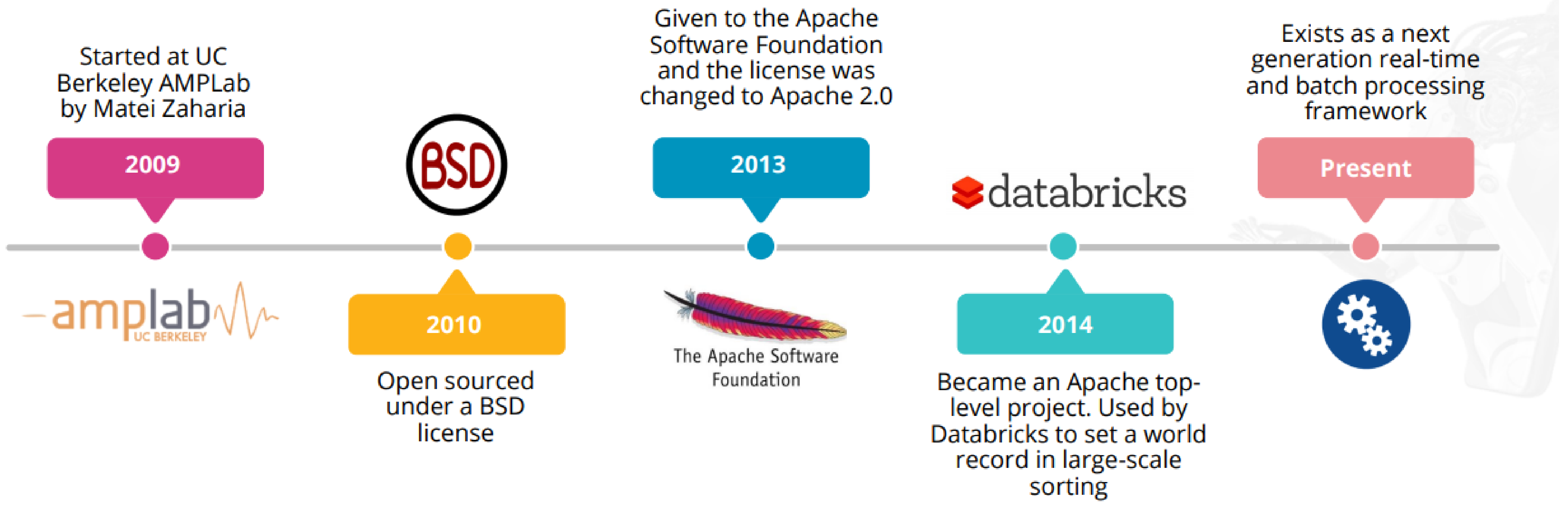
Spark

What is spark?

- A fast and general engine for large-scale data processing
- Apache **Spark** is an open-source, distributed processing system **used for** big data workloads. It utilizes in-memory caching and optimized query execution for fast queries against data of any size. Simply put, **Spark** is a fast and general engine for large-scale data processing.

History Of Spark



2009

Started at UC Berkeley AMPLab by Matei Zaharia

2010

Open sourced under a BSD license

2013

Given to the Apache Software Foundation and the license was changed to Apache 2.0

2014

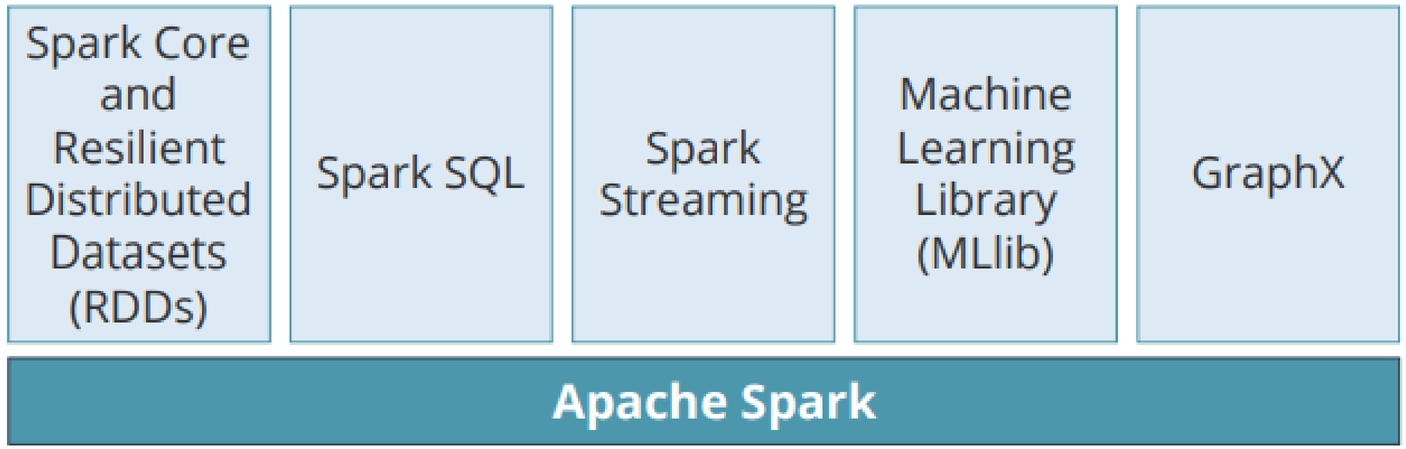
Became an Apache top-level project. Used by Databricks to set a world record in large-scale sorting

Present

Exists as a next generation real-time and batch processing framework

Introduction to Apache Spark

- Is suitable for real-time processing, trivial operations, and processing larger data on a network
- Is an open source cluster computing framework
- Provides up to 100 times faster performance for a few applications with in-memory primitives, compared to the two-stage disk-based MapReduce paradigm of Hadoop
- Is suitable for machine learning algorithms, as it allows programs to load and query data repeatedly


Spark Core and Resilient Distributed Datasets (RDDs)
Spark SQL
Spark Streaming
Machine Learning Library (MLlib)
GraphX
Apache Spark

Components of a Spark Project

The components of a Spark project are explained below:

Spark Core and RDDs

As the foundation, it provides basic I/O, distributed task dispatching, and scheduling. RDDs can be created by applying coarse-grained transformations or referencing external datasets.

Spark SQL

As a component lying on the top of Spark Core, it introduces SchemaRDD, which can be manipulated. It supports SQL with ODBC/JDBC server and command-line interfaces.

Components of a Spark Project

The components of a Spark project are explained below:

Spark Streaming	It leverages the fast scheduling capability of Spark Core, ingests data in small batches, and performs RDD transformations on them.
MLlib	As a distributed machine learning framework on top of Spark, it is nine times faster than the Hadoop disk-based version of Apache Mahout.
GraphX	Being a distributed graph processing framework on top of Spark, it gives an API and provides an optimized runtime for the Pregel abstraction.

Application of In-Memory Processing

In column-centric databases, informations that are similar, can be stored together. The working of in-memory processing can be explained as below:

- The entire information is loaded into memory, eliminating the need for indexes, aggregates, optimized databases, star schemas, and cubes.
- Compression algorithms are used by most of the in-memory tools, thereby reducing the in-memory size.
- Querying the data loaded into the memory is different from caching.
- With in-memory tools, the analysis of data can be flexible in size and can be accessed within seconds by concurrent users with an excellent analytics potential.
- It is possible to access visually rich dashboards and existing data sources.

Language Flexibility in Spark

Spark is popular for its performance benefits over MapReduce.
Another important benefit is language flexibility, as explained below:

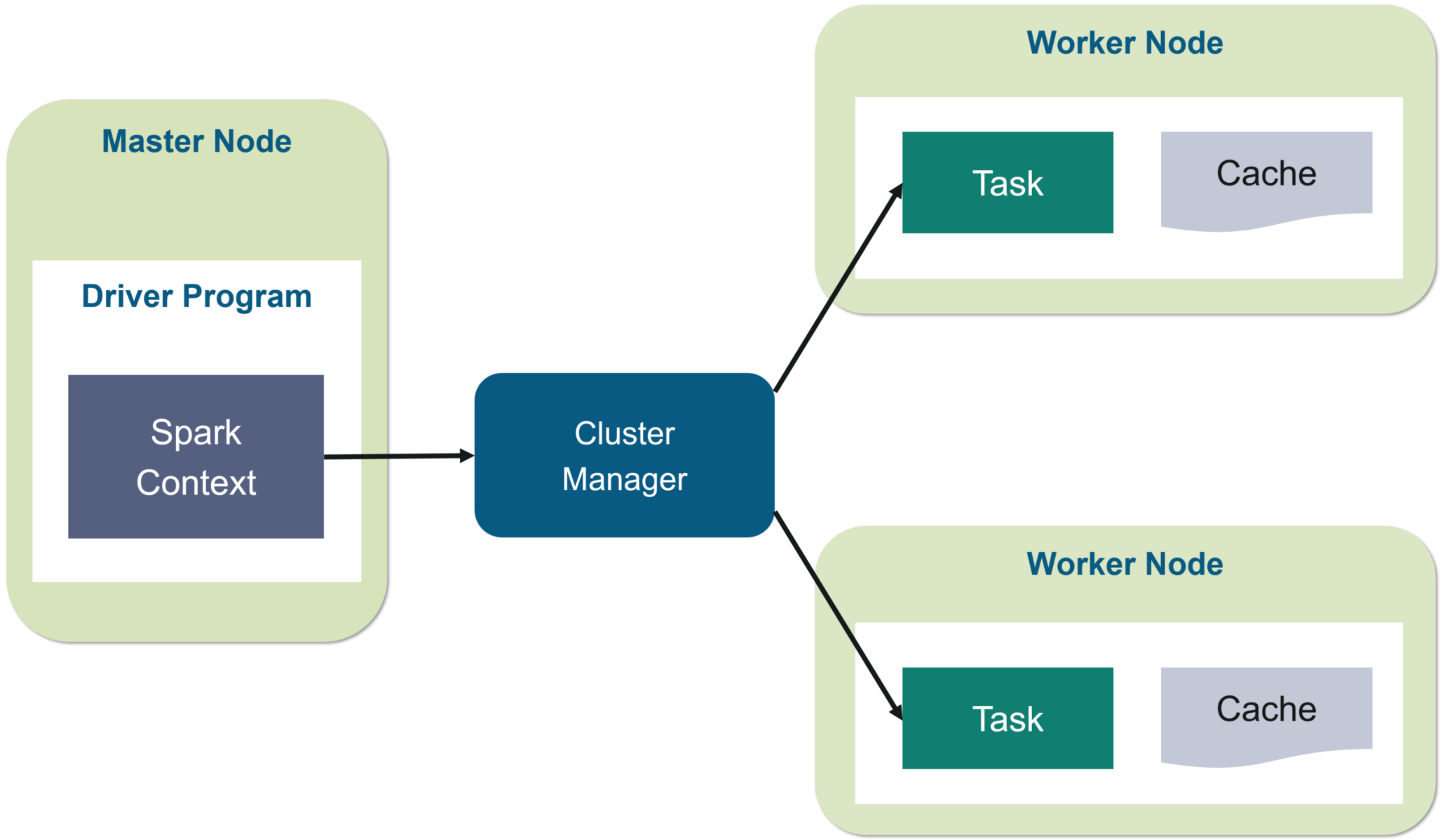
Support for various development languages

Spark supports popular development languages like Java, Scala, and Python and will support R.

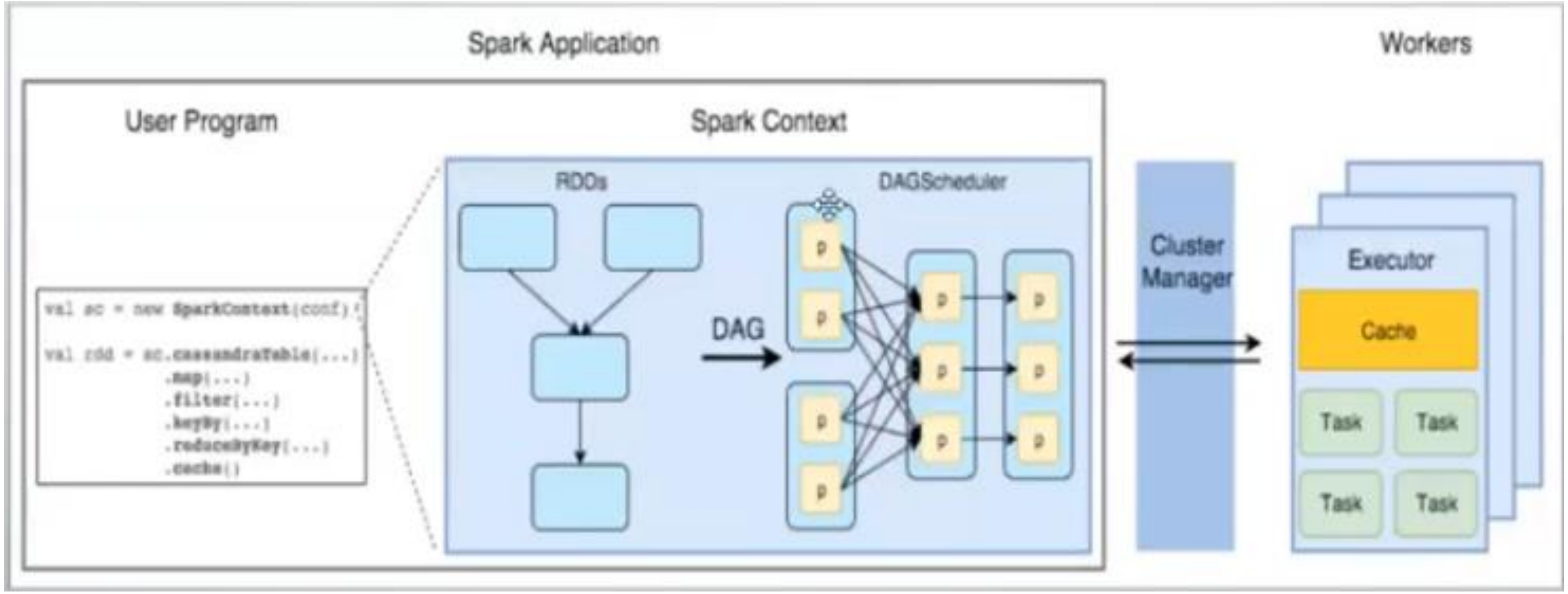
Capability to define functions in-line

With the temporary exception of Java, a common element in these languages is that, they provide methods to express operations using lambda functions and closures.

Spark Architecture



Spark Execution workflow



```
val sc = new SparkContext(conf)

val rdd = sc.cassandraTable(...)
            .map(...)
            .filter(...)
            .keyBy(...)
            .reduceByKey(...)
            .cache()
```