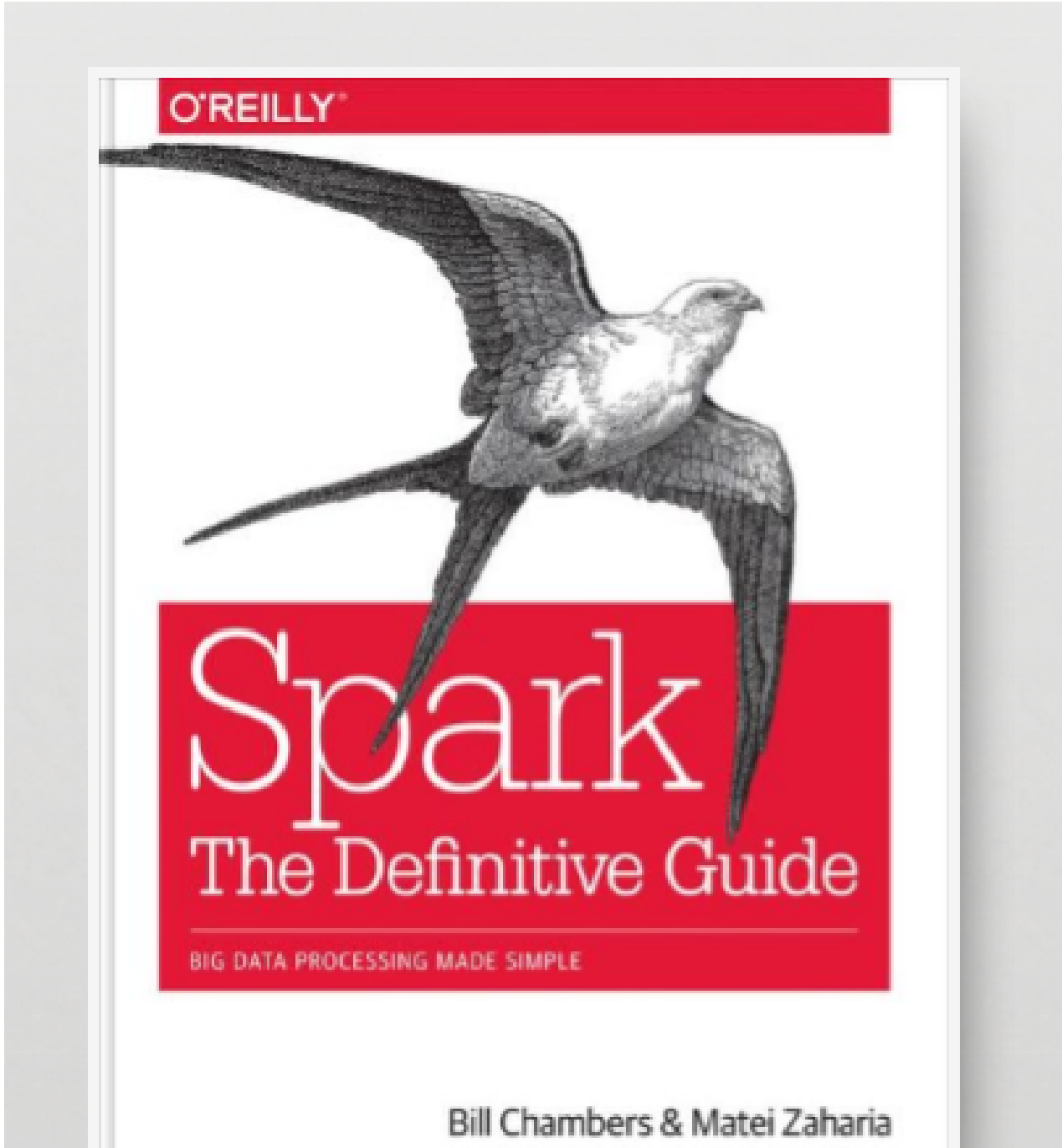
# Spark the Definitive Guide 2nd Edition

Chapter 01

Spark Philosophy

# A Gentle Overview

# Text Book


O'REILLY
Spark
The Definitive Guide
BIG DATA PROCESSING MADE SIMPLE
Bill Chambers & Matei Zaharia

# Preface

- What part of Spark will we cover?
  - High Level Structured APIs
  - DataFrames
  - DataSets
  - SparkQL
  - Structured Streaming
- We will focus in Application development more than operations
- RDDs and DStreams are deprecated and won't be covered

# Overview

# Spark Philosophy

- Spark is a **Unified Computing Engine**
- Spark is a set of libraries for parallel data processing on computer clusters

# Native Language Support

- Scala
- Java
- Python
- R
- SQL

# Architecture

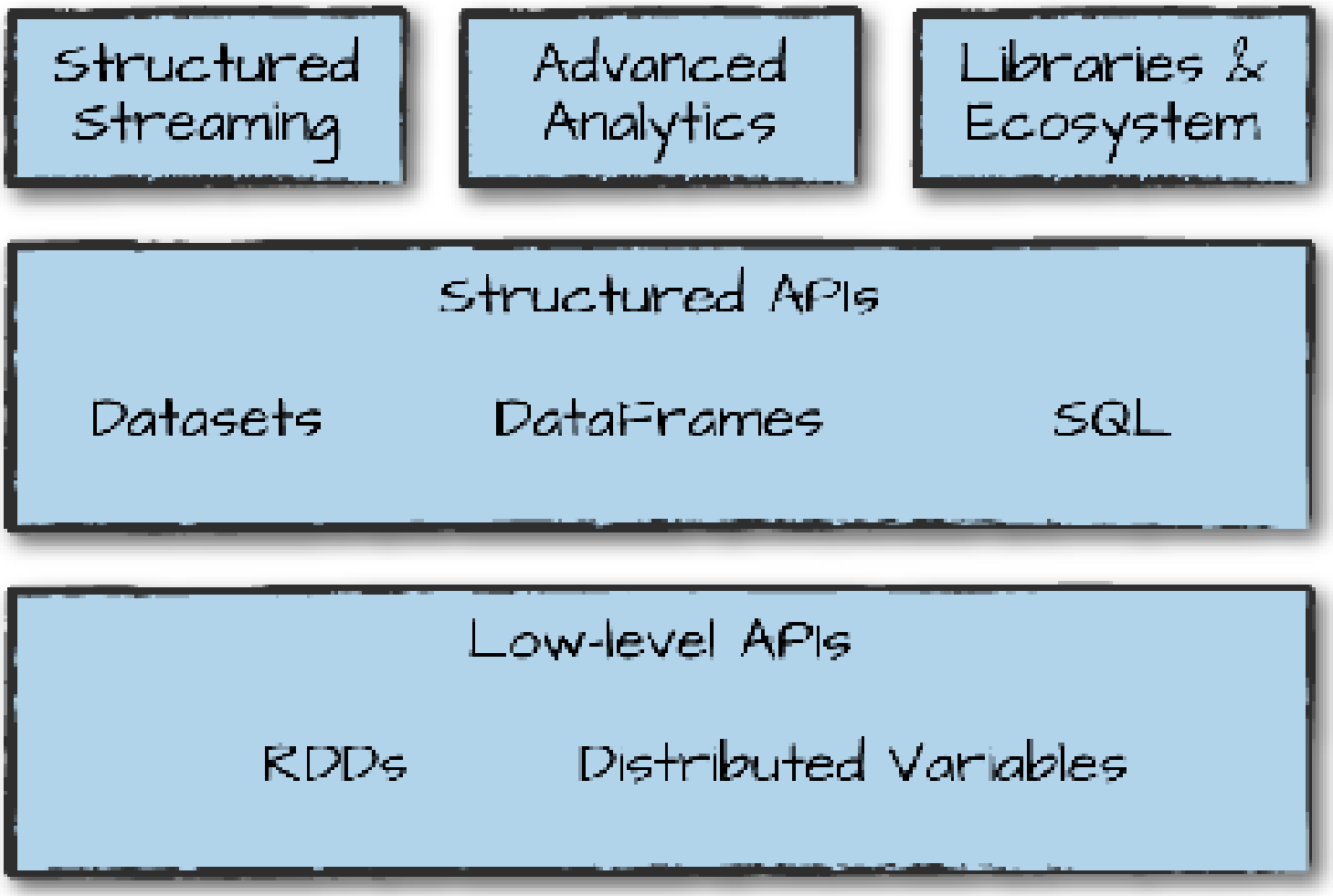



*Figure 1-1. Spark's toolkit*

# Breakdown

- A Unified Computing Engine and set of libraries for big data
- Lets break this design down and analyze the parts

# Unified

- Spark offers a **Unified Platform** for Big Data
  - Spark supports data loading (called injesting of data)
  - Spark supports SQL queries
  - Native support of Machine Learning (in memory and iterative data processing)
  - Native support for Streaming Computation (realtime data such as kafka)
- All of these things are included in the standard library of Spark
  - You get this all in one place
  - In the past you needed different software for each process

# Unified APIs

- Unified APIs allow you to mix and match above APIs
  - For instance use SparkQL to load data via an SQL query into a Machine Learning Algorithm
  - Use native Python or RIang data analytics libraries and tools on data
  - The Web has standardized frameworks, Nodejs, Django, or ASP.Net, why not big data?

# Unified DataTypes

- Unified Engine for parallel data processing
  - Structured APIs
  - DataFrames
  - DataSets
  - SQL

# Computing Engine

- Spark decided it wanted to only be a compute engine not a storage engine
- Spark realized in 2012, it no longer had to overcome hardware limitations
  - RDBMS
  - No T-SQL
  - No MS-SQL
  - No Hadoop/HDFS/Pig/Sqoop/Hive
- Spark does not have a native storage engine
  - This is a good thing

# Focus

- This creates a simplified focus only computing not storage, latency, CAP, consistency
- It has libraries that will load from your storage types
  - CSV/TSV
  - SQL
  - HDFS/Hadoop
  - Parquet
  - Orc
  - NoSQL, Cassandra and MongoDB
  - AWS S3 and OpenStack Swift, object storage

# Difference with Spark and Hadoop

- Hadoop was designed in a time where there was slow disk and non-optimal file systems
- Hadoop has its own filesystem–HDFS
- Hadoop used only MapReduce in version 1, opened up to other frameworks in version 2
- Spark made the decision not to have a storage engine

# Libraries

- What is the difference between a library and a framework?
- Spark has the ability to add more native libraries over time into the Core Spark Library
  - Recently Microsoft contributed a native Spark Library for C#
- You can see all additional packages available at https://spark-packages.org

# Context

- Why do we need this at all?
- Let look at the historical context of Big Data
  - Michael Stonebreaker father of RDMBS
  - Ingress
  - Size, Speed, and Latency of disk
  - Ubiquitous Internet and network latency
  - Filesystems not engineered for data integrity

# Hadoop and HDFS

- Created in ~2005 by Doug Cutting while at Yahoo!
  - 2006 became an Apache Foundation project
- Part of an attempt to reverse engineer the Google Search Engine
- Originally used MySql and an opensource web-crawler named Nutch
  - Found that MySql didn't scale to web-scraping at that time
  - Deficiencies in indexing that much data, and storing and retrieving data
- Two research papers from Google published in 2003 and 2004 lead to the ideas of Hadoop
  - The Google File System
  - MapReduce: Simplified Data Processing on Large Clusters

# Hardware Changes

- Processor speeds began to level off in the mid-2000s
  - Number of processor cores per CPU increased
  - Amount and speed of memory increased
  - Disk Storage capacity of 1 TB drops by 2x every 14 months
- But with ubiquitous internet, everything generates more data
  - Jet Engines
  - Self Driving Cars
  - Banking Apps
  - Facial Recognition
  - Twitter

# Spark History

- Spark was a research paper released at UC Berkely in 2009
  - Original Spark Paper
  - Matei Zaharia and Mosharaf Chowdhury
  - First software release in 2012, then code assigned to the Apache Foundation
  - First Apache Spark release in 2014
  - ~8 years after Hadoop
  - They studied the usage patterns of researchers using Hadoop at UC Berkeley and determined the weaknesses and strengths and sought to correct this based on the changes in hardware that had taken place since Hadoop was developed

# Results of Spark Paper

- Two things became clear
  - Cluster based parallel computing was a good thing in Hadoop as many new applications could be solved in parallel
  - MapReduce Framework made developing things difficult (MapReduce vs SQL in a sense)
- No need to discard all of the knowledge and work that had been done in and with SQL over the many years of its history
- Spark is used by many large companies such as Netflix, NASA, and CERN as well as many small companies.

# Conclusion

- Spark is three things:
  - A Unified Platform
  - A Computing Engine
  - A set of expandable core libraries
- A Unified Computing Engine and set of libraries for big data