Western Digital

Flash Storage Complementing a Data Lake for Real-Time Insight

Dr. Sanhita Sarkar
Global Director, Analytics Software Development

August 7, 2018

Flash Memory Summit 2018, Santa Clara, CA

Agenda

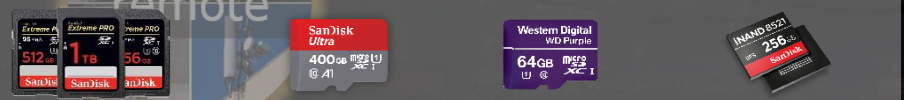
1. Delivering insight along the entire spectrum from edge to core
2. Coupling big data with fast data: complementing a data lake for real-time insight
3. Stream ingestion to ActiveScale™ object storage system as a component within a data lake
4. IoT speed layer – implementation
5. A real-world IoT use case

Evolving Landscape from Core to Edge

Evolving Landscape from Core to Edge

platforms & systems

embedded & removable

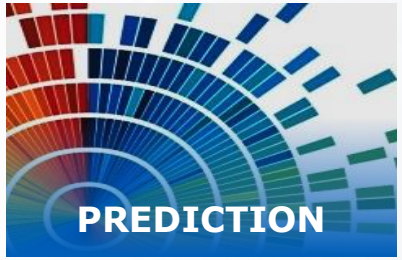


solid state & hard disk drives

Delivering Value to Data from Core to Edge

Enable tight coupling between Big Data and Fast Data

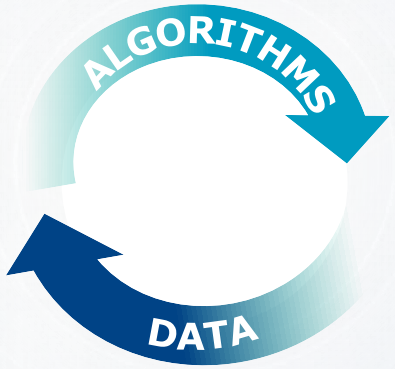
Big Data



Scale

Data Aggregation

Streaming Analytics



Batch Analytics

Machine Learning

Modeling

Artificial Intelligence

Fast Data

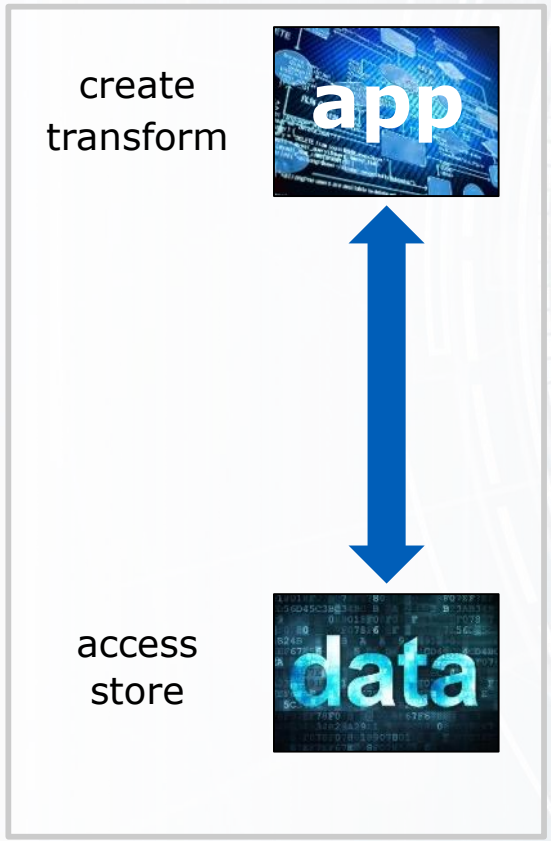
Performance

Delivering Value to Data from Core to Edge

Enable data sharing and utilization of system resources across diverse applications

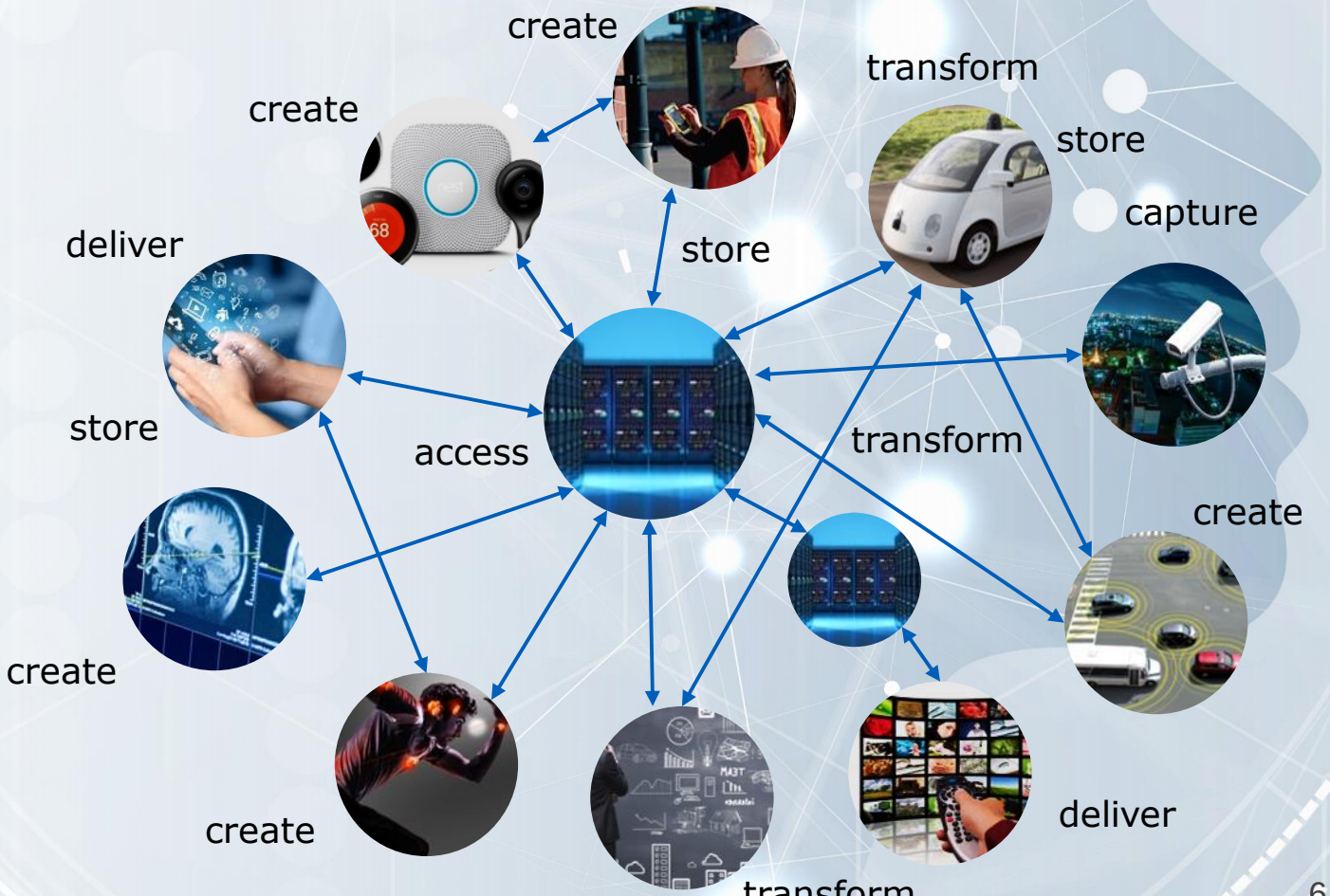
Past

Data Held Captive by Single Application



Current and Future

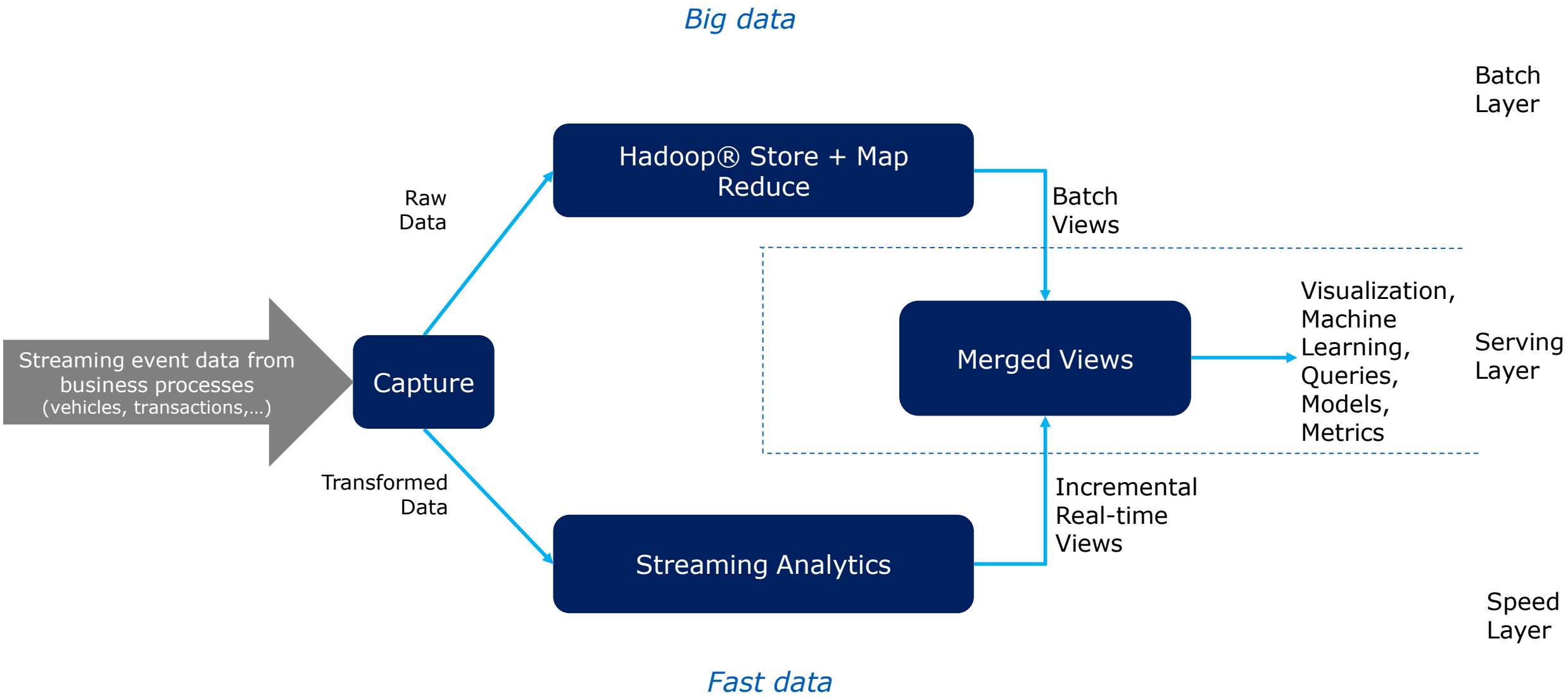
Data Pooled and Shared by Multiple Applications



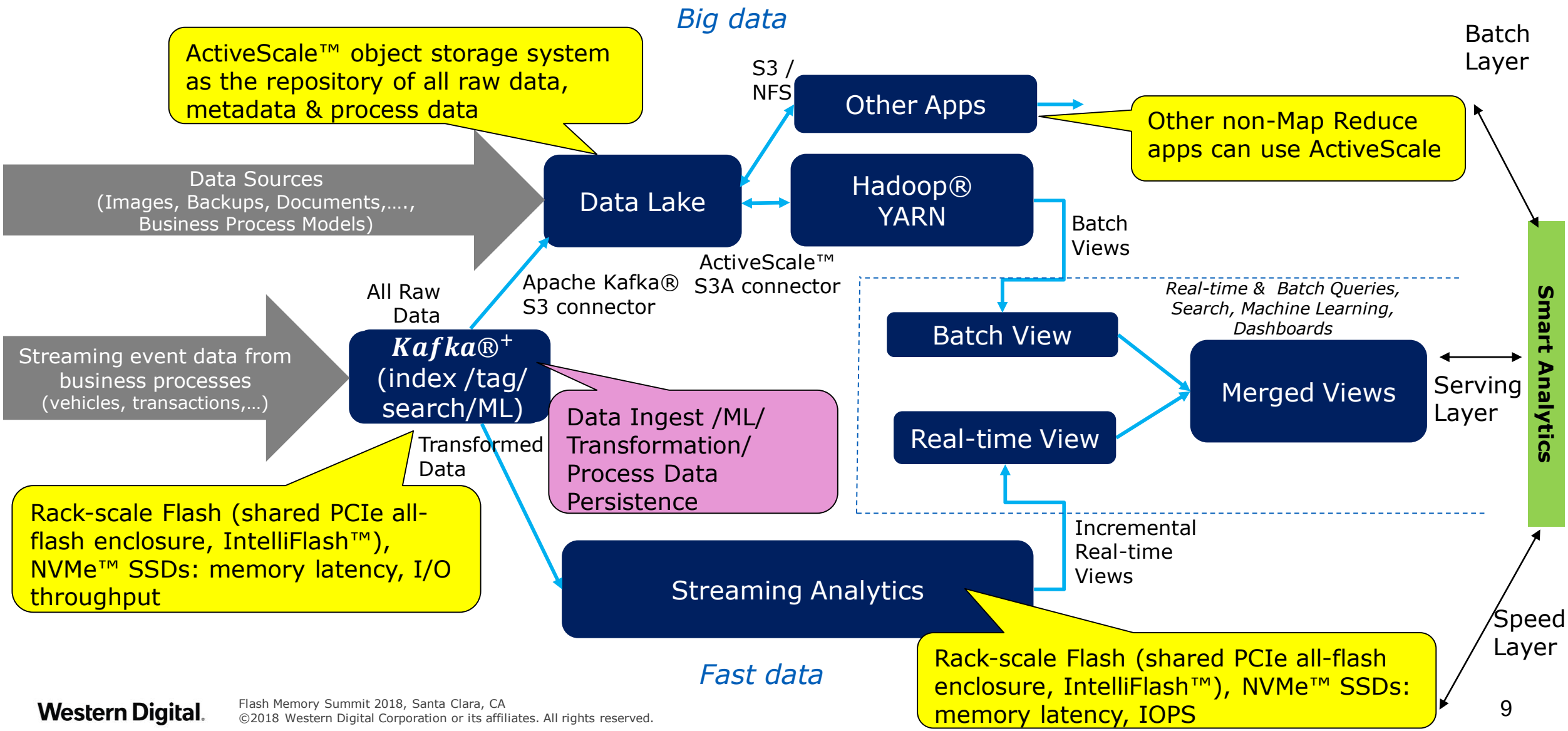
Agenda

1. Delivering insight along the entire spectrum from edge to core
2. **Coupling big data with fast data: complementing a data lake for real-time insight**
3. Stream ingestion to ActiveScale™ object storage system as a component within a data lake
4. IoT speed layer – implementation
5. A real-world IoT use case

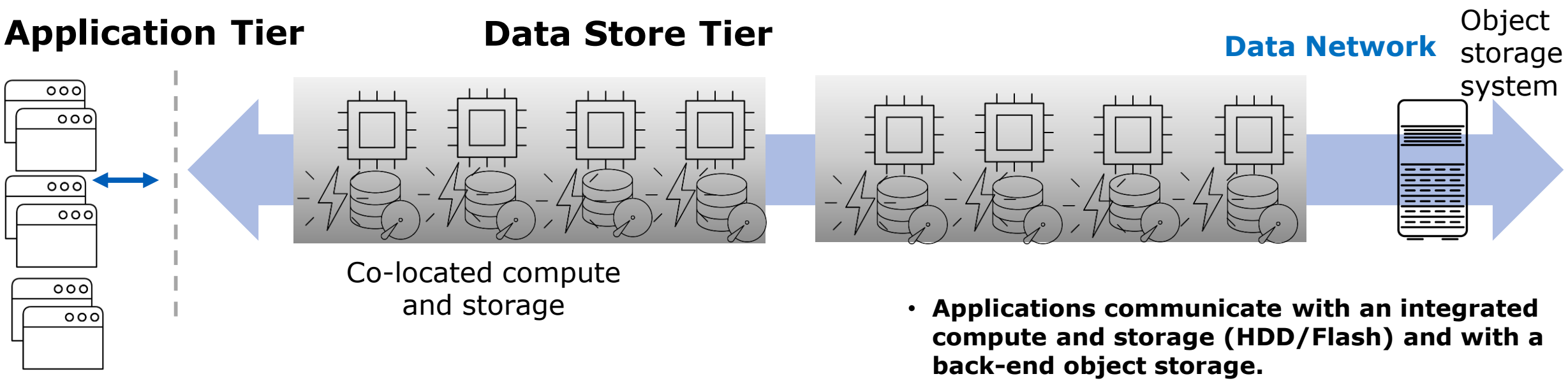
A Unified Workflow for Analytics on the 3rd Platform



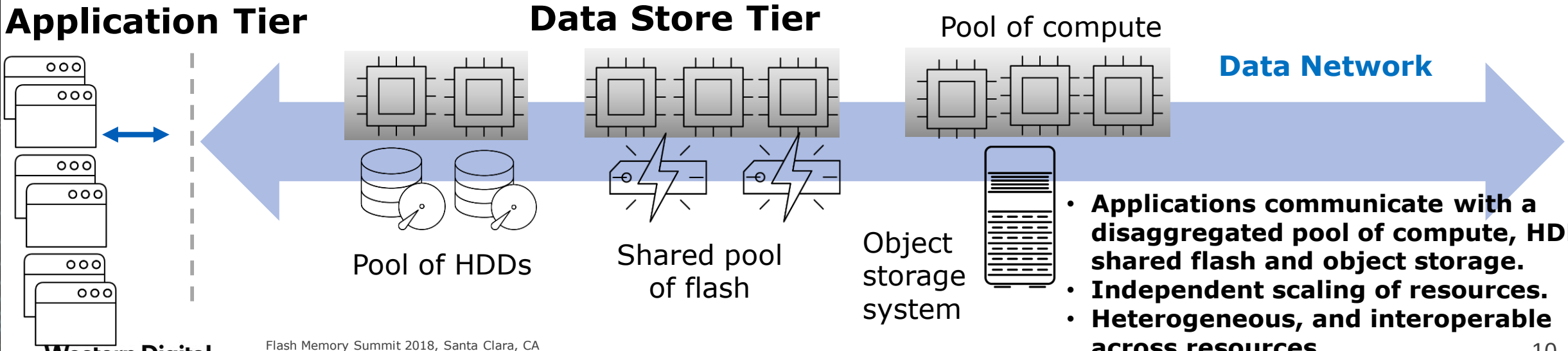
An Enhanced Workflow for Analytics on the 3rd Platform



Aggregated vs. Disaggregated Architectures for Analytics



- **Applications communicate with an integrated compute and storage (HDD/Flash) and with a back-end object storage.**
- **Homogenous across resources.**



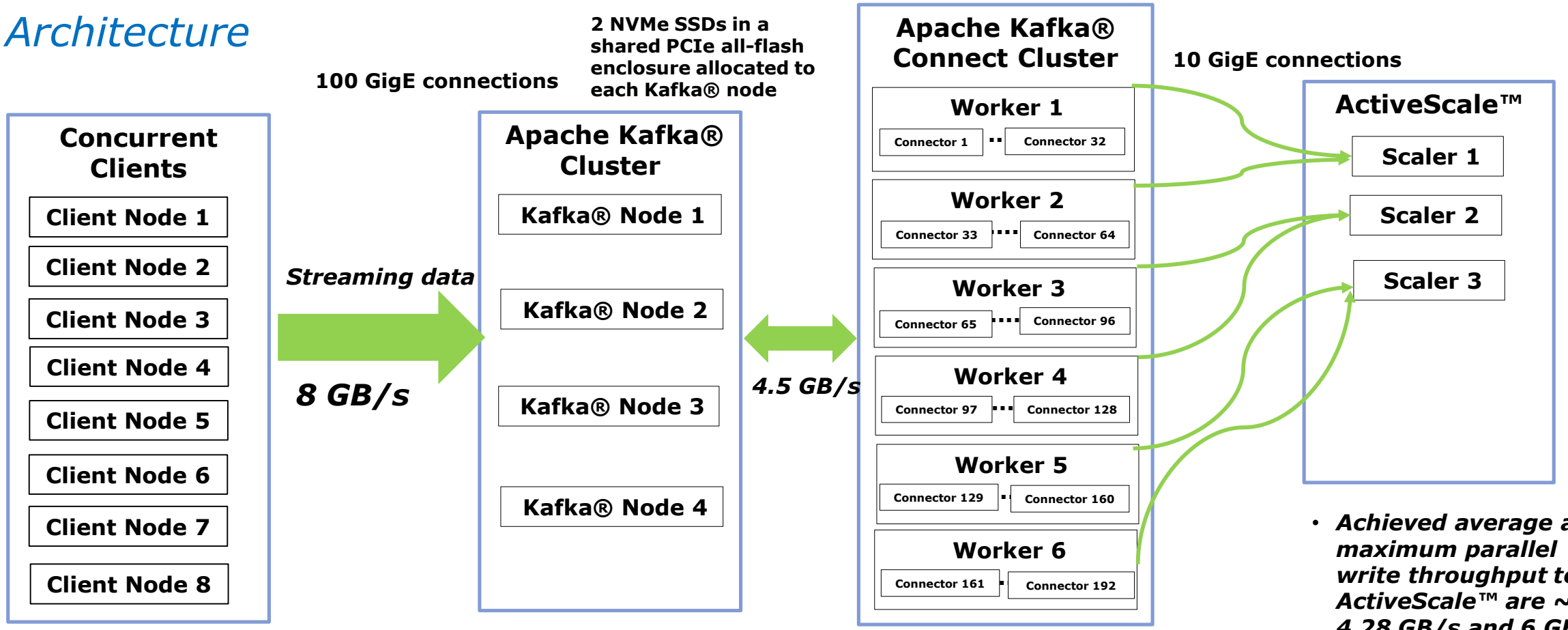
- **Applications communicate with a disaggregated pool of compute, HDDs, shared flash and object storage.**
- **Independent scaling of resources.**
- **Heterogeneous, and interoperable across resources.**

Agenda

1. Delivering insight along the entire spectrum from edge to core
2. Coupling big data with fast data: complementing a data lake for real-time insight
3. **Stream ingestion to ActiveScale™ object storage system as a component within a data lake**
4. IoT speed layer – implementation
5. A real-world IoT use case

Stream Ingestion from Apache Kafka® to ActiveScale

Architecture



For each benchmark run on 8 clients:

- ***Record size: 500 bytes/message.***
- ***Sends 800 M messages at 8 GB/s.***
- ***Publishes to a Kafka® topic, 32 M messages at a time.***

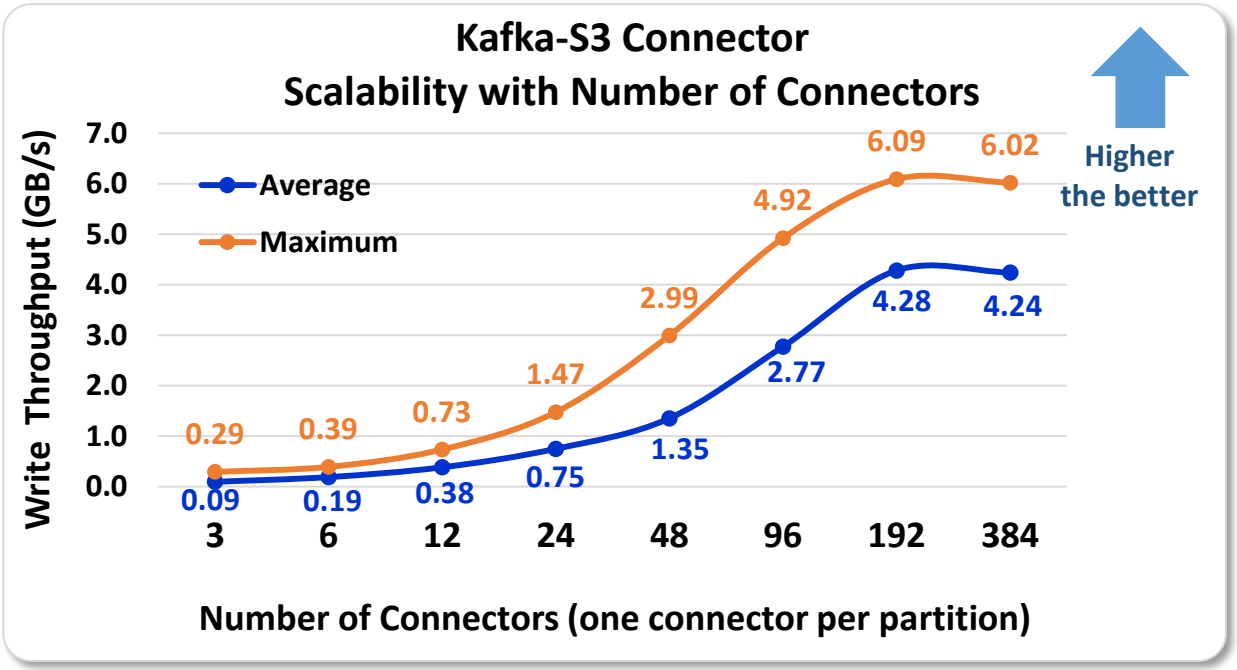
For each benchmark run, Kafka® cluster can process:

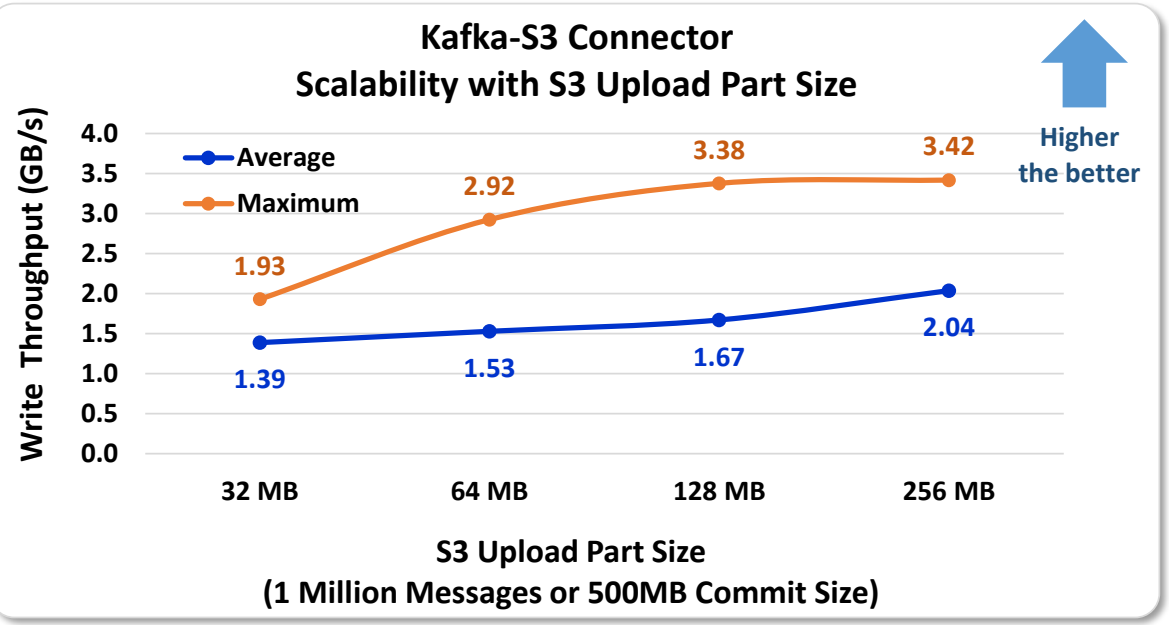
- ***16 M messages/sec, for the message size of 500 bytes.***
- ***Average latency of acknowledgment to the clients is 75 ms/message.***

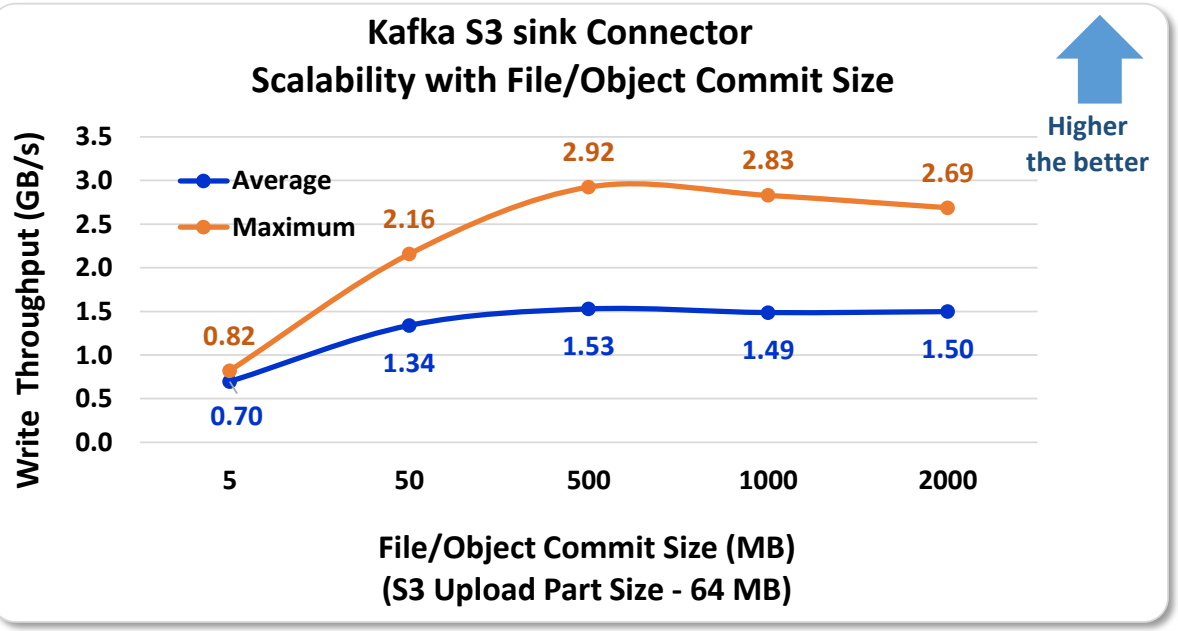
- ***Each connector in the Kafka® Connect Cluster works as a subscriber to a Kafka® topic in the Kafka® Cluster and sinks the data to the ActiveScale™.***
- ***A total of 192 connectors is writing to ActiveScale™ and saturating the write throughput to a single ActiveScale™ system.***

- ***Achieved average and maximum parallel write throughput to ActiveScale™ are ~ 4.28 GB/s and 6 GB/s, respectively.***
- ***Projected throughput will increase by 2x with 12 worker nodes in the Kafka® Connect cluster.***

Apache Kafka® S3 Connector to ActiveScale™: Performance

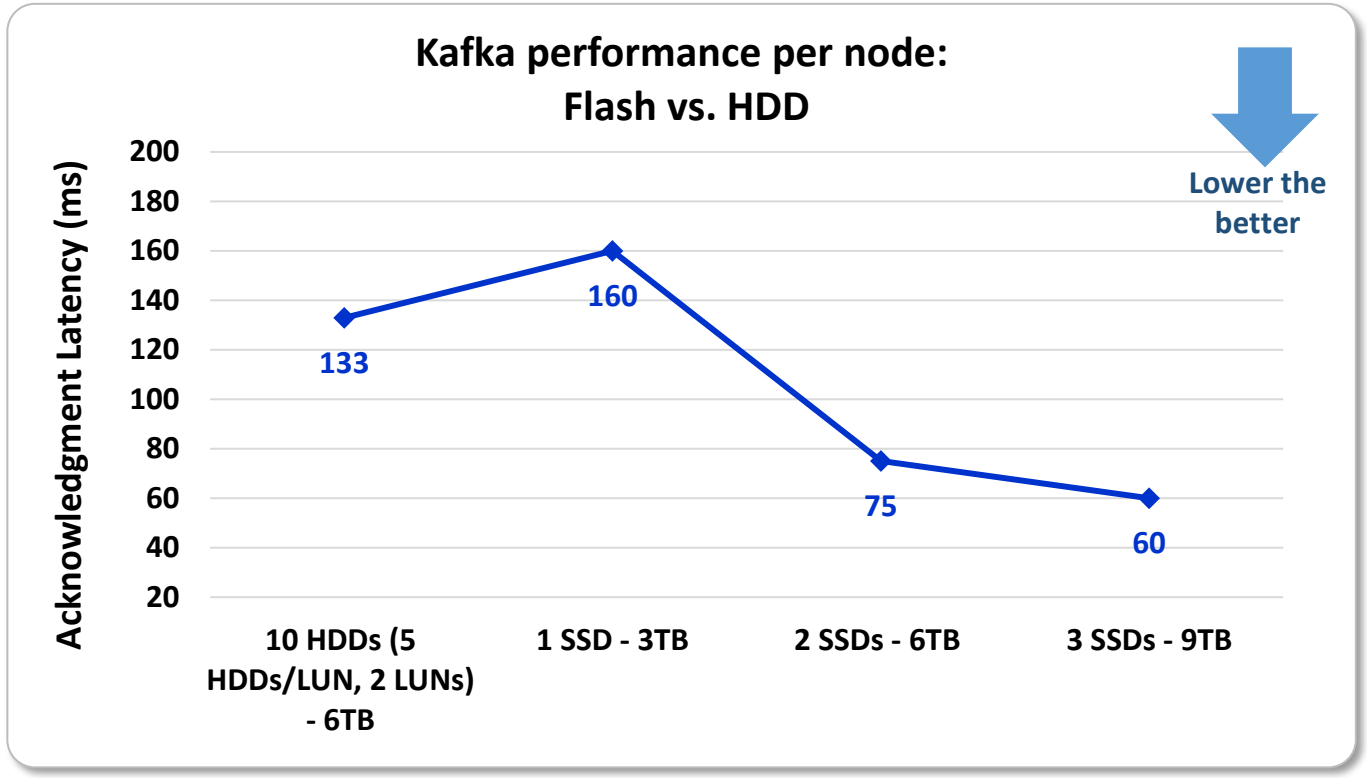






- **An optimal throughput is achieved with a total of 192 connectors writing to a single ActiveScale™ system.**
- **An optimal and stable performance is achieved with the S3 upload part size of 128 MB and with a commit size of 1 Million messages (500 MB), irrespective of the number of connector partitions.** 48 connectors have been used for these performance tests.

Apache Kafka® Performance on Flash vs. HDDs



- *With 16 M messages/sec, for the message size of 500 bytes:*

  The average latency of acknowledgment to the clients by Kafka is 75 ms/message with 2 NVMe SSDs in a shared PCIe all-flash enclosure allocated to each of the Kafka nodes.

- *A low latency of Kafka with 2 SSDs allocated to each node of a 4-node cluster has allowed for achieving a rate of 16 M messages/s.*

- *With 1 SSD ≈10 HDDs, in terms of latency, it would be necessary to use 20 HDDs to achieve the above message rate by Kafka.*

Agenda

1. Delivering insight along the entire spectrum from edge to core
3. Coupling big data with fast data: complementing a data lake for real-time insight
4. Stream ingestion to ActiveScale™ object storage system as a component within a data lake
5. **IoT speed layer – implementation**
6. A real-world IoT use case

IoT Speed layer: Stream Processing and Storing

Requirements: Stream Processing

- **Process events in "near-real time": low latency;**
- **High Velocity;**
- **Guaranteed processing;**
- **Fault-tolerant;**
- **Scales 'easily'.**

Choices

- **Spark Streaming**
  - Low-Latency, High Throughput, Fault-tolerant;
  - Uses Spark core for large-scale stream processing, In-flight messages;
  - Component of Apache Spark™ Ecosystem;
  - Uses: ETL on streaming data, Operational dashboards, anomaly & fraud detection, NLP analysis, sensor data analytics;
  - High Adoption Rate.
- **Others: Apache Storm™, Flink®, Apache Apex™, Apache Samza™, Apache NiFi™, etc.**

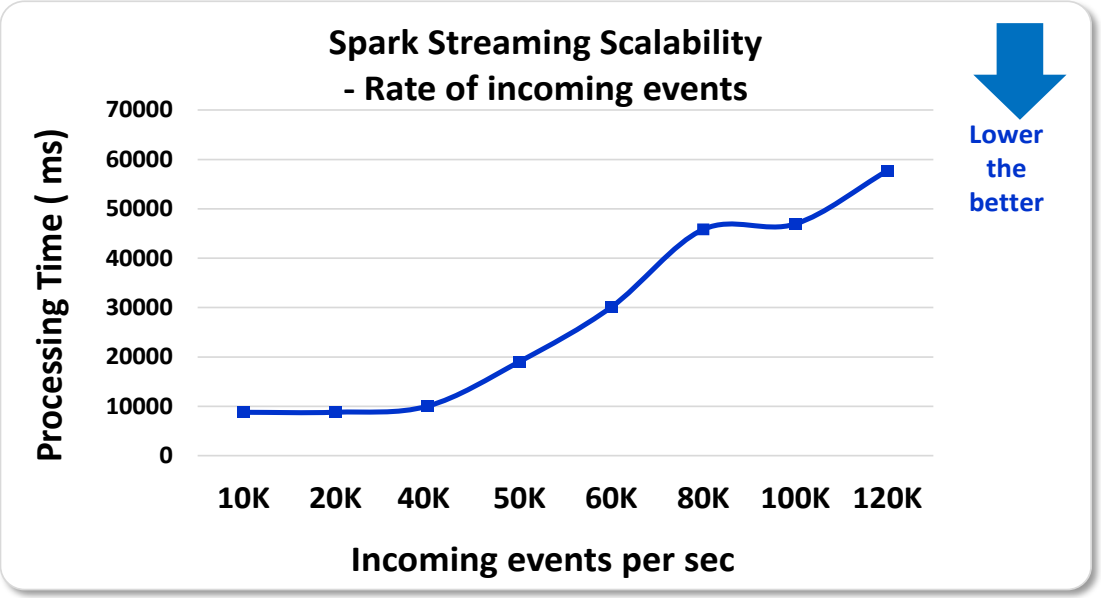
Requirements: Storing

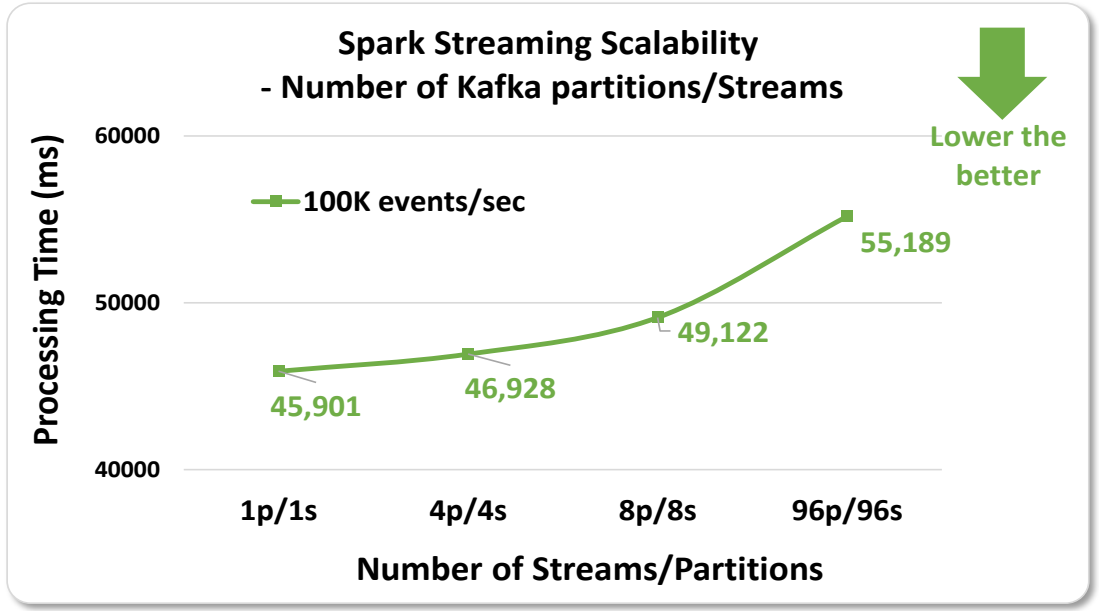
- **Fast/real-time lookup storage;**
- **Support for write-intensive workloads;**
- **Support for concurrent reads and writes of multiple data sizes within a minimal response time.**

Choices

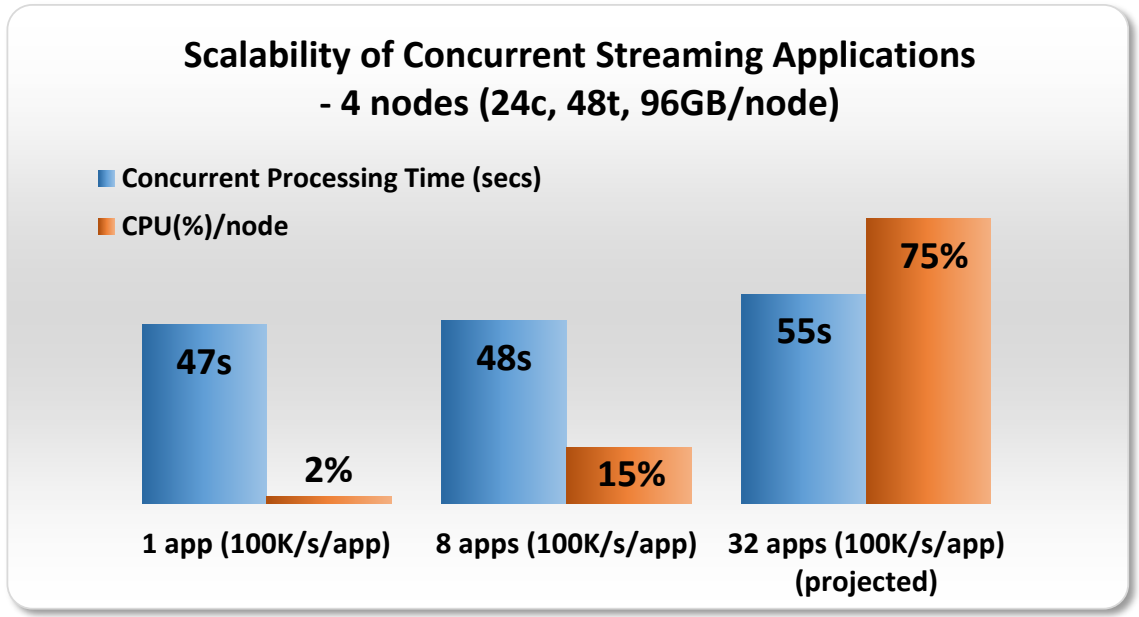
- **NoSQL is a good option**
  - **Cassandra®**
    - Best-in-class, scalable NoSQL data management system;
    - Parallel, distributed, high throughput, 100% fault-tolerant;
    - No dependency on Hadoop; has Spark connector;
    - Typical applications are IoT, fraud detection, recommendation engines, messaging apps.
- **Others: HBase®, Redis, Accumulo®, MongoDB™, etc.**

Apache Spark™ Streaming Performance





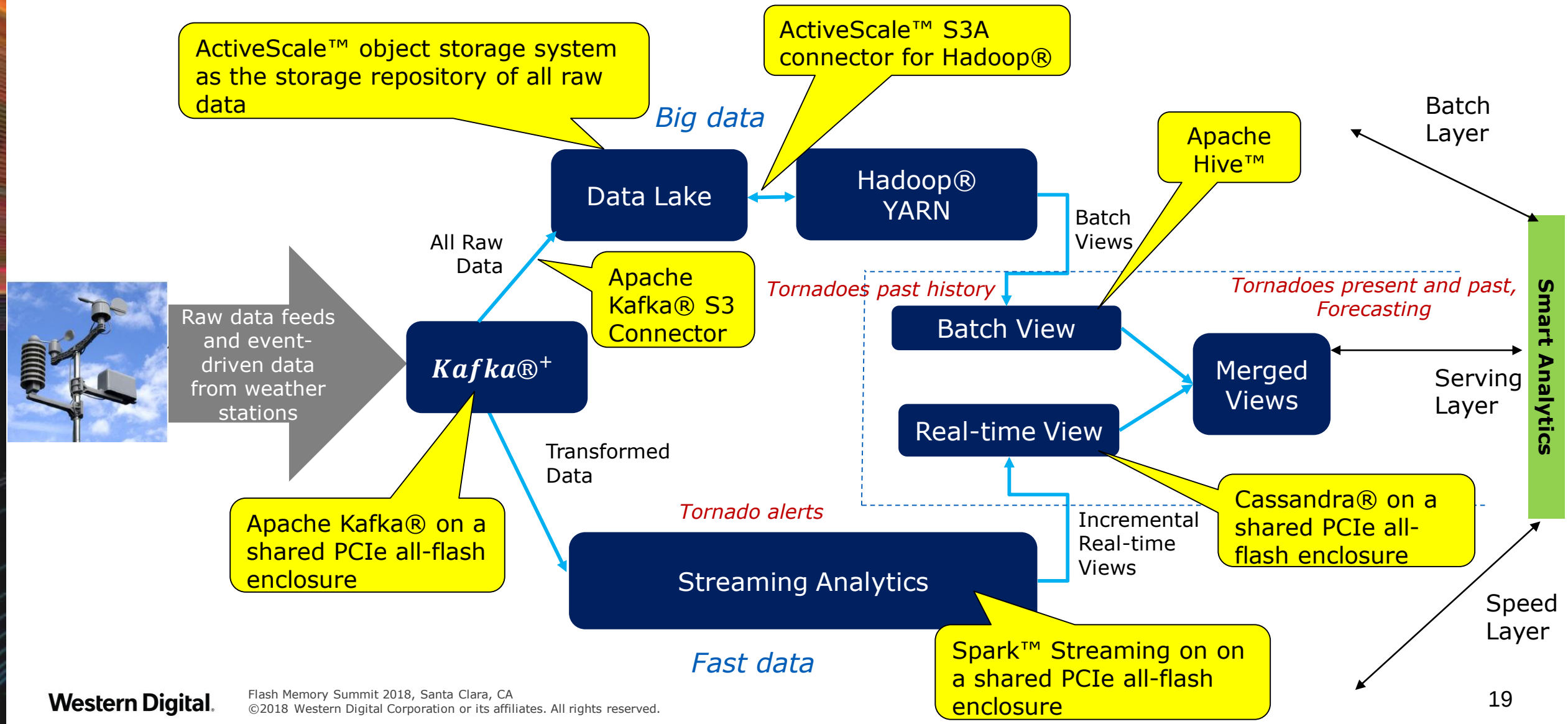
- **Spark Streaming is implemented on 4 nodes (24c, 48t, 96GB) with 2 NVMe SSDs in a shared PCIe all-flash enclosure allocated to each node.**
- **The processing time of a "single" streaming application reaches a threshold at 100K events/s.**
- **Increasing the number of partitions or streams for a "single" application, does not improve the processing time, due to the overhead of aggregating results across multiple partitions.**
- **From measured data, it is estimated that we could easily execute multiple streaming applications concurrently (a total of 3.2M events) within a 55 sec response time and CPU usage of 75-80%.**



Agenda

1. Delivering insight along the entire spectrum from edge to core
3. Coupling big data with fast data: complementing a data lake for real-time insight
4. Stream ingestion to ActiveScale™ object storage system as a component within a data lake
5. IoT speed layer – implementation
6. A real-world IoT use case

Real-world Use Case: Unified Streaming and Batch Analytics on Climate IoT Data



Summary and Best Practices

- **For efficient coupling between big data and fast data for IoT use cases, it is recommended to:**
  - **Implement ActiveScale™ object storage system as a component within a data lake, for storing all the incoming raw streaming data from the ingestion layer.**
  - **Tune the number of connectors configurable in the Kafka® Connect Cluster, the S3 upload part size, and the File/Object commit size (also known as flush size), for ingesting streaming data from Apache Kafka® to ActiveScale™ object storage system with an optimal write throughput.**
  - **Leverage ActiveScale™ S3A connector for Hadoop to perform various long-running batch analytics and ad-hoc queries against the petabyte-scale data stored in ActiveScale™, without any physical data movement.**
  - **Implement a "shared" all-flash storage for the speed layer. Allocate the shared flash storage to compute nodes, for a balanced throughput, IOPS and capacity, required by the applications in the speed layer including ingestion, stream processing and real-time store.**
  - **Implement the "shared" all-flash storage to complement an ActiveScale™ object storage system to achieve a disaggregation of compute and storage for the whole solution, without compromising on long-term data persistence, centralization or quality, durability, high availability and cost.**
  - **Tune the number of concurrent applications, number of streams per application, and also the allocation of compute and JVM memory to achieve an optimal performance in the speed layer.**