

“Big Data” in HEP: A comprehensive use case study - Highlights

[Oliver Gutsche](#), Matteo Cremonesi,, Bo Jayatilaka, Jim Kowalkowski, Saba Sehrish, Cristina Mantilla Suárez,
Nhan Tran - Fermi National Accelerator Laboratory
Peter Elmer, Jim Pivarski, Alexey Svyatkovskiy - Princeton University

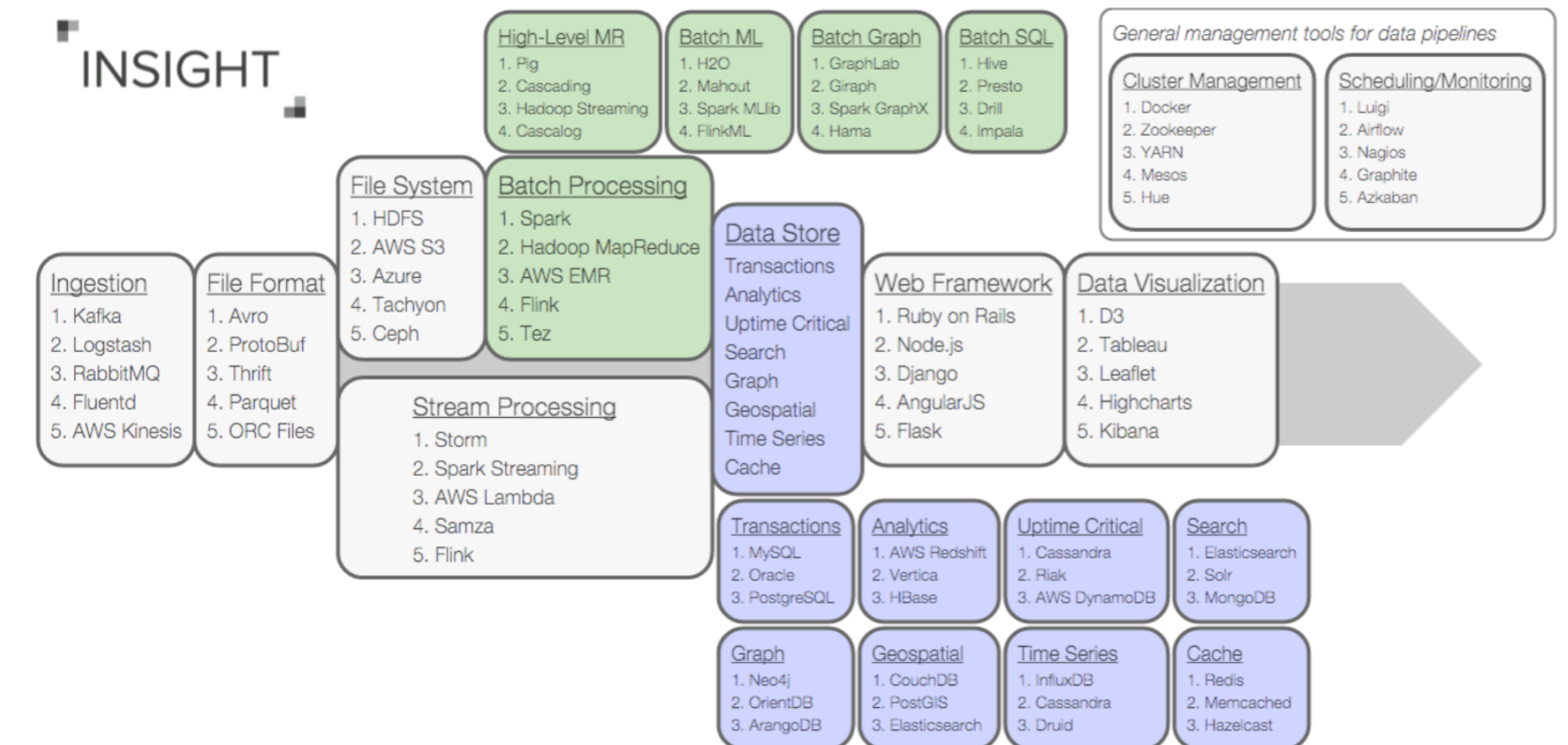
22nd International Conference on Computing in High energy and Nuclear Physics, 10.-14. October 2016

Big Data

- New toolkits and systems collectively called “Big Data” technologies have emerged to support the analysis of PB and EB datasets in industry.

- Our goals applying these technologies to HEP analysis challenge:

- Reduce time-to-physics
- Educate our graduate students and post docs to use industry-based technologies
 - Improves chances on the job market outside academia
 - Increases the attractiveness of our field
- Use tools developed in larger communities reaching outside of our field



- We want to **use an active LHC Run 2 analysis**, searching for dark matter with the CMS detector, as a **testbed for “Big Data” technologies**

- Starting point: **Apache Spark**

Usability tests

We are looking at the “physicist” use case, we are not assuming users to be GRID and HTC experts

▪ ROOT workflow: lxplus/lxbatch cluster at CERN

▪ Spark workflow: Princeton cluster

Multi-pass workflow beta-tested with two users

Analysis requires sums of event weights as input to analysis code

- Complicated, uses a script to generate scripts: very complicated and inefficient.
 - Inefficiency could be fixed, but the complexity is a hurdle
- First pass executed serially
- Second pass submitted in batch mode (lxbatch)

- Analysis code easy to write and maintain
 - ROOT/C++ is well known in community

- Scripts designed around specific batch systems (could not be moved easily)
- Partitioning (“job splitting) handled through sophisticated suite of hand-written shell scripts
 - Relies on physical location of data (i.e. files on EOS at CERN)

- Two lines of Scala code
- Spark/Scala caches (“persists”) a dataset in the first pass in memory
 - But: Cache maintained manually
- Second pass over the same dataset mostly or entirely in-memory

Analysis code

- Scala is a new language
 - Learning curve

Bookkeeping

- Very portable (from Princeton system to lxplus in no time)
- Partitioning can use automatic or custom facilities within Spark
 - example: `RDD.repartition(numPartitions: Int)`