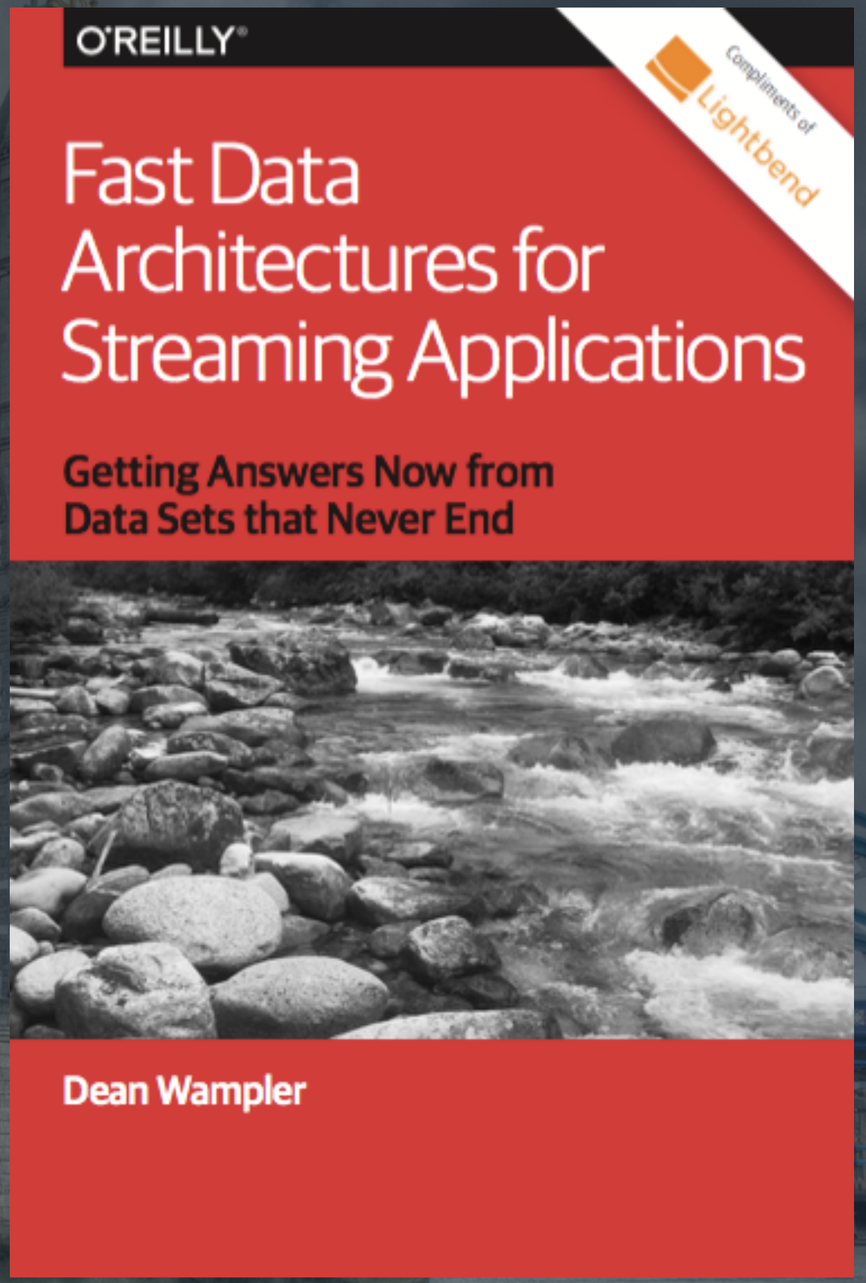
Executive Briefing: What You Need to Know about Fast Data

Dean Wampler, Ph.D.
dean@lightbend.com
@deanwampler
polyglotprogramming.com/talks

Lightbend

Based on this report

lightbend.com/fast-data-platform



What We'll Discuss

- Why streaming? Why now?
- How should I choose technologies?
- How will this impact the rest of my organization?

What We'll Discuss

Why Streaming?

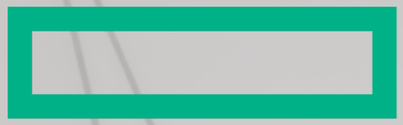
- New opportunities that require streaming
- Upgrading batch applications for competitive advantage

Why Streaming?

Fast Data Use Cases

Predictive Analytics	IoT	Real-time Personalization	Real-time Financial Processes	Legacy Modernization
Apply ML models to large volumes of device data to pre-empt failures / outages	Real-time consumer and industrial Device and Supply Chain management at scale	Real-time marketing based on behavior, location, inventory levels, product promotions, etc.	Drive better business outcomes through real-time risk, fraud detection, compliance, audit, governance, etc.	Accelerate decision making processes and optimize infrastructure costs by moving from batch to streaming
Hewlett Packard Enterprise	STARBUCKS COFFEE	Royal Caribbean International	Capital One	IQVIA

More at: https://www.lightbend.com/customers



Predictive Analytics

Hewlett Packard
Enterprise

- ML models applied to device telemetry to detect anomalies
- Preemptive maintenance prevents user impacts from potential failures

Core Idea

Handle anomaly: move activity off component, schedule maintenance window to replace it.

Anomaly Handler

Probable Anomalies

Train models to look for anomalies… and score incoming telemetry.

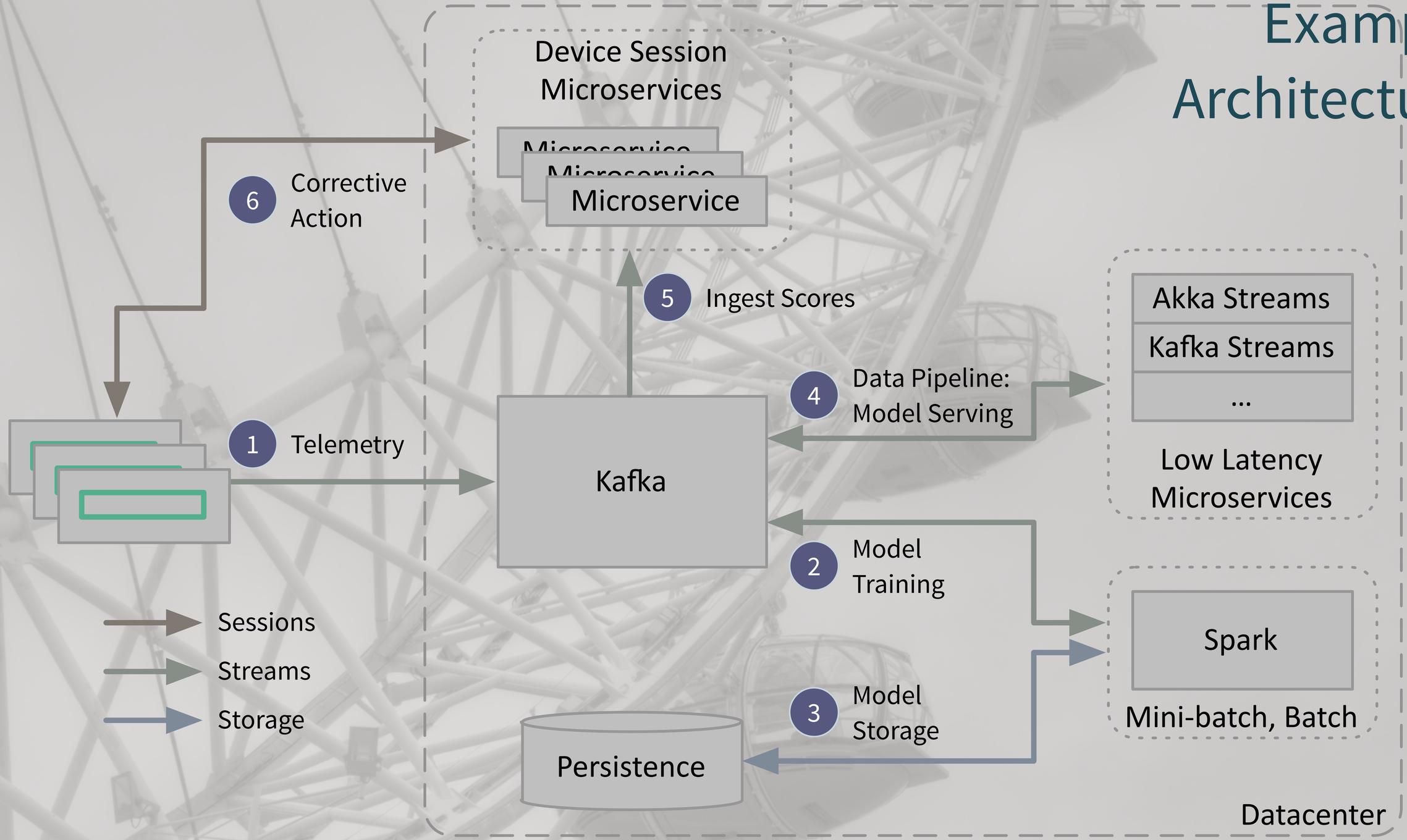
Anomaly Detection: Model

Corrective Actions

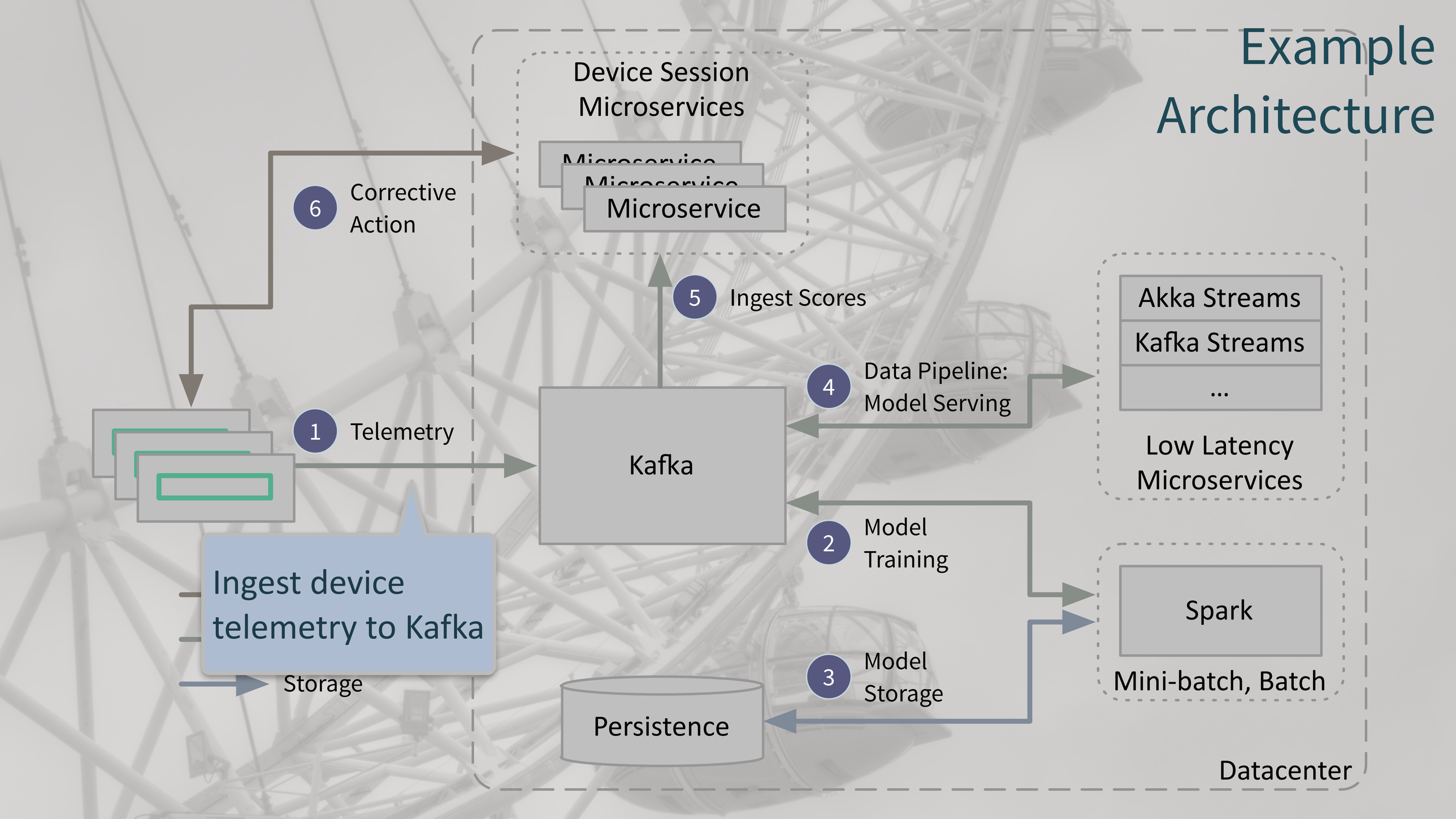
Telemetry Records

Ingest telemetry from edge devices.

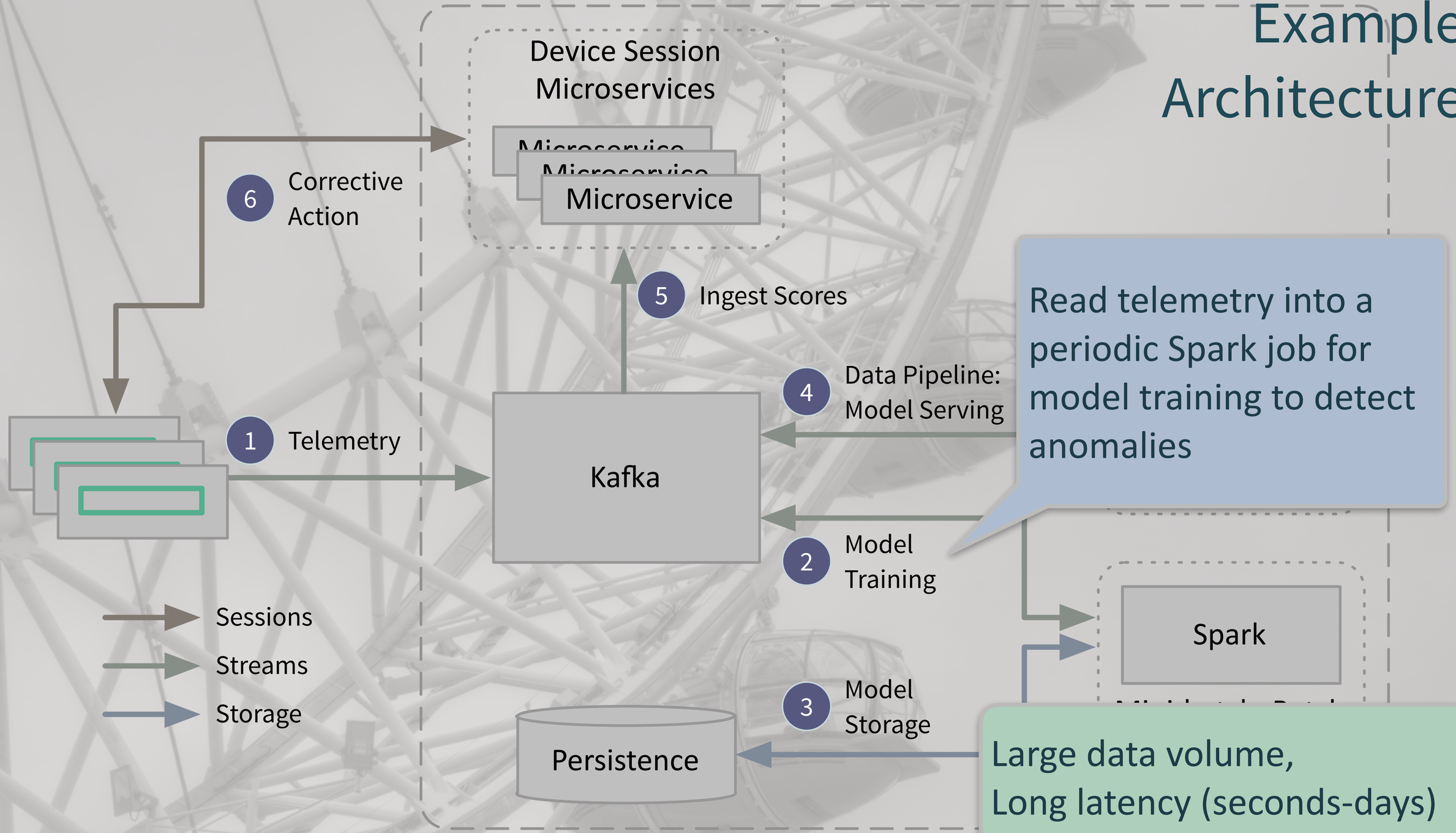
Example Architecture



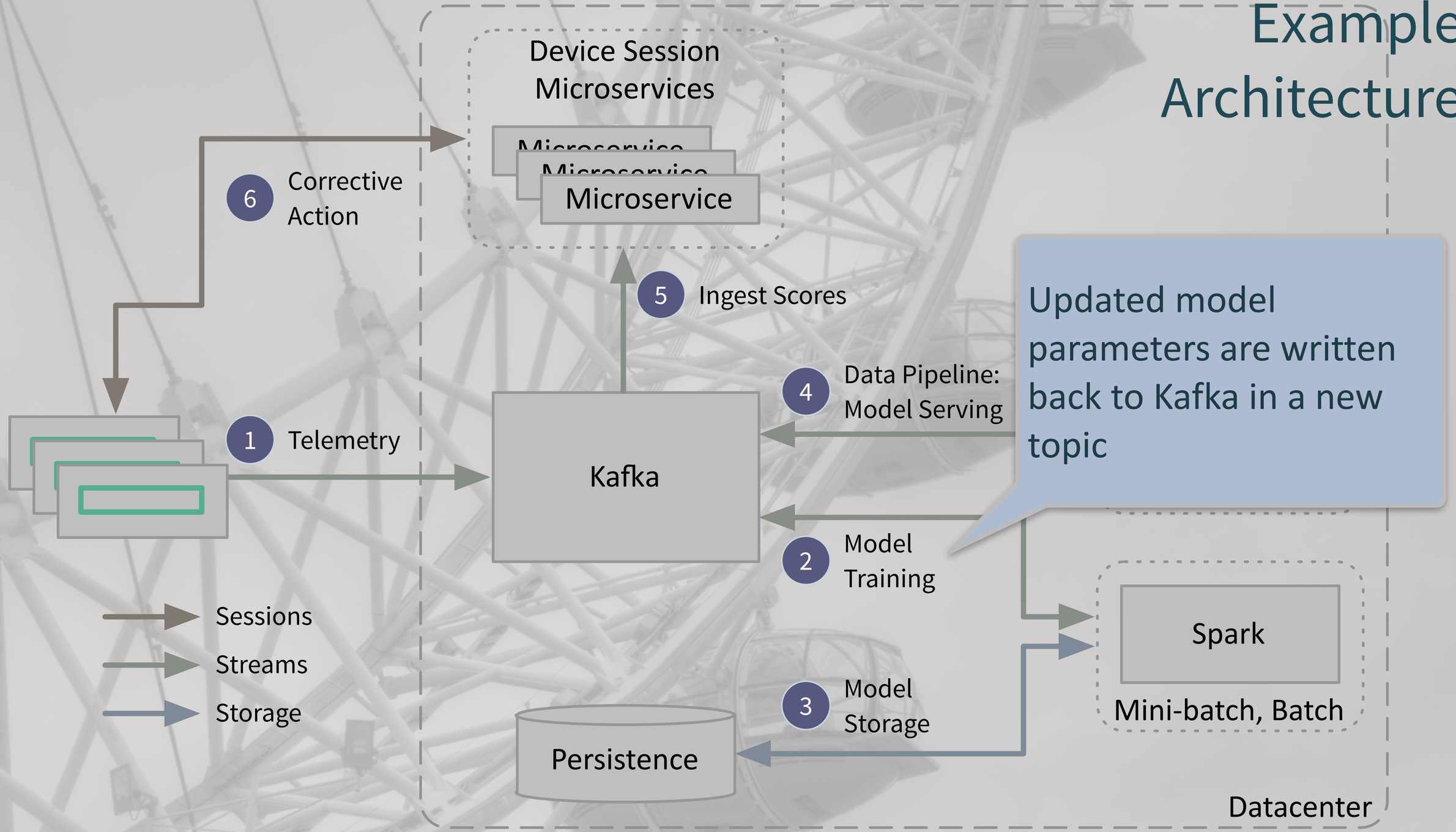
Example Architecture



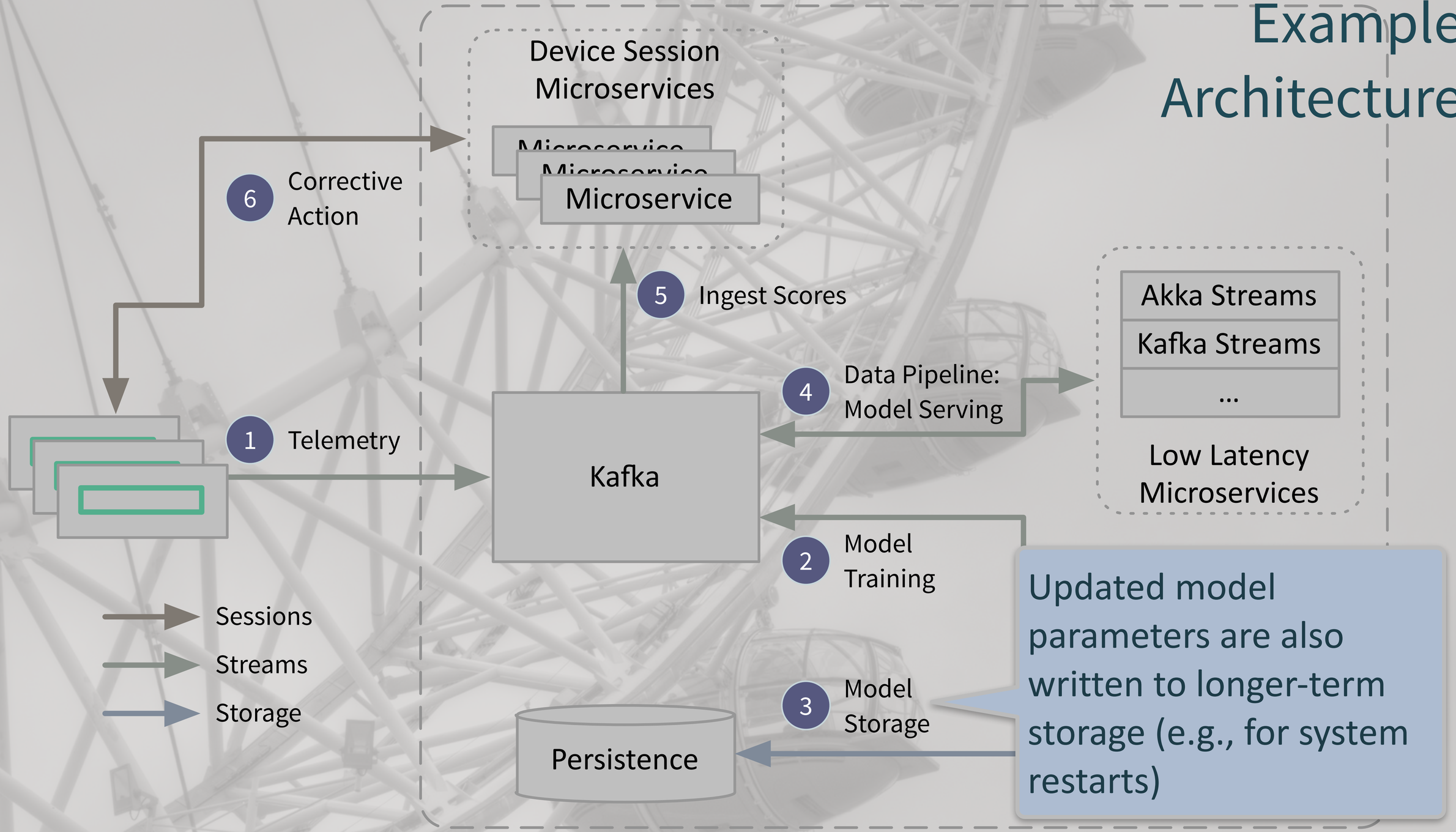
Example Architecture



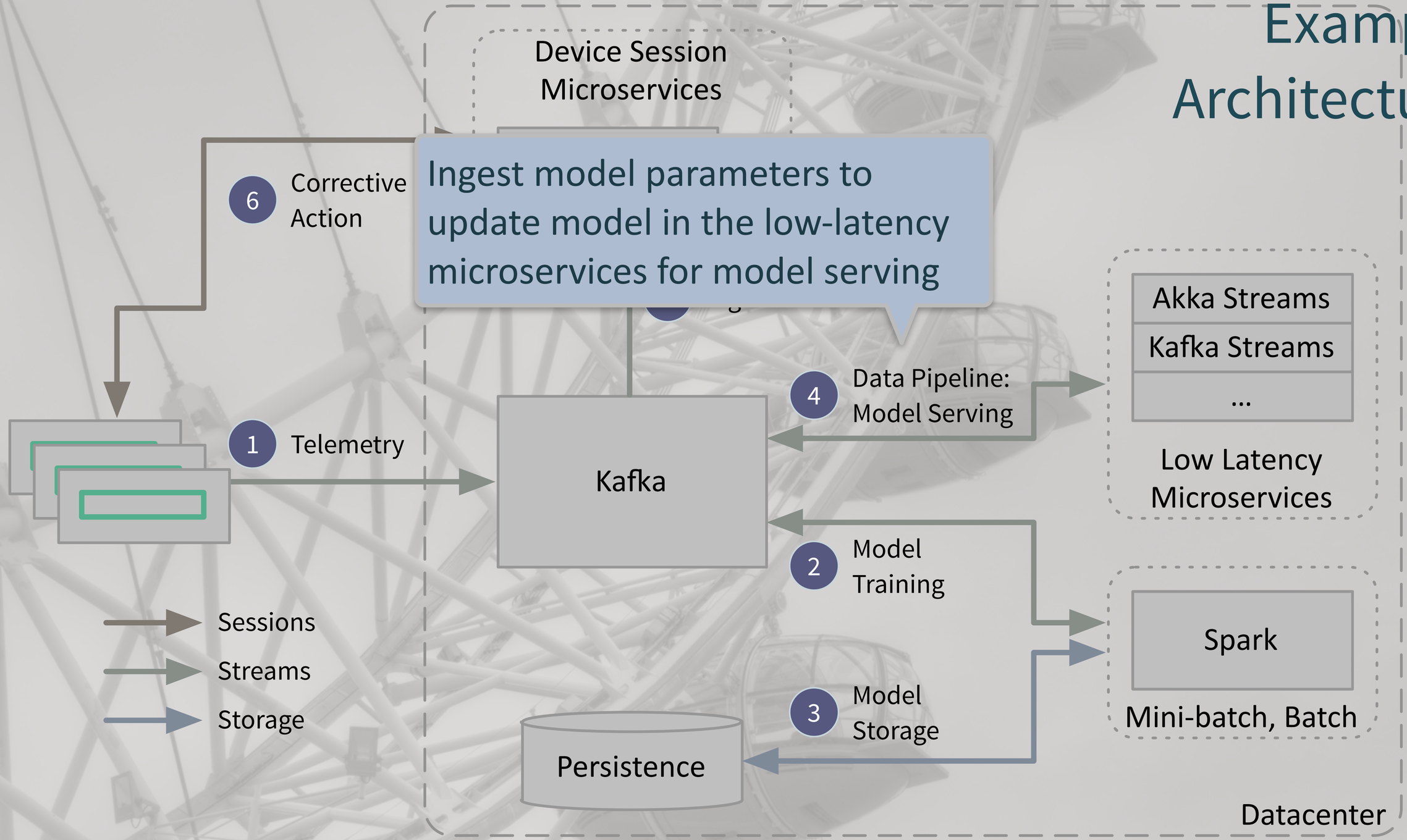
Example Architecture

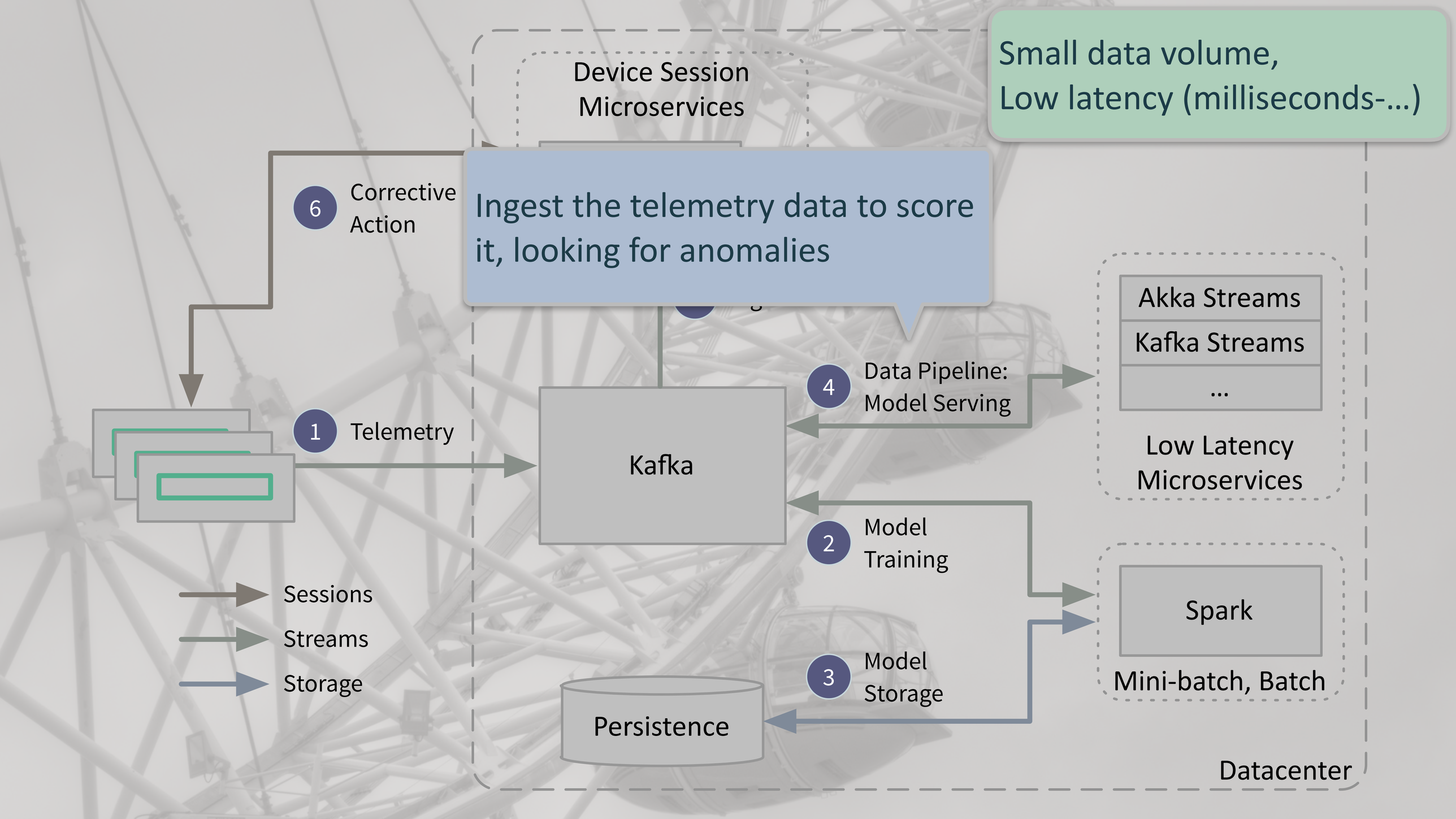


Example Architecture

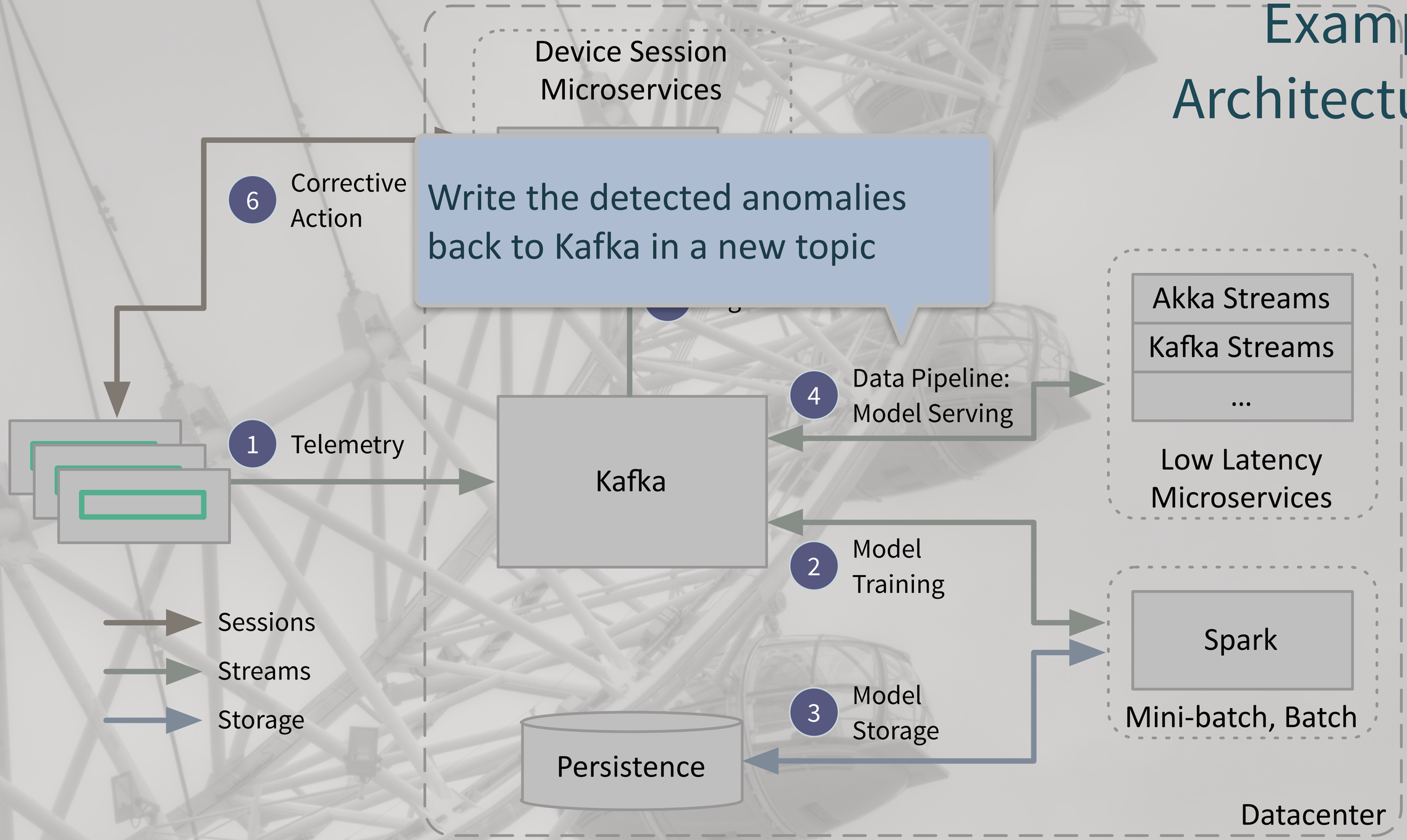


Example Architecture

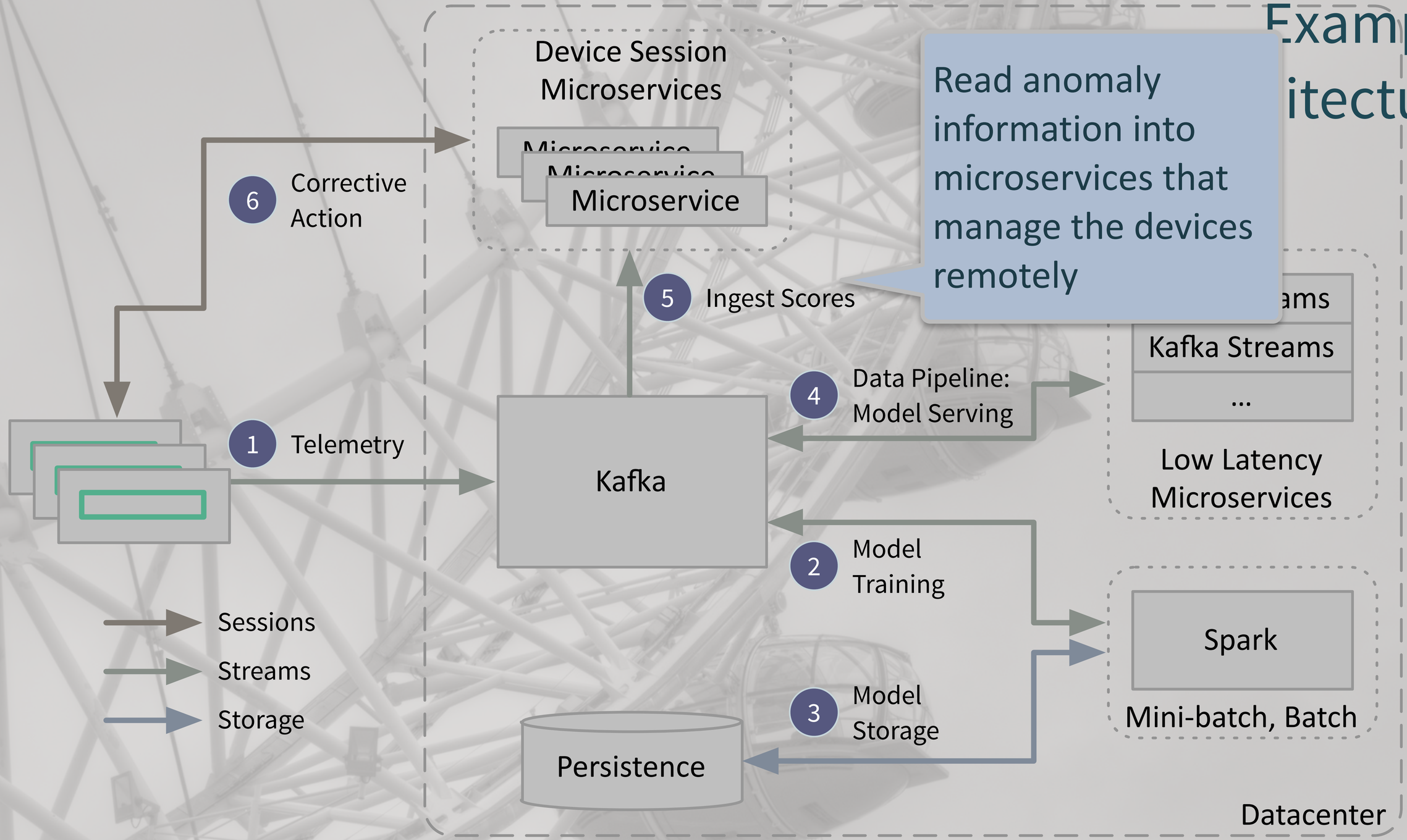



Small data volume,
Low latency (milliseconds-...)
Device Session
Microservices
Ingest the telemetry data to score it, looking for anomalies
6
Corrective
Action
1
Telemetry
Kafka
4
Data Pipeline:
Model Serving
Akka Streams
Kafka Streams
...
Low Latency
Microservices
2
Model
Training
Spark
Mini-batch, Batch
3
Model
Storage
Persistence
Sessions
Streams
Storage
Datacenter

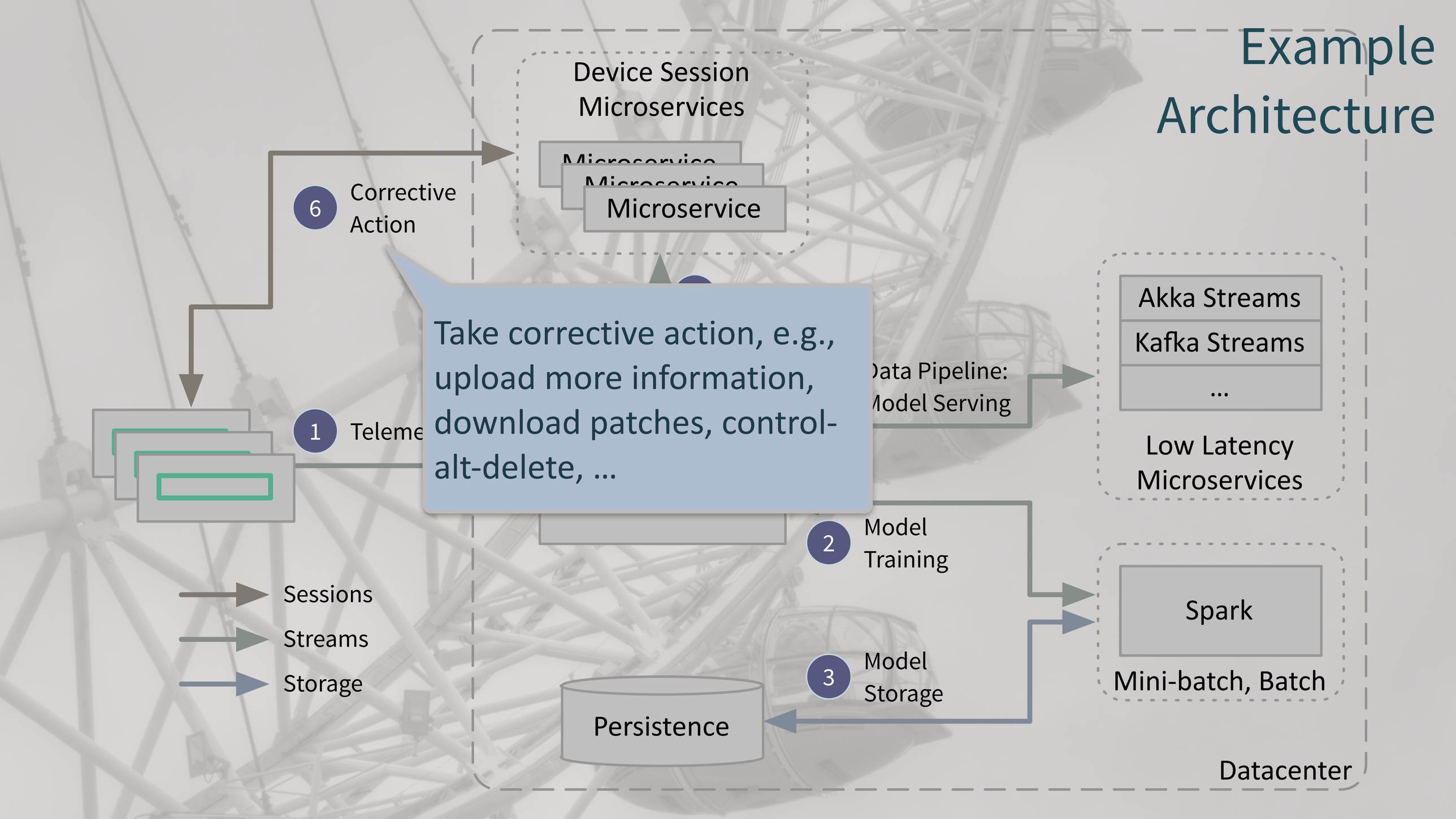
Example Architecture



Example itecture

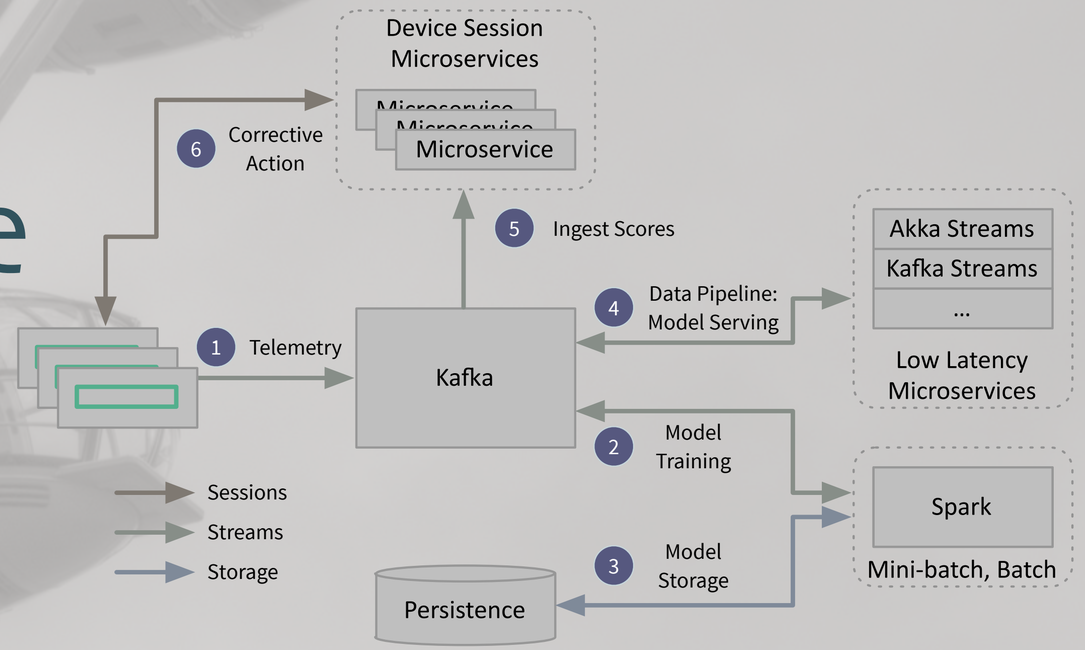


Example Architecture



Challenges

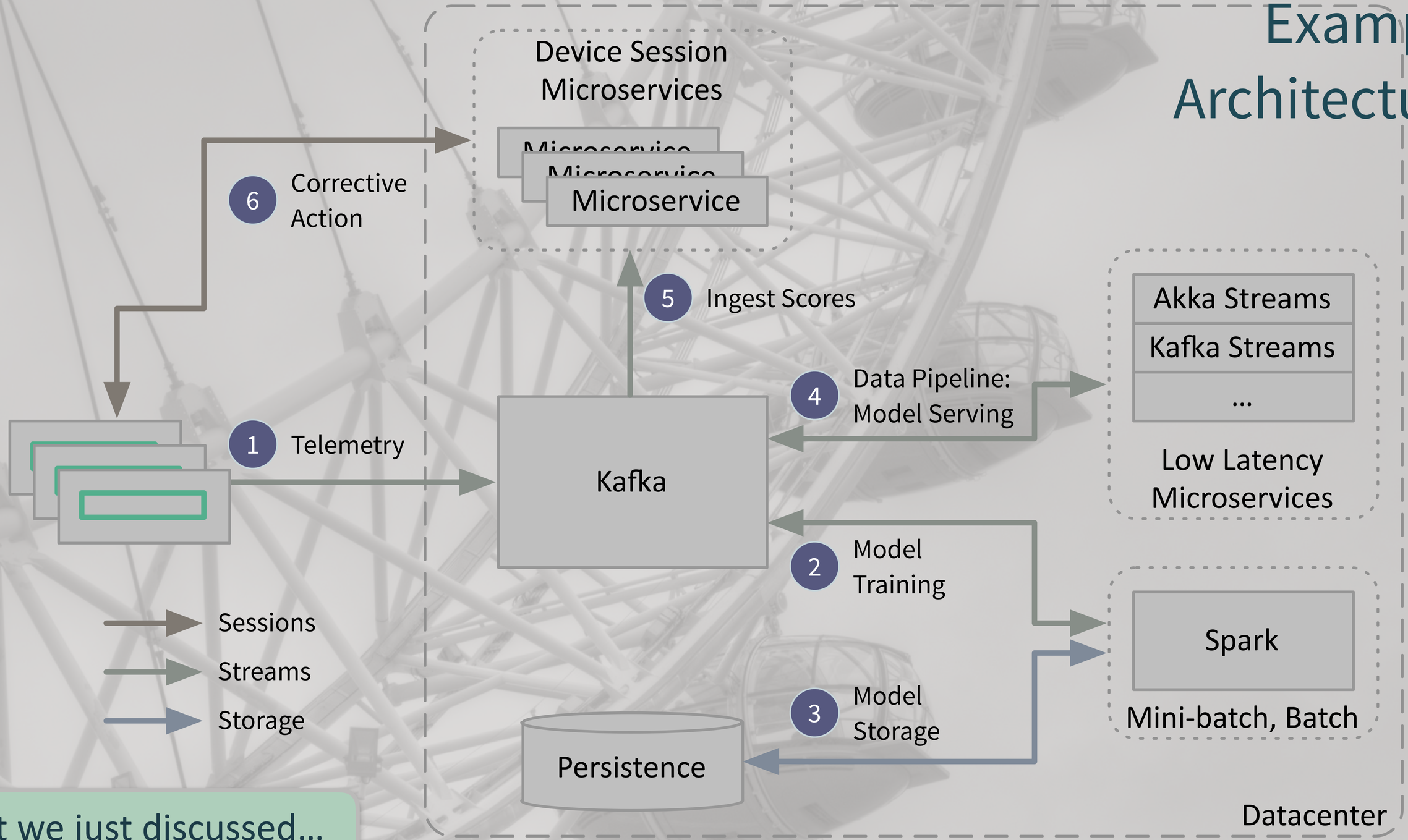
- Network overhead for telemetry ingestion too high?
- Model serving latency too long?
- Idea: Serve model on the device!



Internet of Things

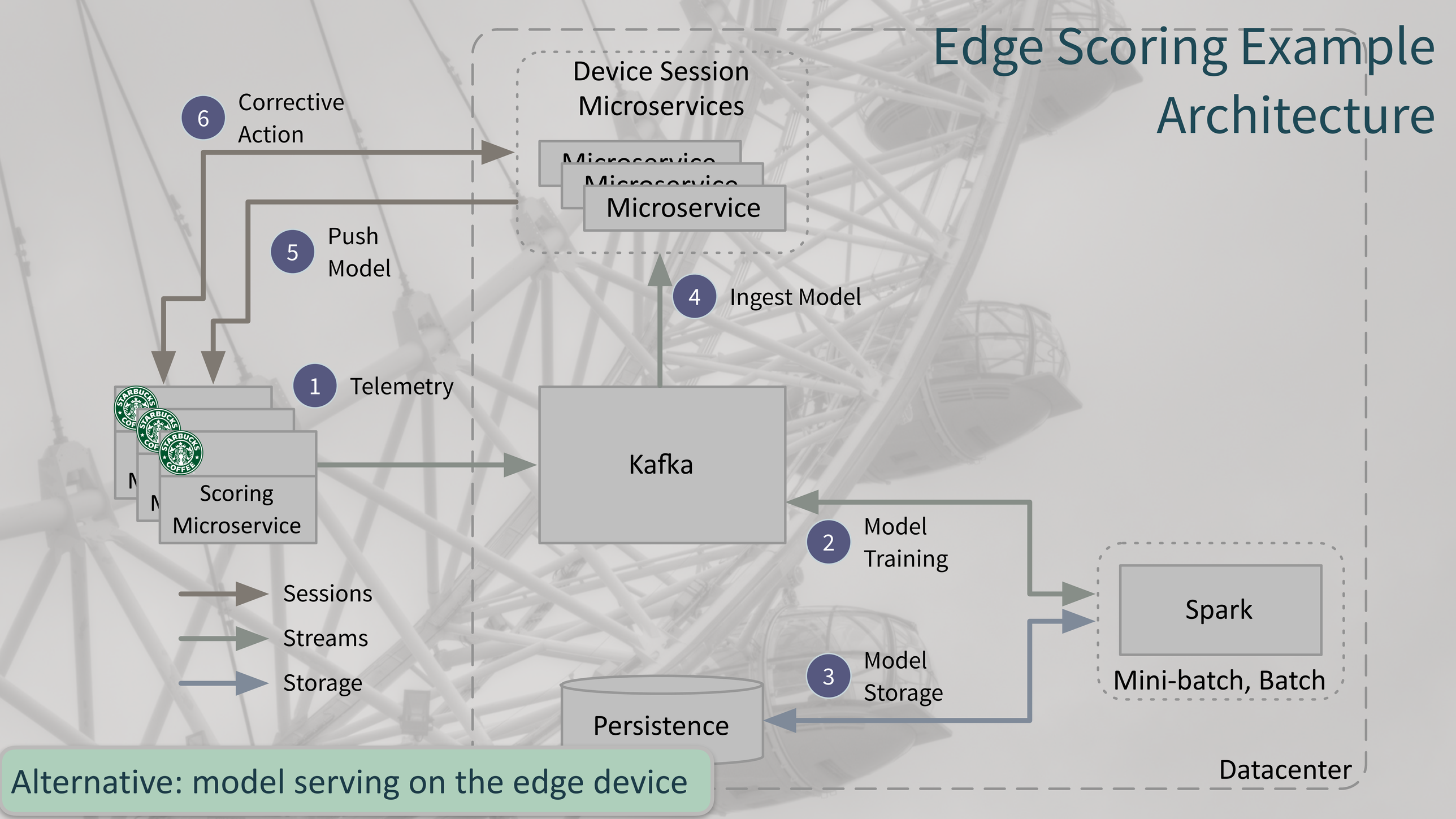
- Real-time consumer and industrial device and supply chain management at scale

Example Architecture



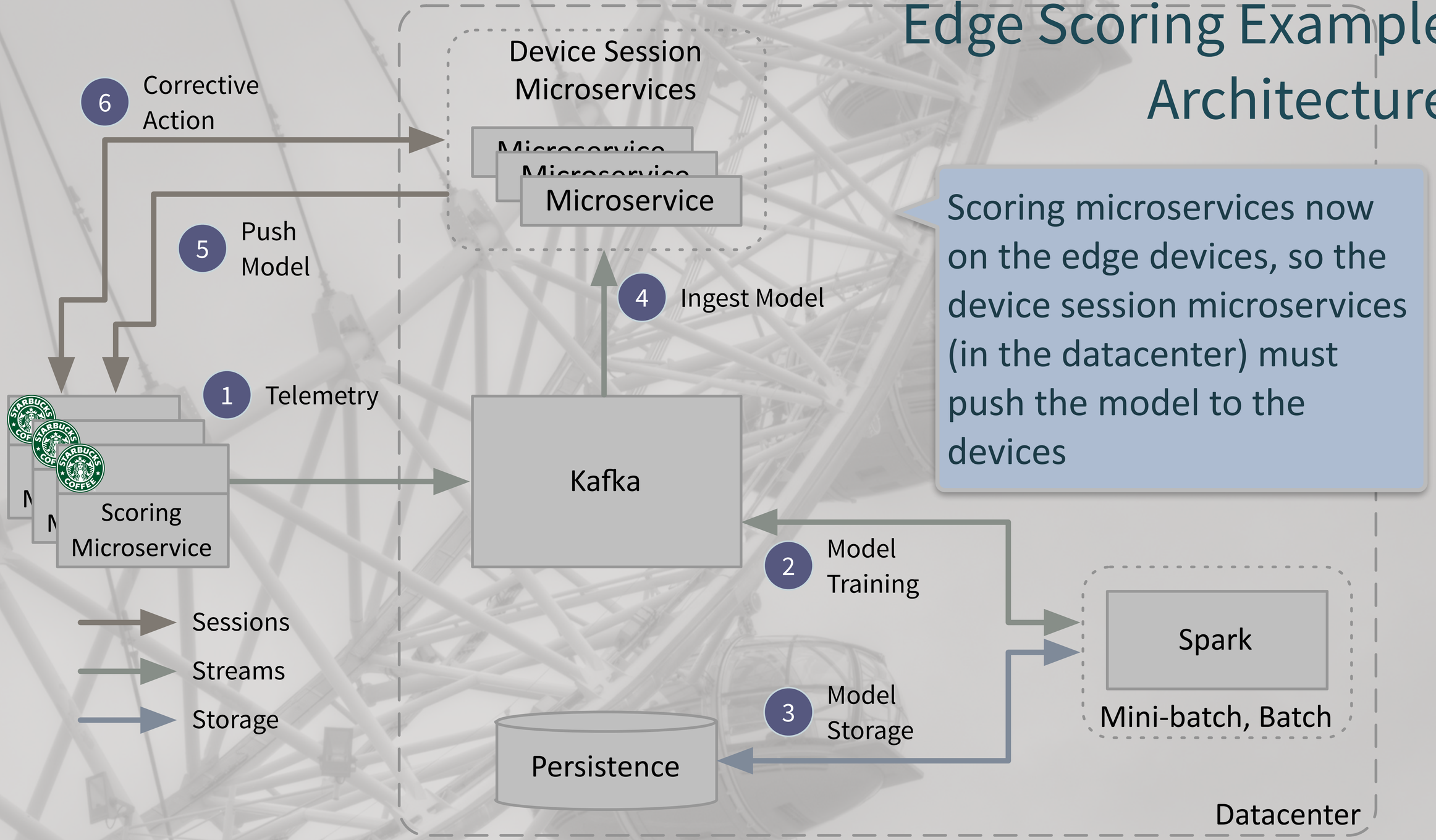
What we just discussed...

Edge Scoring Example Architecture



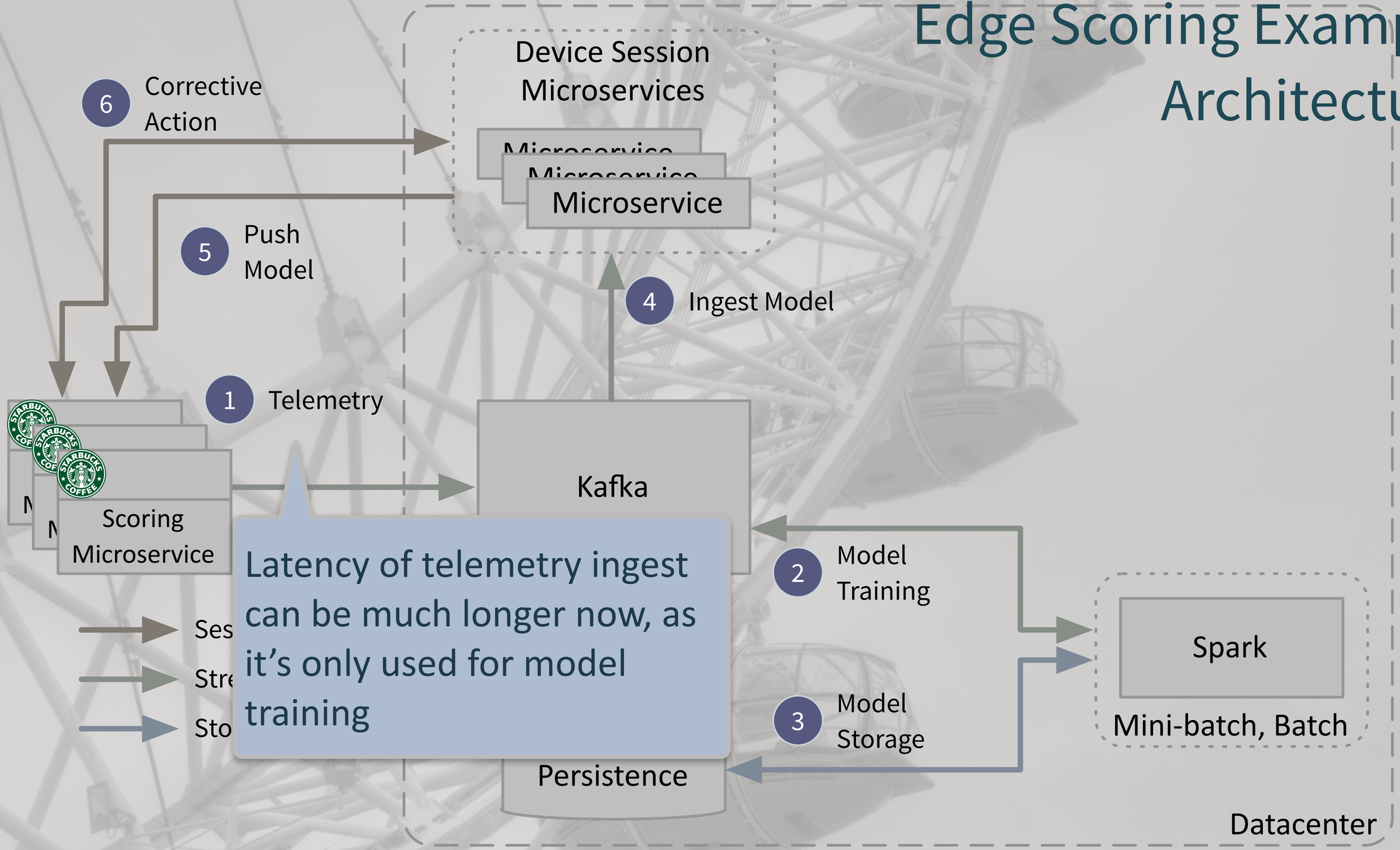
Alternative: model serving on the edge device

Edge Scoring Example Architecture



Scoring microservices now on the edge devices, so the device session microservices (in the datacenter) must push the model to the devices

Edge Scoring Example Architecture



Real-time Personalization

- Model users to provide real-time marketing based on behavior, location, inventory levels, product promotions, etc.

A few more examples. Similar architectures…

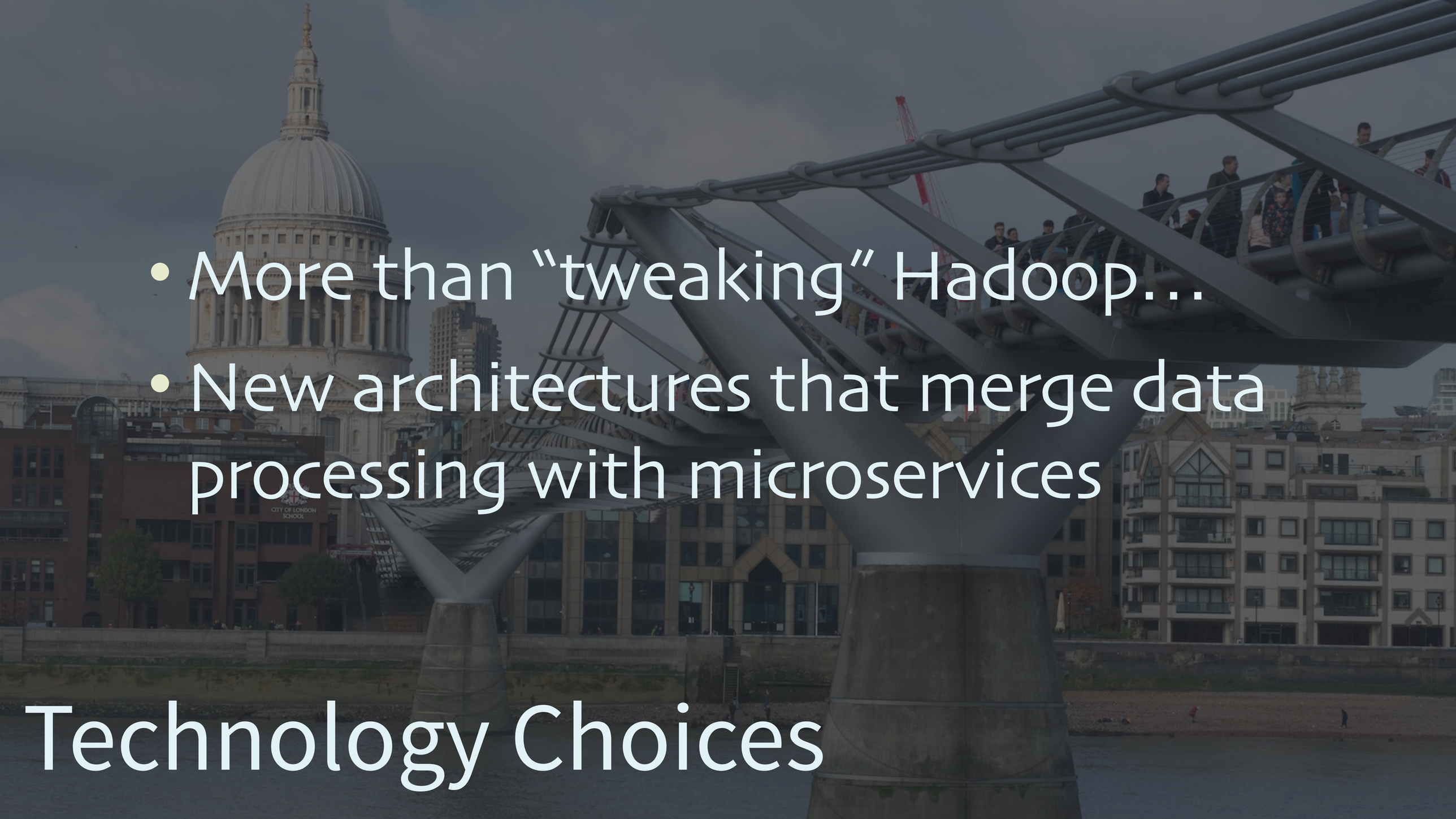
Real-time Financial

- Drive better business outcomes through real-time risk, fraud detection, compliance, audit, governance, etc.

Legacy Modernization

- Accelerate decision making processes and optimize infrastructure costs by moving from batch to streaming
- Hadoop replacement

Technology Choices

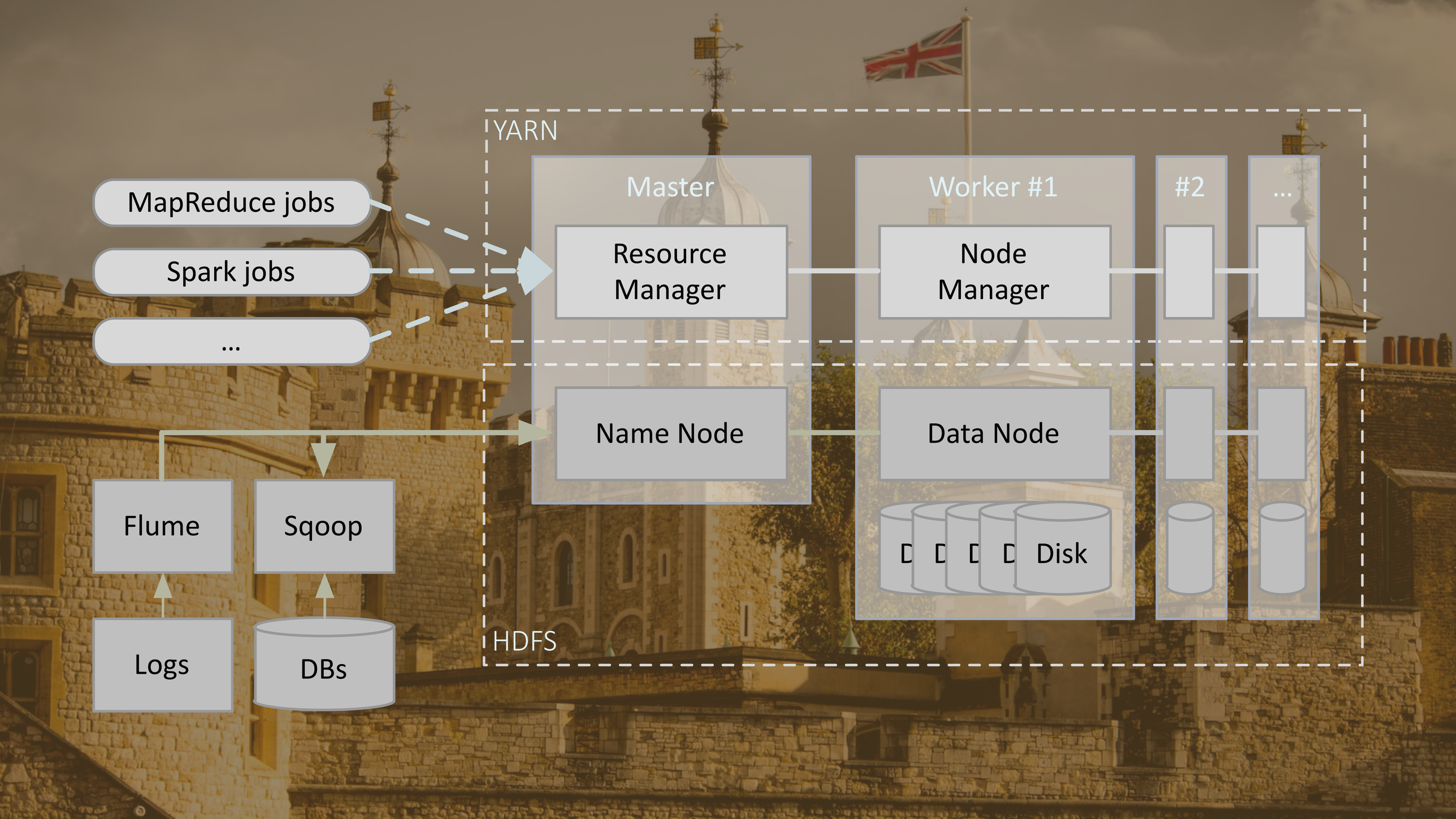


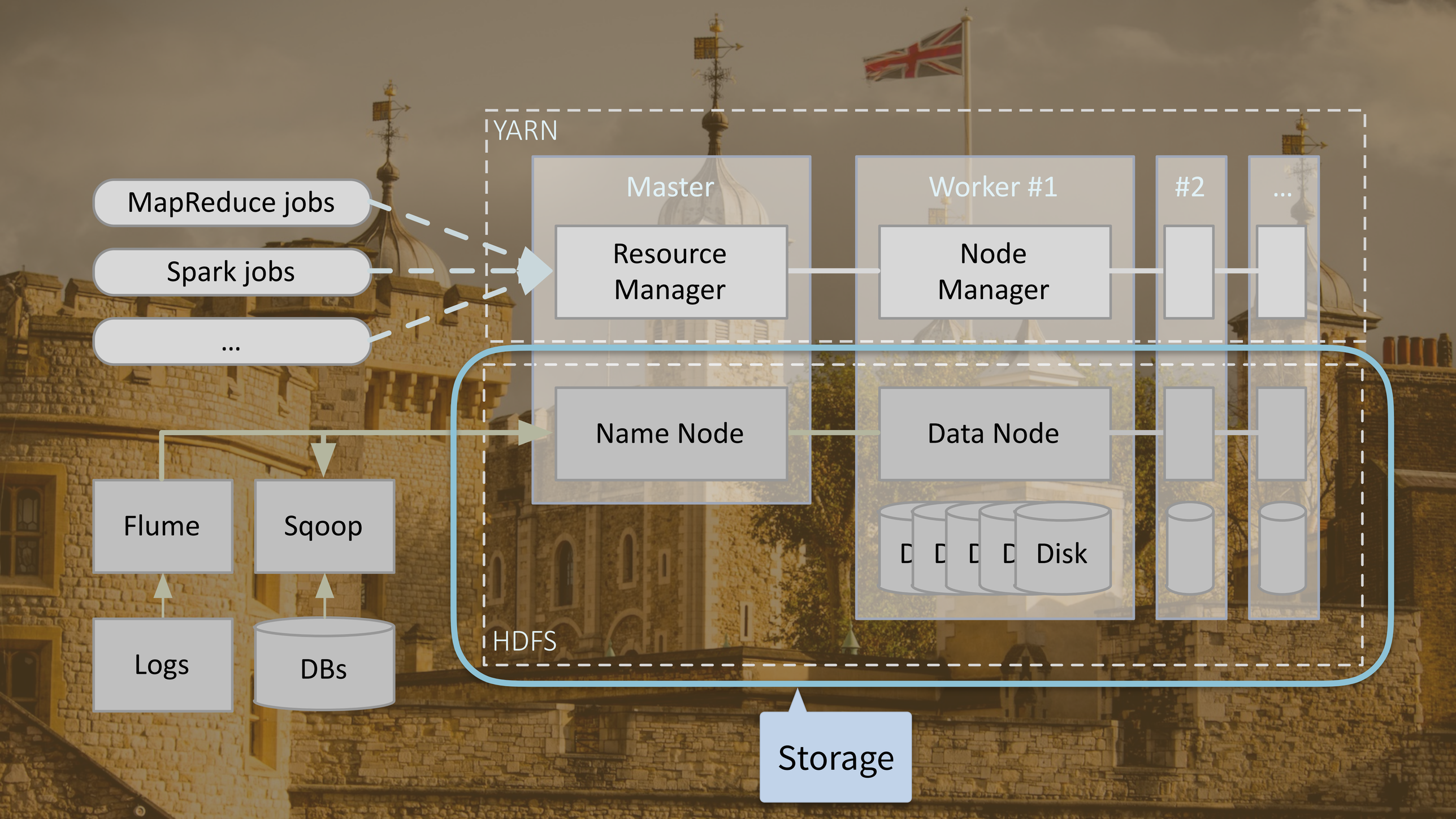
- More than "tweaking" Hadoop…
- New architectures that merge data processing with microservices

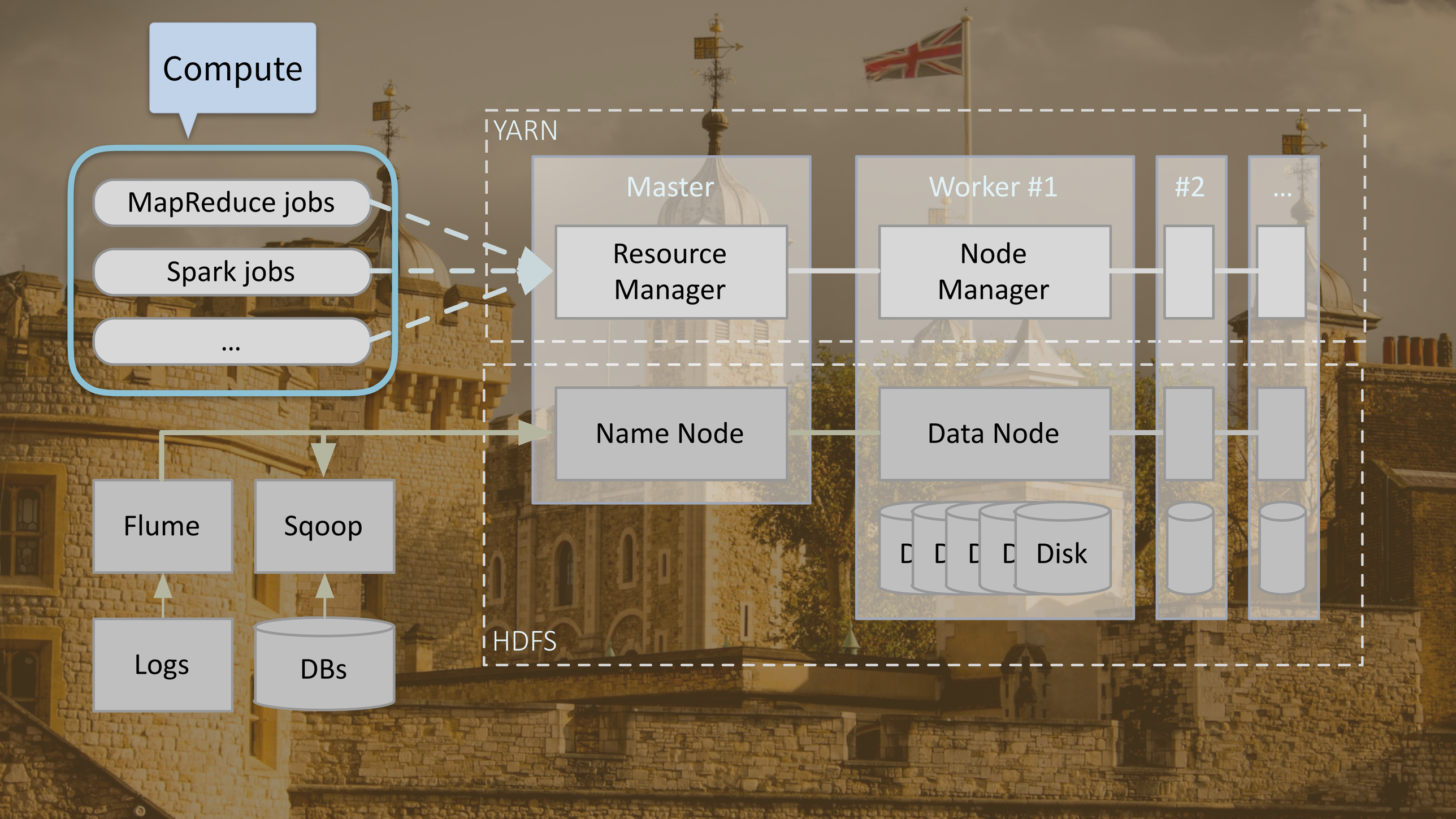
Technology Choices

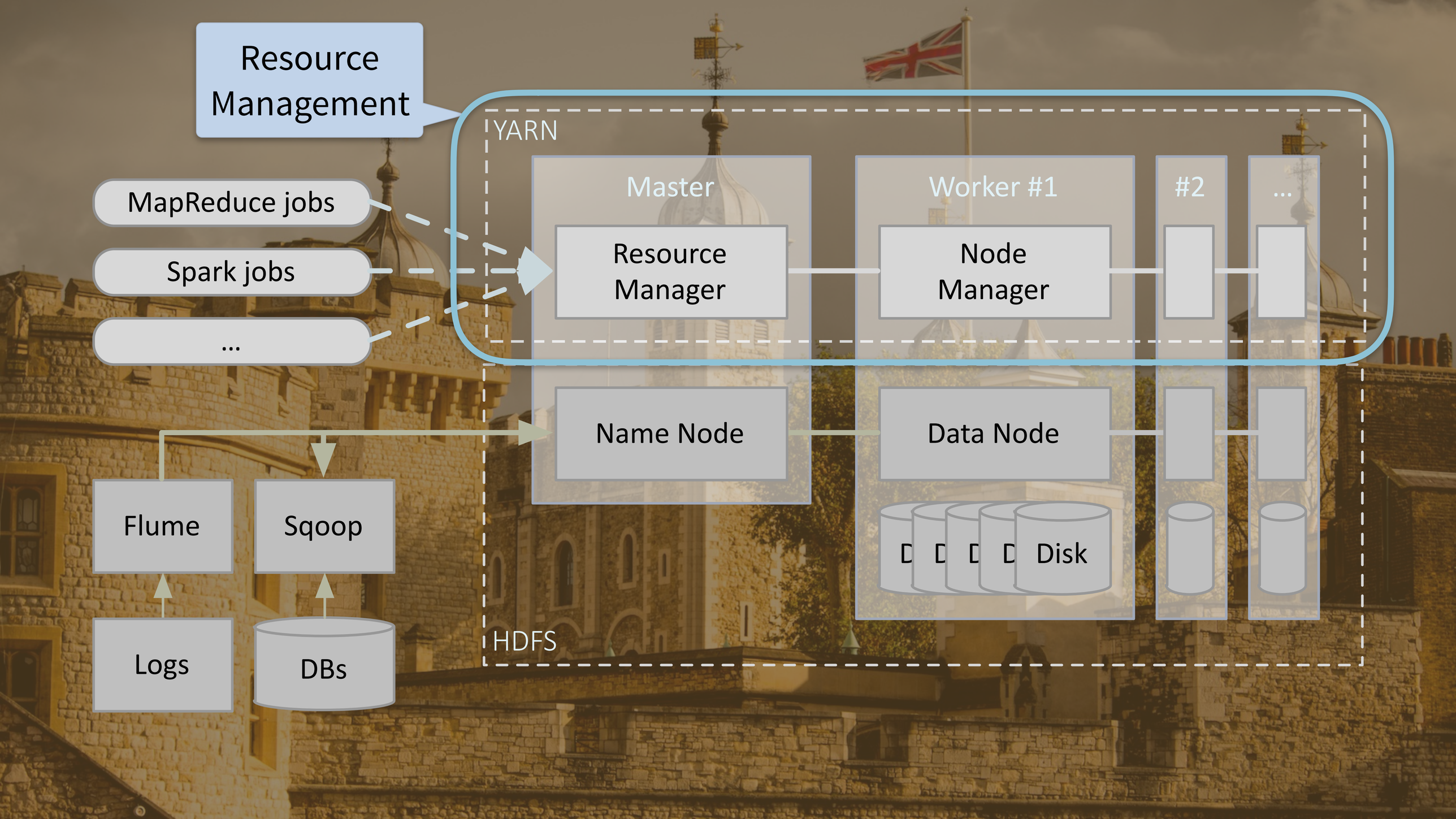
Recall Hadoop…

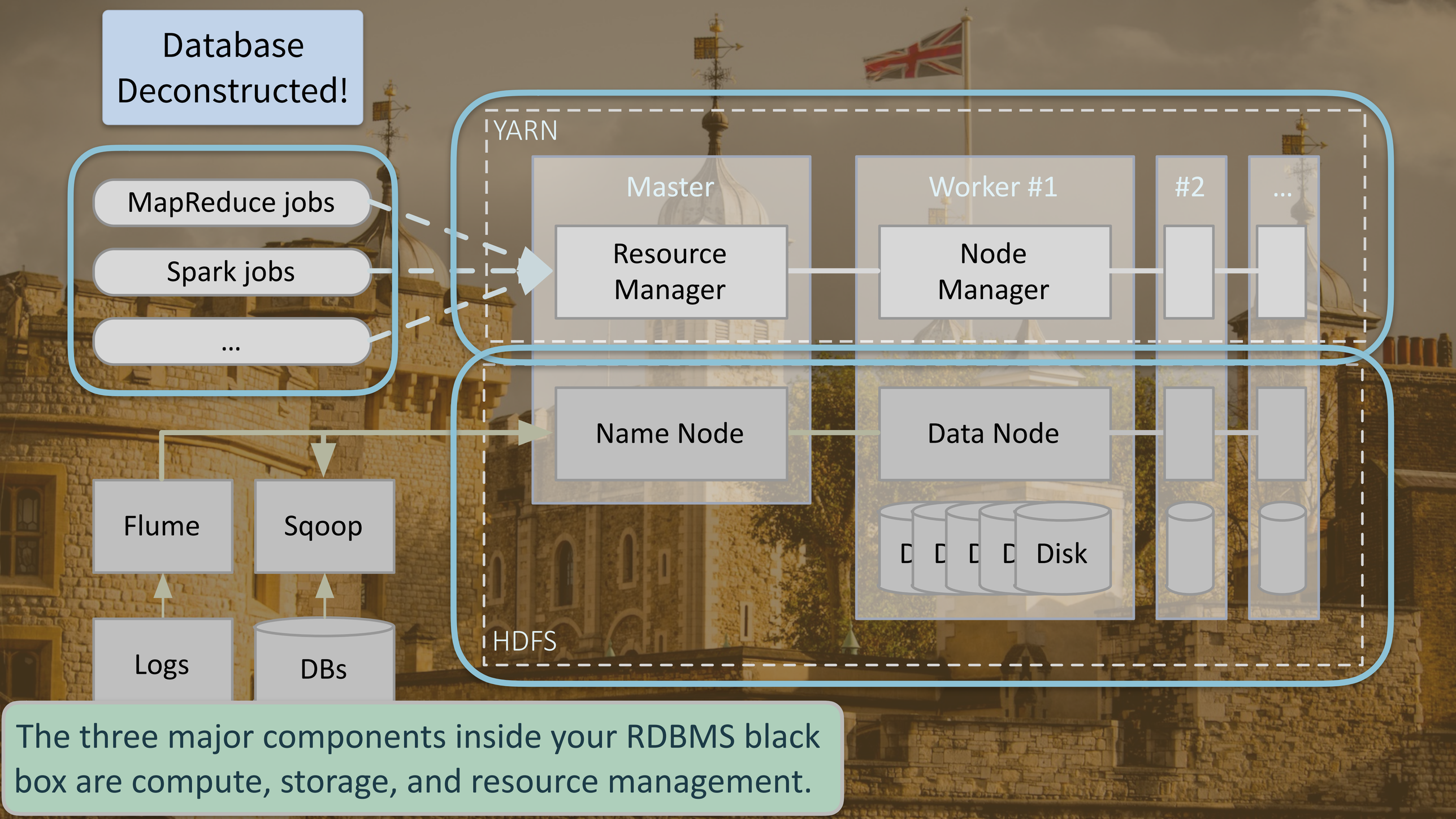
- Data warehouse replacement
- Historical analysis
- Interactive exploration
- Offline training of machine learning models
- …

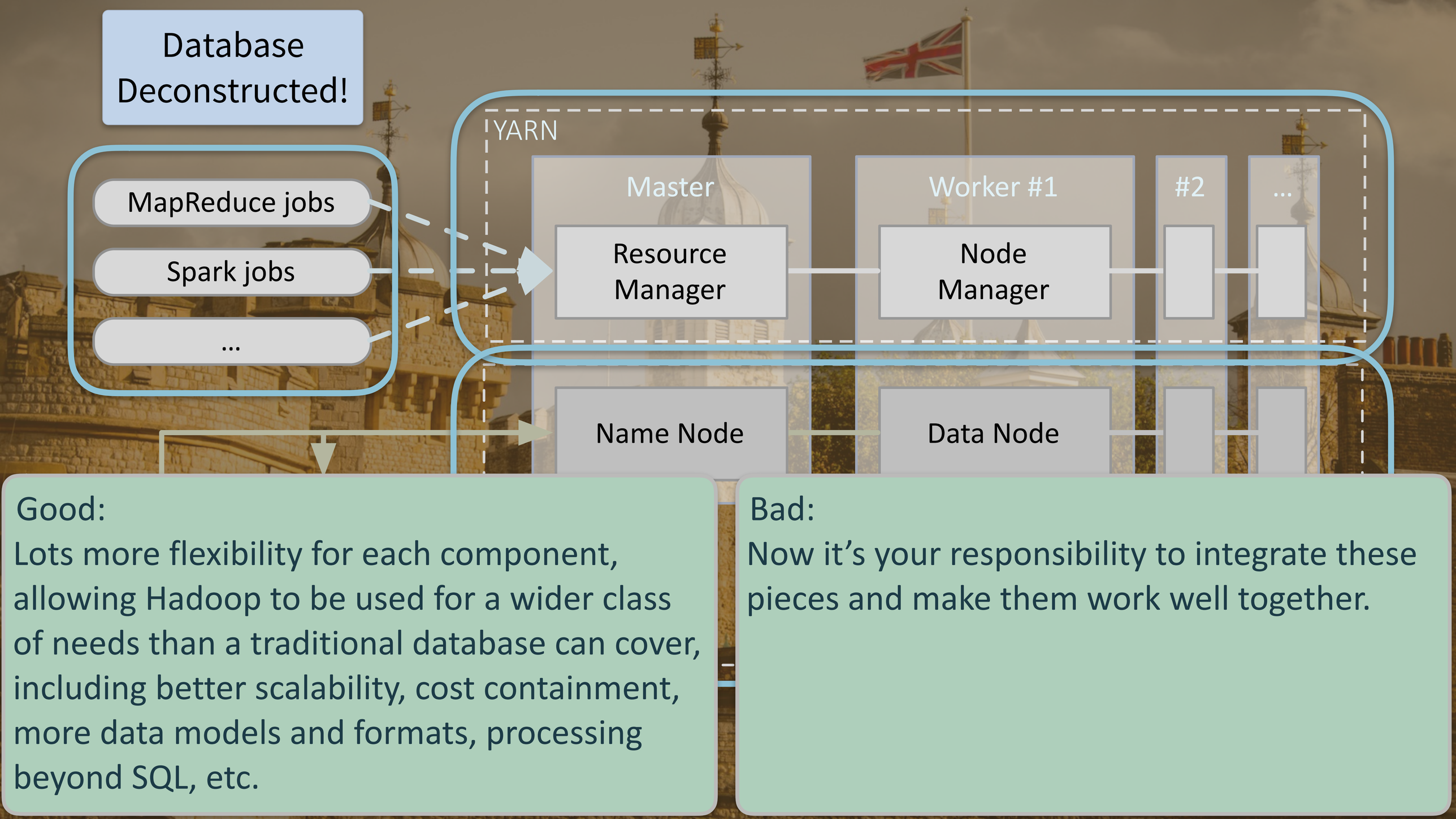

YARN
Master
Worker #1
#2
...
MapReduce jobs
Spark jobs
...
Resource Manager
Node Manager
Name Node
Data Node
Disk
Flume
Sqoop
Logs
DBs
HDFS


YARN
Master
Worker #1
#2
...
MapReduce jobs
Spark jobs
...
Resource Manager
Node Manager
Name Node
Data Node
Disk
Flume
Sqoop
Logs
DBs
HDFS
Storage


Compute
MapReduce jobs
Spark jobs
...
YARN
Master
Worker #1
#2
...
Resource Manager
Node Manager
Name Node
Data Node
Disk
HDFS
Flume
Sqoop
Logs
DBs

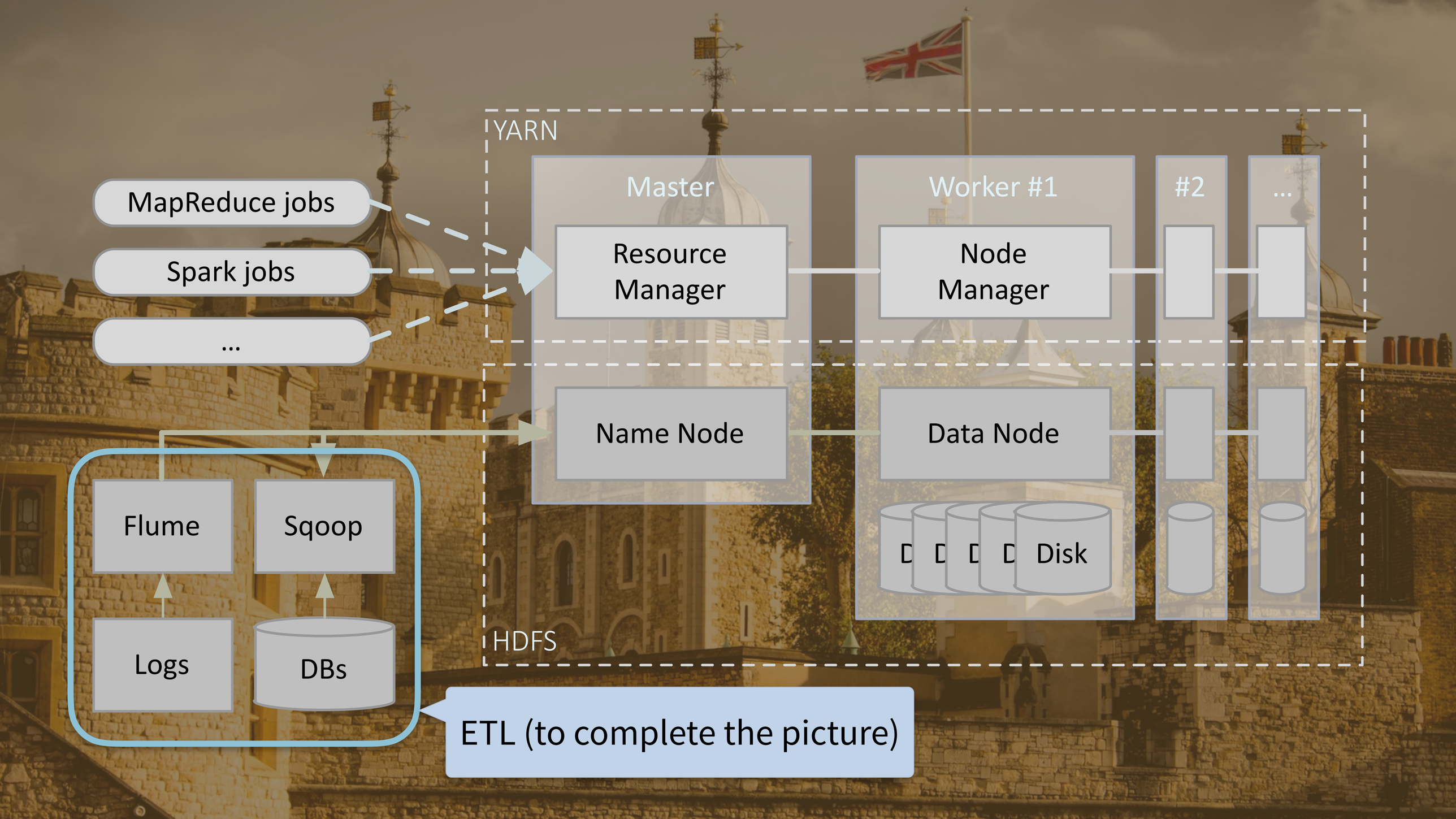

Resource Management
YARN
Master
Worker #1
#2
...
MapReduce jobs
Spark jobs
...
Resource Manager
Node Manager
Name Node
Data Node
Disk
Flume
Sqoop
Logs
DBs
HDFS


Database Deconstructed!
MapReduce jobs
Spark jobs
...
YARN
Master
Worker #1
#2
...
Resource Manager
Node Manager
Name Node
Data Node
Disk
HDFS
Flume
Sqoop
Logs
DBs
The three major components inside your RDBMS black box are compute, storage, and resource management.

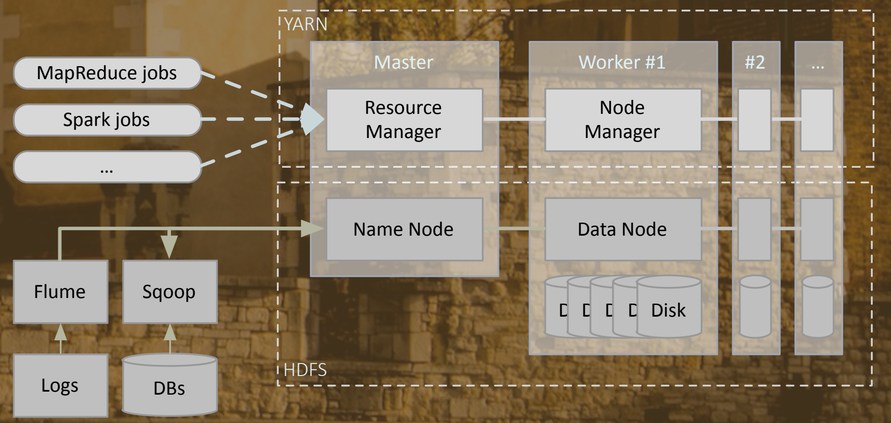


Good:
Lots more flexibility for each component, allowing Hadoop to be used for a wider class of needs than a traditional database can cover, including better scalability, cost containment, more data models and formats, processing beyond SQL, etc.

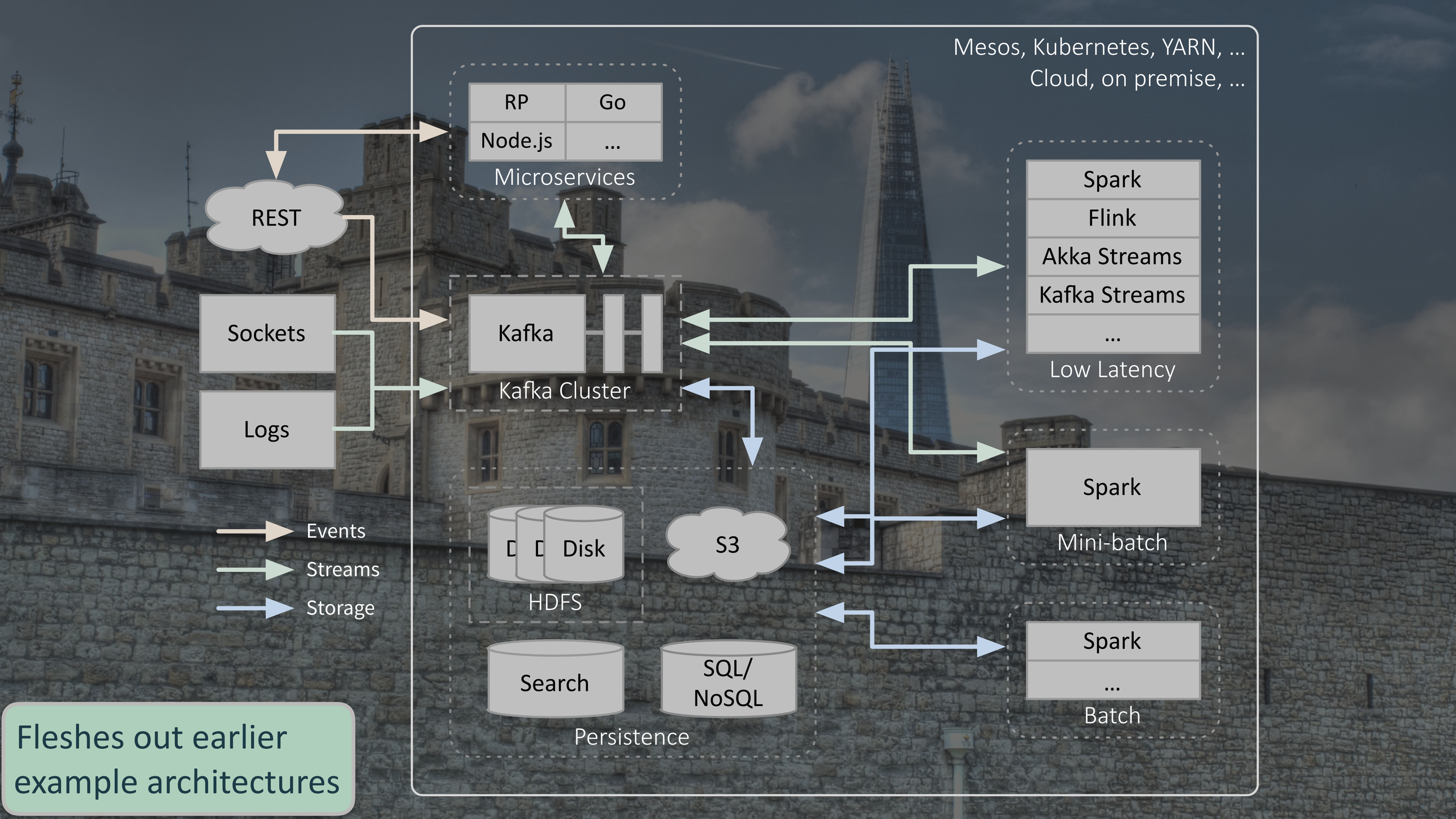
Bad:
Now it's your responsibility to integrate these pieces and make them work well together.

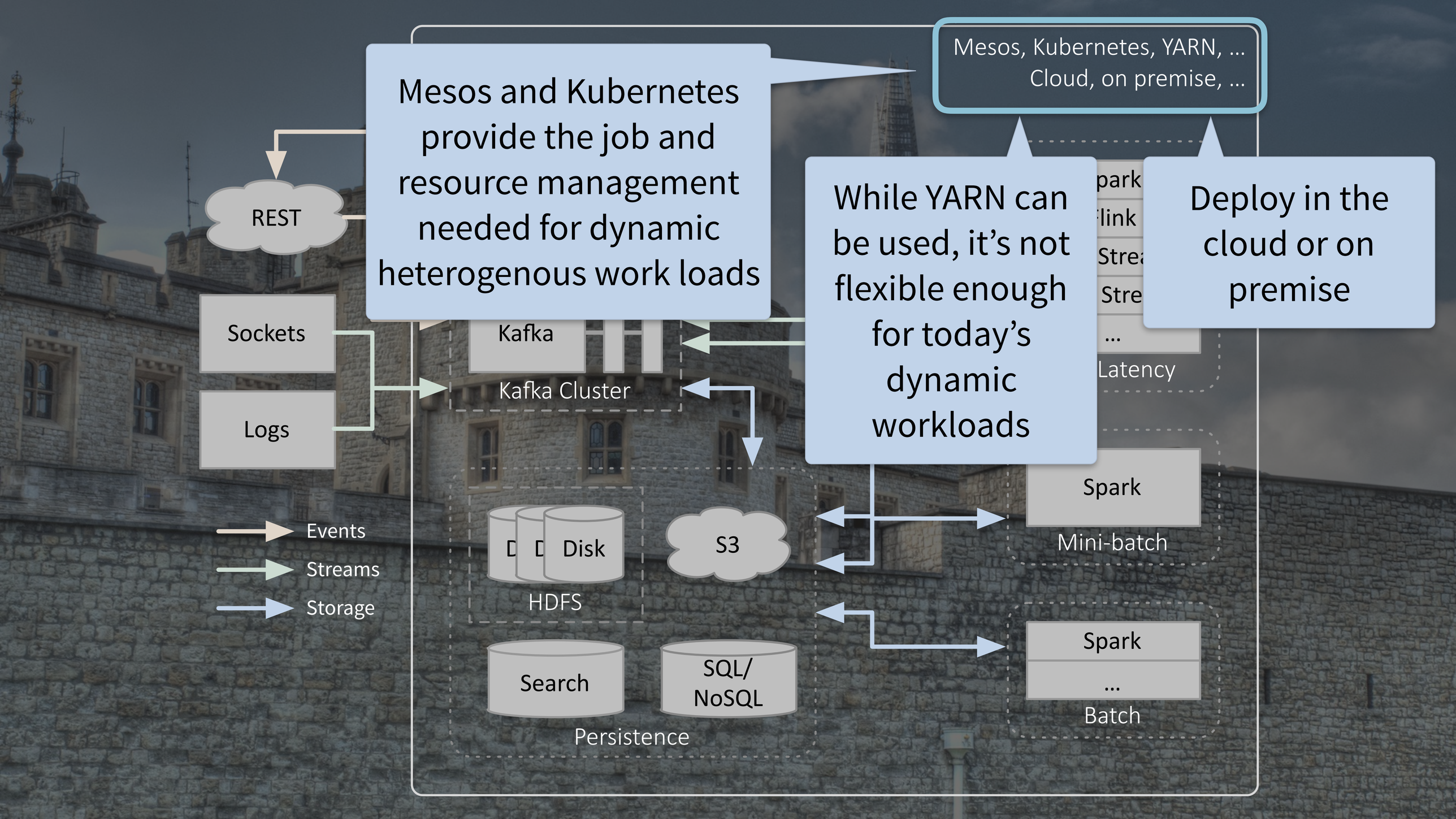

YARN
Master
Worker #1
#2
...
MapReduce jobs
Spark jobs
...
Resource Manager
Node Manager
Name Node
Data Node
Disk
HDFS
Flume
Sqoop
Logs
DBs
ETL (to complete the picture)

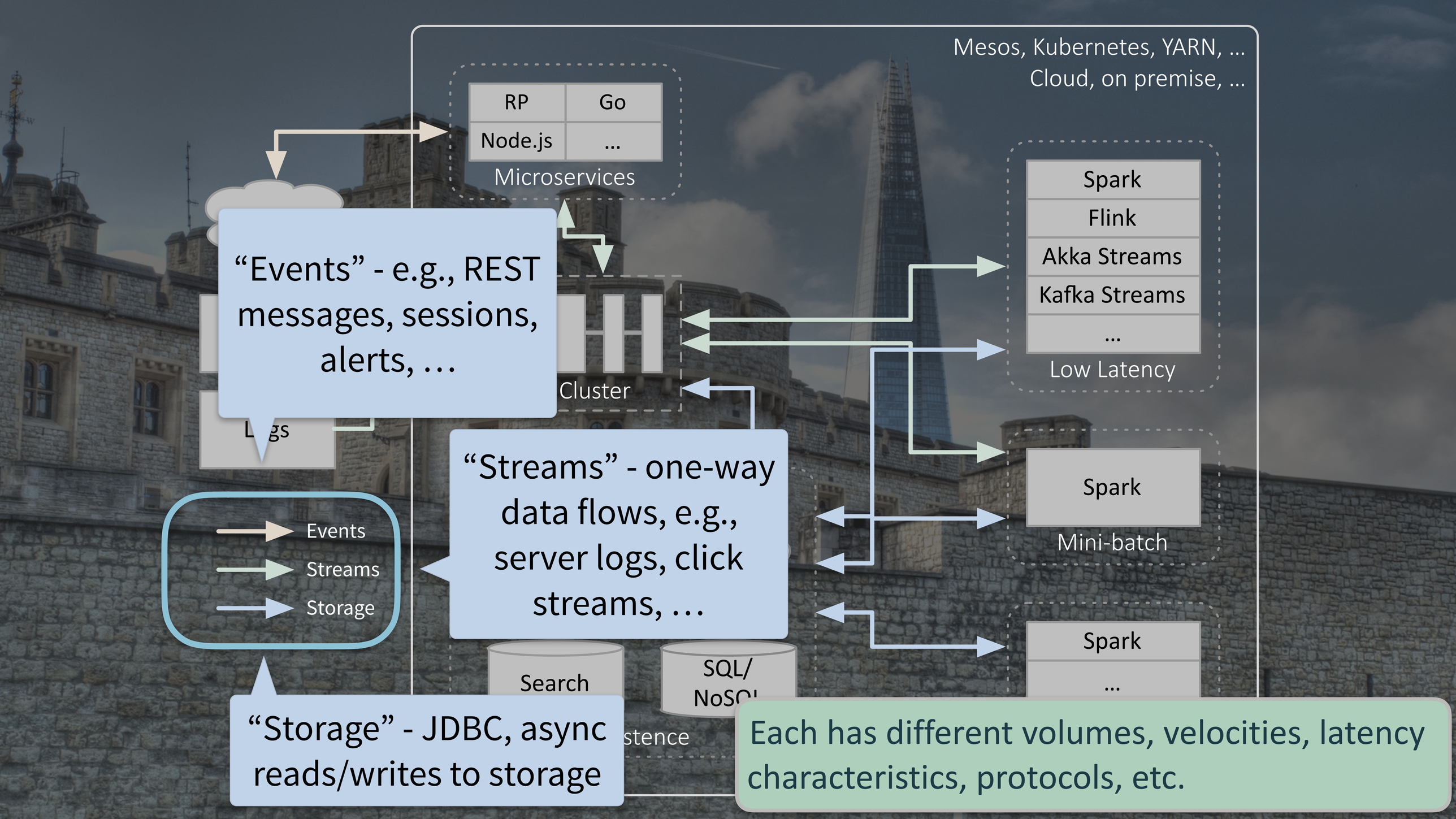
- 10000 feet (3048 meter ;^) view:
  - Hadoop is the database deconstructed and reimagined
  - Ideal for batch and interactive apps
- … but also constrained by that model

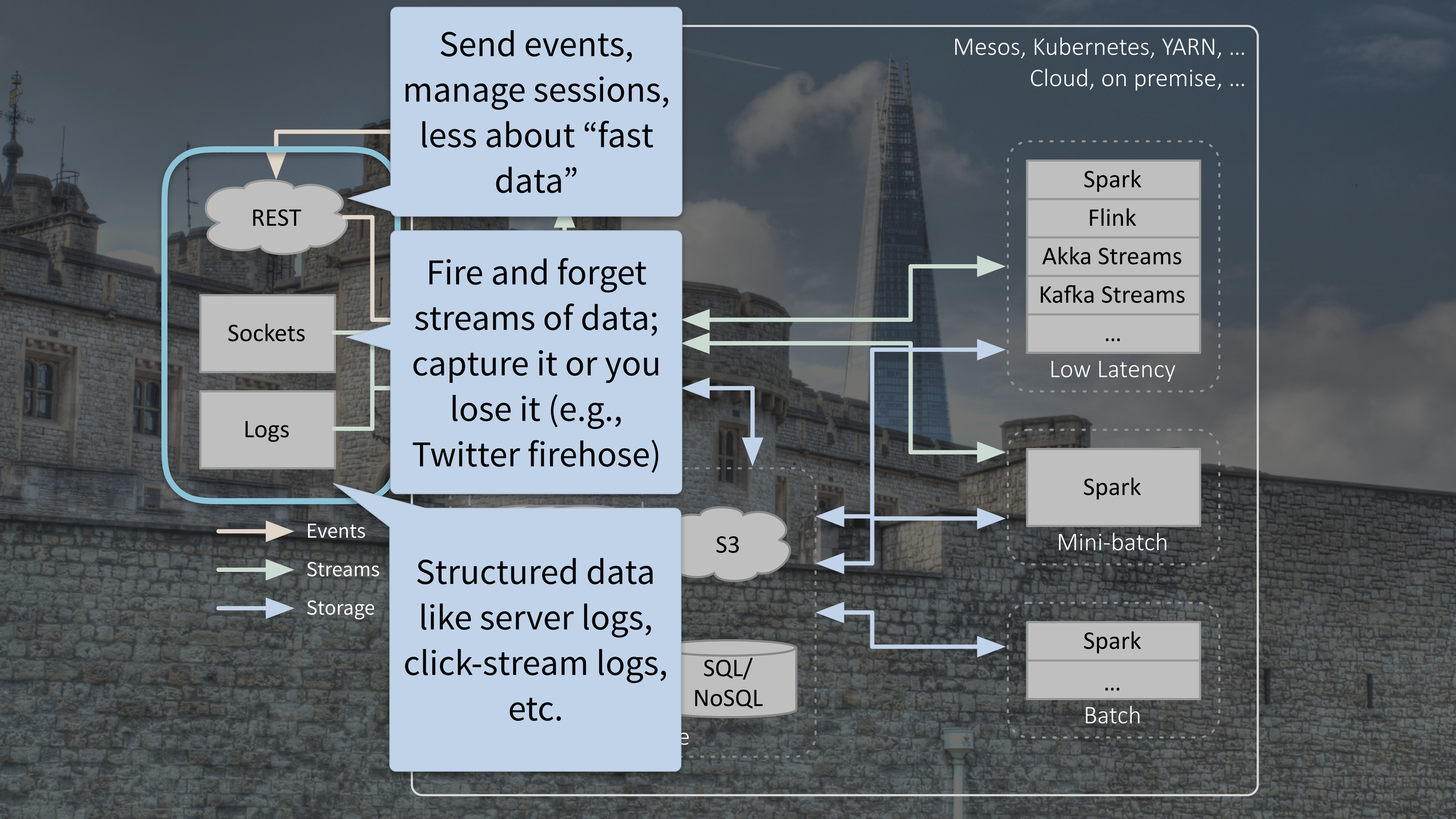


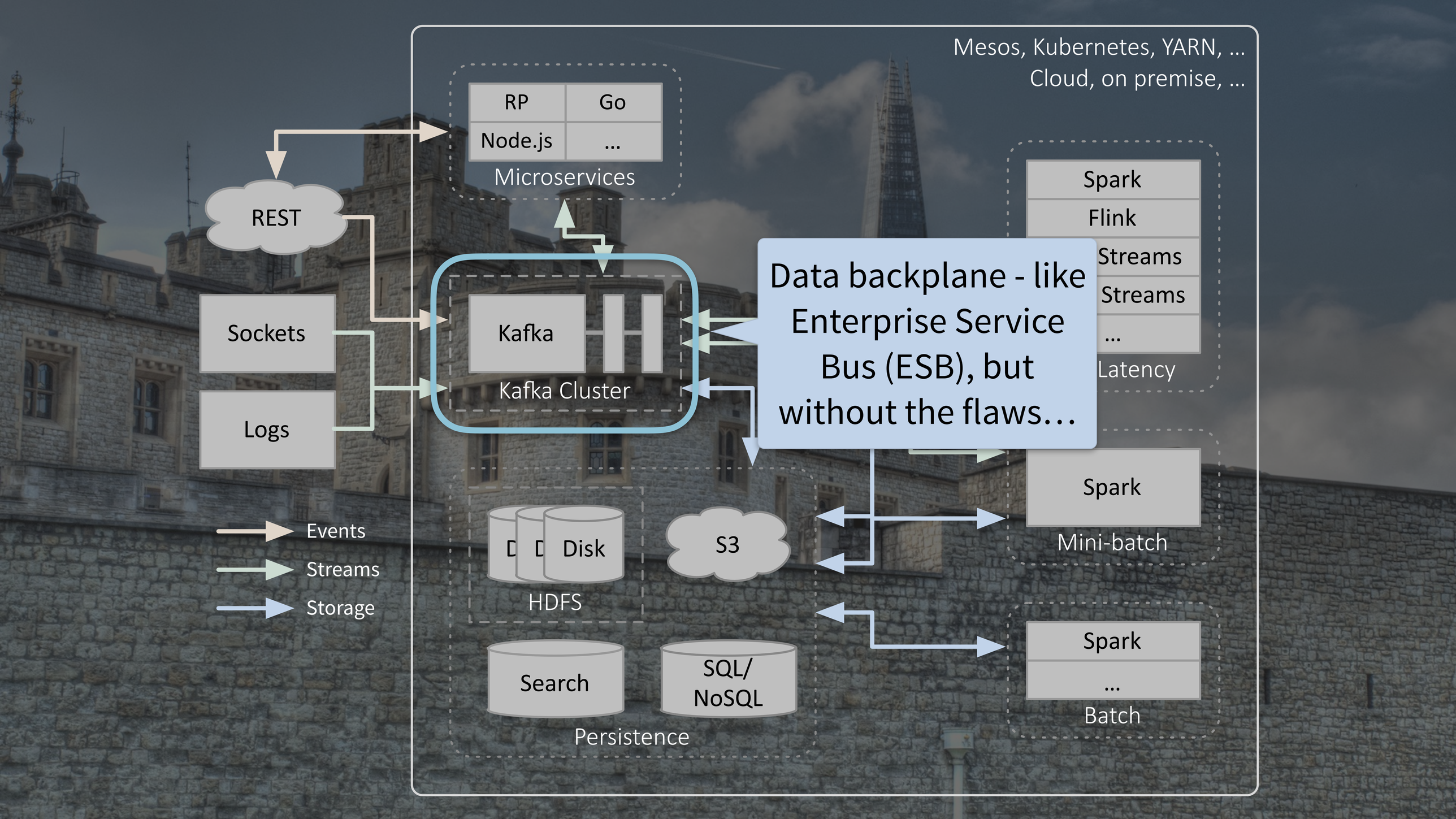
New Fast Data Architecture


Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Go
Node.js
…
Microservices
REST
Sockets
Logs
Kafka
Kafka Cluster
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Disk
HDFS
S3
Search
SQL/
NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage
Fleshes out earlier example architectures

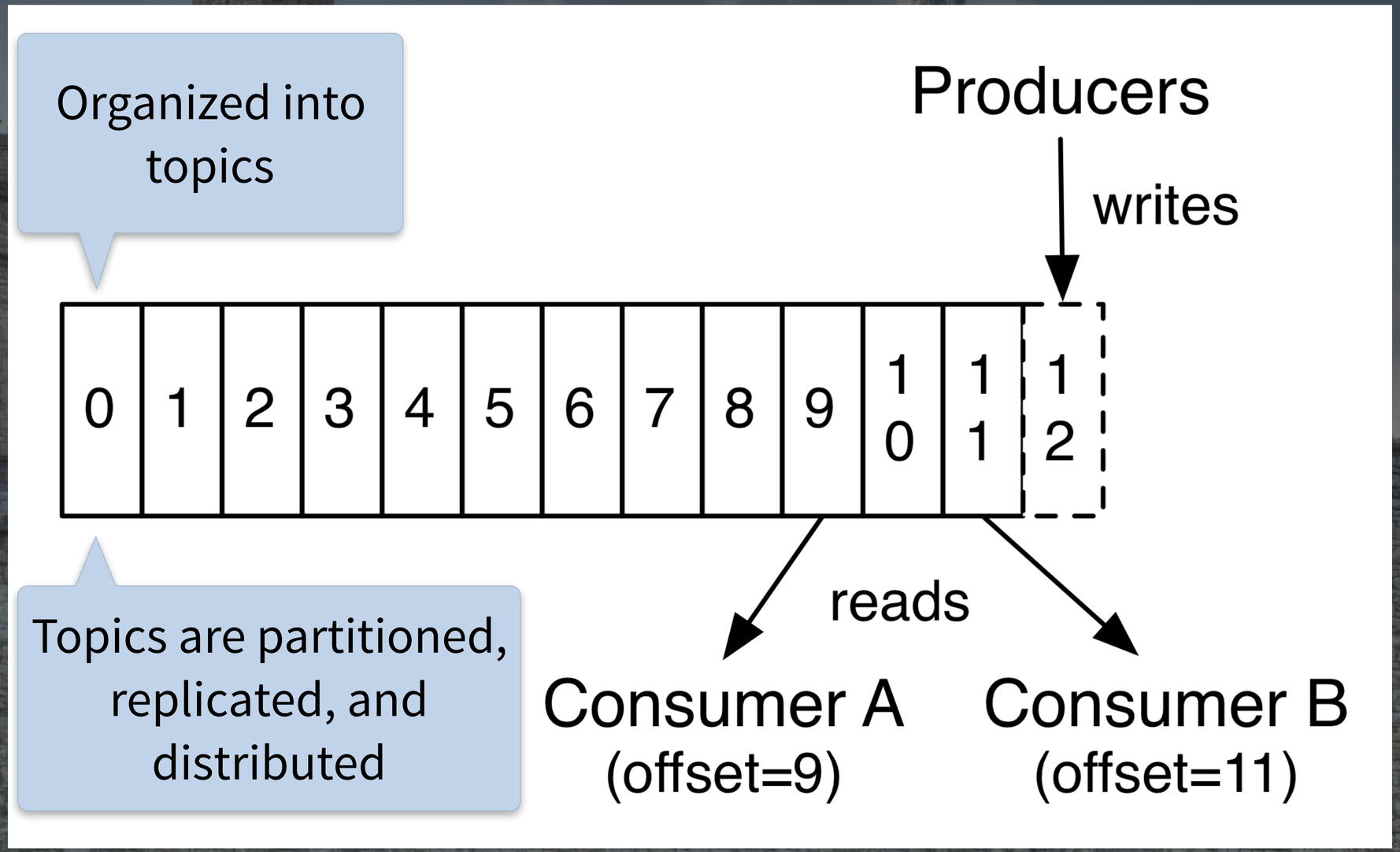

Mesos, Kubernetes, YARN, …
Cloud, on premise, …
Mesos and Kubernetes provide the job and resource management needed for dynamic heterogenous work loads
While YARN can be used, it's not flexible enough for today's dynamic workloads
Deploy in the cloud or on premise
REST
Sockets
Logs
Kafka
Kafka Cluster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


Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Go
Node.js
…
Microservices
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Cluster
Spark
Mini-batch
Spark
…
Events
Streams
Storage
Search
SQL/
NoSQL
stence
“Events” - e.g., REST messages, sessions, alerts, …
“Streams” - one-way data flows, e.g., server logs, click streams, …
“Storage” - JDBC, async reads/writes to storage
Each has different volumes, velocities, latency characteristics, protocols, etc.

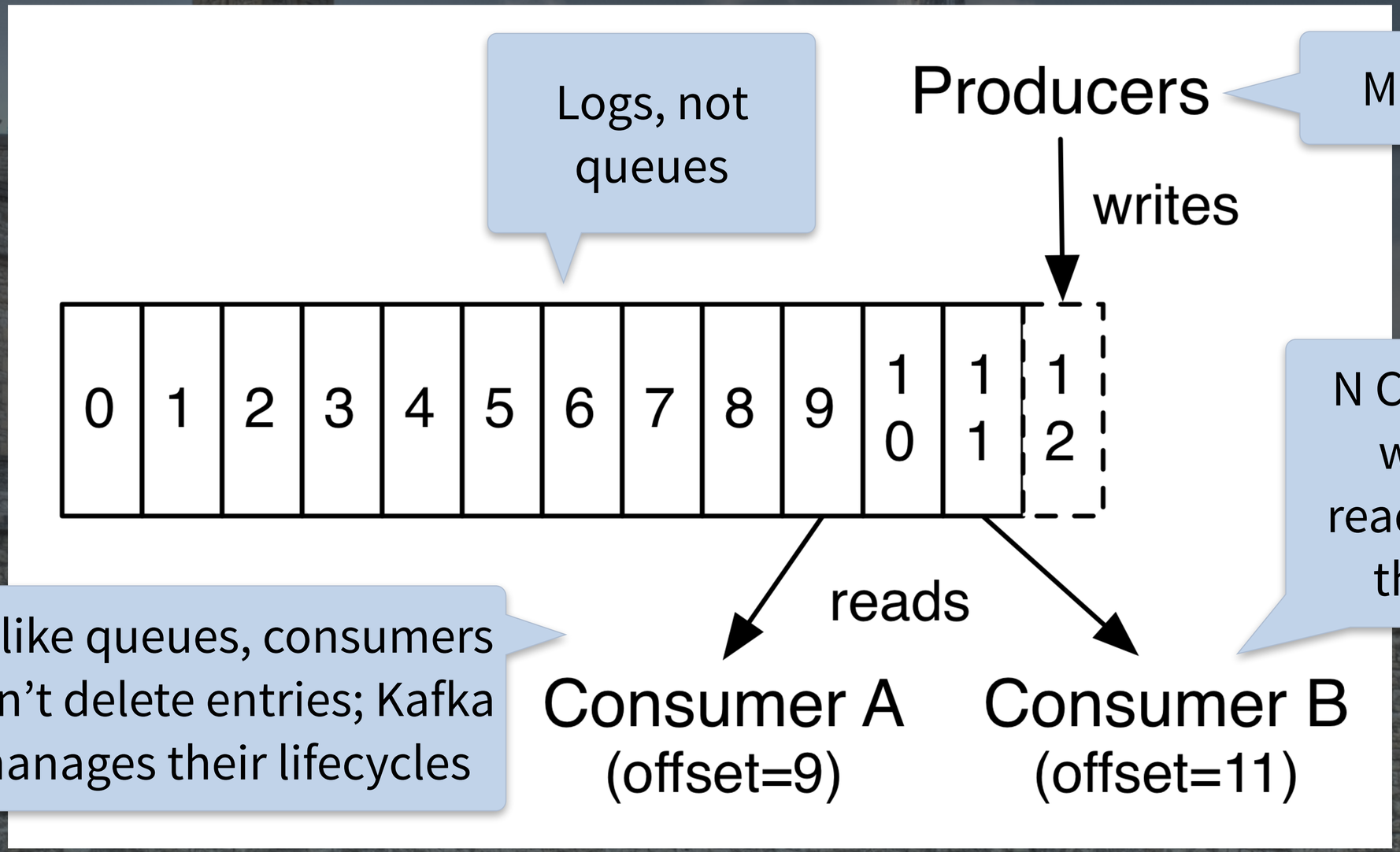

Send events, manage sessions, less about “fast data”
Fire and forget streams of data; capture it or you lose it (e.g., Twitter firehose)
Structured data like server logs, click-stream logs, etc.
Mesos, Kubernetes, YARN, …
Cloud, on premise, …
REST
Sockets
Logs
Events
Streams
Storage
S3
SQL/
NoSQL
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Spark
…
Batch


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

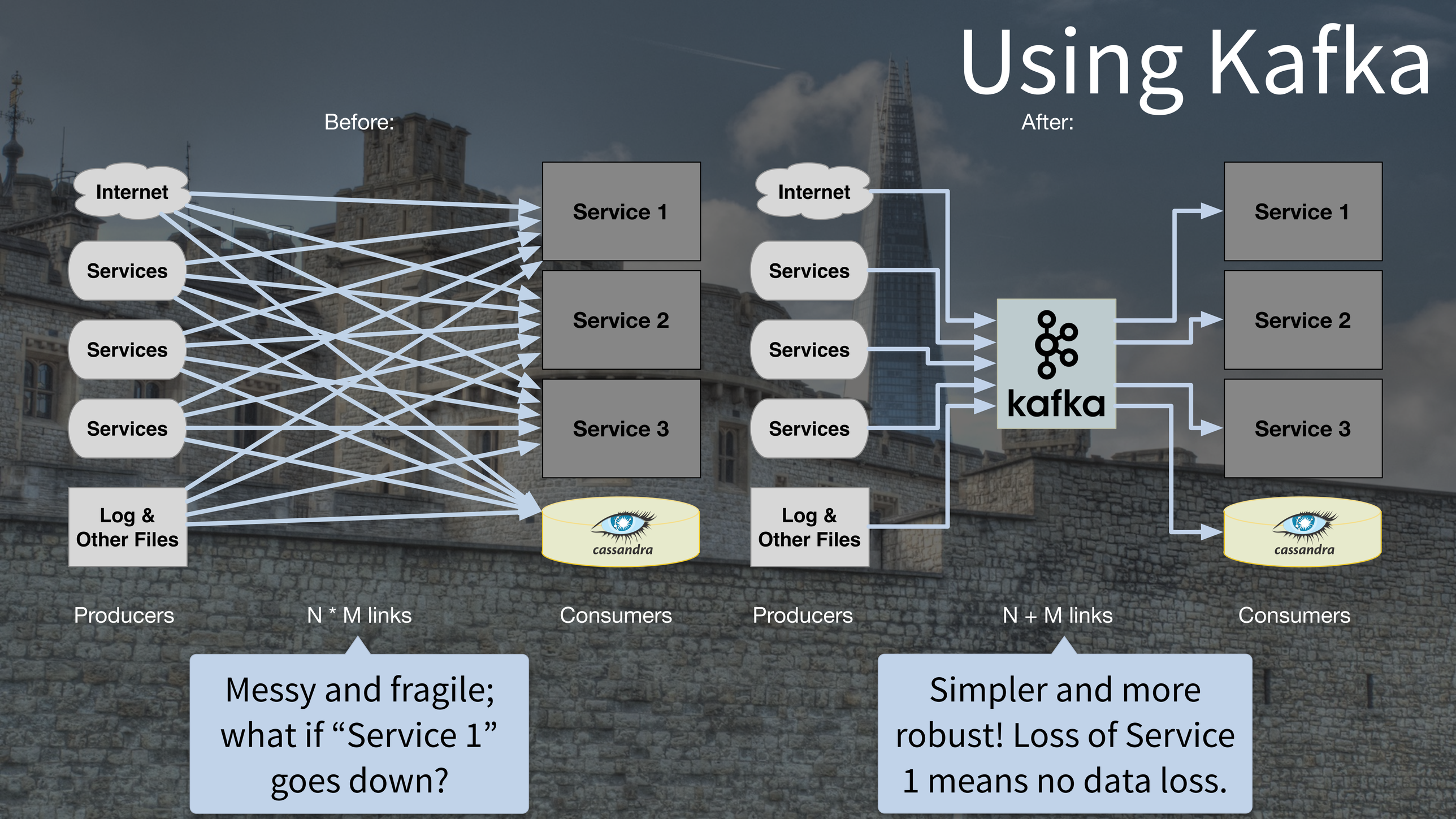
Why Kafka?


Organized into topics
Producers
writes
0
1
2
3
4
5
6
7
8
9
10
11
12
reads
Consumer A
(offset=9)
Consumer B
(offset=11)
Topics are partitioned, replicated, and distributed

Why Kafka?

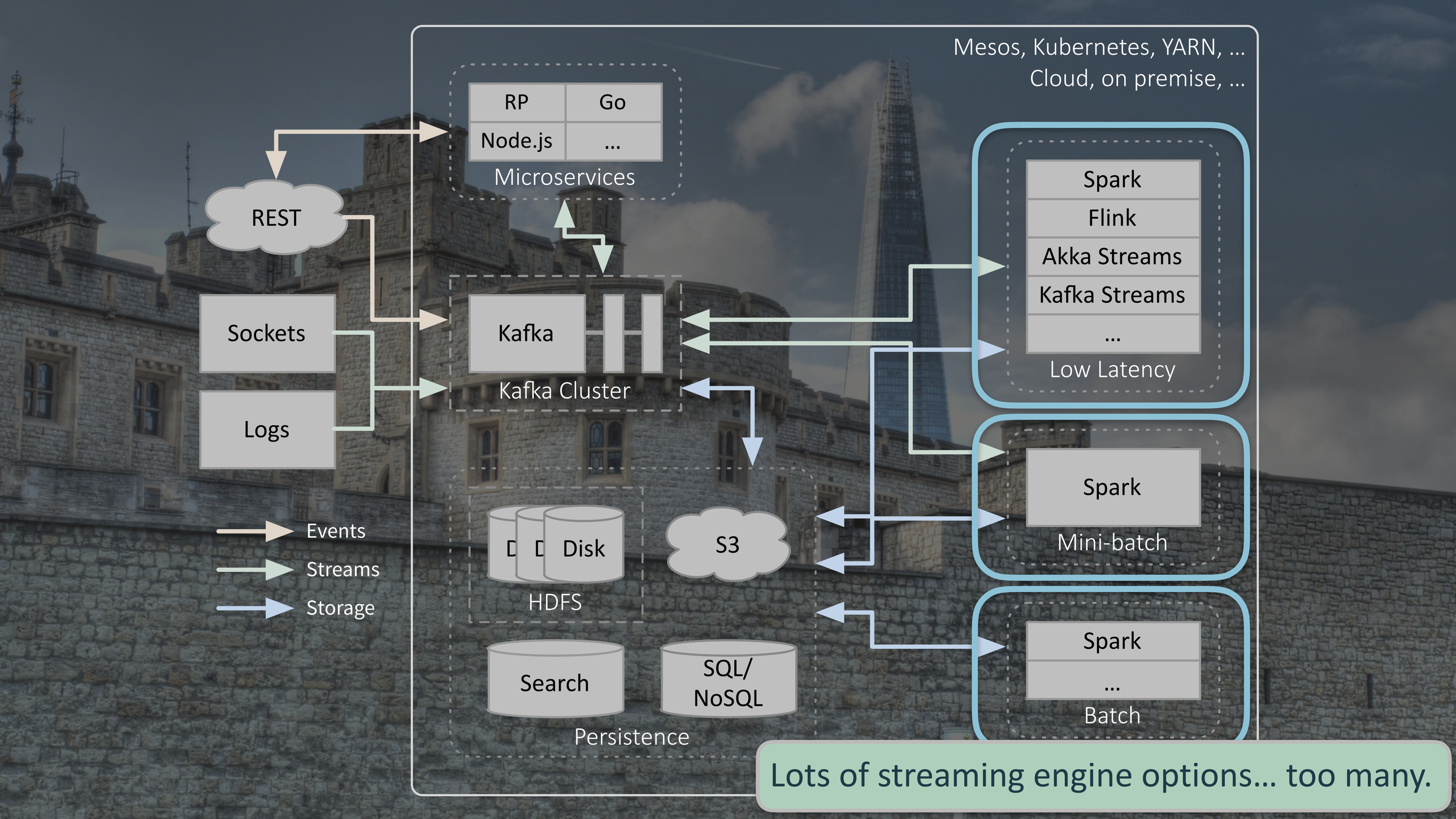

Logs, not queues
Producers
M Producers
writes
0
1
2
3
4
5
6
7
8
9
10
11
12
N Consumers, who start reading where they want
reads
Unlike queues, consumers don’t delete entries; Kafka manages their lifecycles
Consumer A
(offset=9)
Consumer B
(offset=11)

Using Kafka



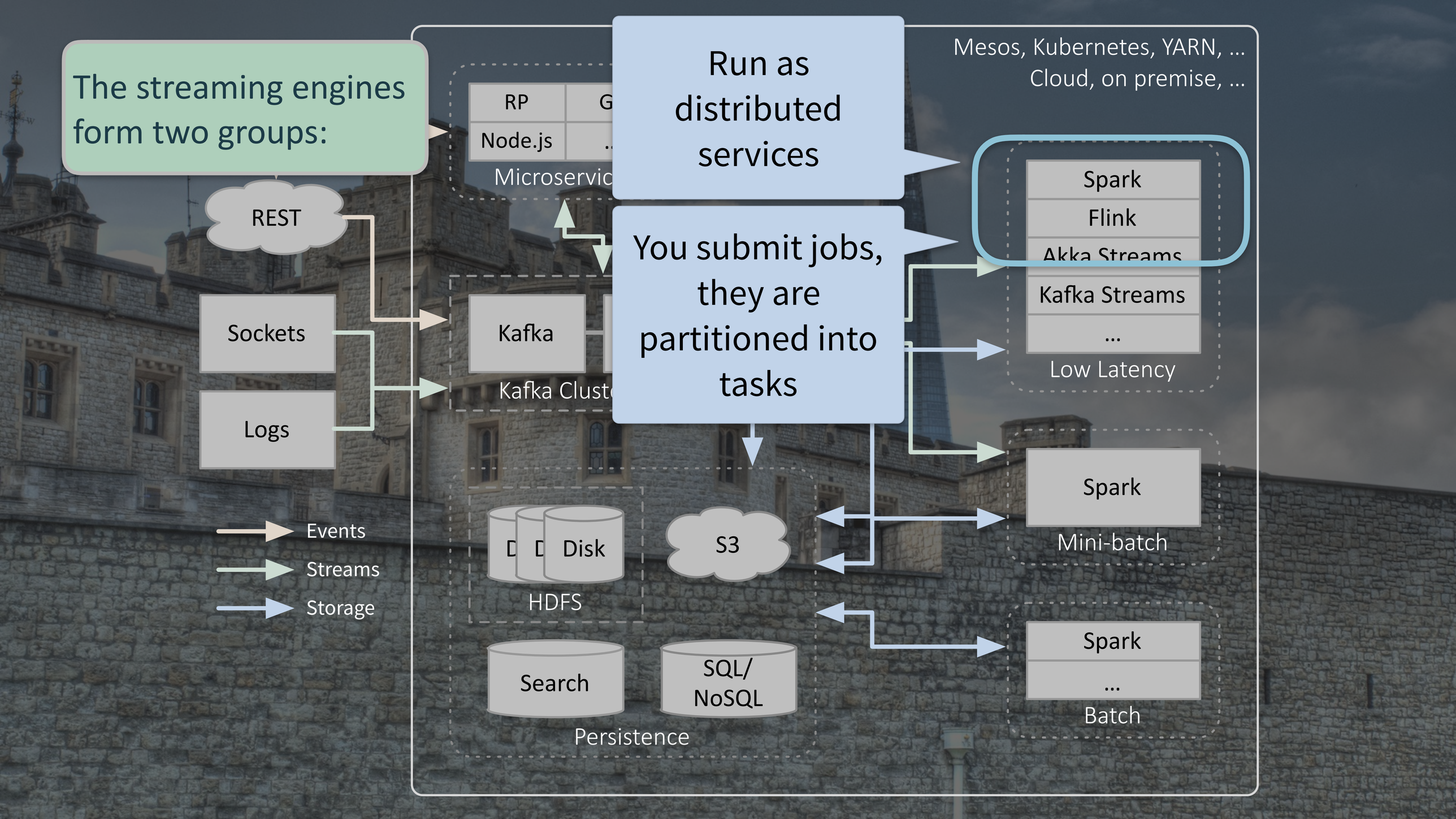
Messy and fragile; what if “Service 1” goes down?

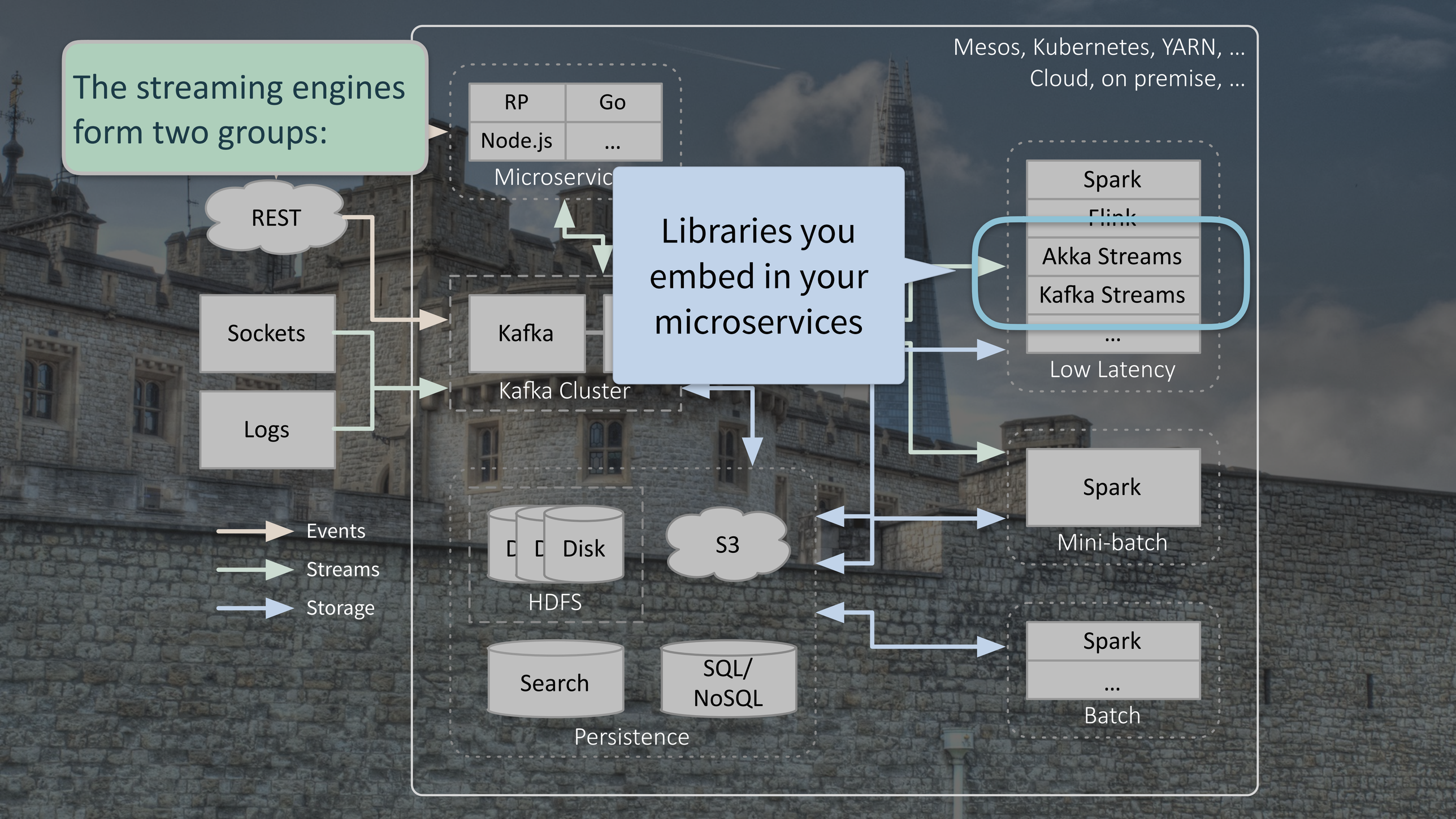
Simpler and more robust! Loss of Service 1 means no data loss.

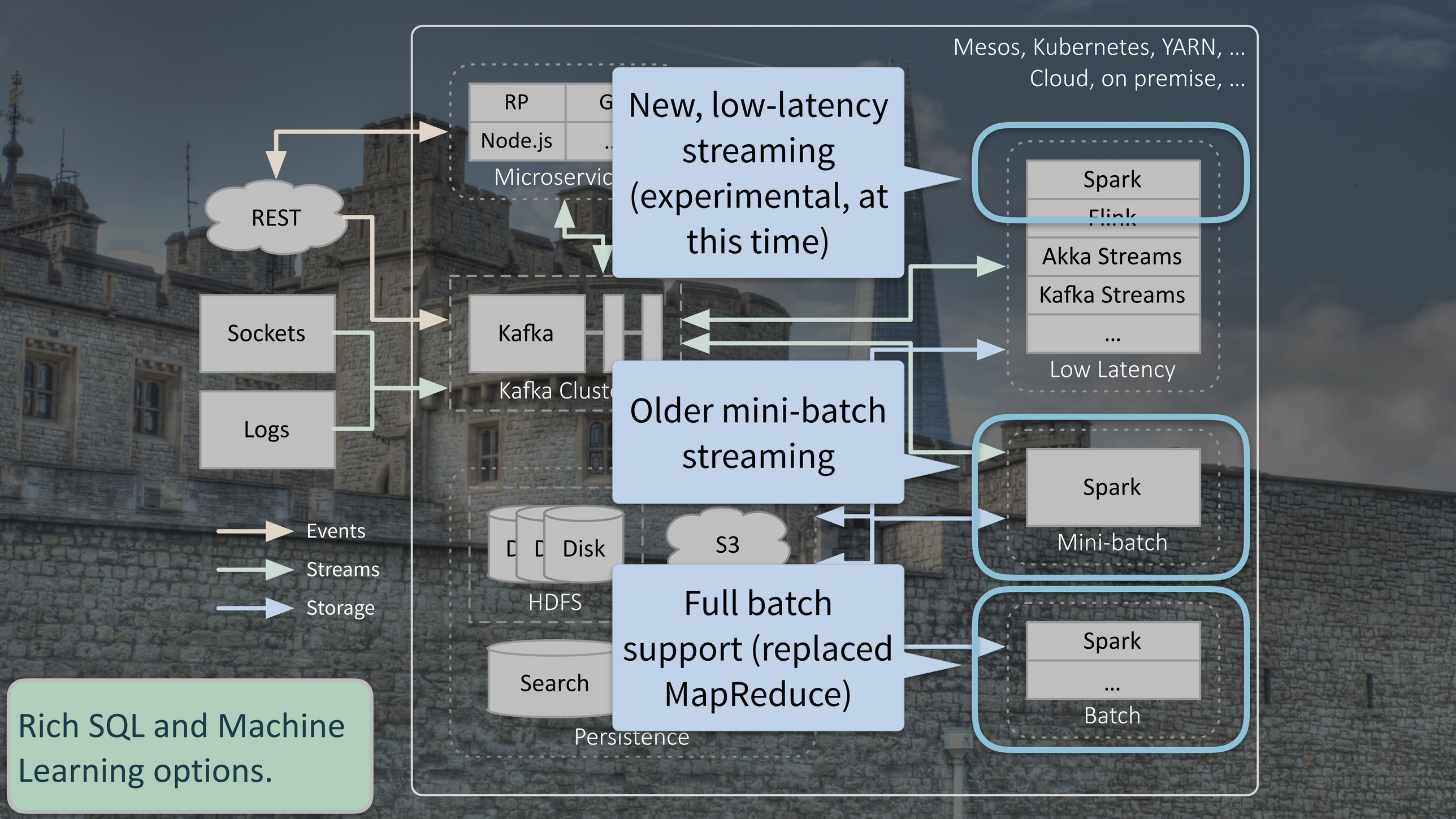

Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Go
Node.js
…
Microservices
REST
Sockets
Logs
Kafka
Kafka Cluster
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Spark
…
Batch
Disk
HDFS
S3
Search
SQL/
NoSQL
Persistence
Events
Streams
Storage
Lots of streaming engine options… too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

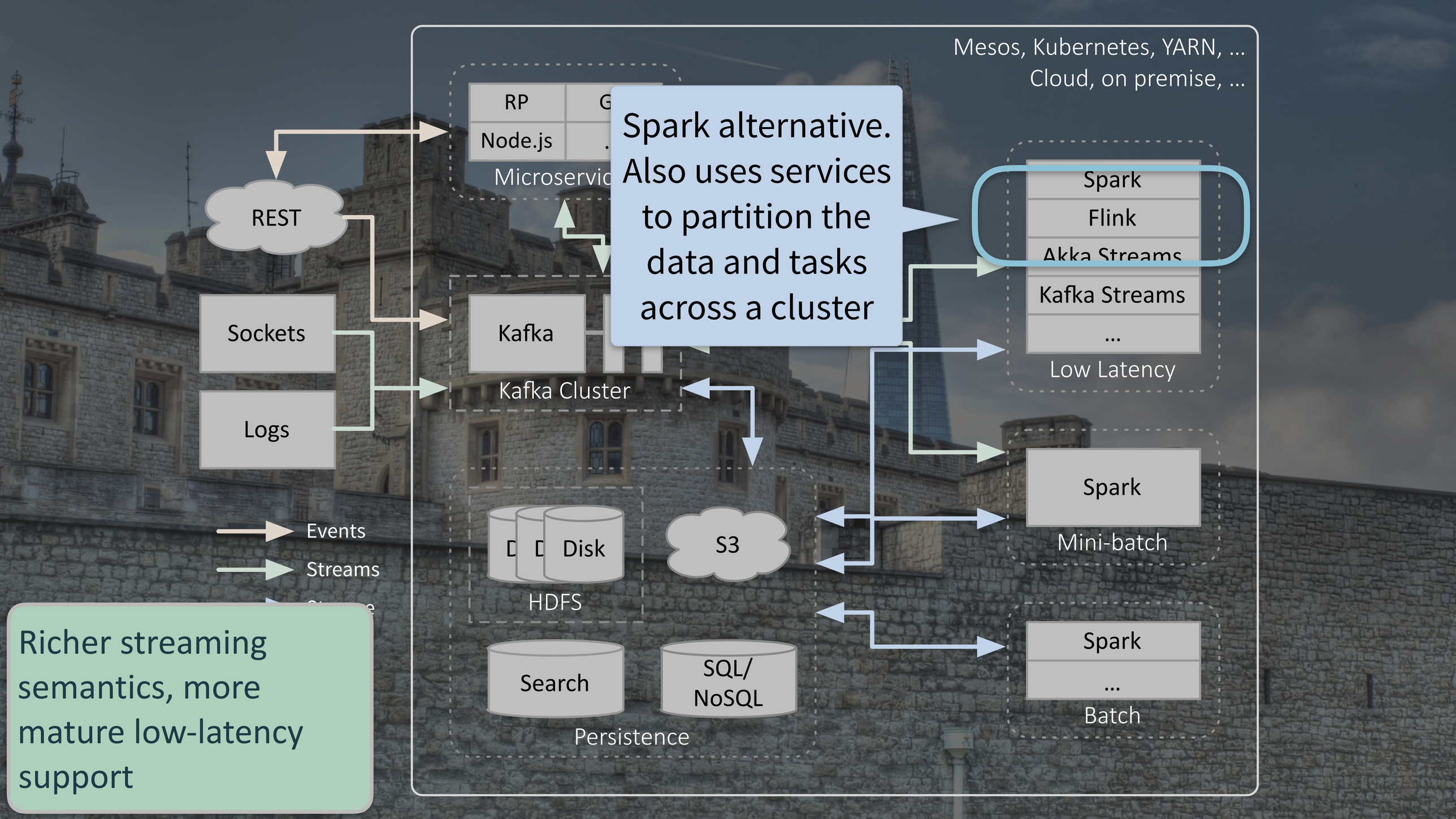
How do you choose?

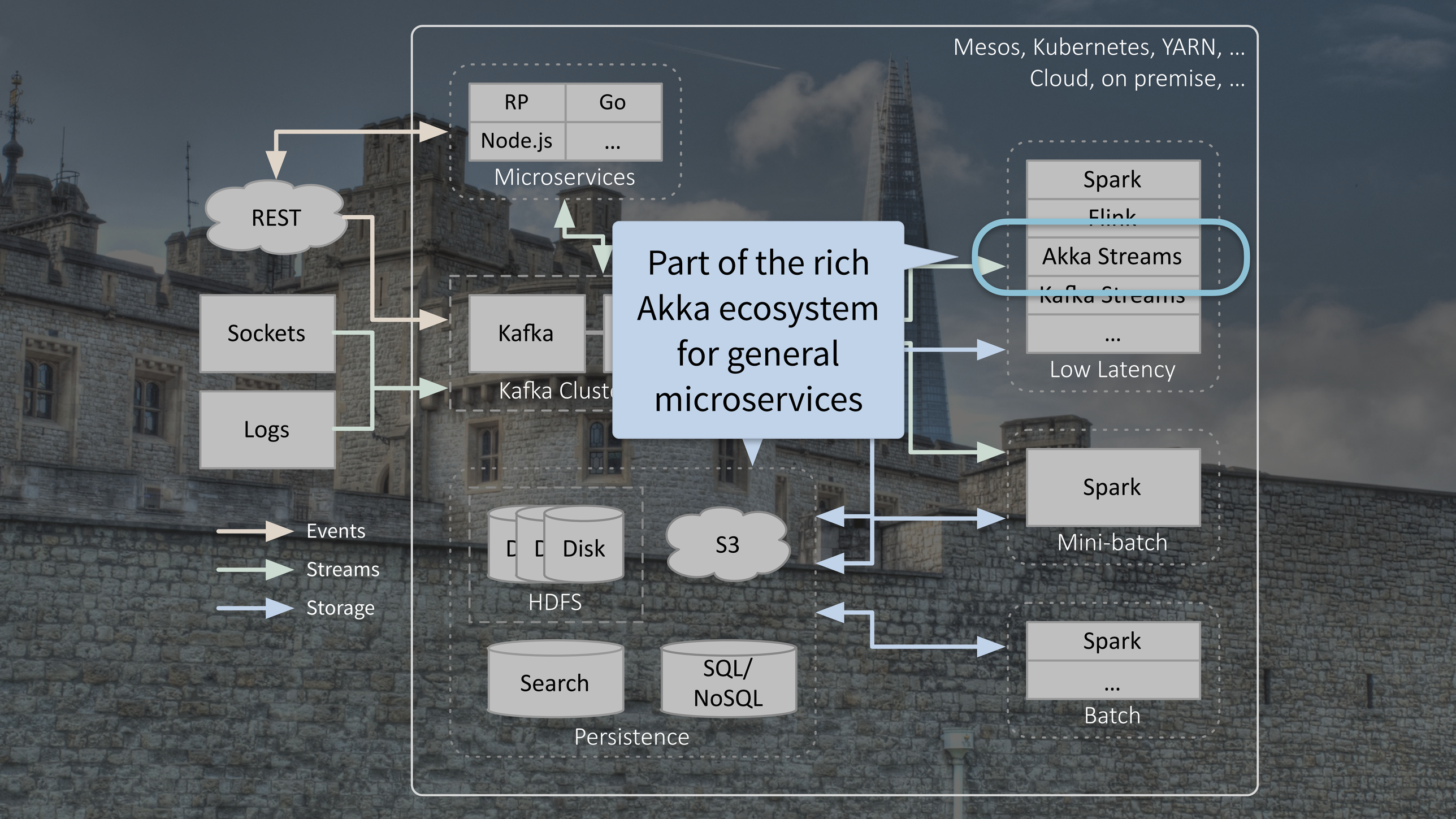
- Latency? How low?
- Volume: How high?
- Which kinds of data processing?
- How do you want to build, deploy, and manage these services?

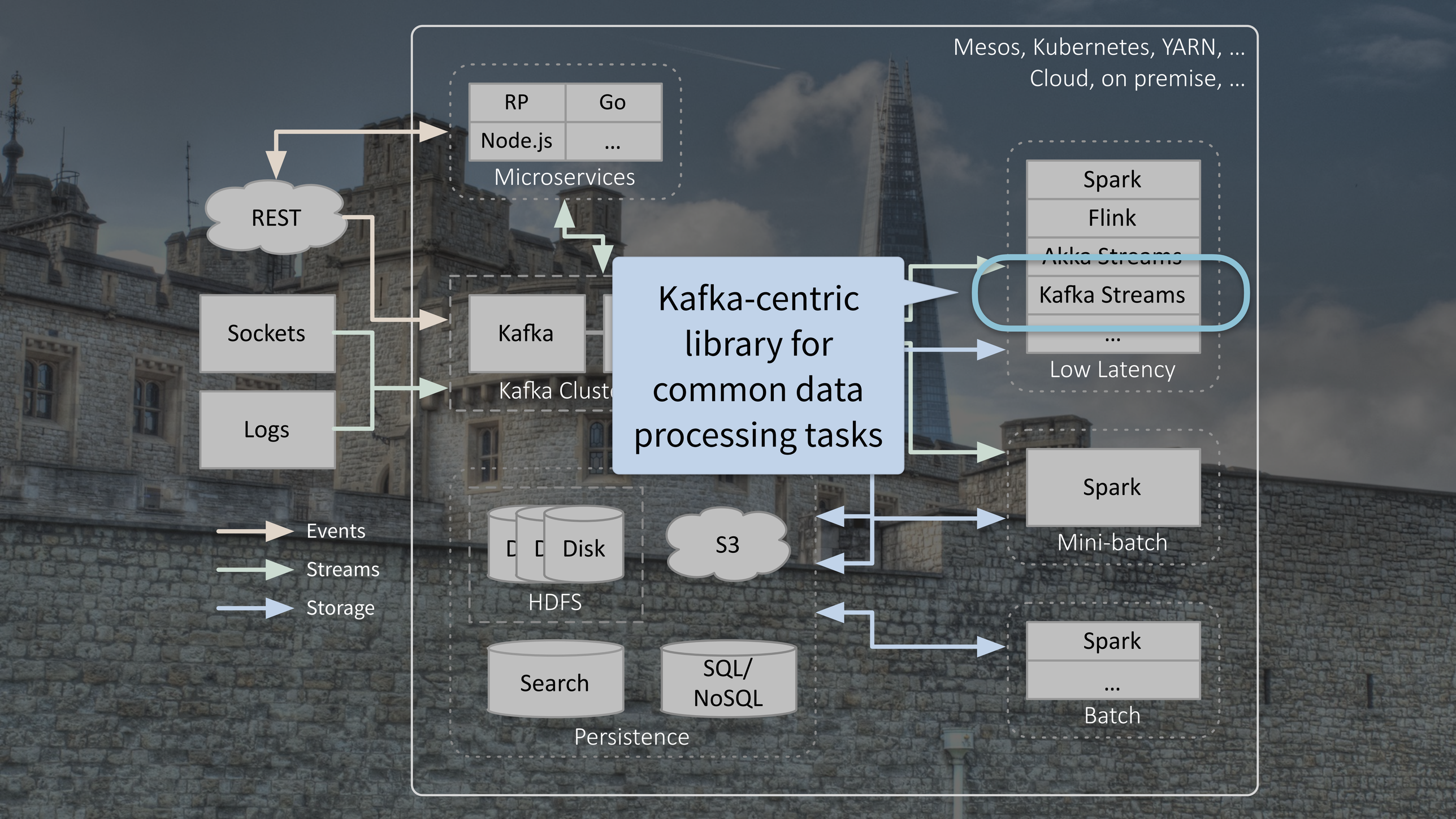

The streaming engines form two groups:
Run as distributed services
You submit jobs, they are partitioned into tasks
Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Node.js
Microservic
REST
Sockets
Logs
Kafka
Kafka Clust
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Disk
HDFS
S3
Search
SQL/
NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage

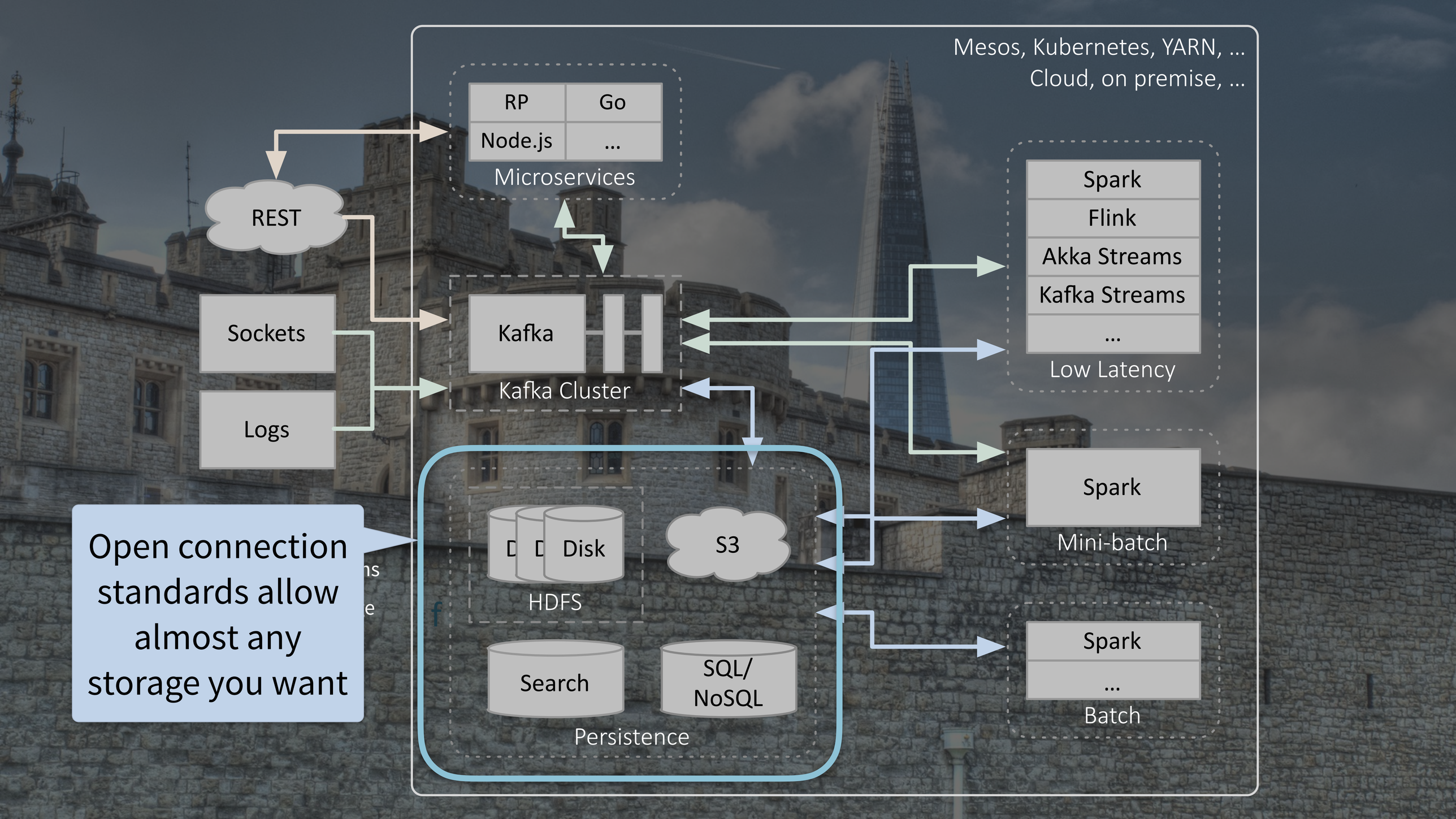

The streaming engines form two groups:
Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Go
Node.js
…
Microservic
REST
Sockets
Logs
Kafka
Kafka Cluster
Libraries you embed in your microservices
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Disk
HDFS
S3
Search
SQL/ NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage

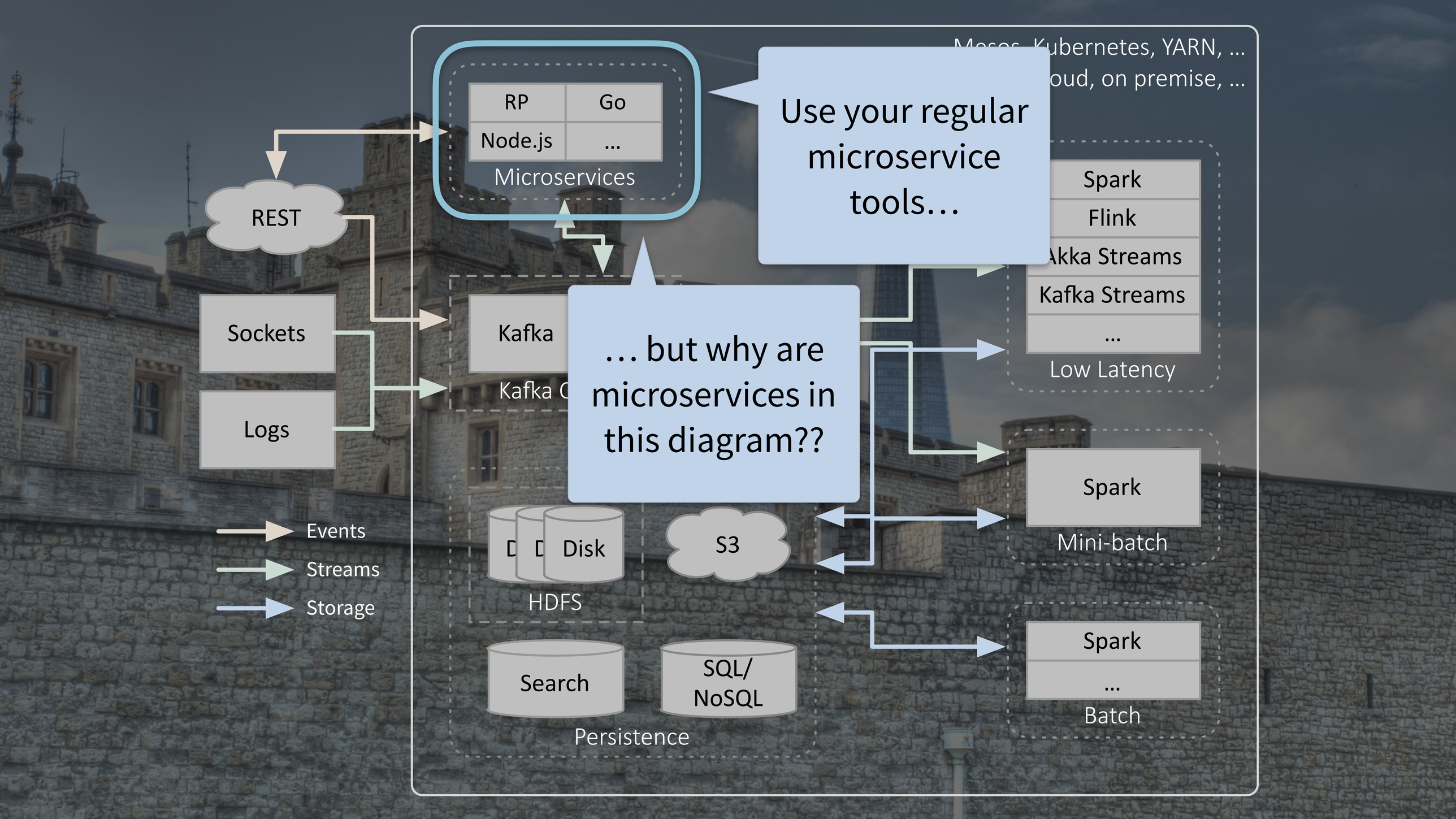

Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Node.js
Microservic
REST
Sockets
Logs
Kafka
Kafka Clust
New, low-latency streaming (experimental, at this time)
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Older mini-batch streaming
Spark
Mini-batch
Disk
S3
HDFS
Search
Persistence
Full batch support (replaced MapReduce)
Spark
…
Batch
Events
Streams
Storage
Rich SQL and Machine Learning options.


Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Node.js
Microservic
REST
Sockets
Logs
Kafka
Kafka Cluster
Spark alternative. Also uses services to partition the data and tasks across a cluster
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Spark
…
Batch
Disk
HDFS
S3
Search
SQL/
NoSQL
Persistence
Events
Streams
Richer streaming semantics, more mature low-latency support


Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Go
Node.js
…
Microservices
REST
Sockets
Logs
Kafka
Kafka Clust
Part of the rich Akka ecosystem for general microservices
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Disk
HDFS
S3
Search
SQL/ NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage


Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Go
Node.js
…
Microservices
REST
Sockets
Logs
Kafka
Kafka Clust
Kafka-centric library for common data processing tasks
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Disk
HDFS
S3
Search
SQL/ NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage


Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Go
Node.js
…
Microservices
REST
Sockets
Logs
Kafka
Kafka Cluster
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Spark
…
Batch
Disk
HDFS
S3
Search
SQL/
NoSQL
Persistence
Open connection standards allow almost any storage you want


Mesos, Kubernetes, YARN, …
Cloud, on premise, …
RP
Go
Node.js
…
Microservices
Use your regular microservice tools…
REST
Sockets
Logs
Kafka
Kafka C
… but why are microservices in this diagram??
Spark
Flink
Akka Streams
Kafka Streams
…
Low Latency
Spark
Mini-batch
Disk
HDFS
S3
Search
SQL/
NoSQL
Persistence
Spark
…
Batch
Events
Streams
Storage

Why Microservices in Fast Data?

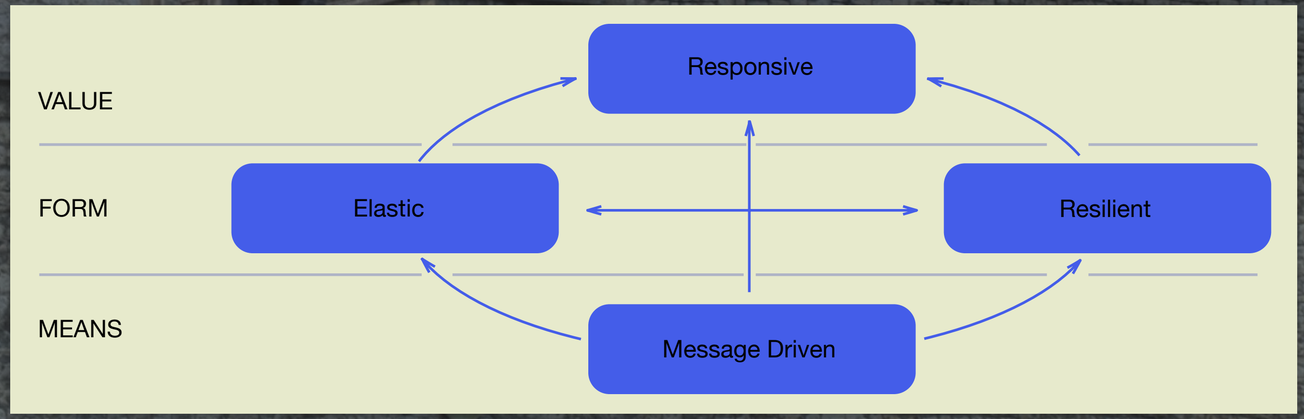
1. The trend is to run everything in big clusters using Kubernetes or Mesos
   - In the cloud or on-premise

Why Microservices in Fast Data?

2. If streaming gives you information faster…
   - … you'll want quick access to it in your other services!

Why Microservices in Fast Data?

3. Streaming raises the bar on data services
   - Compared to batch services, long-running streaming services must be more:
     - Scalable
     - Resilient
     - Flexible



https://www.reactivemanifesto.org/

Why Microservices in Fast Data?

4. This leads to our last major point…

Organizational Impact

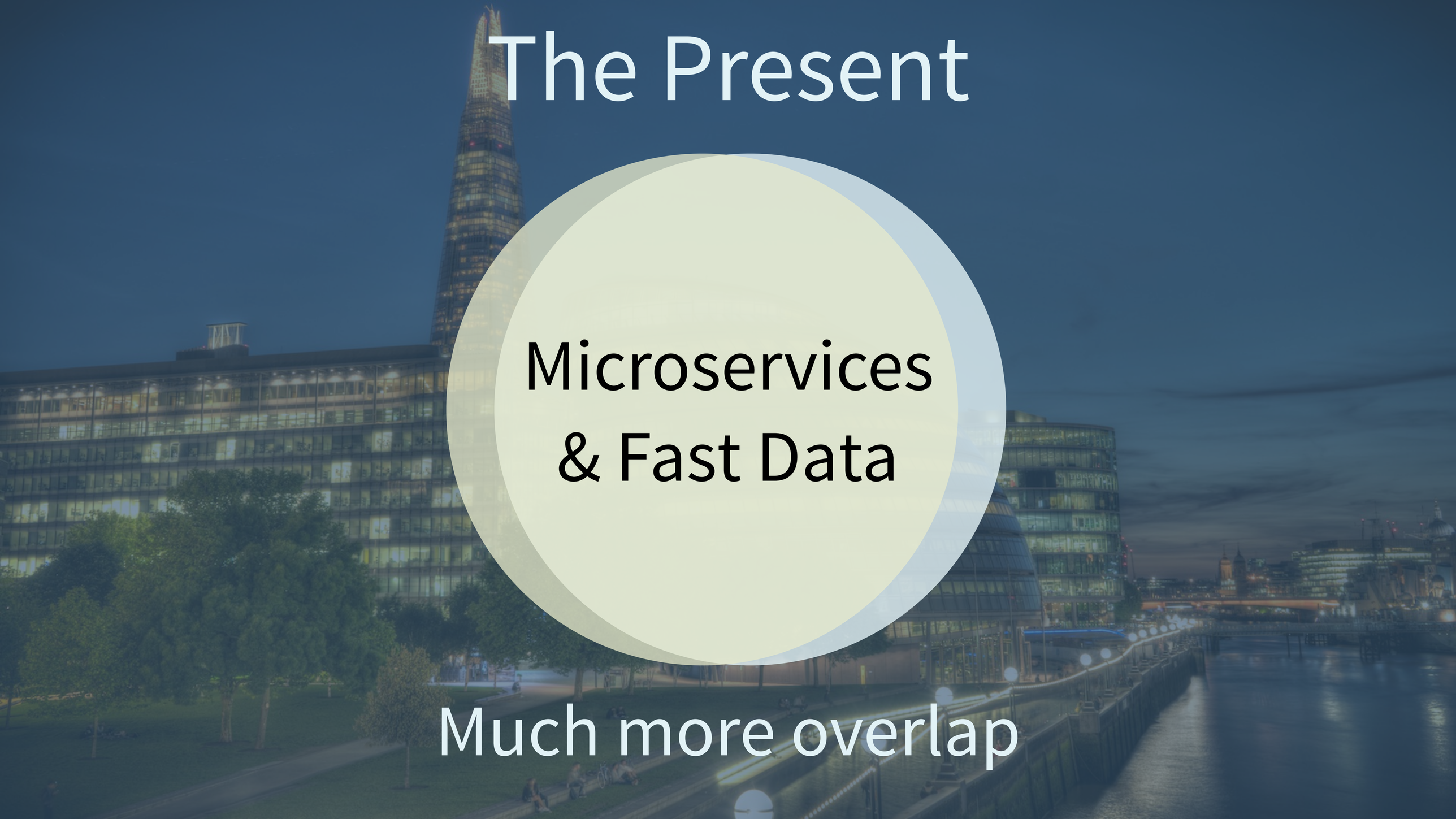
Organizational Impact

- Data engineers have to become good at highly-available microservices
- Microservice engineers have to become good at data

The Past

Some overlap in concerns, architecture

The Present



Much more overlap

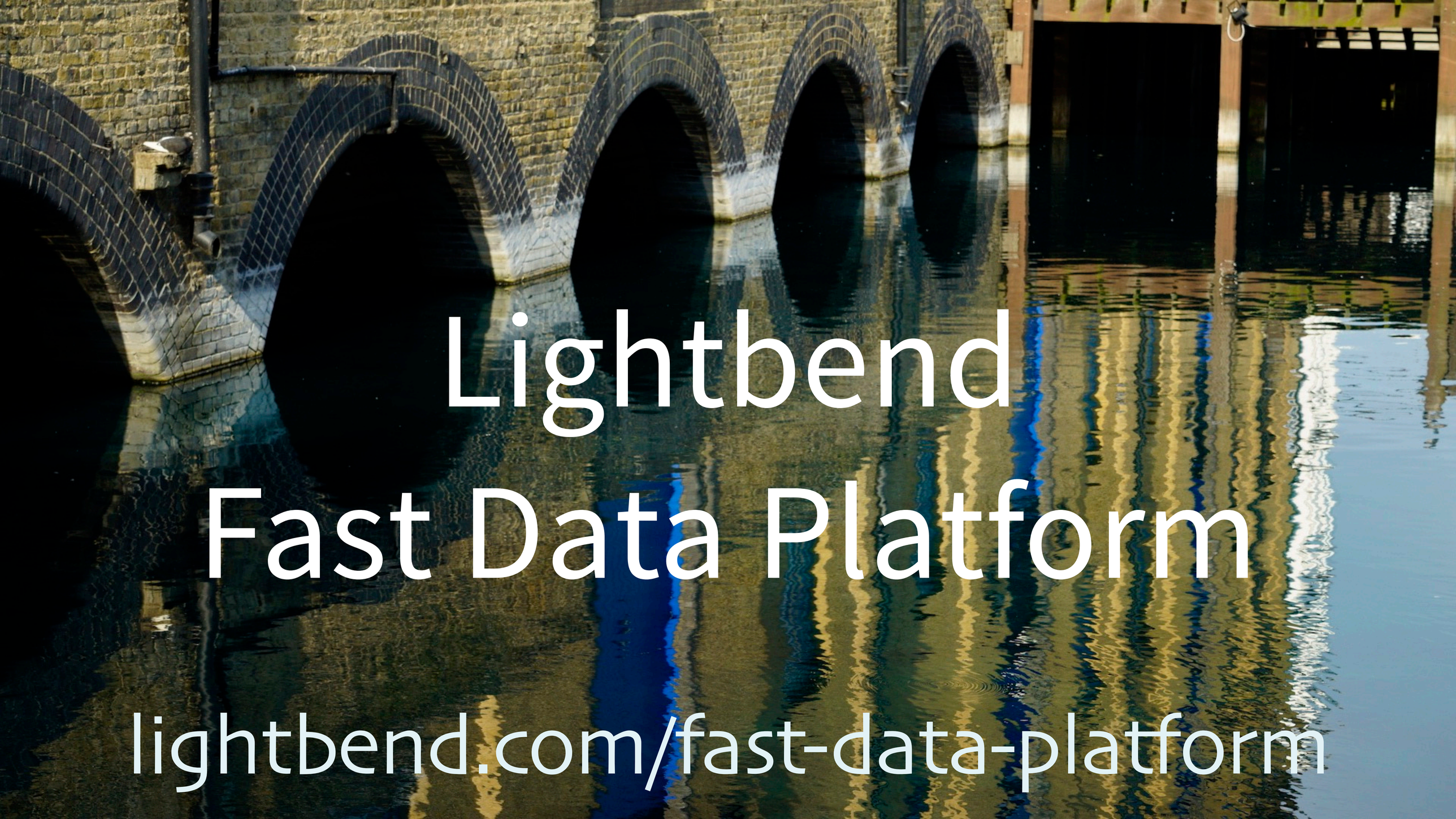
The Future?

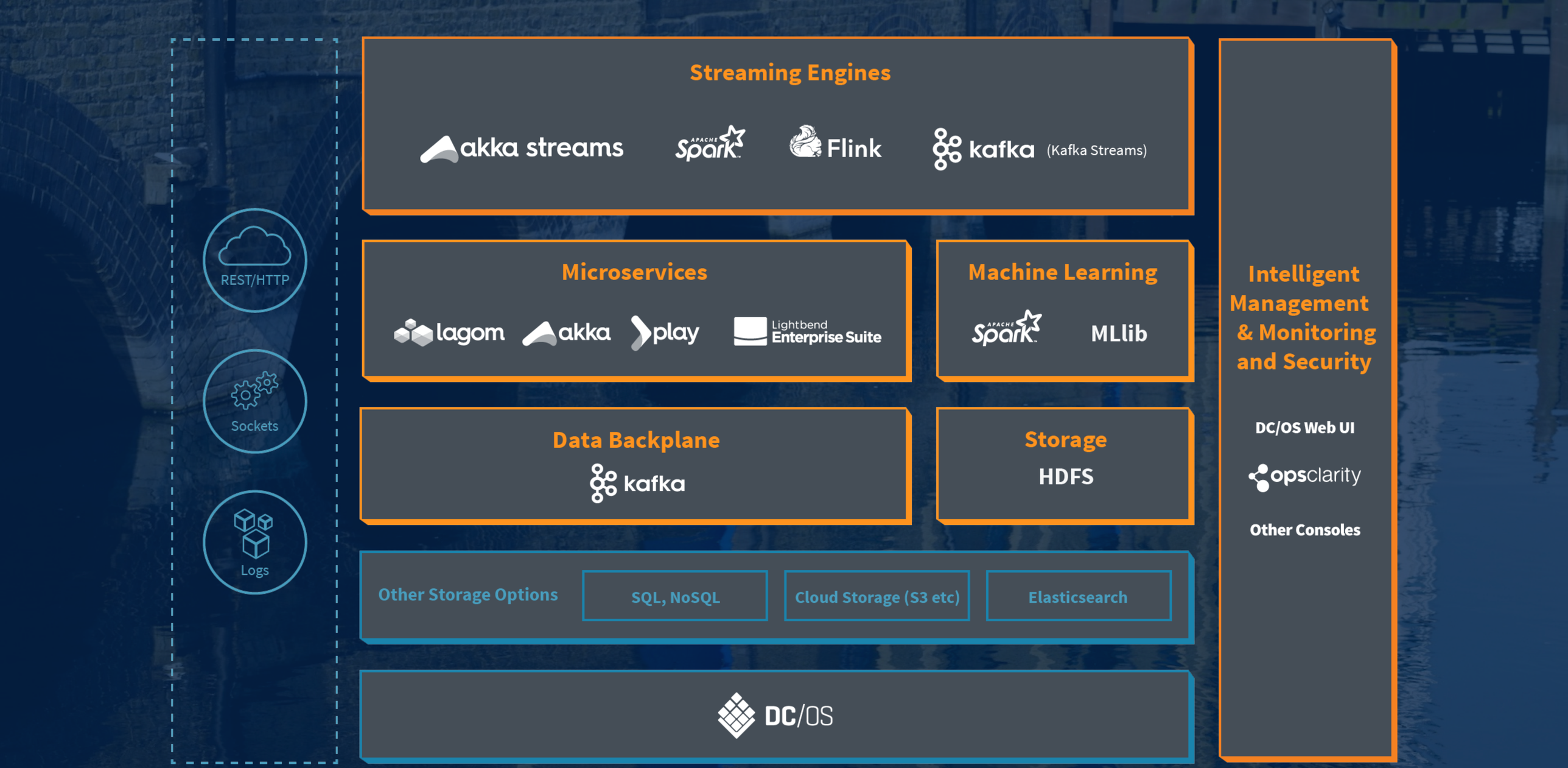
Why? Since streams process data incrementally, there is less need for large-scale tools like Spark, Flink

… and using microservices for everything simplifies development and operations

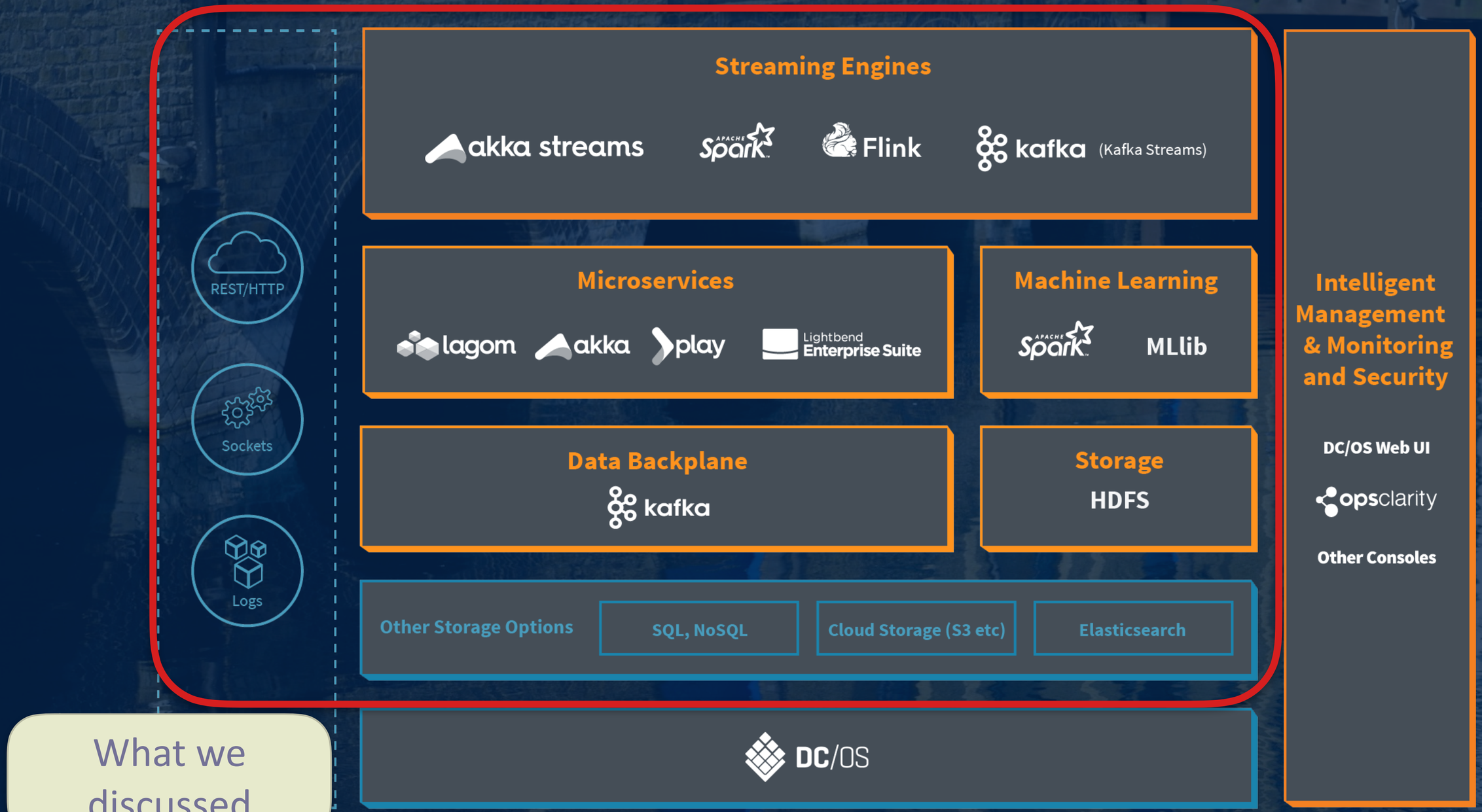
Microservices *for* Fast Data

Much more microservice focused?


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

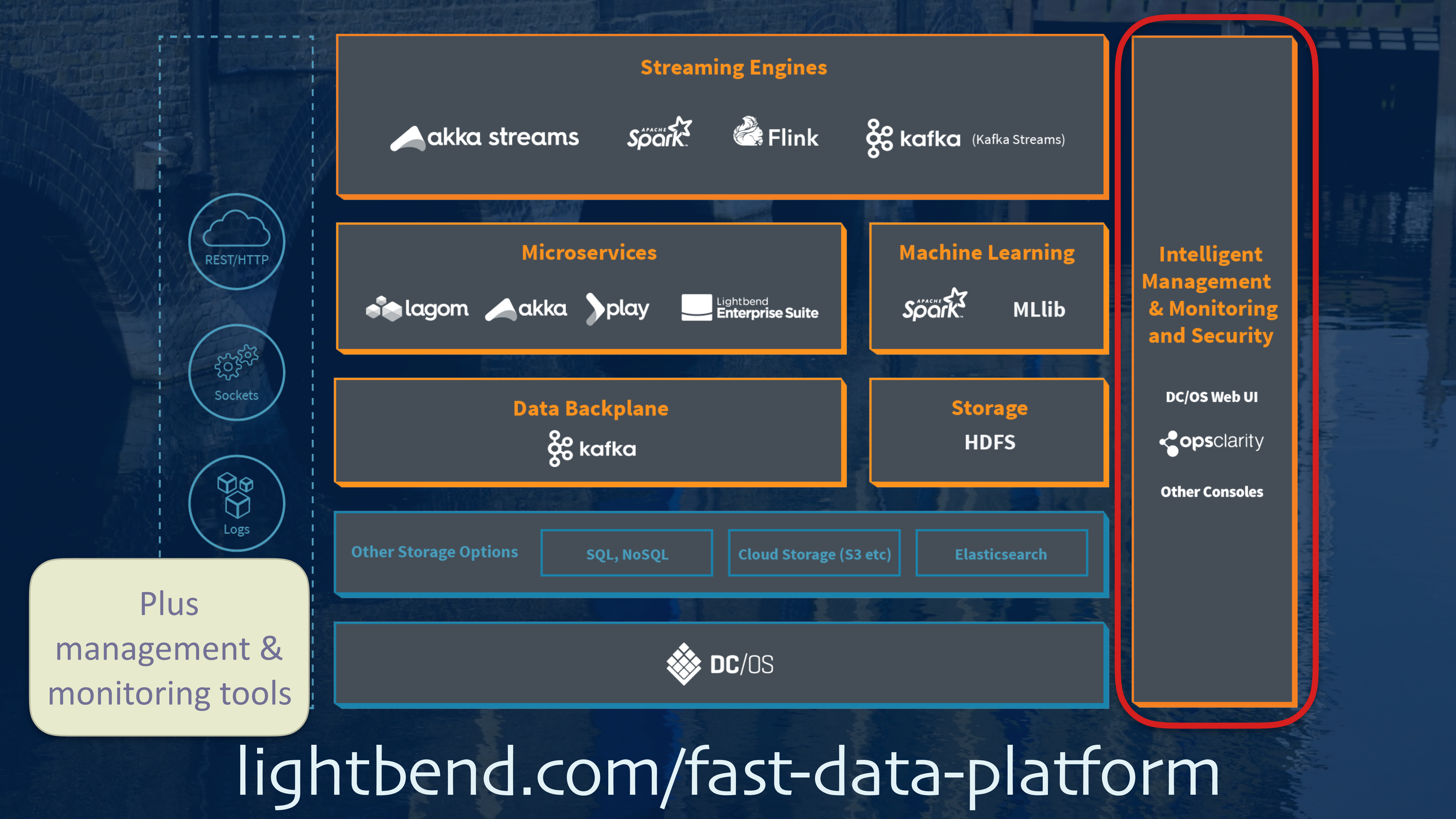


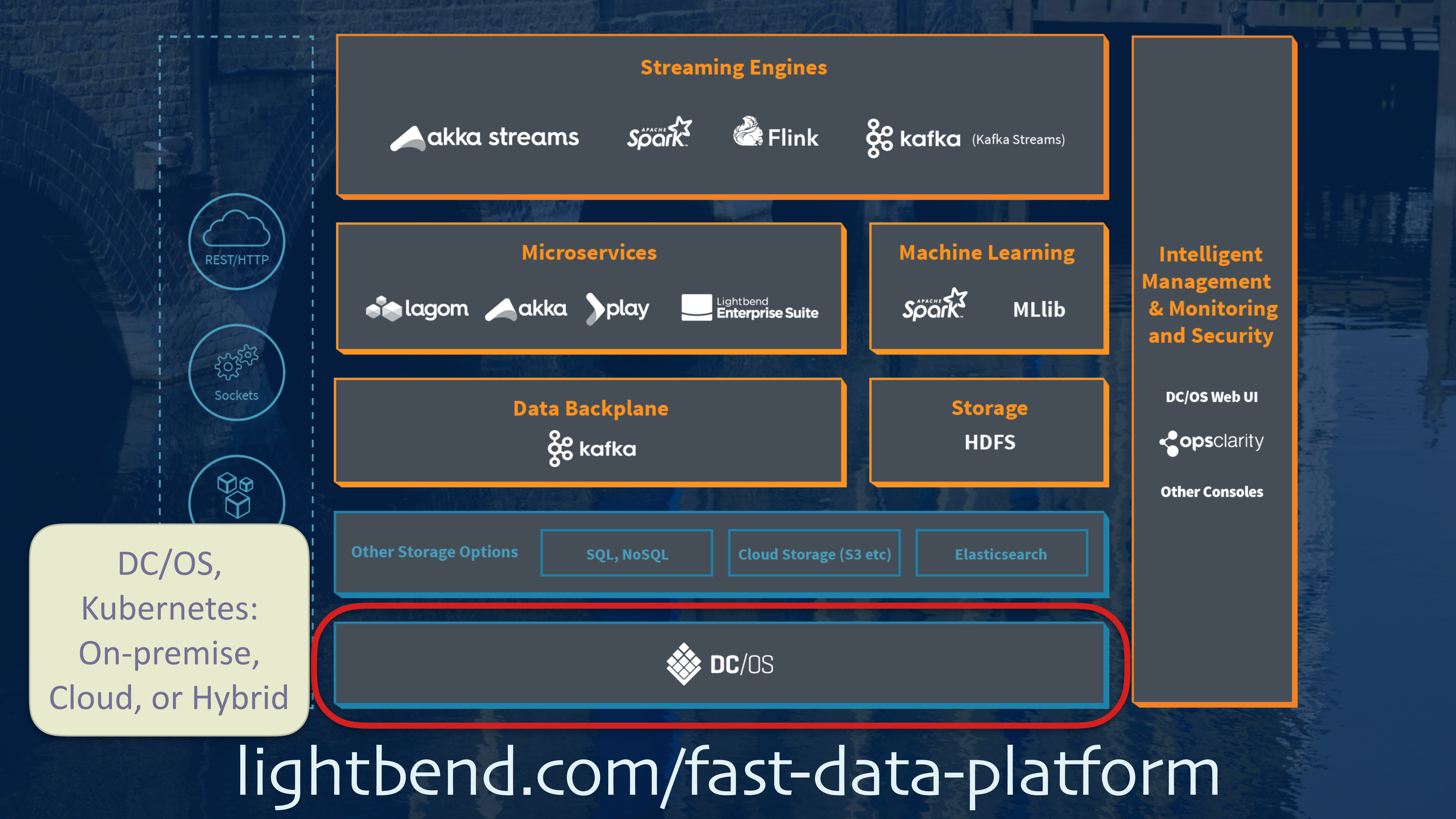
lightbend.com/fast-data-platform

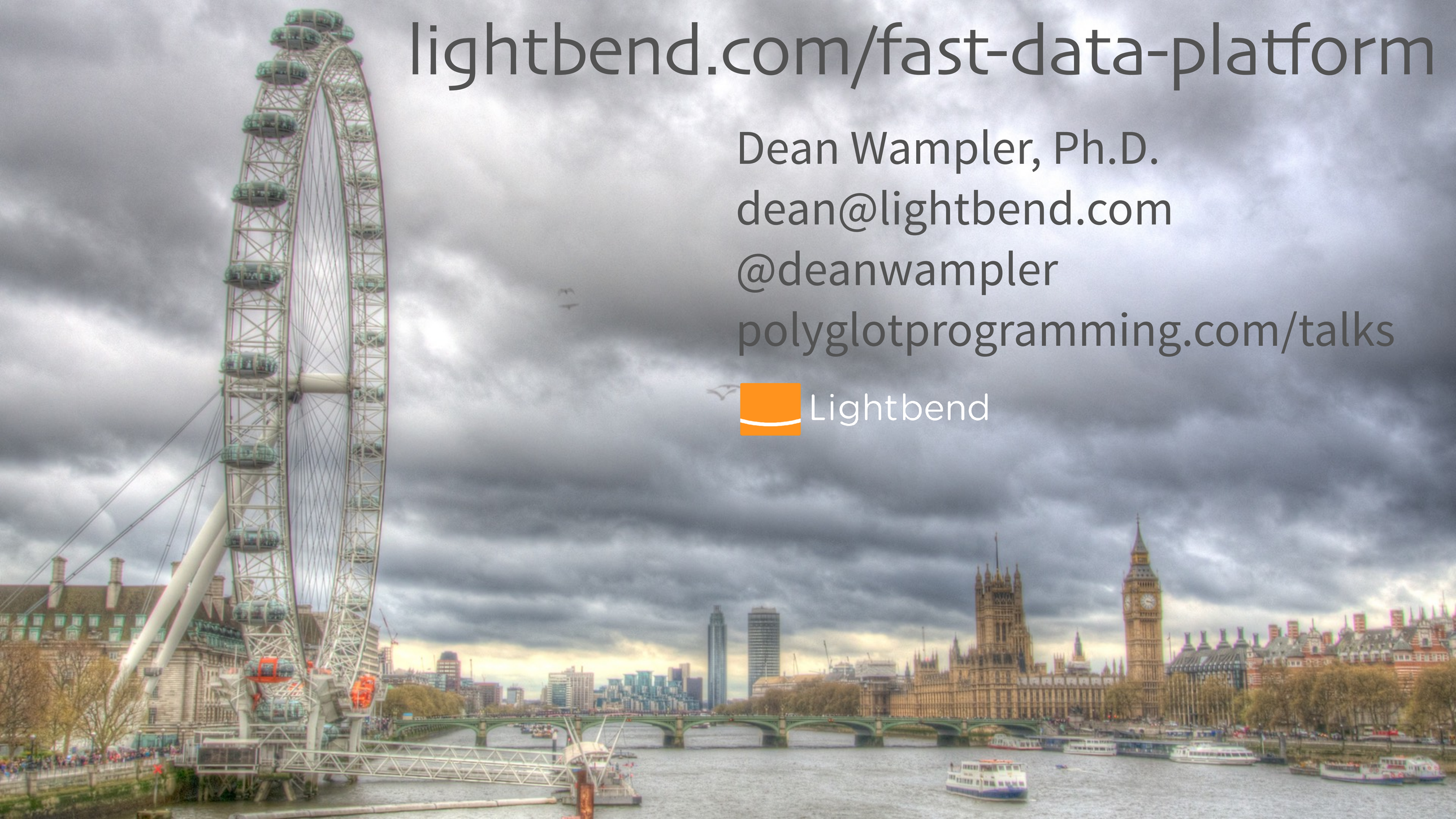


What we discussed

lightbend.com/fast-data-platform


Streaming Engines
akka streams
Spark
Flink
kafka (Kafka Streams)
Microservices
lagom
akka
play
Lightbend Enterprise Suite
Machine Learning
Spark
MLlib
Data Backplane
kafka
Storage
HDFS
Other Storage Options
SQL, NoSQL
Cloud Storage (S3 etc)
Elasticsearch
DC/OS
Intelligent Management & Monitoring and Security
DC/OS Web UI
opsclarity
Other Consoles
REST/HTTP
Sockets
Logs
Plus management & monitoring tools
lightbend.com/fast-data-platform


Streaming Engines
akka streams
Spark
Flink
kafka (Kafka Streams)
Microservices
lagom
akka
play
Lightbend Enterprise Suite
Machine Learning
Spark
MLlib
Data Backplane
kafka
Storage
HDFS
Other Storage Options
SQL, NoSQL
Cloud Storage (S3 etc)
Elasticsearch
DC/OS
Intelligent Management & Monitoring and Security
DC/OS Web UI
opsclarity
Other Consoles
REST/HTTP
Sockets
DC/OS, Kubernetes: On-premise, Cloud, or Hybrid
lightbend.com/fast-data-platform



lightbend.com/fast-data-platform

Dean Wampler, Ph.D.
dean@lightbend.com
@deanwampler
polyglotprogramming.com/talks

Lightbend