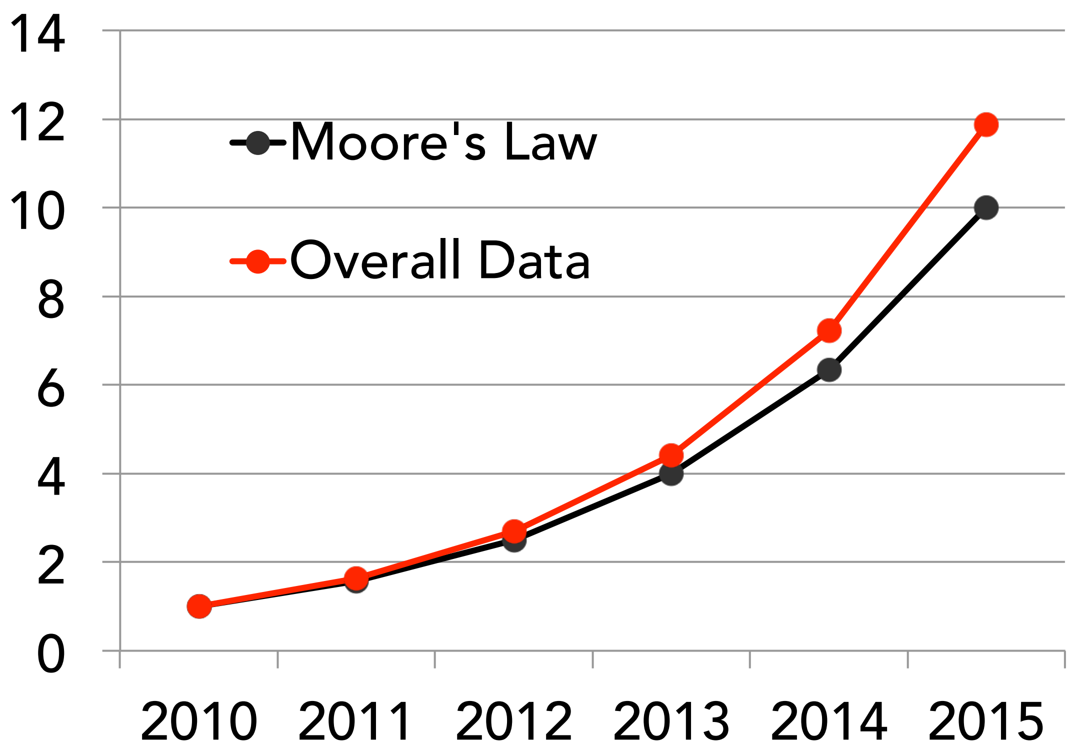
Berkeley Data Analytics Stack (BDAS)

Ion Stoica
UC Berkeley / Databricks / Conviva

Data is Everywhere

Easier and cheaper than ever to collect

Data grows faster than Moore's law



(IDC report*)

The New Gold Rush

Everyone wants to extract value from data

- » Big companies & startups alike

Huge potential

- » Already demonstrated by Google, Facebook, …

But, untapped by most companies

- » “We have lots of data but no one is looking at it!”

Extracting Value from Data Hard

Data is massive, unstructured, and dirty

Question are complex

Processing, analysis tools still in their “infancy”

Need tools that are

- » Faster
- » More sophisticated
- » Easier to use

Turning Data into Value

Insights, diagnosis, e.g.,

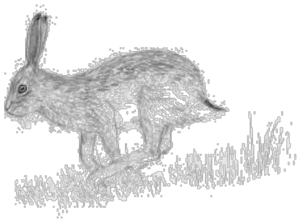
- » Why is user engagement dropping?
- » Why is the system slow?
- » Detect spam, DDoS attacks

Decisions, e.g.,

- » Decide what feature to add to a product
- » Personalized medical treatment
- » Decide when to change an aircraft engine part
- » Decide what ads to show

Data only as useful as the decisions it enables

What do We Need?

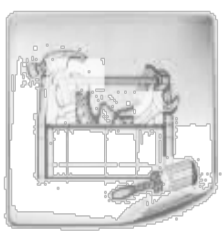


Interactive queries: enable faster decisions

» E.g., identify why a site is slow and fix it

Queries on streaming data: enable decisions on real-time data

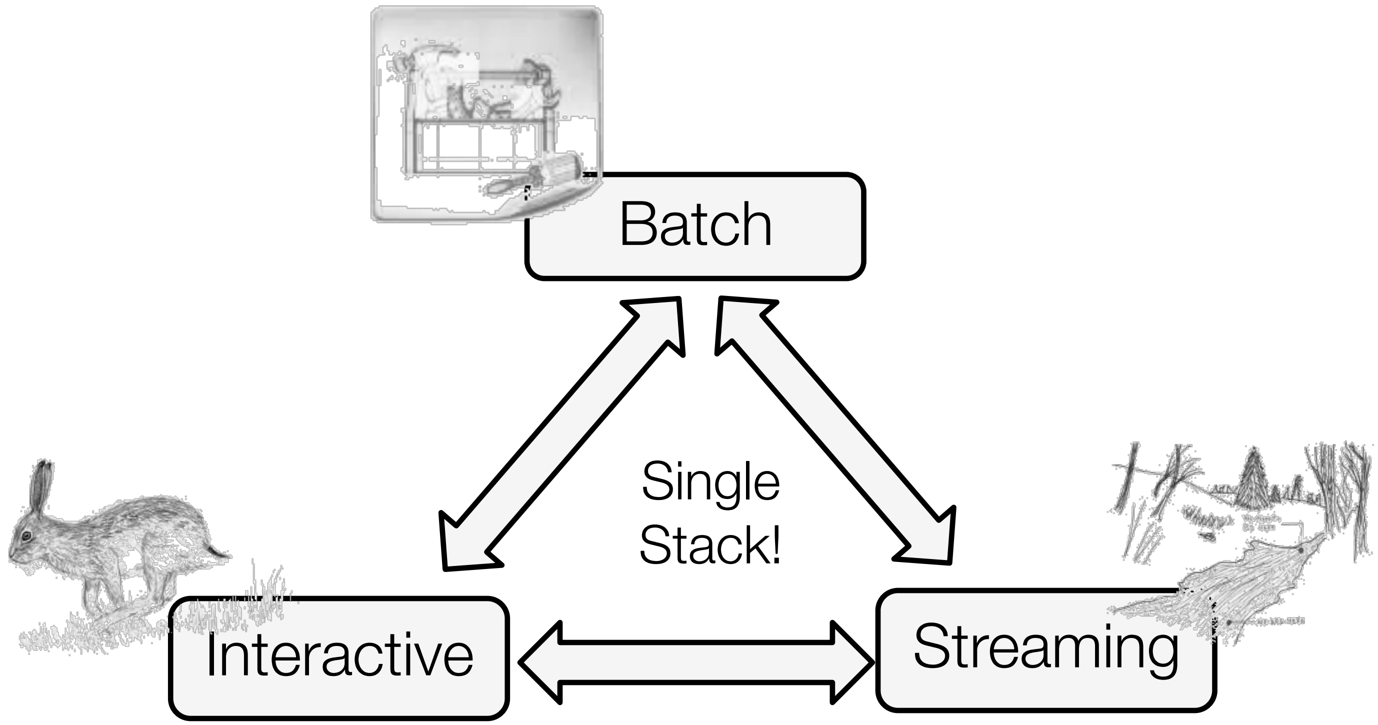
» E.g., fraud detection, detect DDoS attacks



Sophisticated data processing: enable "better" decisions

» E.g., anomaly detection, trend analysis

Our Goal



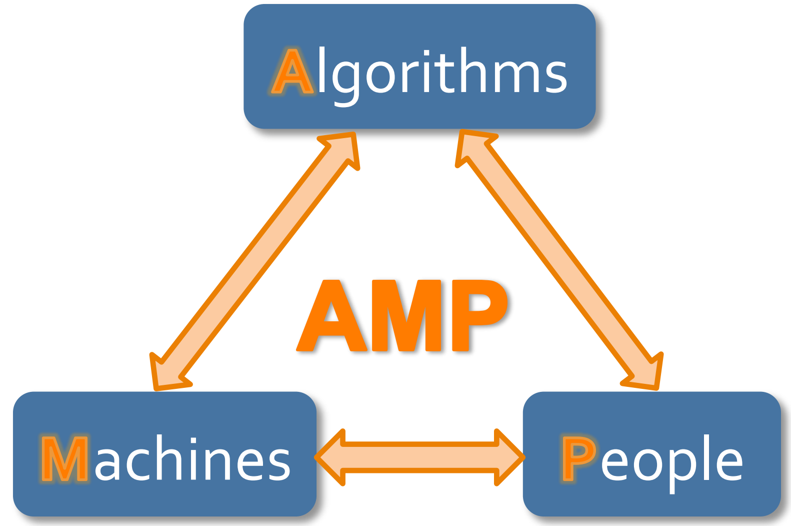
Support *batch*, *streaming*, and *interactive* computations…

… in a unified framework

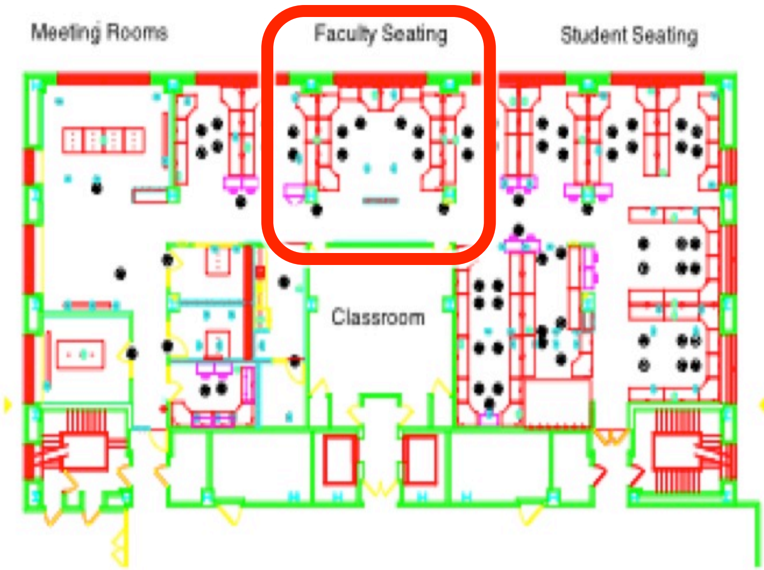
Easy to develop *sophisticated* algorithms (e.g., graph, ML algos)

The Berkeley AMPLab

January 2011 – 2017
- » 8 faculty
- » > 40 students and postdocs
- » 3 software engineer team



Organized for collaboration



3 day retreats
(twice a year)

AMPCamp3
(August, 2013)

220 campers
(100+ companies)

The Berkeley AMPLab

Governmental and industrial funding:

Goal: Next generation of open source data analytics stack for industry & academia:
Berkeley Data Analytics Stack (BDAS)

BDAS Stack (Feb, 2013)

Enable multiple frameworks to share same cluster resources (e.g., Hadoop, Storm, Spark) (2009)

Scale to thousands of servers (e.g., Twitter)

Third party schedulers, e.g., Chronos, Aurora

Mesos

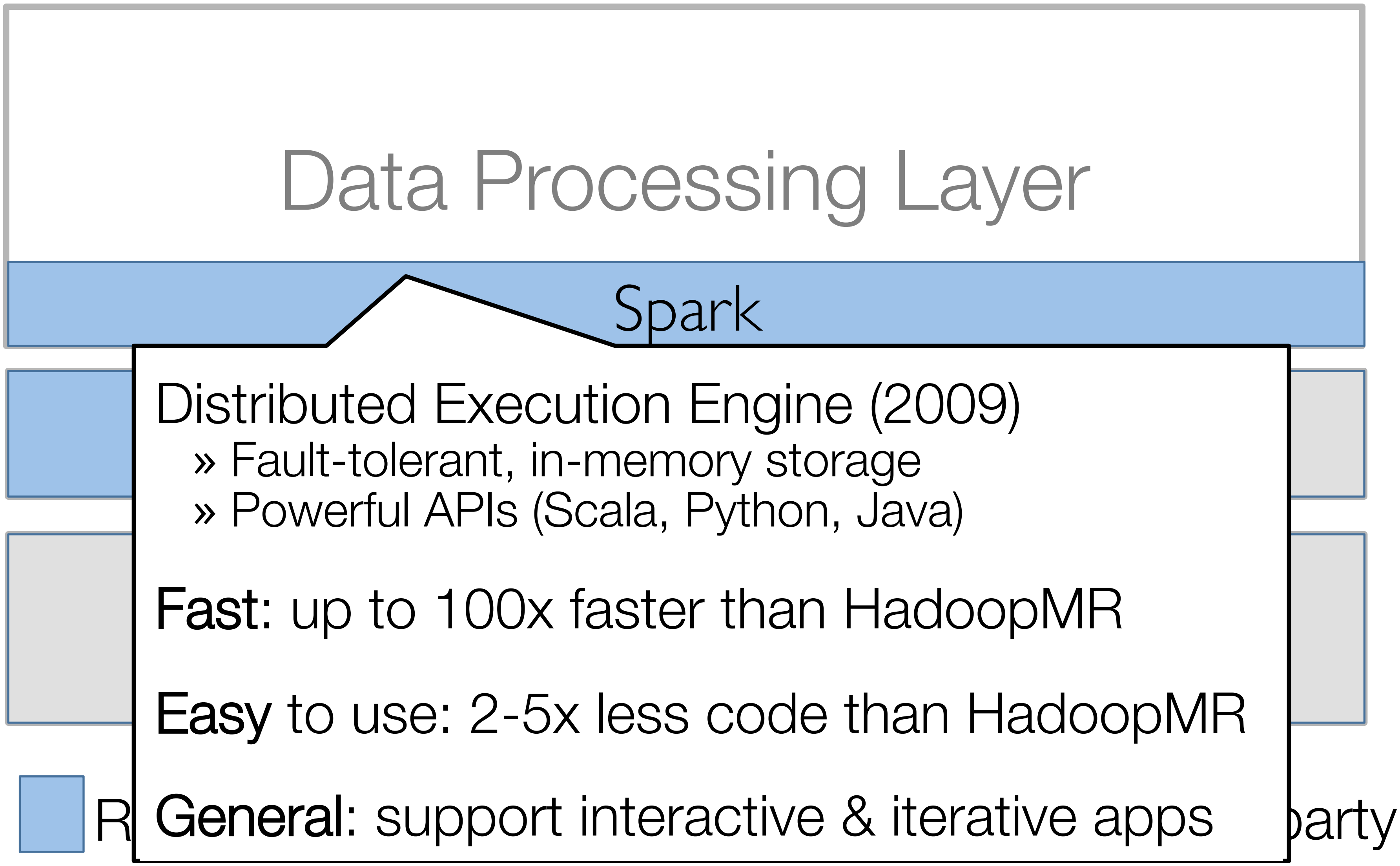
HDFS, S3, …

Releases

Research Projects

3rd party

BDAS Stack (Feb, 2013)



Distributed Execution Engine (2009)

» Fault-tolerant, in-memory storage

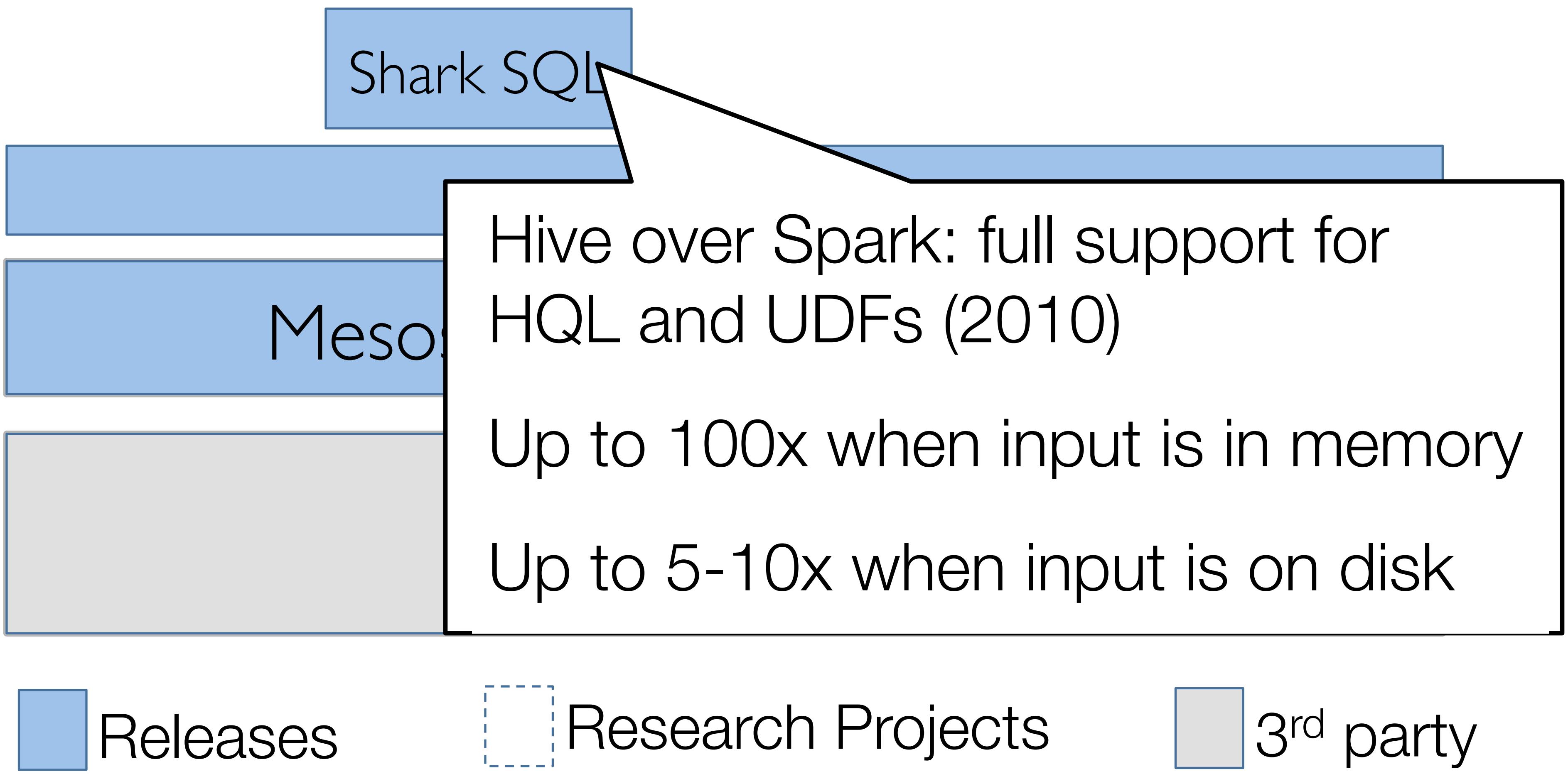
» Powerful APIs (Scala, Python, Java)

Fast: up to 100x faster than HadoopMR

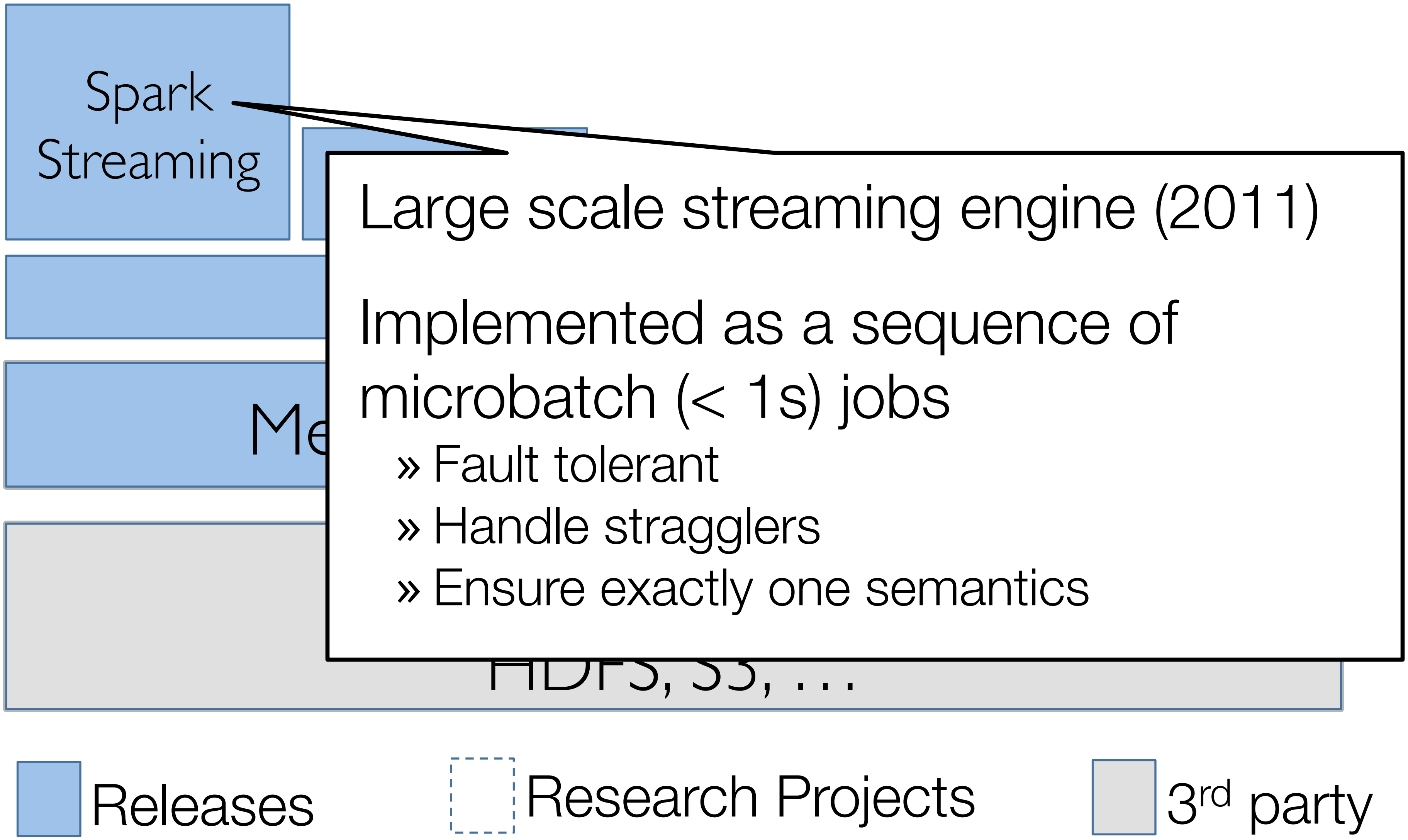
Easy to use: 2-5x less code than HadoopMR

General: support interactive & iterative apps

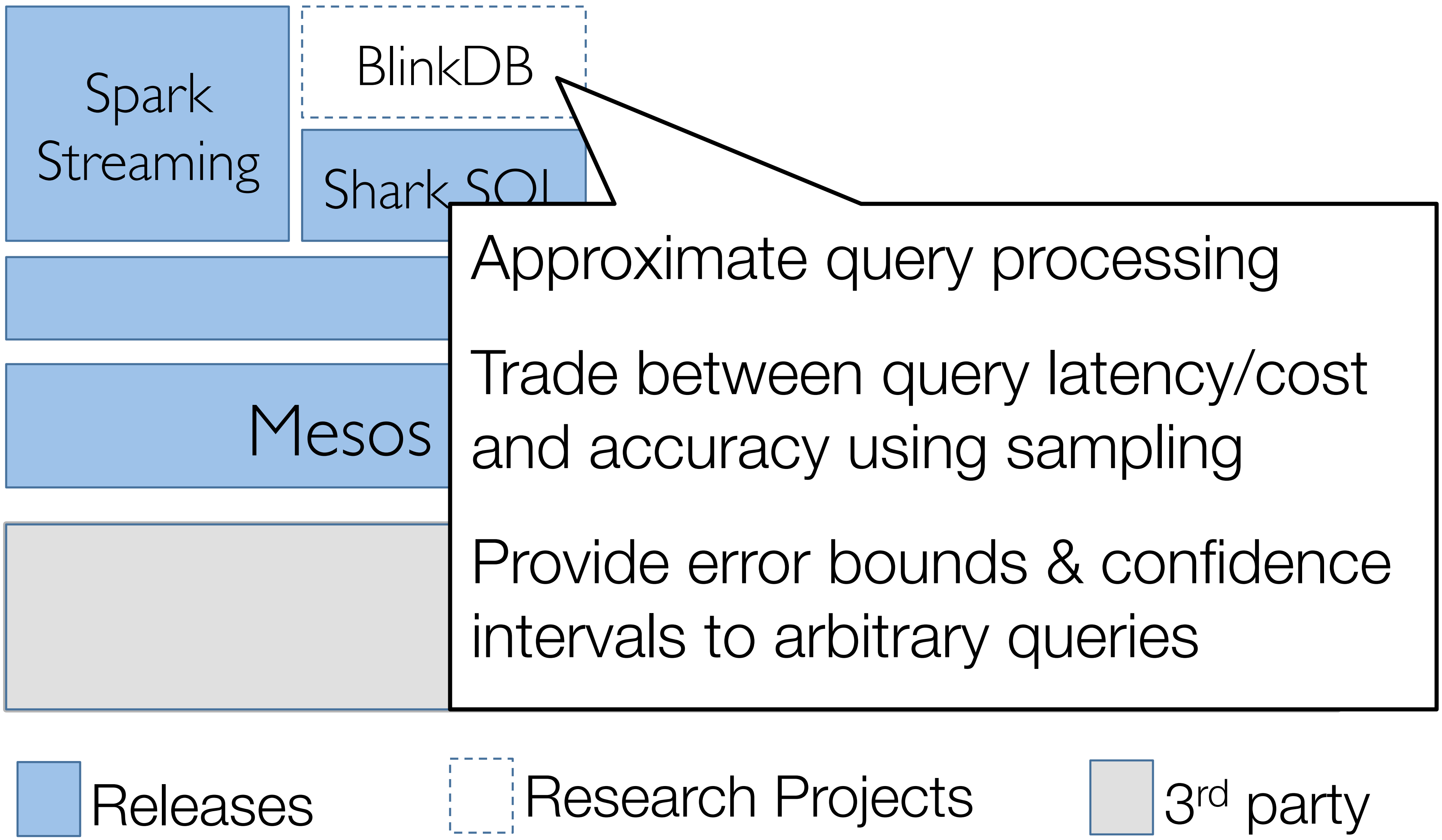
BDAS Stack (Feb, 2013)



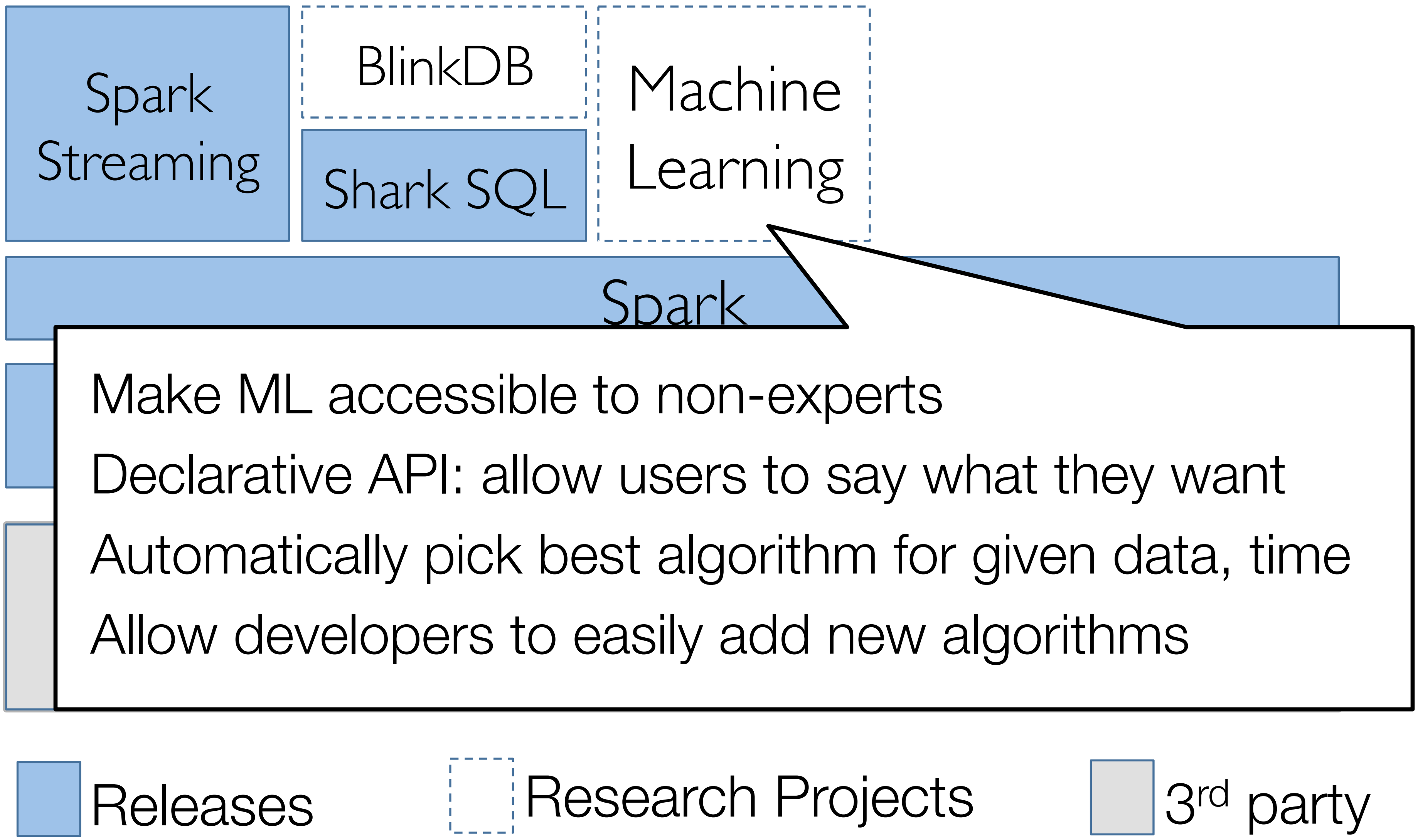
BDAS Stack (Feb, 2013)



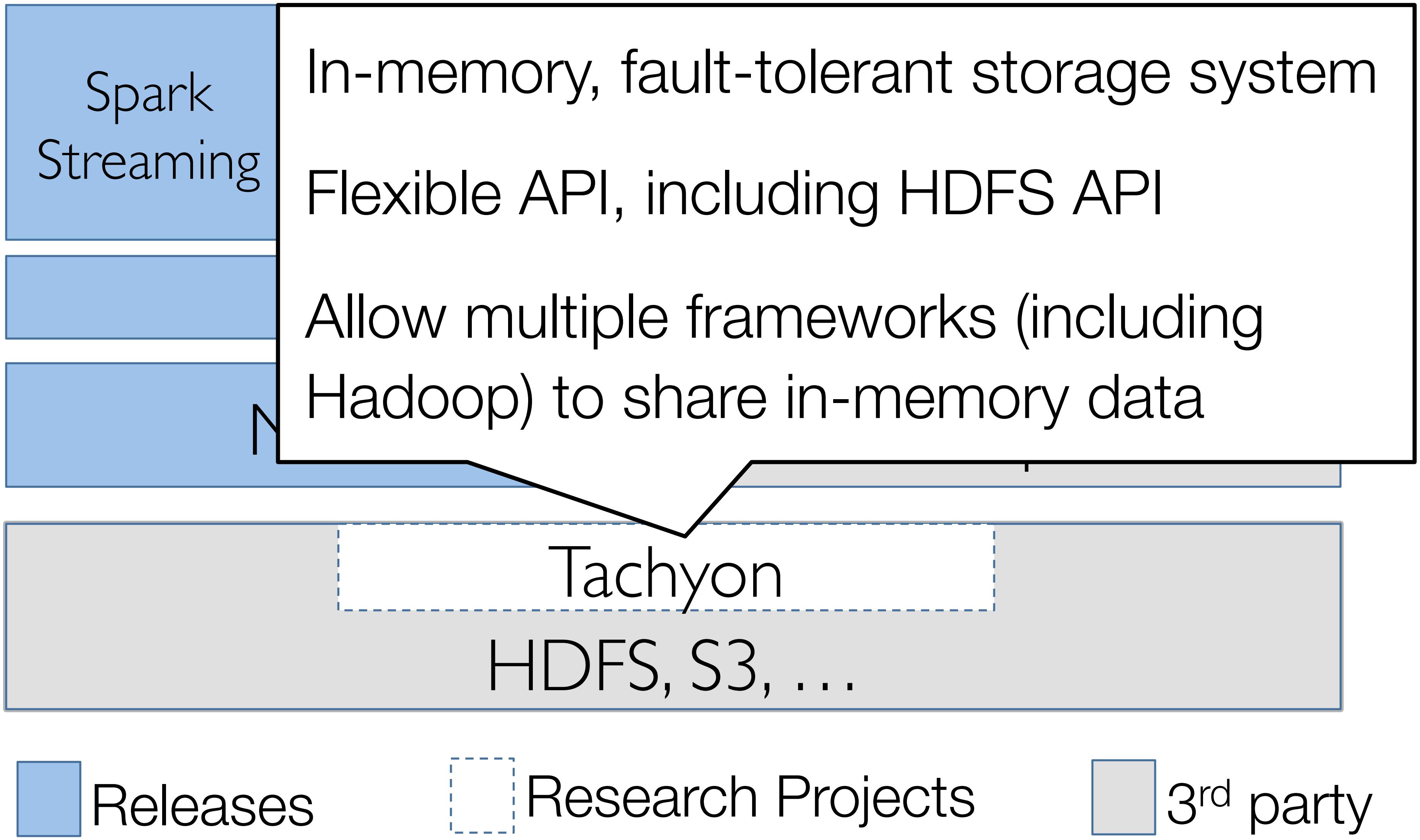
BDAS Stack (Feb, 2013)



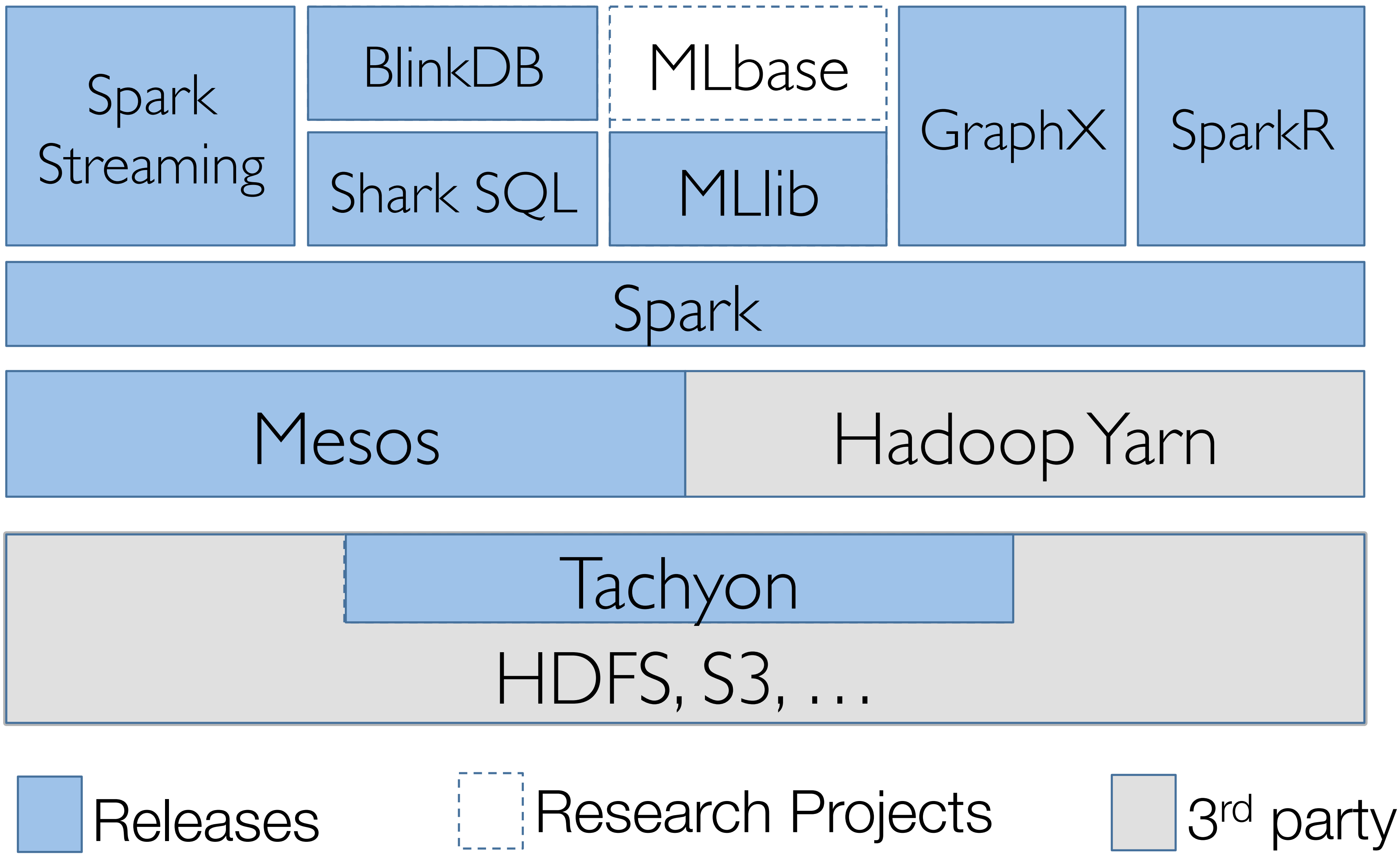
BDAS Stack (Feb, 2013)



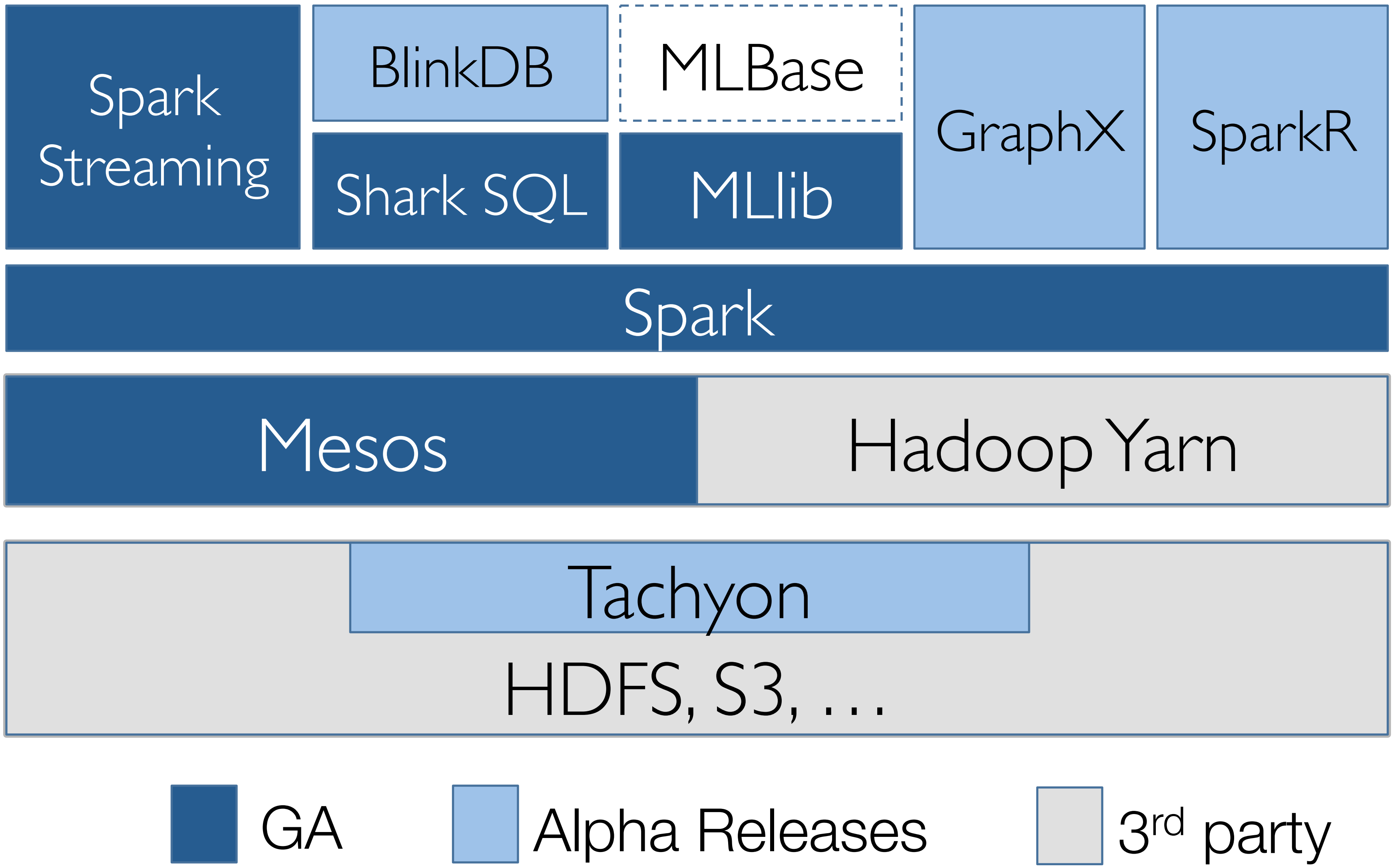
BDAS Stack (Feb, 2013)



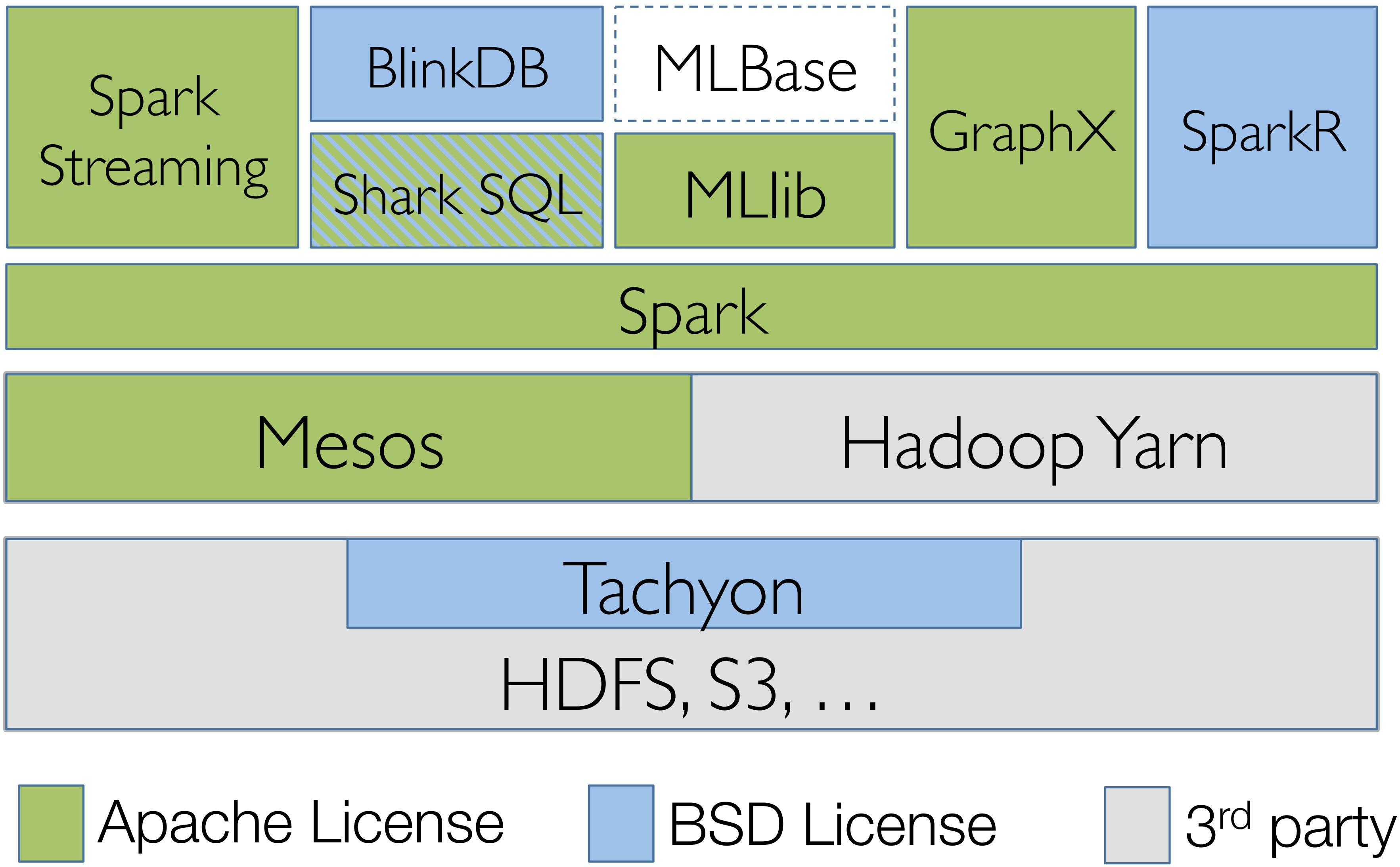
BDAS Stack (Feb, 2014)



BDAS Stack (Feb, 2014)



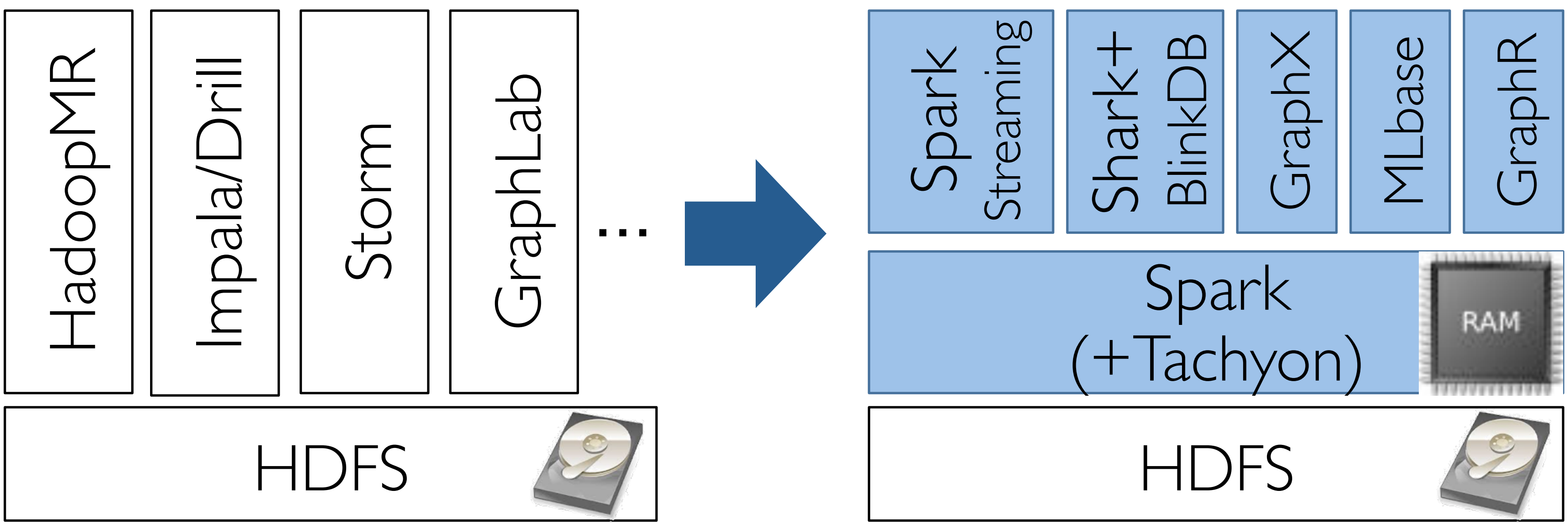
BDAS Stack (Feb, 2014)



Unification: One Size Fits Many!

Using Spark & Tachyon BDAS unifies

» Batch
» Streaming
» Interactive
» Iterative (e.g., graph and ML algorithms)



Unification Examples

Real-time and historical data analysis

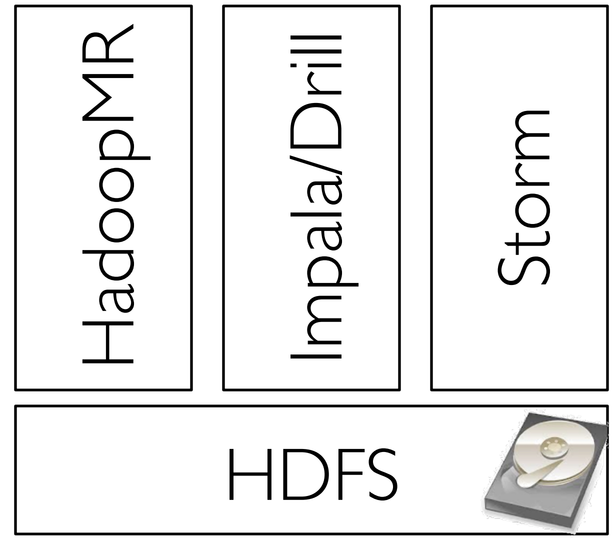
Streaming and machine-learning

Graph processing and ETLs

Unify Real-time and Historical Analytics

Today: separate stacks

- » Historical analysis (Hadoop, Hive)
- » Streaming (Storm)
- » Interactive queries (Impala)



Disadvantages:

- » Hard to maintain and operate
- » Hard to integrate: cannot support interactive ad-hoc queries on streaming data

Unify Real-time and Historical Analytics

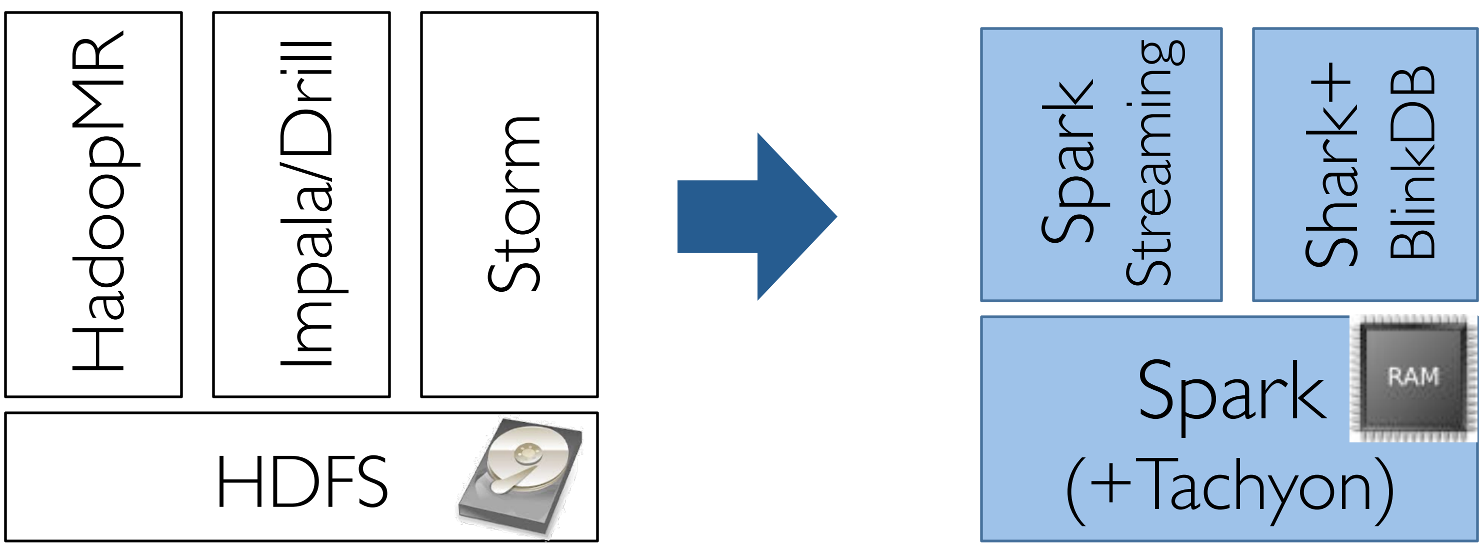
Spark (+ Streaming, Shark): single stack

» Easier to build and maintain

» Cheaper to operate

» Interactive queries on streaming data: faster decisions

» Simplify development



Unify Real-time and Historical Analytics

Batch and streaming codes virtually the same

» Easy to develop and maintain consistency

```
// count words from a file (batch)
val file = sc.textFile("hdfs://.../pagecounts-*.gz")
val words = file.flatMap(line => line.split(" "))
val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)
wordCounts.print()
```

```
// count words from a network stream, every 10s (streaming)
val ssc = new StreamingContext(args(0), "NetCount", Seconds(10), ..)
val lines = ssc.socketTextStream("localhost", 3456)
val words = lines.flatMap(_.split(" "))
val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)
wordCounts.print()
ssc.start()
```

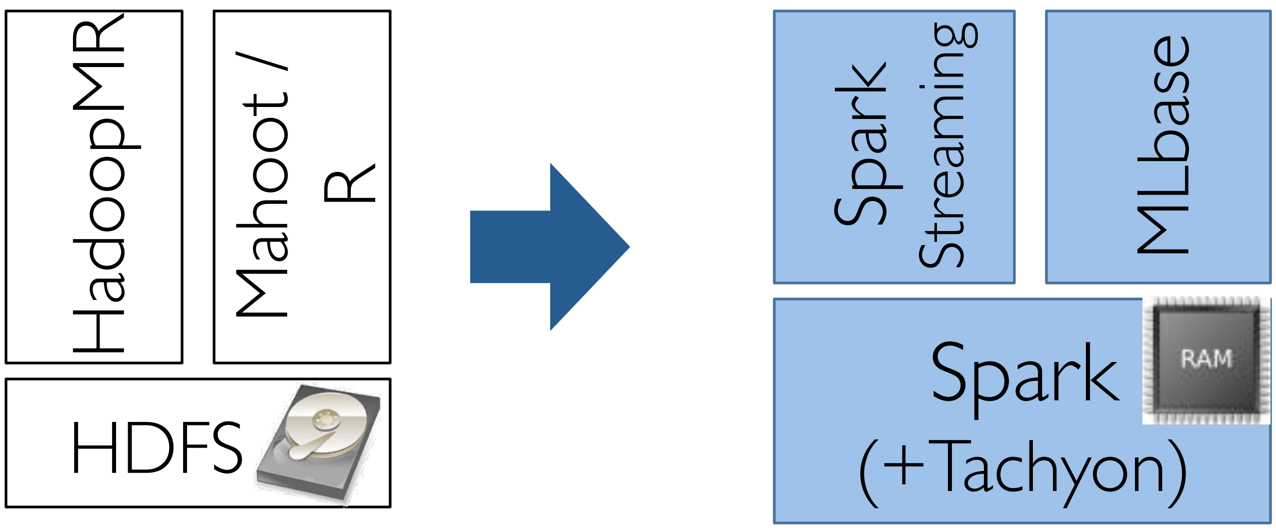
Unify Streaming and ML

Today: ML done mostly off-line

Spark (+ Streaming, MLbase): Real-time diagnosis & decisions

» Fraud detection

» Early notification of service degradation and failures



Unify Graph Processing and ETL

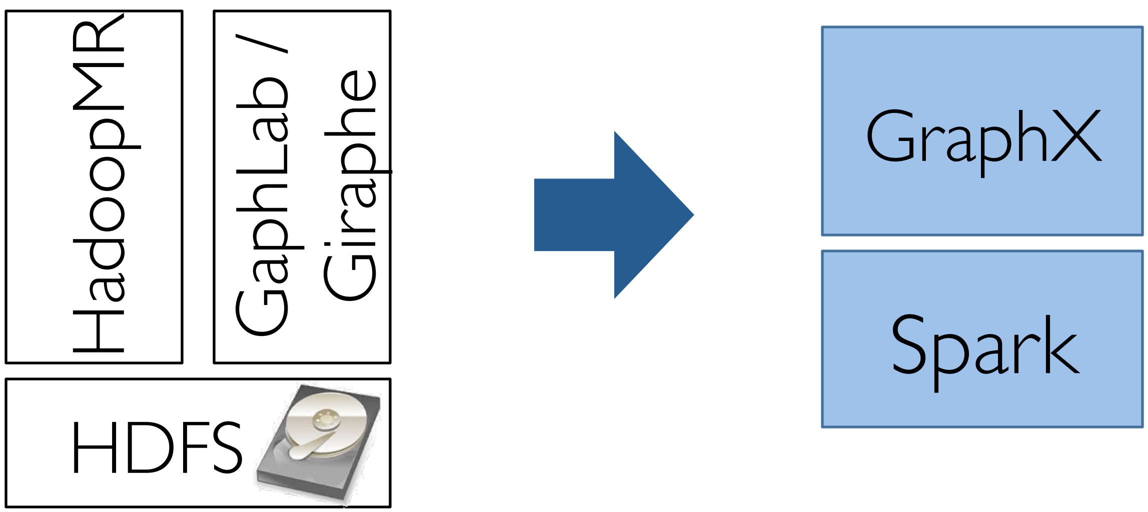
Today: Graph-parallel systems (Pregel, GraphLab)

» Fast and scalable, but…

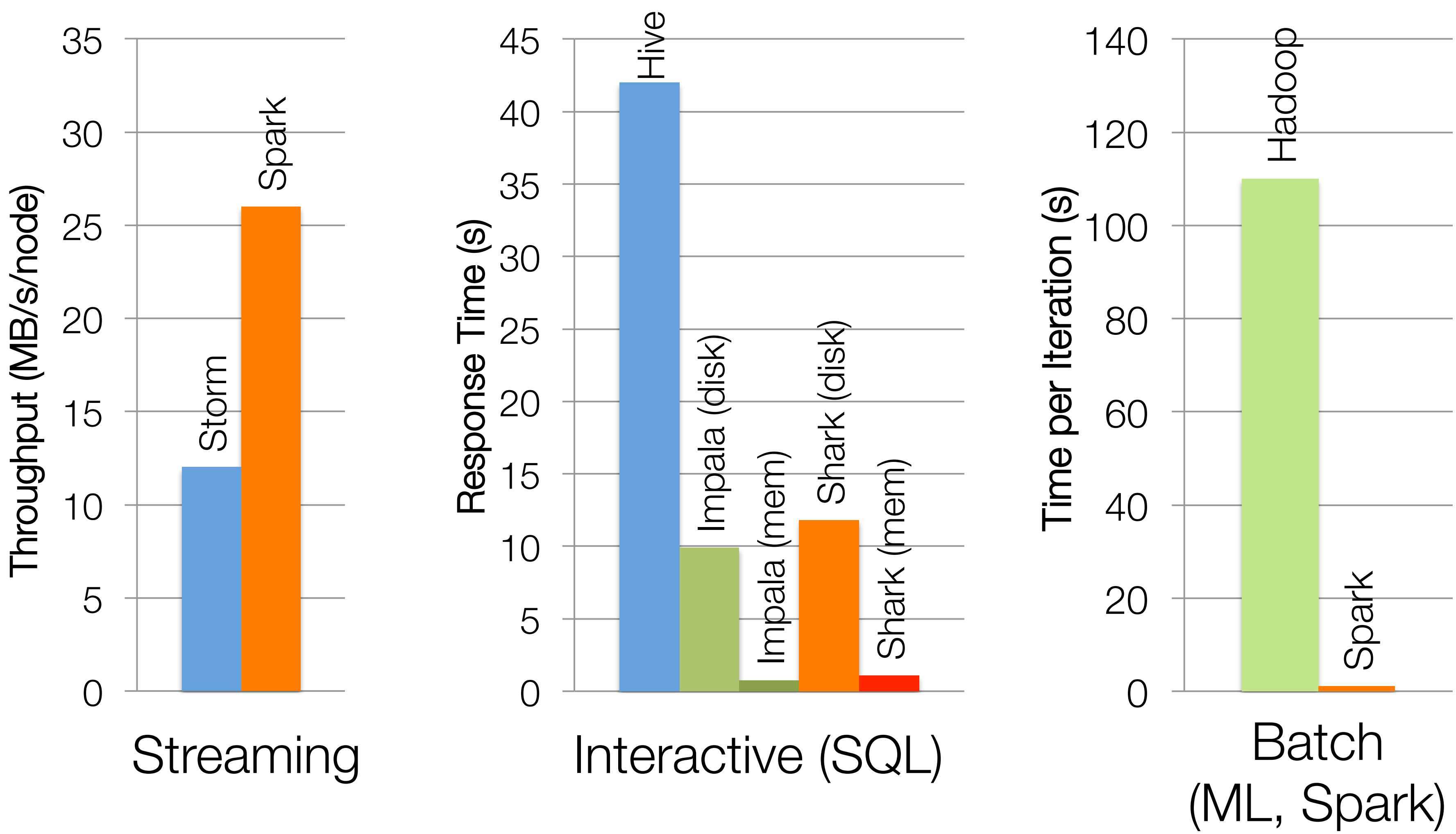
» … inefficient for graph creation, post-processing

Spark (+ GraphX): unifies graph processing & ETL

» Faster to get social network insights



Not Only General, but Fast!

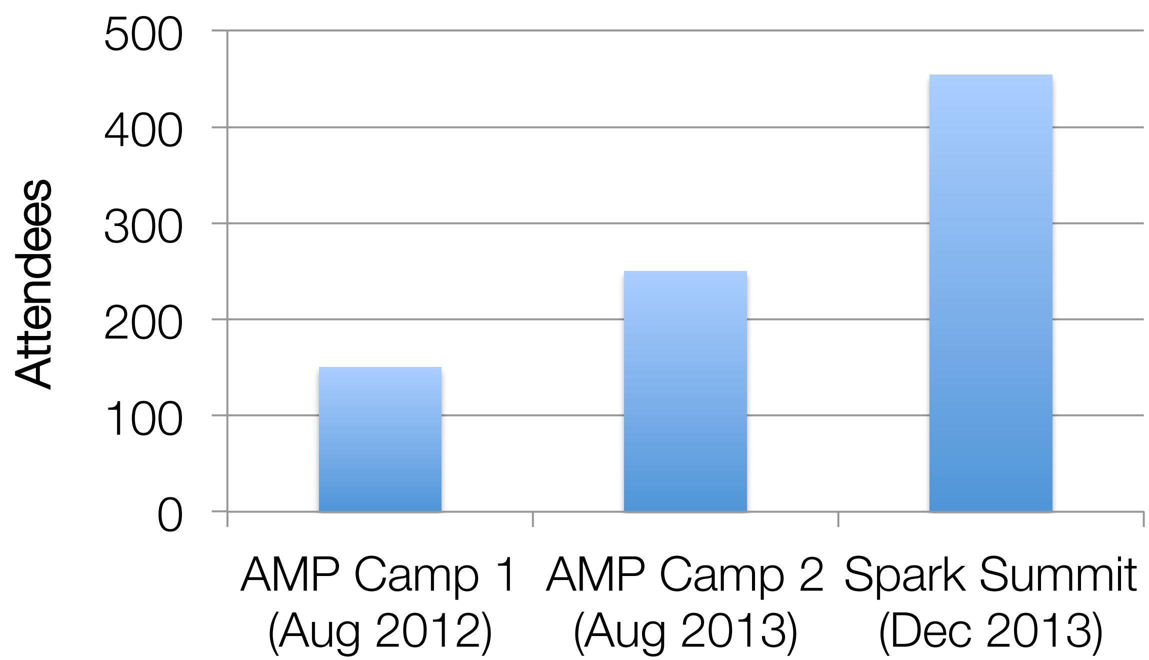


Gaining Rapid Traction

1,500+ Spark meetup users

30+ companies and over 100+ users contributing code

YAHOO! intel Adobe CONVIVA webtrends bizo TAGGED AdMobius twitter ClearStory DATA Now You See It 阿里巴巴 Alibaba.com quantiFind



Cloudera Partnership

Integrate Spark (including SparkStreaming, MLlib) with Cloudera Manager

Spark will become part of CDH

Enterprise class support and professional services available for Spark

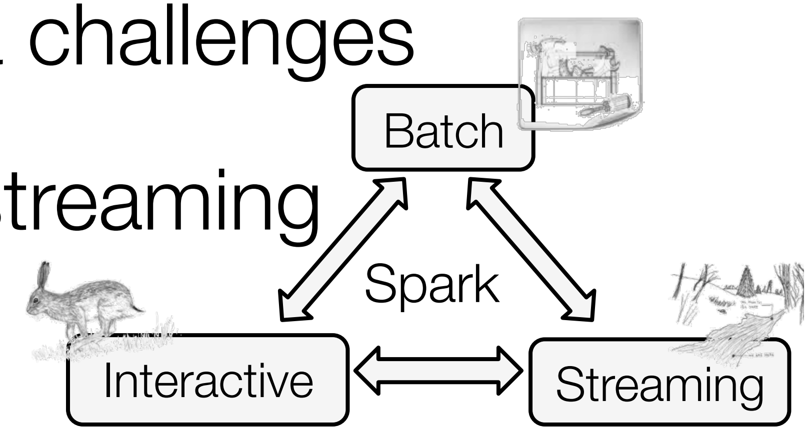
DATABRICKS

cloudera®
Ask Bigger Questions

Summary

BDAS: address next Big Data challenges

Unify batch, interactive, and streaming computations



Easy to develop sophisticate applications

» Support graph & ML algorithms, approximate queries

Witnessed significant adoption

» 30+ companies, 100+ individuals contributing code

Exciting ongoing work

» MLbase, GraphX, BlinkDB, …