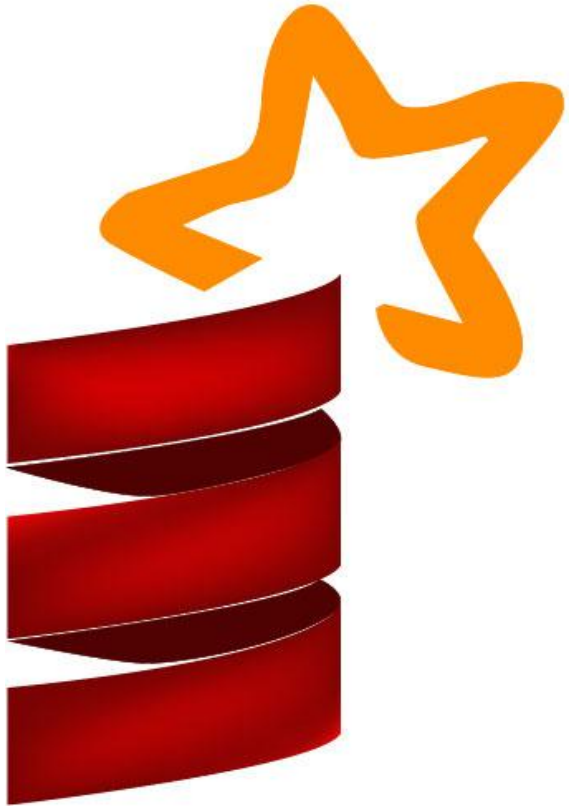
skillspeed

for the serious learner



Apache Spark and Scala

Module 5: Spark and Big Data

Course Topics

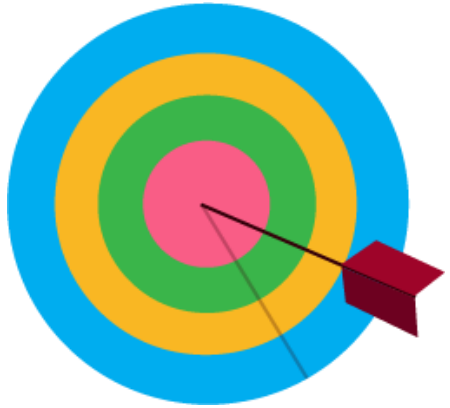
skillspeed
for the serious learner

Module 1 Getting Started / Introduction to Scala	**Module 2** Scala – Essentials and Deep Dive	**Module 3** Introducing Traits and OOPS in Scala	**Module 4** Functional Programming in Scala
Module 5 Spark and Big Data	**Module 6** Advanced Spark Concepts	**Module 7** Understanding RDDs	**Module 8** Shark, SparkSQL and Project Discussion

Session Objectives

In this session, you will understand:

- Analyze Batch Processing and Real-time Processing
- Understand Spark Ecosystem
- Analyze MapReduce Limitations
- Go through Spark History
- Analyze Spark Architecture
- Understand Spark and Hadoop Advantages
- Analyze benefits of Spark and Hadoop combined
- Install Spark



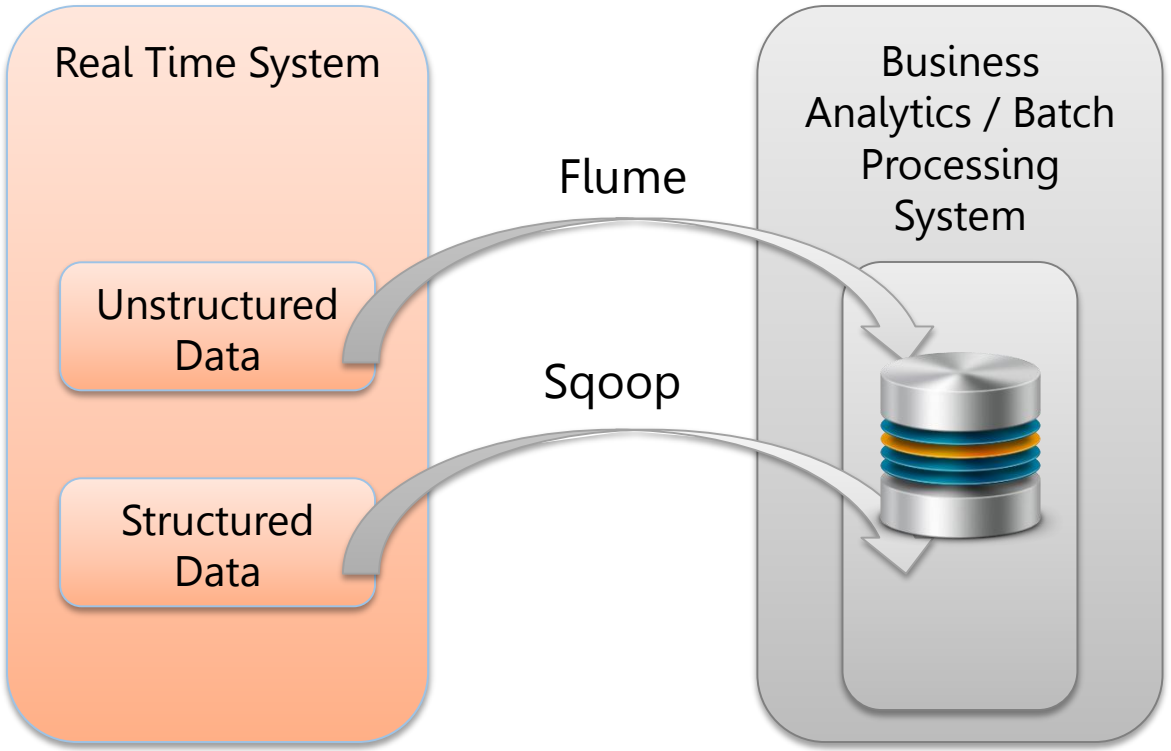
Bombay Stock Exchange – Big Data Case Study

- When Bombay Stock Exchange (the seventh largest stock exchange in the world, in terms of market capitalization) wanted to ramp up / scale up its operations, the company faced major challenges
- These challenges were in terms of exponential growth of data (read big data), need for complex analytics and managing information that was scattered across multiple and monolithic system
- DataMetica (a Mumbai / Pune based big data organization) suggested a 3 phased solution to BSE:
  - In the first phase, they created a POC which demonstrated how a Hadoop based Big data implementation can work for BSE
  - In the second phase, they worked with BSE to pick up the most critical business use cases (which had the maximum ROI for BSE) and implemented them
  - Finally in the third phase they delivered the complete solution in a multi-faced manner for a full fledged implementation
- That's how Hadoop got implemented at BSE in a cost effective and scalable fashion

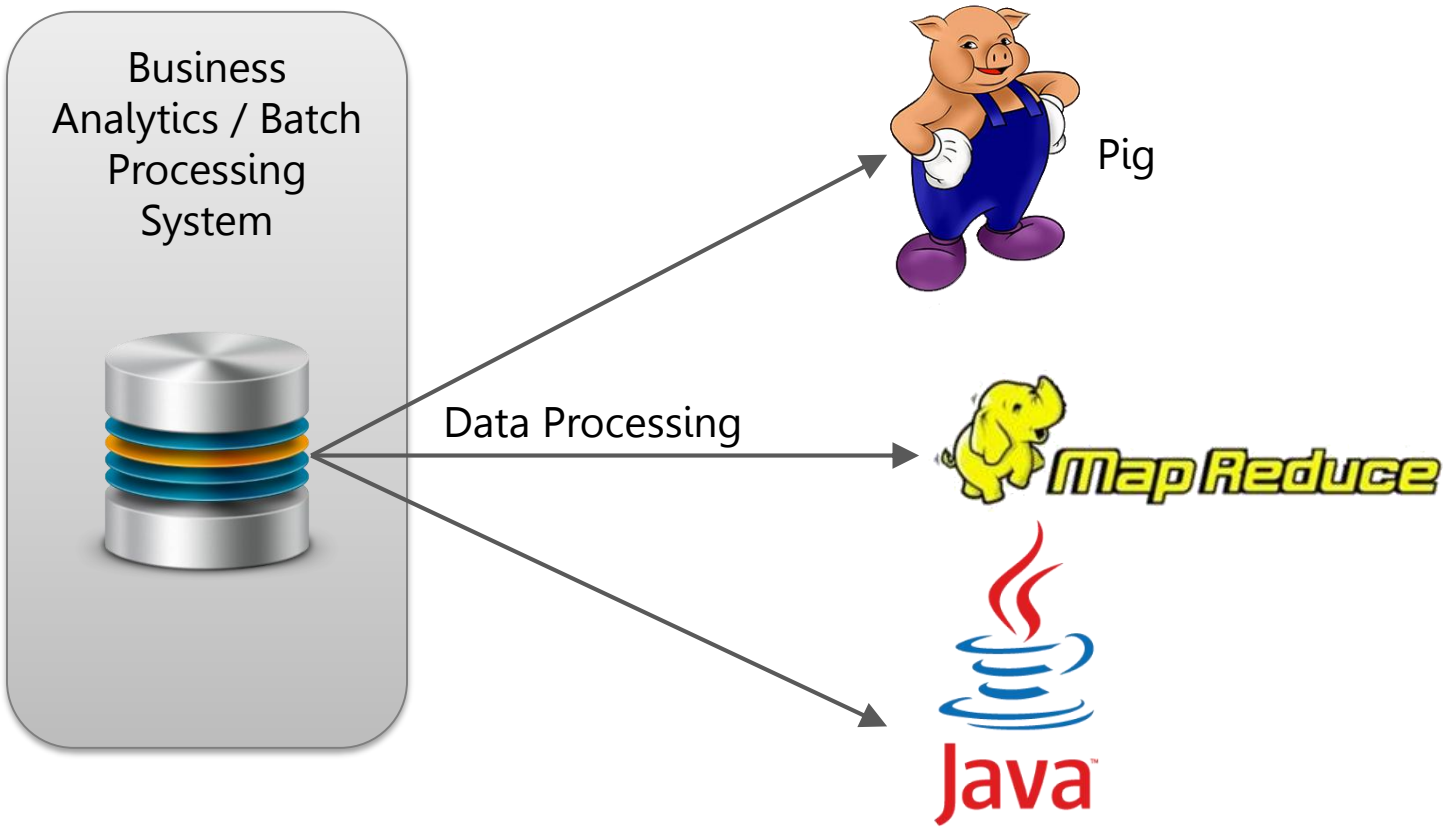
Batch Processing Phase / Life Cycle

- Processing transactions in a group or batch
- Following three phases are common to batch processing or business analytics project, irrespective of the type of data (structured or unstructured)

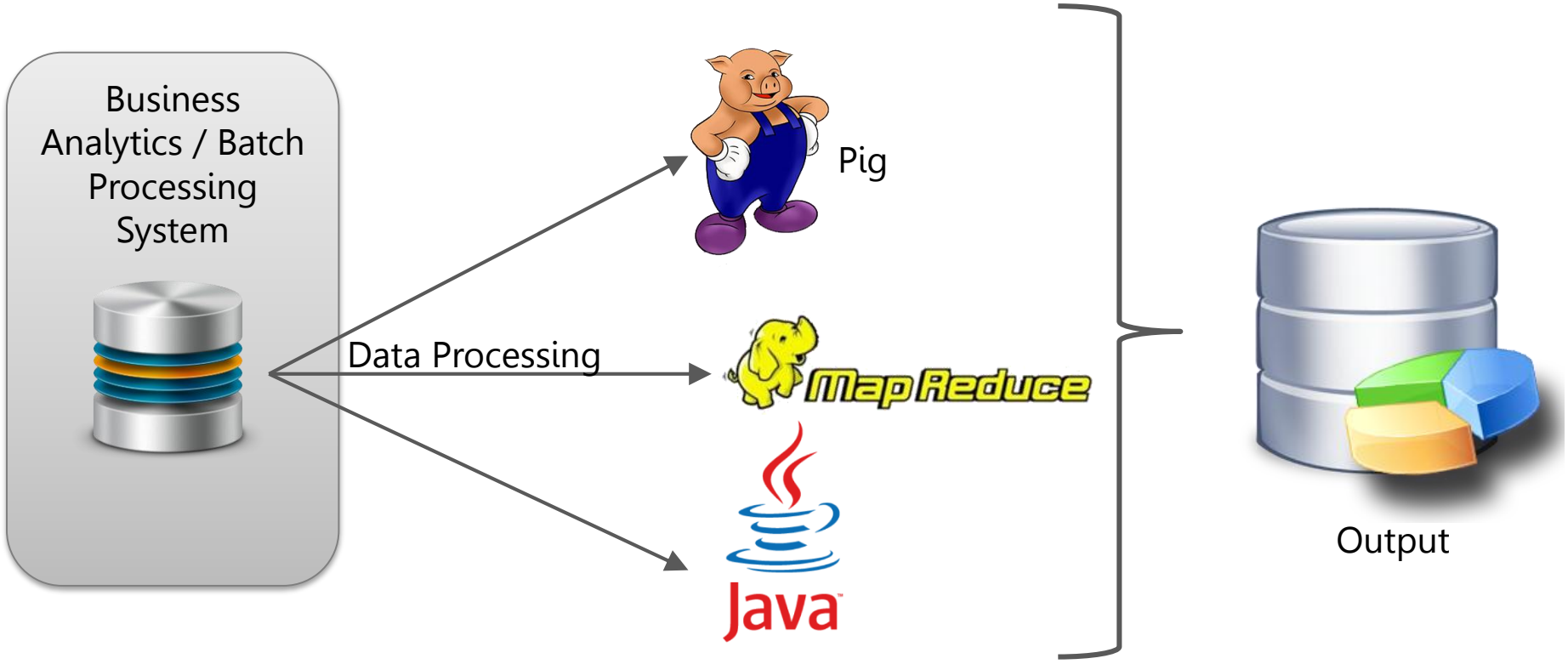
Data Collection



Data Preparation



Data Presentation



Real Time Analytics Examples – WindyGrid

- City of Chicago uses a MongoDB based Real Time Analytics Platform called WindyGrid
- This platform integrates unstructured data from various city departments to predict co-relations and outcomes in a proactive manner. E.g. How a rodent complaint will follow well within 7 days of a garbage complaint

WindyGrid in Practice

Real Time Analytics Examples – WindyGrid (Cont'd)

- With MongoDB based system, WindyGrid created a central nervous system for Chicago, helping improve services, cut costs, and create a more livable city
- By pulling together 311 and 911 calls, tweets, and bus locations, the city can better manage traffic and incidents and get streets cleaned and opened up more quickly
- The city of Chicago collects more than seven million rows of data every day. With MongoDB's flexible data schema, This system doesn't need to worry about unwieldy and constantly changing schema requirements

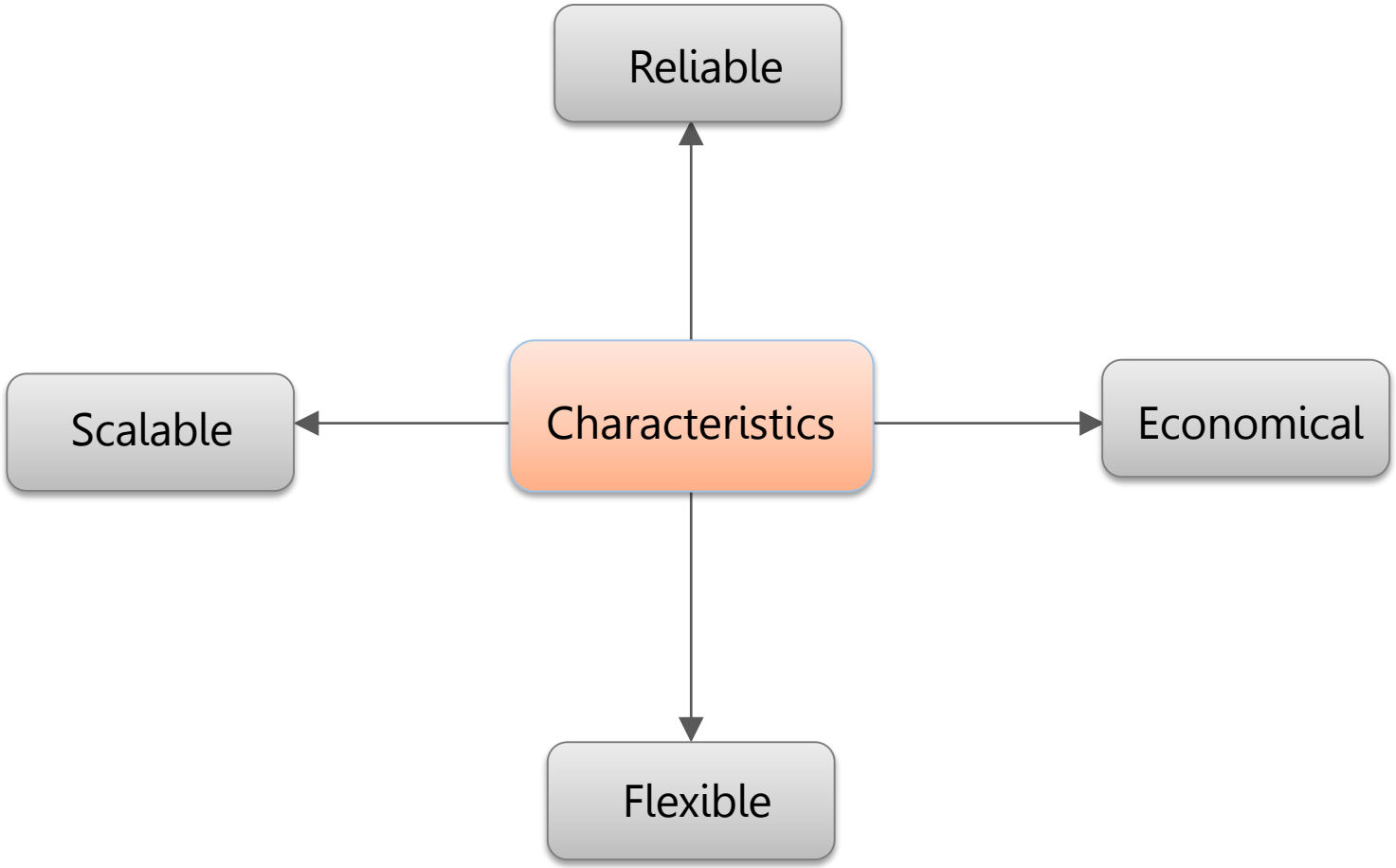
The next step for WindyGrid is building an open-source, predictive analytics system called the SmartData Platform to anticipate problems before they occur, and propose solutions in an even faster manner

What is Hadoop?

Apache Hadoop is a framework that allows the distributed processing of large data sets across clusters of commodity computers using a simple programming mode

It is an Open-source Data Management with scale-out storage and distributed processing

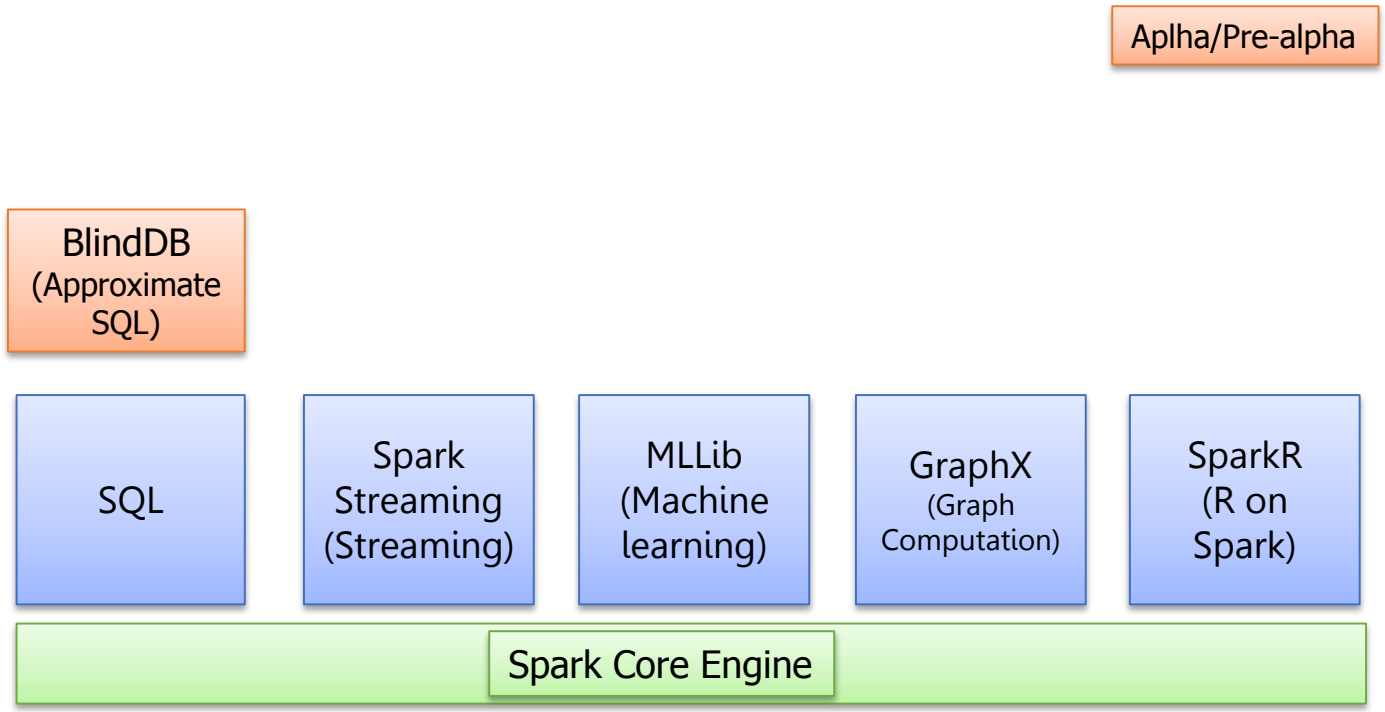
Hadoop Key Characteristics



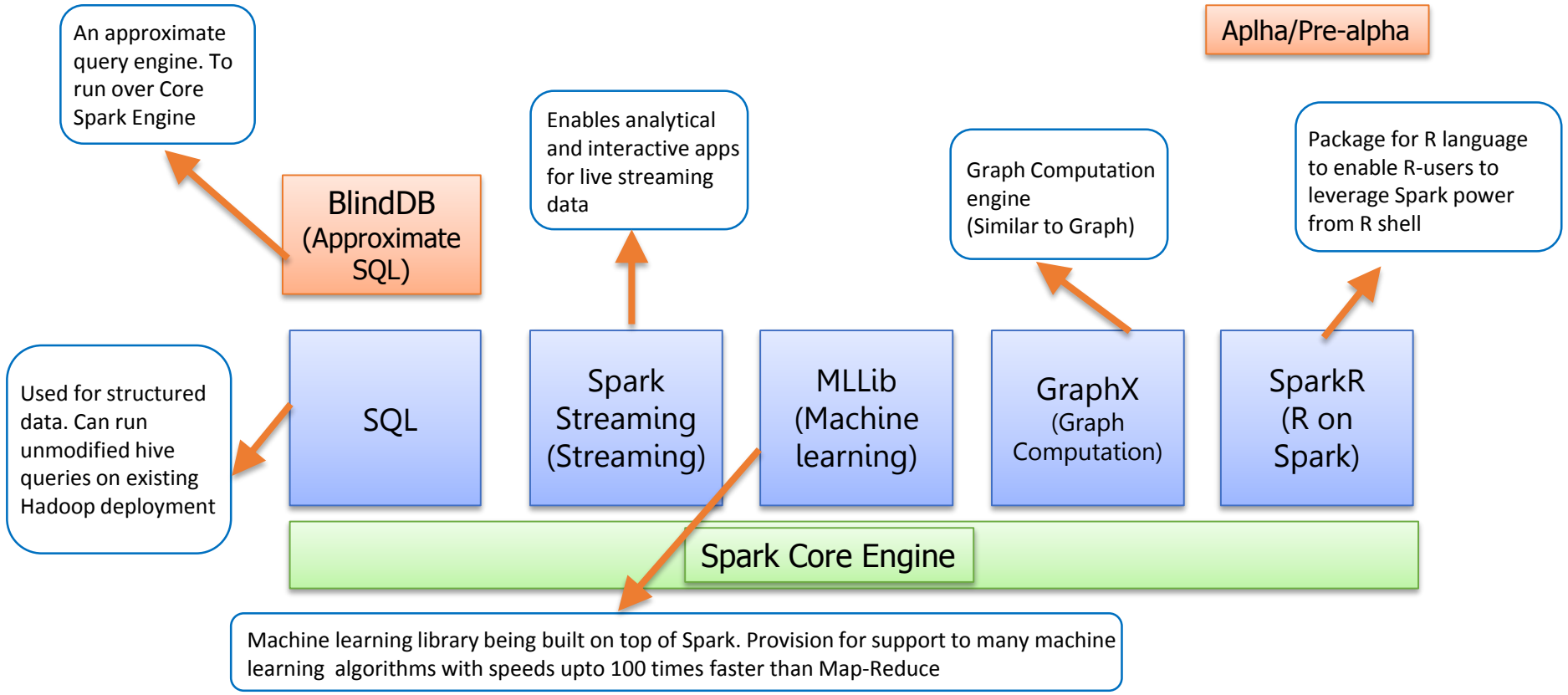
What is Spark?

- Apache Spark is a fast and general engine for large-scale data processing
- Apache Spark is a general-purpose cluster in-memory computing system
- It is used for fast data analytics
- It abstracts APIs in Java, Scala and Python, and provides an optimized engine that supports general execution graphs
- Provides various high level tools like Spark SQL for structured data processing, Mlib for Machine Learning and more

Spark Ecosystem



Spark Ecosystem (Cont'd)



Spark Ecosystem (Cont'd)

- Spark Core Engine
  - The core engine for entire Spark framework
  - Provides utilities and architecture for other components
- Spark SQL
  - Spark SQL is the newest component of Spark and provides a SQL like interface
  - Used for Structured data
  - Can expose many datasets as tables
  - Spark SQL is tightly integrated with the various spark programming languages like hive
- Spark Streaming
  - Spark Streaming is an extension of the core Spark API that enables scalable, high-throughput, fault-tolerant stream processing of live data streams
  - A good alternative of Storm

Spark Ecosystem (Cont'd)

- BlinkDB
  - An approximate query engine. To run over Core Spark Engine
  - Accuracy trade-off for response time
- MLLib
  - Machine learning library being built on top of Spark
  - Provision for support to many machine learning algorithms with speeds upto 100 times faster than Map-Reduce
  - Mahout is also being migrated to MLLib
- GraphX
  - Graph Computation engine (Similar to Giraph)
  - Combines data-parallel and graph-parallel concepts
- SparkR
  - Package for R language to enable R-users to leverage Spark power from R shell

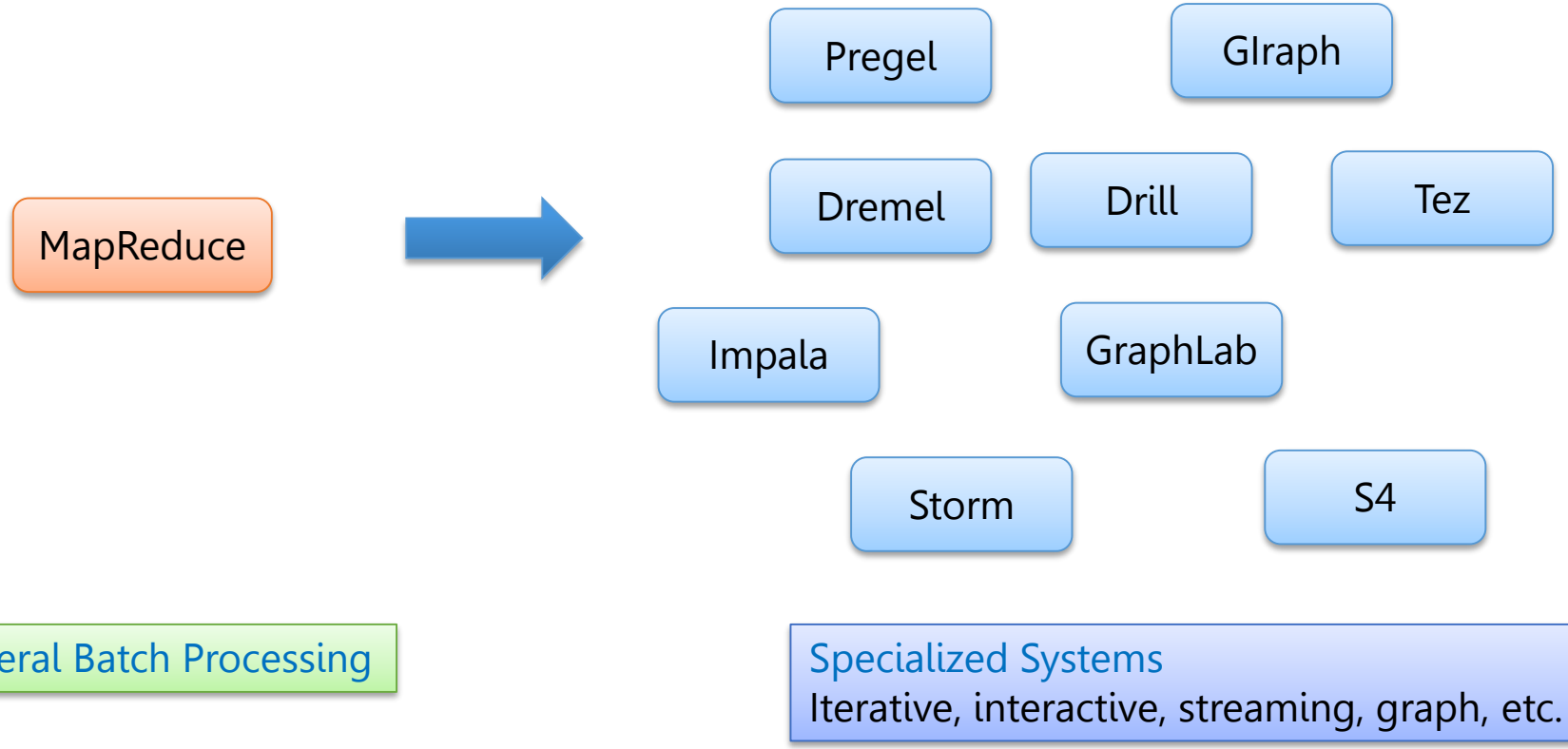
Why Spark?

- Spark exposes a simple programming layer which provides powerful caching and disk persistence capabilities
- The Spark framework can be deployed through Apache Mesos, Apache Hadoop via Yarn, or Spark's own cluster manager
- Spark framework is polyglot – Can be programmed in several programming languages (Currently Scala, Java and Python supported)
- Has super active community
- Spark fits well with existing Hadoop ecosystem
  - Can be launched in existing YARN Cluster
  - Can fetch the data from Hadoop 1.0
  - Can be integrated with Hive

Brief History: M/R Limitations

- Map Reduce is a very powerful programming paradigm, but it has some limitations:
  - Difficult to Program an algorithm directly in Native Map Reduce
  - Performance bottlenecks, specifically for small batch not fitting the use cases
  - Many categories of algorithms not supported (e.g. iterative algorithms, asynchronous algorithms etc.)
- In short, MR doesn't compose well for large applications
- We are forced to take "hybrid" approaches many times
- Therefore, many specialized systems evolved over a period of time as workarounds

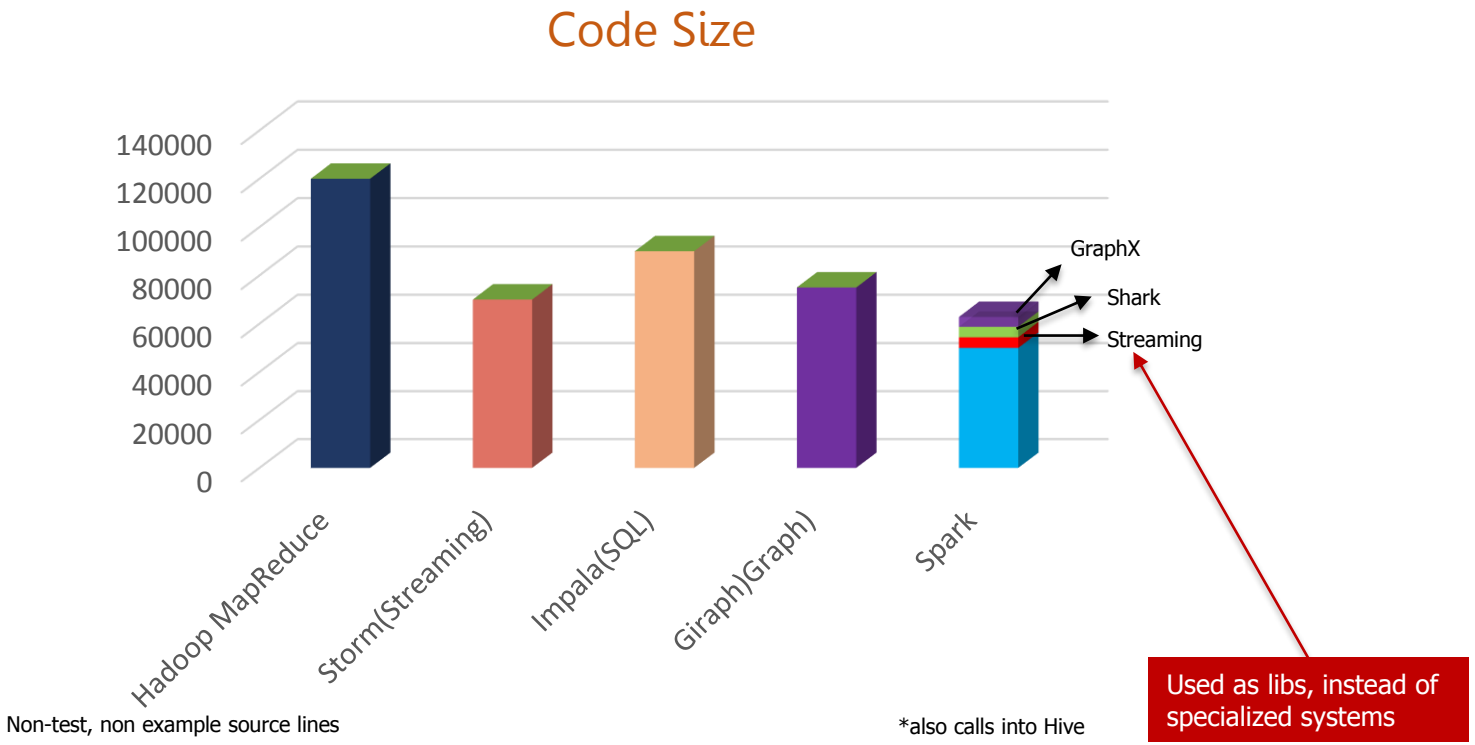
Brief History: Evolution of Specialized Systems



Brief History: Spark

- Unlike other evolved specialized systems, Spark's design goal is to generalize Map Reduce concept to support new apps within same engine
- Two reasonably small additions are enough to express the previous models:
  - Fast data sharing (For Faster Processing)
  - General DAGs (For Lazy Processing)
- This allows for an approach which is more efficient for the engine, and much simpler for the end users

Brief History: Spark Key Points



The State of Spark, and where we're going next
Matei Zaharia
Spark Summit(2013)
you.be/nU6v02EJAb4

Brief History: Spark Key Points (Cont'd)

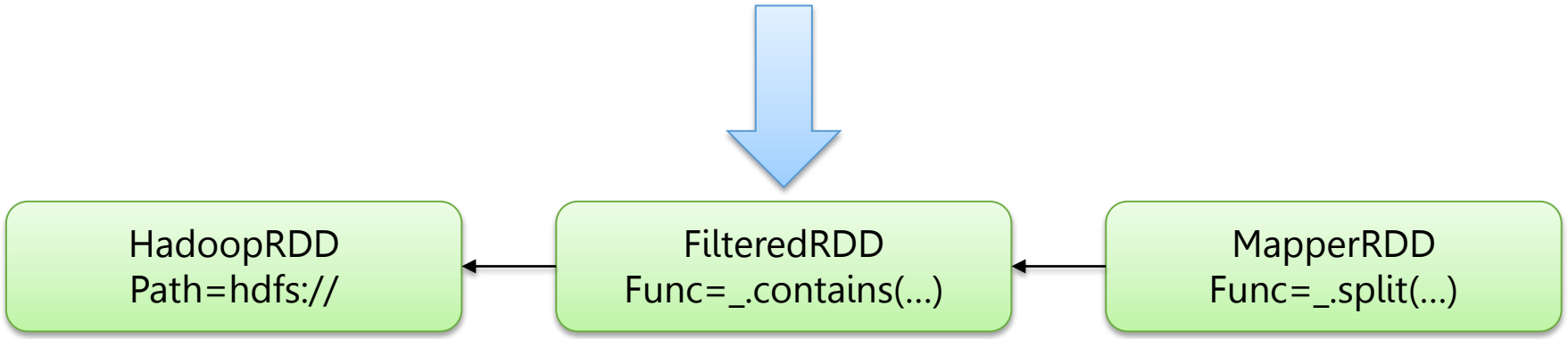
skillspeed
for the serious learner

RDD Fault Tolerance

RDDs track the series of transformation used to build them (their lineage) to recomputed lost data

Example:

```
messages=textFile(...).filter(_.contains("error"))
                      .map(_.split('\t')(2))
```



Spark in Industry

Spark Advantages

- Easier APIs
- Python, Scala, Java

EASE OF DEVELOPMENT

- RDDs
- DAGs Unify Processing

IN-MEMORY PERFORMANCE

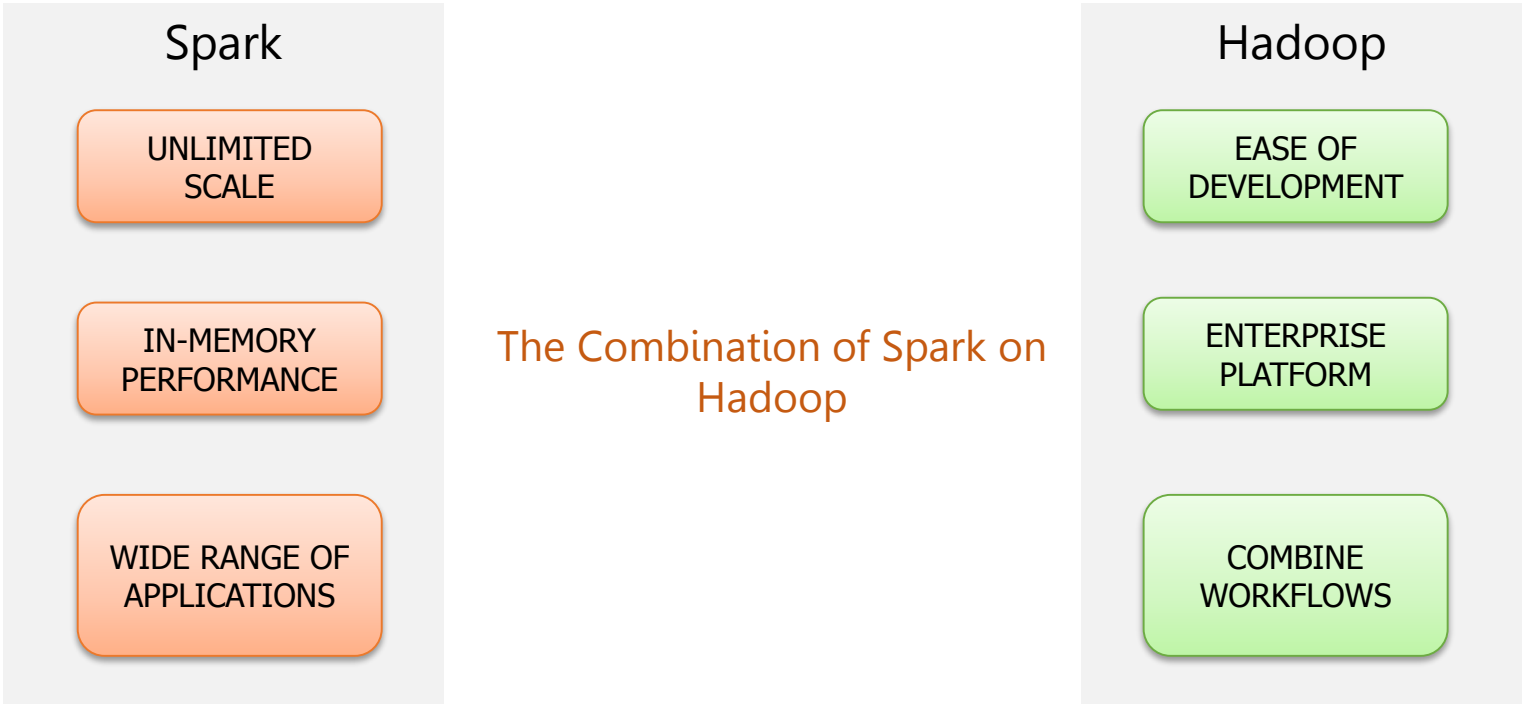
- SQL, ML, Streaming, GraphX

COMBINE WORKFLOWS

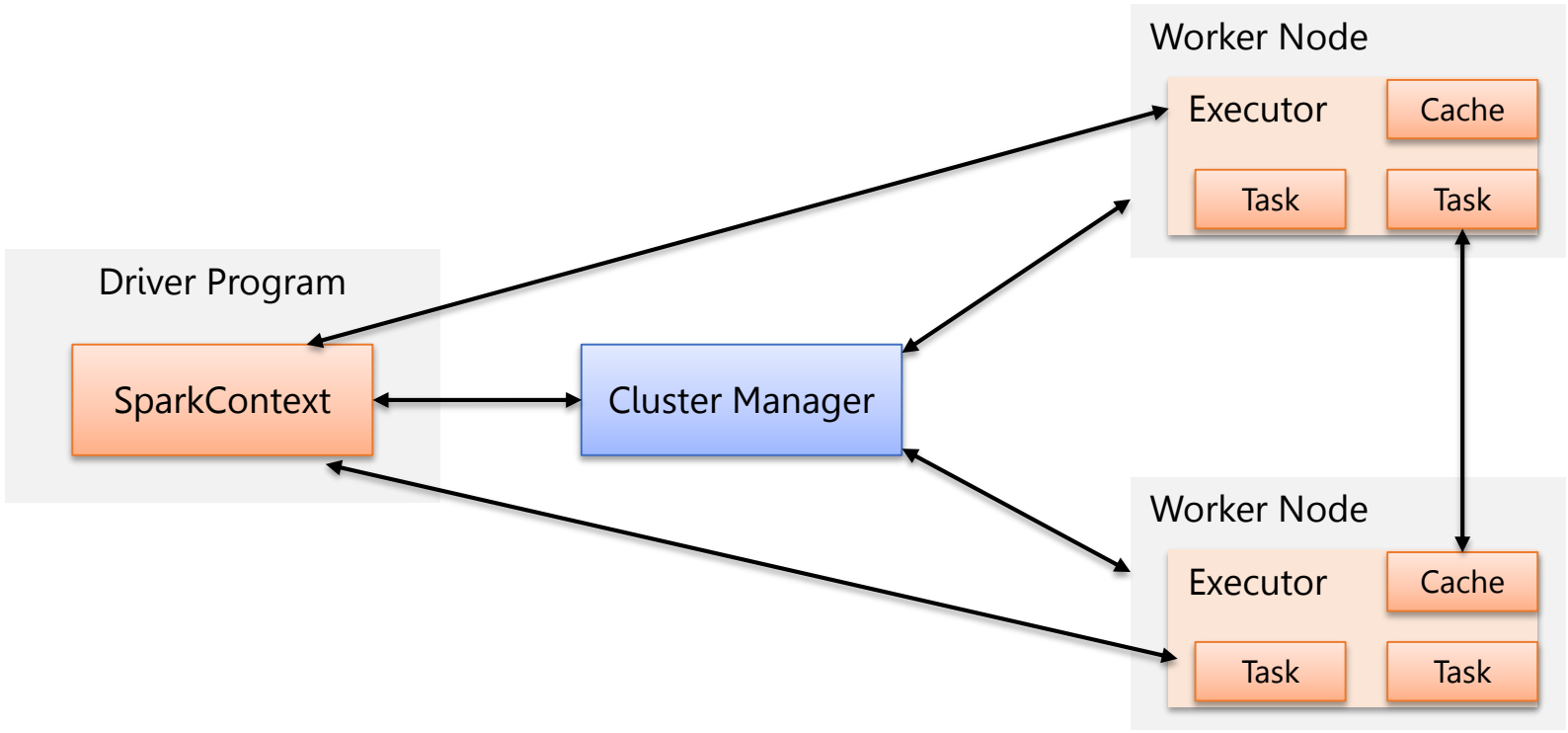
Spark + Hadoop

skillspeed
for the serious learner

Operational Applications Augmented by In-Memory Performance



Spark Cluster Manager



Spark Architecture – SparkContext

- Spark apps run as separate set of process on a cluster
- All of the distributed process is coordinated by SparkContext object in the driver program
- SparkContext object then connects to one type of cluster Manager (Standalone/Yarn/Mesos) for resource allocation across cluster
- Cluster Managers provide Executors, which are essentially JVM process to run the logic and store app data
- Then, the SparkContext object sends the application code (jar files/python scripts) to executors
- Finally, the SparkContext executes tasks in each executor

SBT Demo

- Sbt stands for Scala Build Tool
- A build tool provides facility to compile, run, test, package your projects
- SBT is a modern build tool. While it is written in Scala and provides many Scala conveniences, it is a general purpose build tool

Why SBT?

- Full Scala language support for creating tasks
- Continuous command execution
- Launch REPL in project context

Note: Given SBT installation guide separately

SBT Demo (Cont'd)

skillspeed
for the serious learner

SBT console:

```
skillspeed@ubuntuvms:~$ sbt
[info] Set current project to skillspeed (in build file:/home/skillspeed/)
>
```

Testing in the Console:

- SBT can be used both as a command line script and as a build console
- We'll be primarily using it as a build console, but most commands can be run standalone by passing the command as an argument to SBT, e.g.

```
skillspeed@ubuntuvms:~/test$ sbt test
[info] Set current project to test (in build file:/home/skillspeed/test/)
[info] Updating {file:/home/skillspeed/test/}test...
[info] Resolving org.fusesource.jansi#jansi;1.4 ...
[info] Done updating.
[success] Total time: 3 s, completed 6 Aug, 2015 11:10:16 AM
```

SBT Demo (Cont'd)

SBT allows you to start a Scala REPL with all your project dependencies loaded. It compiles your project source before launching the console, providing us a quick way to bench test our parser

```
skillspeed@ubuntuvms:~/test$ sbt
[info] Set current project to test (in build file:/home/skillspeed/test/)
> console
[info] Starting scala interpreter...
[info]
Welcome to Scala version 2.10.4 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_80
).
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Simple Spark Apps: Word Count

This simple program provides a good test case for parallel processing, since it:

- Requires a minimal amount of code
- Demonstrates use of both symbolic and numeric values
- Isn't many steps away from search indexing

```
val f = sc.textFile("README.md")

val wc = f.flatMap(l => l.split(" ")).map(word => (word, 1)).reduceByKey(_ + _)

wc.saveAsTextFile("wc_out")
```

Using Hadoop as Storage

- Spark can use Hadoop as Storage
  - Spark is NOT limited to HDFS only for it's storage needs
  - HDFS provides distributed storage of large datasets
  - High Availability is assured natively through HDFS
  - No extra software installation is required
  - Compatible with Hadoop 1.x also. Using HDFS as storage doesn't require Hadoop 2.x
  - Data Loss during computation is handled by HDFS itself

Using Hadoop as Execution Engine

- Spark can use Hadoop as execution engine
  - Spark can be integrated with Yarn for it's execution
  - Spark can be used with other engines (like Mesos, Spark Clsuter manager) also
  - Yarn integration automatically provides processing scalability to Spark
  - Spark needs Hadoop 2.0+ versions in order to use it for execution
  - Every node in Hadoop cluster need Spark also to be installed
  - Using Hadoop cluster for Spark processes, requires RAM upgrading of data nodes
  - The integration distribution of Spark is quite new and still in the process of stabilization

Questions

skillspeed
for the serious learner