PYSPARK IN PRACTICE

PySpark as a stack for IoT Data Analysis

Agenda

- 4 Stage Solution Architecture for IoT
- Unique Challenges for IoT Analytics
- Short introduction of Spark and PySpark
- Data structures in Spark
- Transformations
- Actions
- Walk through of some code
- Industry use cases
- NCR’s Predictive Maintenance using Spark
- Questions

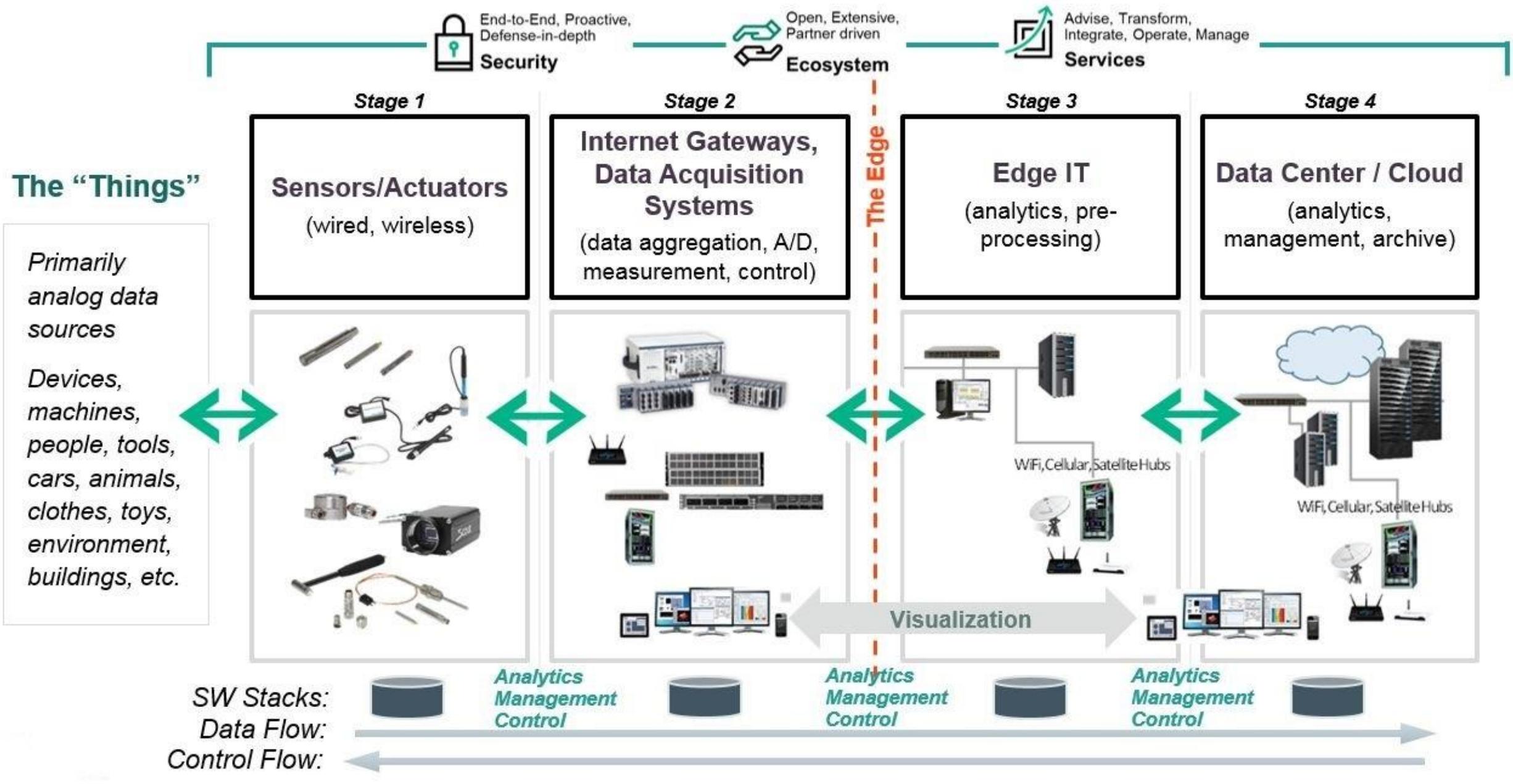
About Me....

Swaroop Kallakuri

Enterprise Architect | Data Scientist | Project Manager

Confidential • Stanford University Graduate School of Business

The 4 Stage IoT Solutions Architecture



Source: https://techbeacon.com/4-stages-iot-architecture

Unique Challenges for IoT Analytics

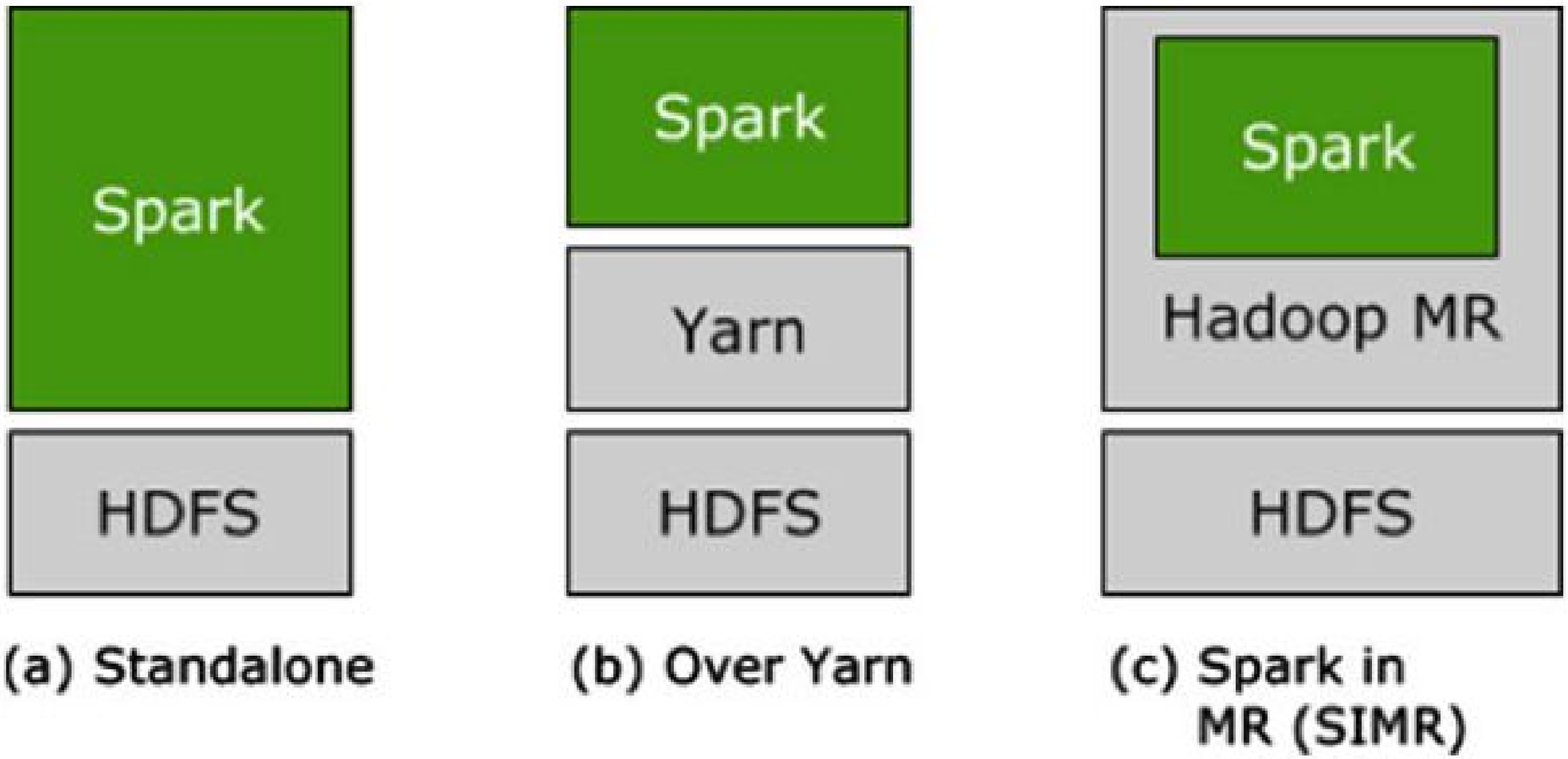
IoT projects generally fall into four categories:

1. Operational efficiencies (cited by 45% of ESG research respondents), such as preventative and predictive maintenance.
2. Better and differentiated customer service (39%), such as connectivity to enhance products like connected cars, consumer appliances, and industrial/construction equipment.
3. Creation of new products and services (38%), via visibility into product or equipment performance and diagnostics.
4. Development of new business models (26%), including shifting from product sales to “as-a-service” models.

What is Apache Spark?

- A fast and general engine for large-scale data processing
- An Open-source cluster computing framework
- End-to-End Analytics platform
- Developed to overcome limitations of Hadoop/Map Reduce
- Runs from a single desktop or a huge cluster
- Iterative, interactive or stream processing
- Supports multiple languages –Scala, Python, R, Java
- Major companies like Amazon, eBay, Yahoo use Spark.

Spark Execution Models

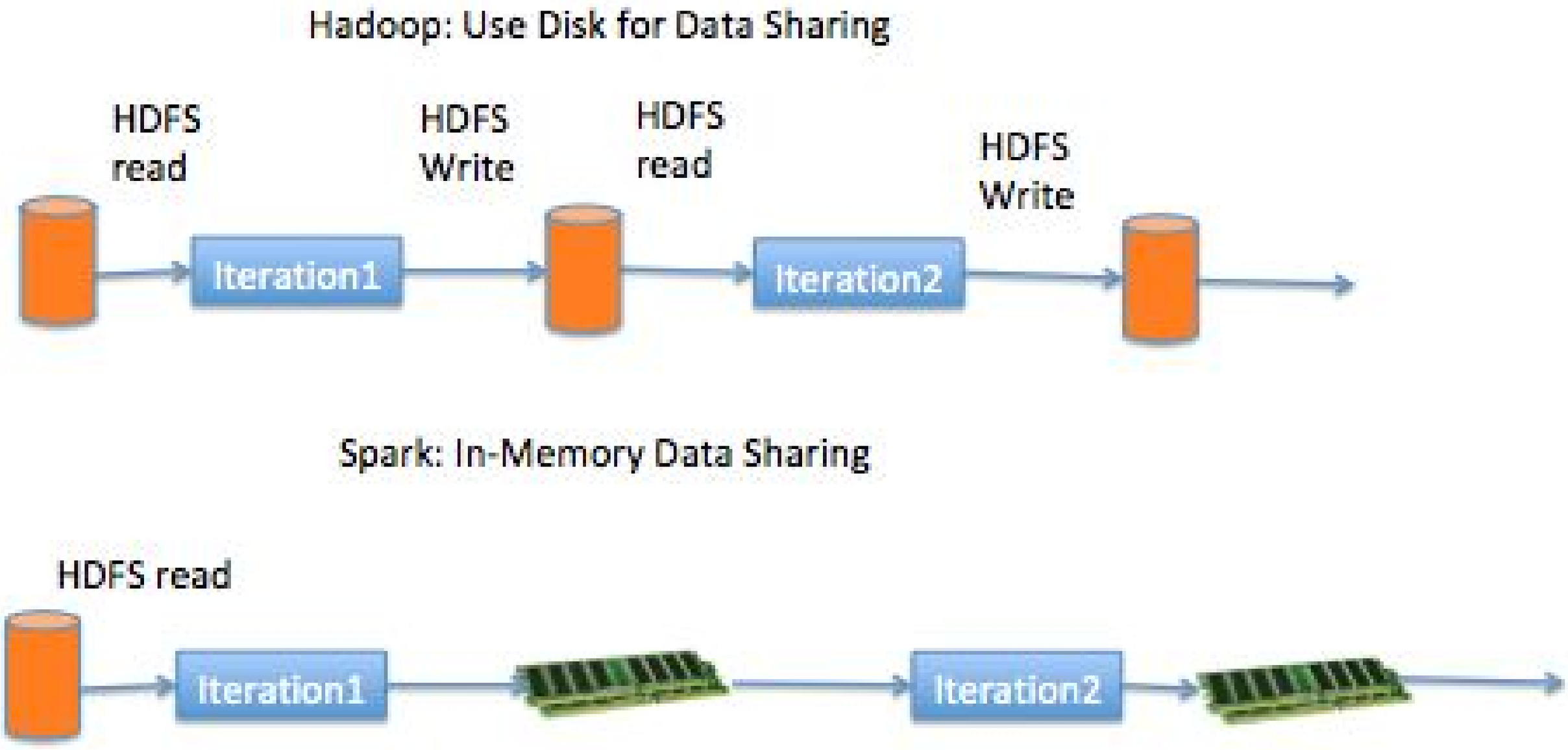


(a) Standalone (b) Over Yarn (c) Spark in MR (SIMR)

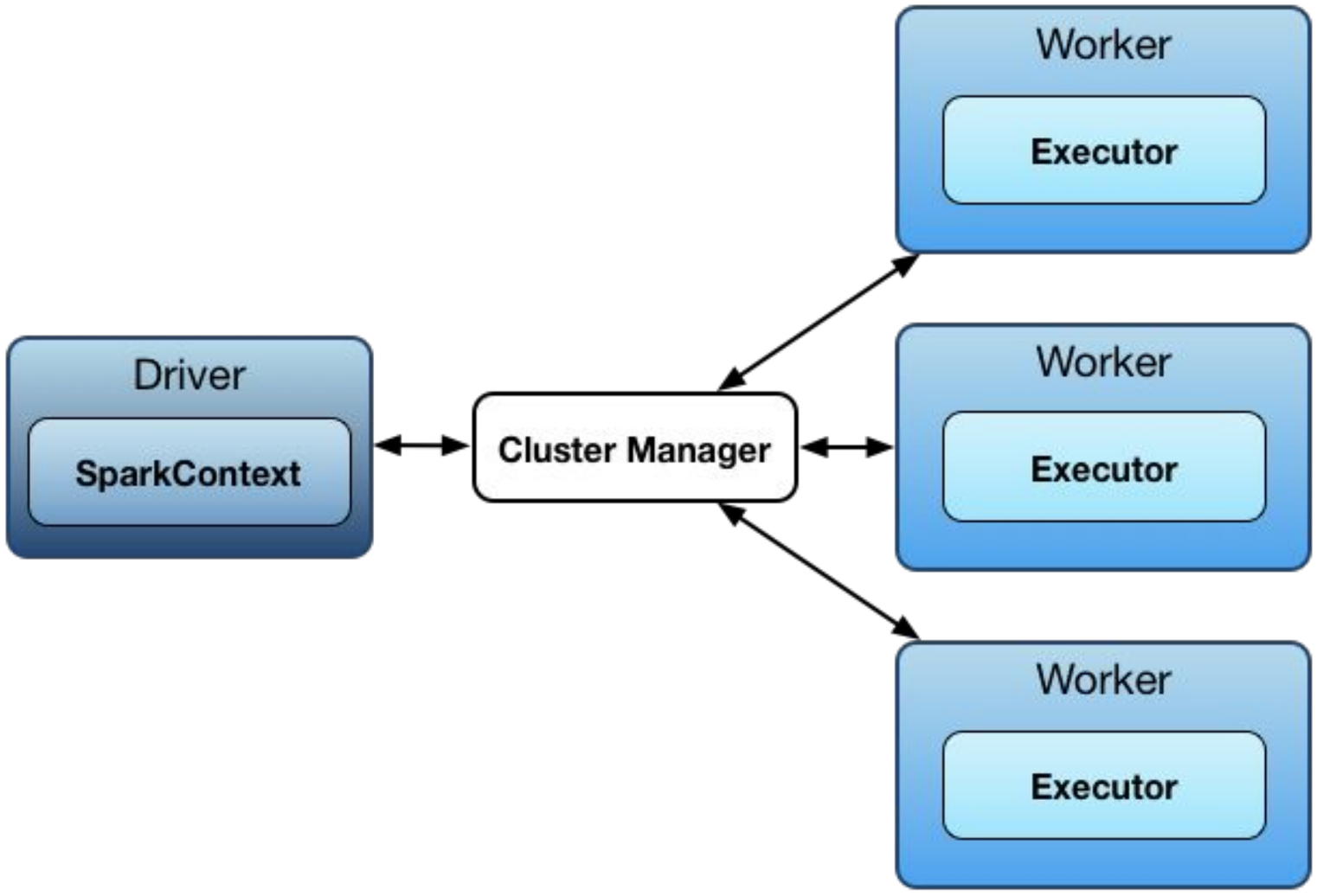
There are three ways Apache Spark can run :

1. **Standalone** – The Hadoop cluster can be equipped with all the resources statically and Spark can run with MapReduce in parallel. This is the simplest deployment.
2. **On Hadoop YARN** – Spark can be executed on top of YARN without any pre-installation. This deployment utilizes the maximum strength of Spark and other components.
3. **Spark In MapReduce (SIMR)** – If you don't have YARN, you can also use Spark along with MapReduce. This reduces the burden of deployments.

Spark Uses Memory instead of Disk

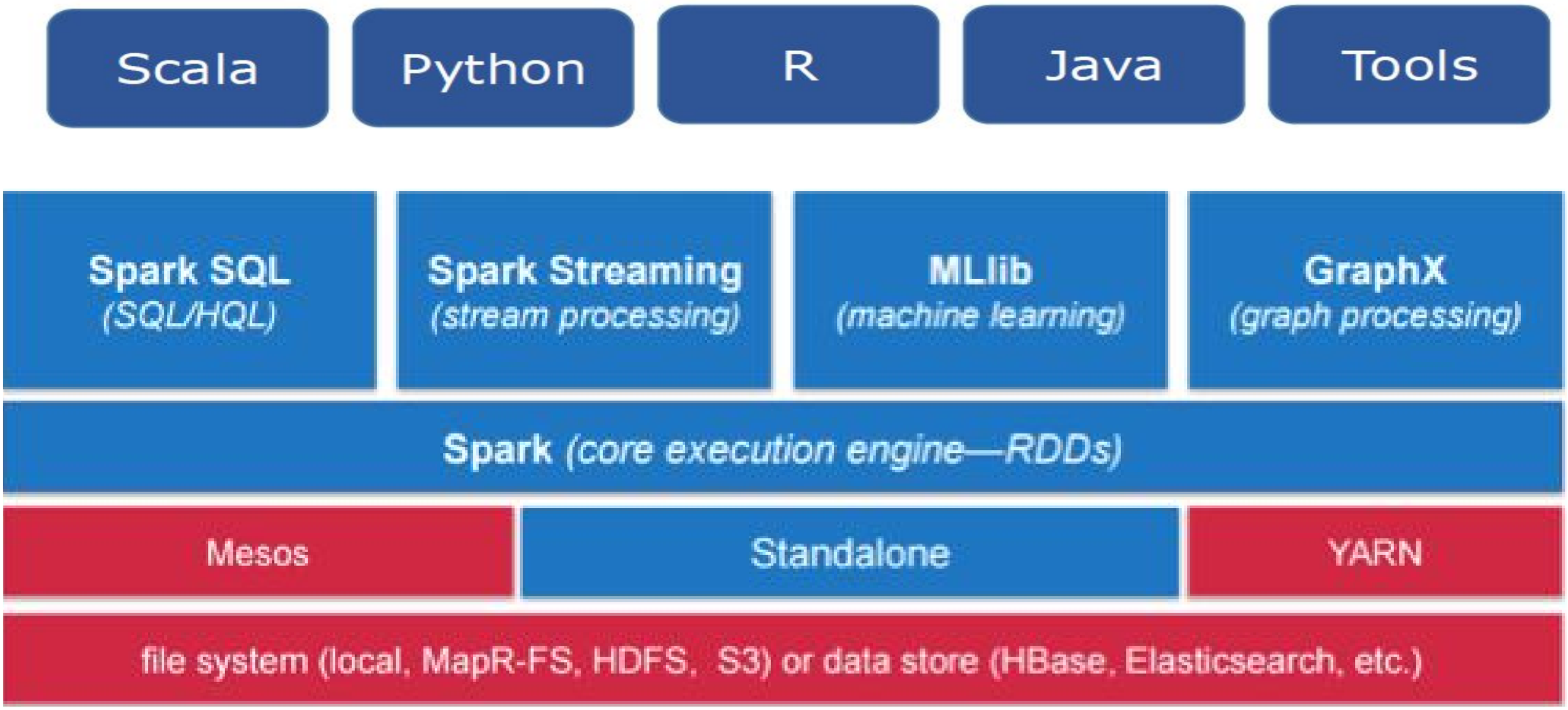


Spark Architecture



- Spark uses a **Master/Worker** Architecture.
- There is a Driver talks to a single coordinator called **Master / Cluster Manager** that manages **workers** in which the **executors** run.

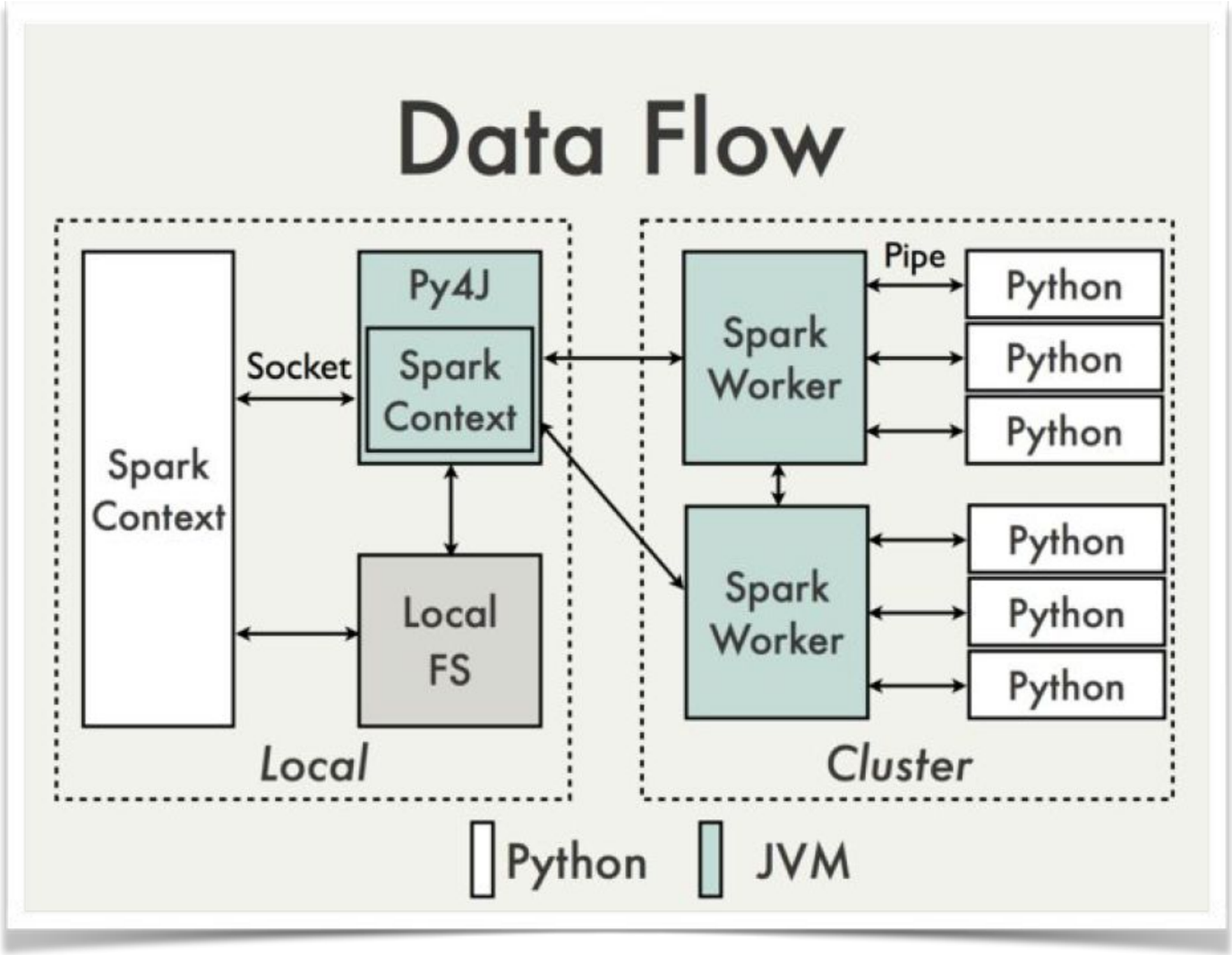
Spark Framework



One Solution in Spark

- A new general framework, which solves many of the shortcomings of MapReduce
- It capable of leveraging the Hadoop ecosystem, e.g. HDFS, YARN, HBase, S3, …
- Has many other workflows, i.e. join, filter, flatMapdistinct, groupByKey, reduceByKey, sortByKey, collect, count, first…
  - (around 30 efficient distributed operations)
  - in-memory caching of data (for iterative, graph, and machine learning algorithms, etc.)
- Native Scala, Java, Python, and R support
- Supports interactive shells for exploratory data analysis
- Spark API is extremely simple to use
- Developed at AMPLab UC Berkeley, now by Databricks.com

What is PySpark



- **PySpark** helps users interface with Resilient Distributed Datasets in apache spark and python.
- Py4J is a popularl library integrated within **PySpark** that lets python interface dynamically with JVM objects (RDD's).

DATA STRUCTURES

There 3 Data Structures in Spark,

1. RDD
2. DataFrames
3. DataSets

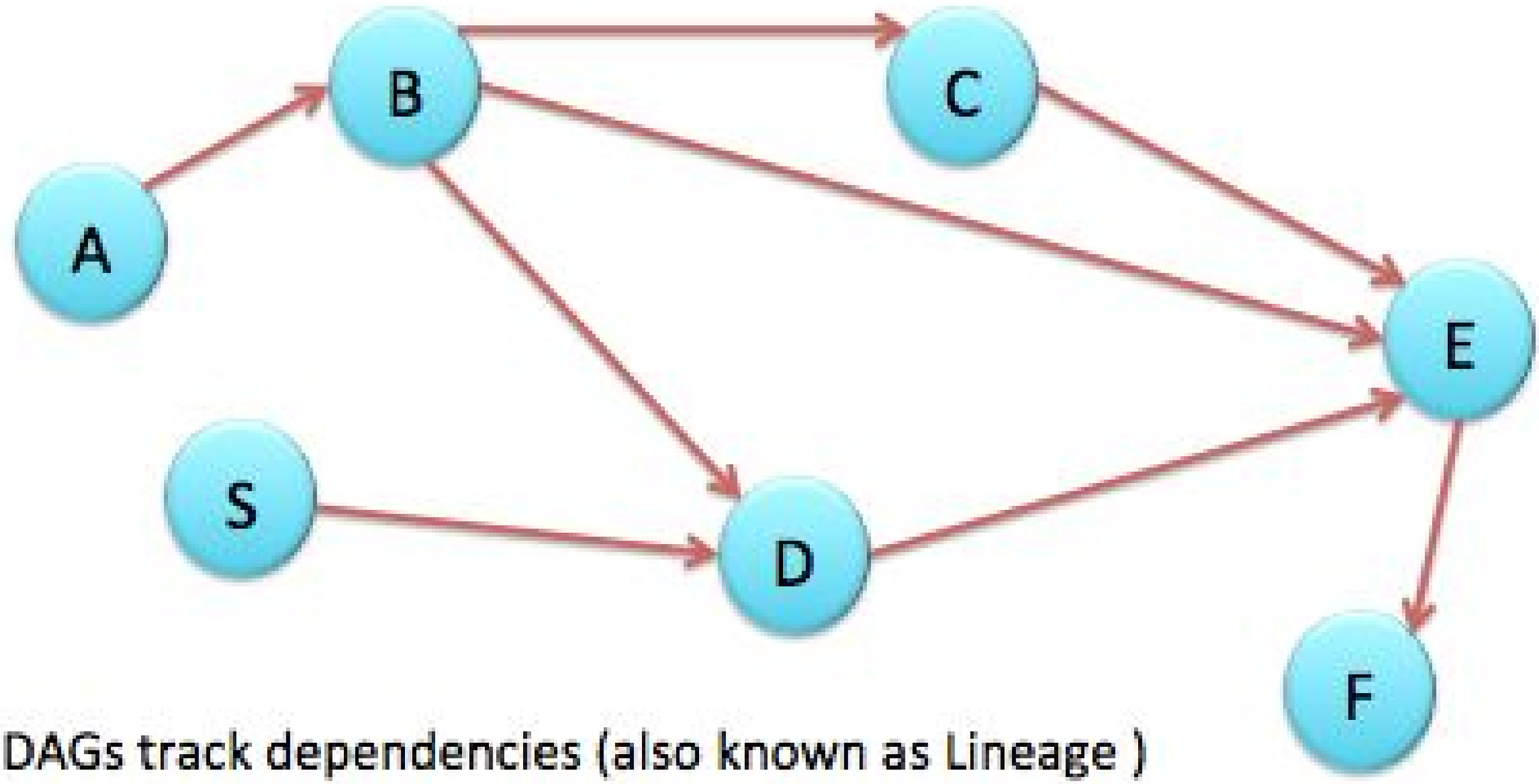
RDD abstraction

- → Resilient Distributed Datasets
- → Building blocks of Spark and original API's
- → Partitioned collection of records
- → Spread across the cluster
- → Read-only
- → caching dataset in memory
  - ◆ different storage levels available
    - The Storage tab on the **Spark** UI shows where partitions exist (**memory** or disk) across the cluster at any given point in time. Note that **cache**() is an alias for persist(StorageLevel.MEMORY_ONLY) which may not be ideal for **datasets** larger than available cluster **memory**.
  - ◆ fallback to disk possible

DataFrames & SparkSQL

- DataFrames (DFs) is one of the other distributed datasets organized in named columns
- Similar to a relational database, Python Pandas Dataframe or R's DataTables
  - Immutable once constructed
  - Track lineage - DAG's
  - Enable distributed computations
- How to construct Dataframes
  - **Read from file(s)**
  - **Transforming an existing DFs(Spark or Pandas)**
  - **Parallelizing a python collection list**
  - **Apply transformations and actions**

Directed Acyclic Graphs (DAG)

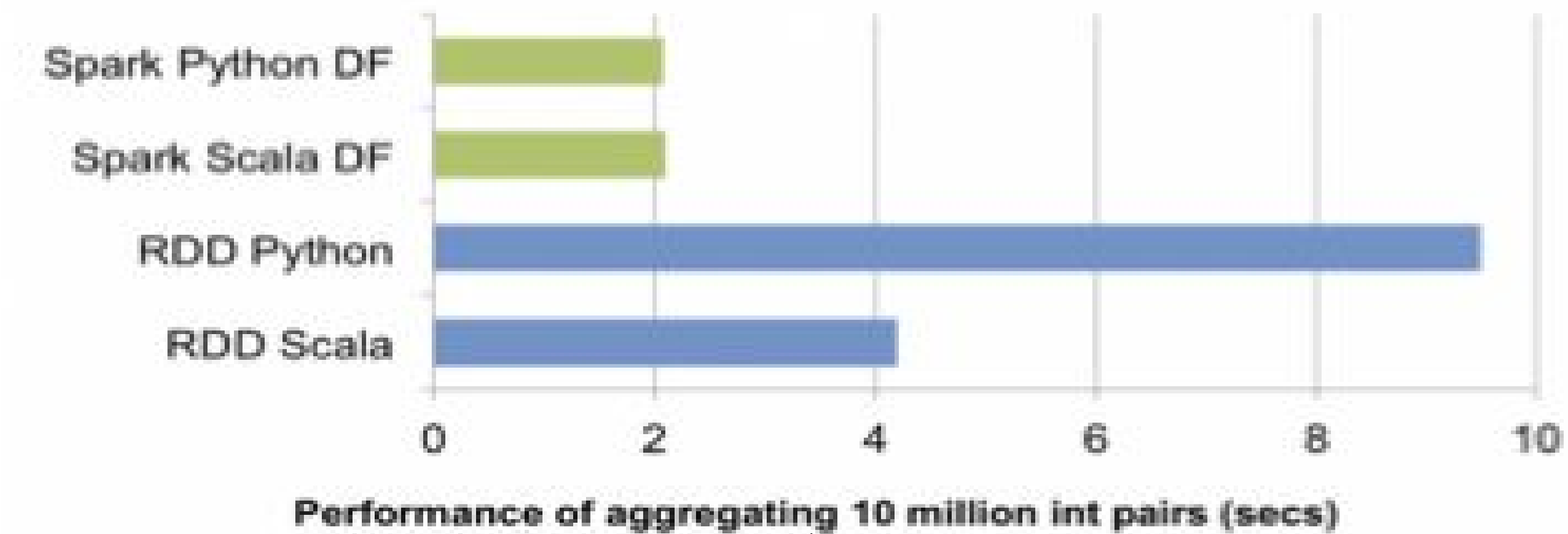


DAGs track dependencies (also known as Lineage)

- nodes are RDDs
- arrows are Transformations

RDDs vs. DataFrames

- RDDs provide a low level interface into Spark
- DataFrames have a schema
- DataFrames are cached and optimized by Spark
- DataFrames are built on top of the RDDs and the core Spark API



Spark operations

- *transformations* to build RDDs through deterministic operations on other RDDs
  - transformations include *map*, *filter*, *join*
  - lazy operation
- *actions* to return value or export data
  - actions include *count*, *collect*, *save*
  - triggers execution

Let's walk through the code samples for these and continue further

Spark in the Real World (I)

- Uber – the online taxi company gathers terabytes of event data from its mobile users every day.
  - By using Kafka, Spark Streaming, and HDFS, to build a continuous ETL pipeline
  - Convert raw unstructured event data into structured data as it is collected
  - Uses it further for more complex analytics and optimization of operations

- Pinterest – Uses a Spark ETL pipeline
  - Leverages Spark Streaming to gain immediate insight into how users all over the world are engaging with Pins—in real time.
  - Can make more relevant recommendations as people navigate the site
  - Recommends related Pins
  - Determine which products to buy, or destinations to visit

Spark in Real World(II)

- **NCR – National Cash Register company**
  - With an enterprise-grade data lake built on Hadoop/Spark, NCR can monitor every ATM it manufactured, everywhere, and build predictive models based on data from 100% of those ATMs.
  - By using Kafka, Spark Streaming, and HDFS, to build a continuous ETL pipeline
  - Convert raw unstructured event data into structured data as it is collected
  - A Predictive Maintenance Model is been built on Spark
  - The Model is productionised in May 2017 and is running successfully since then.

When to use Spark?

- Data Integration and ETL
- Interactive Analytics
- High Performance Batch computation
- Machine Learning and Advanced Analytics
- Real time stream processing and IoT
- **Example applications**
  a. Credit Card Fraud Detection
  b. Network Intrusion Detection
  c. Advertisement Targeting
  d. Predictive Maintenance for ATM’s

Spark: when not to use

- Even though Spark is versatile, that doesn't mean Spark's in-memory capabilities are the best fit for all use cases:
  - For many simple use cases Apache MapReduce and Hive might be a more appropriate choice
  - Spark was not designed as a multi-user environment
  - Spark users are required to know that memory they have is sufficient for a dataset
  - Adding more users adds complications, since the users will have to coordinate memory usage to run code

Questions?

- You can ask me any questions you have now. In future if you wanted to reach me, you can @ ksjpswaroop@gmail.com
- The code and slides are on github : https://github.com/ksjpswaroop/IoTConference-2017


DANKSCHEEN
SPASSIBO
NUHUN
SNACHALHUYA
CHALTU
TASHAKKUR ATU
YAQHANYELAY
YUSPAGARÀTAM
WABEEJA
MAITEKA
HUI
SUKSAMA
EKHMET
SPASIBO
DENKAUJA
NENACHALHYA
TINGKI
BİYAN
SHUKRIA
THANK
YOU
GRACIAS
ARIGATO
SHUKURIA
TAVTAPUCH
MEDAWAGSE
GOZAIMASHITA
EFCHARISTO
AGUYJE
FAKAAUE
MERASTAWHY
GAEJTHO
SANCO
KOMAPSUMNIDA
MAAKE
LAH
GRAZIE
MEHRBANI
PALDIES
JUSPAXAR
BAIKA
BOLZÏN
MERCI
MINMONCHAR
MAKETAI
SIKOMO
EKOJU
HATUR
GUI
UNALCHÉESH
ATTO
AHHA
MIRSI