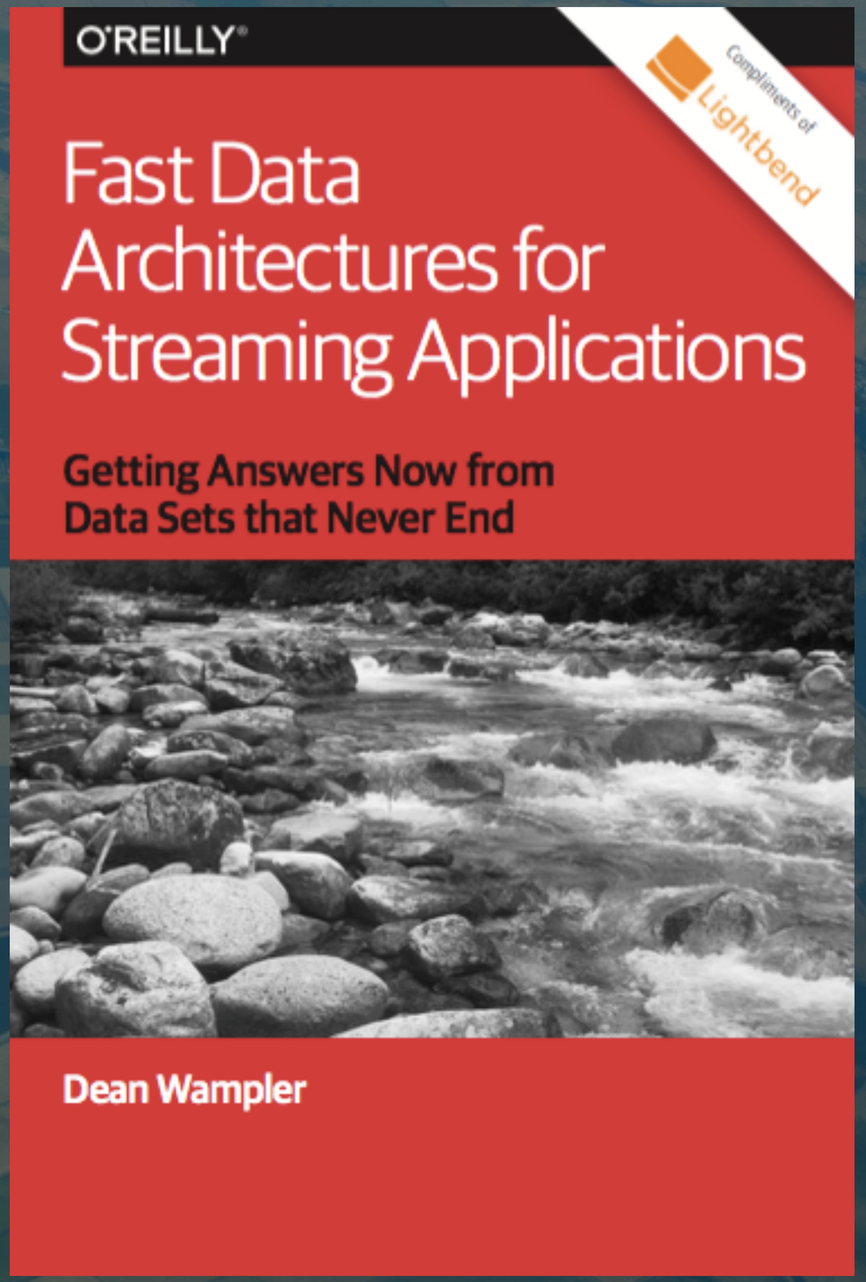
Stream All the Things!

Architectures for Data that Never Ends

Dean Wampler, Ph.D.
dean@lightbend.com
@deanwampler

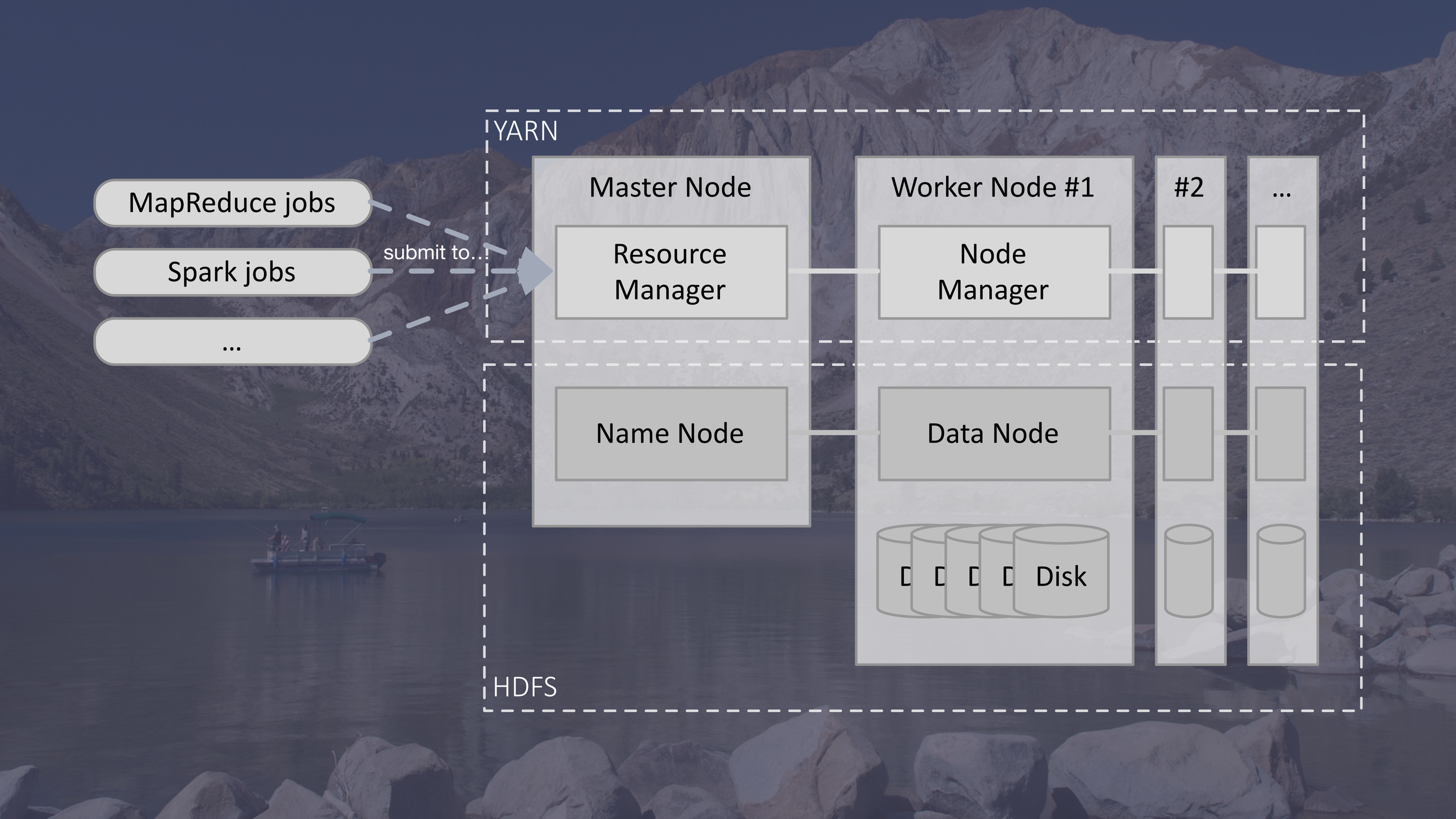
Free as in 🍺

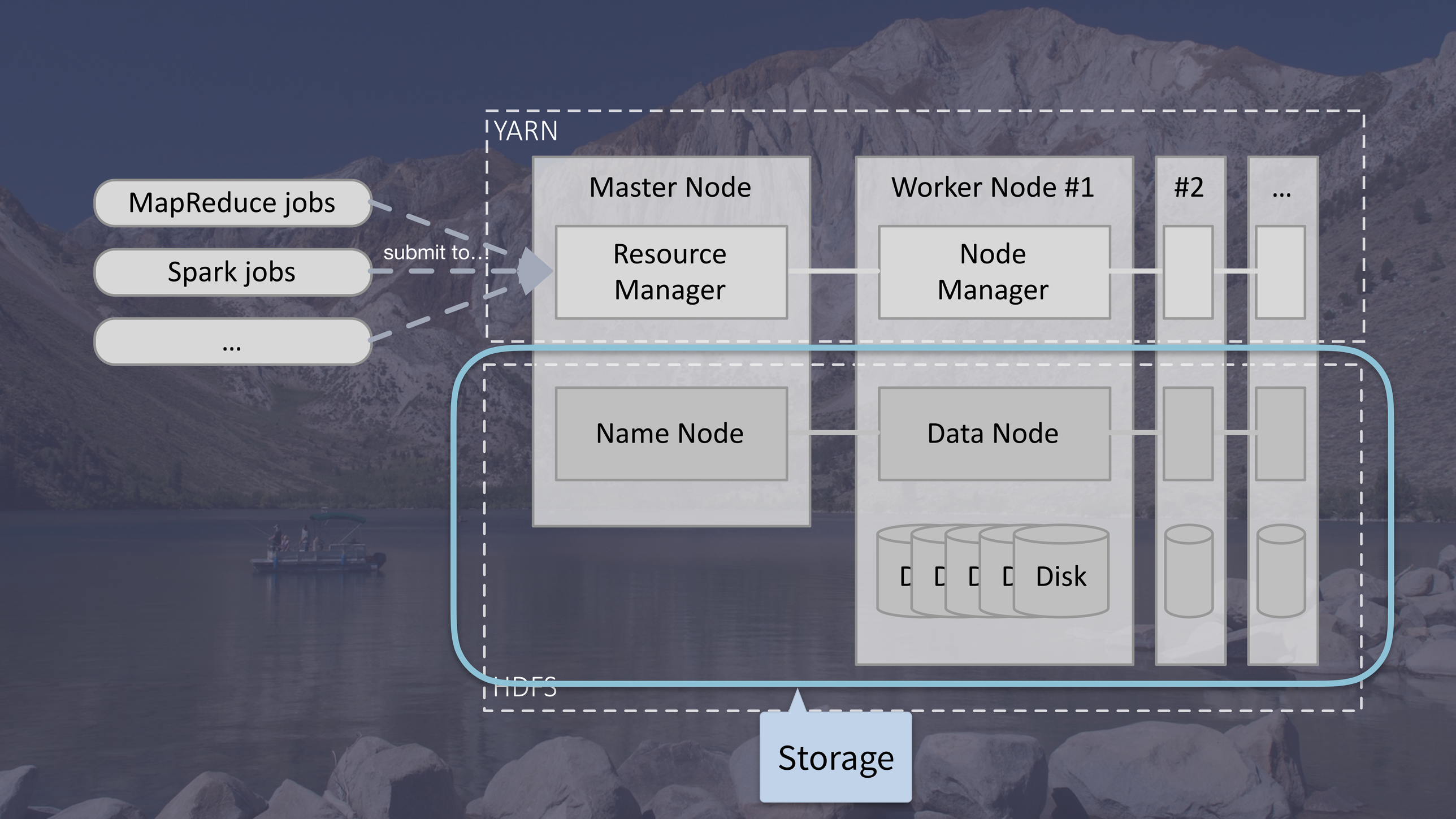
lightbend.com/fast-data-platform
(2nd Edition coming soon!)

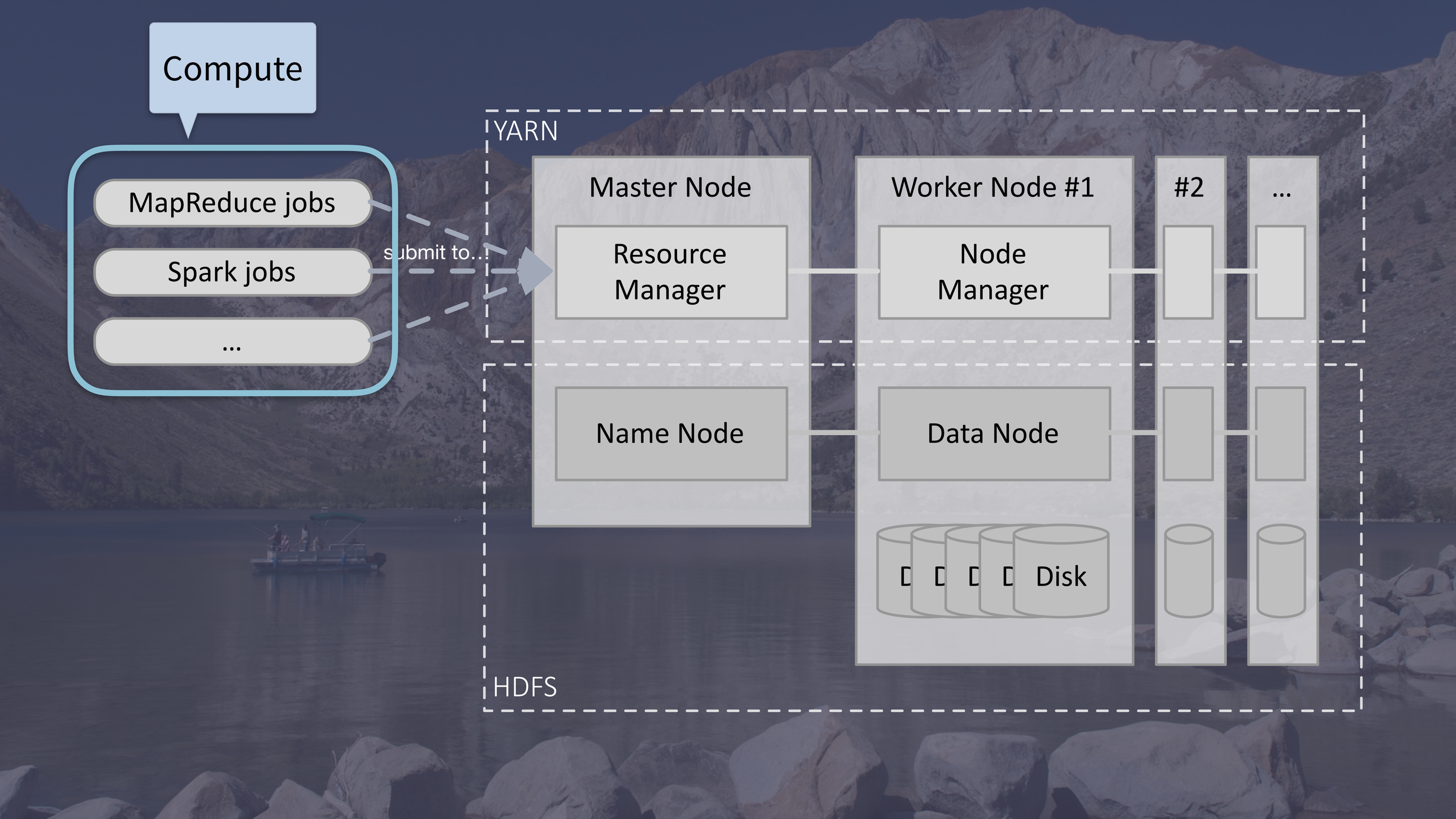


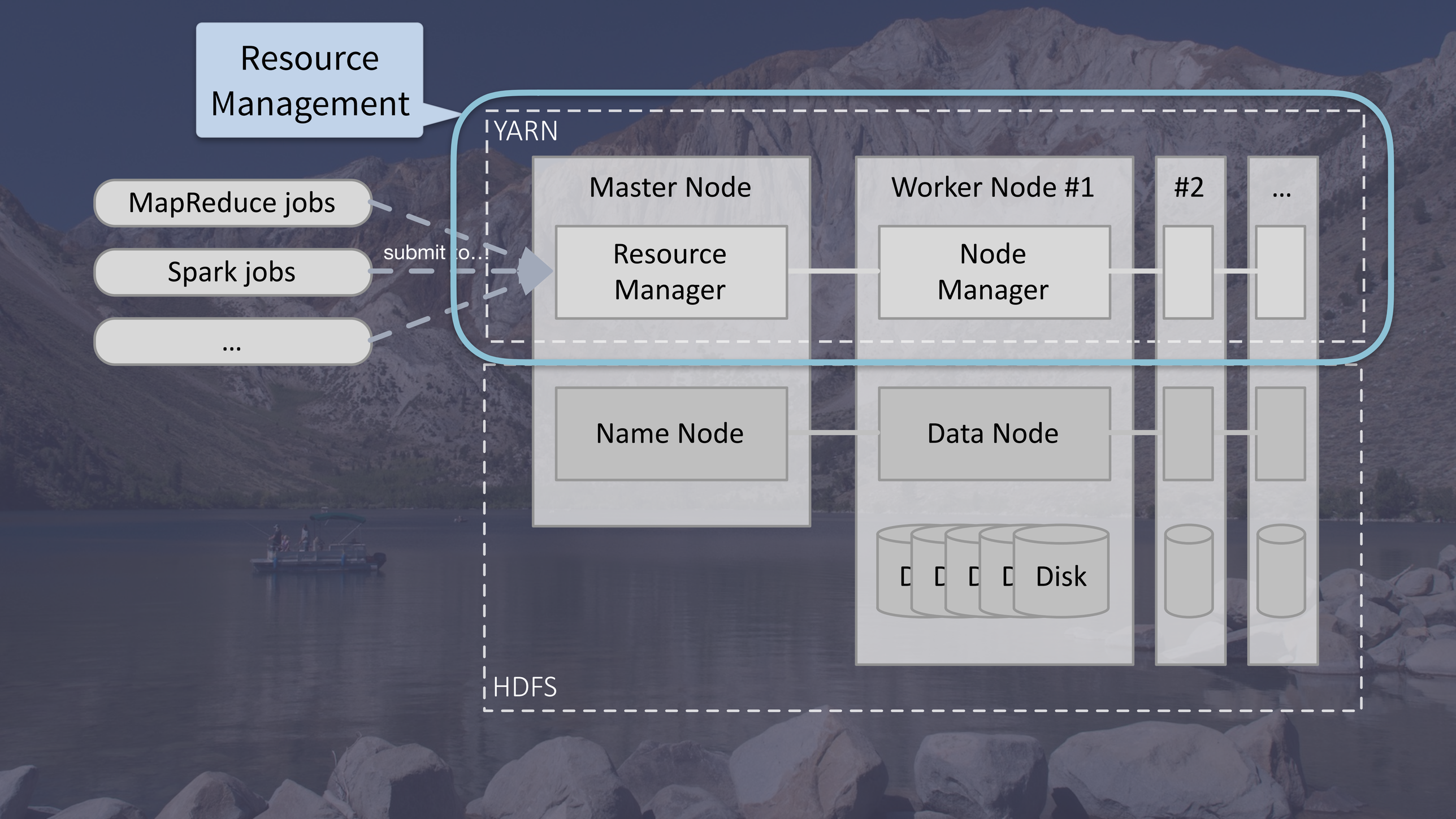
Streaming in Context…

Hadoop: Classic Batch Architecture

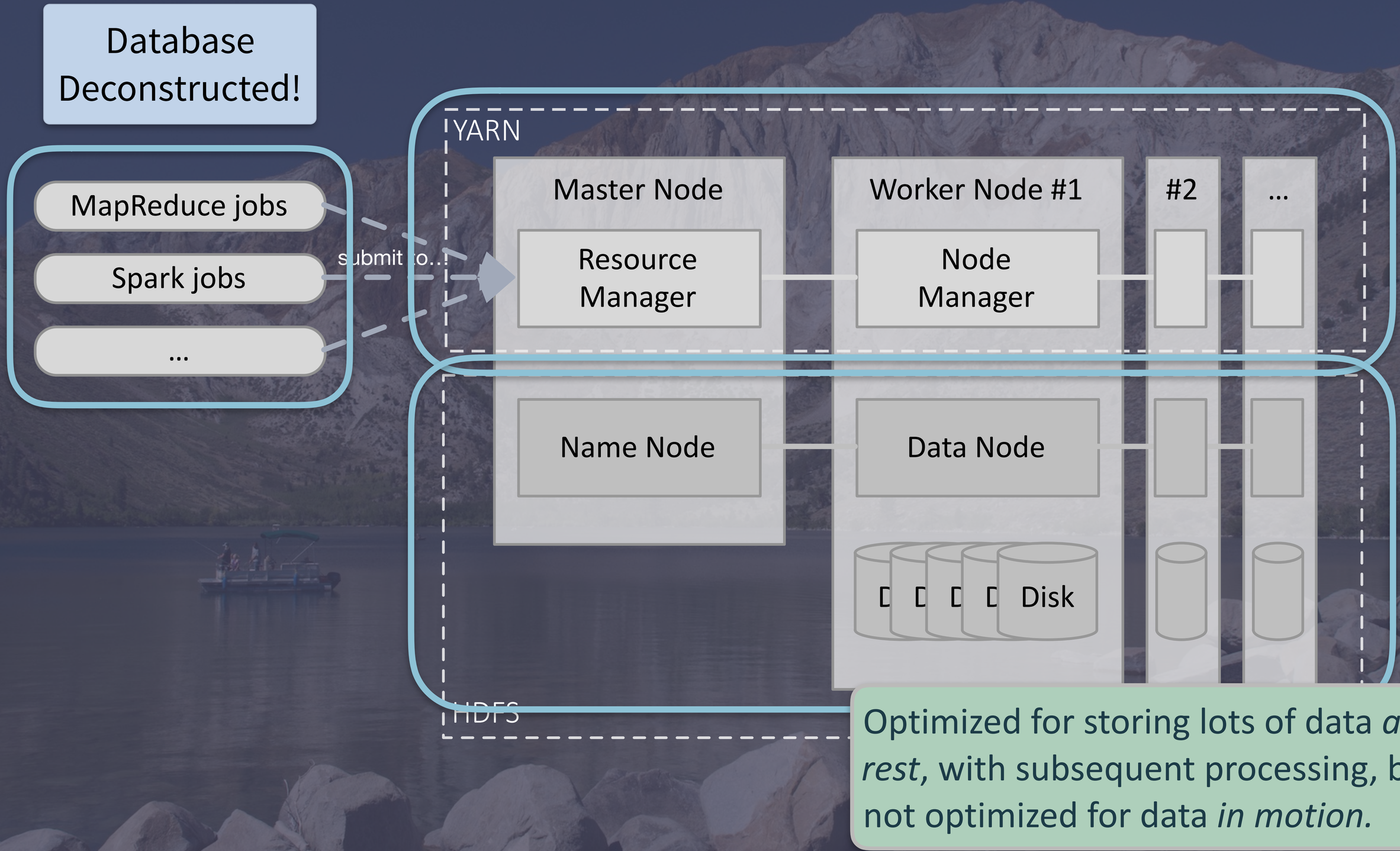

YARN
MapReduce jobs
Spark jobs
…
submit to…
Master Node
Resource Manager
Worker Node #1
Node Manager
#2
…
Name Node
Data Node
Disk
HDFS


YARN
Master Node
Worker Node #1
#2
…
MapReduce jobs
Spark jobs
…
submit to…
Resource Manager
Node Manager
Name Node
Data Node
Disk
HDFS
Storage


Compute
MapReduce jobs
Spark jobs
...
submit to...
YARN
Master Node
Resource Manager
Worker Node #1
Node Manager
#2
...
Name Node
Data Node
Disk
HDFS

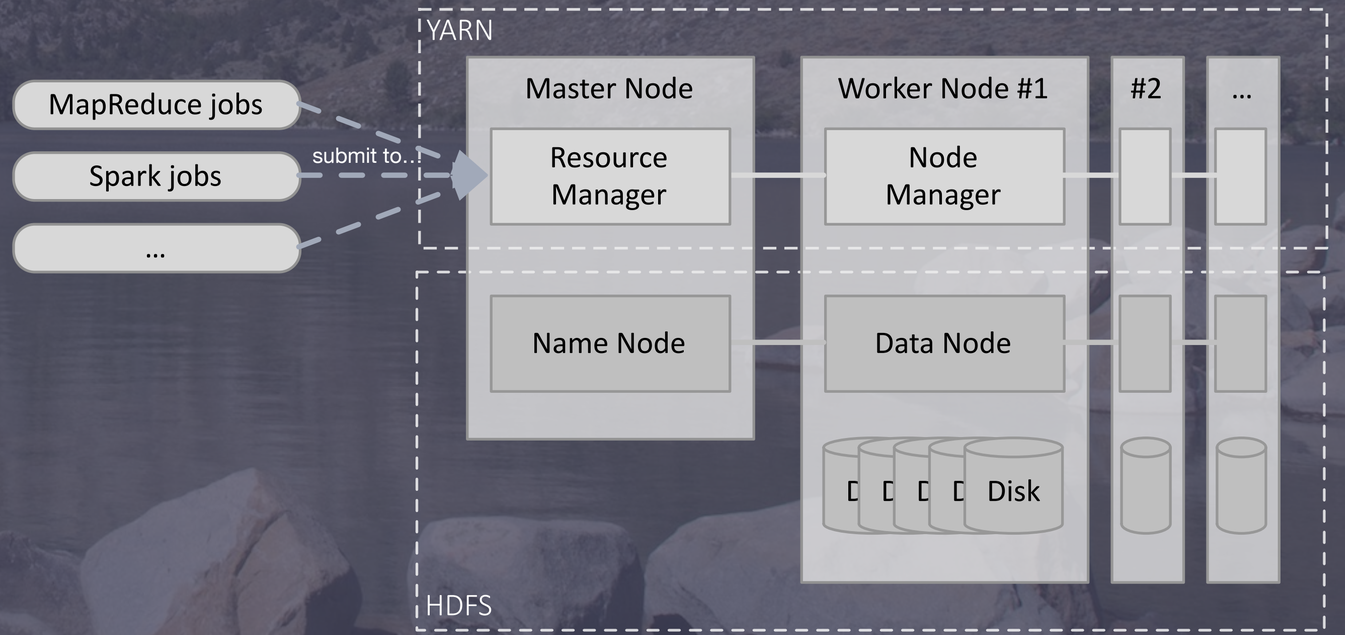

Resource Management
YARN
MapReduce jobs
Spark jobs
...
submit to...
Master Node
Resource Manager
Worker Node #1
Node Manager
#2
...
Name Node
Data Node
Disk
HDFS

Database Deconstructed!

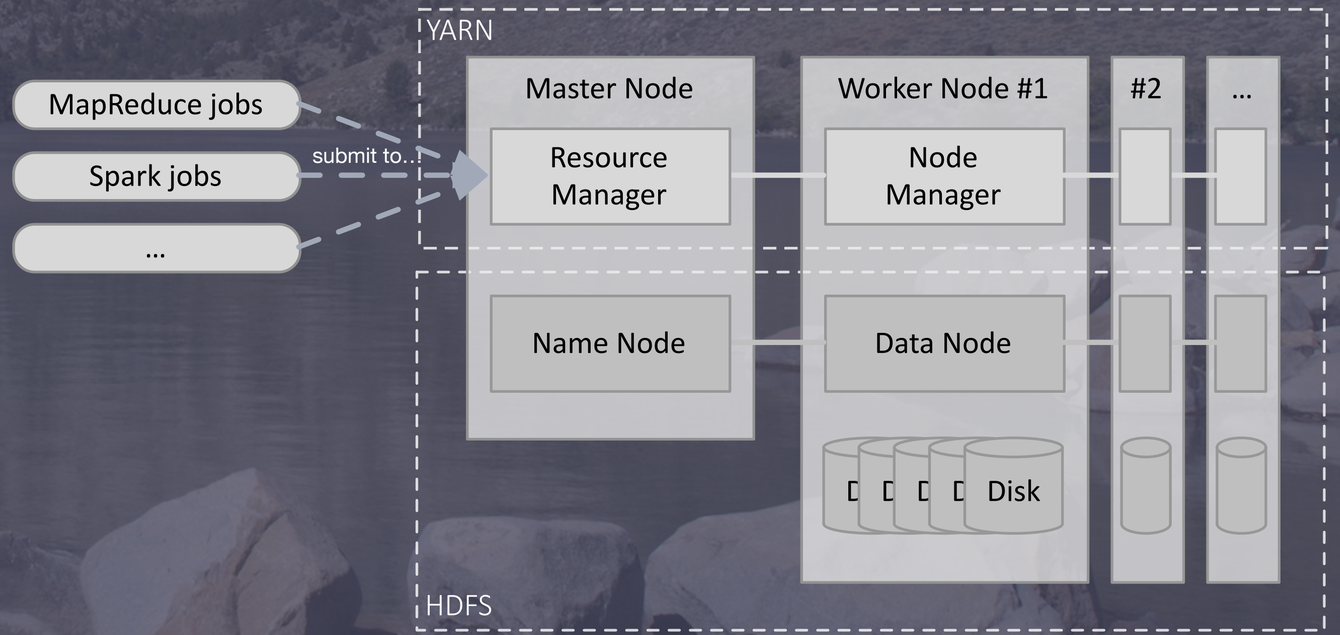


Optimized for storing lots of data *at rest*, with subsequent processing, but not optimized for data *in motion.*

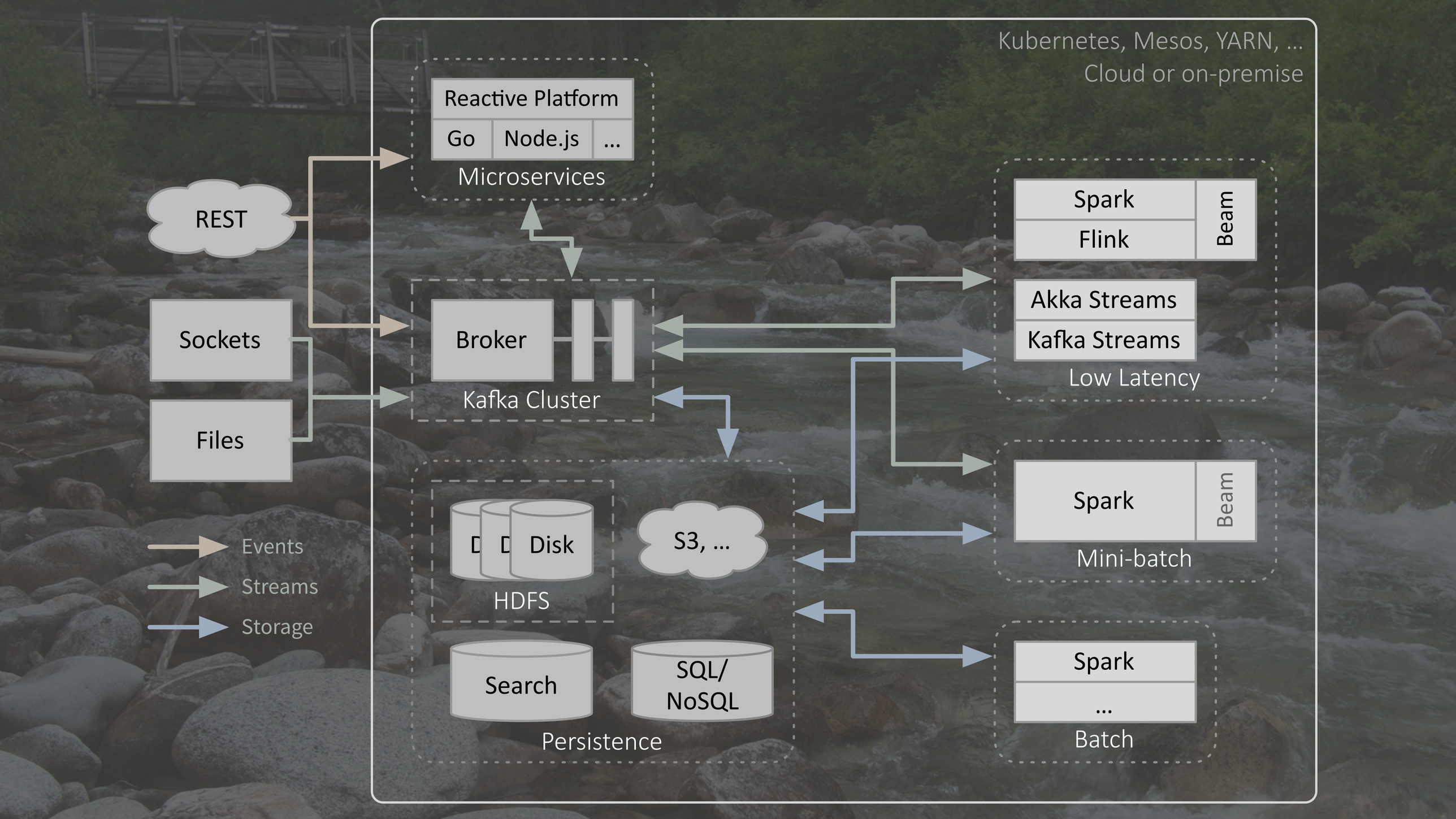
- Characteristics
  - Batch and interactive
  - Massive storage - HDFS is the data "backplane"
  - Multiuser jobs

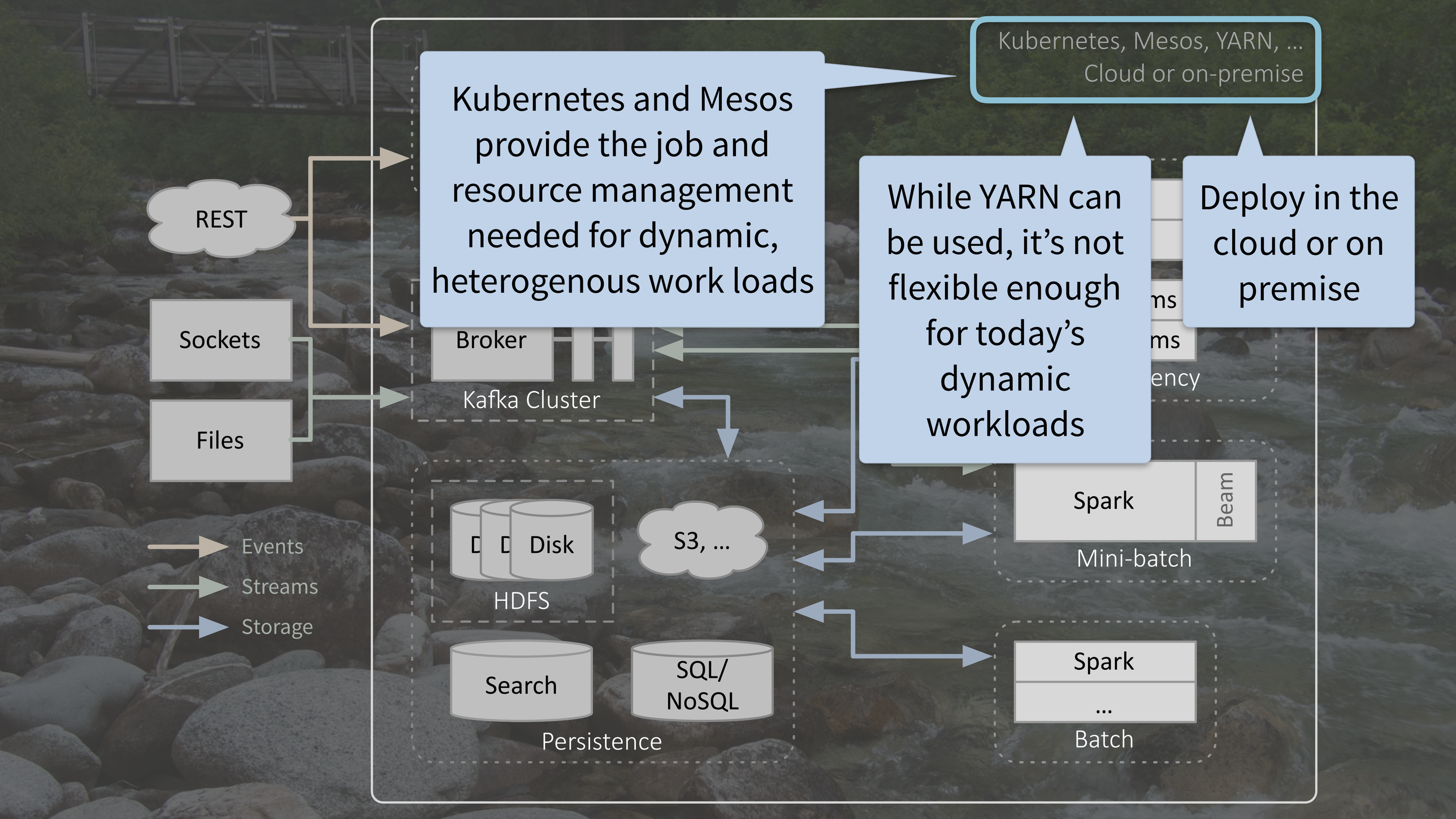


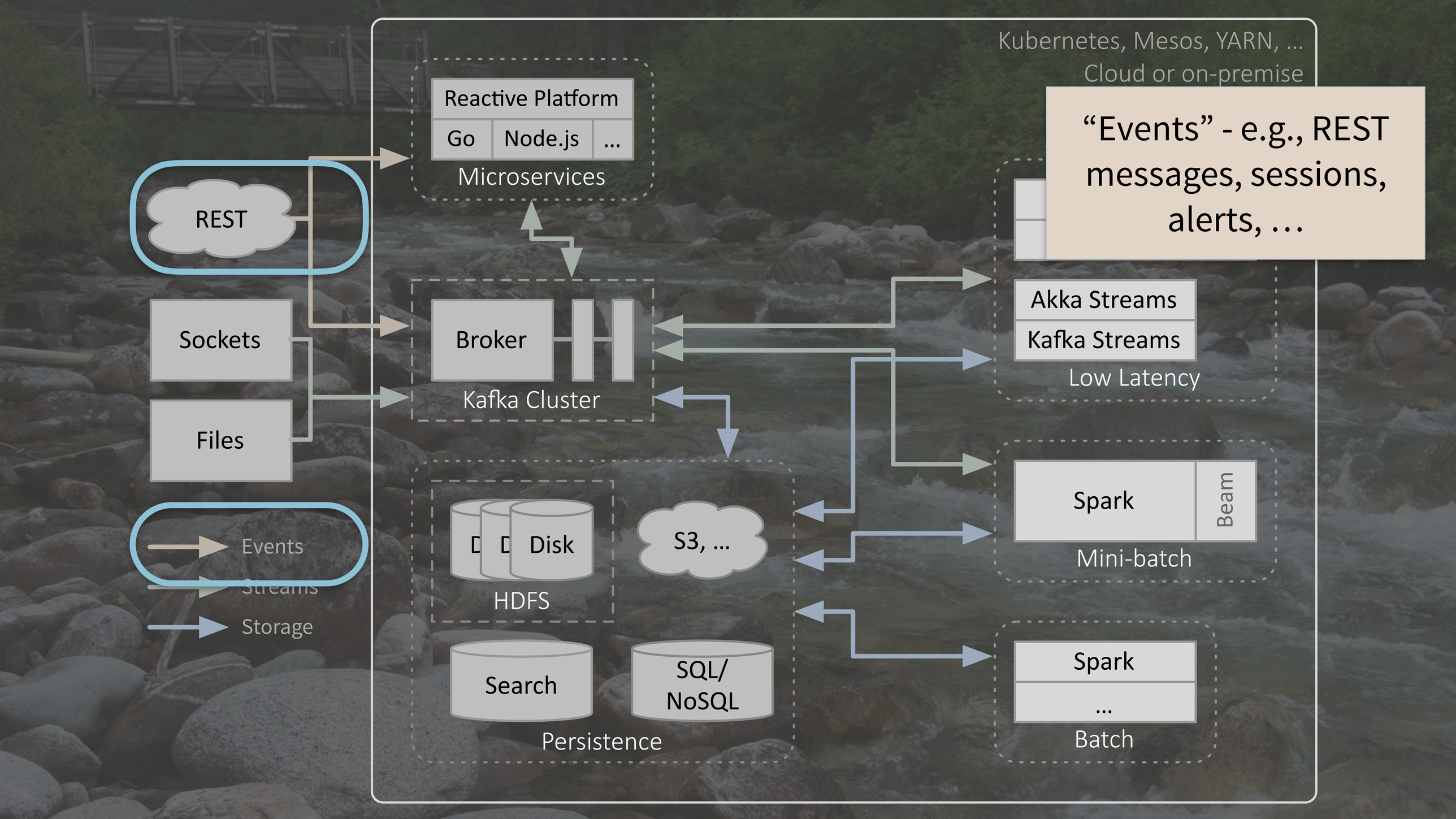
- Use Cases
  - Data warehouse replacement
  - Interactive exploration
  - Offline ML model training
  - …

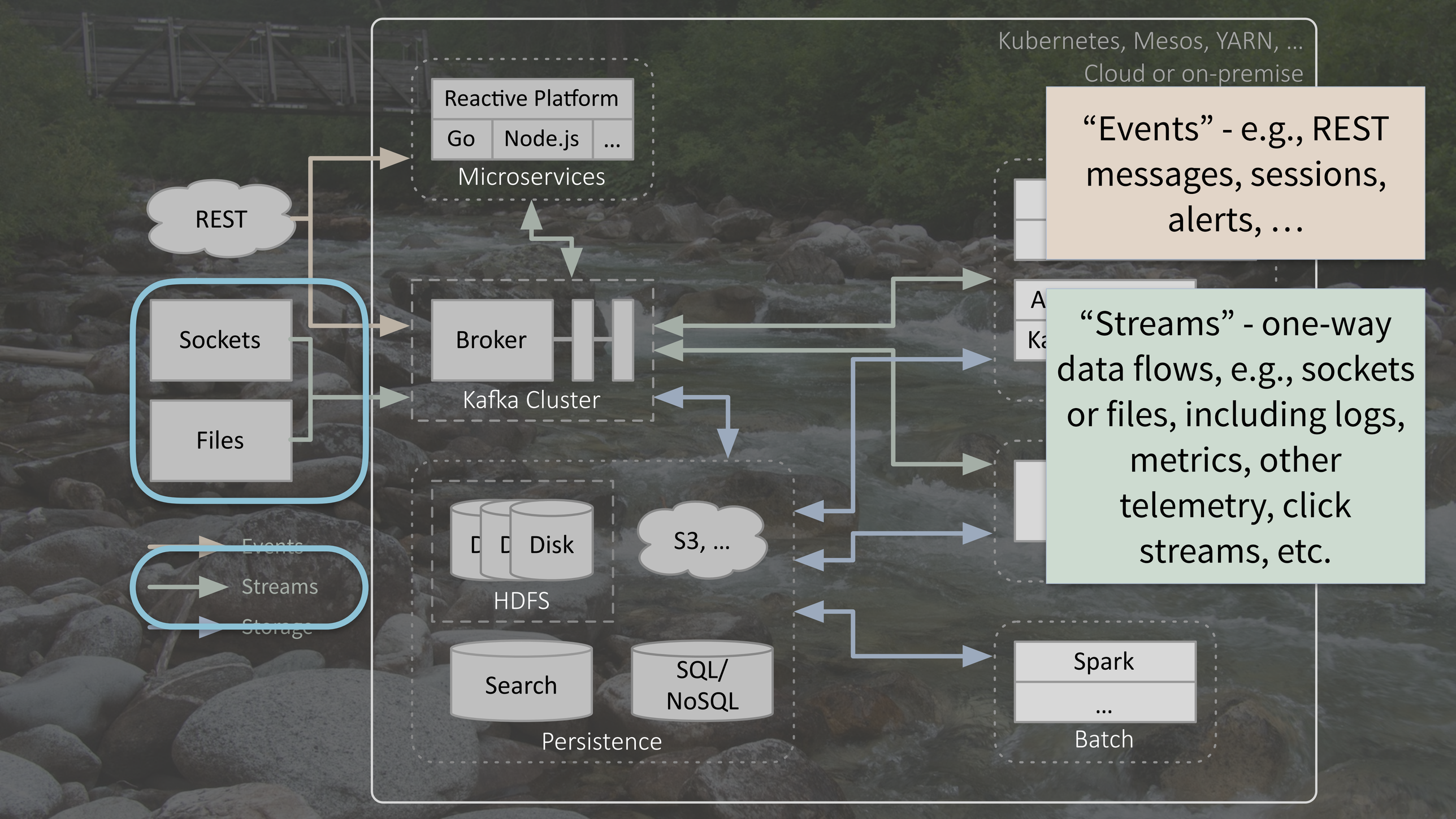


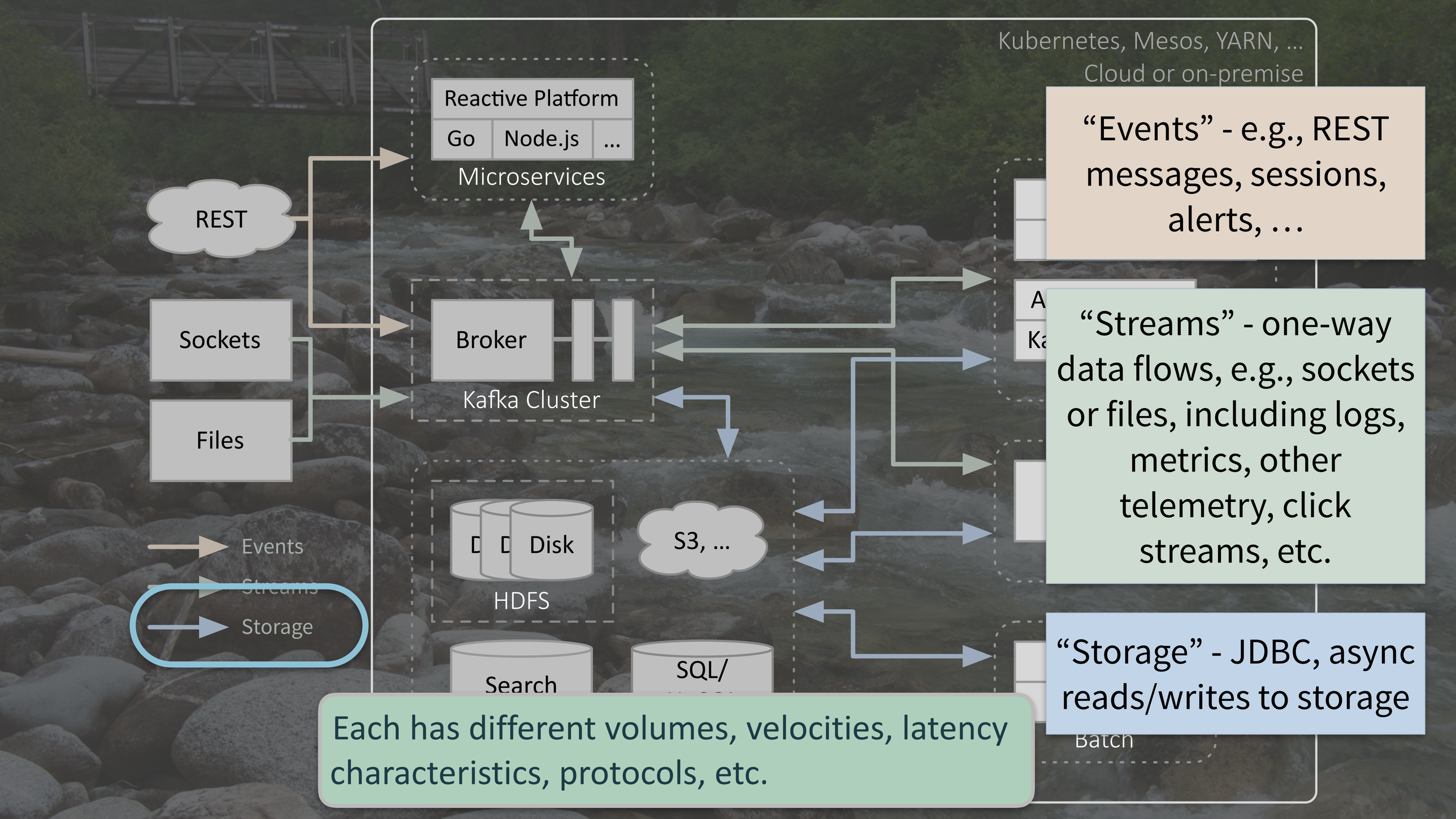
New Streaming, “Fast Data” Architecture

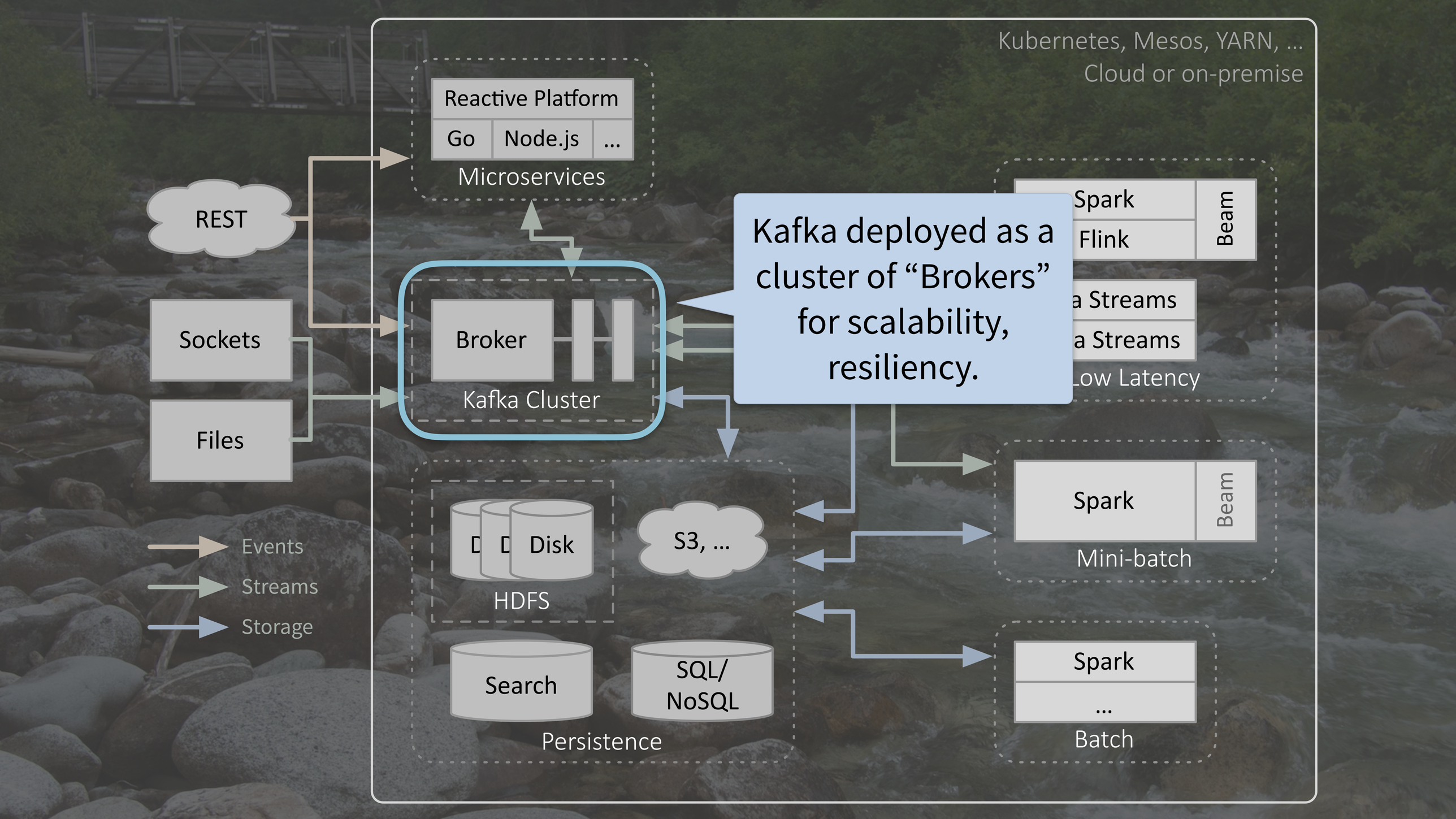

Kubernetes, Mesos, YARN, ...
Cloud or on-premise
Reactive Platform
Go
Node.js
...
Microservices
REST
Sockets
Files
Broker
Kafka Cluster
Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency
Spark
Beam
Mini-batch
Disk
HDFS
S3, ...
Search
SQL/
NoSQL
Persistence
Spark
...
Batch
Events
Streams
Storage

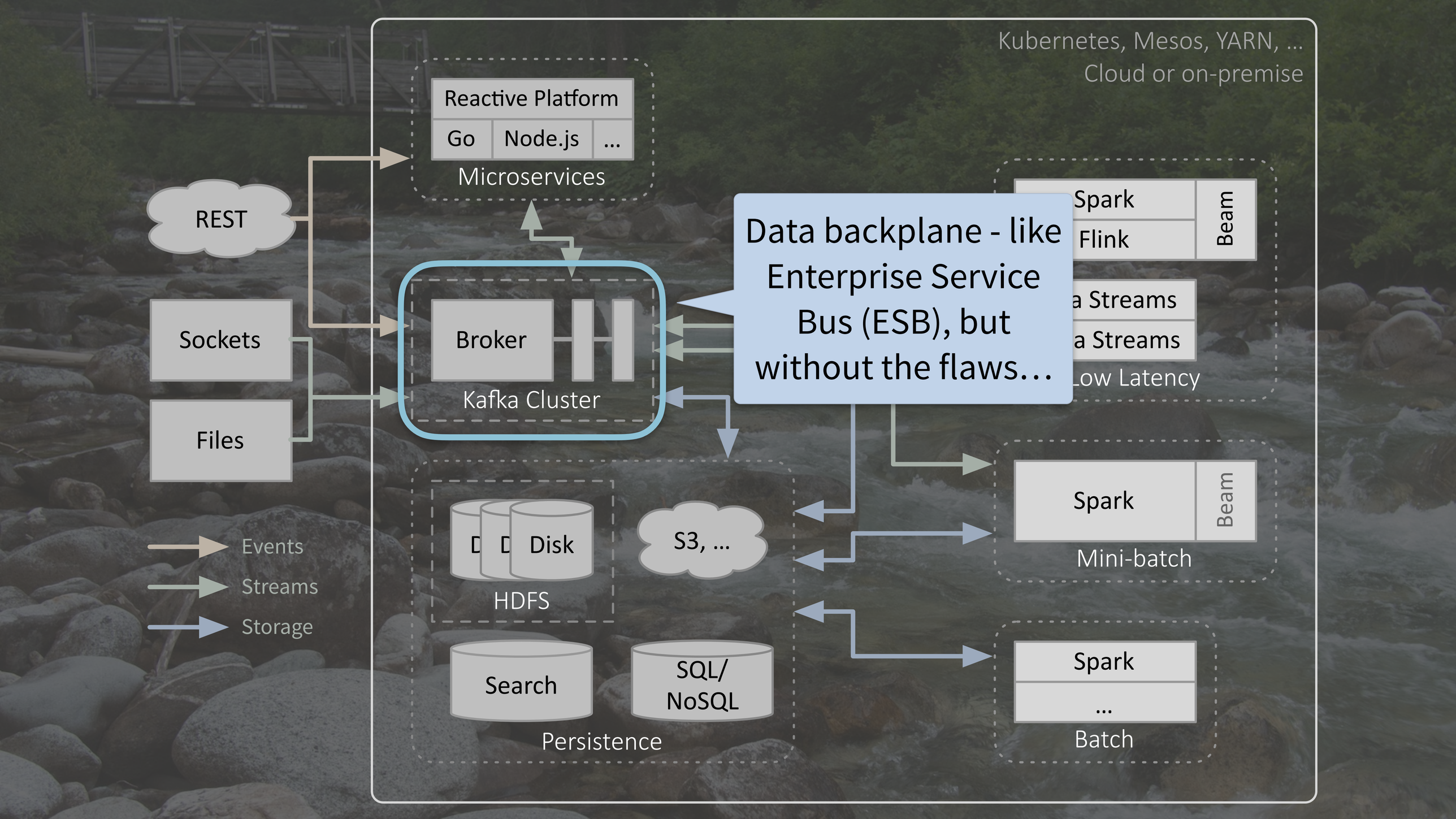

Kubernetes, Mesos, YARN, …
Cloud or on-premise
Kubernetes and Mesos provide the job and resource management needed for dynamic, heterogenous work loads
While YARN can be used, it's not flexible enough for today's dynamic workloads
Deploy in the cloud or on premise
REST
Sockets
Files
Broker
Kafka Cluster
Disk
HDFS
S3, …
Search
SQL/
NoSQL
Persistence
Spark
Beam
Mini-batch
Spark
…
Batch
Events
Streams
Storage


Kubernetes, Mesos, YARN, …
Cloud or on-premise
Reactive Platform
Go
Node.js
…
Microservices
REST
Sockets
Files
Broker
Kafka Cluster
Akka Streams
Kafka Streams
Low Latency
Spark
Beam
Mini-batch
Spark
…
Batch
Disk
HDFS
S3, …
Search
SQL/
NoSQL
Persistence
Events
Streams
Storage
“Events” - e.g., REST messages, sessions, alerts, …


Kubernetes, Mesos, YARN, …
Cloud or on-premise
Reactive Platform
Go
Node.js
…
Microservices
REST
Sockets
Files
Broker
Kafka Cluster
Disk
HDFS
S3, …
Search
SQL/
NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage
“Events” - e.g., REST messages, sessions, alerts, …
“Streams” - one-way data flows, e.g., sockets or files, including logs, metrics, other telemetry, click streams, etc.


Kubernetes, Mesos, YARN, …
Cloud or on-premise
Reactive Platform
Go
Node.js
…
Microservices
REST
Sockets
Files
Broker
Kafka Cluster
Disk
HDFS
S3, …
Search
SQL/
Batch
Events
Streams
Storage
“Events” - e.g., REST messages, sessions, alerts, …
“Streams” - one-way data flows, e.g., sockets or files, including logs, metrics, other telemetry, click streams, etc.
“Storage” - JDBC, async reads/writes to storage
Each has different volumes, velocities, latency characteristics, protocols, etc.

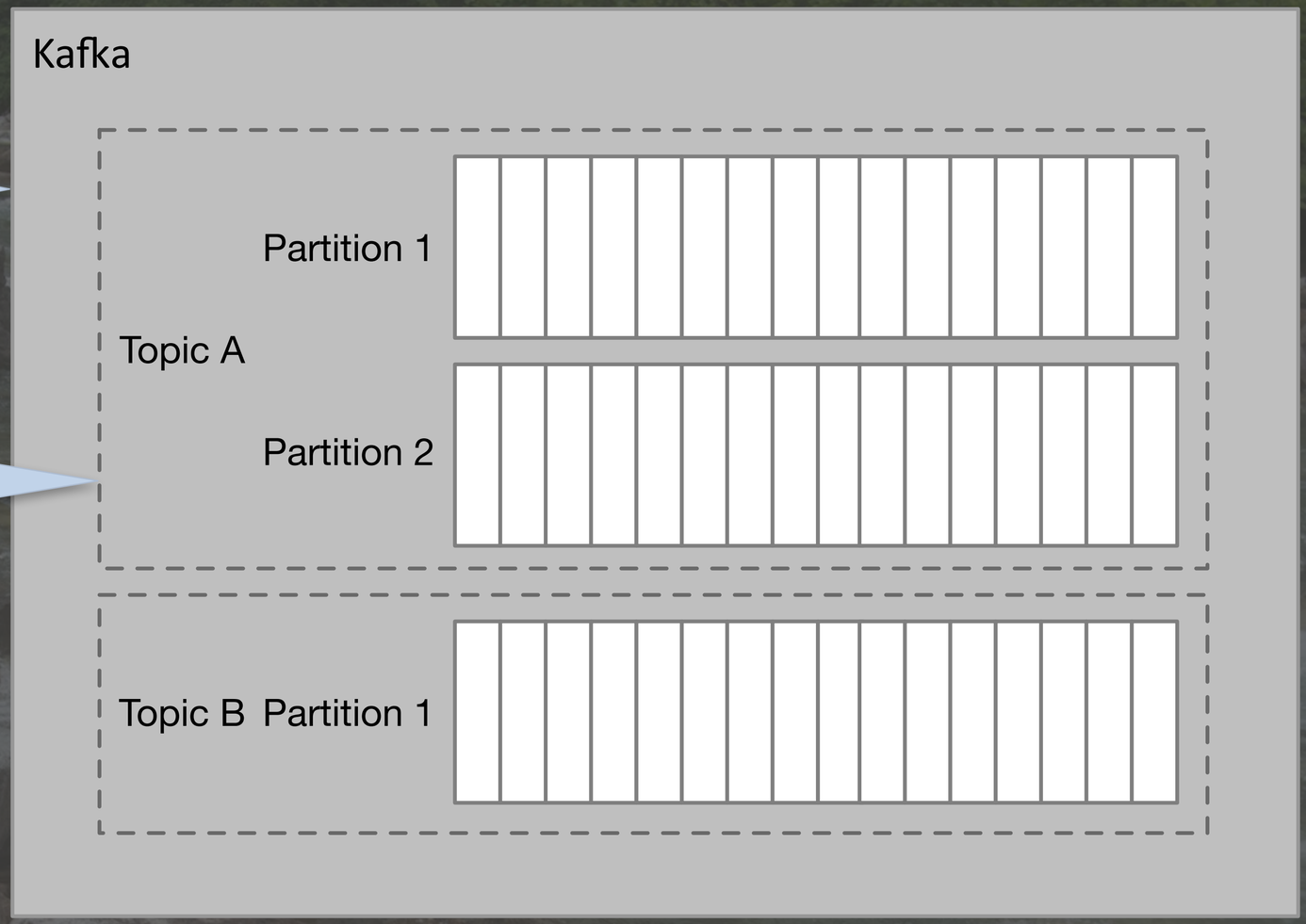

Kubernetes, Mesos, YARN, …
Cloud or on-premise
Reactive Platform
Go
Node.js
…
Microservices
REST
Sockets
Files
Broker
Kafka Cluster
Kafka deployed as a cluster of “Brokers” for scalability, resiliency.
Spark
Flink
Beam
a Streams
a Streams
Low Latency
Spark
Beam
Mini-batch
Disk
HDFS
S3, …
Search
SQL/
NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage


Kubernetes, Mesos, YARN, …
Cloud or on-premise
Reactive Platform
Go
Node.js
…
Microservices
REST
Sockets
Files
Broker
Kafka Cluster
Data backplane - like Enterprise Service Bus (ESB), but without the flaws…
Spark
Flink
Beam
a Streams
a Streams
Low Latency
Spark
Beam
Mini-batch
Disk
HDFS
S3, …
Search
SQL/
NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage

Why Kafka?

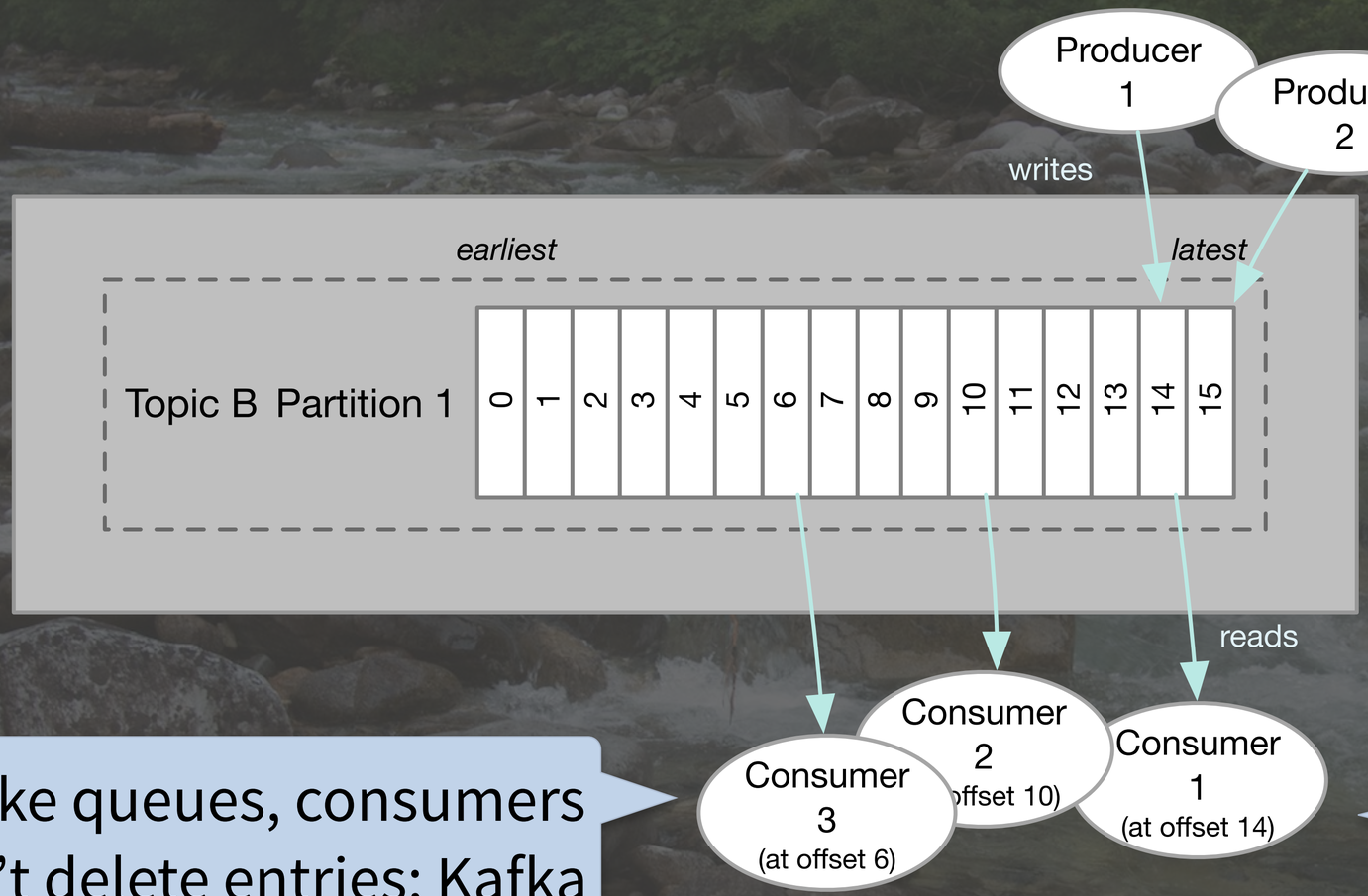
Organized into topics

Topics are partitioned, replicated, and distributed



Why Kafka?

Logs, *not* queues!

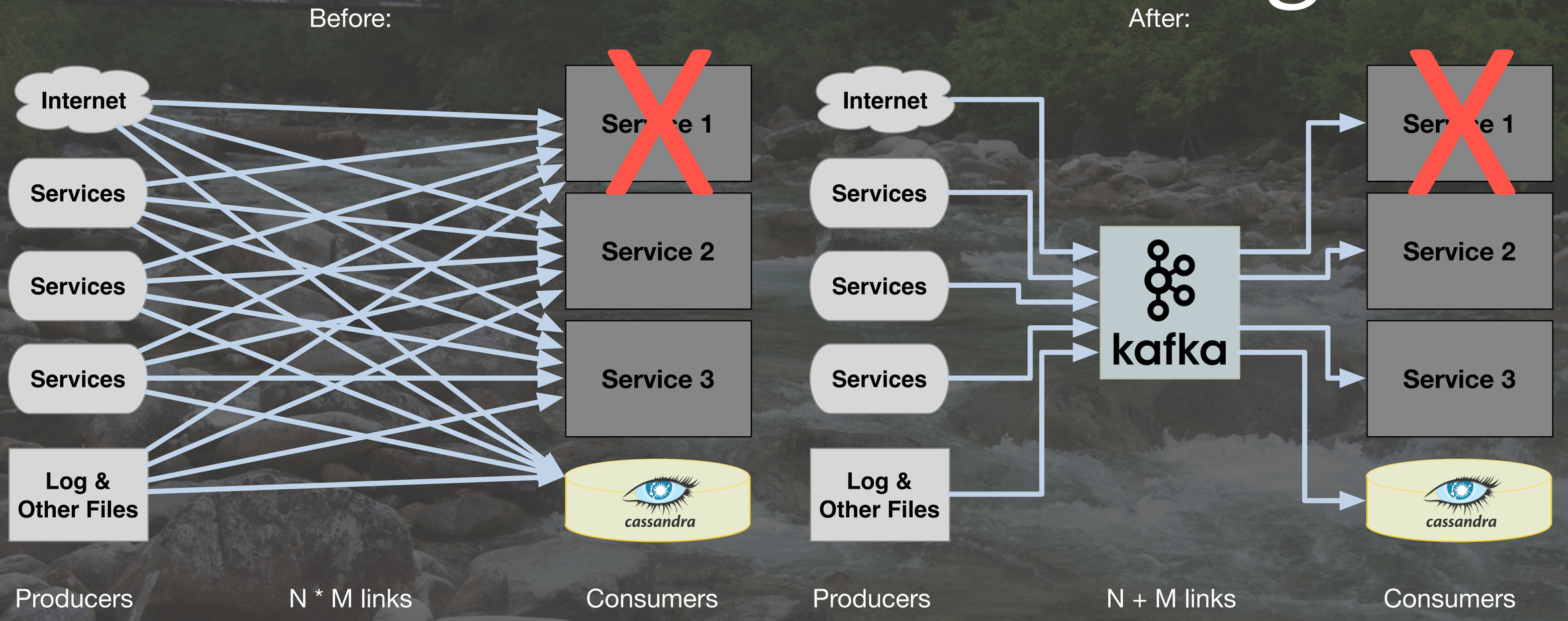


M Producers

N Consumers, who start reading where they want

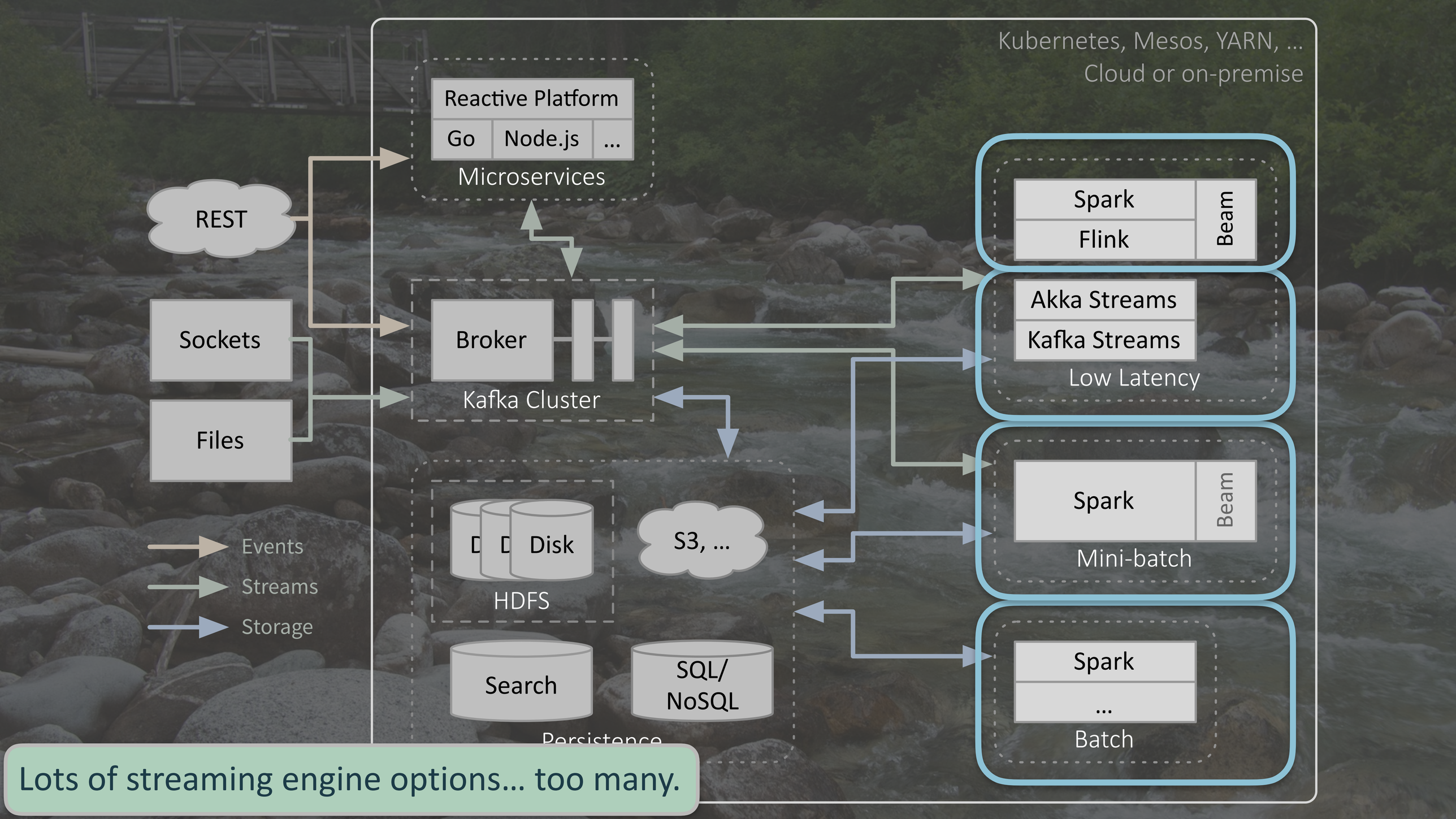
Unlike queues, consumers don't delete entries; Kafka manages their lifecycles

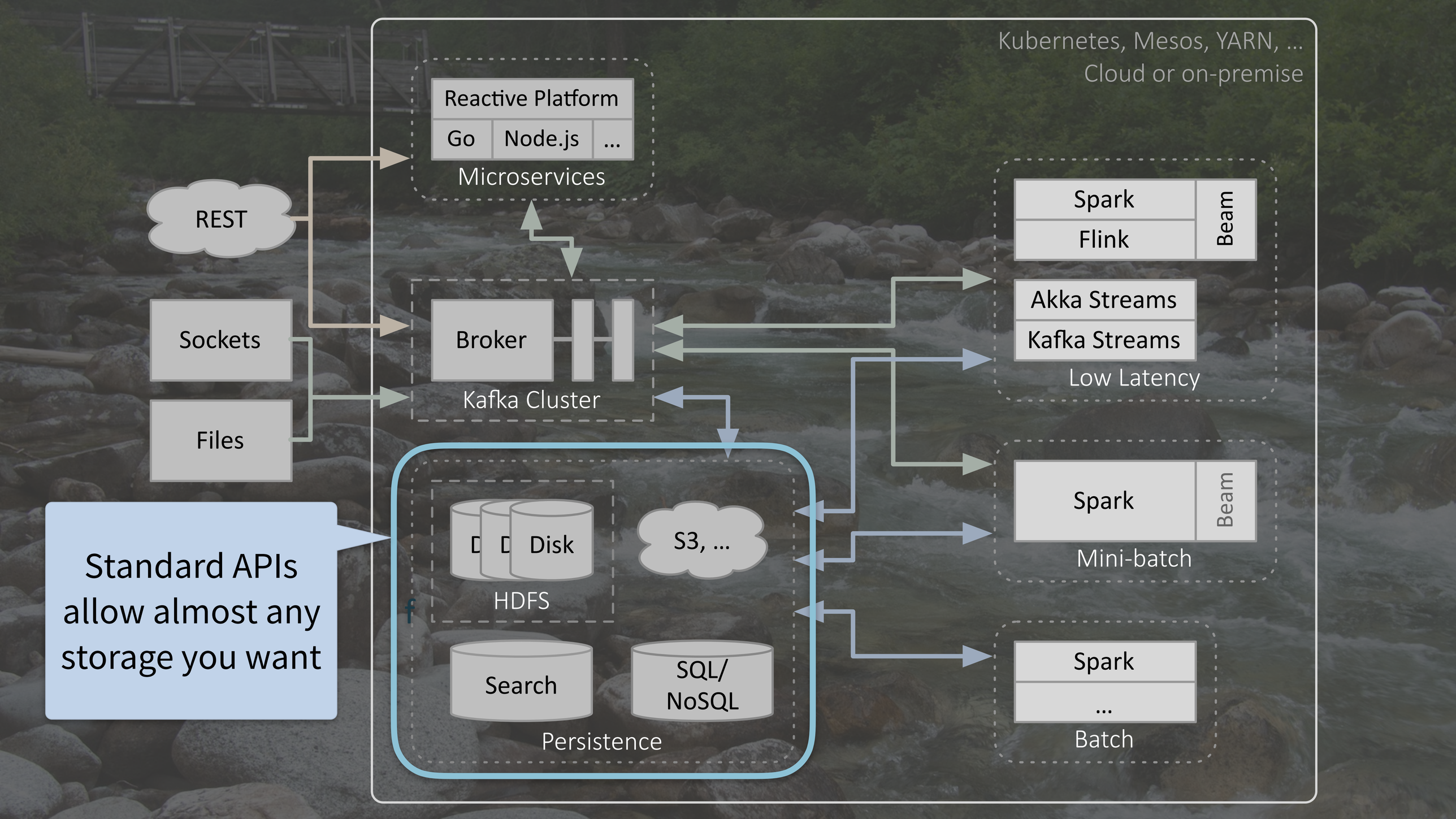
Using Kafka

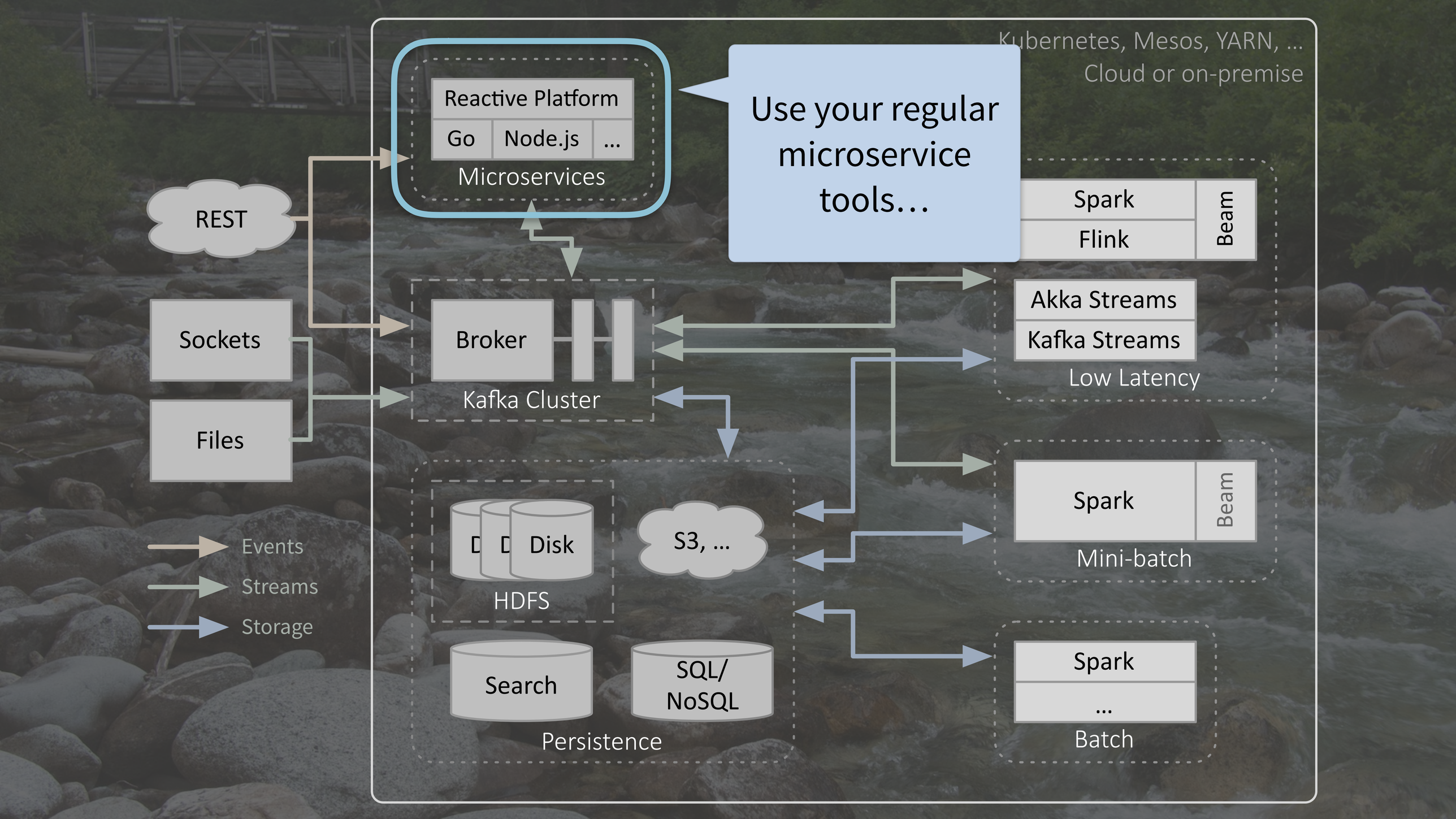


Messy and fragile; what if “Service 1” goes down?

Simpler and more robust! Loss of Service 1 means no data loss.

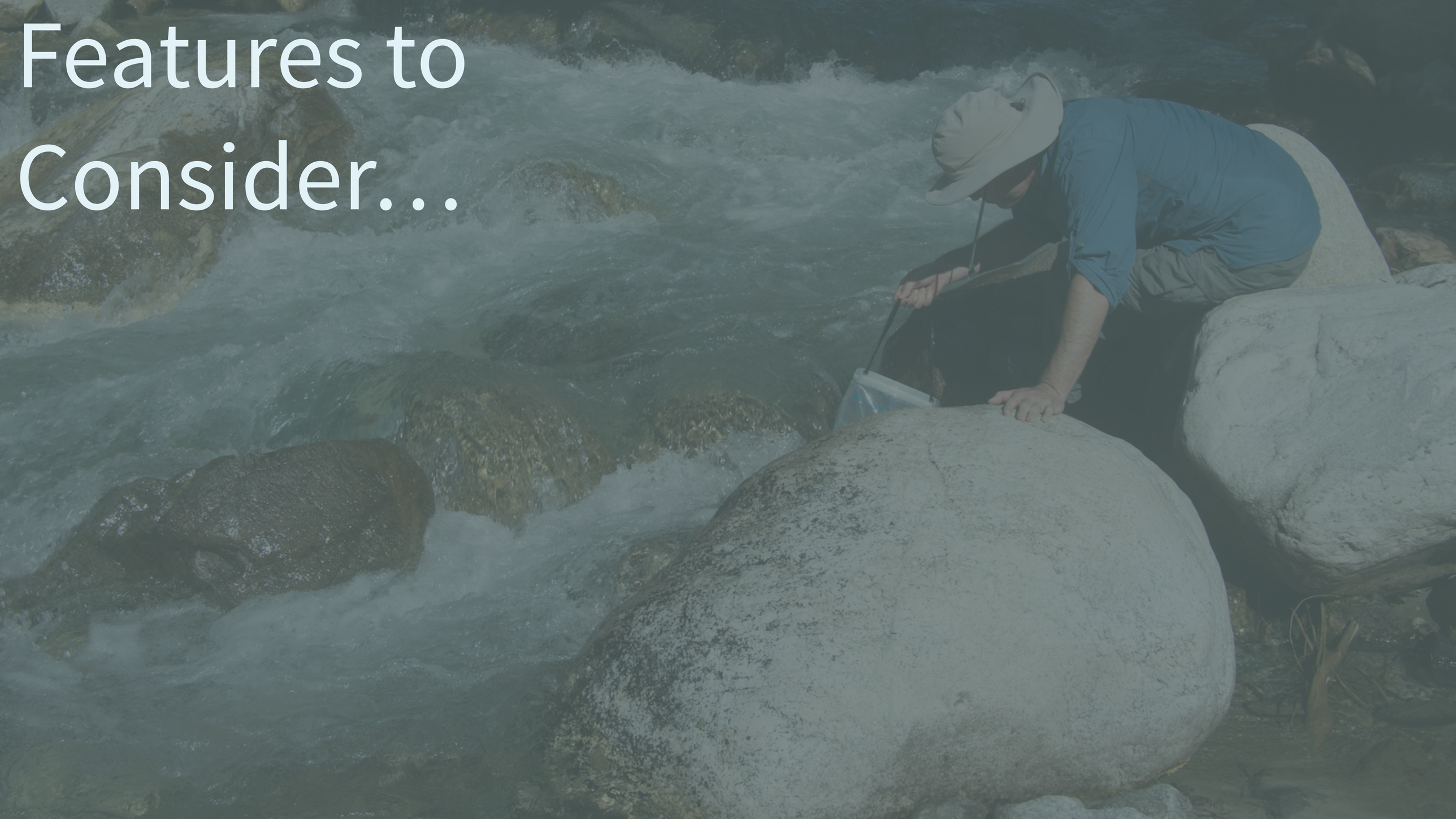

Kubernetes, Mesos, YARN, …
Cloud or on-premise
Reactive Platform
Go
Node.js
…
Microservices
REST
Sockets
Files
Broker
Kafka Cluster
Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency
Spark
Beam
Mini-batch
Spark
…
Batch
Disk
HDFS
S3, …
Search
SQL/
NoSQL
Persistence
Events
Streams
Storage
Lots of streaming engine options… too many.

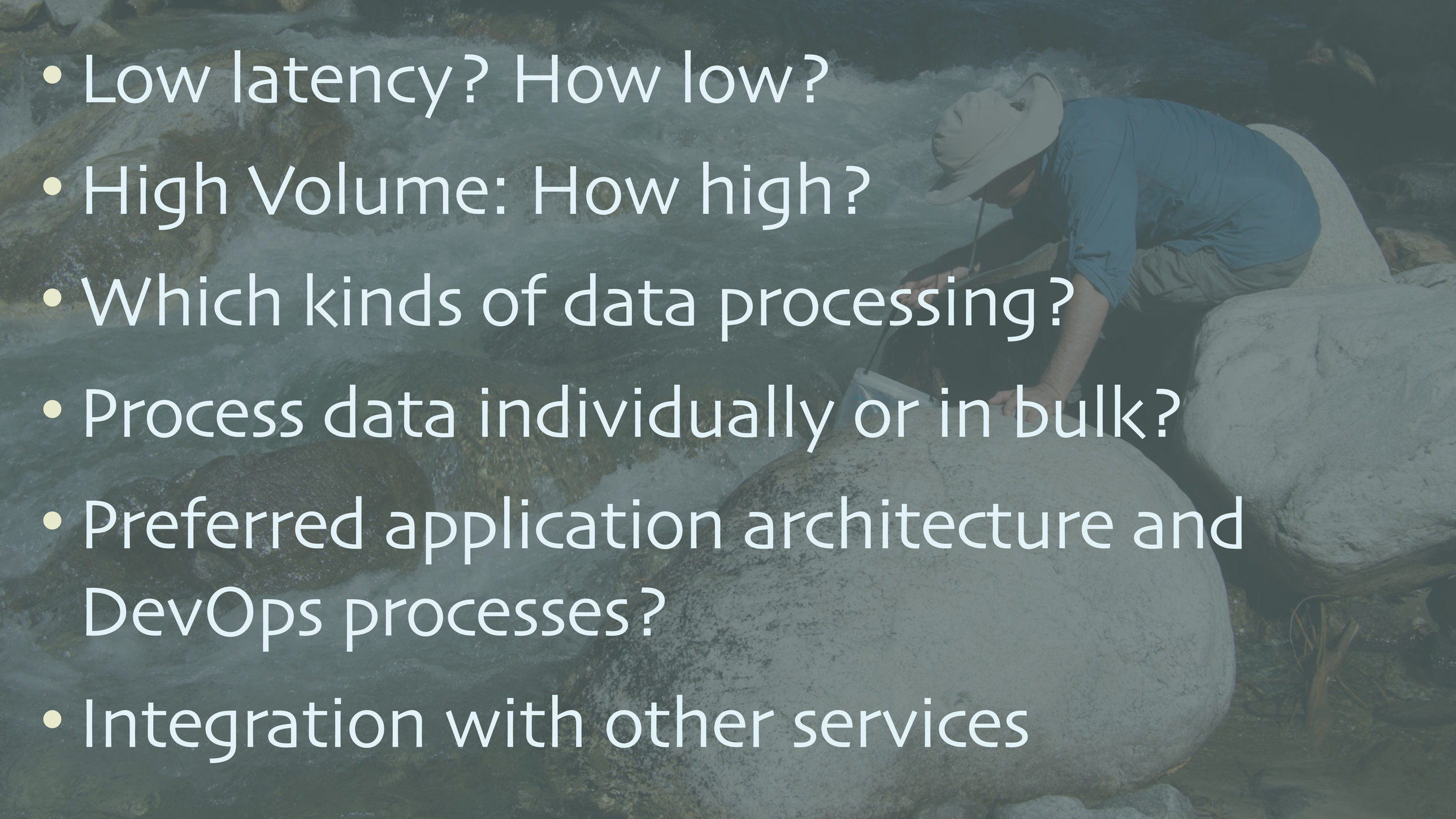

Kubernetes, Mesos, YARN, ...
Cloud or on-premise
Reactive Platform
Go
Node.js
...
Microservices
REST
Sockets
Files
Broker
Kafka Cluster
Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency
Spark
Beam
Mini-batch
Spark
...
Batch
Disk
HDFS
S3, ...
Search
SQL/
NoSQL
Persistence
Standard APIs
allow almost any
storage you want


Kubernetes, Mesos, YARN, …
Cloud or on-premise
Reactive Platform
Go
Node.js
…
Microservices
Use your regular microservice tools…
REST
Sockets
Files
Broker
Kafka Cluster
Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency
Spark
Beam
Mini-batch
Disk
HDFS
S3, …
Search
SQL/
NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage

Streaming Engines

Features to Consider…





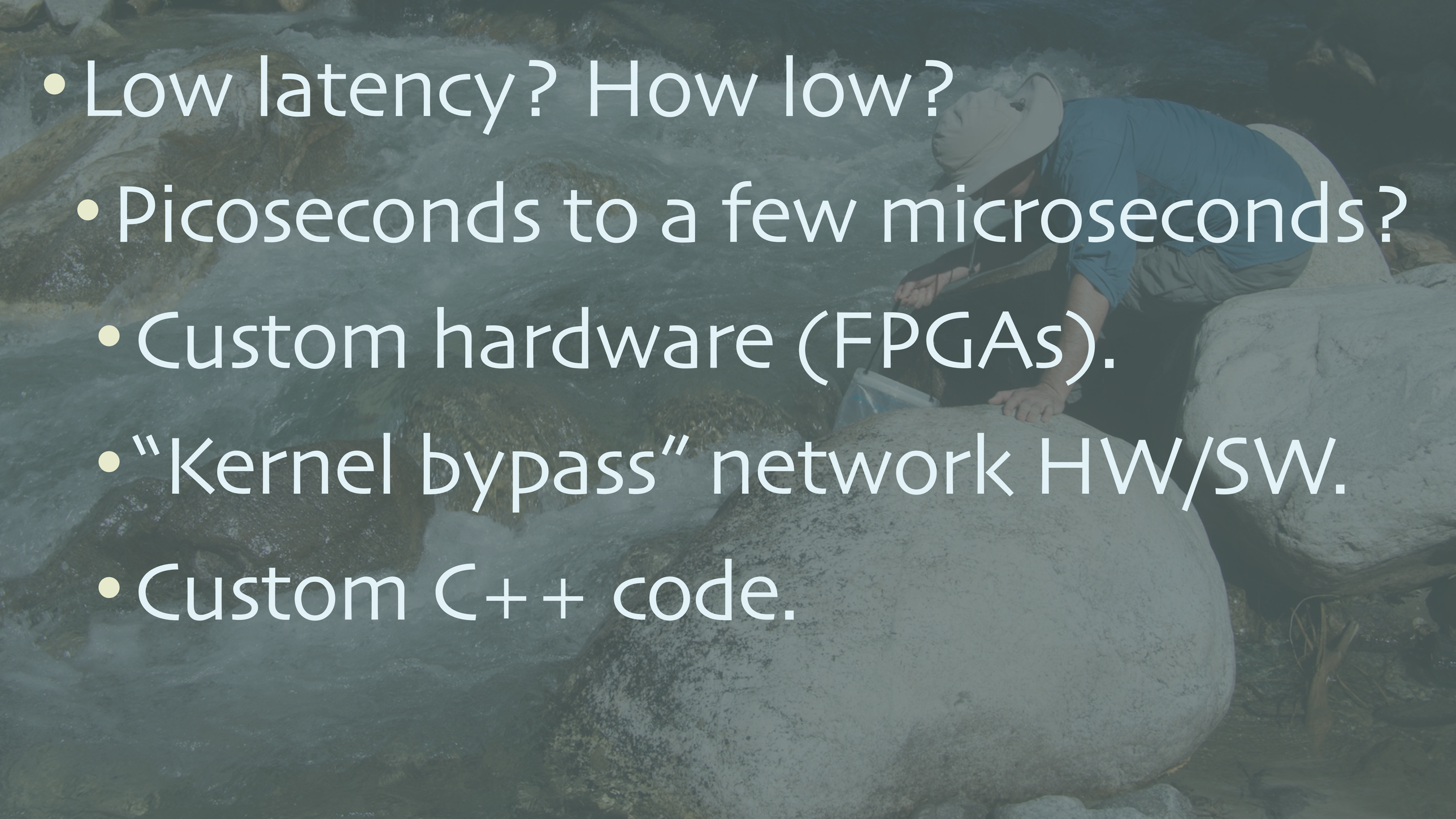
- Low latency? How low?
- High Volume: How high?
- Which kinds of data processing?
- Process data individually or in bulk?
- Preferred application architecture and DevOps processes?
- Integration with other services

- Low latency? How low?

- Low latency? How low?
  - Picoseconds to a few microseconds?

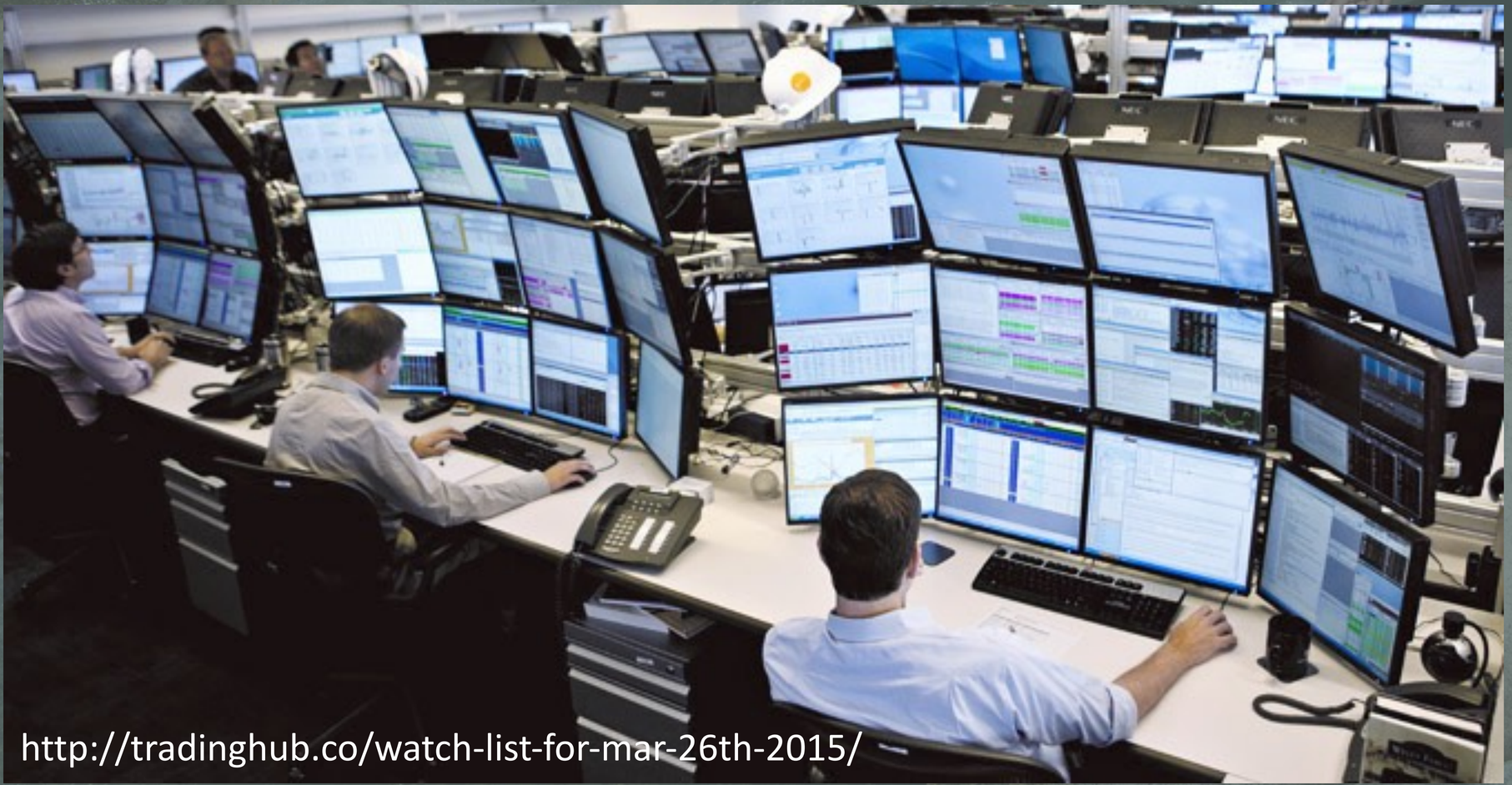
True “Real Time”

http://www.spacex.com/news



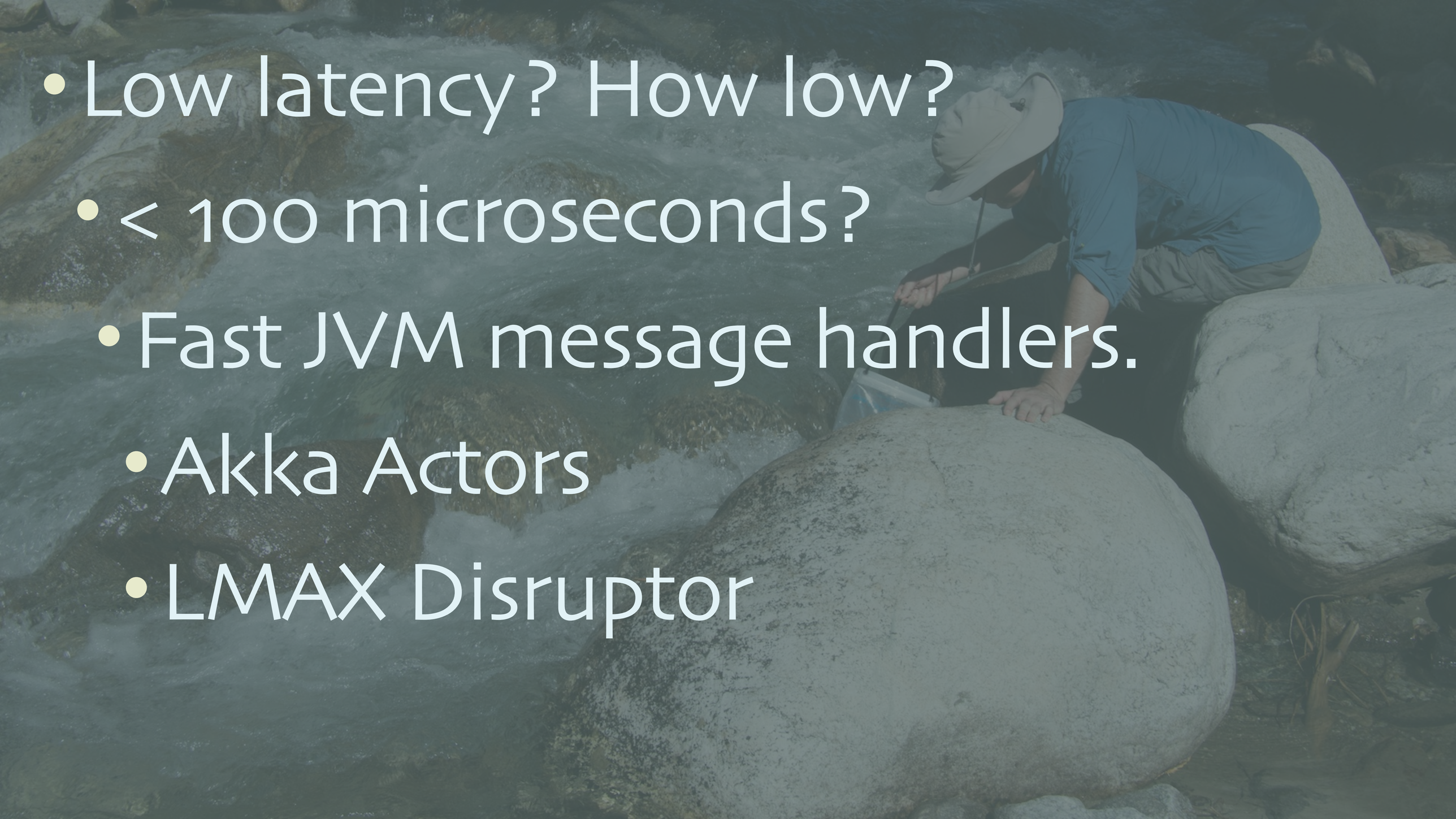
- Low latency? How low?
  - Picoseconds to a few microseconds?
    - Custom hardware (FPGAs).
    - “Kernel bypass” network HW/SW.
    - Custom C++ code.

- Low latency? How low?
  - < 100 microseconds?



http://tradinghub.co/watch-list-for-mar-26th-2015/

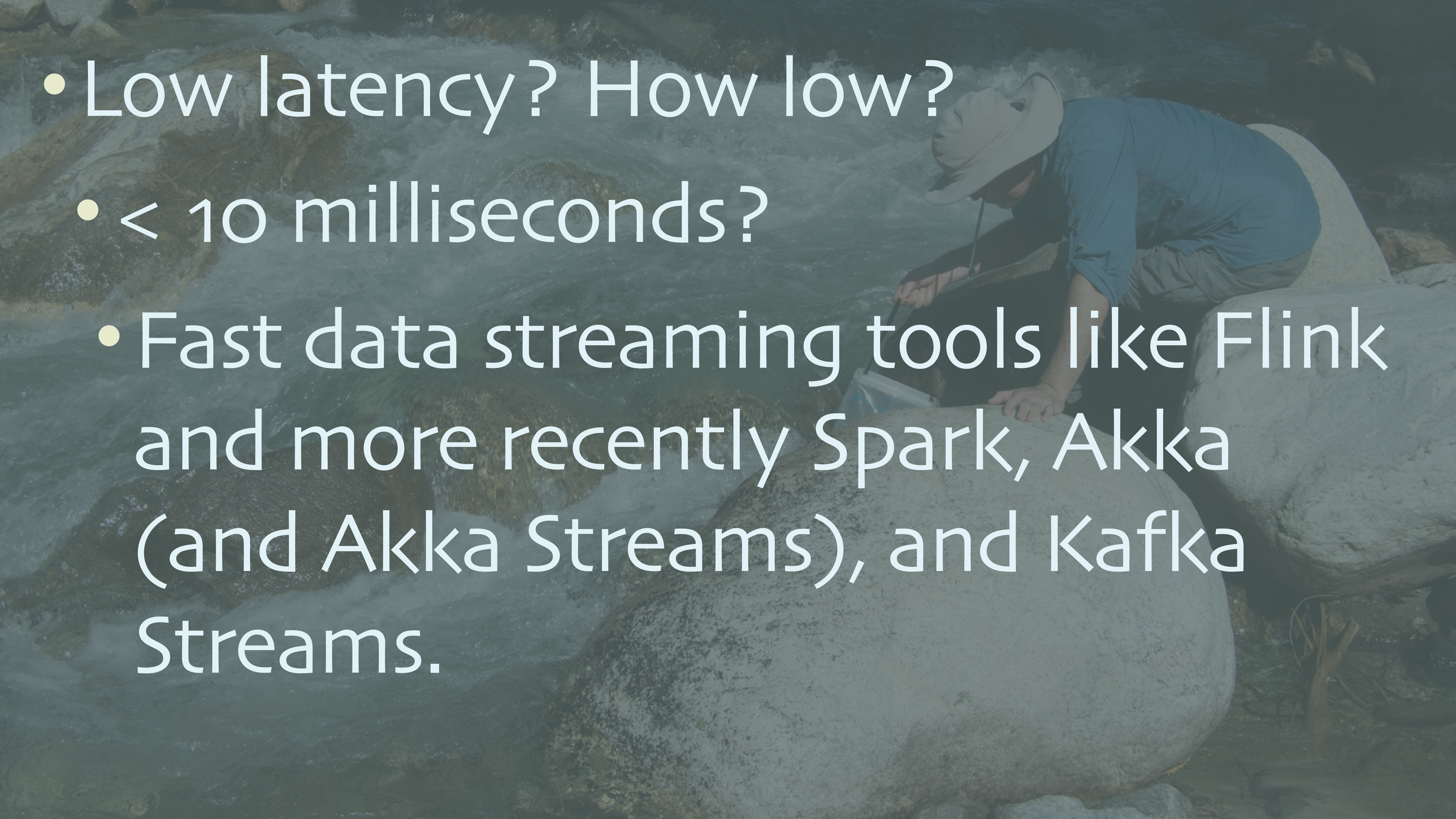
http://www.usa.philips.com/



- Low latency? How low?
  - < 100 microseconds?
    - Fast JVM message handlers.
      - Akka Actors
      - LMAX Disruptor

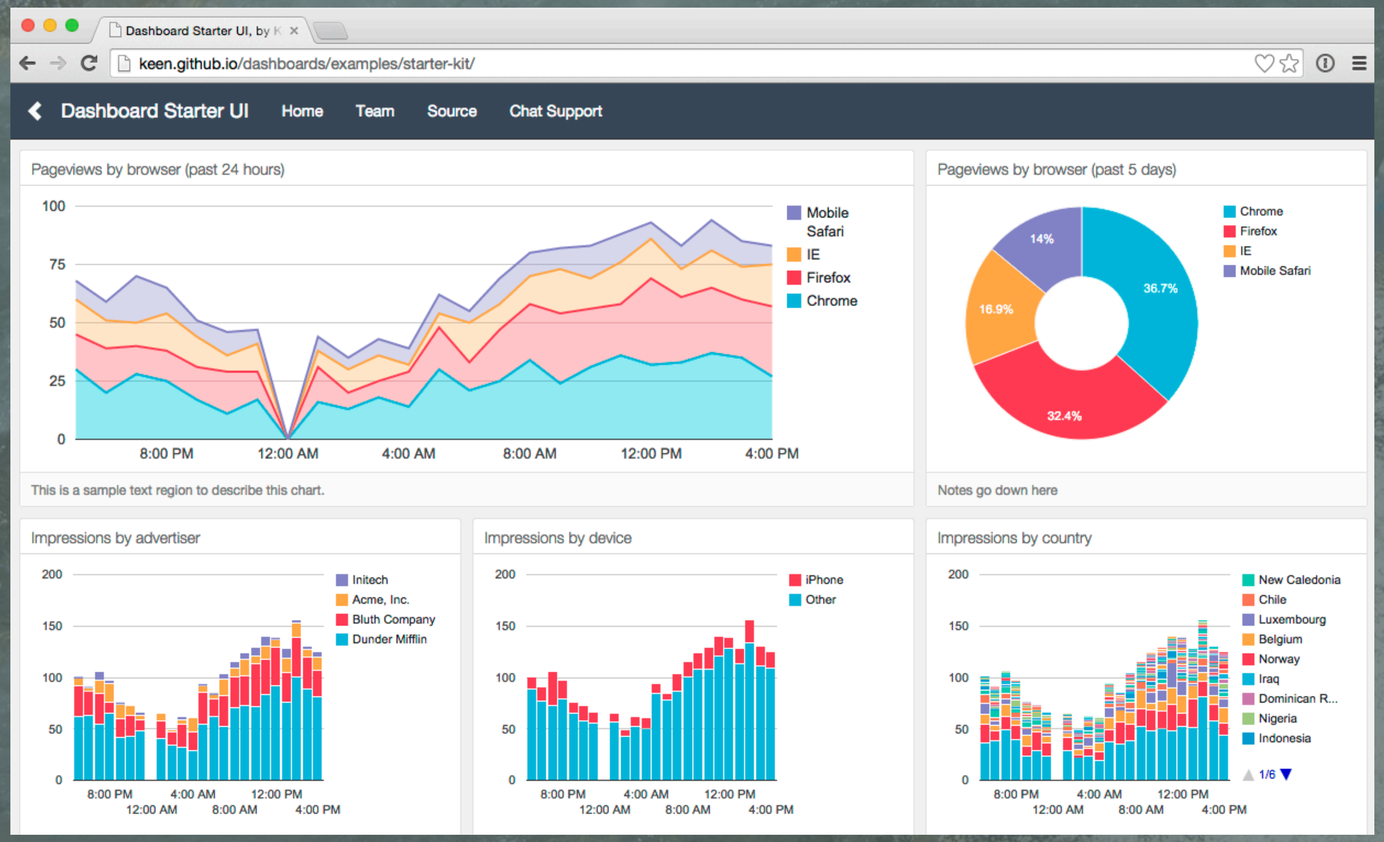
- Low latency? How low?
  - < 10 milliseconds?

http://money.cnn.com/2017/05/12/pf/credit-card-mistakes/index.html



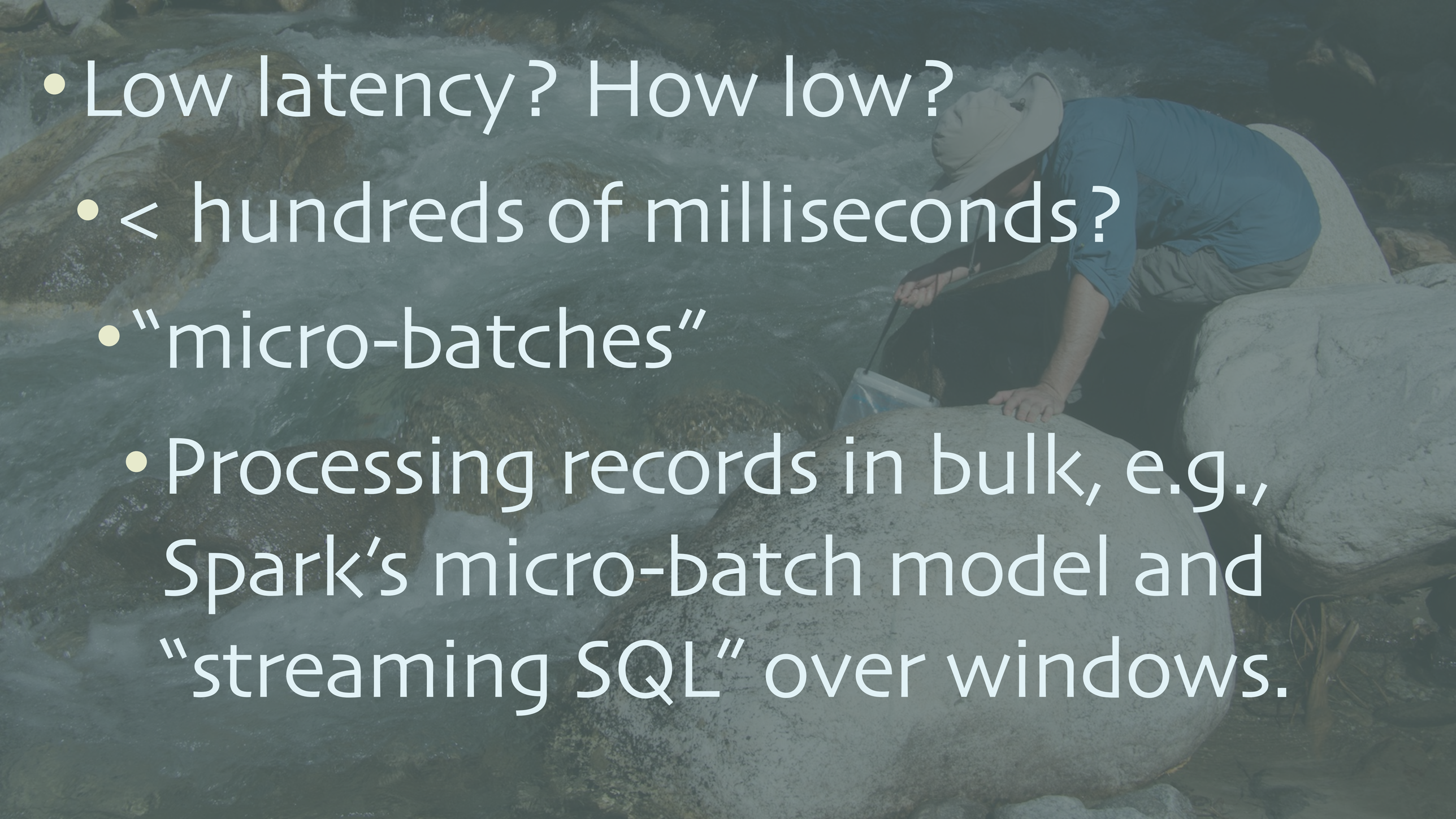
- Low latency? How low?
  - < 10 milliseconds?
    - Fast data streaming tools like Flink and more recently Spark, Akka (and Akka Streams), and Kafka Streams.

- Low latency? How low?
- < hundreds of milliseconds?



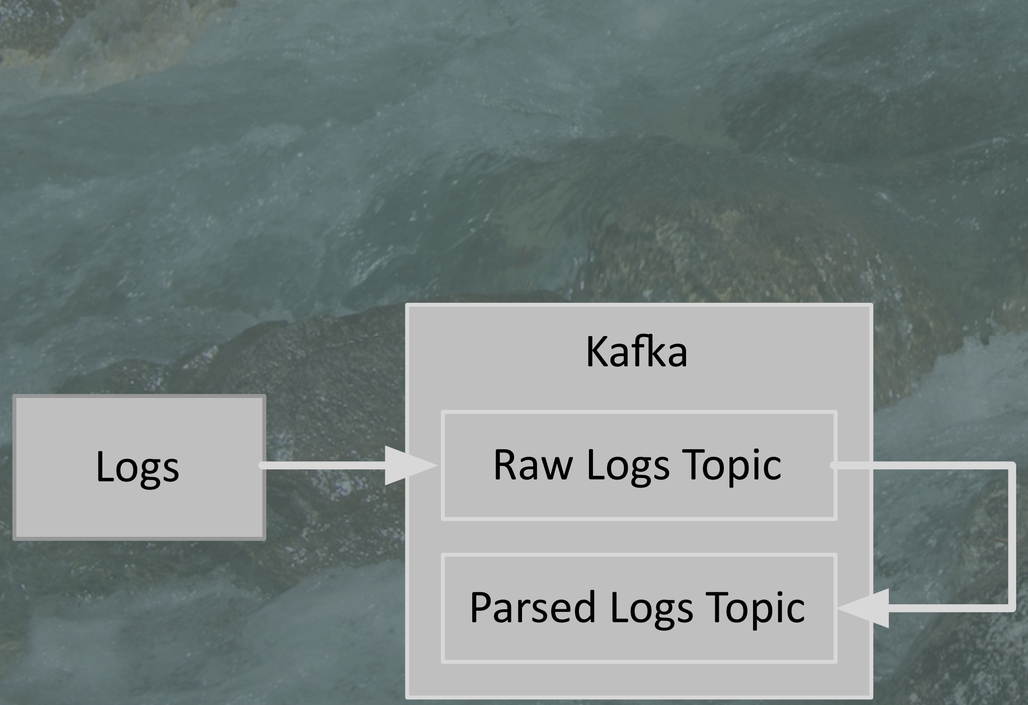
https://github.com/keen/dashboards

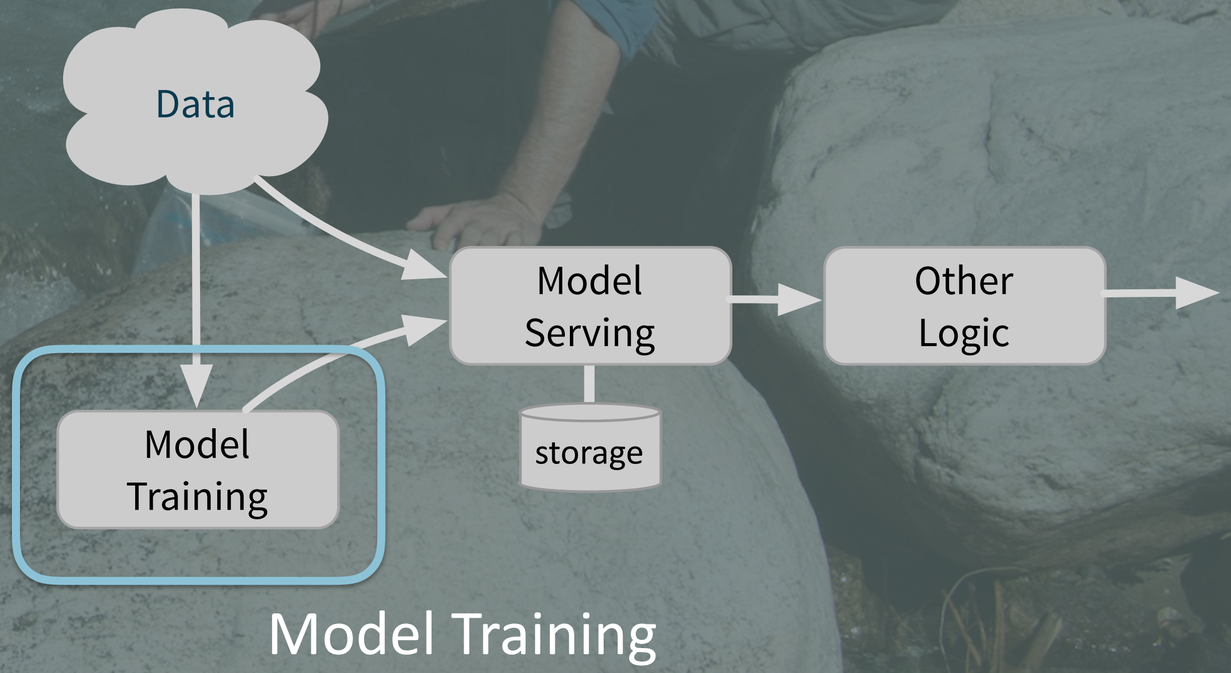
https://www.coursera.org/learn/machine-learning

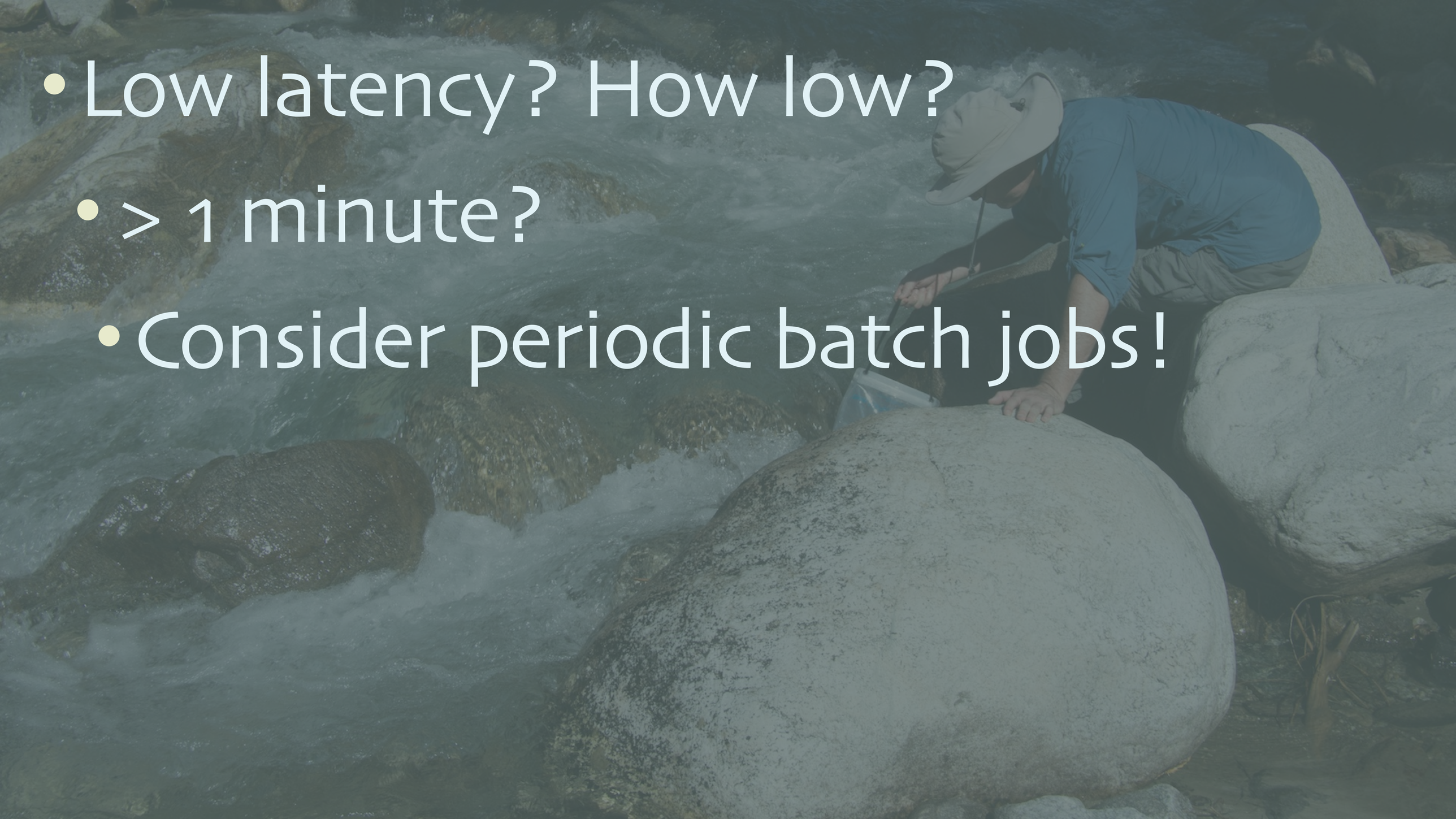


- Low latency? How low?
  - < hundreds of milliseconds?
    - "micro-batches"
      - Processing records in bulk, e.g., Spark's micro-batch model and "streaming SQL" over windows.

- Low latency? How low?
  - < 1 second to minutes?



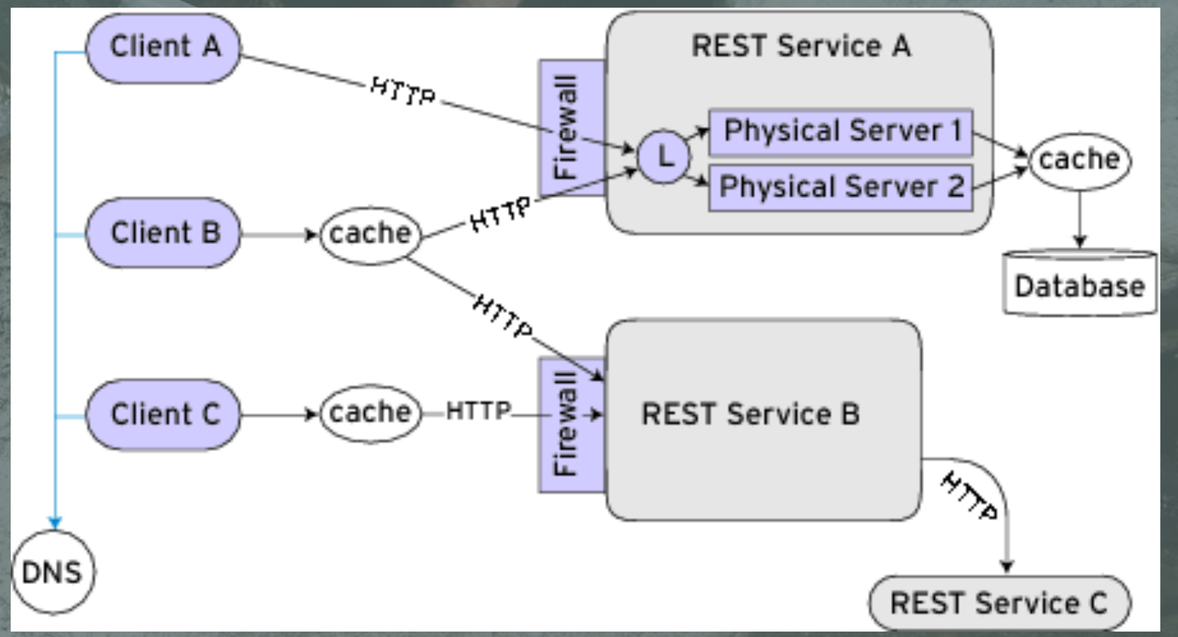




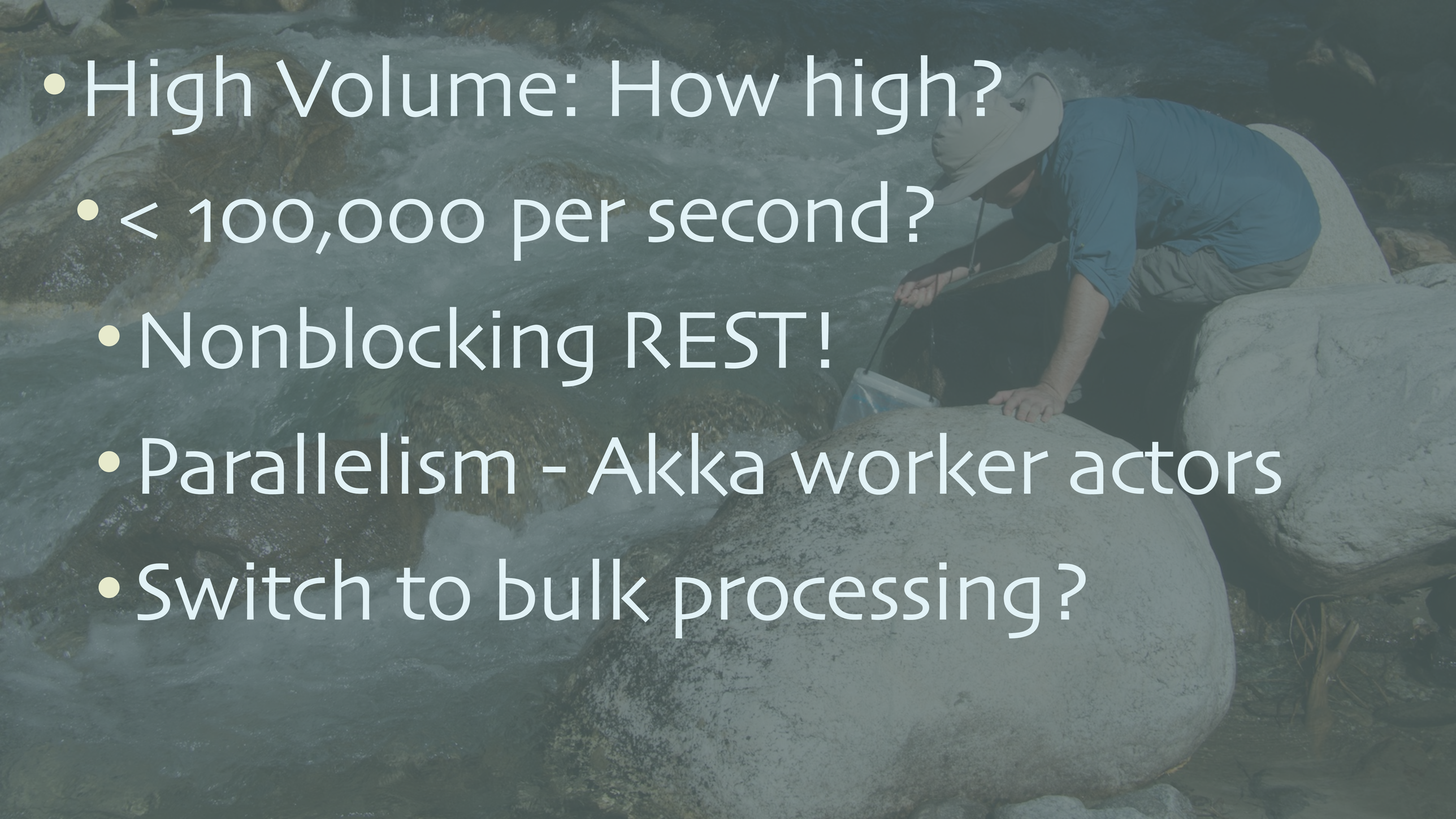
- Low latency? How low?
  - > 1 minute?
    - Consider periodic batch jobs!

- High Volume: How high?

- High Volume: How high?
  - < 10,000 events/second?
    - REST
    - One at a time…



http://www.drdobbs.com/web-development/soa-web-services-and-restful-systems/199902676



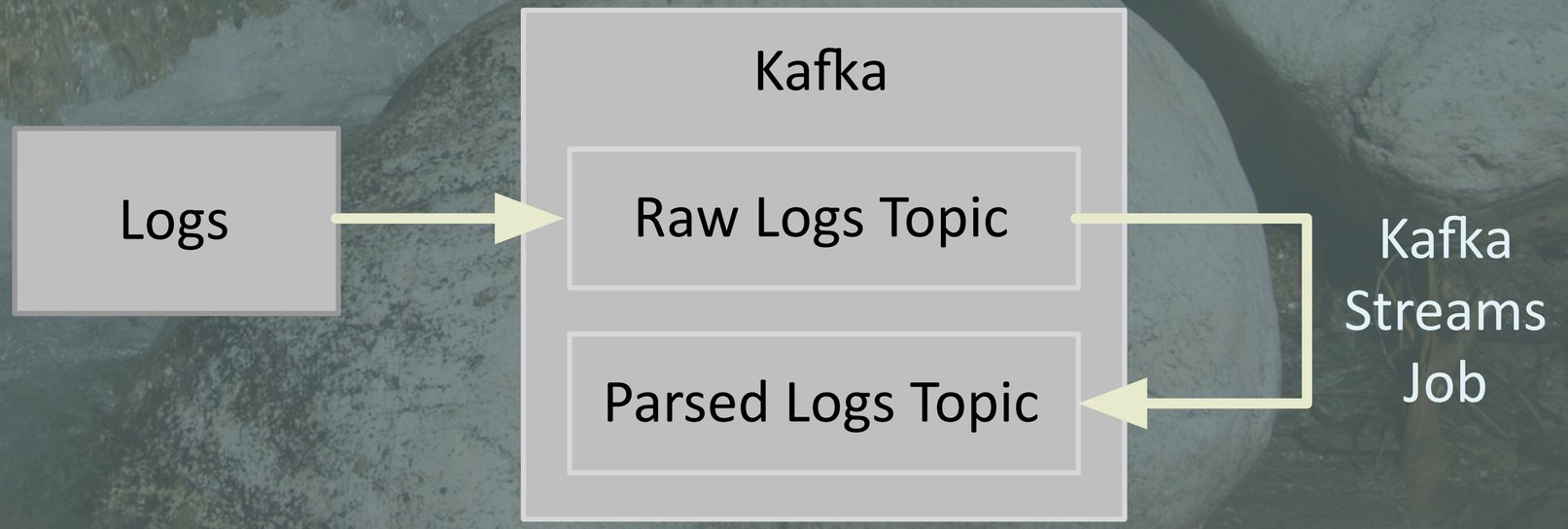
- High Volume: How high?
  - < 100,000 per second?
    - Nonblocking REST!
    - Parallelism - Akka worker actors
    - Switch to bulk processing?

- High Volume: How high?
  - 1,000,000s per second?
    - Flink or Spark Streaming
    - Process in bulk

https://store.nest.com/product/thermostat/

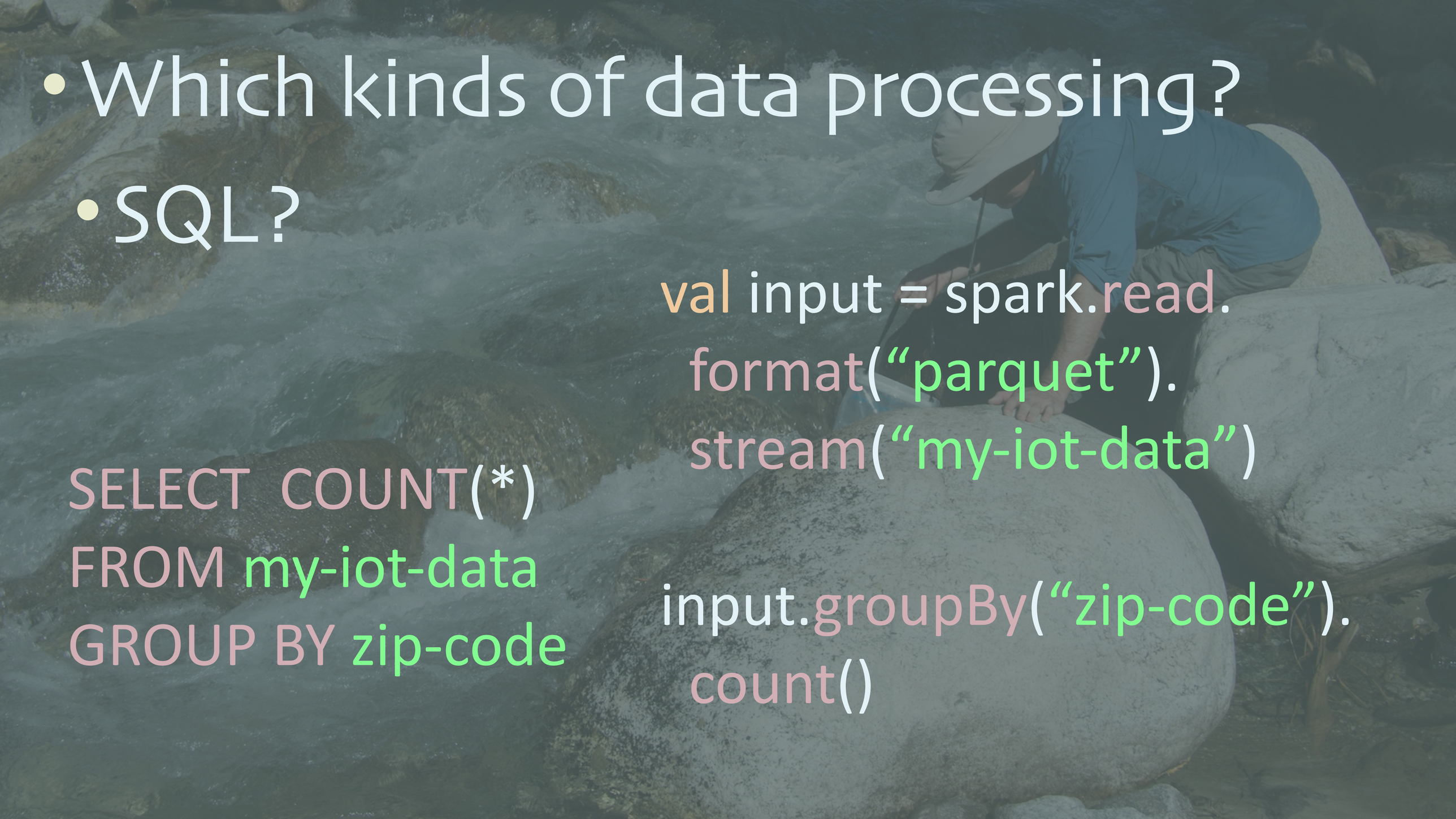
- Which kinds of data processing?

- Which kinds of data processing?
  - Extract, transform, and load (ETL)?



- Which kinds of data processing?
- "Dataflow" pipelines

```
val sc = new SparkContext("local[*]", "Inverted Idx")
sc.textFile("data/crawl")
  .map { line => val Array(path, text) = line.split("\t",2);
(path, text)
  } flatMap {
   case (path, text) => text.split("""\W+""").map((_, path))
  } map {
   case (w, p) => ((w, p), 1)
  } reduceByKey {
   case (n1, n2) => n1 + n2
```

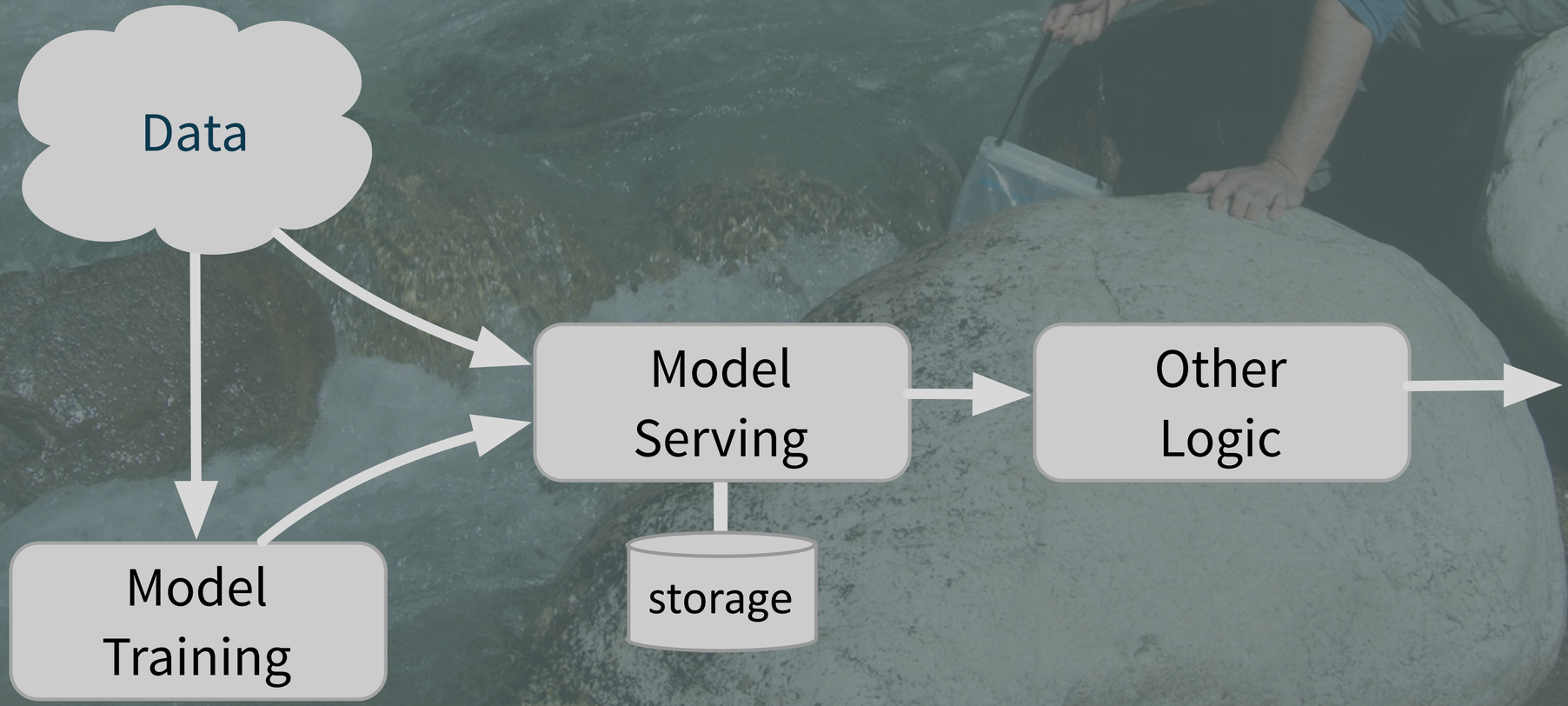


- Which kinds of data processing?
  - SQL?

```
SELECT  COUNT(*)
FROM my-iot-data
GROUP BY zip-code
```

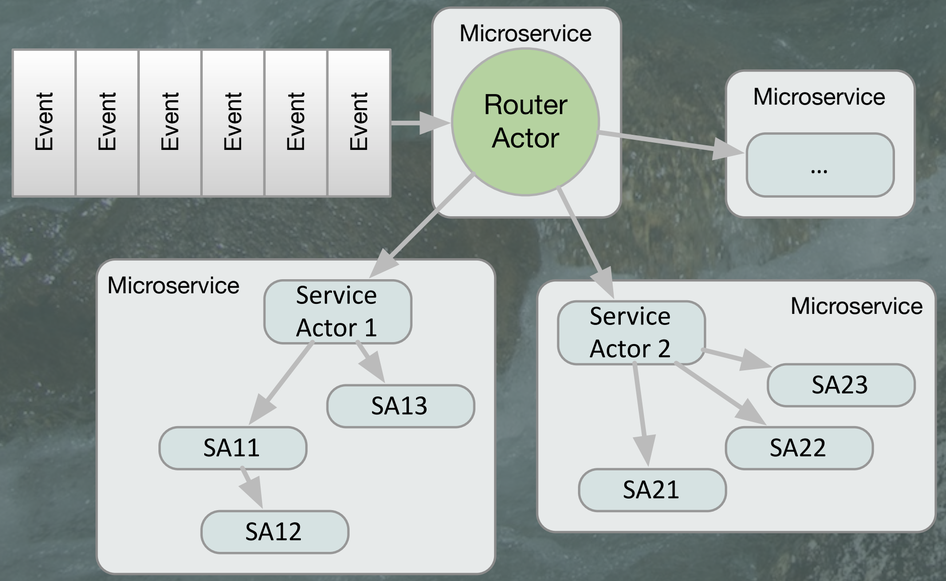
```
val input = spark.read.
  format(“parquet”).
  stream(“my-iot-data”)

input.groupBy(“zip-code”).
  count()
```

- Which kinds of data processing?
- Train and serve ML models?



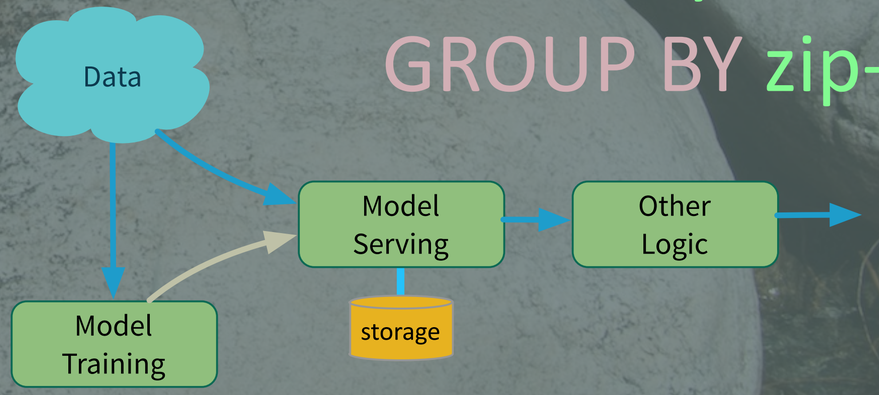
• Process data individually or in bulk?

Event-driven µ-services



"Record-centric" µ-services

```
SELECT  COUNT(*)
FROM my-iot-data
GROUP BY zip-code
```

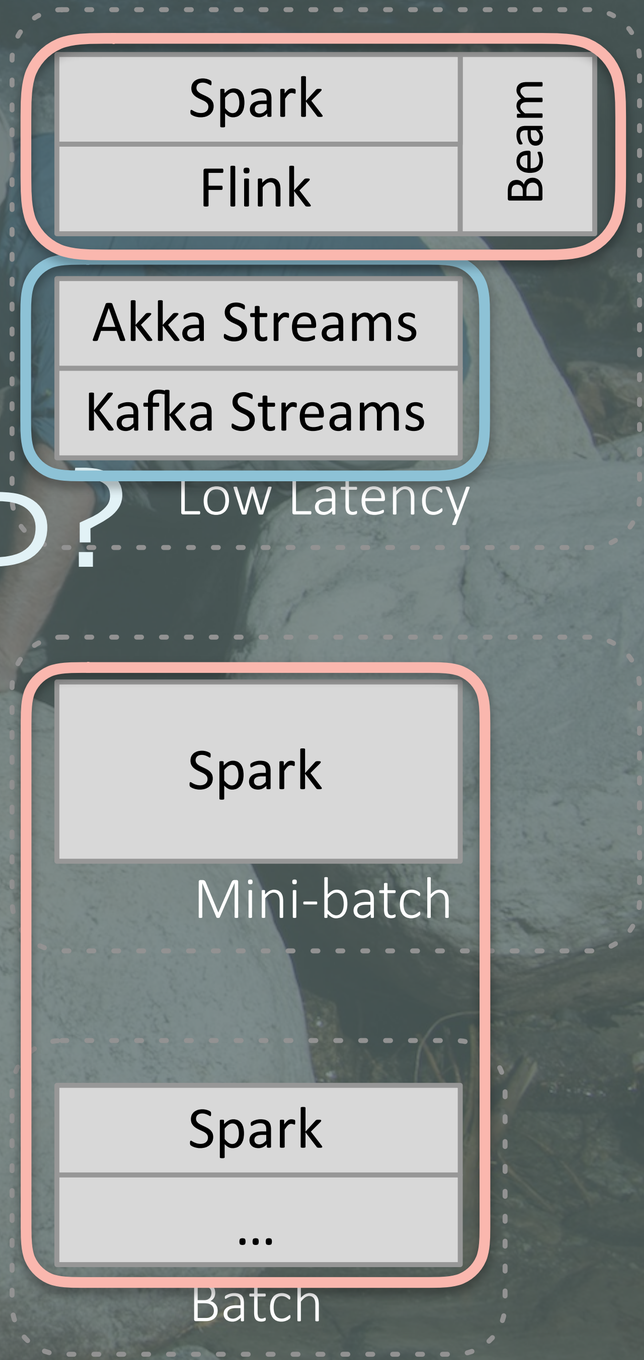


Events

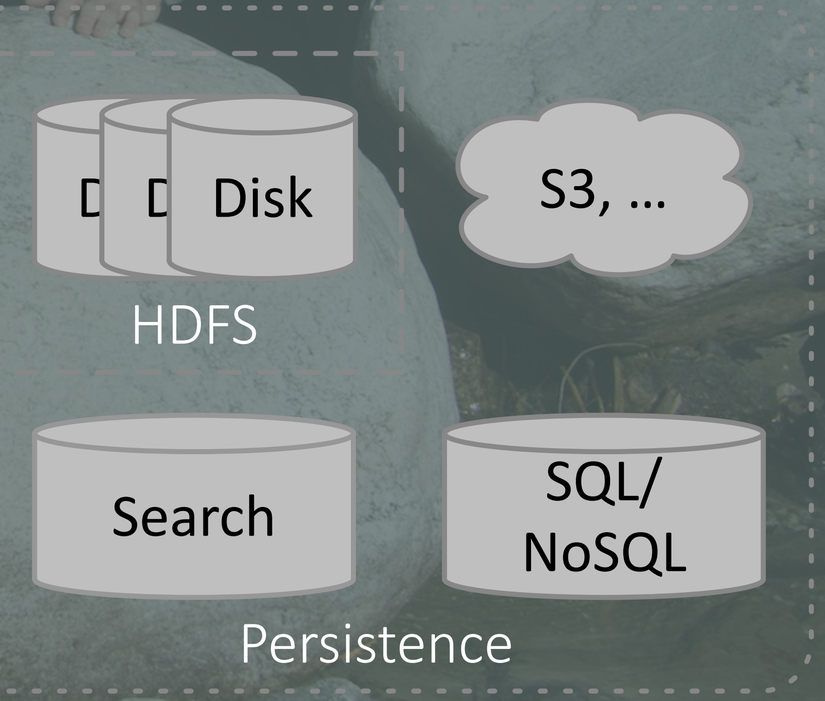
Records

- Preferred application architecture?
- Streaming library in an app?
- or, distributed services running your job?

Already have a microservices-based, DevOps CI/CD workflow? Stream processing with microservices may fit better into your environment!



- Integration with other tools.
  - Akka, Flink, & Spark integrate with Databases, Kafka, file systems, REST, …
  - Kafka Streams only read & write Kafka topics.



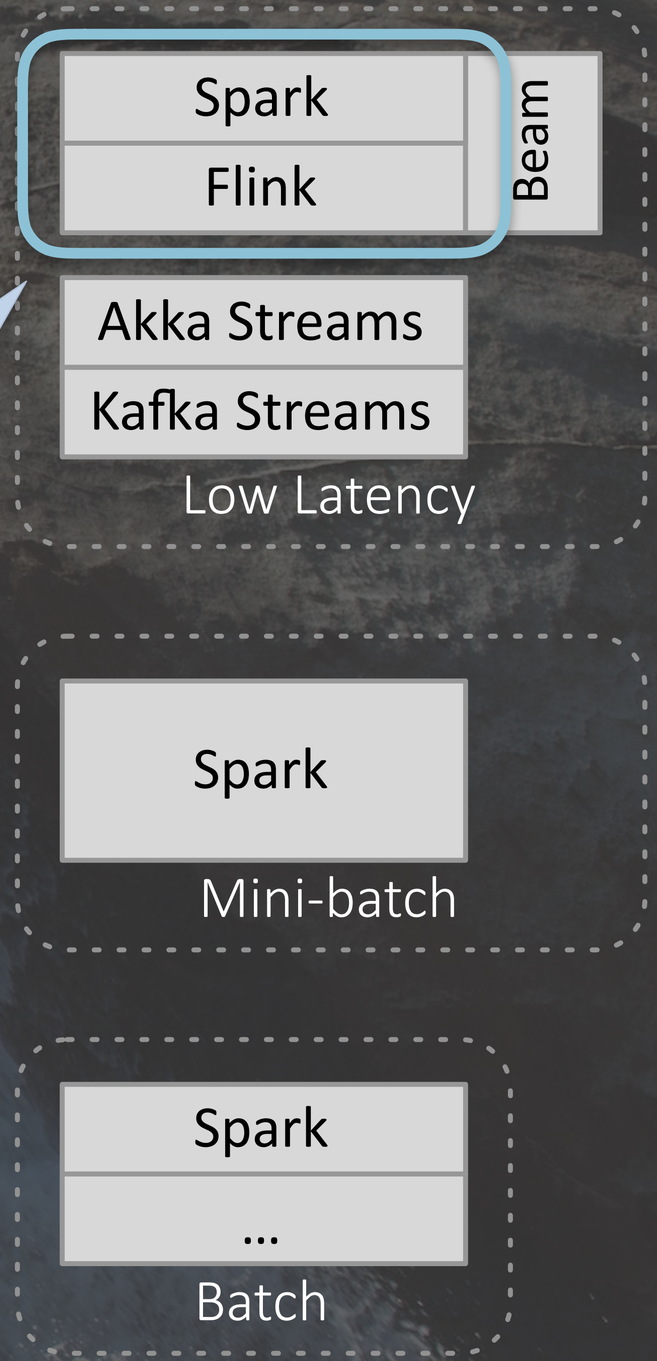

Best of Breed
Streaming
Engines

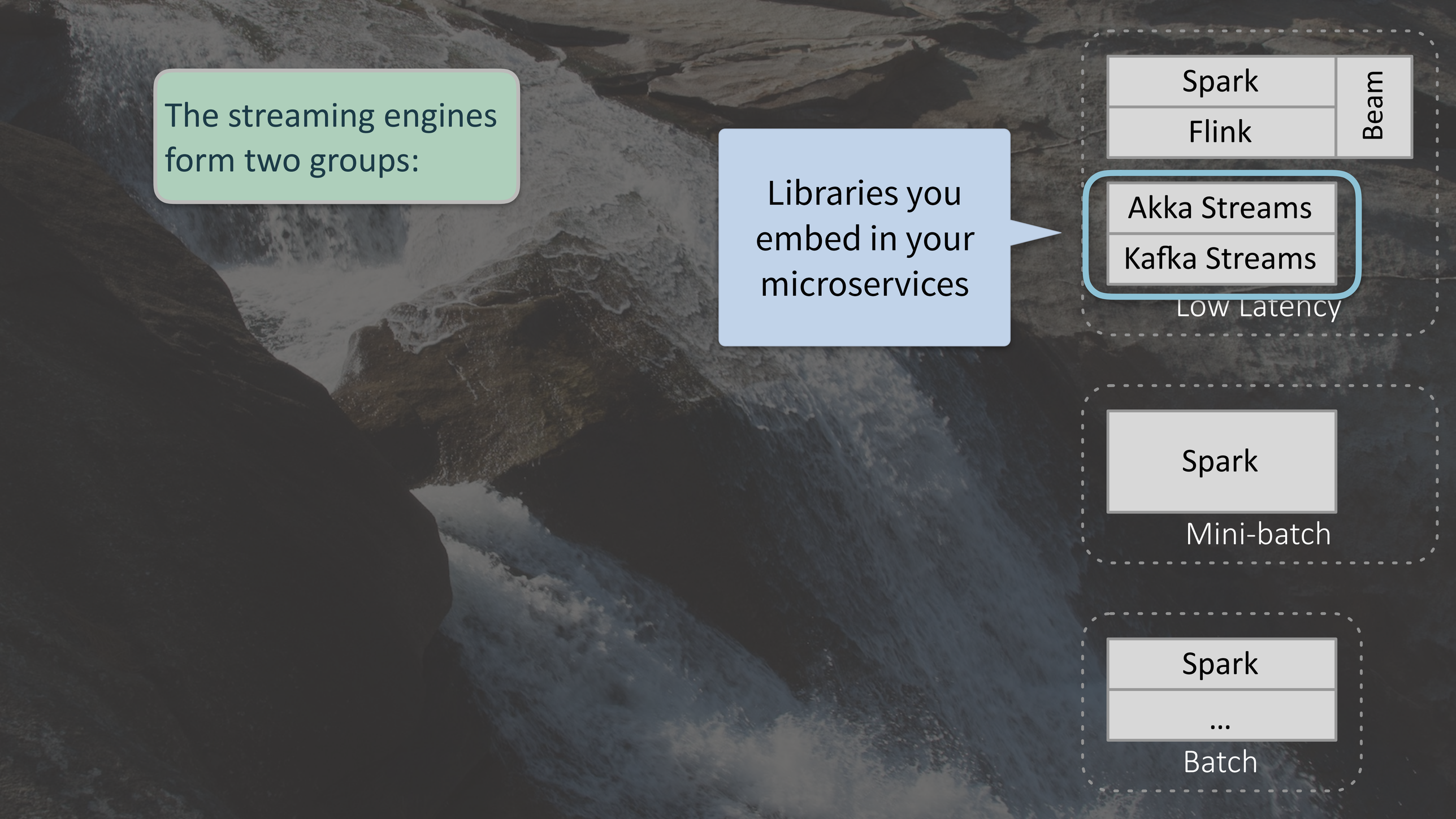

Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency
Spark
Mini-batch
Spark
...
Batch

The streaming engines form two groups:

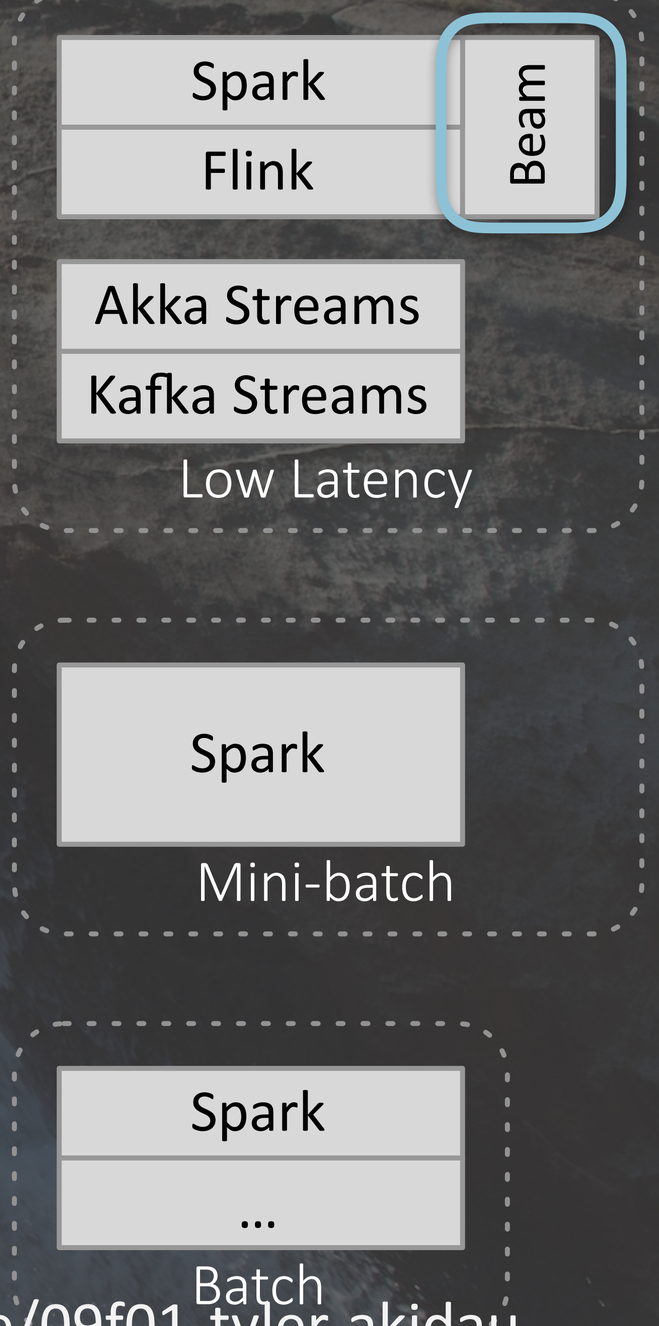
Run as distributed services

You submit jobs, they are partitioned into tasks

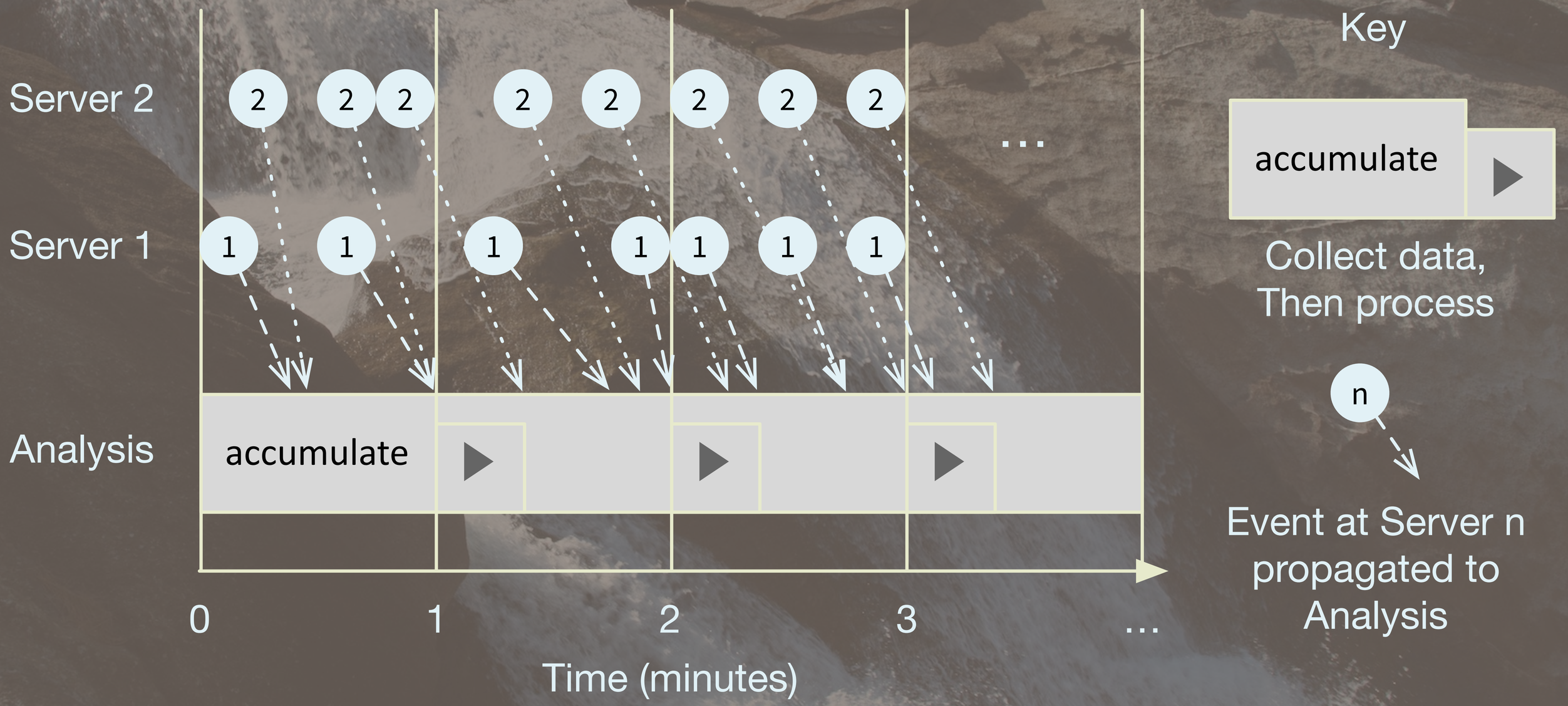



The streaming engines form two groups:
Libraries you embed in your microservices
Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency
Spark
Mini-batch
Spark
...
Batch

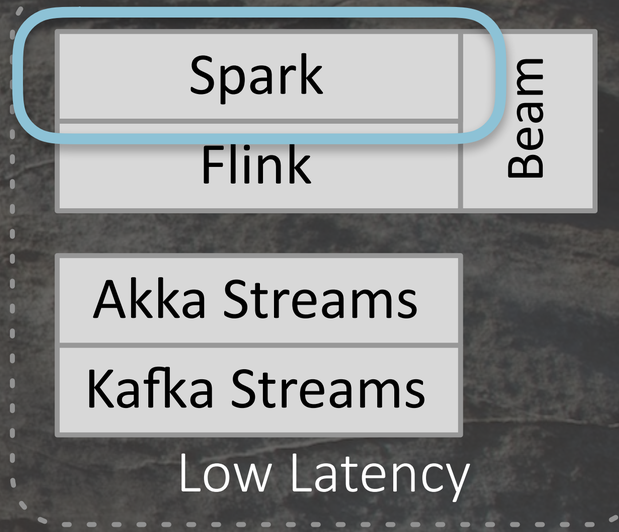
- Apache Beam
  - (Google Dataflow)
  - Requires a "runner"
  - Most sophisticated streaming semantics

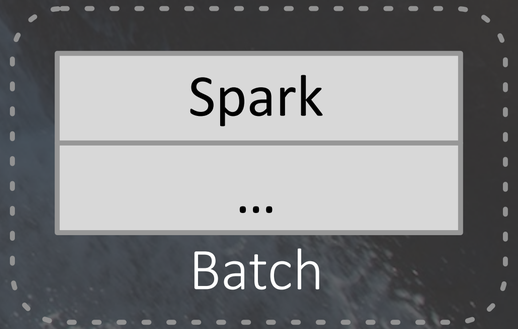


See these blog posts: https://www.oreilly.com/people/09f01-tyler-akidau

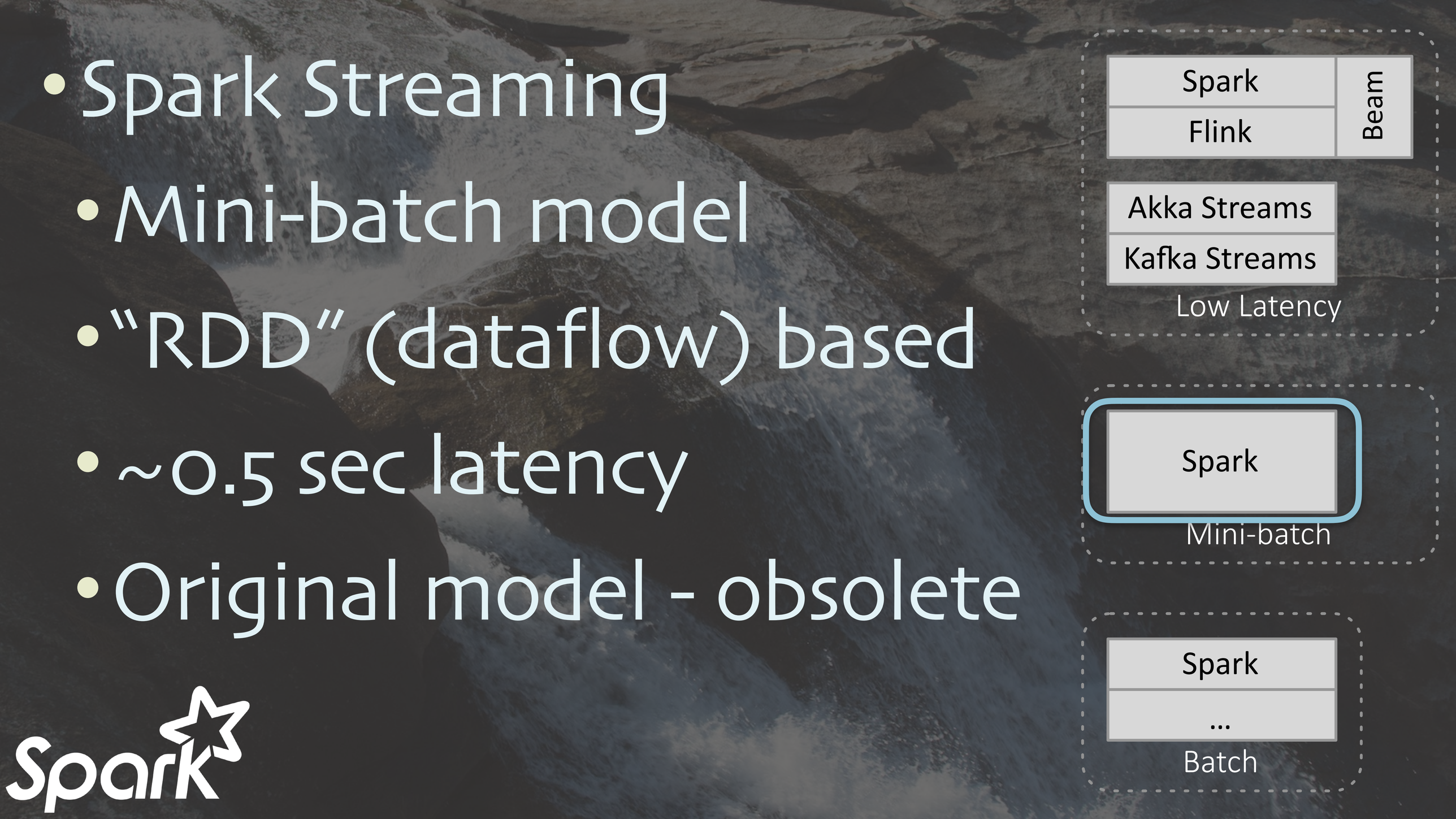

Server 2
Server 1
Analysis
accumulate
2
1
...
0
1
2
3
...
Time (minutes)
Key
accumulate
Collect data,
Then process
n
Event at Server n
propagated to
Analysis

- Spark Structured Streaming
  - "Dataset" - SQL
  - Millisecond latency
  - Ideal for Rich SQL, ML.

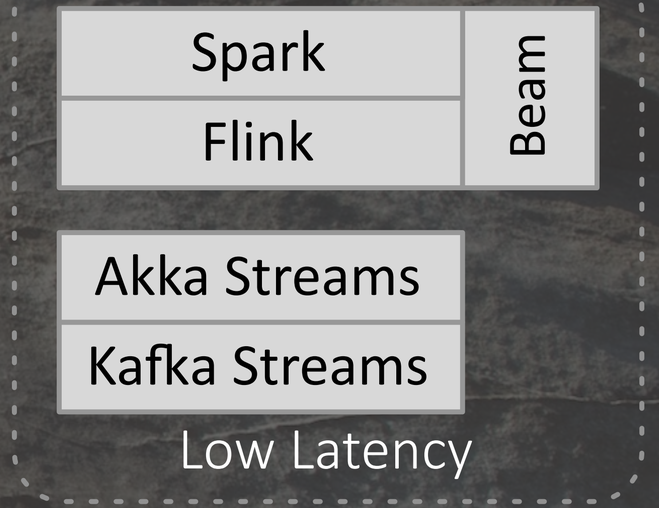


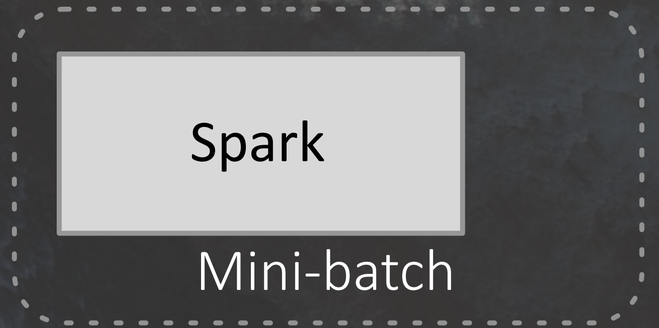


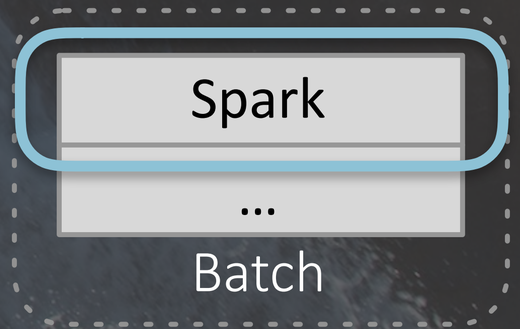
- Spark Streaming
  - Mini-batch model
  - "RDD" (dataflow) based
  - ~0.5 sec latency
  - Original model - obsolete



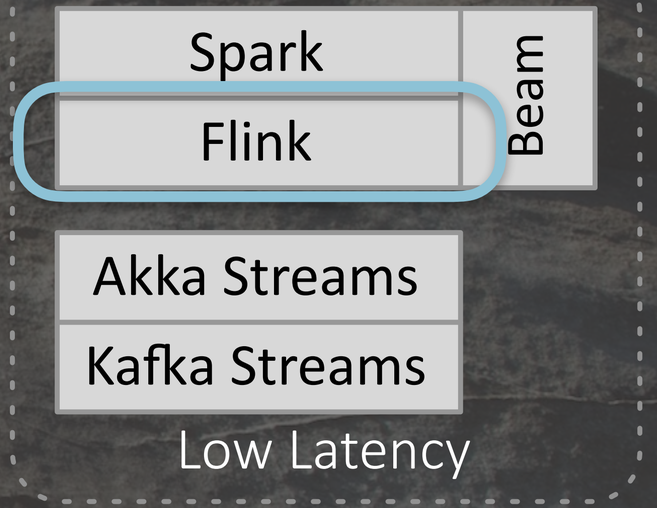
- Spark Batch
  - Same Dataset and RDD features as streaming.
  - Massive scalability
  - Excellent performance

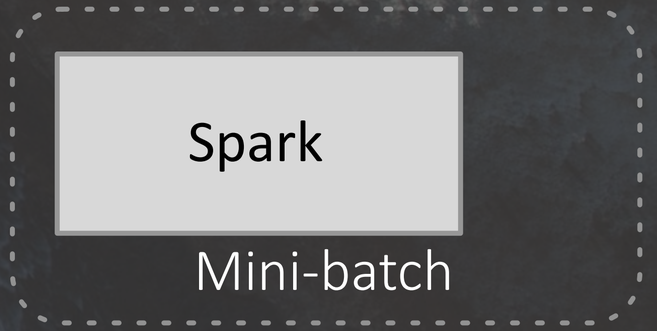


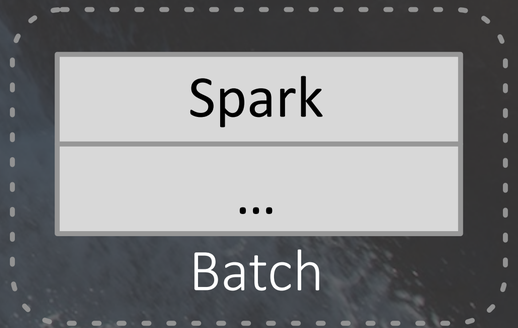




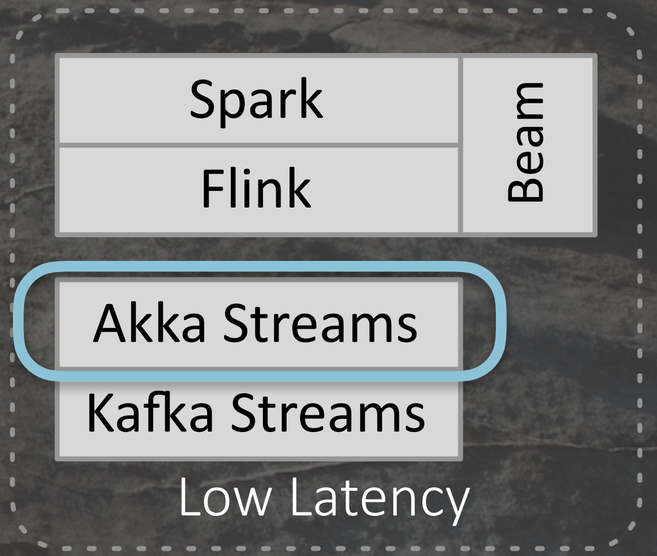
- Apache Flink
  - High volume, low latency
  - Sophisticated streaming (Beam) semantics
  - SQL, evolving ML support

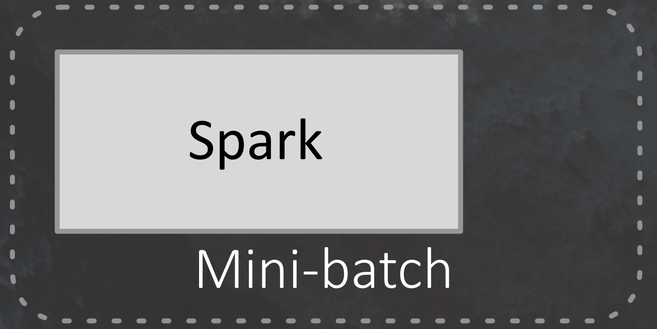


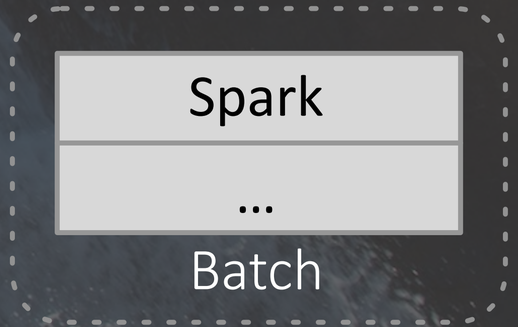




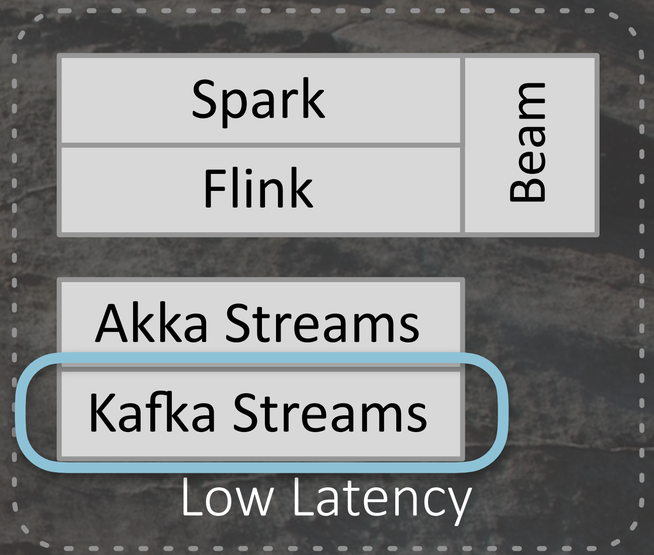
- Akka Streams
  - Low latency
  - Complex Event Processing
    - Efficient, per event
    - Mid-volume pipelines

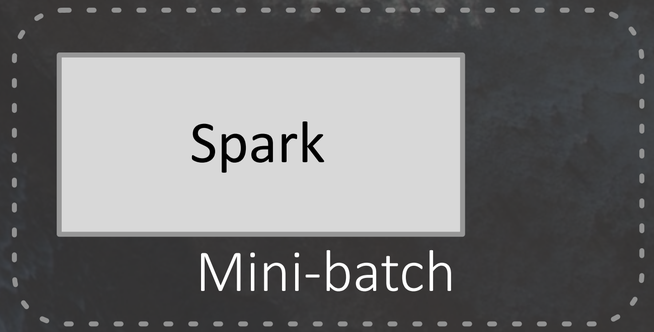


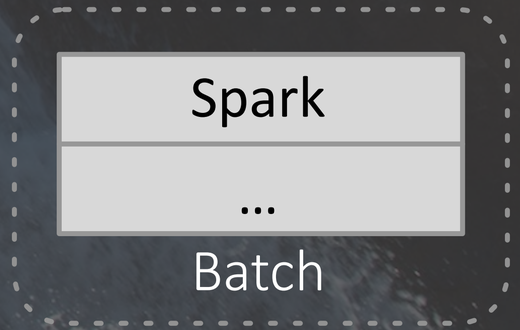




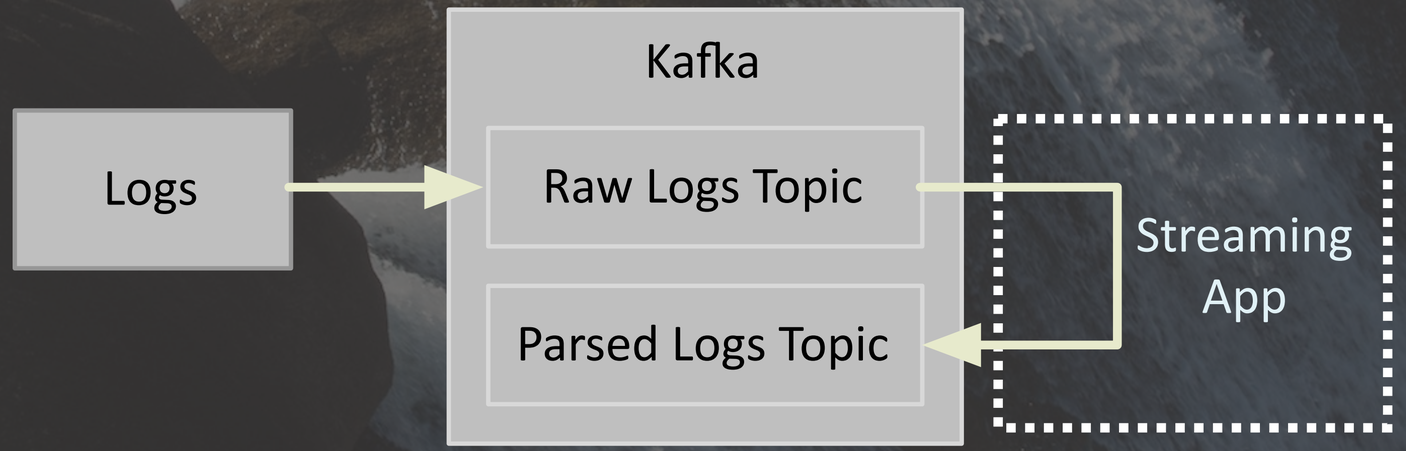
- Kafka Streams
  - Low overhead Kafka topic processing
  - Ideal for ETL and aggregations

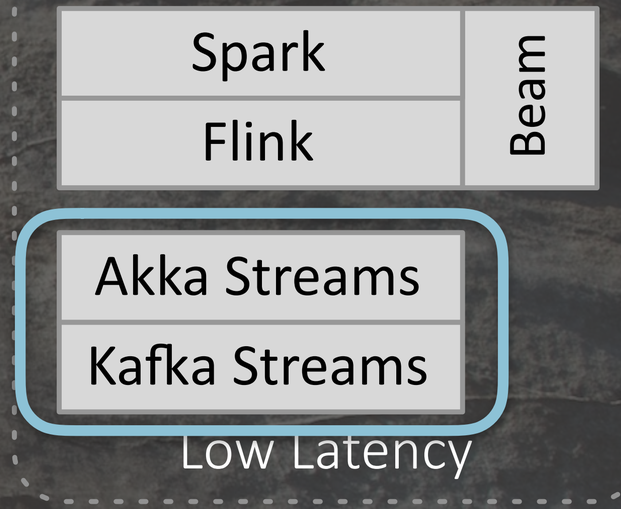


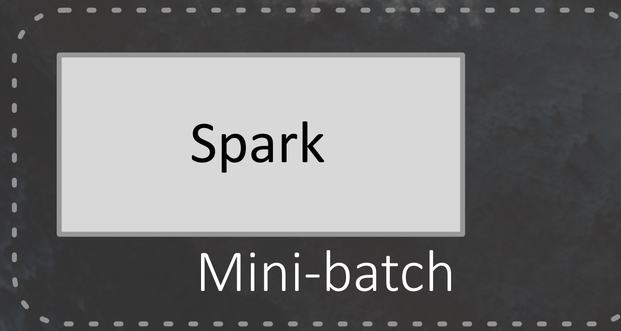


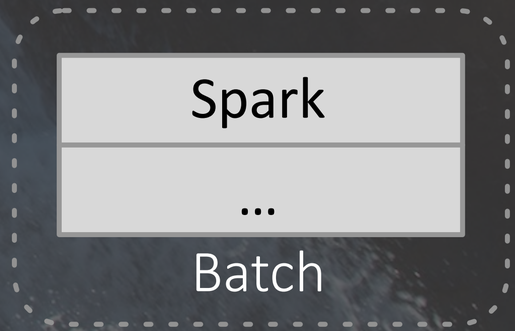


- Akka and Kafka Streams
  - "Exactly once" with transactions





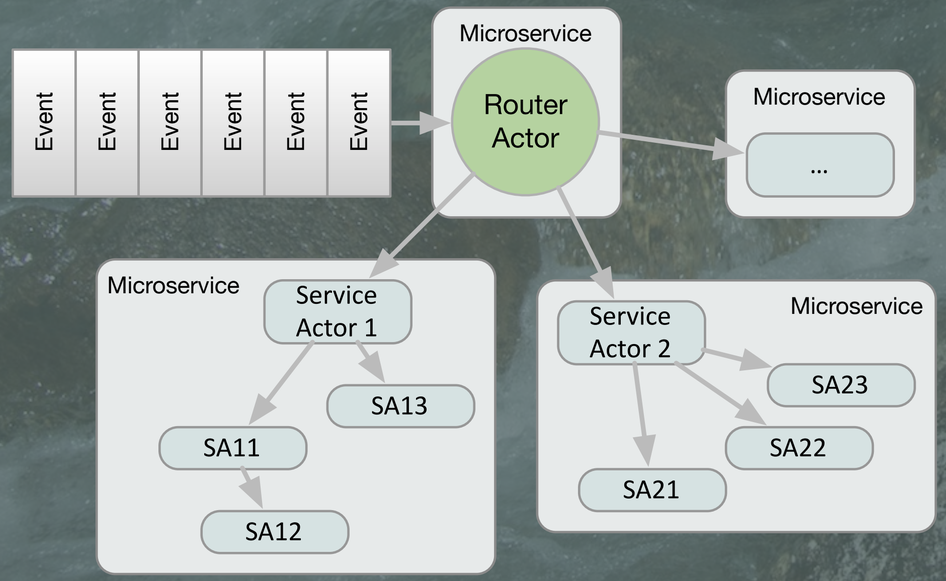




• Process data individually or in bulk?

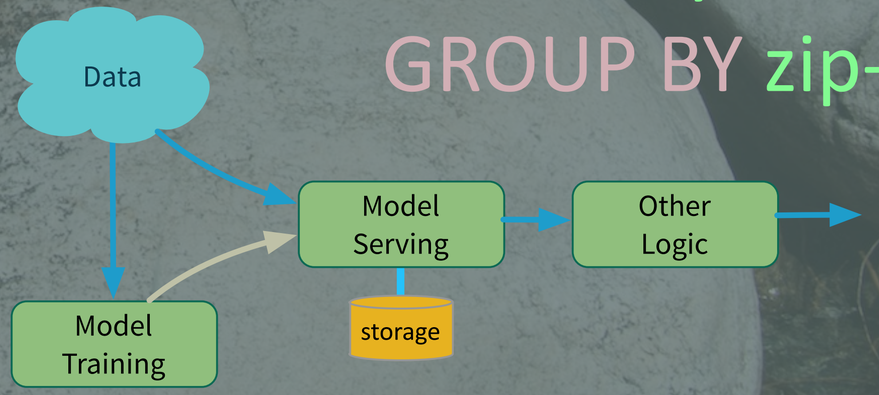
Each grew out of one end of this spectrum…

Event-driven µ-services



• "Record-centric" µ-services

```
SELECT  COUNT(*)
FROM my-iot-data
GROUP BY zip-code
```



Events

Records

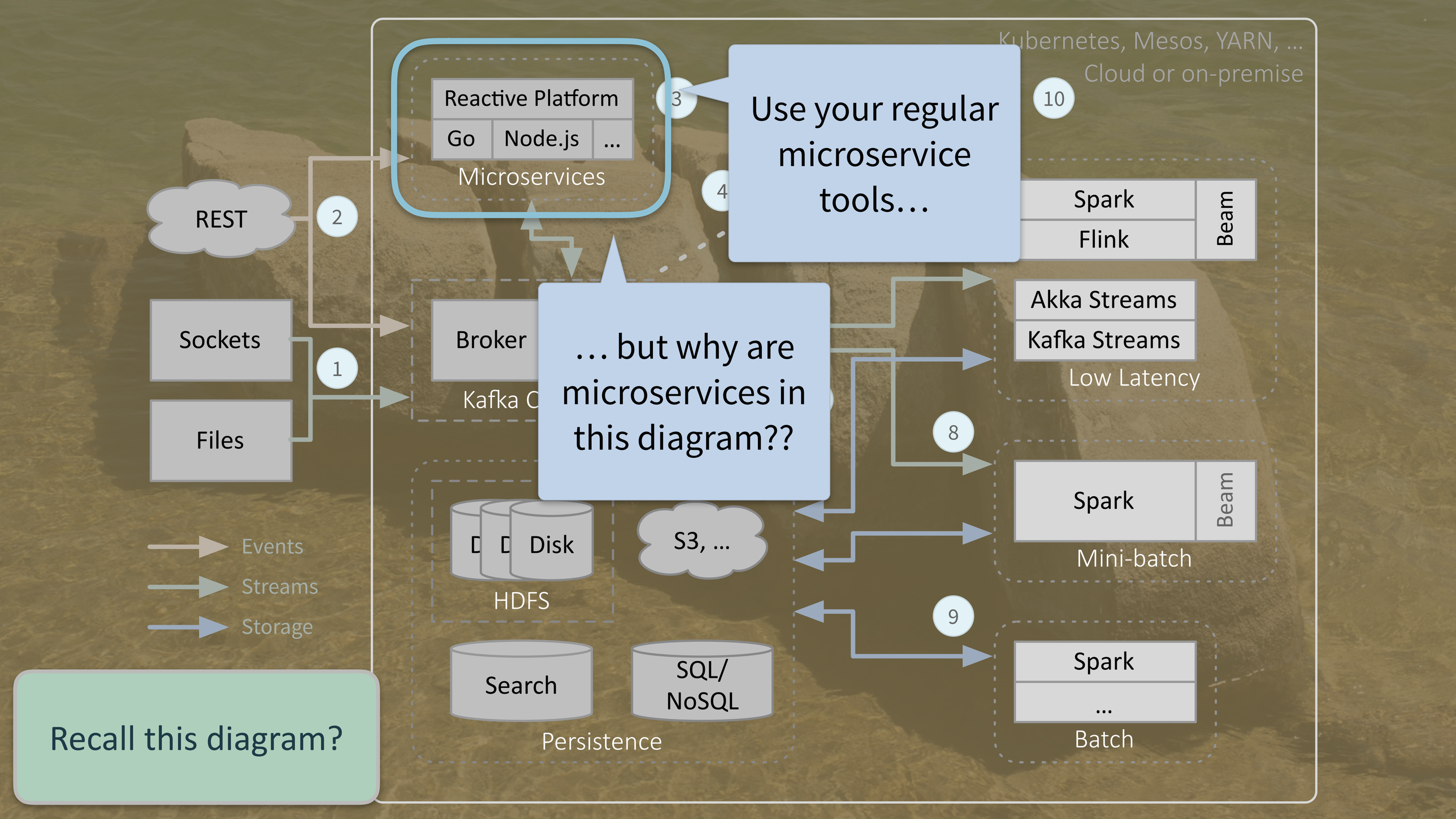
- Akka Streams vs. Kafka Streams talk
  - Also at polyglotprogramming.com/talks/

Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency

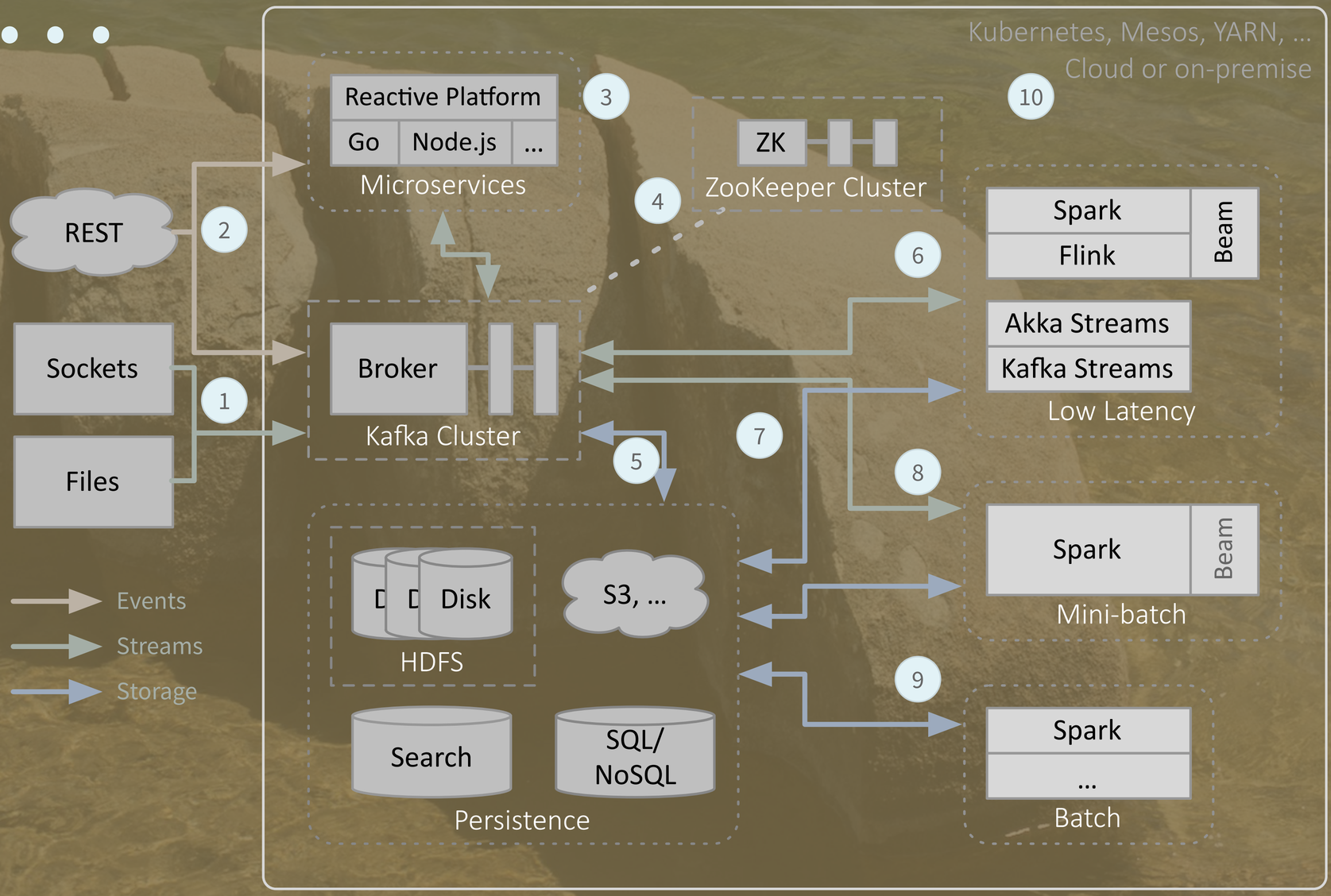
Spark
Mini-batch

Spark
...
Batch

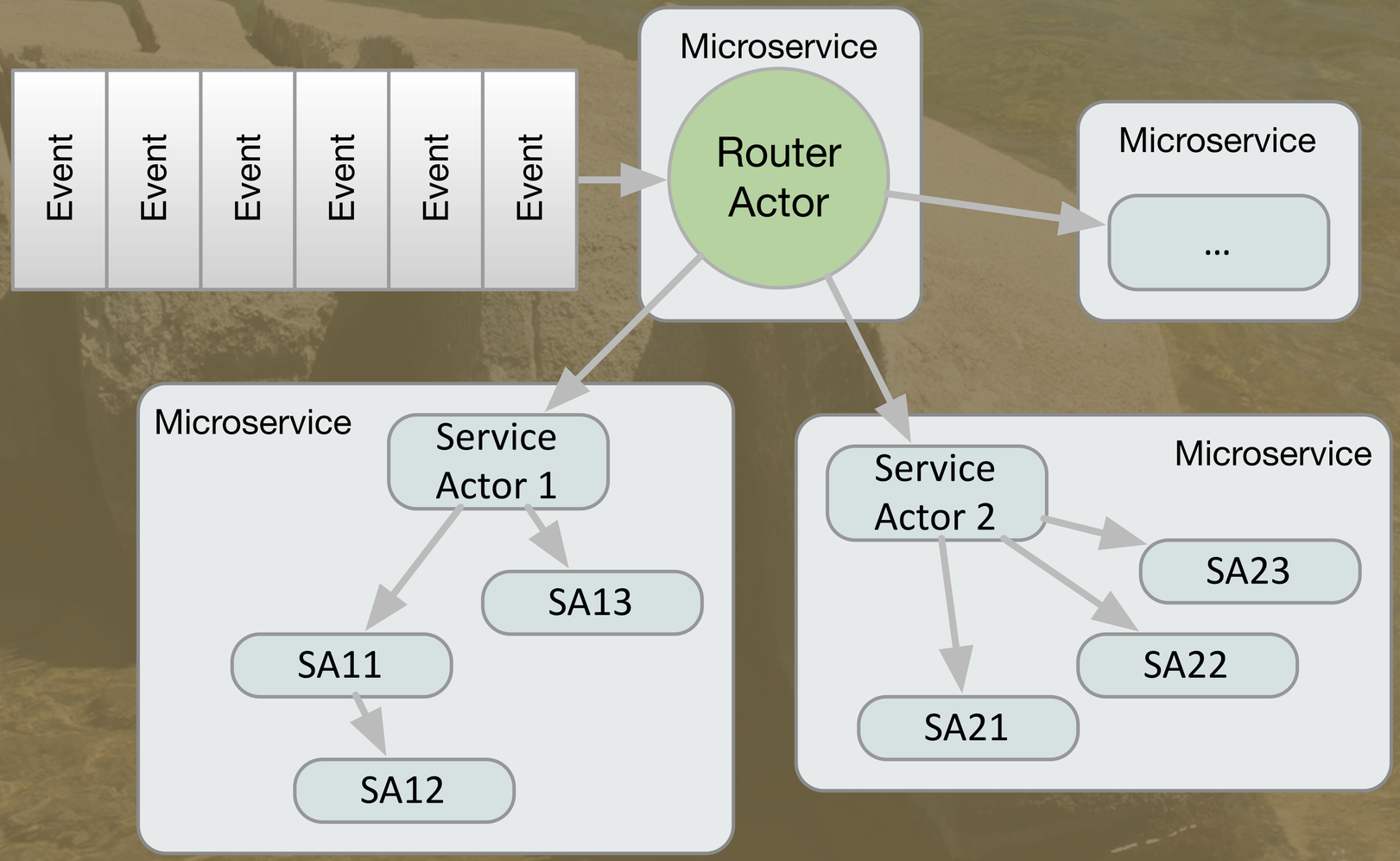
Microservices and Fast Data


Kubernetes, Mesos, YARN, ...
Cloud or on-premise
Reactive Platform
Go
Node.js
...
Microservices
3
Use your regular microservice tools…
10
4
REST
2
Sockets
1
Files
Broker
Kafka C
… but why are microservices in this diagram??
Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency
8
Spark
Beam
Mini-batch
Disk
S3, ...
HDFS
9
Search
SQL/
NoSQL
Persistence
Spark
...
Batch
Events
Streams
Storage
Recall this diagram?

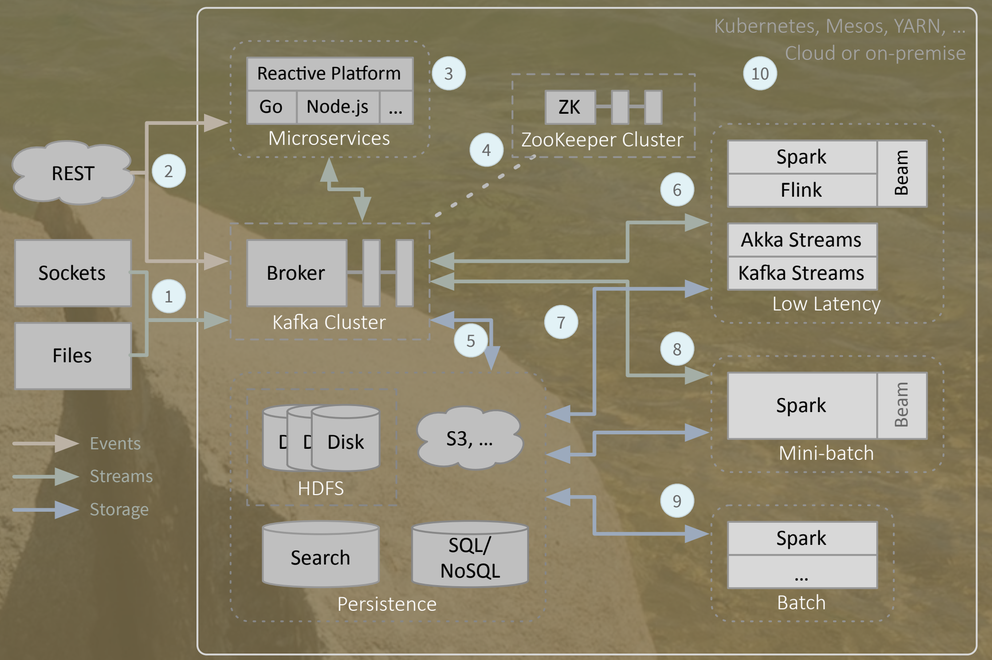
How is this…

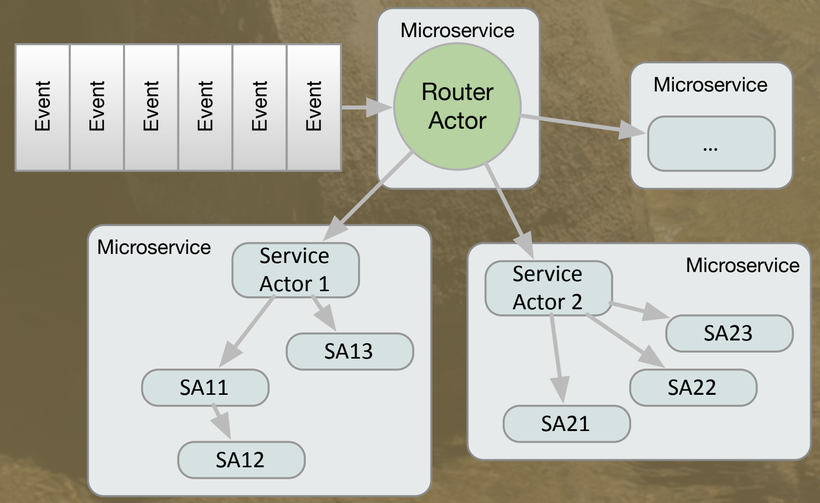

Kubernetes, Mesos, YARN, …
Cloud or on-premise
10
Reactive Platform
Go
Node.js
…
Microservices
3
ZK
ZooKeeper Cluster
4
REST
2
Sockets
1
Files
Broker
Kafka Cluster
5
6
Spark
Flink
Beam
Akka Streams
Kafka Streams
Low Latency
7
8
Spark
Beam
Mini-batch
Disk
HDFS
S3, …
9
Search
SQL/
NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage

… like this?

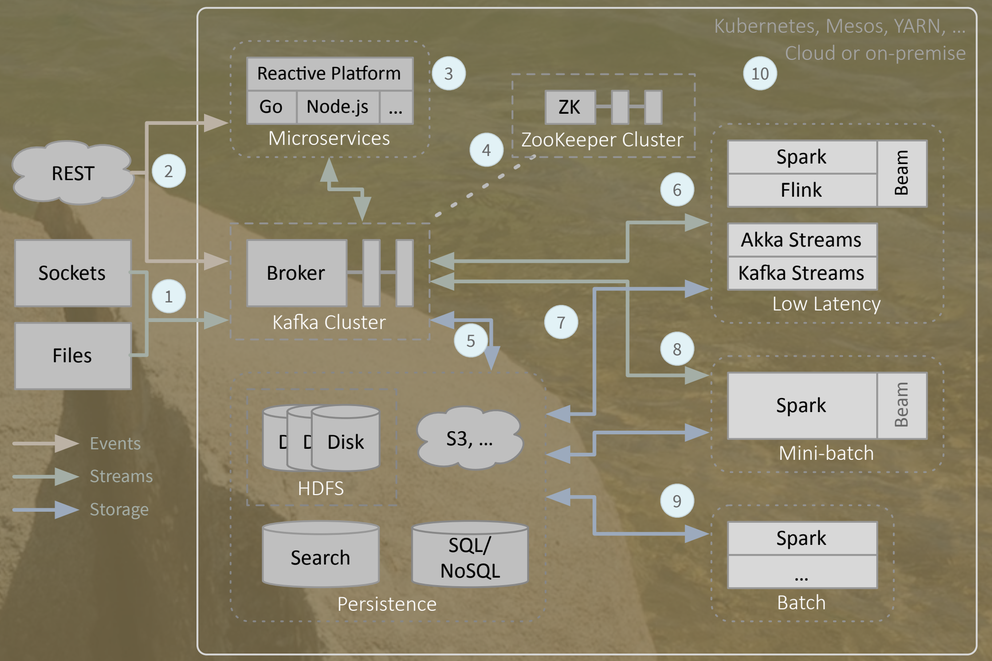

Event
Event
Event
Event
Event
Event
Microservice
Router Actor
Microservice
...
Microservice
Service Actor 1
SA13
SA11
SA12
Service Actor 2
Microservice
SA23
SA22
SA21

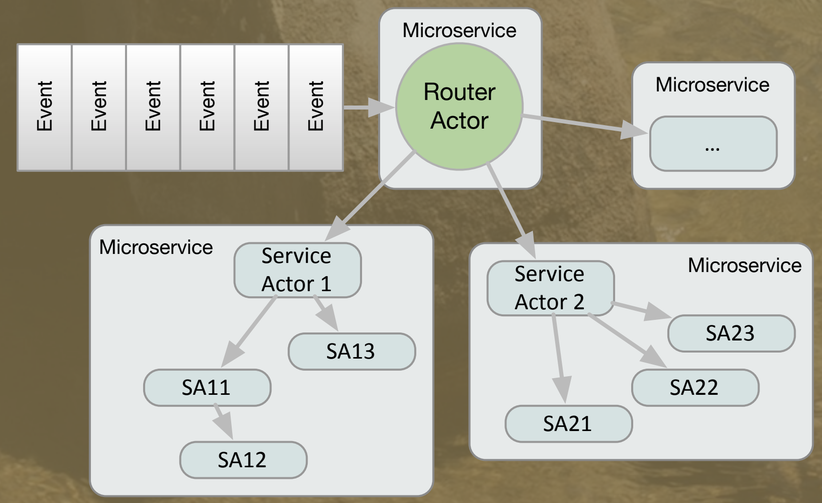
- A data app / microservice:
  - A single responsibility.
  - …





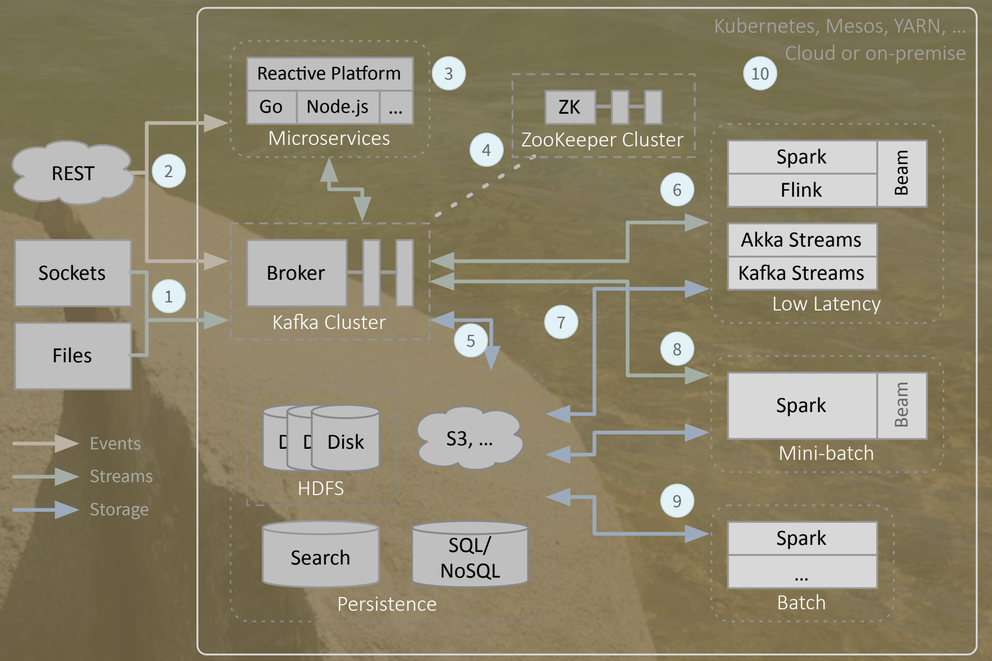
- A data app / microservice:
  - A single responsibility.
  - The input never ends.

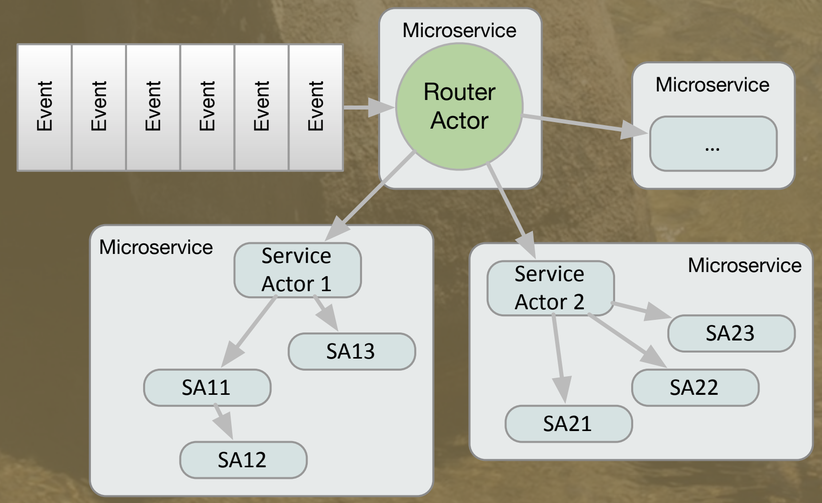




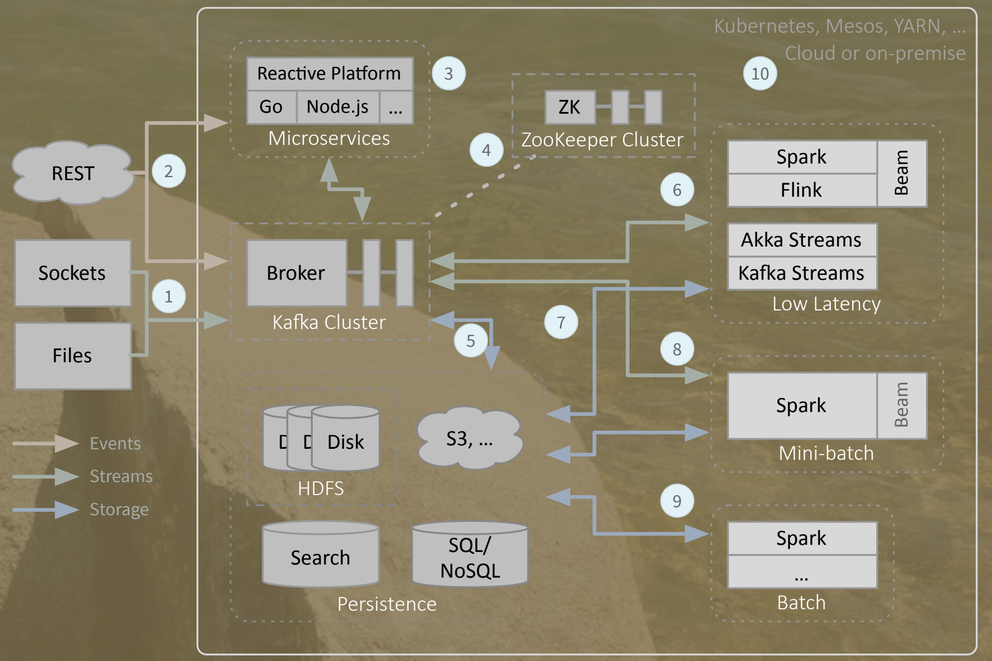
- A data app/microservice:
  - A single responsibility.
  - The input never ends.
    - So, both must be available, responsive, resilient, & scalable. I.e., reactive

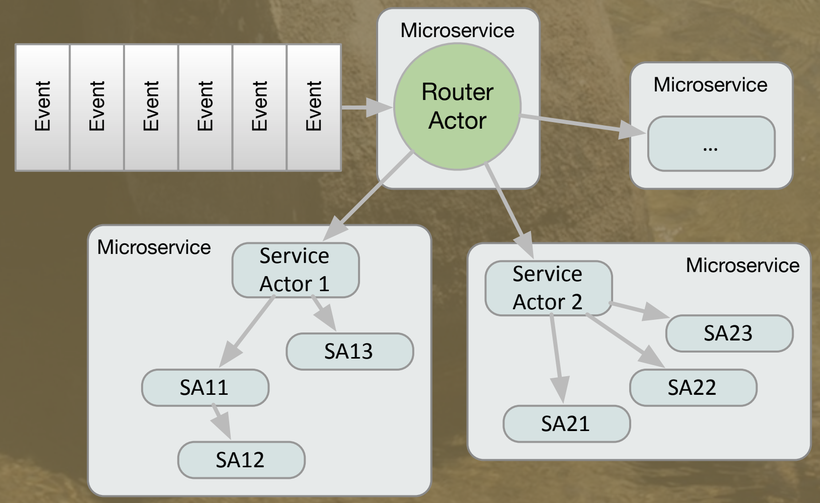
http://www.reactivemanifesto.org/



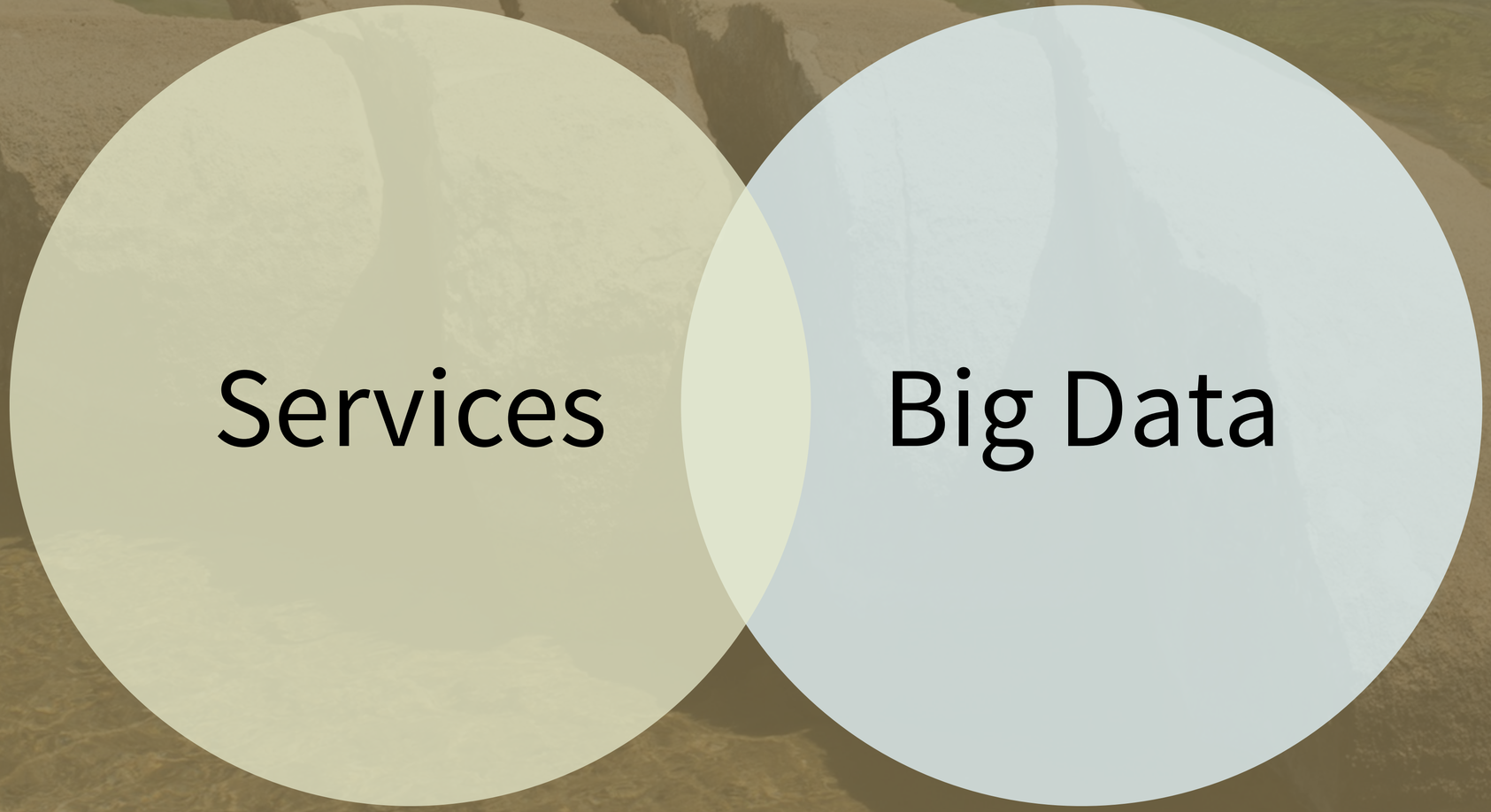


- Going the other way, “small” microservice architectures become data-centric, as the data grows.





The Recent Past



Some Overlap: Concerns, Architecture

The Present

Microservices
& *Fast* Data

Much More Overlap

The Future?

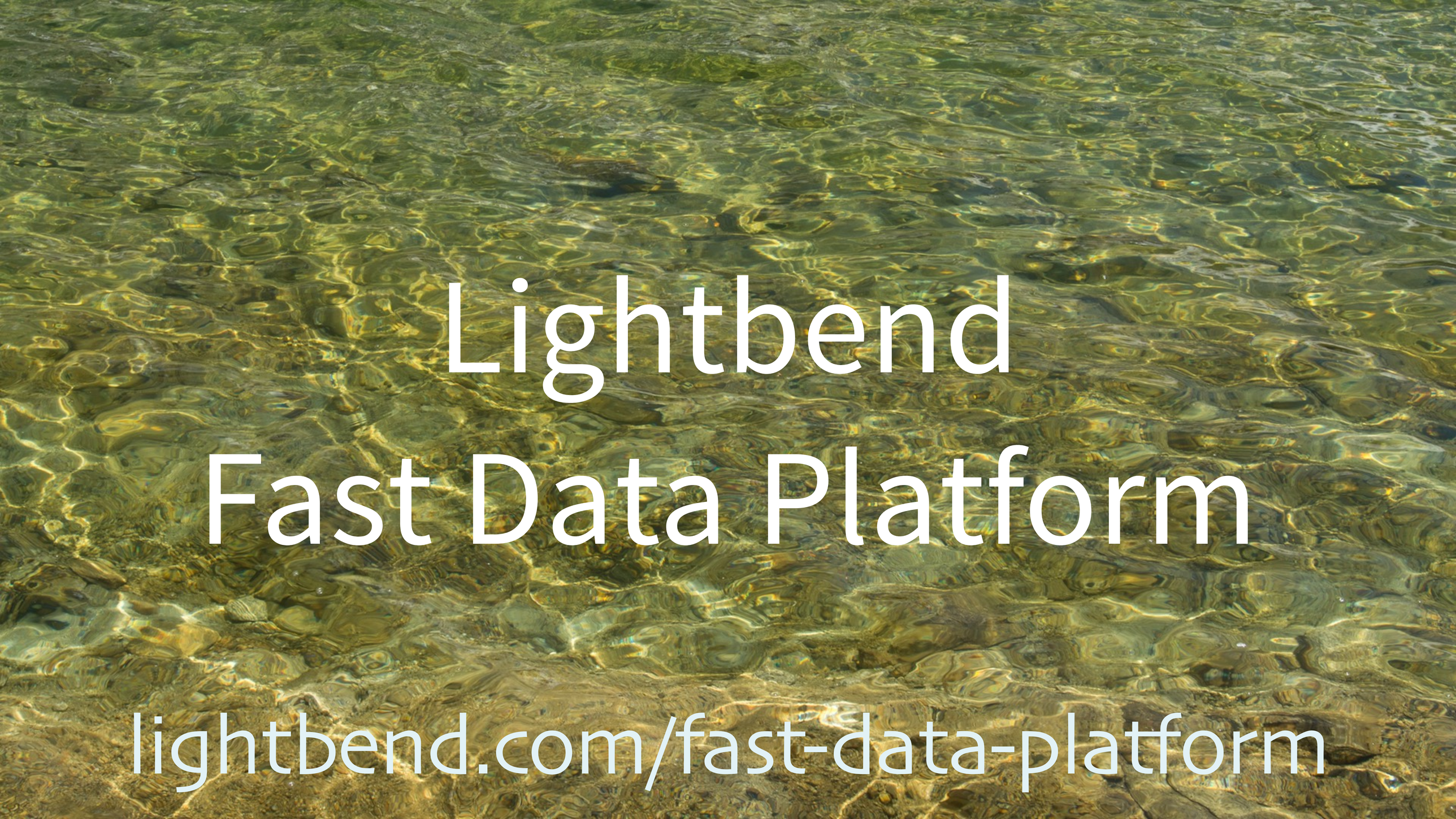
Why? Since streams process data incrementally, there is less need for large-scale tools like Spark, Flink

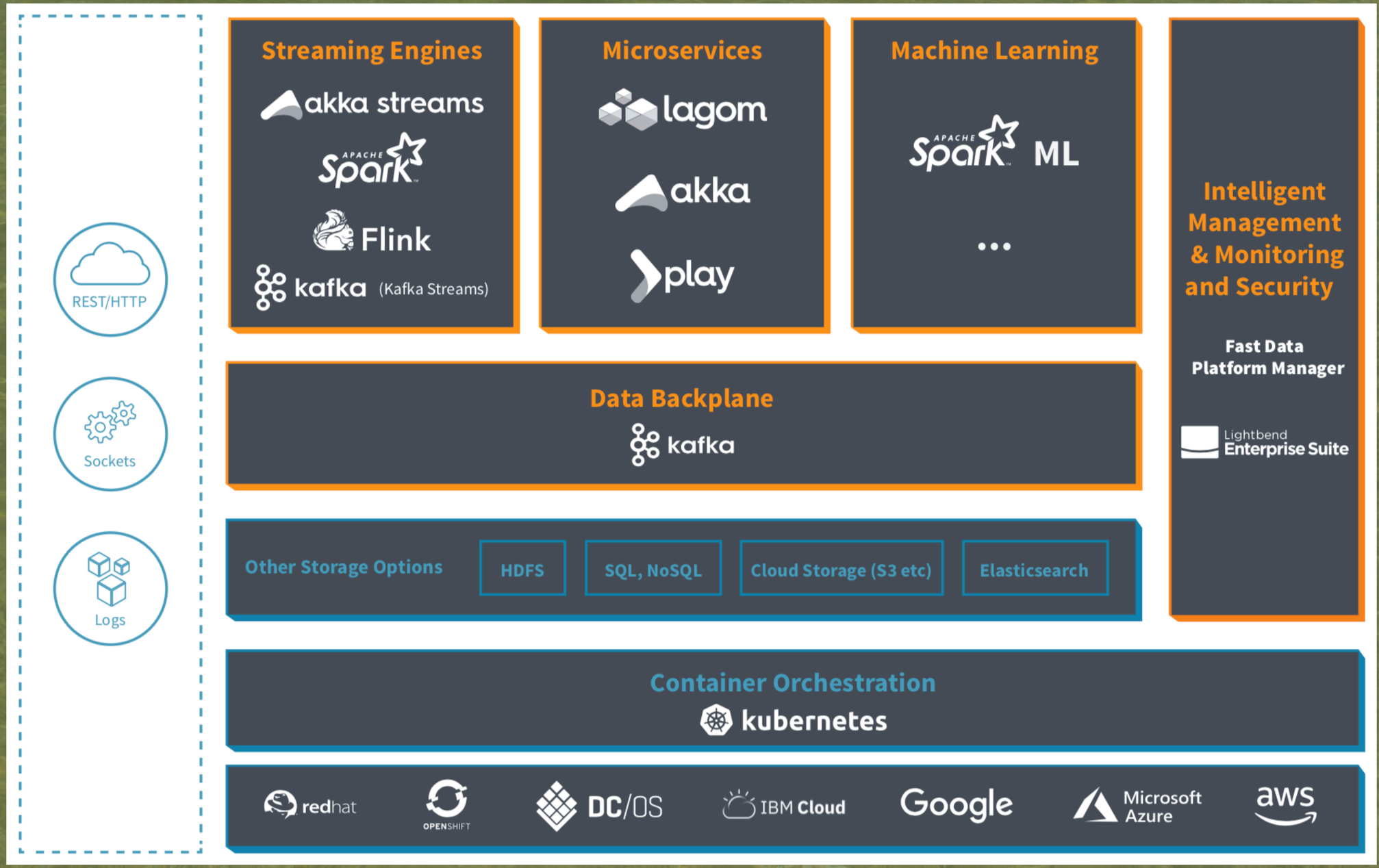
… and using microservices for everything simplifies development, deployment, and operations

Microservices *for* Fast Data

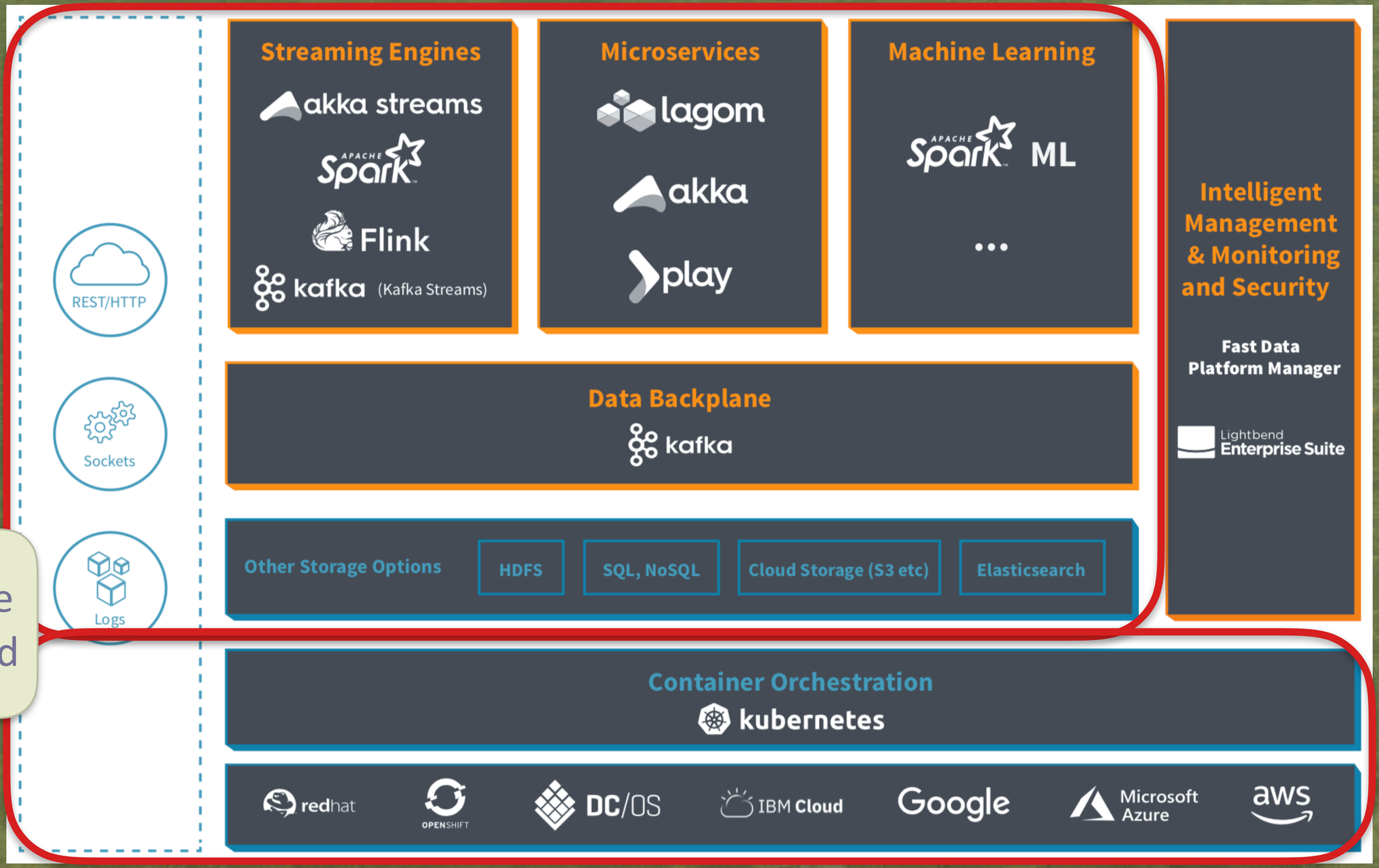
Unclear if this helps to integrate with the data science workflows we love.

Much more microservice focused?


Lightbend
Fast Data Platform
lightbend.com/fast-data-platform

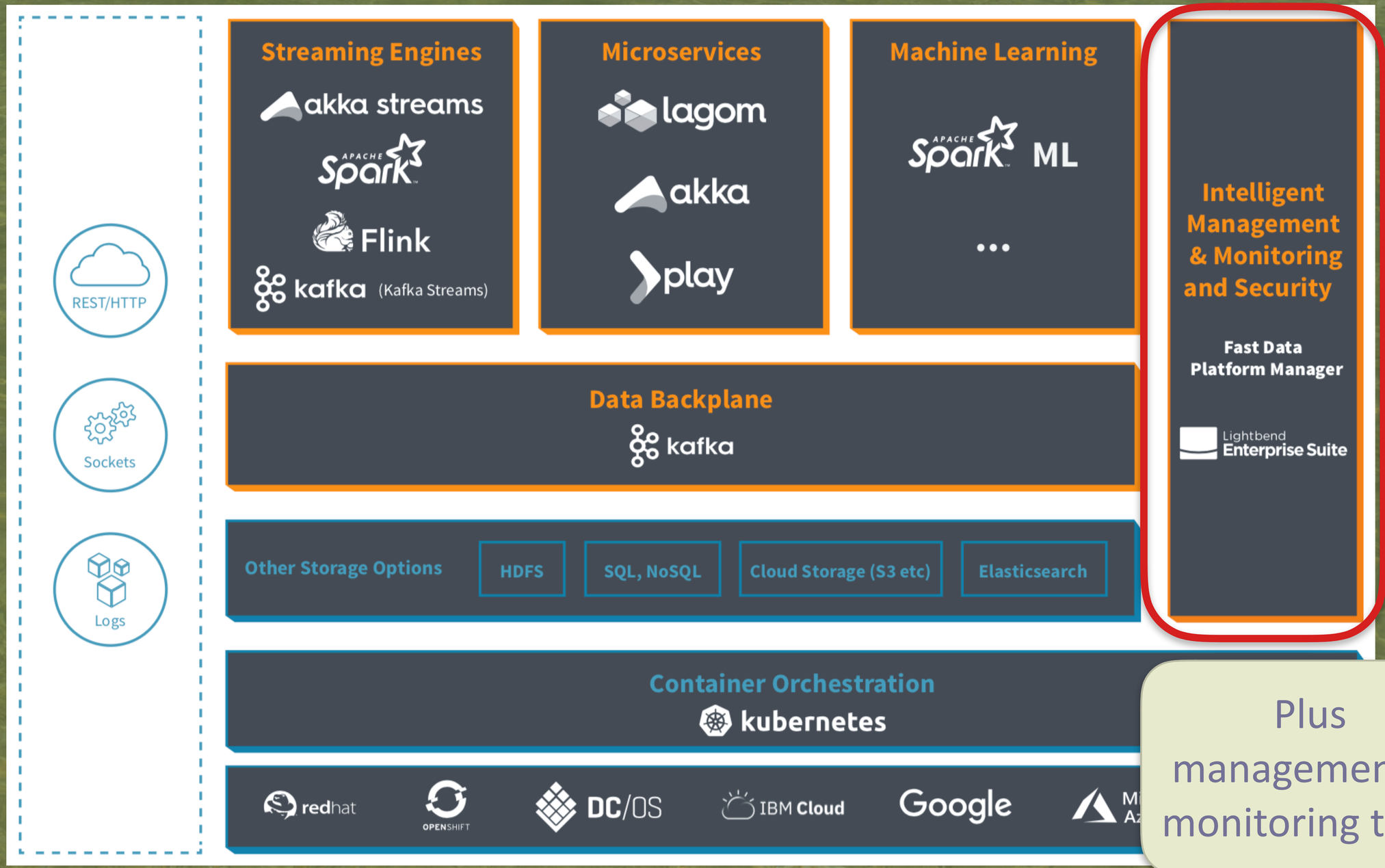


lightbend.com/fast-data-platform



What we discussed

lightbend.com/fast-data-platform



Plus management & monitoring tools

lightbend.com/fast-data-platform


Lightbend
Questions?
Dean Wampler, Ph.D.
dean@lightbend.com
@deanwampler
polyglotprogramming.com/talks
lightbend.com/fast-data-platform