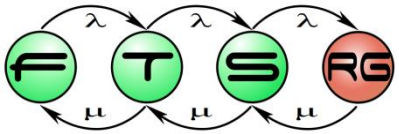
Apache Spark

2017 ősz, 8. alkalom
Kocsis Imre
ikocsis@mit.bme.hu

Budapesti Műszaki és Gazdaságtudományi Egyetem
Hibatűrő Rendszerek Kutatócsoport



APACHE HADOOP

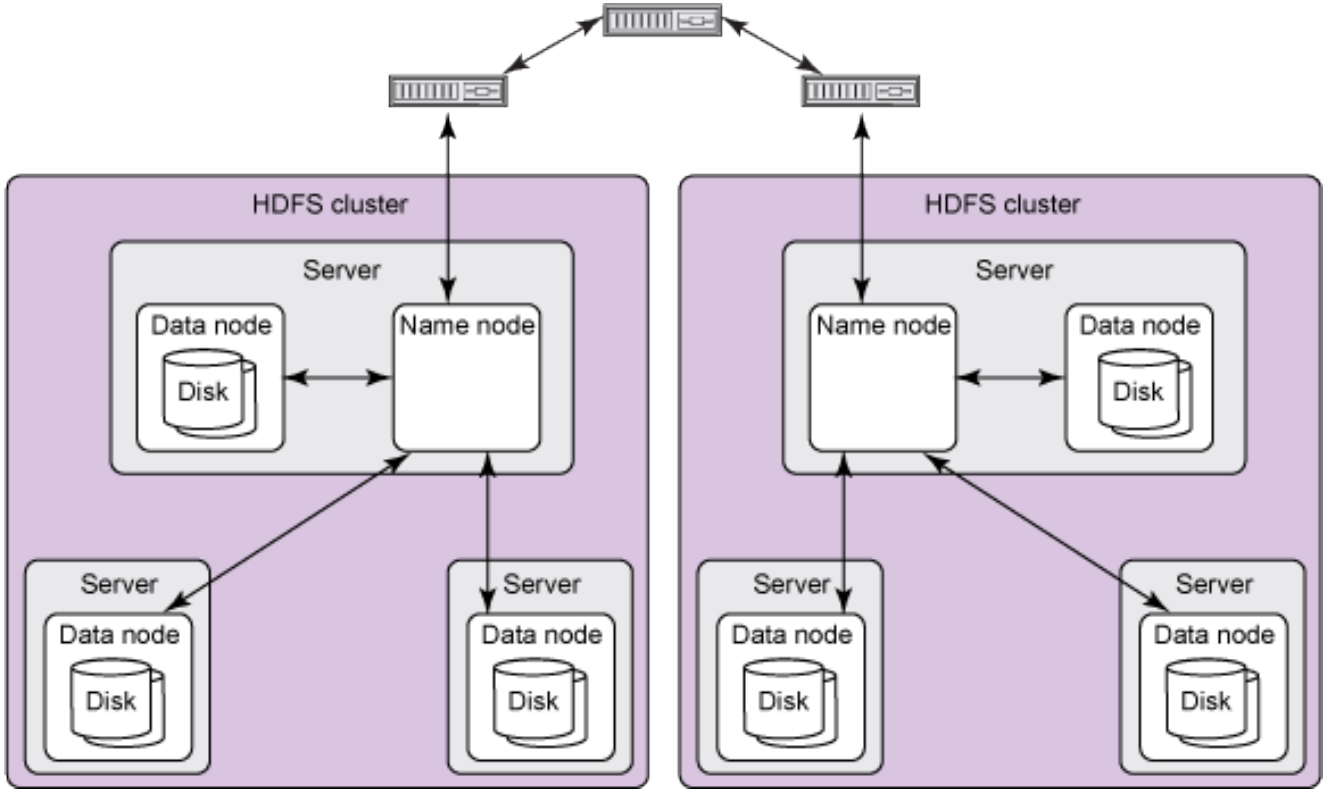
Big Data == Hadoop?

- Google MapReduce, GFS, BigTable cikkek…
- … Apache Hadoop

Law of Betteridge applies

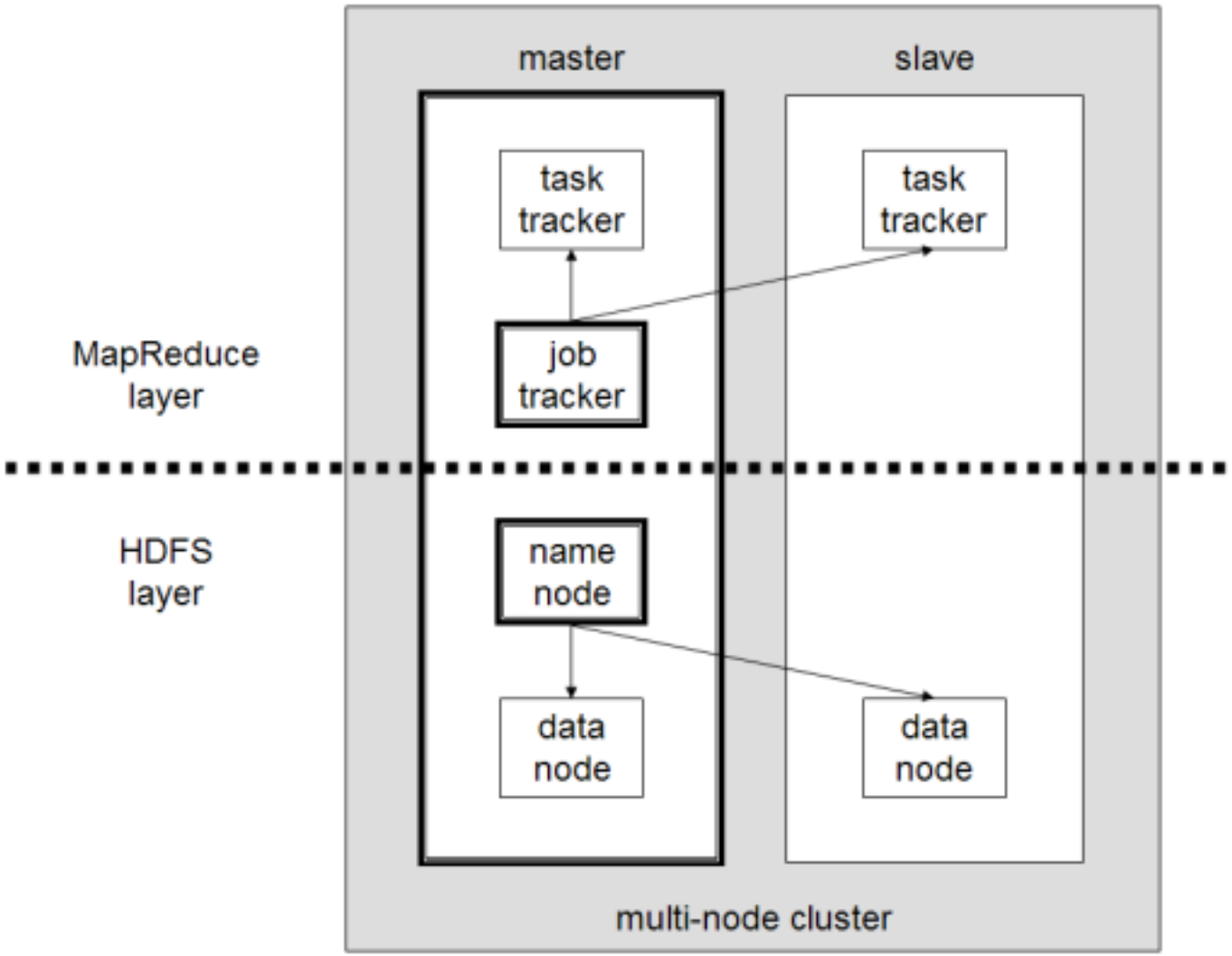
- Nyílt forráskódú, Java alapú keretrendszer
- Hadoop Distributed File System (HDFS)
- Eredetileg: MapReduce programozási paradigma
- Ráépülő/kiegészítő/kapcsolódó projektek
- “Pay for support” disztribúciók – Cloudera, Hortonworks, MapR, …

HDFS (leegyszerűsítve)

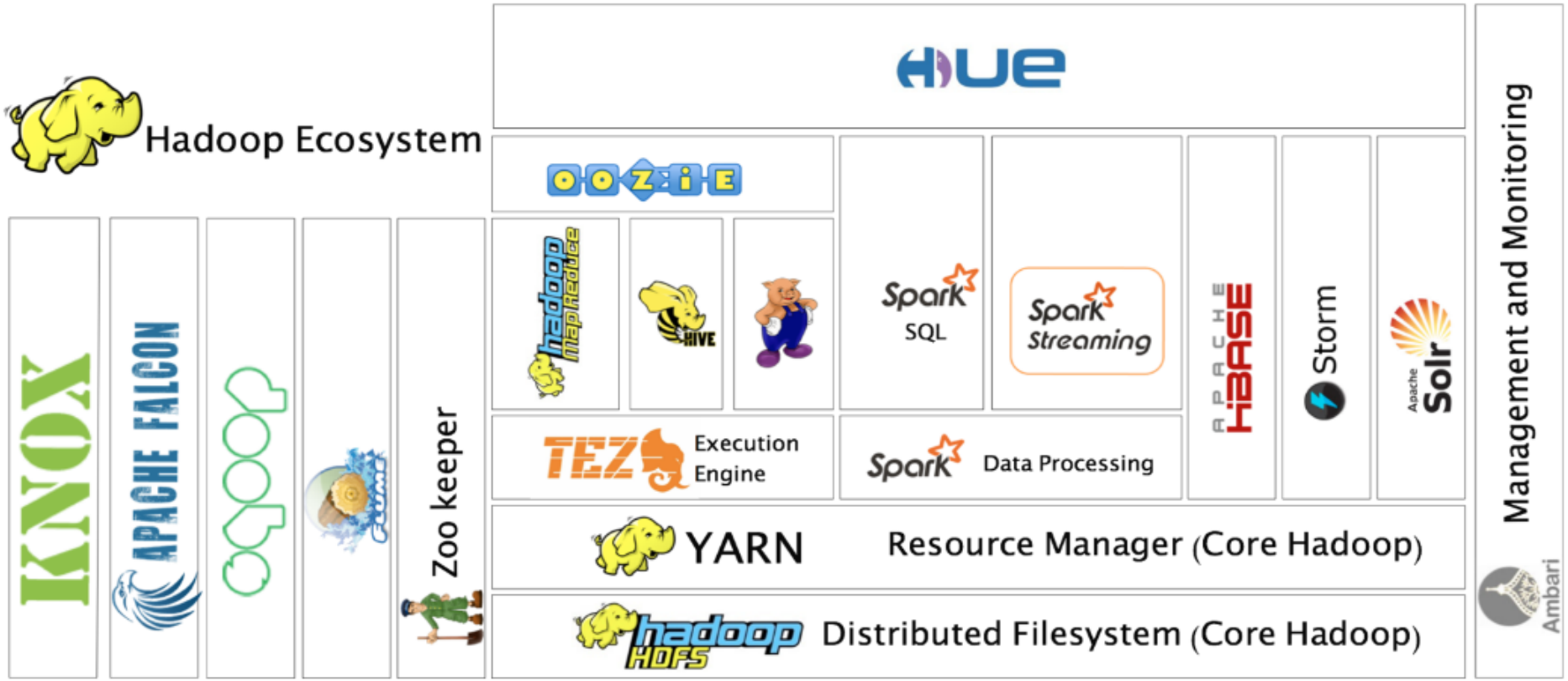


~Klasszikus állományrendszer
Nagy (64MB) blokkok, szétterítve és replikálva

Hadoop (eredetileg)



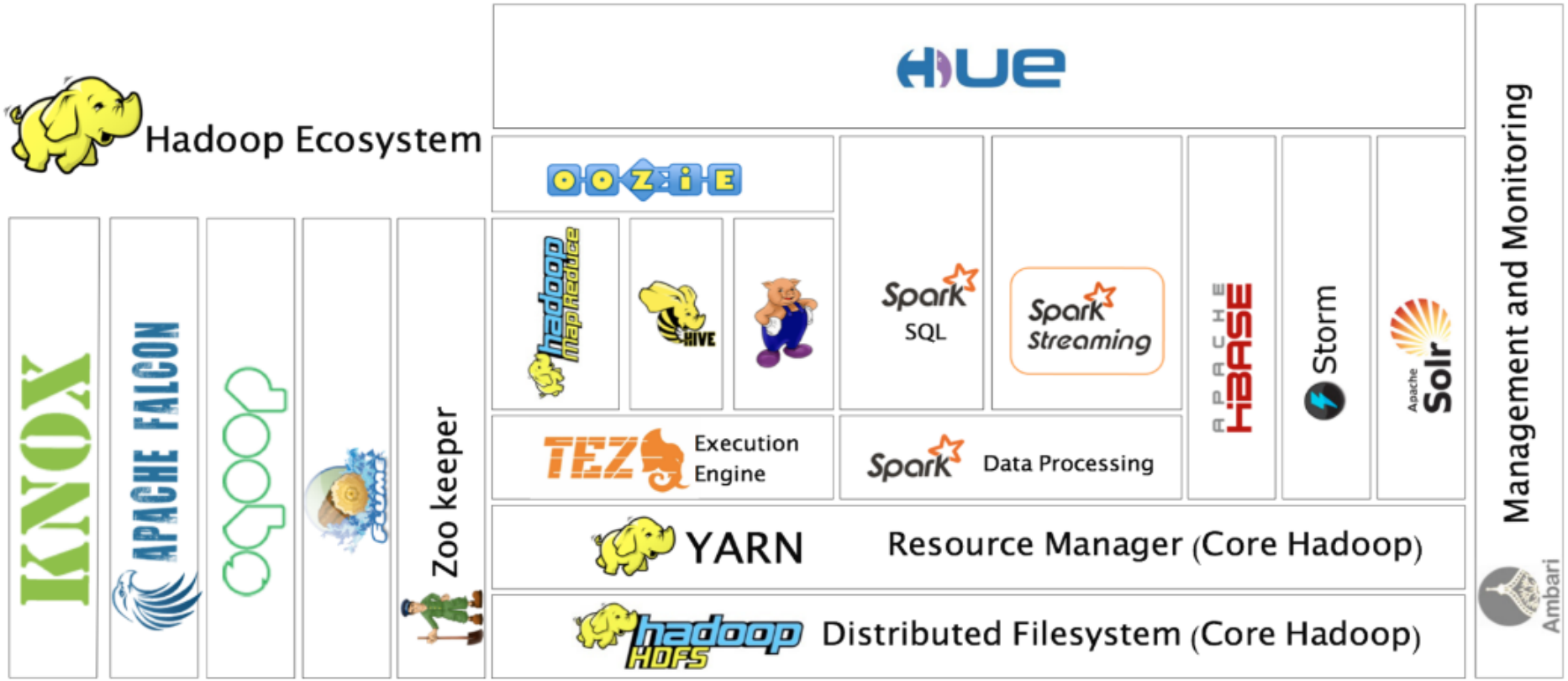
Hadoop ökoszisztéma – egy nézet



Forrás: http://slides.com/racku/deck-4-3-2#/4/13

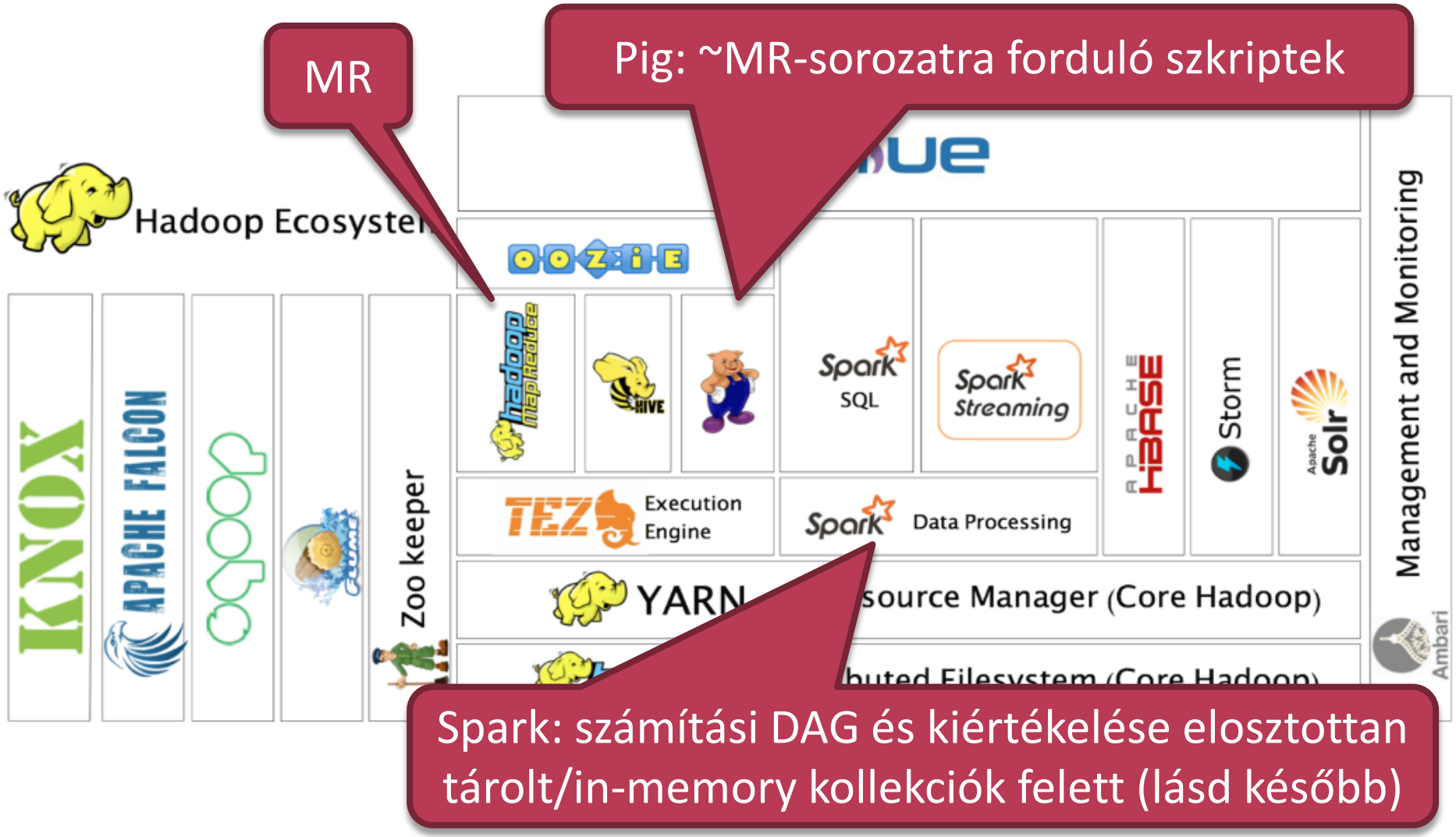
Hadoop ökoszisztéma – egy nézet

Rengeteg további kombináció és lehetséges komponens!



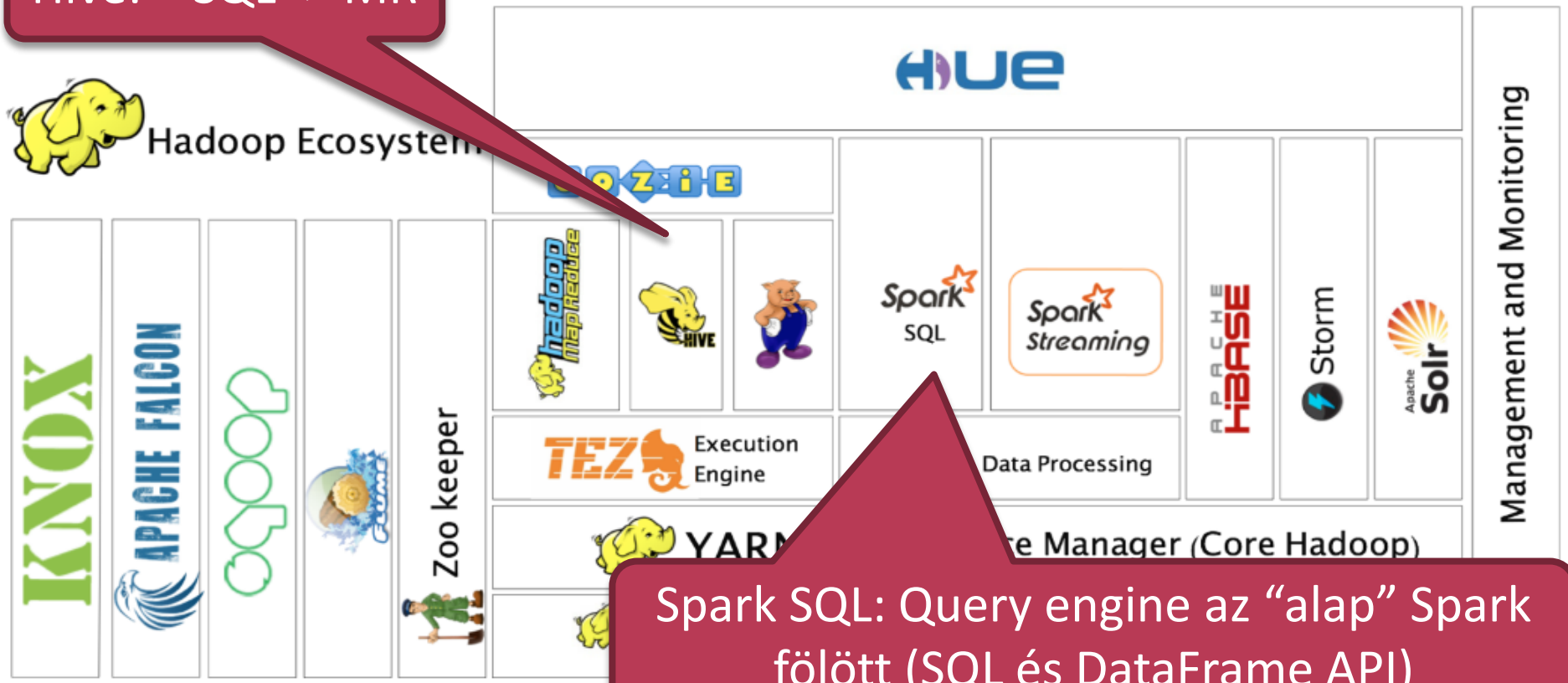
Forrás: http://slides.com/racku/deck-4-3-2#/4/13

Hadoop ökoszisztéma – adatfeldolgozás



Hadoop ökoszisztéma – SQL (look who's back)

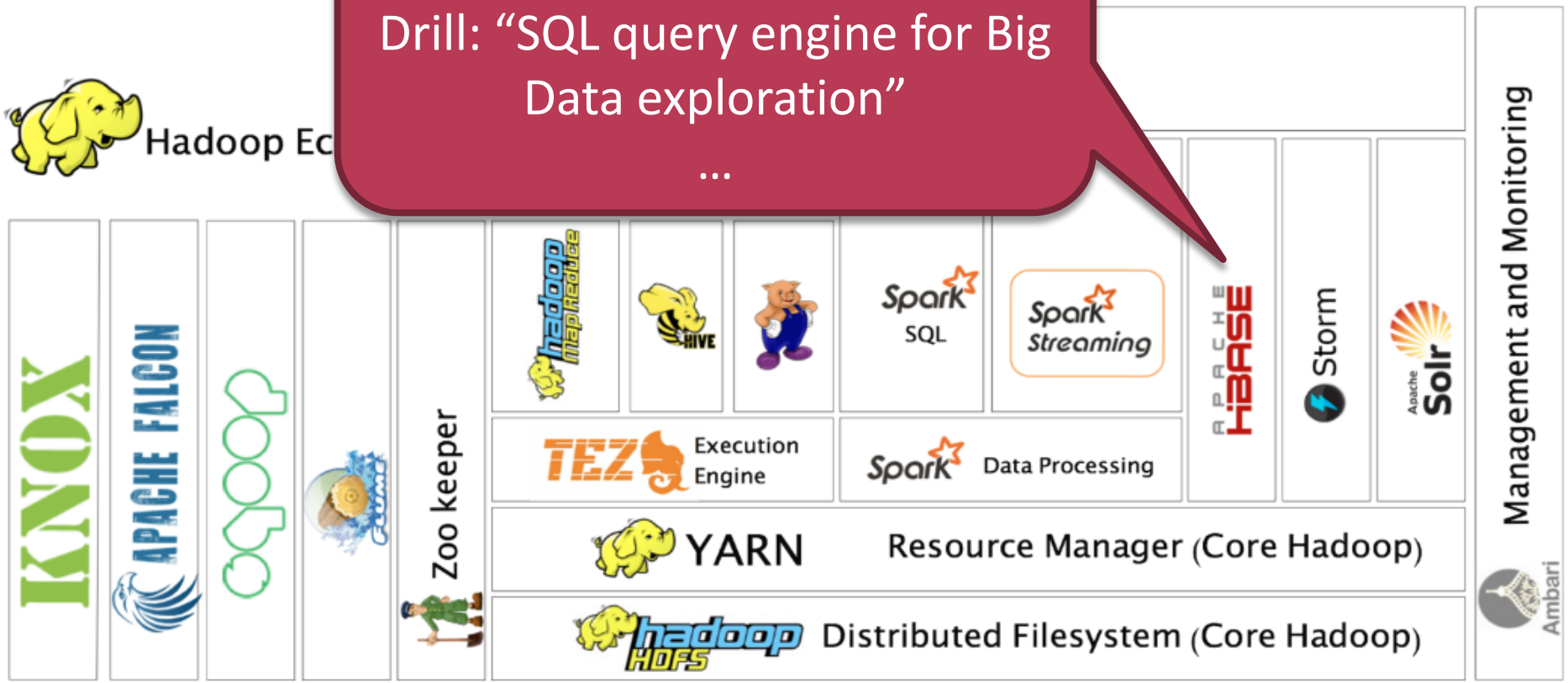
Hive: ~ SQL -> MR



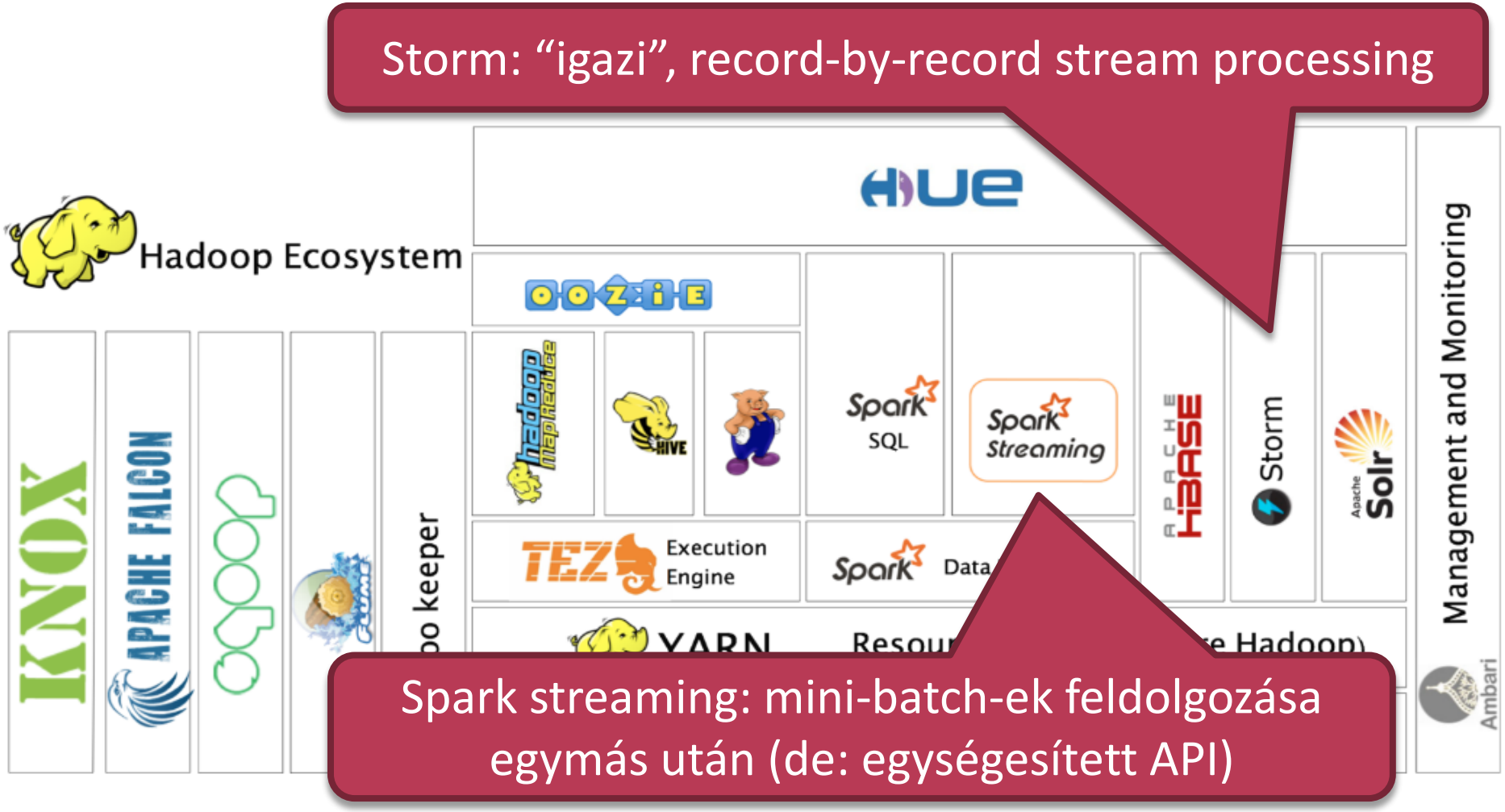
Spark SQL: Query engine az "alap" Spark fölött (SQL és DataFrame API)

Hadoop ökoszisztéma – NoSQL

HBase: columnar storage
Impala: MPP SQL query engine
Drill: “SQL query engine for Big Data exploration”
…

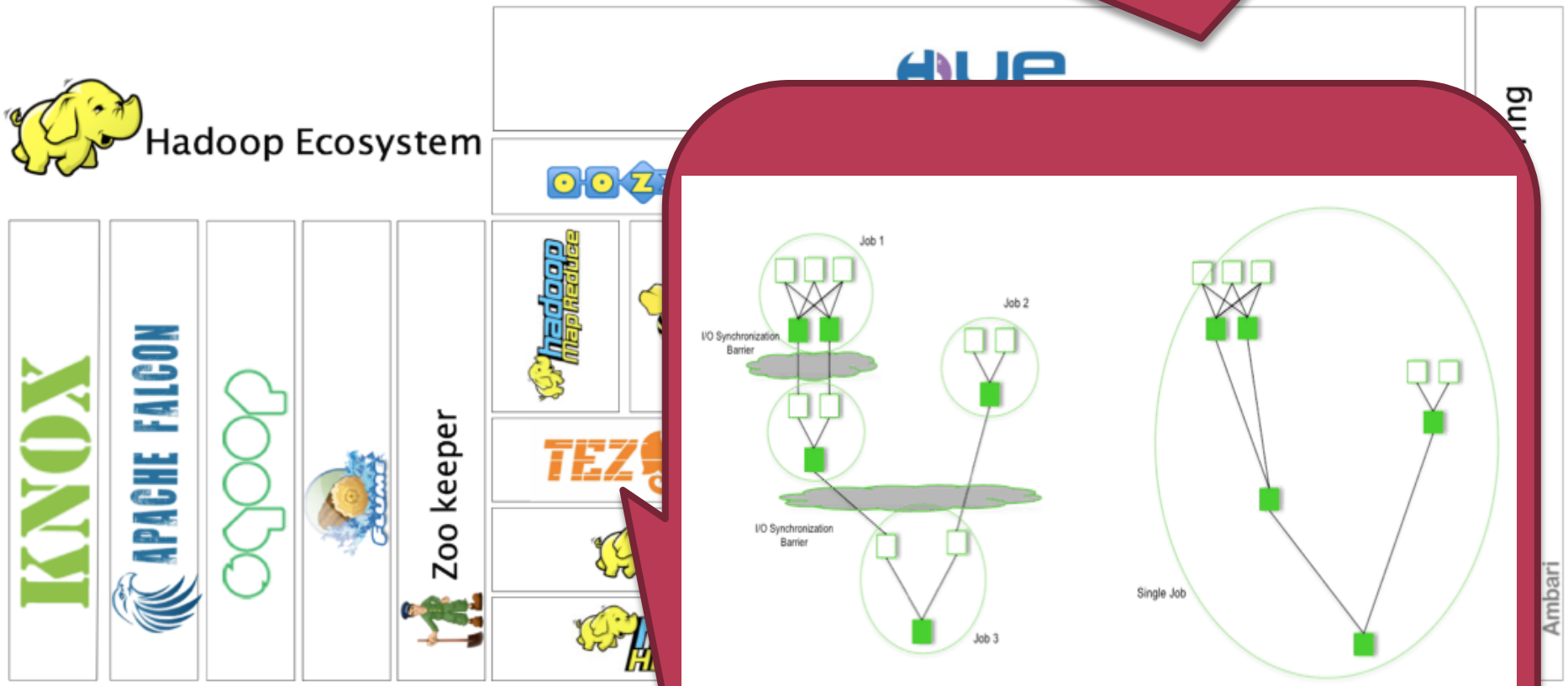


Hadoop ökoszisztéma – Stream processing



Hadoop ökoszisztéma – egyéb

Hadoop UI Experience (Cloudera)



SPARK

Apache Spark

- Eredetileg: UC Berkley AMPLab (2009)
  - 2014: top-level Apache project
  - Databricks: az eredeti fejlesztő (még PhD hallgatóként) cége

- (Eredeti) MR: stabil tár -> Map -> Reduce -> stabil tár
  - Aciklikus adatfolyam: hibatűrő, környezet tud ütemezni
- Munkakészletet (working set) használó alkalmazások?
  - Iteratív algoritmusok
  - Interaktív munkamenetek (és toolok)

- Praktikusan
  - Gyors
  - Interaktív (REPL szintig)
  - Nem MapReduce-t kell írni(!)

Programozási modell

- Adatmodell: *RDD - Resilient Distributed Dataset*
  - Objektumok immutábilis, partícionált kollekciója
  - Létrehozás: stabil tárban (elosztottan) tárolt adatok feletti *transzformációkkal*
  - Rekurzívan -> DAG
  - Köztestárazás, adott esetben “kényszerítéssel
  - Cél: diszkhozzáférés minimalizálása
  - Lusta kiértékelés

- “Kiolvasás”: akciók

Wordcount

Python Scala Java

```python
text_file = sc.textFile("hdfs://...")
counts = text_file.flatMap(lambda line: line.split(" ")) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile("hdfs://...")
```

Python Scala Java

```java
JavaRDD<String> textFile = sc.textFile("hdfs://...");
JavaPairRDD<String, Integer> counts = textFile
    .flatMap(s -> Arrays.asList(s.split(" ")).iterator())
    .mapToPair(word -> new Tuple2<>(word, 1))
    .reduceByKey((a, b) -> a + b);
counts.saveAsTextFile("hdfs://...");
```

https://spark.apache.org/examples.html

Java MR: ~60 sor

Főbb transzformációk (RDD API)

Transformation	Meaning
map(*func*)	Return a new distributed dataset formed by passing each element of the source through a function *func*.
filter(*func*)	Return a new dataset formed by selecting those elements of the source on which *func* returns true.
flatMap(*func*)	Similar to map, but each input item can be mapped to 0 or more output items (so *func* should return a Seq rather than a single item).
mapPartitions(*func*)	Similar to map, but runs separately on each partition (block) of the RDD, so *func* must be of type Iterator<T> => Iterator<U> when running on an RDD of type T.
mapPartitionsWithIndex(*func*)	Similar to mapPartitions, but also provides *func* with an integer value representing the index of the partition, so *func* must be of type (Int, Iterator<T>) => Iterator<U> when running on an RDD of type T.
sample(*withReplacement*, *fraction*, *seed*)	Sample a fraction *fraction* of the data, with or without replacement, using a given random number generator seed.
union(*otherDataset*)	Return a new dataset that contains the union of the elements in the source dataset and the argument.
intersection(*otherDataset*)	Return a new RDD that contains the intersection of elements in the source dataset and the argument.
distinct([*numTasks*]))	Return a new dataset that contains the distinct elements of the source dataset.

Főbb transzformációk (RDD API)

groupByKey([*numTasks*])	When called on a dataset of (K, V) pairs, returns a dataset of (K, Iterable<V>) pairs. **Note:** If you are grouping in order to perform an aggregation (such as a sum or average) over each key, using `reduceByKey` or `aggregateByKey` will yield much better performance. **Note:** By default, the level of parallelism in the output depends on the number of partitions of the parent RDD. You can pass an optional `numTasks` argument to set a different number of tasks.
reduceByKey(*func*, [*numTasks*])	When called on a dataset of (K, V) pairs, returns a dataset of (K, V) pairs where the values for each key are aggregated using the given reduce function *func*, which must be of type (V,V) => V. Like in `groupByKey`, the number of reduce tasks is configurable through an optional second argument.
aggregateByKey(*zeroValue*)(*seqOp*, *combOp*, [*numTasks*])	When called on a dataset of (K, V) pairs, returns a dataset of (K, U) pairs where the values for each key are aggregated using the given combine functions and a neutral "zero" value. Allows an aggregated value type that is different than the input value type, while avoiding unnecessary allocations. Like in `groupByKey`, the number of reduce tasks is configurable through an optional second argument.
sortByKey([*ascending*], [*numTasks*])	When called on a dataset of (K, V) pairs where K implements Ordered, returns a dataset of (K, V) pairs sorted by keys in ascending or descending order, as specified in the boolean `ascending` argument.
join(*otherDataset*, [*numTasks*])	When called on datasets of type (K, V) and (K, W), returns a dataset of (K, (V, W)) pairs with all pairs of elements for each key. Outer joins are supported through `leftOuterJoin`, `rightOuterJoin`, and `fullOuterJoin`.
cogroup(*otherDataset*, [*numTasks*])	When called on datasets of type (K, V) and (K, W), returns a dataset of (K, (Iterable<V>, Iterable<W>)) tuples. This operation is also called `groupWith`.
cartesian(*otherDataset*)	When called on datasets of types T and U, returns a dataset of (T, U) pairs (all pairs of elements).

Akciók (RDD API)

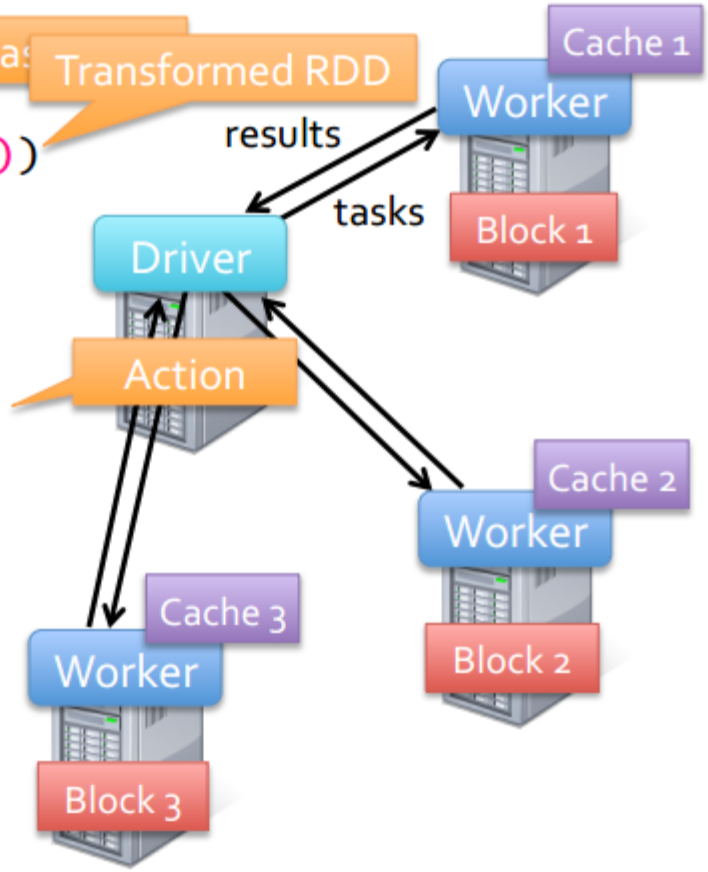
Action	Meaning
reduce(*func*)	Aggregate the elements of the dataset using a function *func* (which takes two arguments and returns one). The function should be commutative and associative so that it can be computed correctly in parallel.
collect()	Return all the elements of the dataset as an array at the driver program. This is usually useful after a filter or other operation that returns a sufficiently small subset of the data.
count()	Return the number of elements in the dataset.
first()	Return the first element of the dataset (similar to take(1)).
take(*n*)	Return an array with the first *n* elements of the dataset.
takeSample(*withReplacement*, *num*, [*seed*])	Return an array with a random sample of *num* elements of the dataset, with or without replacement, optionally pre-specifying a random number generator seed.
takeOrdered(*n*, *[ordering]*)	Return the first *n* elements of the RDD using either their natural order or a custom comparator.
saveAsTextFile(*path*)	Write the elements of the dataset as a text file (or set of text files) in a given directory in the local filesystem, HDFS or any other Hadoop-supported file system. Spark will call toString on each element to convert it to a line of text in the file.
saveAsSequenceFile(*path*) (Java and Scala)	Write the elements of the dataset as a Hadoop SequenceFile in a given path in the local filesystem, HDFS or any other Hadoop-supported file system. This is available on RDDs of key-value pairs that implement Hadoop's Writable interface. In Scala, it is also available on types that are implicitly convertible to Writable (Spark includes conversions for basic types like Int, Double, String, etc).
saveAsObjectFile(*path*) (Java and Scala)	Write the elements of the dataset in a simple format using Java serialization, which can then be loaded using `SparkContext.objectFile()`.
countByKey()	Only available on RDDs of type (K, V). Returns a hashmap of (K, Int) pairs with the count of each key.
foreach(*func*)	Run a function *func* on each element of the dataset. This is usually done for side effects such as updating an Accumulator or interacting with external storage systems. **Note**: modifying variables other than Accumulators outside of the `foreach()` may result in undefined behavior. See Understanding closures for more details.

Klaszter-nézet

Load error messages from a log into memory, then interactively search for various patterns

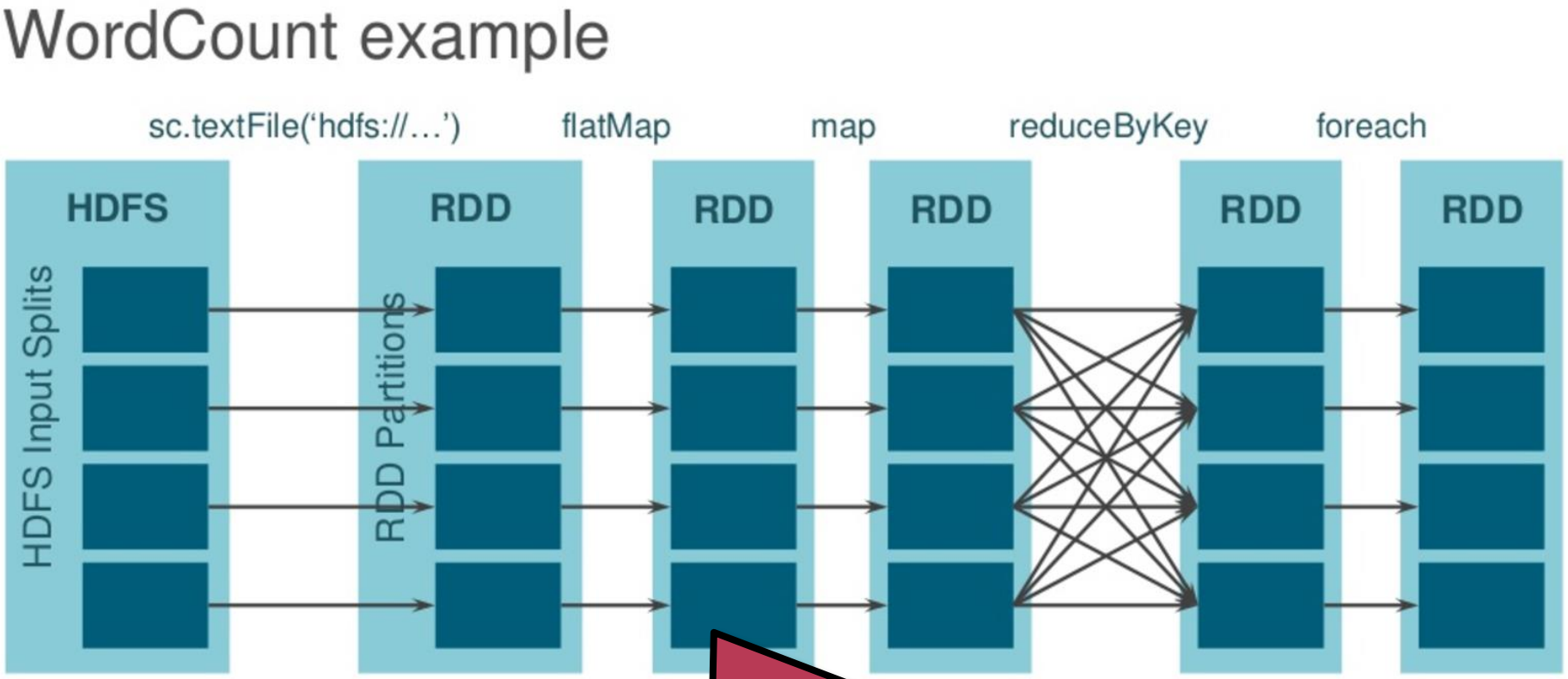
```
lines = spark.textFile("hdfs://...")
errors = lines.filter(_.startsWith("ERROR"))
messages = errors.map(_.split('\t')(2))
cachedMsgs = messages.cache()

cachedMsgs.filter(_.contains("foo")).count
cachedMsgs.filter(_.contains("bar")).count
. . .
```

Result: scaled to 1 TB data in 5-7 sec
(vs 170 sec for on-disk data)



Forrás: http://spark.apache.org/talks/overview.pdf

DAG



A "lineage" alapján a partíciók rekurzívan újraszámolhatóak (hiba, köztestár-kiürítés)

https://www.slideshare.net/AGrishchenko/apache-spark-architecture

SQL, DataFrame és Dataset API-k

- DataFrame: RDD + séma
  - “Egységesített transzformációs interfész”
  - A “megszokott” dataframe – elrejti az RDD szintet
- SQL: well…
- Dataset: Java/Scala objektumok a kollekciókban + Encoder

- Catalyst Optimizer ezen a szinten (RDD: nem)
- Várhatóan a felhasználók többsége ide fog migrálni

- Részletek: https://www.slideshare.net/databricks/deep-dive-into-catalyst-apache-spark-20s-optimizer

További könyvtárak

- Spark Streaming
  - Következő alkalom
- Mllib
  - DataFrame API, RDD: maintenance
  - Feature Extractors/Transformers/Selectors
  - Classification, regression
  - Clustering
  - Frequent Pattern Mining
  - …
- Mahout
  - “Previously on Hadoop MapReduce, Mahout has switched to using Spark as the backend” (spark.apache.org)
- GraphX
  - Pregel API
- …

Javasolt eszköz + irodalom

- https://hub.docker.com/r/jupyter/all-spark-notebook/
  - Docker van Windows-ra is!
  - Local mode!
  - Kitematic: PowerShell averzió esetén javasolt
- Megértéshez:
  - https://spark.apache.org/docs/latest/
- RDD szintű programozás (Python is!):
  - https://spark.apache.org/docs/latest/rdd-programming-guide.html
- Datasets, DataFrames, SQL (Python is!):
  - https://spark.apache.org/docs/latest/sql-programming-guide.html