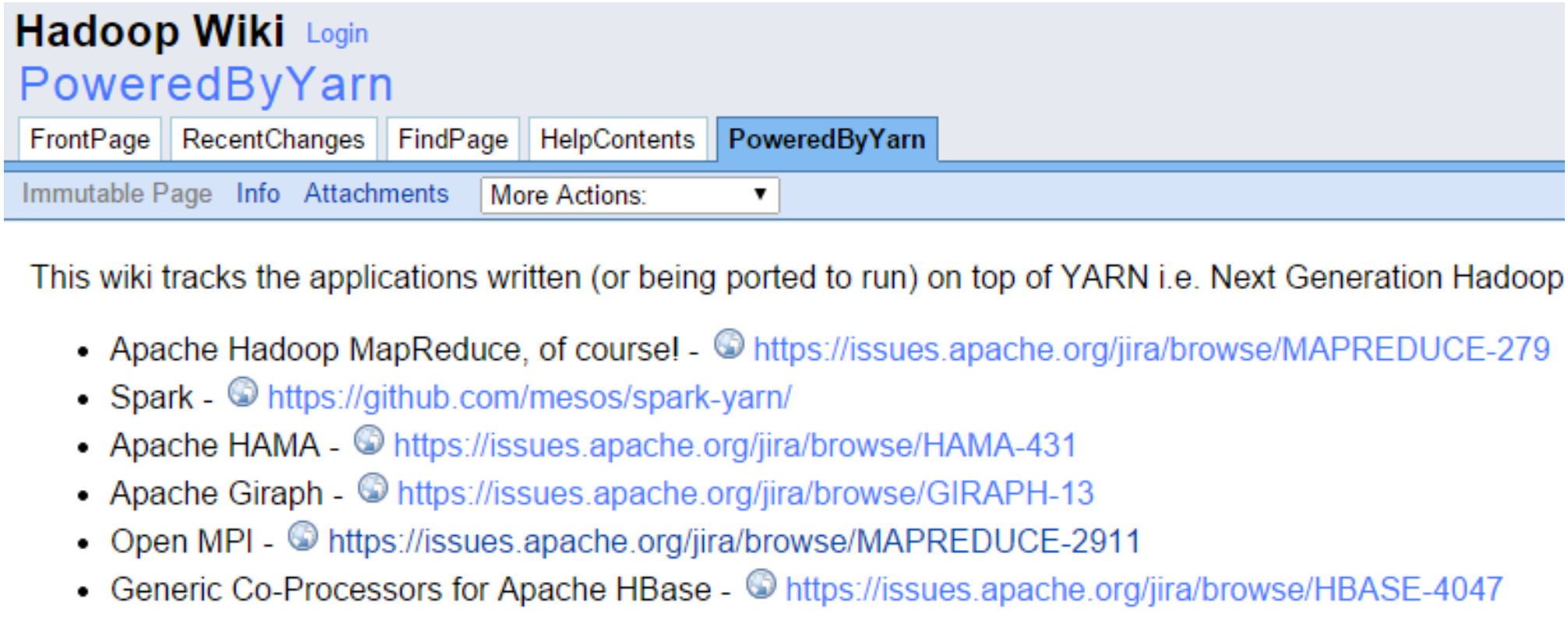


Job History and Logs

- YARN does not keep track of job history
- MapReduce Job History Server
  - Archives job's metrics and metadata
  - Can be accessed through Job History UI or Hue

`http://rmhost:19888/jobhistory`

Non-MR2 YARN Applications



Hadoop Wiki Login

PoweredByYarn

FrontPage | RecentChanges | FindPage | HelpContents | **PoweredByYarn**

Immutable Page Info Attachments More Actions:

This wiki tracks the applications written (or being ported to run) on top of YARN i.e. Next Generation Hadoop

- Apache Hadoop MapReduce, of course! - https://issues.apache.org/jira/browse/MAPREDUCE-279
- Spark - https://github.com/mesos/spark-yarn/
- Apache HAMA - https://issues.apache.org/jira/browse/HAMA-431
- Apache Giraph - https://issues.apache.org/jira/browse/GIRAPH-13
- Open MPI - https://issues.apache.org/jira/browse/MAPREDUCE-2911
- Generic Co-Processors for Apache HBase - https://issues.apache.org/jira/browse/HBASE-4047

Other ideas:

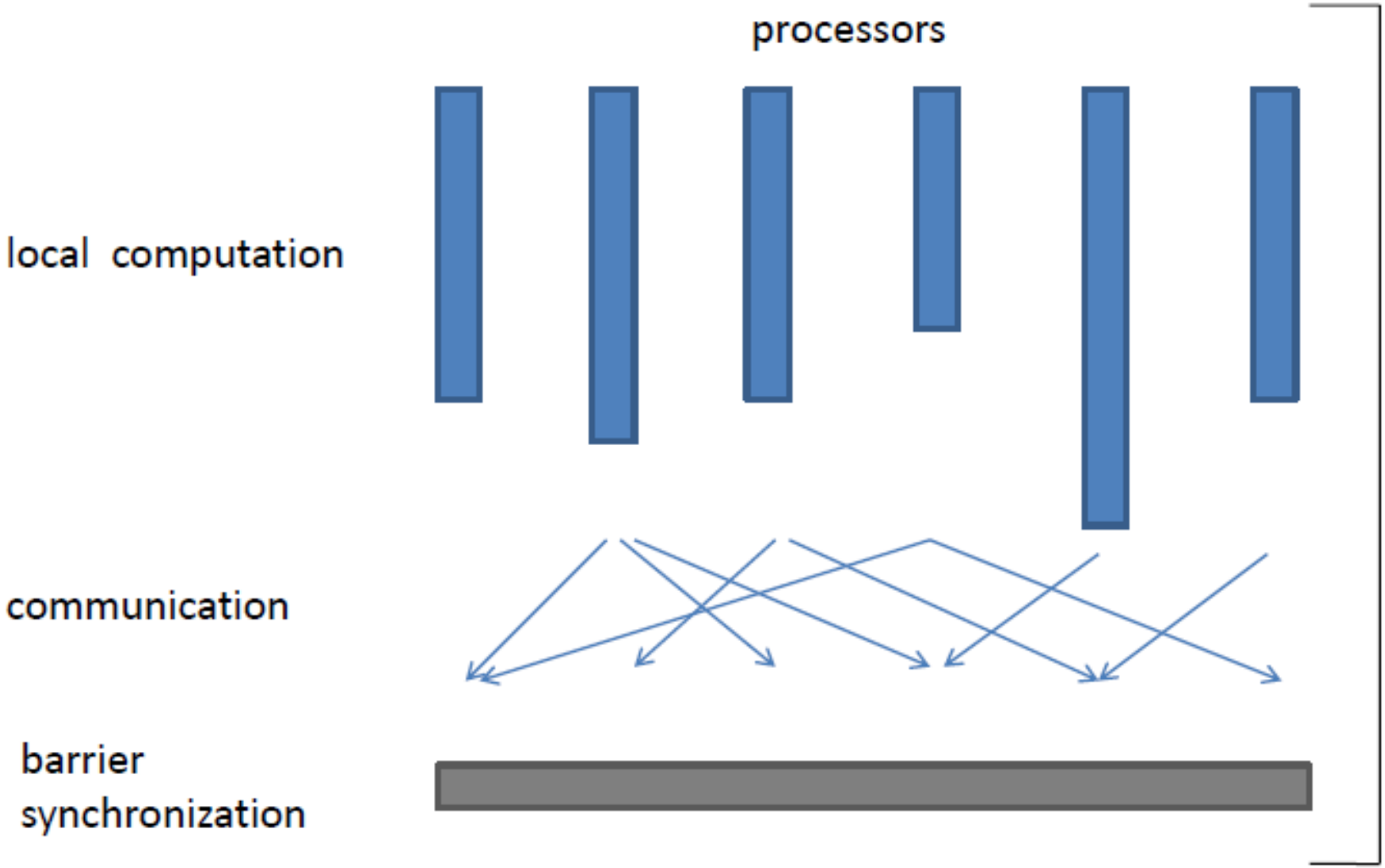
- Apache HBase deployment using YARN - https://issues.apache.org/jira/browse/HBASE-4329

Doug Cutting Video

http://www.cloudera.com/content/cloudera/en/resources/library/aboutcloudera/beyond-batch-the-evolution-of-the-hadoop-ecosystem-doug-cutting-video.html

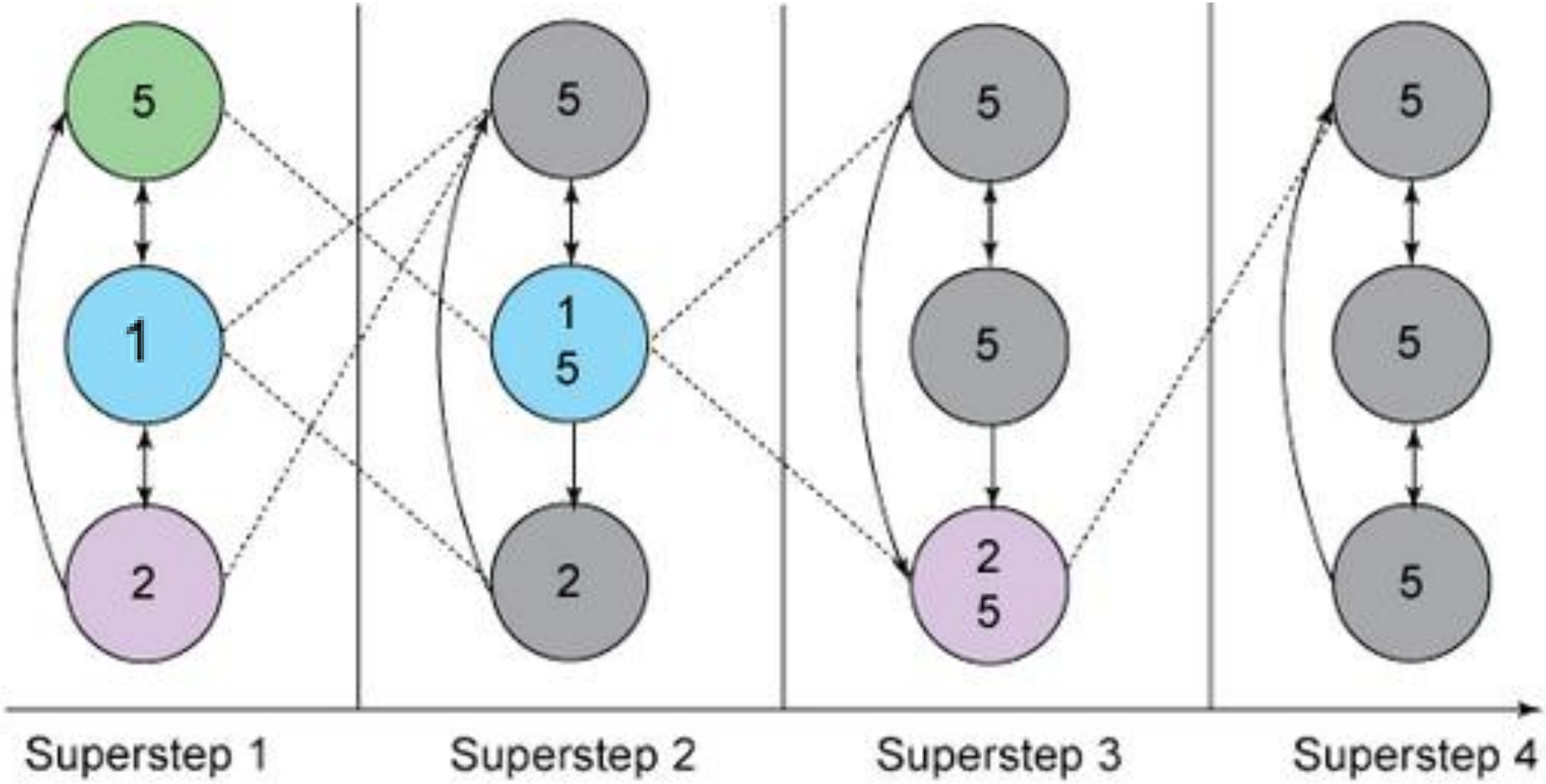
BULK SYNCHRONOUS PARALLEL (BSP) PROCESSING

Les Valiant's Bulk Synchronous Parallel processing model (1990)



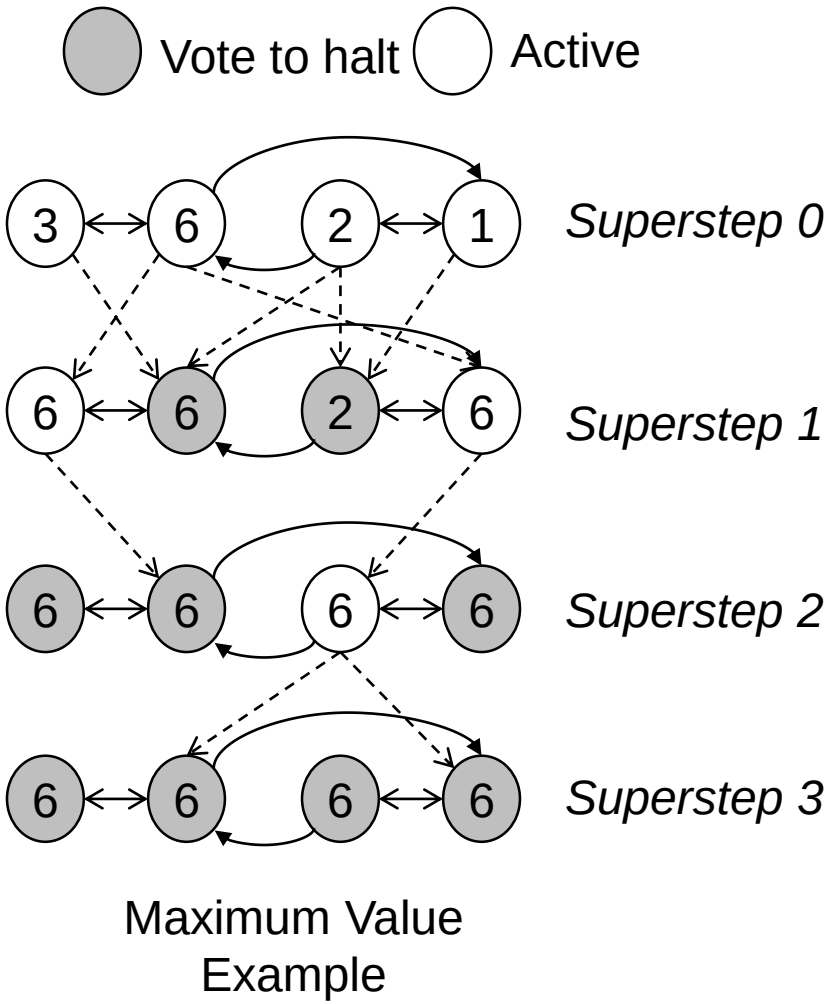
A BSP algorithm is composed by a sequence of "super steps"

BSP example: Find the largest vertex



Pregel & Apache Giraph

- Computation Model
  - Superstep as iteration
  - Vertex state machine: Active and Inactive, vote to halt
  - Message passing between vertices
  - Combiners
  - Aggregators
  - Topology mutation
- Master/worker model
- Graph partition: hashing
- Fault tolerance: checkpointing and confined recovery



Maximum Value Example

Giraph Page Rank Code Example


```
public class PageRankComputation
    extends BasicComputation<IntWritable, FloatWritable, ..> {
  /** Number of supersteps */
  public static final String SUPERSTEP_COUNT = "giraph.pageRank.superstepCount";

  @Override
  public void compute(Vertex<IntWritable ..> vertex, Iterable<FloatWritable>
messages) throws IOException {
    if (getSuperstep() >= 1) {
      float sum = 0;
      for (FloatWritable message : messages) {
        sum += message.get();
      }
      vertex.getValue().set((0.15f / getTotalNumVertices()) + 0.85f * sum);
    }
    if (getSuperstep() < getConf().getInt(SUPERSTEP_COUNT, 0)) {
      sendMessageToAllEdges(vertex,
          new FloatWritable(vertex.getValue().get() / vertex.getNumEdges()));
    } else {
      vertex.voteToHalt();
    }
  }
}
```

Why BSP

- Simple programming model
  - Super steps semantics are easy
- Preserve data locality
  - Improve performance
- Well-suited for iterative algorithms

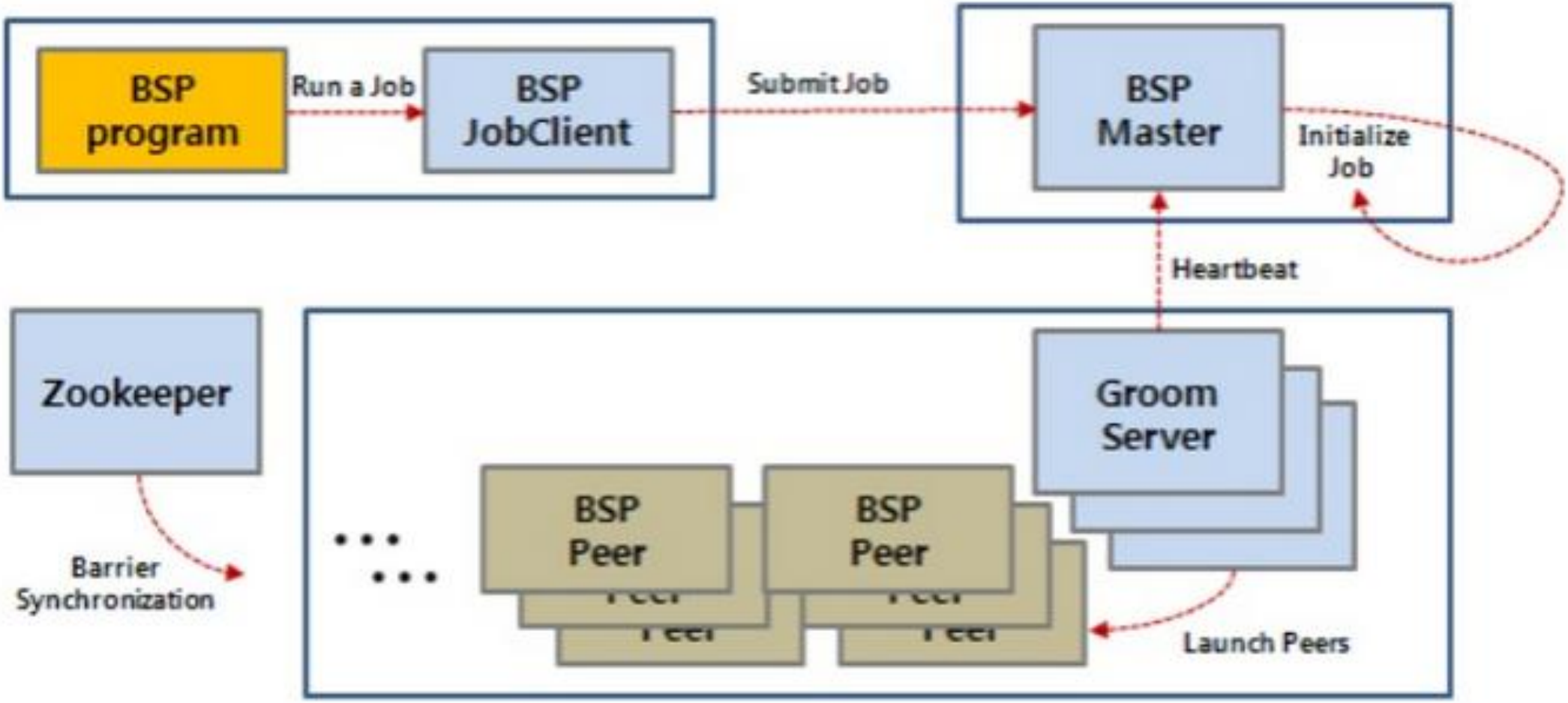
Apache Hama

BSP on Hadoop

BSP Implementation on Hadoop

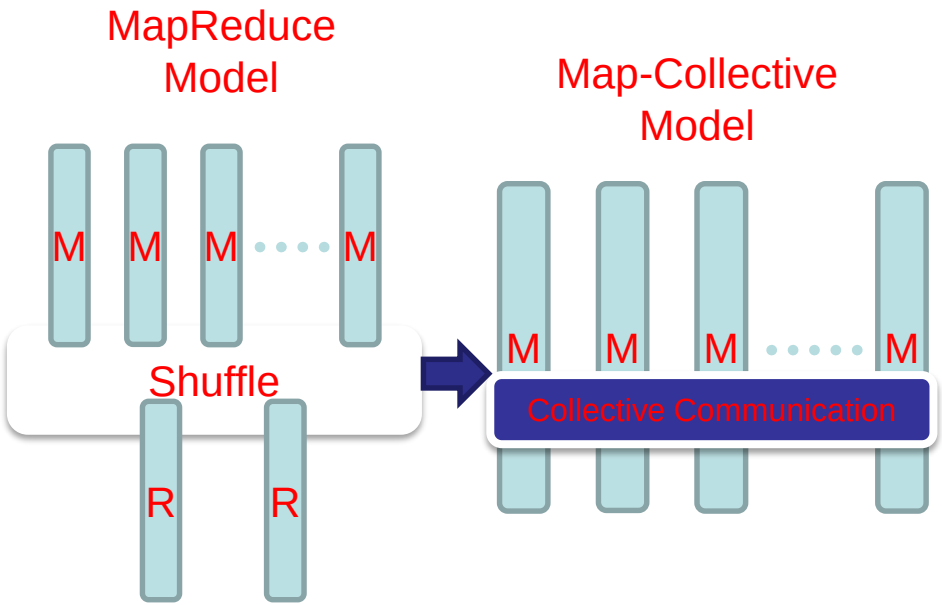
- Written in Java
- The BSP package is now available in the Hama repository.
  - Implementation available for Hadoop version greater than 0.20.x
  - Allows to develop new applications
- **Hadoop RPC** is used for BSP peers to communicate each other.
- Barrier Sync mechanism is helped by **Zookeeper**.

HAMA - Architecture

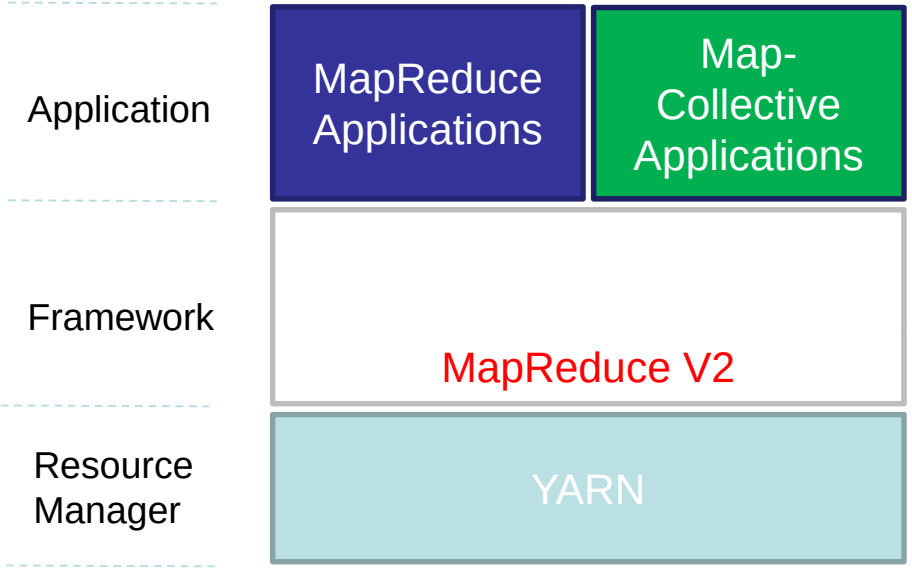


MR versus Giraph

Parallelism Model



Architecture



Lightning-Fast Cluster Computing

What is Spark?

Fast and Expressive Cluster Computing System
Compatible with Apache Hadoop

Up to **10×** faster on disk,
100× in memory

2-5× less code

Efficient

- General execution graphs
- In-memory storage

Usable

- Rich APIs in Java, Scala, Python
- Interactive shell

From McDonough Spark tutorial from Spark Summit 2013

Key Concepts

Write programs in terms of transformations on distributed datasets

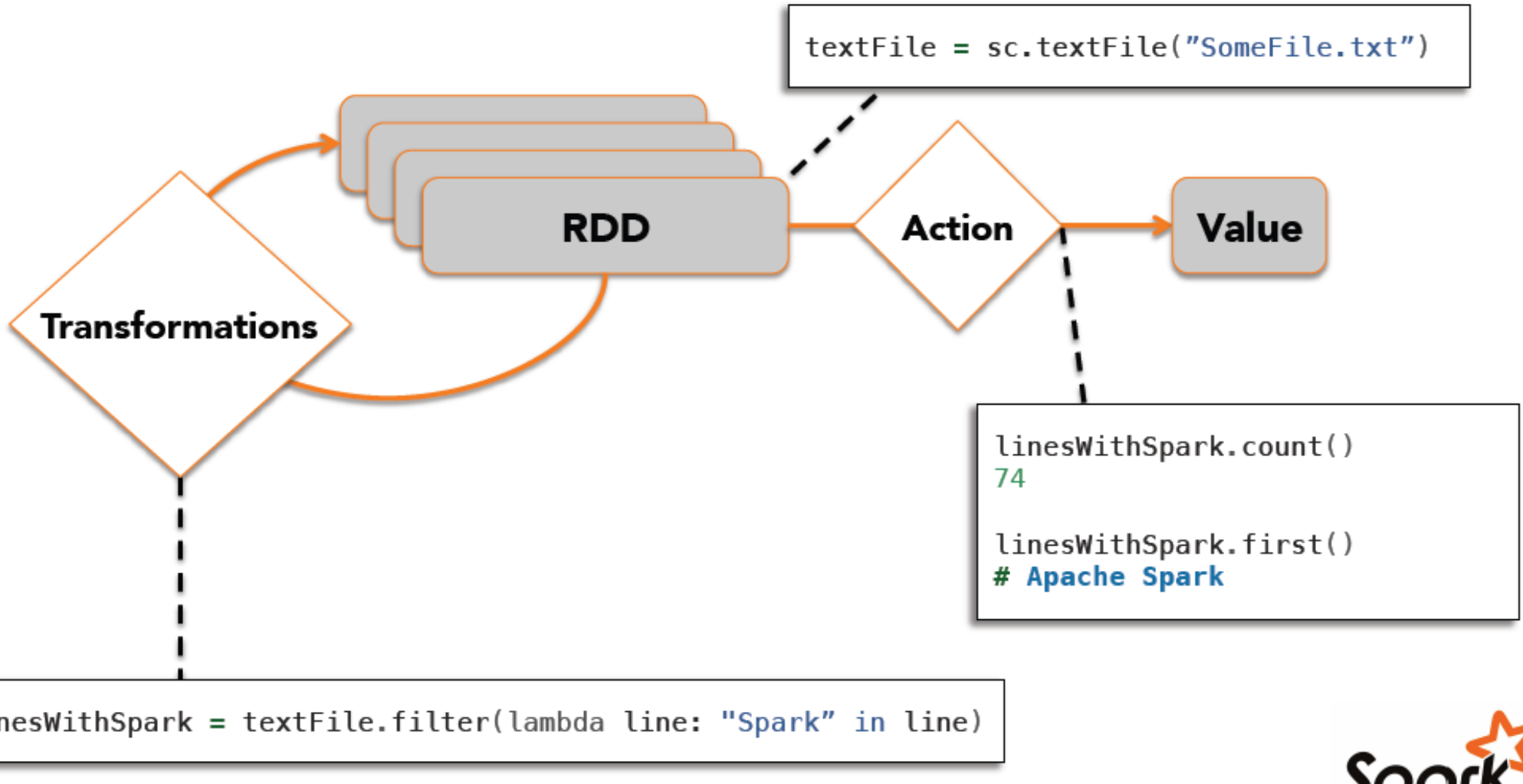
Resilient Distributed Datasets

- Collections of objects spread across a cluster, stored in RAM or on Disk
- Built through parallel transformations
- Automatically rebuilt on failure

Operations

- Transformations (e.g. map, filter, groupBy)
- Actions (e.g. count, collect, save)

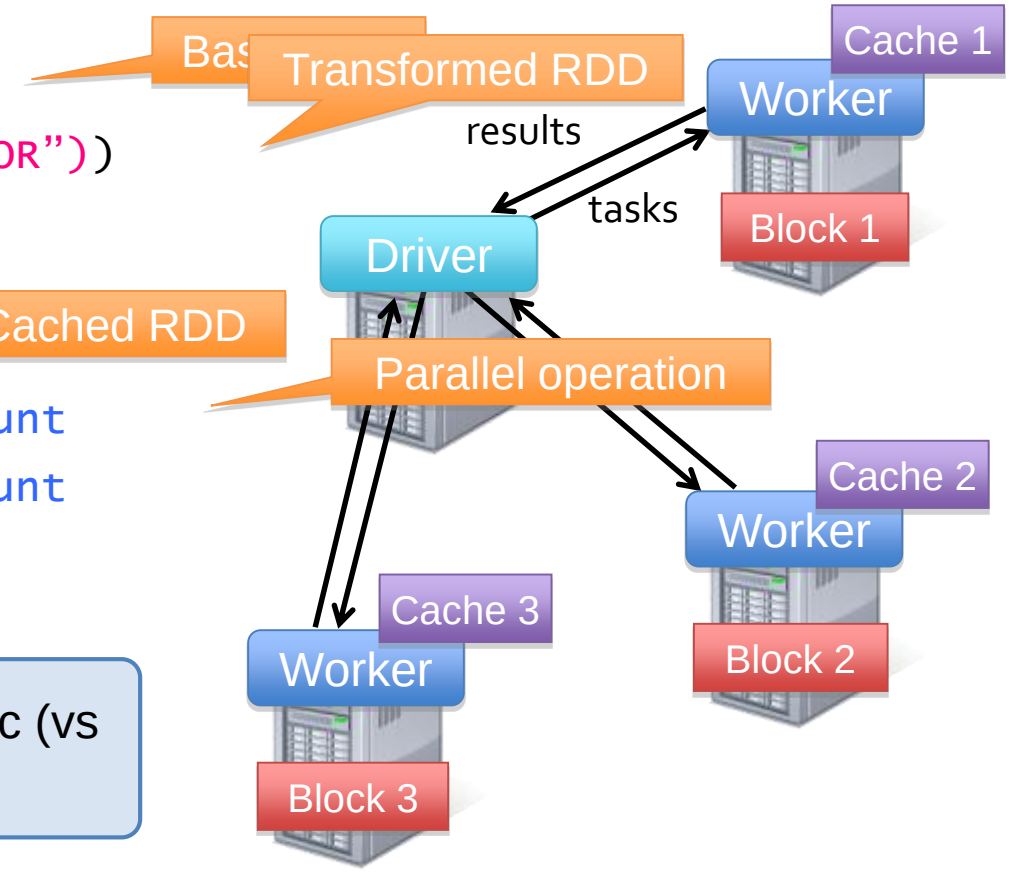
Working With RDDs



Spark Example: Log Mining

- Load error messages from a log into memory, then interactively search for various patterns

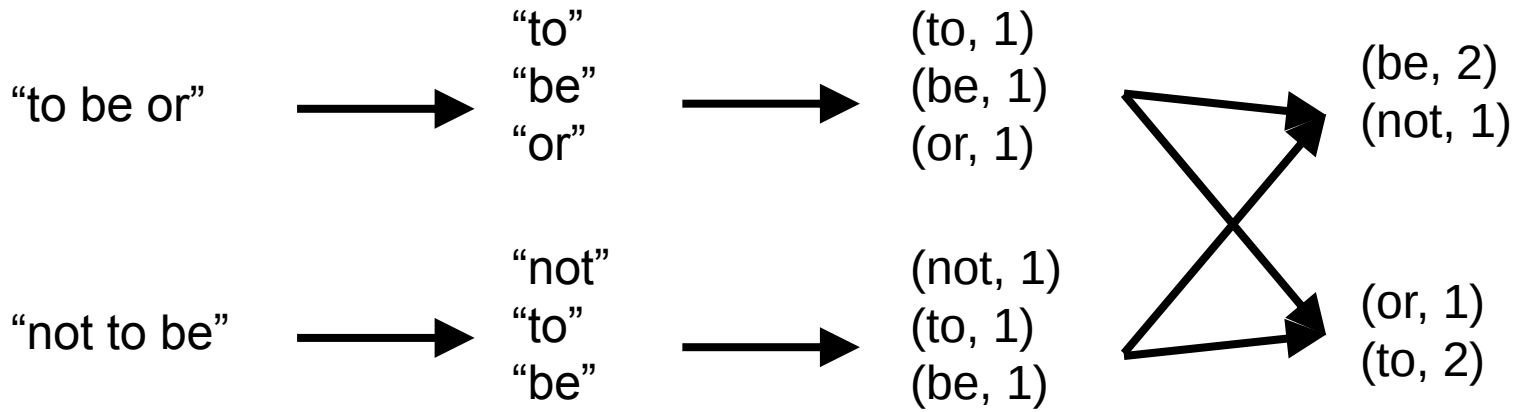
```
lines = spark.textFile("hdfs://...")
errors = lines.filter(_.startsWith("ERROR"))
messages = errors.map(_.split('\t')(2))
cachedMsgs = messages.cache()

cachedMsgs.filter(_.contains("foo")).count
cachedMsgs.filter(_.contains("bar")).count
. . .
```



Result: full-text search of Wikipedia in <1 sec (vs 20 sec for on-disk data)

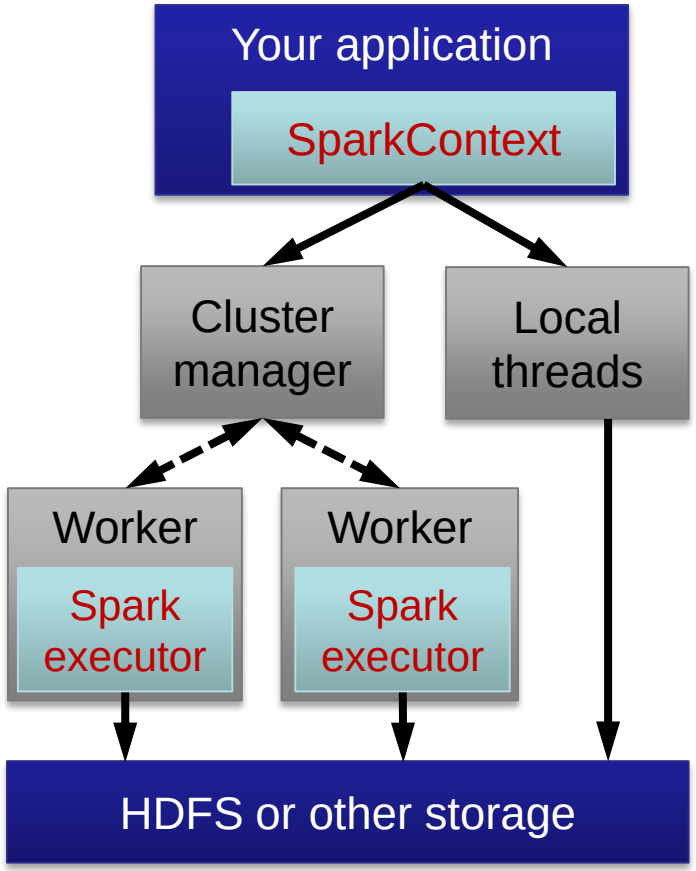
Example: Word Count

```
lines = sc.textFile(“hamlet.txt”)
counts = lines.flatMap(lambda line: line.split(“ ”)) \
              .map(lambda word: (word, 1)) \
              .reduceByKey(lambda x, y: x + y)
```

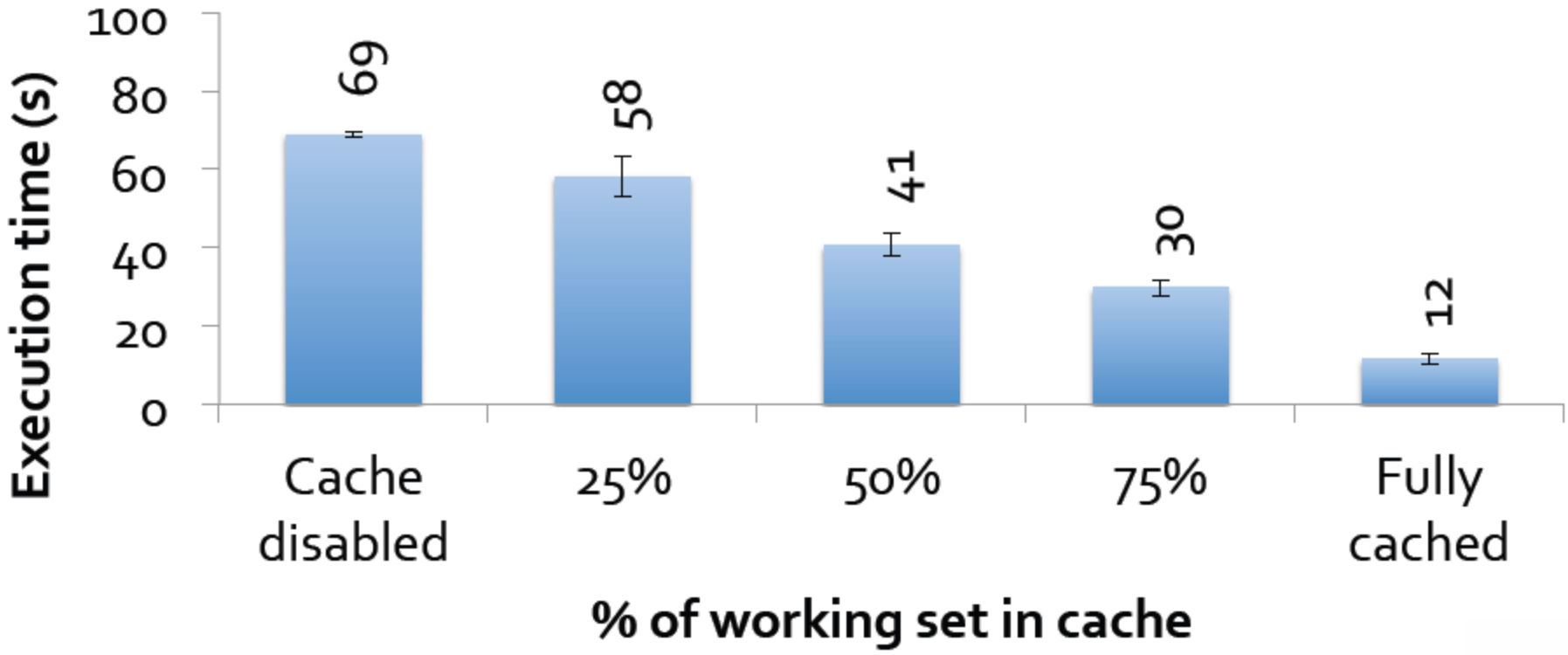


Software Components

- Spark runs as a library in your program (one instance per app)
- Runs tasks locally or on a cluster
  - Standalone deploy cluster, Mesos or YARN
- Accesses storage via Hadoop InputFormat API
  - Can use HBase, HDFS, S3, …



Benefit of In-Memory Computing



INSOFE
Inspire...Educate...Transform.

International School of Engineering

Plot 63/A, 1st Floor, Road # 13, Film Nagar, Jubilee Hills, Hyderabad - 500 033

For Individuals: +91-9502334561/63 or 040-65743991

For Corporates: +91-9618483483

Web: http://www.insofe.edu.in

Facebook: https://www.facebook.com/insofe

Twitter: https://twitter.com/Insofeedu

YouTube: http://www.youtube.com/InsofeVideos

SlideShare: http://www.slideshare.net/INSOFE

LinkedIn: http://www.linkedin.com/company/international-school-of-engineering