DATABRICKS

The Future of Apache Spark

Patrick Wendell

A Week in Spark Development

500 patch updates

200 updates to our issue tracker

140 user list e-mails

80 merged patches

Spark's Future

Spark has seen rapid growth in the last year… where are we going now?

Spark releases and developer process

Technical roadmap over future releases

Goal of the Spark project

Empower data scientists and engineers

Expressive, clean APIs

Unified runtime across many environments

Powerful standard libraries

API stability

In 1.0+ Spark has well defined public API's and well defined experimental API's

Apps written against Spark API will be portable in new versions

Patches that break our API automatically fail our build

Developer-friendly release cadence

Minor releases every 3 months

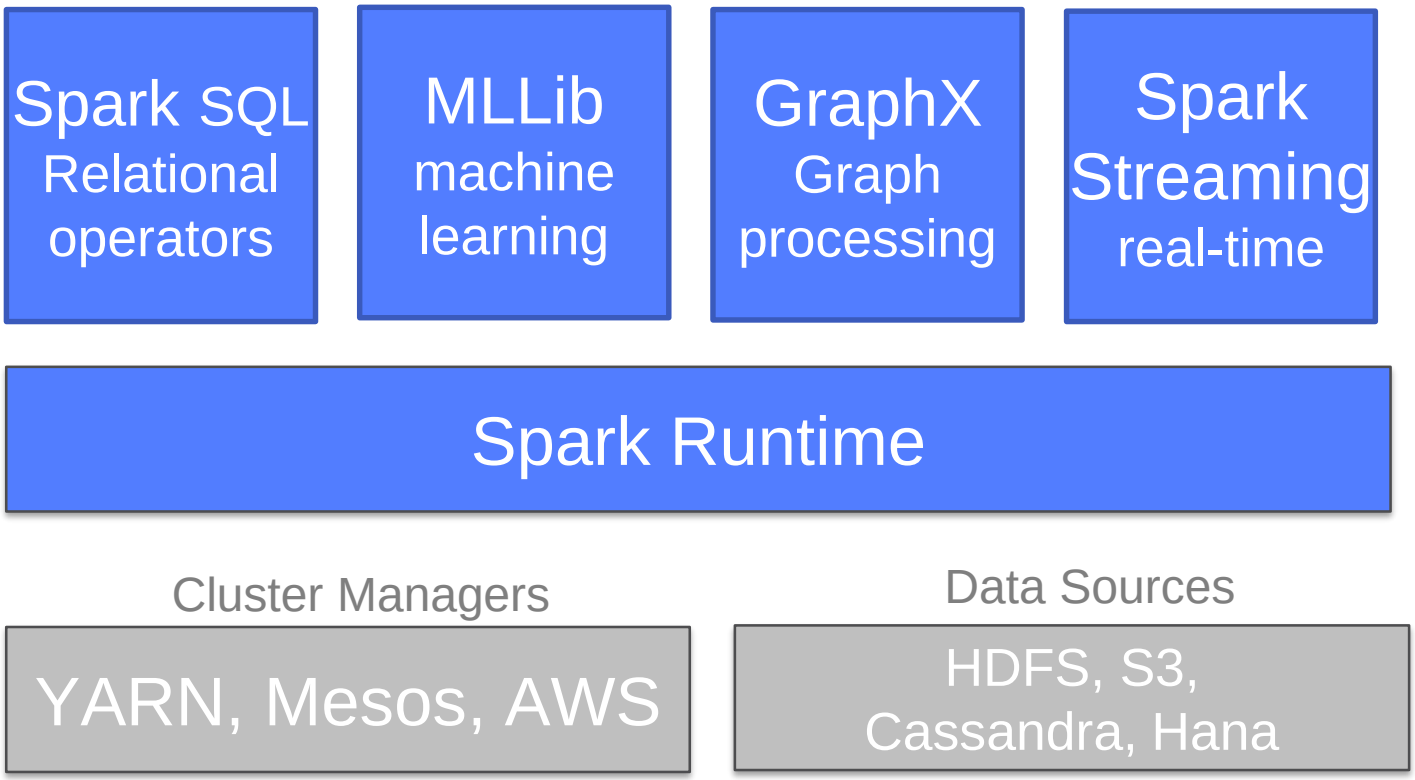
1.1 (August), 1.2, 1.3

Maintenance releases with fixes as necessary

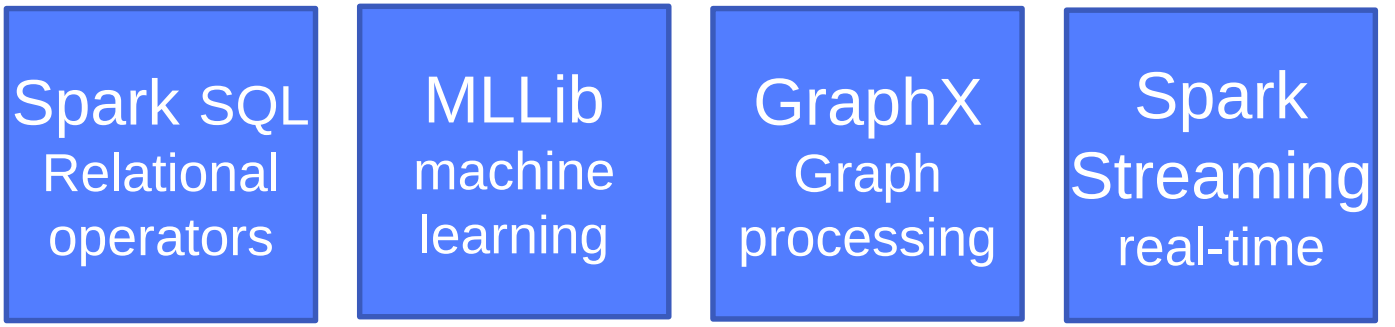
1.0.1, 1.0.2, etc

Extremely conservative about patch releases

The Spark Stack



The Spark Stack



Newer, focused on adding capabilities

More mature, focus on optimization and pluggability

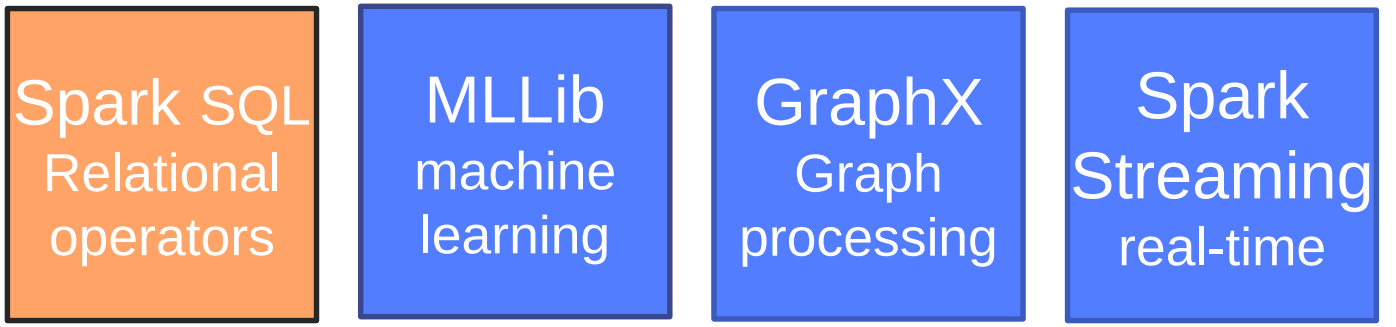
The future of Spark is libraries

Critical component of any successful runtime

Packaged and distributed with Spark to provide full inter-operability

Lead by experts in respective fields, highly curated and integrated with Spark core API

The Spark Stack



Spark SQL

Growing faster than any other component

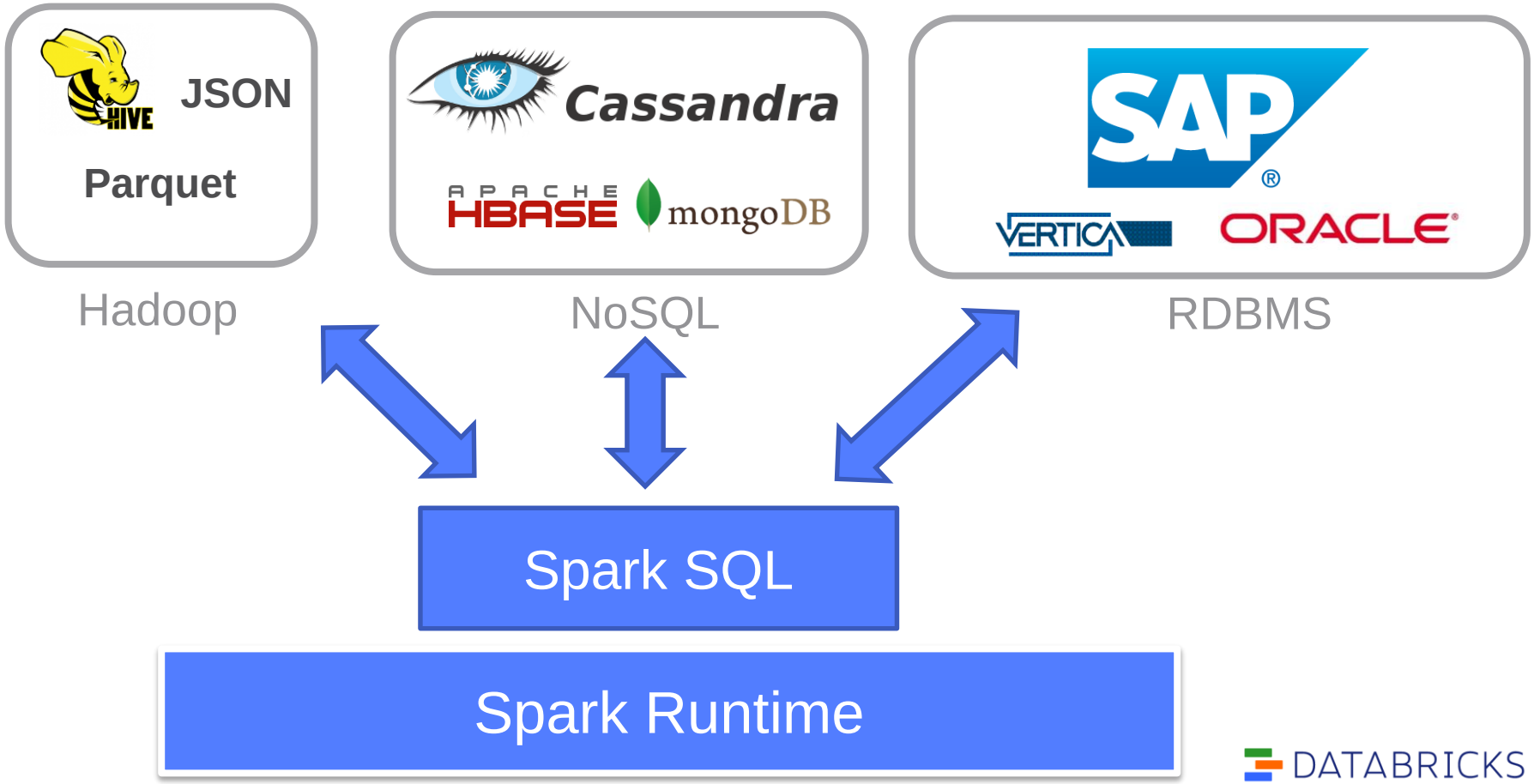
Support for SQL language and notion of typed schema RDDs

Focuses going forward:

- *Optimization* (code gen, faster joins, etc)
- *Language extensions* (towards SQL92)
- *Integration* (next slide…)

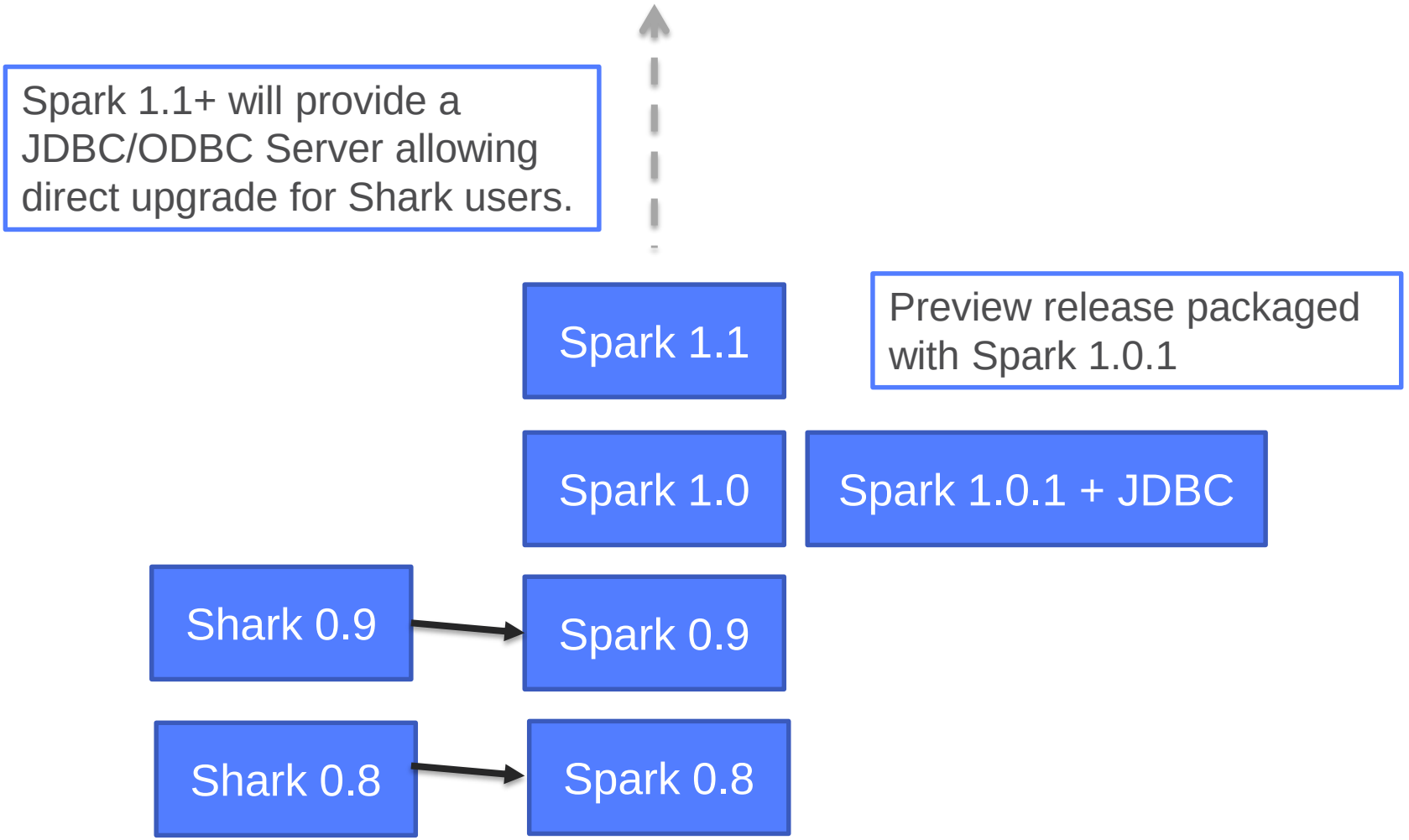
Spark SQL and SchemaRDD

Will facilitate deeper integration with other systems

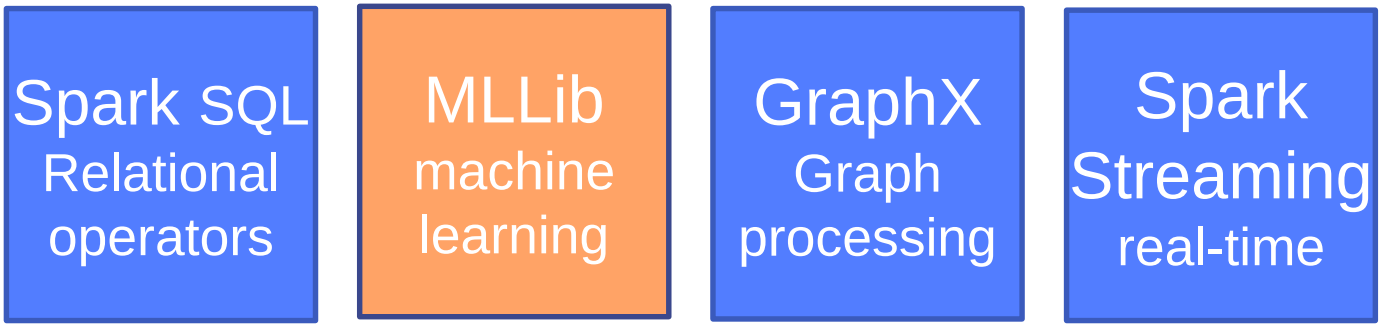


DATABRICKS

Spark SQL and Shark



The Spark Stack



Newer, focused on adding capabilities

More mature, focus on optimization and plugability

MLlib

Second fastest growing component ☺

MLLib 1.0 has about ~15 algorithms

MLLib 1.1 should roughly double that…

traditional descriptive statistics:

sampling, correlation, estimators, tests

learning algorithms:

NMF, Sparse SVD, LDA…

SparkR

Fast

Scalable

Expressive

Numerical

Packages

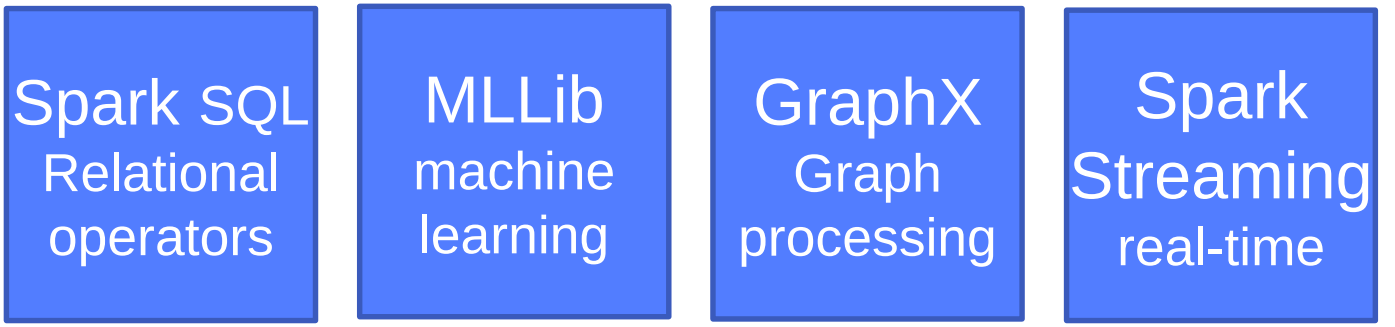
Interactive

Make SparkR “production ready” (Alteryx and Databricks).

Integration with Mllib.

Consolidating the the data frame and RDD concepts.

The Spark Stack



Newer, focused on adding capabilities

More mature, focus on optimization and pluggability

Notable trends

Hardware

Memory prices continue to fall, 256+GB machines not uncommon

SSD's becoming widely deployed

Software

Tachyon and other cluster memory managers

Spark Core

Allow extension/innovation by defining internal API's:

Internal storage API

- Support for SSDs
- Shared memory systems like Tachyon,
  - and (eventually) HDFS caching/DDMs.

Spark shuffle API

- Sort-based shuffle
- Pipelined shuffle

Timeline

Spark 1.0.1
JSON support in Spark SQL

Spark 1.1
Generalized shuffle interface
MLLib stats algorithms
JDBC server
Sort-based shuffle*

Spark 1.2
Refactored storage support

Spark 1.3+
SparkR

I've only scratched the surface…

Streaming: new data sources and tighter flume integration

Graphx: optimizations and API stability

Core: Elastic scaling on YARN, user-defined metrics and counters

[Your work here]

Wrapping it all up

Spark will grow substantially in the next year

Focus is on libraries and improving core internals for future innovation

Release process and cadence provides users with stable releases despite fast growth

DATABRICKS

Thank You!