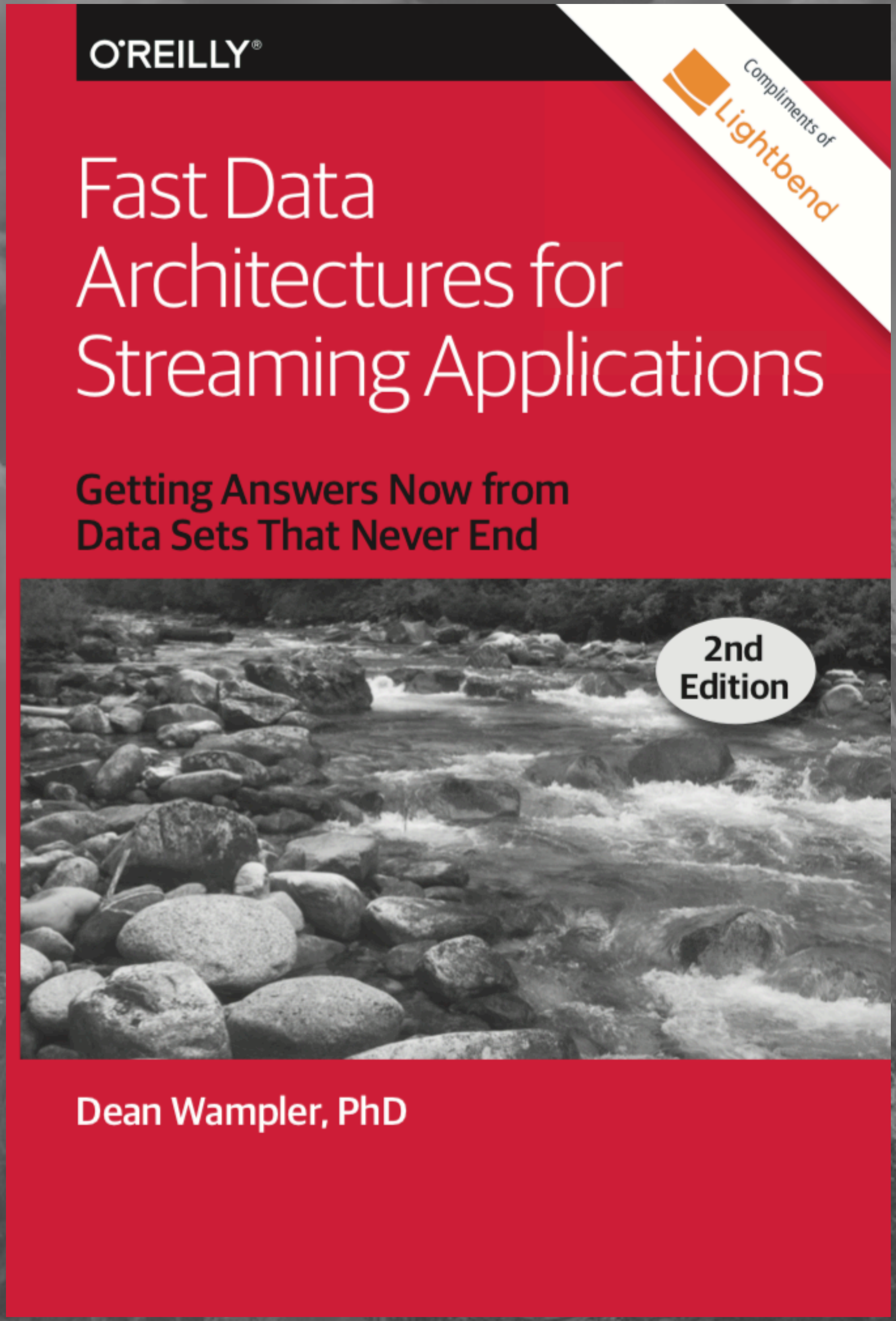
Executive Briefing: What it takes to use machine learning in fast data pipelines

@deanwampler
dean@deanwampler.com
polyglotprogramming.com/talks

Data Streaming, in General

go.lightbend.com/fast-data-architectures-for-streaming-applications-oreilly-2nd-edition



@deanwampler

What We'll Discuss

- Batch vs. streaming… and why
- Data science vs. data engineering
- Serving models in production
  - CI/CD Systems for ML
  - Example architecture
  - Updating Models in Production

Batch vs. streaming… and why

@deanwampler

Telecom

Finance

Medical

Energy

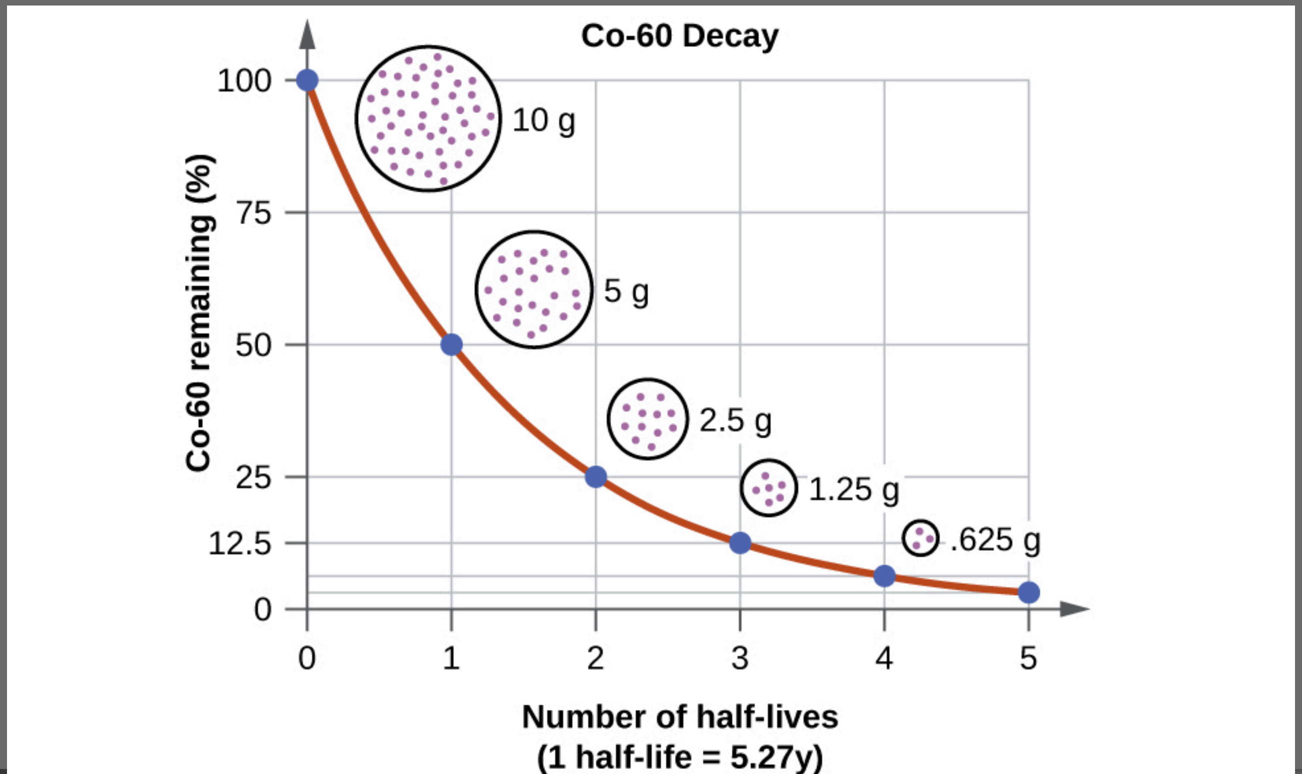
... and IoT

State of the art phone!

Mobile

@deanwampler

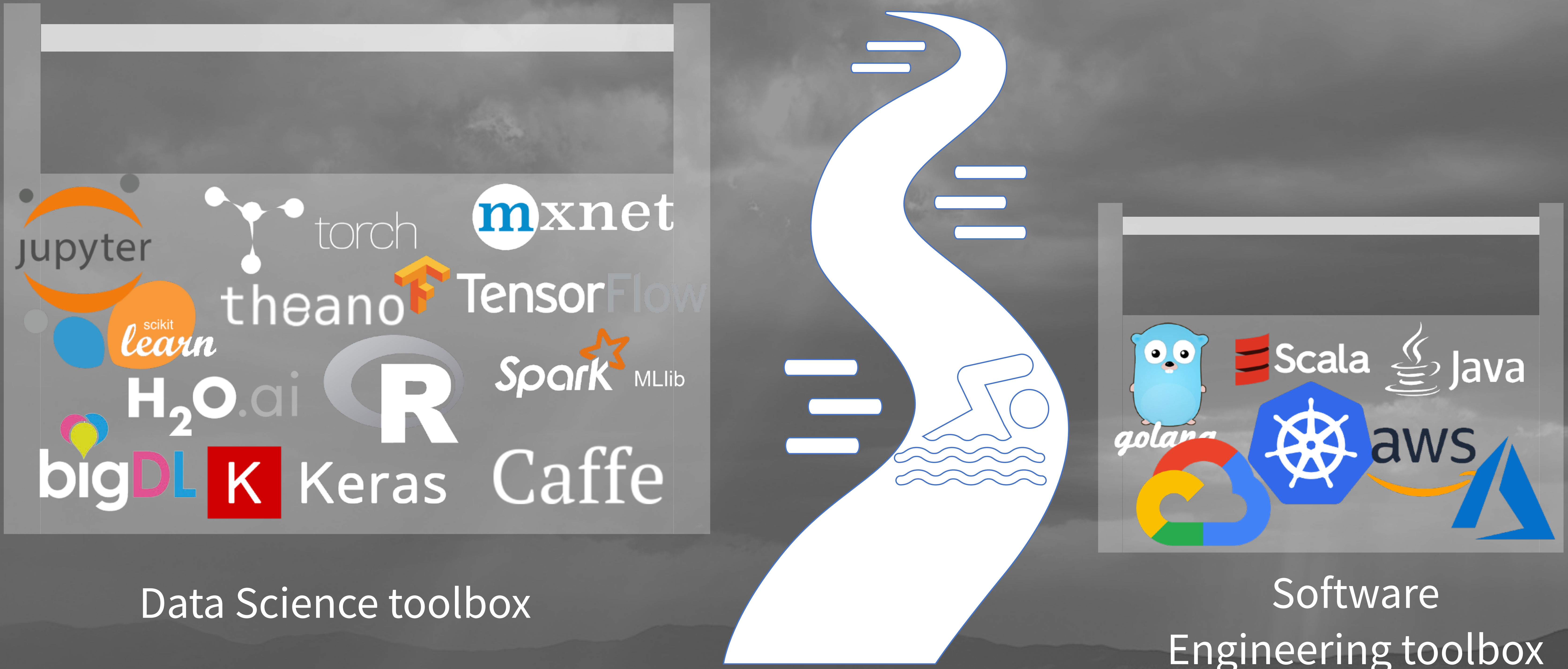
Information value has a half life; it decays with time



@deanwampler

Data Science vs. Data Engineering

@deanwampler



Data Science toolbox

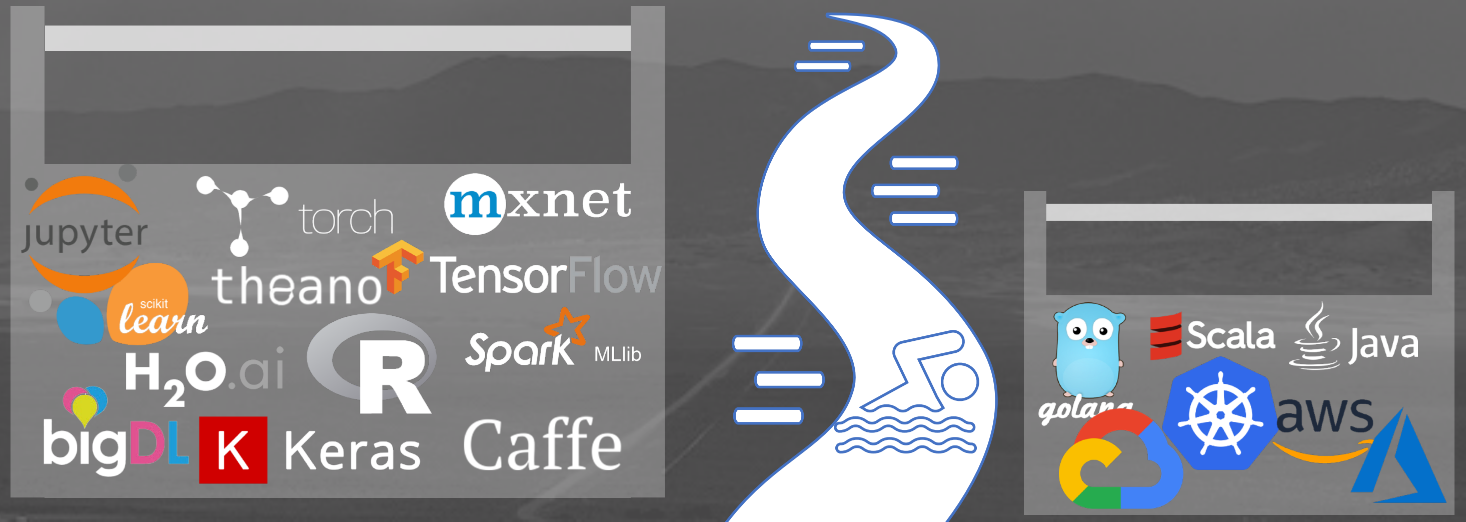
Software
Engineering toolbox

Data Scientists

- Comfortable with uncertainty
- Less process oriented
  - Iterative, experimental

Data Engineers

- Uncomfortable with uncertainty
- Process oriented
  - Agile Manifesto
  - … which does not mention data!



https://derwen.ai/s/6fqt

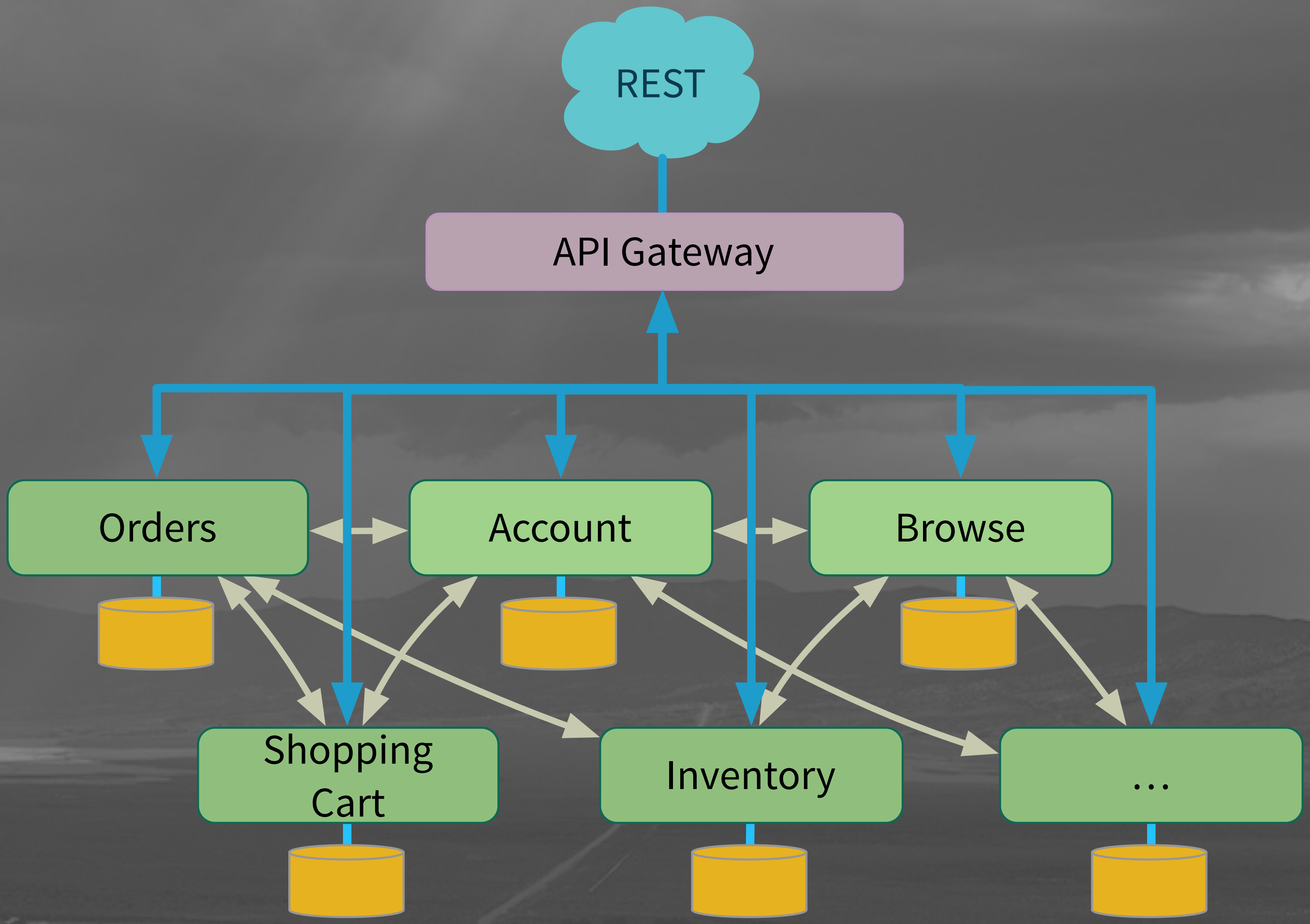
@deanwampler

Streaming Imposes New Requirements

If you run something long enough, all rare problems eventually happen!

- Reliability - fault and “surprise” tolerant
- Availability - “always on”
- Low latency - for some definition of “low”
- Scalability - up and down
- Adaptability - ideally without restarts

In other words: Microservices

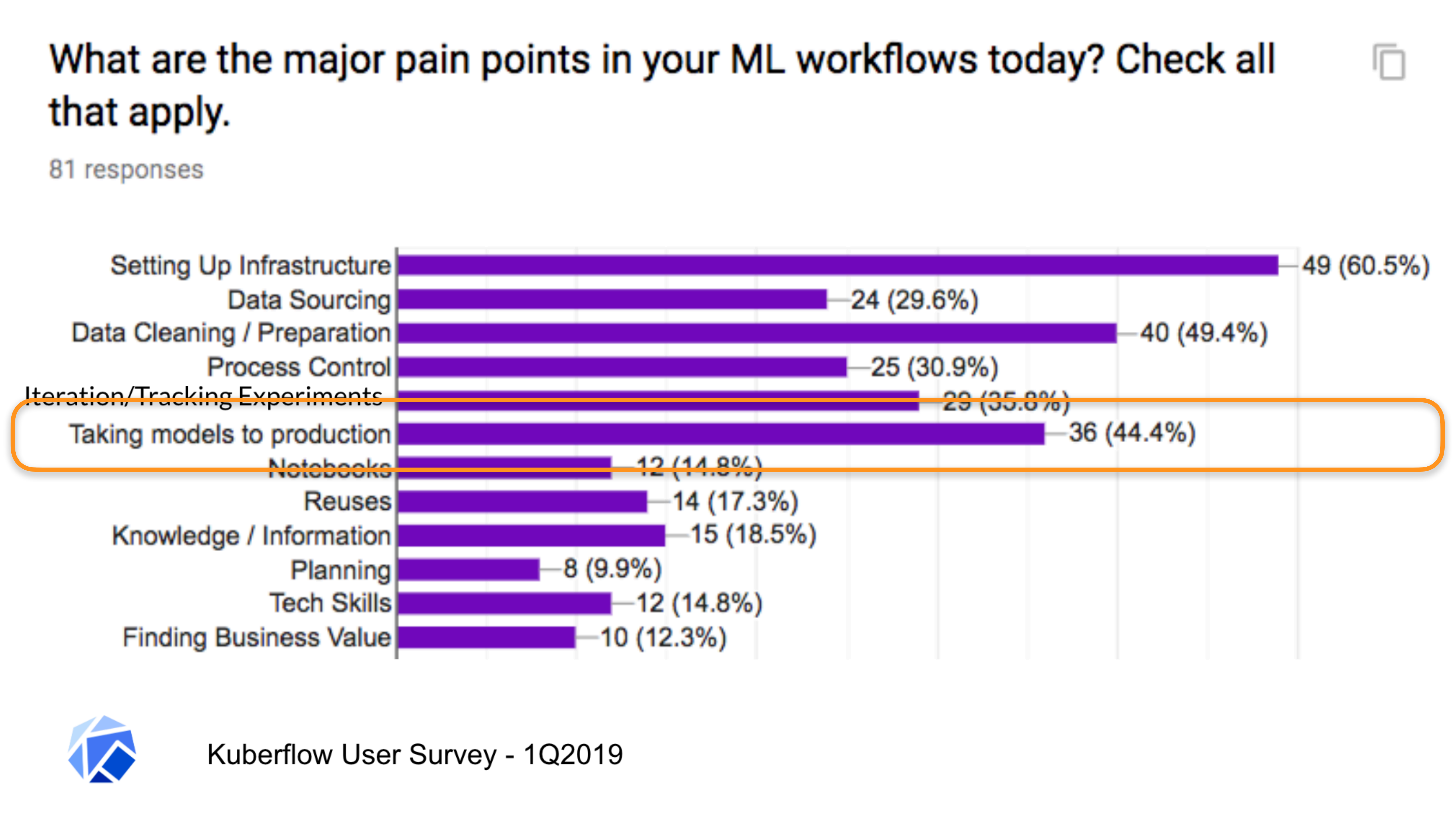


@deanwampler

Serving Models in Production

@deanwampler

A Recent Kubeflow User Survey



From this Kubeflow Overview

@deanwampler

Lack of Tool/Process Integration

- ~60% worry about missed opportunities
- ~50% worry about loss of data team productivity
- ~45% worry about slow time-to-market
- ~40% worry about customer dissatisfaction

From a recent Lightbend survey

Can You Answer this Question?

- Why did the model reject that loan application?

(After you've been sued for discrimination...)

Which model was it?

- Which version of the model was used?
- How was it trained?
- When was this model deployed?
- … and other questions you'll need to answer to understand what happened…


CI/CD for ML?
a.k.a "MLops"
@deanwampler

CI/CD Process Required (1/4)

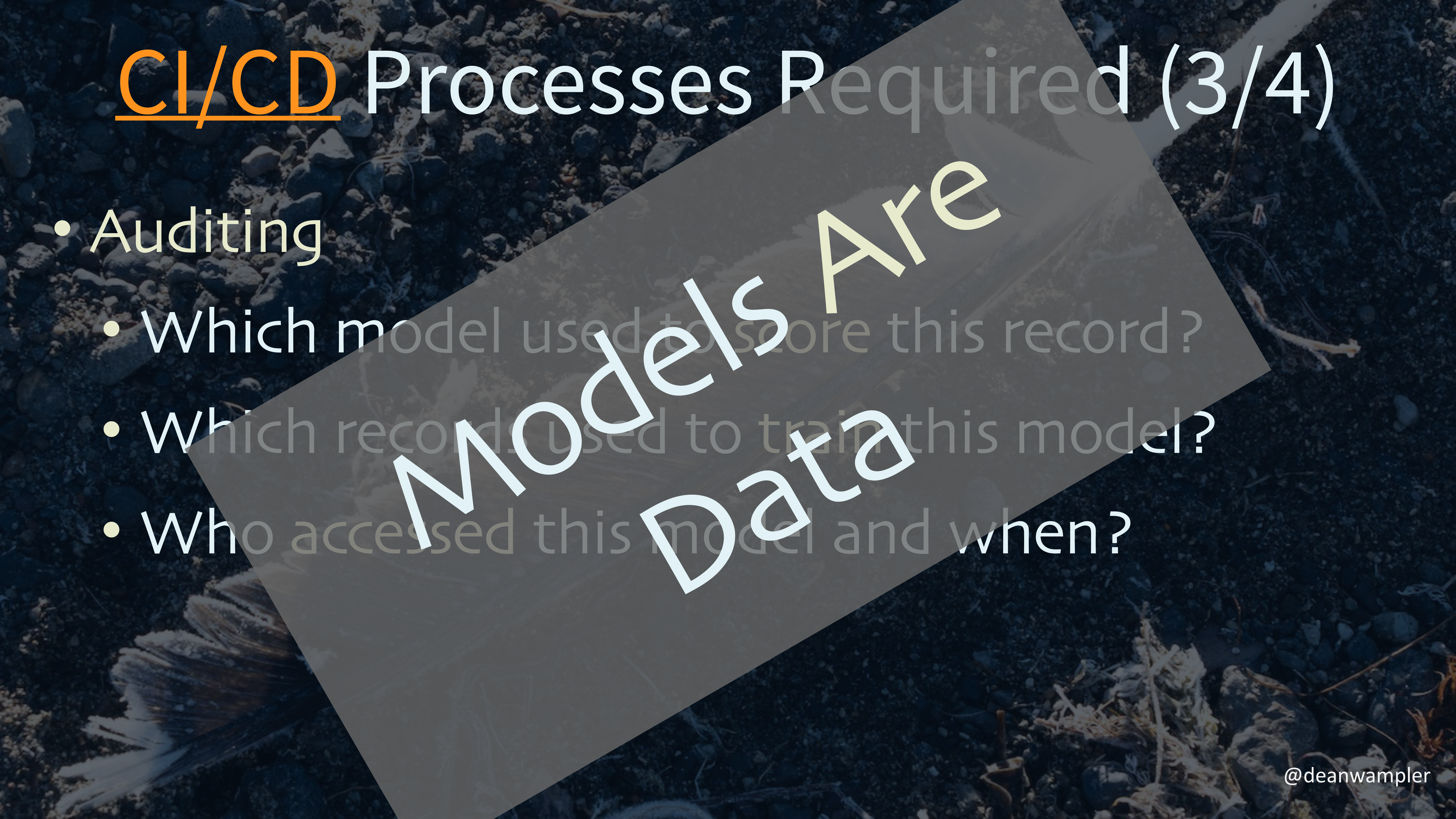
- Version control - for models and code
- Automation - builds, tests, quality checks, artifact management & delivery
  - Necessary for reproducibility

CI/CD Process Required (2/4)

- Supports different launch configurations:
  - "dark" launches
  - A/B, Canary, and other testing scenarios

CI/CD Processes Required (3/4)

- Auditing
  - Which model used to score this record?
  - Which records used to train this model?
  - Who accessed this model and when?


CI/CD Processes Required (3/4)
• Auditing
• Which model used to score this record?
• Which records used to train this model?
• Who accessed this model and when?
Models Are Data
@deanwampler

CI/CD Processes Required (3/4)

- Auditing
  - Which model used to score this record?
  - Which records used to train this model?
  - Who accessed this model and when?

GDPR - What if a customer asks you to delete their data? Do you also delete the models trained with that data?

@deanwampler

CI/CD Processes Required (4/4)

- Monitoring
  - Resource utilization changes?
  - Quality metrics:
    - Match performance during training?
    - Concept drift?

What's Different from Microservice CI/CD?

- AutoML
- Data safety and lineage
- Model fairness and reproducibility
- Model and feature artifact management

https://www.oreilly.com/ideas/9-ai-trends-on-our-radar

What's Different from Microservice CI/CD?

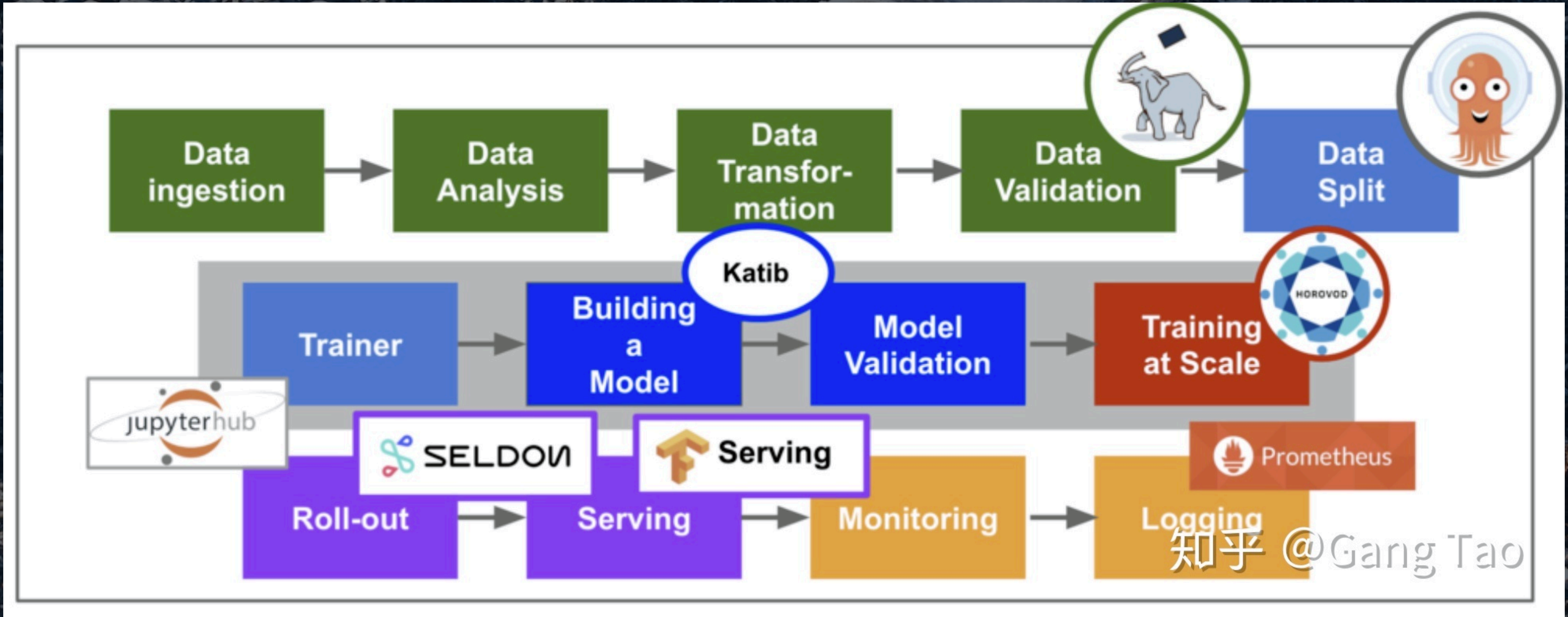
- CI/CD prefers deterministic measures of quality. How should you support the extra statistical indeterminacy data science introduces?

CI/CD Suites for ML

- Kubeflow - for Kubernetes
- SageMaker - for AWS users
- MLFlow - from the Spark community
- … plus emerging vendors

@deanwampler

Systems - Kubeflow

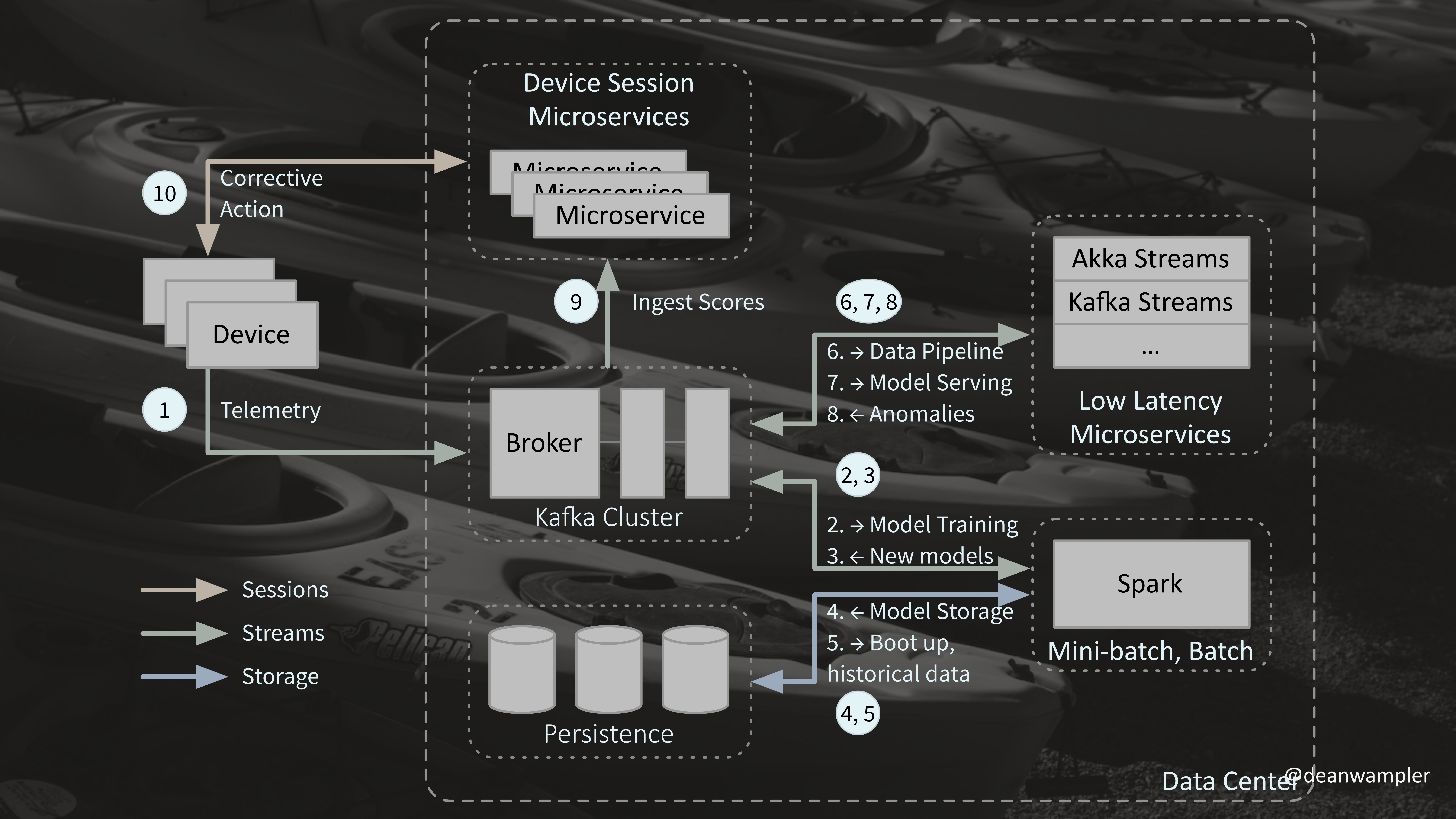


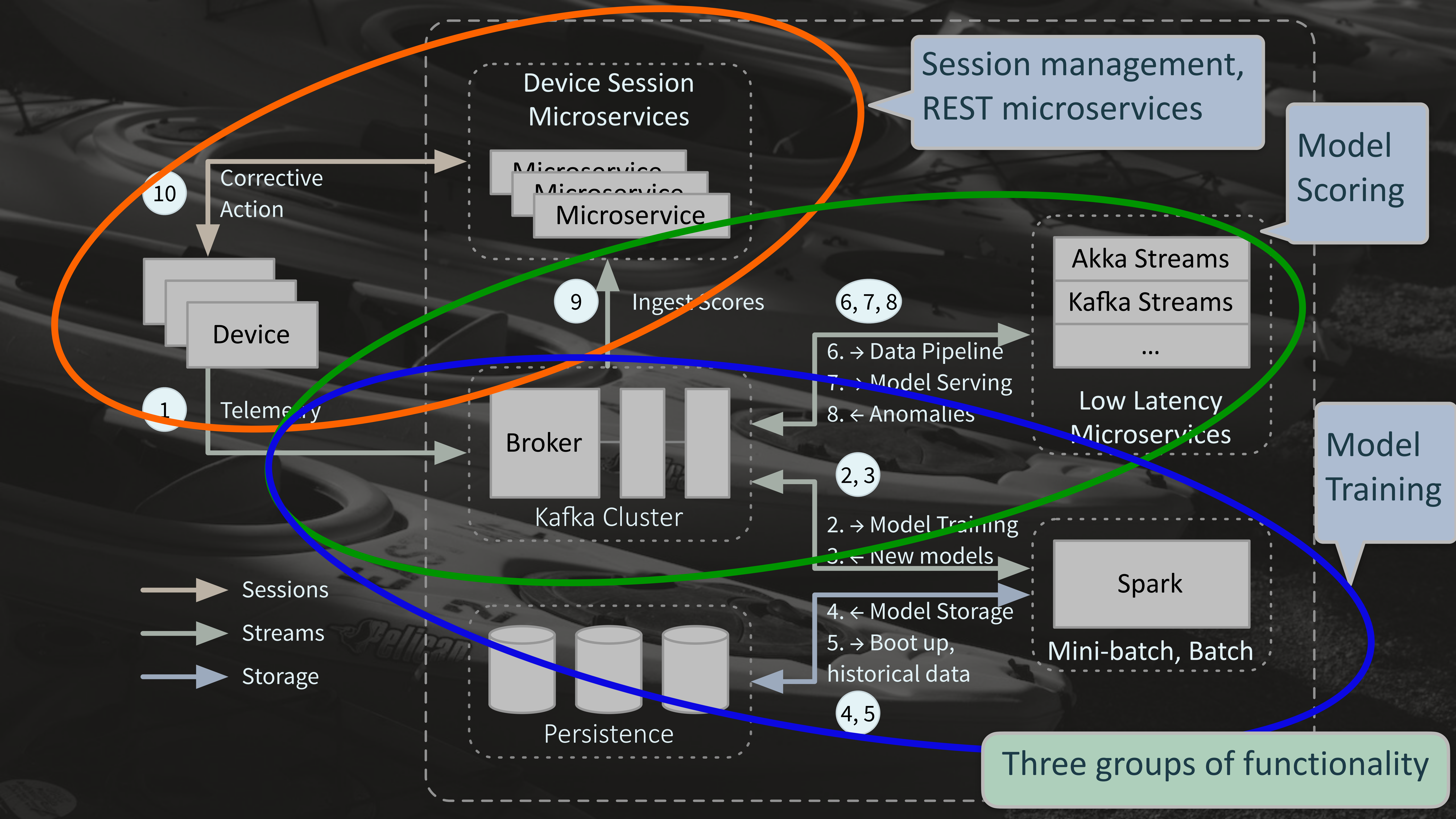
Example Architectures

@deanwampler

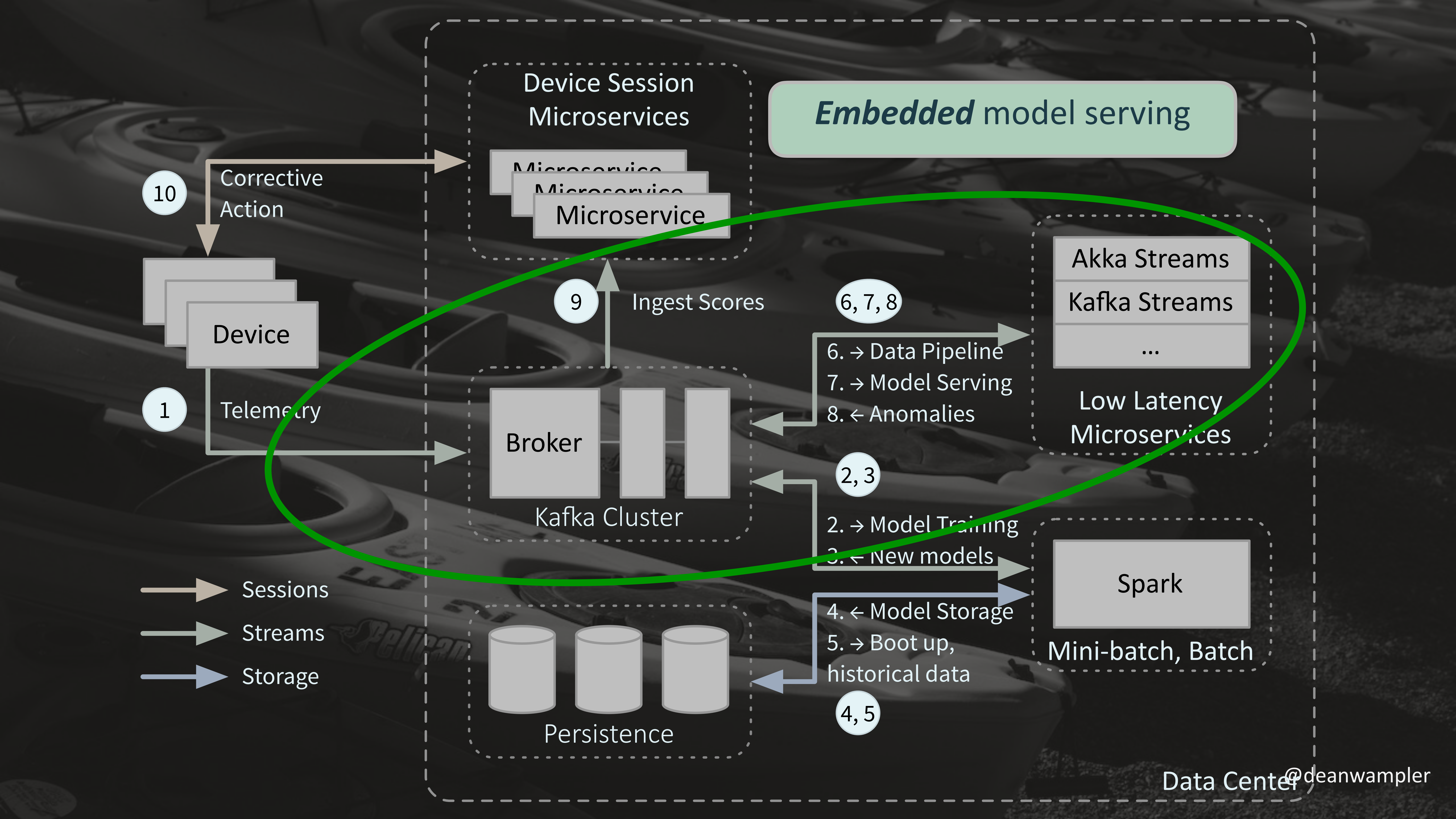
Example Architectures

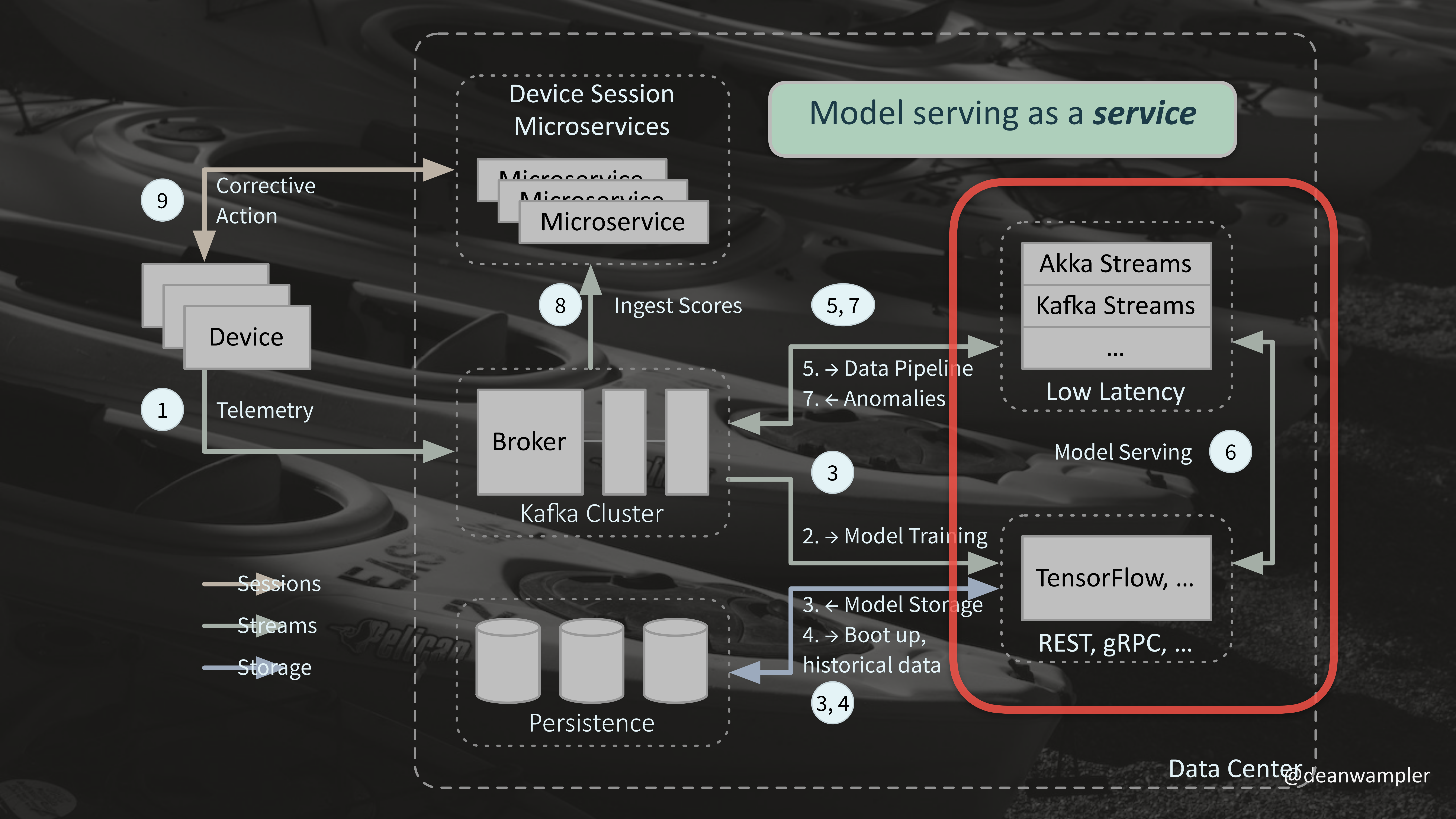
Timely Information Integrated with Your Apps


Device Session
Microservices
Microservice
Microservice
Microservice
10
Corrective
Action
Device
9
Ingest Scores
6, 7, 8
Akka Streams
Kafka Streams
...
Low Latency
Microservices
6. → Data Pipeline
7. → Model Serving
8. ← Anomalies
1
Telemetry
Broker
Kafka Cluster
2, 3
2. → Model Training
3. ← New models
Spark
Mini-batch, Batch
4. ← Model Storage
5. → Boot up,
historical data
4, 5
Sessions
Streams
Storage
Persistence
Data Center
@deanwampler



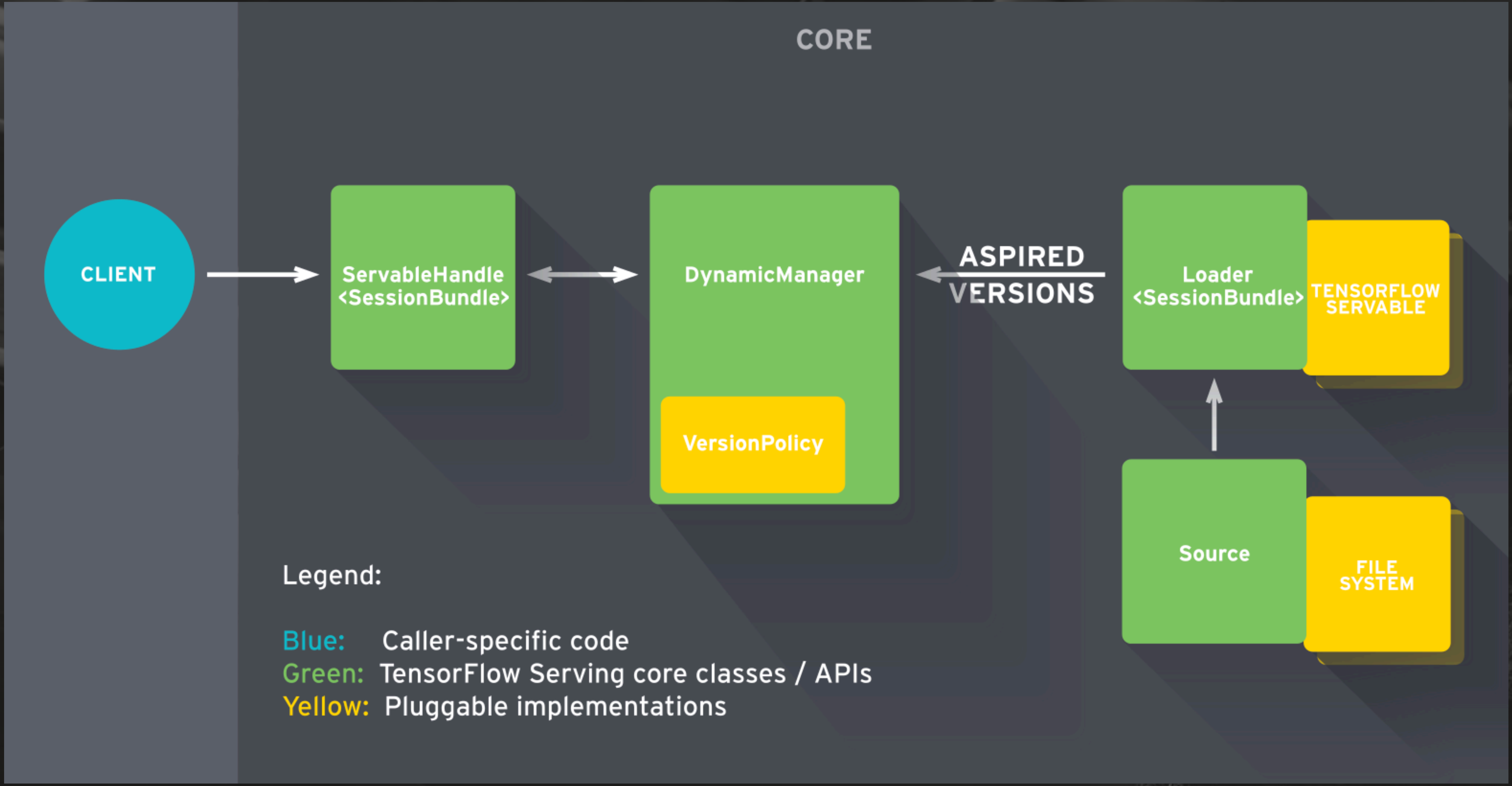
Session management, REST microservices
Model Scoring
Model Training
Device Session Microservices
Microservice
Microservice
Microservice
10
Corrective Action
Device
1
Telemetry
9
Ingest Scores
6, 7, 8
6. → Data Pipeline
7. → Model Serving
8. ← Anomalies
Akka Streams
Kafka Streams
...
Low Latency Microservices
Broker
Kafka Cluster
2, 3
2. → Model Training
3. ← New models
Spark
Mini-batch, Batch
4. ← Model Storage
5. → Boot up, historical data
4, 5
Persistence
Sessions
Streams
Storage
Three groups of functionality


Embedded model serving
Device Session Microservices
Microservice
Microservice
Microservice
10
Corrective Action
Device
1
Telemetry
9
Ingest Scores
Broker
Kafka Cluster
6, 7, 8
6. → Data Pipeline
7. → Model Serving
8. ← Anomalies
Akka Streams
Kafka Streams
...
Low Latency Microservices
2, 3
2. → Model Training
3. ← New models
Spark
Mini-batch, Batch
4. ← Model Storage
5. → Boot up, historical data
4, 5
Persistence
Sessions
Streams
Storage
Data Center
@deanwampler



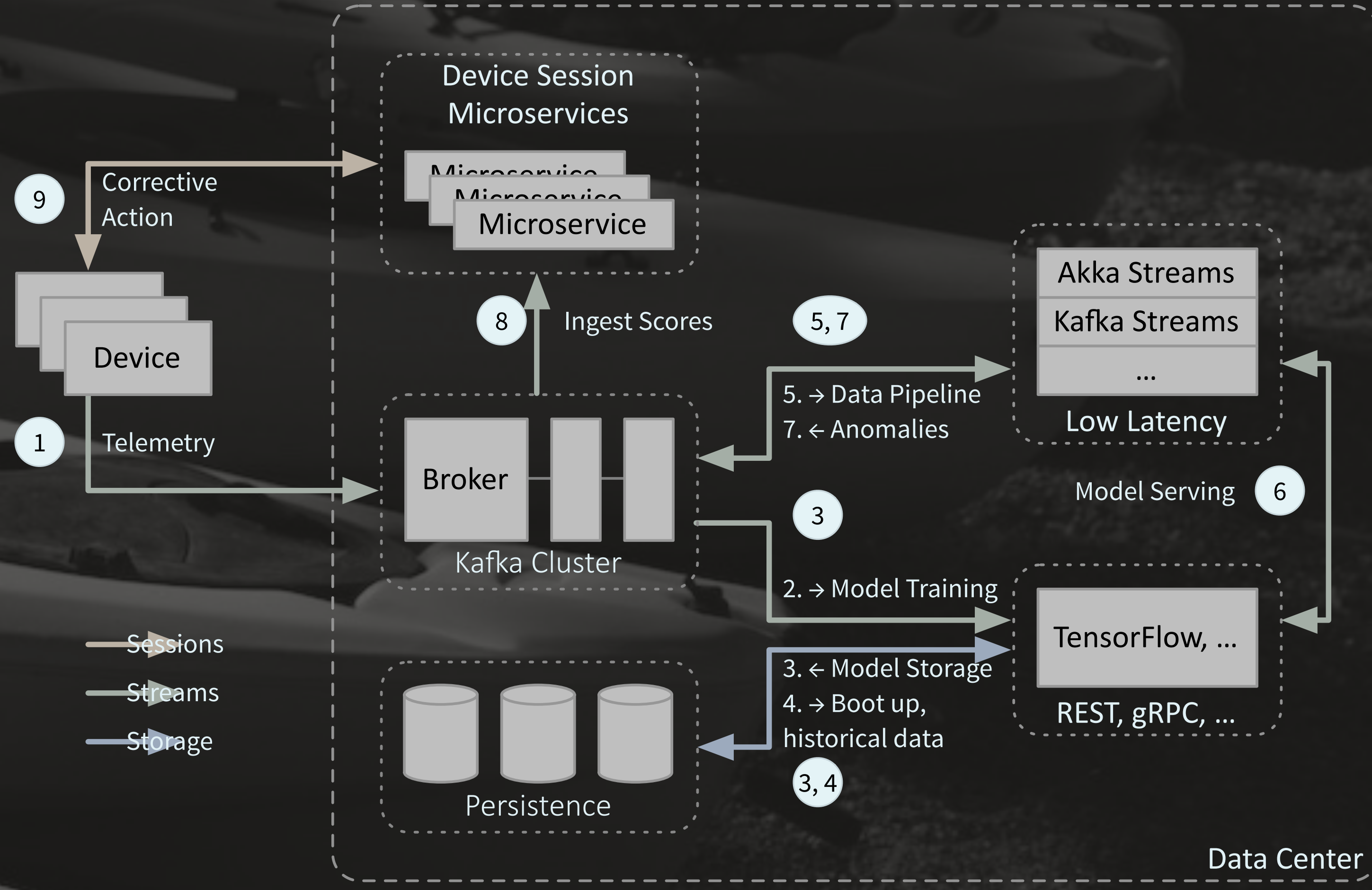
Model serving as a *service*
Device Session
Microservices
Microservice
Microservice
Microservice
9
Corrective
Action
Device
1
Telemetry
8
Ingest Scores
Broker
Kafka Cluster
5, 7
5. → Data Pipeline
7. ← Anomalies
Akka Streams
Kafka Streams
…
Low Latency
Model Serving
6
3
2. → Model Training
TensorFlow, …
REST, gRPC, …
3. ← Model Storage
4. → Boot up,
historical data
3, 4
Sessions
Streams
Storage
Persistence
Data Center
@deanwampler

TensorFlow Serving



Model Serving as a Service

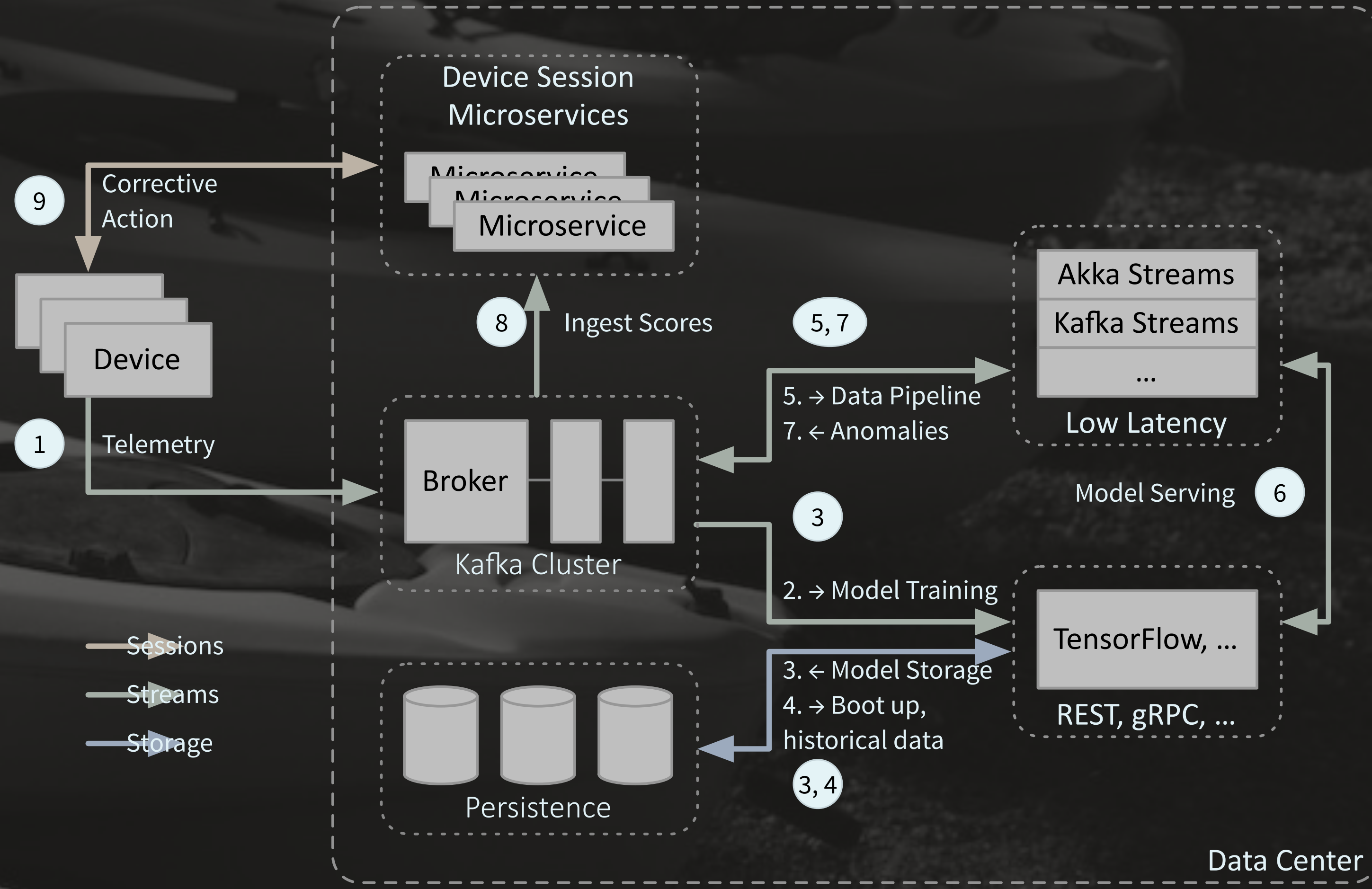
- Pros:
  - A familiar integration pattern
  - Decouples “concerns”: AI tools, scaling, upgrading, …
  - One system for training and scoring



@deanwampler

Model Serving as a Service

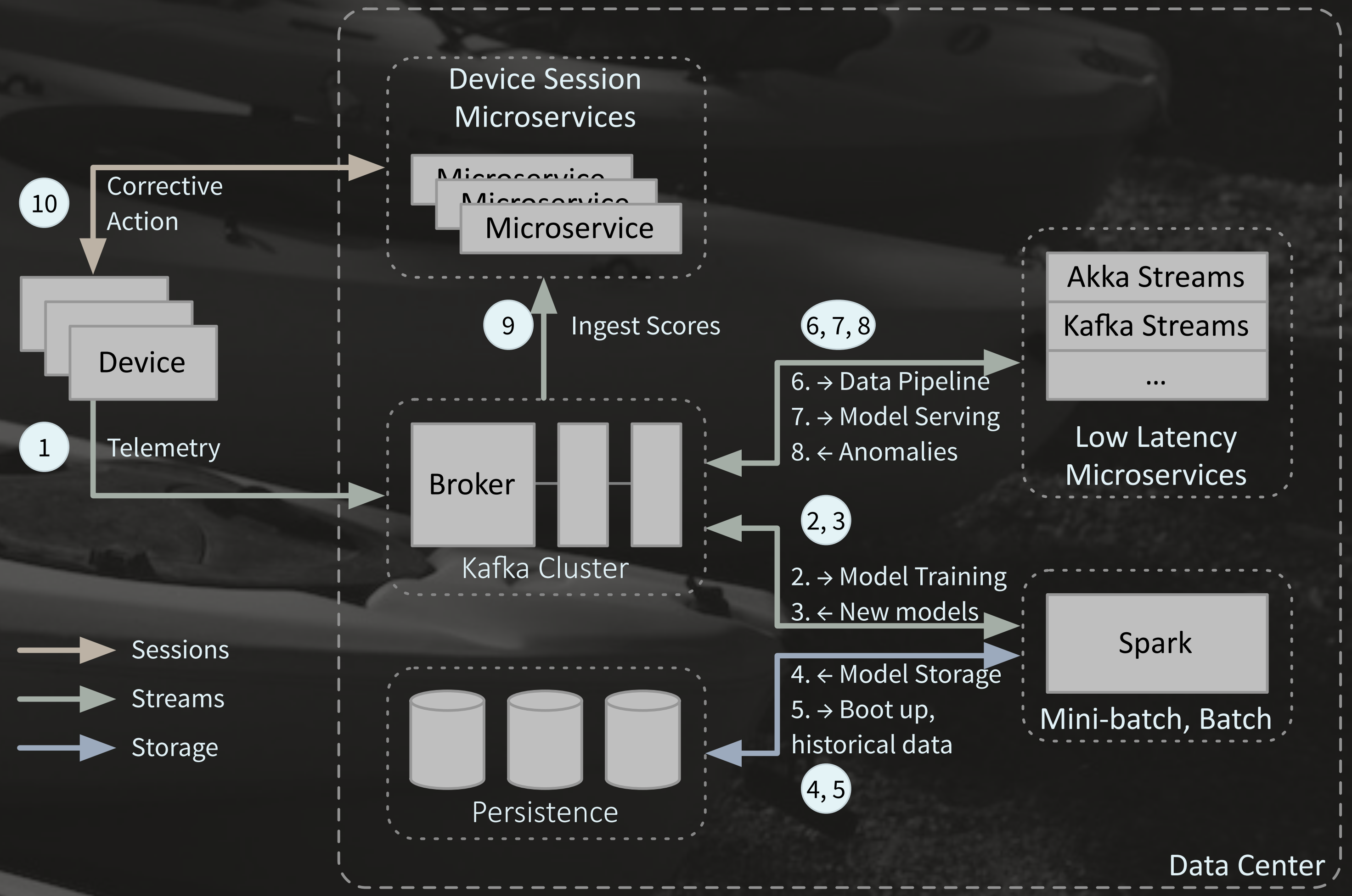
- Cons:
  - Overhead of invocation, e.g., REST
  - ML Pipeline becomes a unique production work flow



@deanwampler

Embedded Model Serving

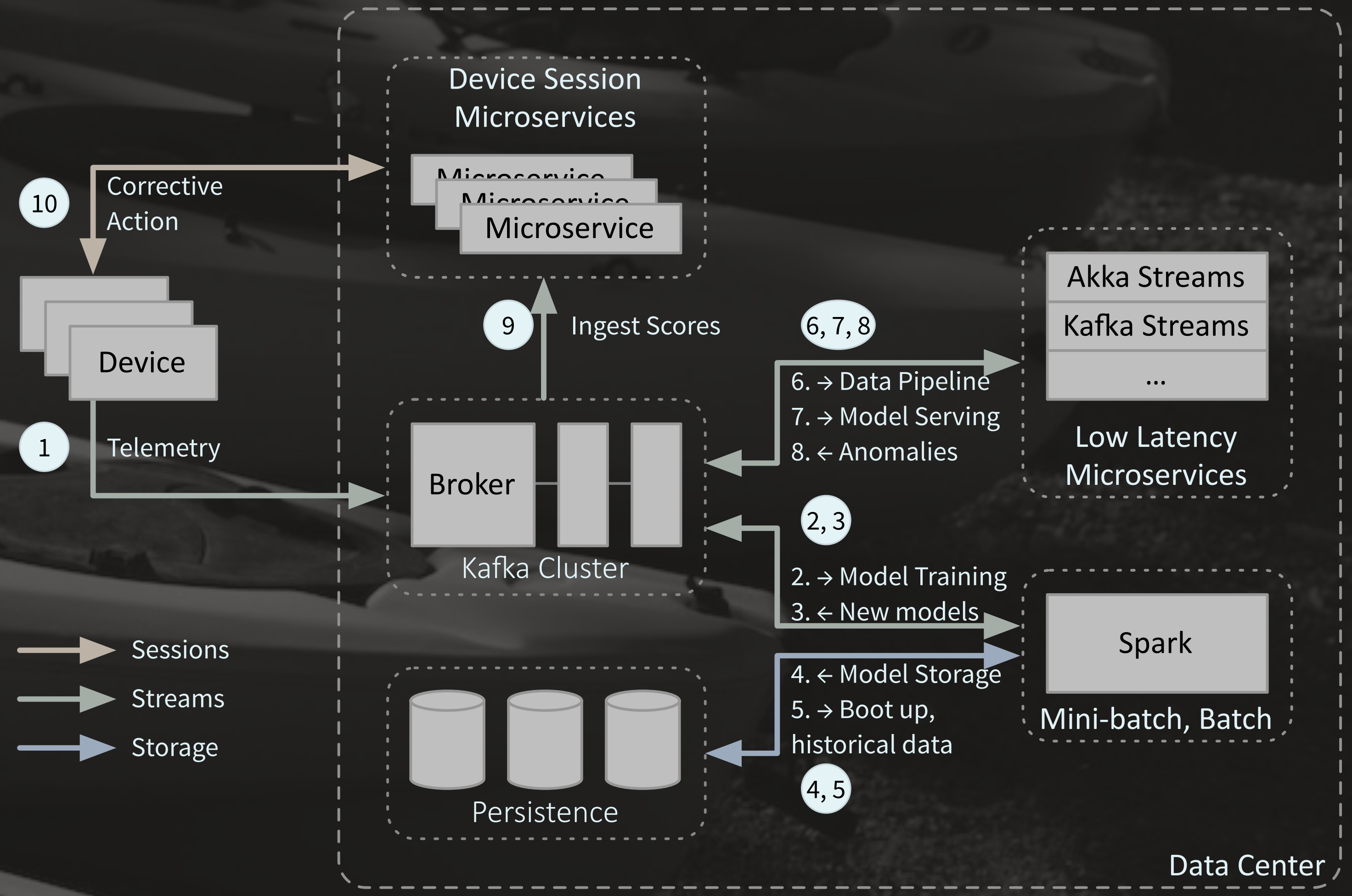
- Pros:
  - Lowest scoring overhead - interprocess communication only used for model updates
  - Performance tuning focuses on one system, the data pipeline



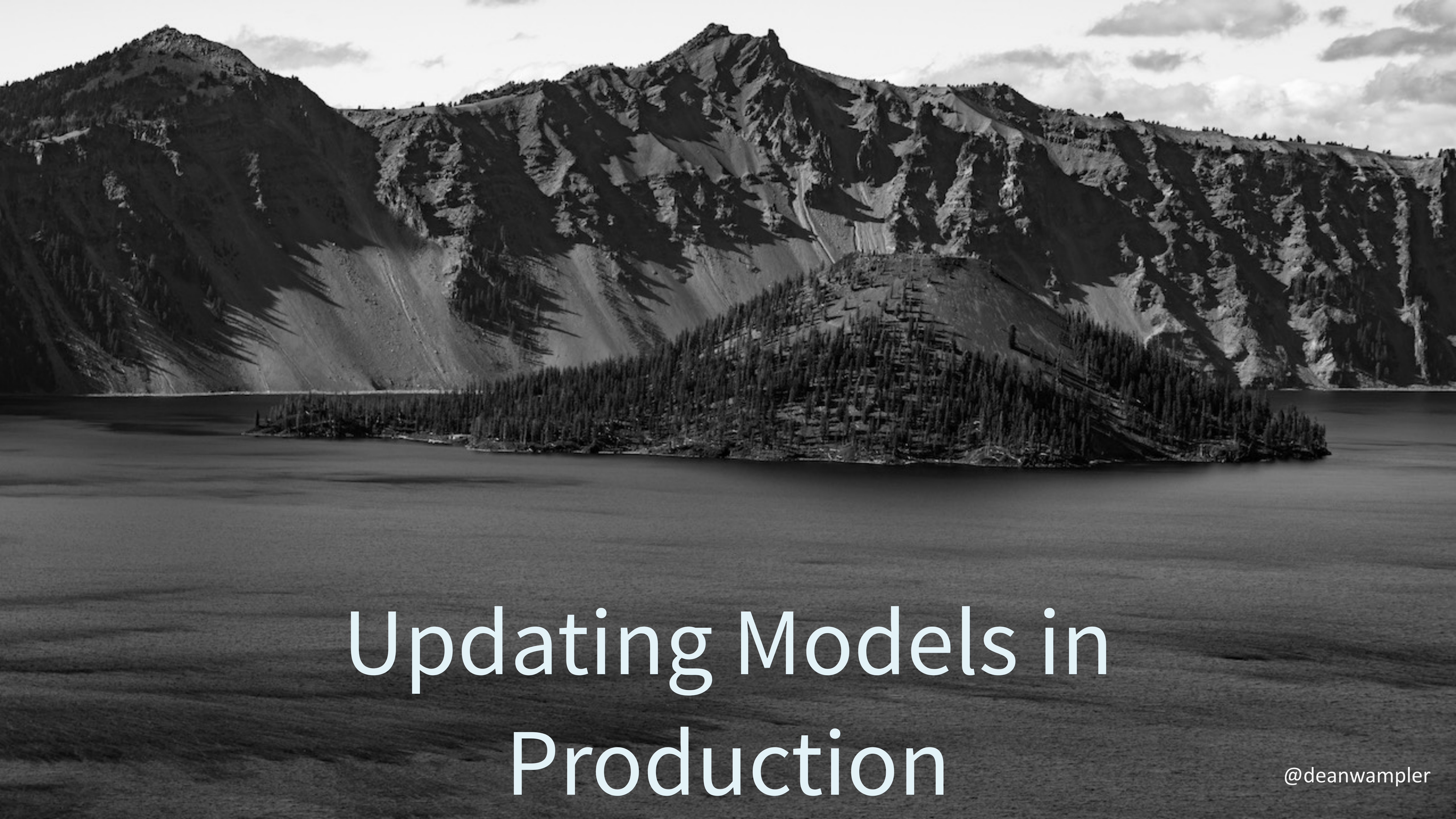
@deanwampler

Embedded Model Serving

- Cons:
  - Model parameters must be serialized
  - More complexity
  - Model serving library must be “compatible” with training system



@deanwampler



Updating Models in Production

@deanwampler

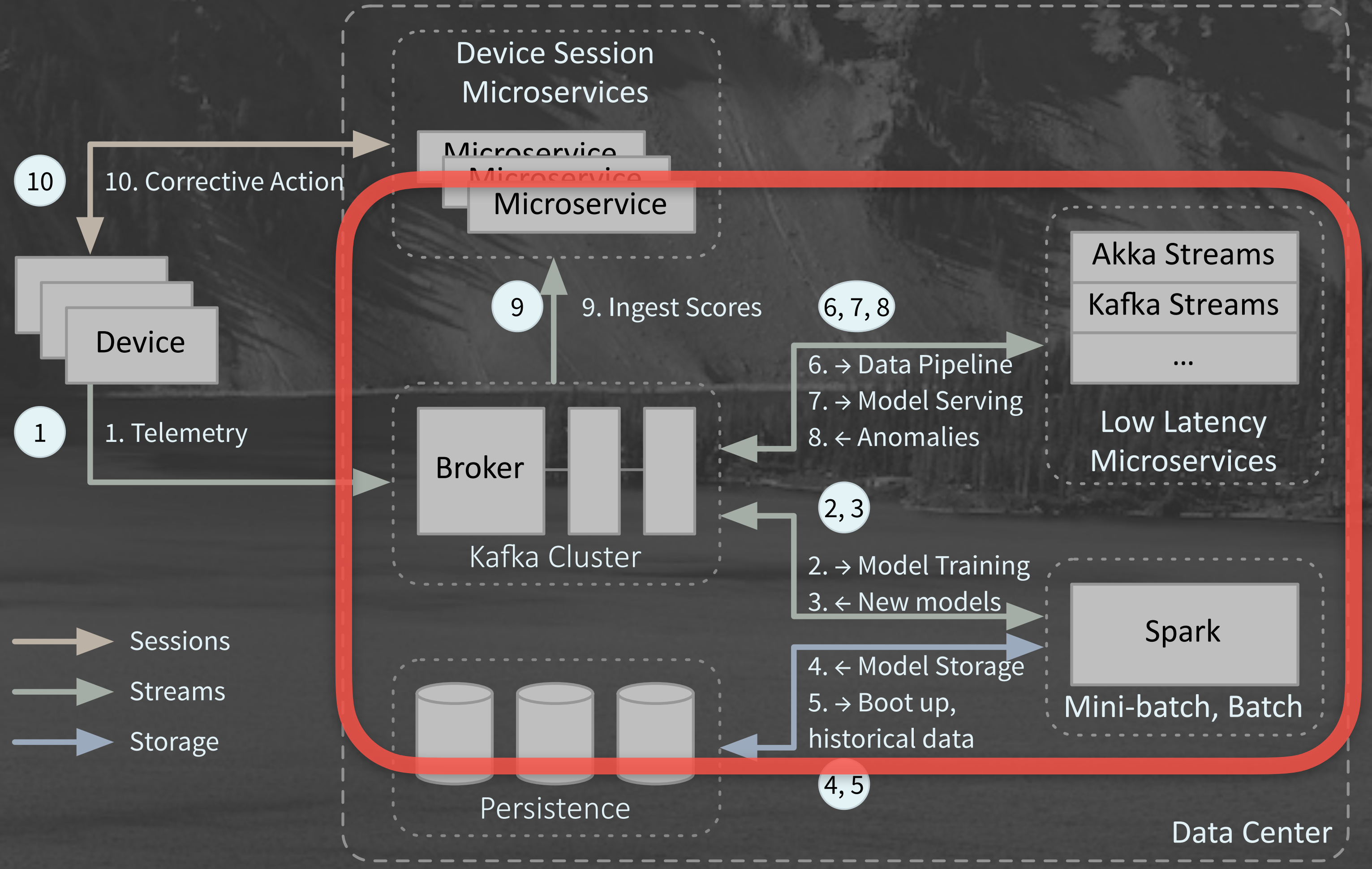
Model Updates

- Concept Drift - models grow stale
  - They have a half life, too
  - So, periodically retrain, then serve the new model, ideally without downtime

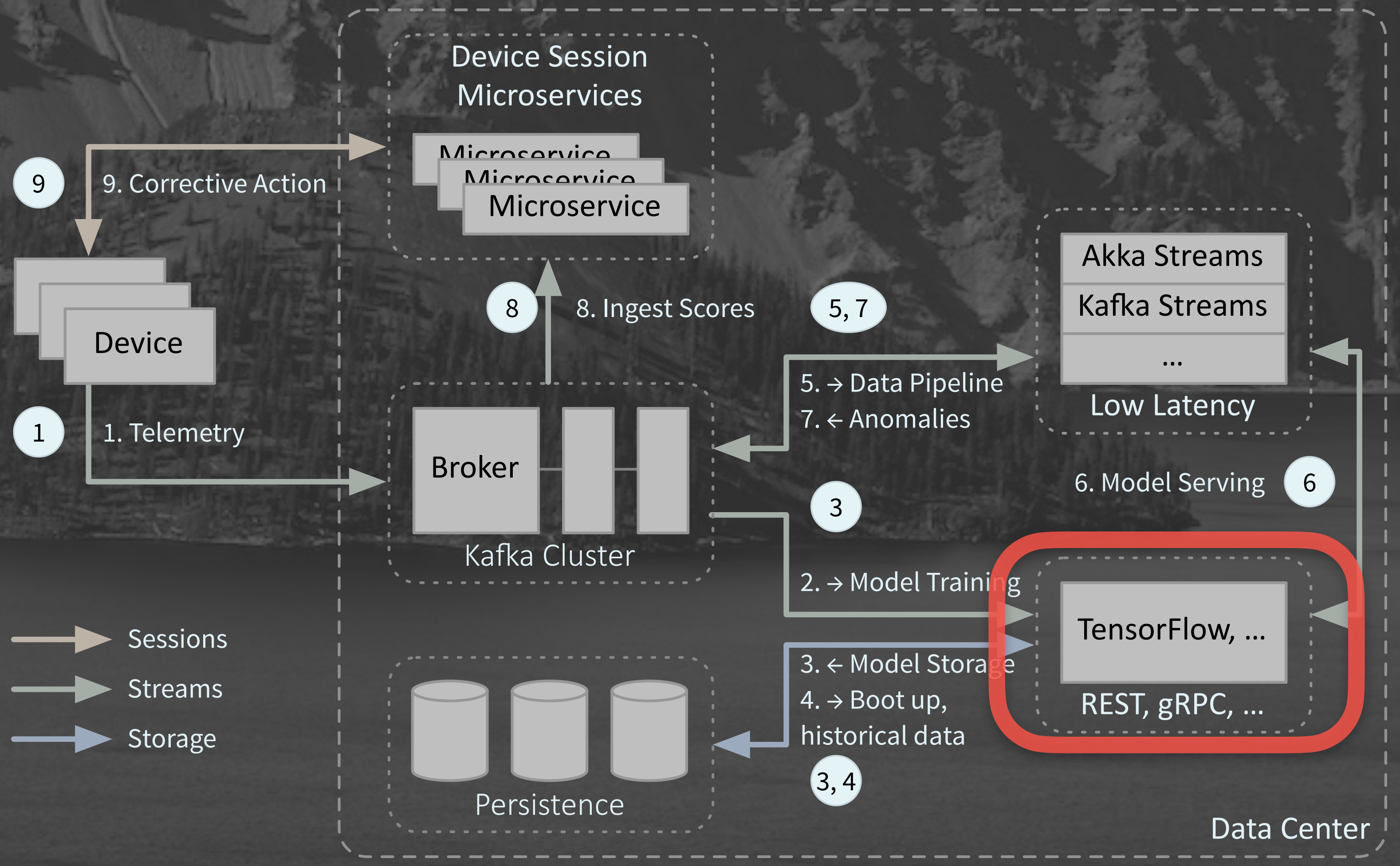
@deanwampler

Retraining Considerations

- How do you measure model quality?
- What's the trade-off between model performance vs. retraining cost?
- How far back in the data set do you go when training?



Complex to update embedded models!



Model updates can be straightforward

@deanwampler

Auditing

- Kind of Model
- Parameters and hyperparameters
- When trained
- Data used for training
- When deployed, undeployed, etc.
- …

@deanwampler

Auditing

- Quality metrics
- Serving metrics (how many records, scoring times…)
- Provenance of decision to retrain
  - The metrics gathered above that were used to decide when to retrain


Dusty Milky Way
Mars last Summer
@deanwampler

References

- Ideas:
  - Ben Lorica on 9 AI Trends
  - Paco Nathan's Data Governance Talk

You can get these slides with the links here:
polyglotprogramming.com/talks

References

- Ideas:
  - O'Reilly Radar: [Data](), [AI](), [others]()
  - [distill.pub]()
  - [The Algorithm]()
  - [The Gradient]()

References

- A few research papers, etc.
  - Incremental training
    - an example
  - Continual learning
  - Explainability

References

- Kubeflow
- MLFlow
- DVC
- AWS SageMaker
- Fiddler (explainable AI)

References

- General Information about Stream Processing
  - My O'Reilly Report on Architectures
  - Streaming Systems Book
  - Stream Processing with Apache Spark
  - Designing Data-Intensive Apps book

References

- Other Talks
    - Strata Talk on ML in a Streaming Context
    - Stream All the Things! (video)
    - Streaming Microservices with Akka Streams and Kafka Streams (video)

References

- Tutorials
  - Model serving in streams
  - Stream processing with Kafka and microservices

Questions?

@deanwampler
dean@deanwampler.com
polyglotprogramming.com/talks