In-Memory Computing SUMMIT EUROPE 2019

Make your data science actionable, real-time machine learning inference with stream processing.

Neil Stevenson, Solution Architect
Hazelcast
3rd June 2019

13:45 – 14:35

neil@hazelcast.com

Which came first ? (Chicken | Egg)

neil@hazelcast.com

In-Memory Computing SUMMIT EUROPE 2019

Chicken

neil@hazelcast.com

In-Memory Computing SUMMIT EUROPE 2019

What relevance is this?!

What is this ?

- You can eat them
- They lay eggs
- They can be pets

- Not just any old chicken but…
- MY CHICKEN
- A Bresse Gauloise

Stream Processing

neil@hazelcast.com

In-Memory Computing SUMMIT EUROPE 2019

Business Challenges for Real-time Applications

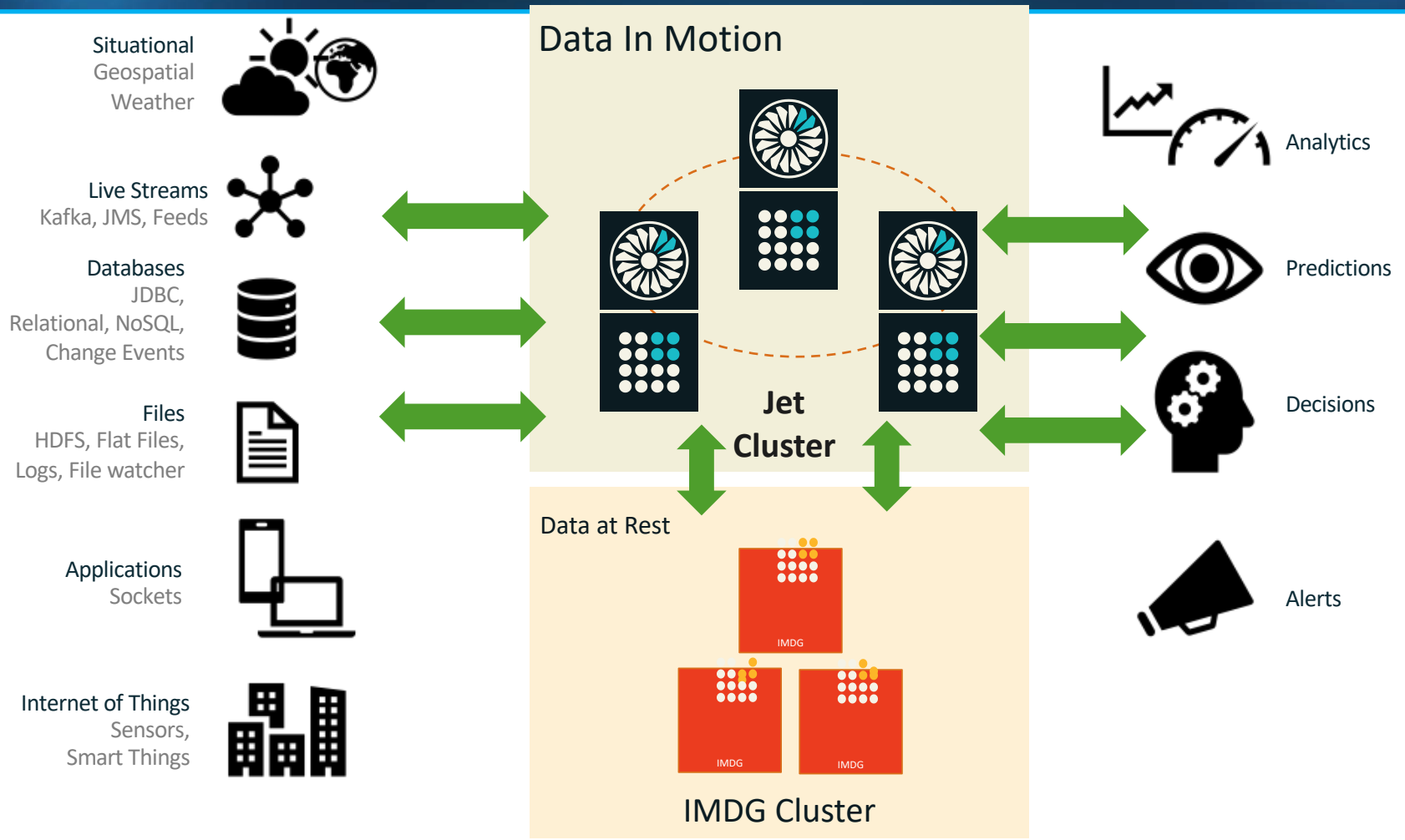
Latency & Speed
Time is money

Scalability
Hazelcast scales effortlessly responding to peaks, valleys for optimal utilization

Real-Time, Continuous Intelligence
Real-time view of constantly changing operational data

Zero Downtime
Built for high resiliency

In-Memory Platform



Hazelcast

Mobile

Apps

Social

Commerce

Communities

IMDG In-Memory Data Grid

Integrate APIs, Microservices, Notifications	**Communicate** Serialization, Protocols
Store/Update Caching, CRUD Persistence	**Compute** Query, Process, Execute
Scale Clustering & Cloud, High Density	**Replicate** WAN Replication, Partitioning
Secure Privacy, Authentication, Authorization	**Available** Rolling Upgrades, Hot Restart

Jet In-Memory Streams

Ingest & Transform Events, Connectors, Filtering	**Combine** Join, Enrich, Group, Aggregate
Stream Windowing, Event-Time Processing	**Compute & Act** Distributed & Parallel Computations
Secure Privacy, Authentication, Authorization	**Available** Job Elasticity, Graceful shutdown

Management Center

Secure | Manage | Operate

Embeddable | Scalable | Low-Latency

Secure | Resilient | Distributed

Analytics
Visualization
Data Lake

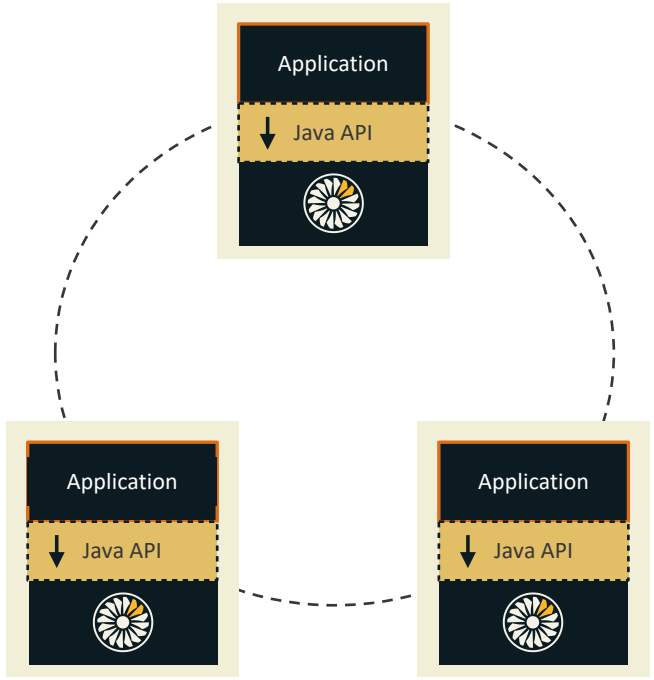
Live Streams
Kafka, JMS, Sensors, Feeds

Databases
JDBC, Relational, NoSQL, Change Events

Files
HDFS, Flat Files, Logs, File watcher

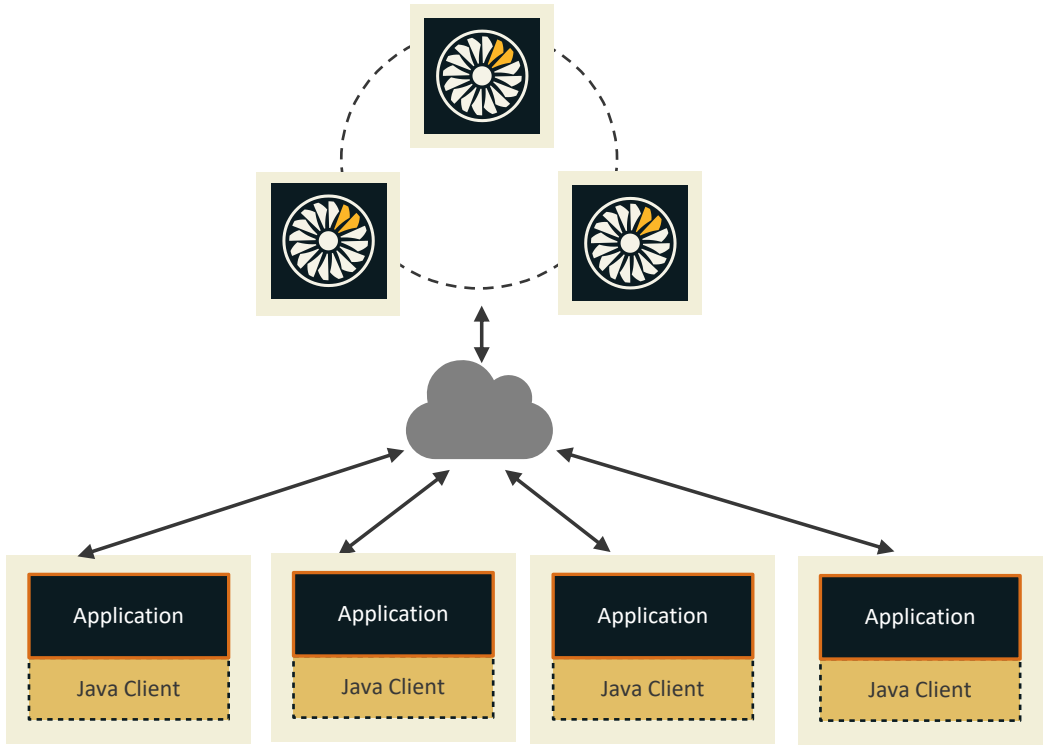
Applications
Sockets

Hazelcast Jet - options



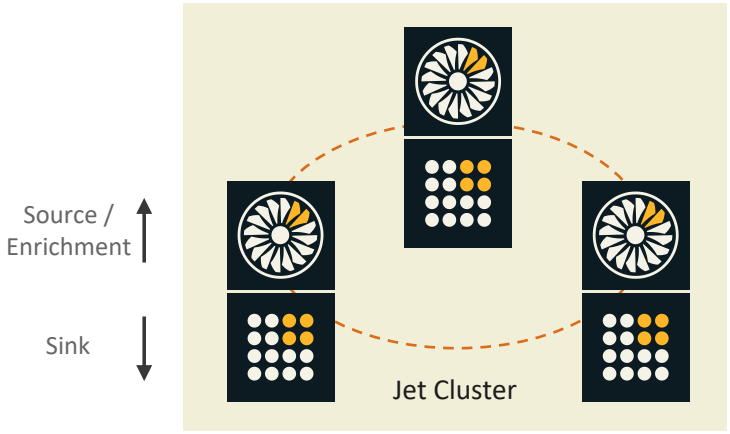
- No separate process to manage
- Great for microservices
- Great for OEM
- Simplest for Ops – nothing extra

Client-Server



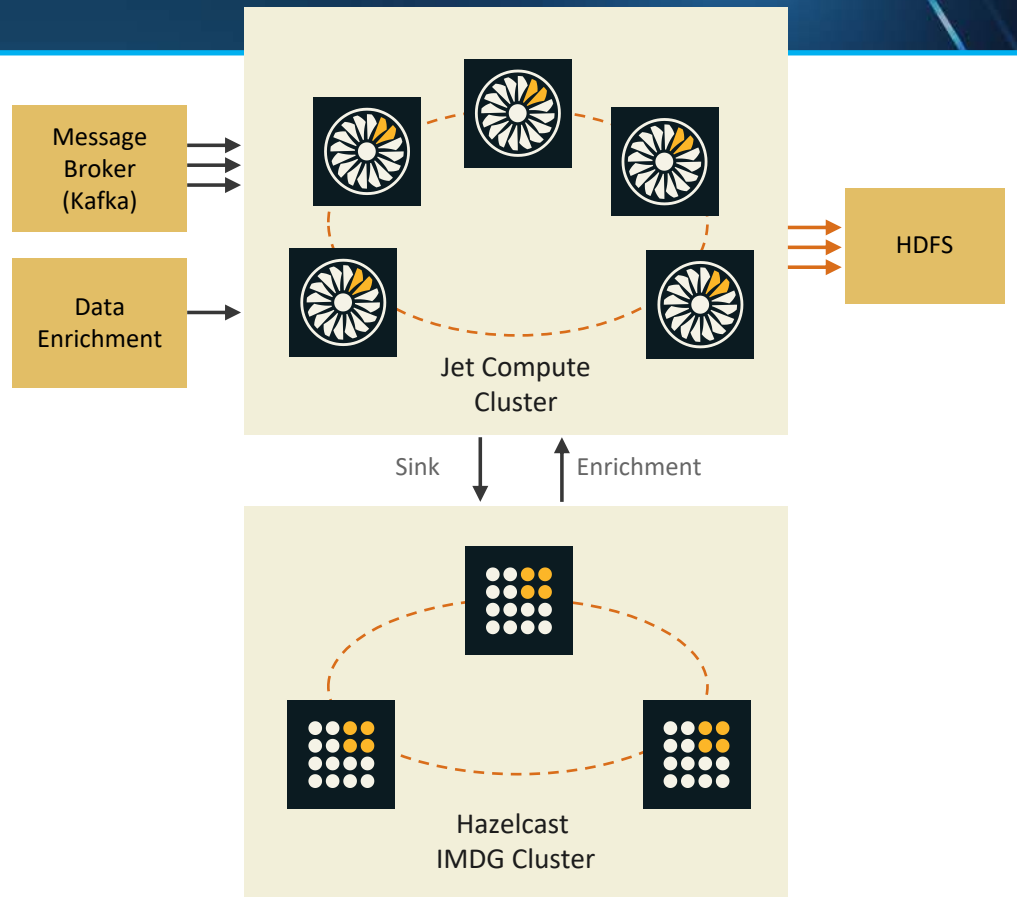
- Separate Jet Cluster
- Scale Jet independent of applications
- Isolate Jet from application server lifecycle
- Managed by Ops

Hazelcast Jet & IMDG



Good when:

- Where source and sink are primarily Hazelcast
- Jet and Hazelcast have equivalent sizing needs

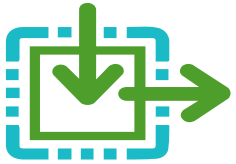


Good when:

- Where source and sink are primarily Hazelcast
- Where you want isolation of the Jet cluster

Streaming Use Cases

Real-time Stream processing



- Big Data in near real-time
- Distributed, in-memory computation
- Aggregating, joining multiple sources, filtering, transforming, enriching
- Elastic scalability
- Super fast
- High availability
- Fault tolerant

ETL/Ingest

- Supports common sources such as HDFS, File, Directory, Sockets
- Custom sources can be easily created
- Batch and streaming
- Streaming ingest from Oracle, SQL Server, MySQL using Striim
- Sink to Hazelcast or other operational data stores

Data-Processing Microservices

- Data-processing microservices
- Isolation of services with many, small clusters
- Service registry
- Network discovery
- Inter-process messaging
- Fully embeddable
- Spring Cloud, Boot Data Services

Edge Processing

- Low-latency analytics and decision making
- Saves bandwidth and keeps data private by processing it locally
- Lightweight – runs on restricted hardware
- Both processing and storage
- Fully embeddable for simple packaging
- Zero dependencies for simple deployment

Hazelcast Jet?

High performance | *Industry Leading Performance*

Stream Processing & Data Grid | *Source, Sink, Enrichment*

Very simple to program | *Leverages existing standards*

Very simple to deploy | *Embed 14MB jar or Client-Server*

Works in every Cloud | *Same as Hazelcast IMDG*

The Evolution of Stream Processing

neil@hazelcast.com

In-Memory Computing SUMMIT EUROPE 2019

Generations

 1st Gen (2000s) Hadoop(batch) or Apama(CEP) ***hard choices***	**Distributed Batch Compute** – MapReduce – scaled, parallelized, distributed, resilient, - not real-time or **Siloed, Real-time** – Complex Event Processing – specialized languages, not resilient, not distributed(single instance), hard to scale, fast, but brittle, proprietary
 2nd Gen (2014) Spark ***hard to manage***	**Micro-batch distributed** – heavy weight, complex to manage, not elastic, require large dedicated environments with many moving parts, not Cloud-friendly, not low-latency
 **3rd Gen (2017 Jet & Flink)** **flexible & scalable** **True "Fast Data"**	**Distributed, real-time streaming** – highly parallel, true streams, advanced techniques (Directed Acyclic Graph) enabling reliable distributed job execution Flexible deployment - Cloud-native, elastic, embeddable, light-weight, supports serverless, fog & edge. Low-latency Streaming, ETL, and fast-batch processing, built on proven data grid

Streams … hiding in plain sight

Unix:

```
ls | tr 'A-Z' 'a-z' | grep txt | wc
```

Pipe == directed acyclic graph!

As in pipeline, mainly linear, no routing or collation

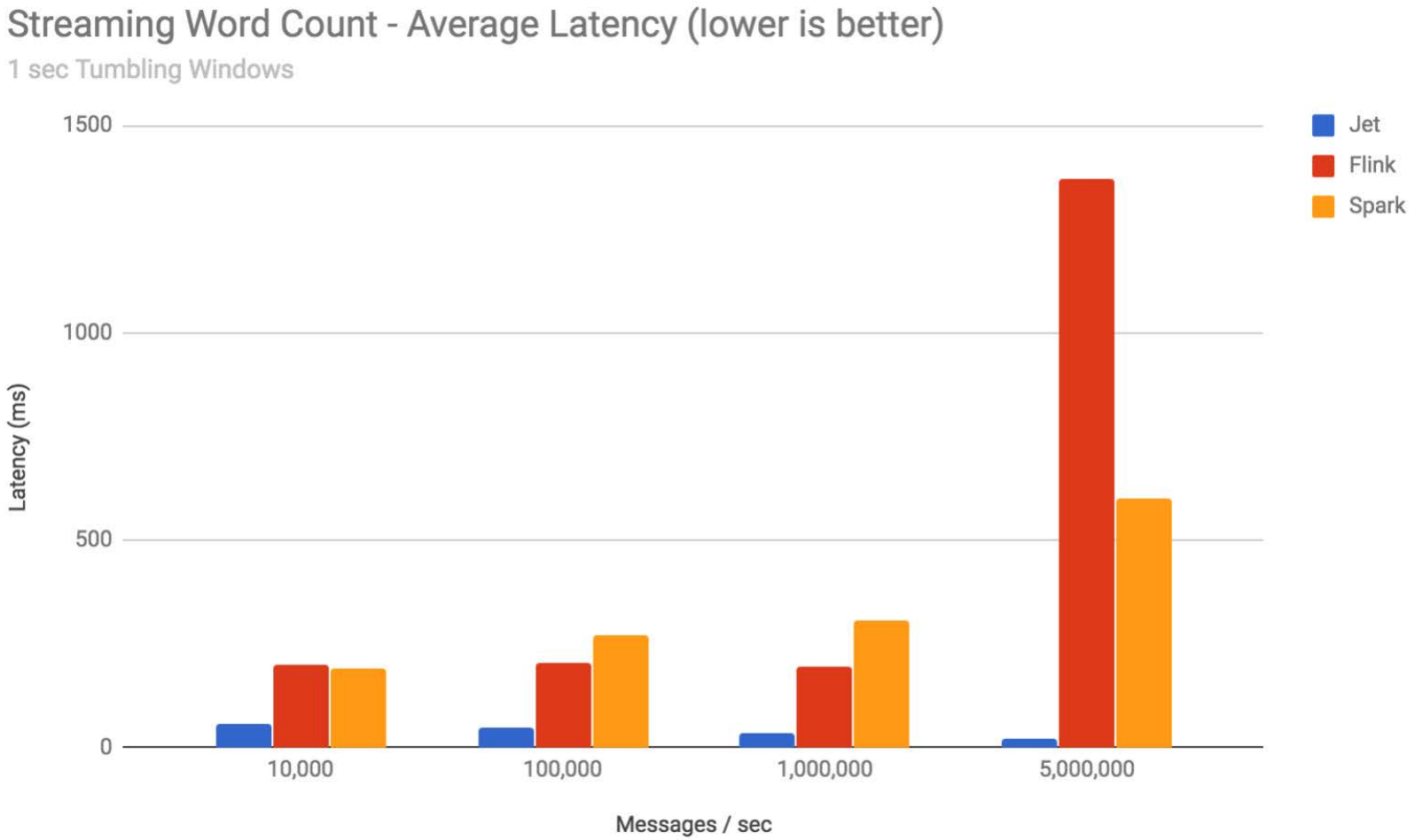
ls – source

tr – intermediate "infinite" stage

grep – intermediate "infinite" stage

wc - sink

Performance

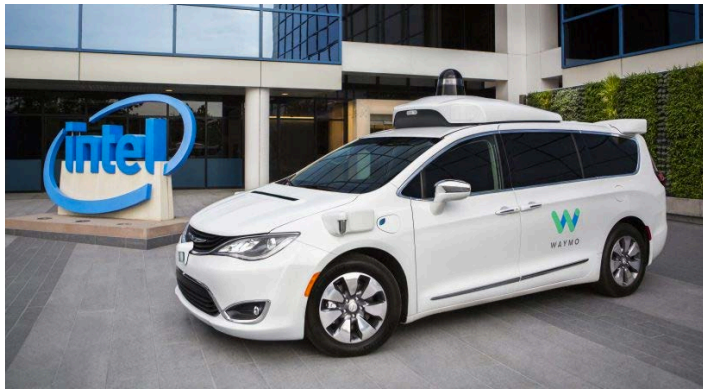


AI

neil@hazelcast.com

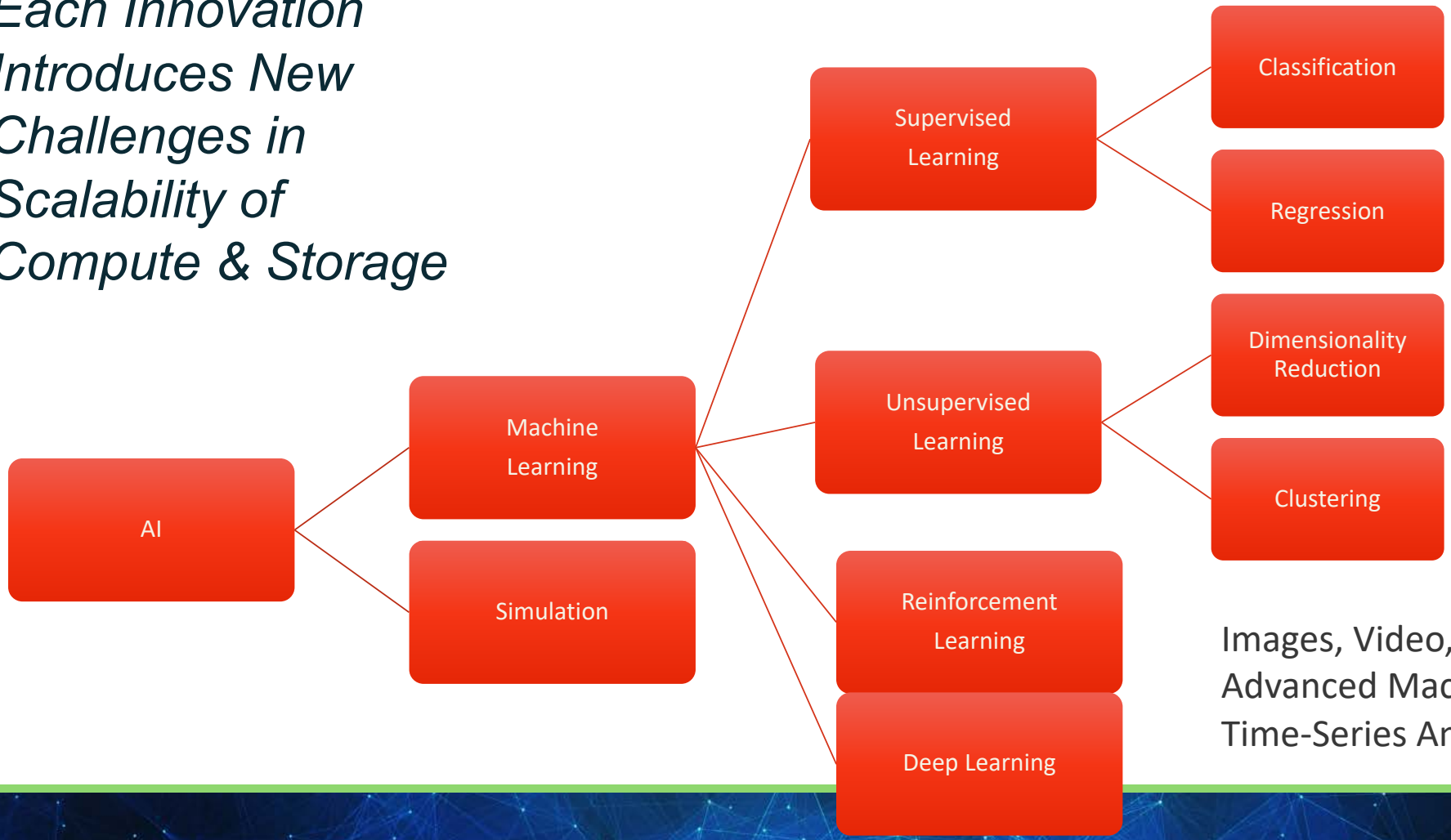
In-Memory Computing SUMMIT EUROPE 2019

Computers… they're out there…



AI Techniques Continue to Expand & Evolve

Each Innovation Introduces New Challenges in Scalability of Compute & Storage



Image/Video Processing
Unstructured Data
Fraud & Anomaly Detectio

Predicting Trends
Structured Numeric Data

Image Processing
Unstructured Data
Feature Extraction

Data Exploration
Feature Extraction

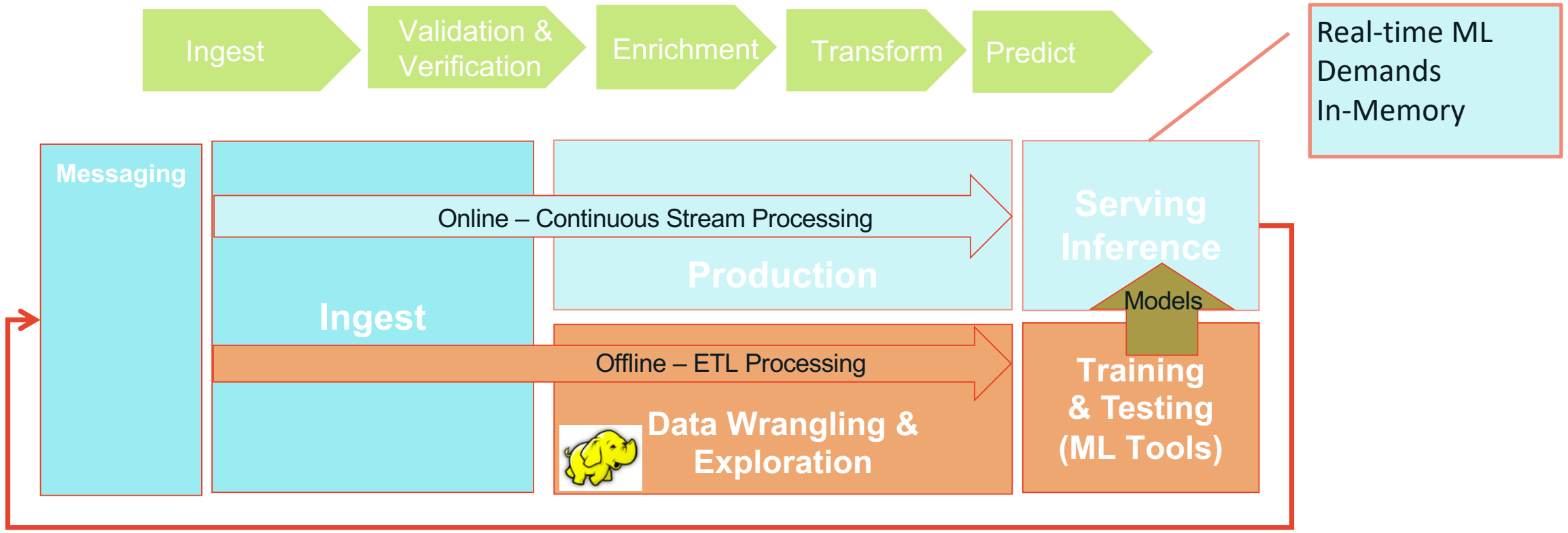
Images, Video, Audio
Advanced Machine Learning & AI
Time-Series Analysis

Machine Learning

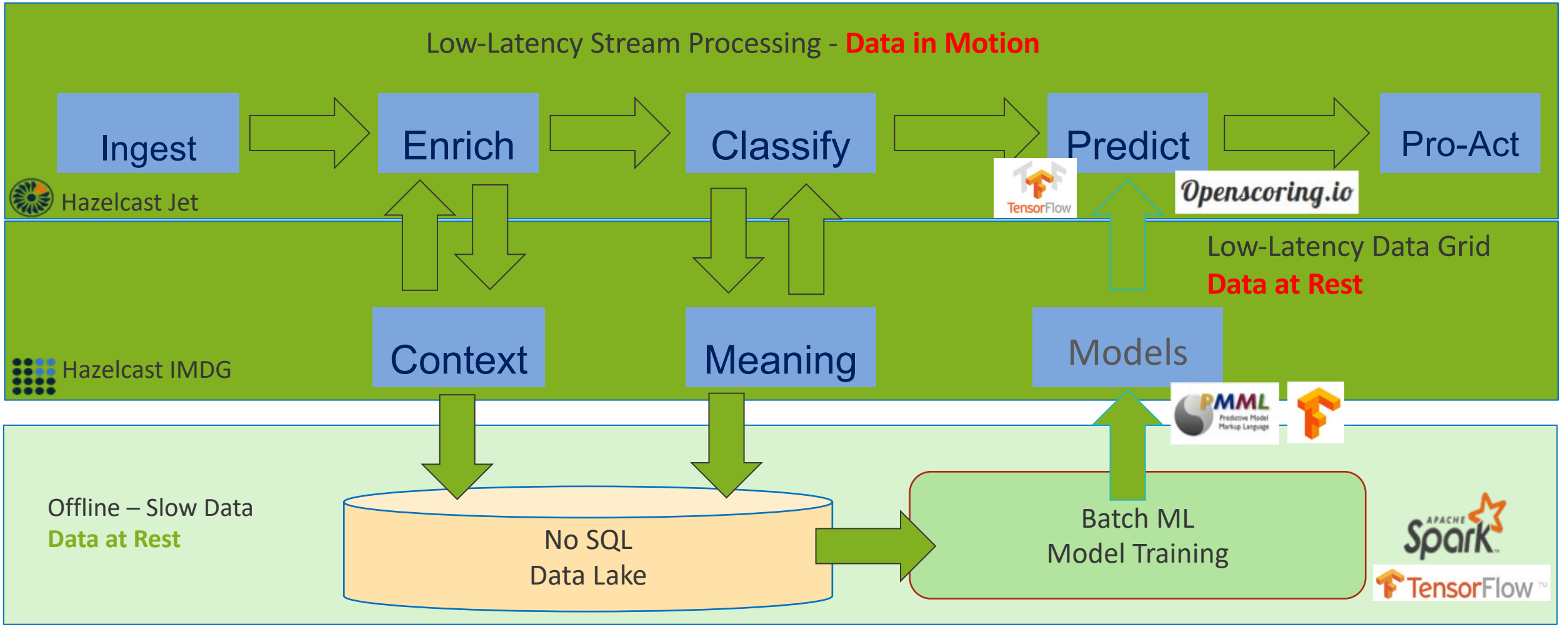
neil@hazelcast.com

In-Memory Computing SUMMIT EUROPE 2019

Information Flow for Machine Learning



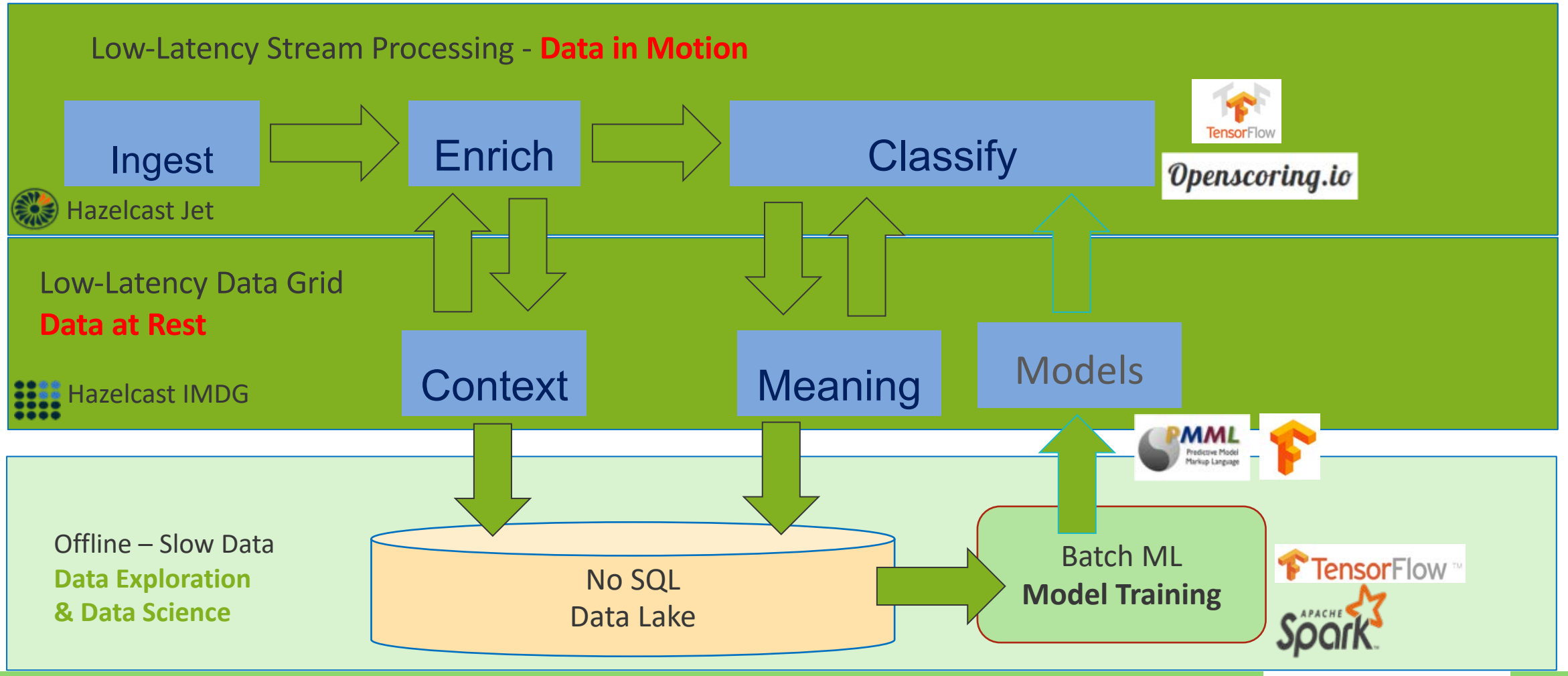
Online Machine Learning within an In-Memory Platform



Advantages of In-Memory Platform for ML

- **Fast**
  - Data Held in Memory for Low Latency Processing
  - Models also held in-memory
  - Compute with Data Locality Further Reduces Latency
- **Elastic**
  - Job Elasticity – Leveraging Directed Acyclic Graph & Cooperative Work Sharing
  - Compute & Data Layers Easy to Scale – Not Bound to Disks
  - Supports Microservices and Serverless Architectures
- **Resilient**
  - Multi-Data Center Architectures Enable 99.999% Uptime at Scale
  - Lossless Job Recovery and Exactly-One Processing Achieved with In-Memory Replicated State

Feature Engineering

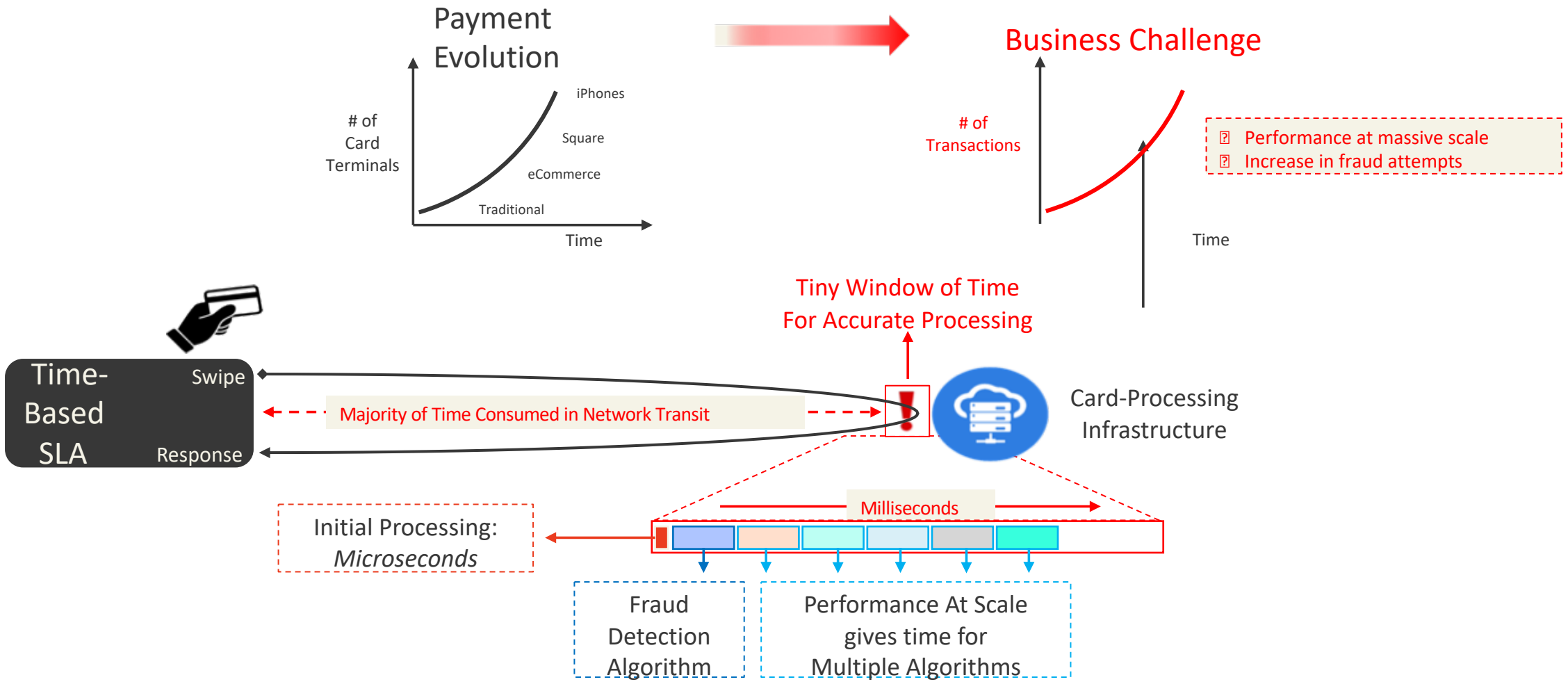


Speed Matters

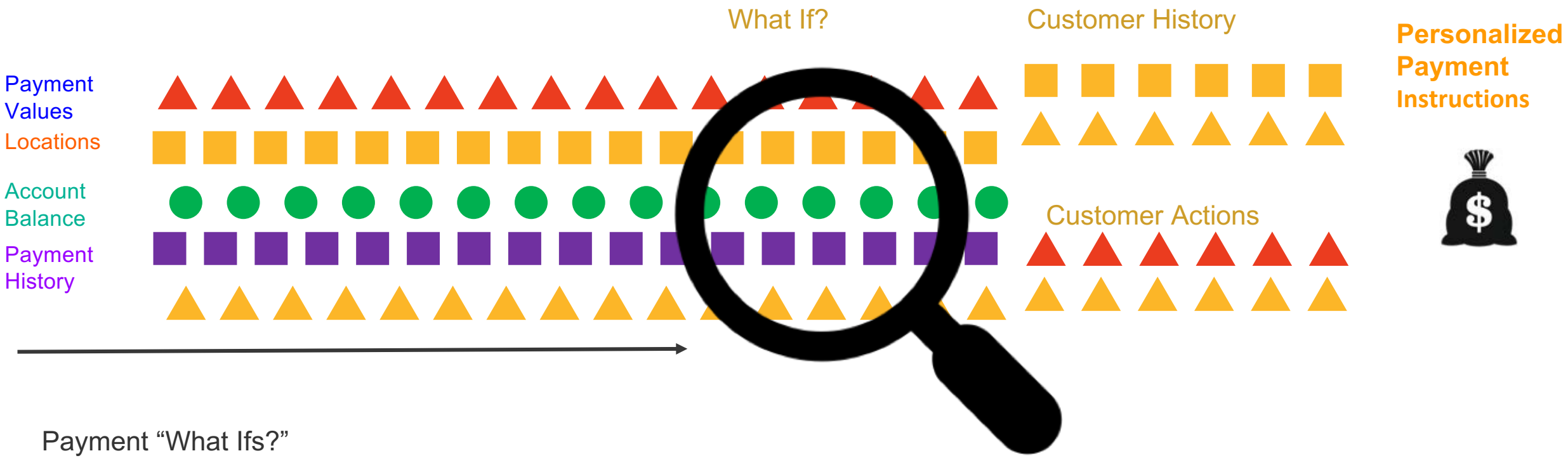
neil@hazelcast.com

In-Memory Computing SUMMIT EUROPE 2019

Eg. Credit Card fraud analysis



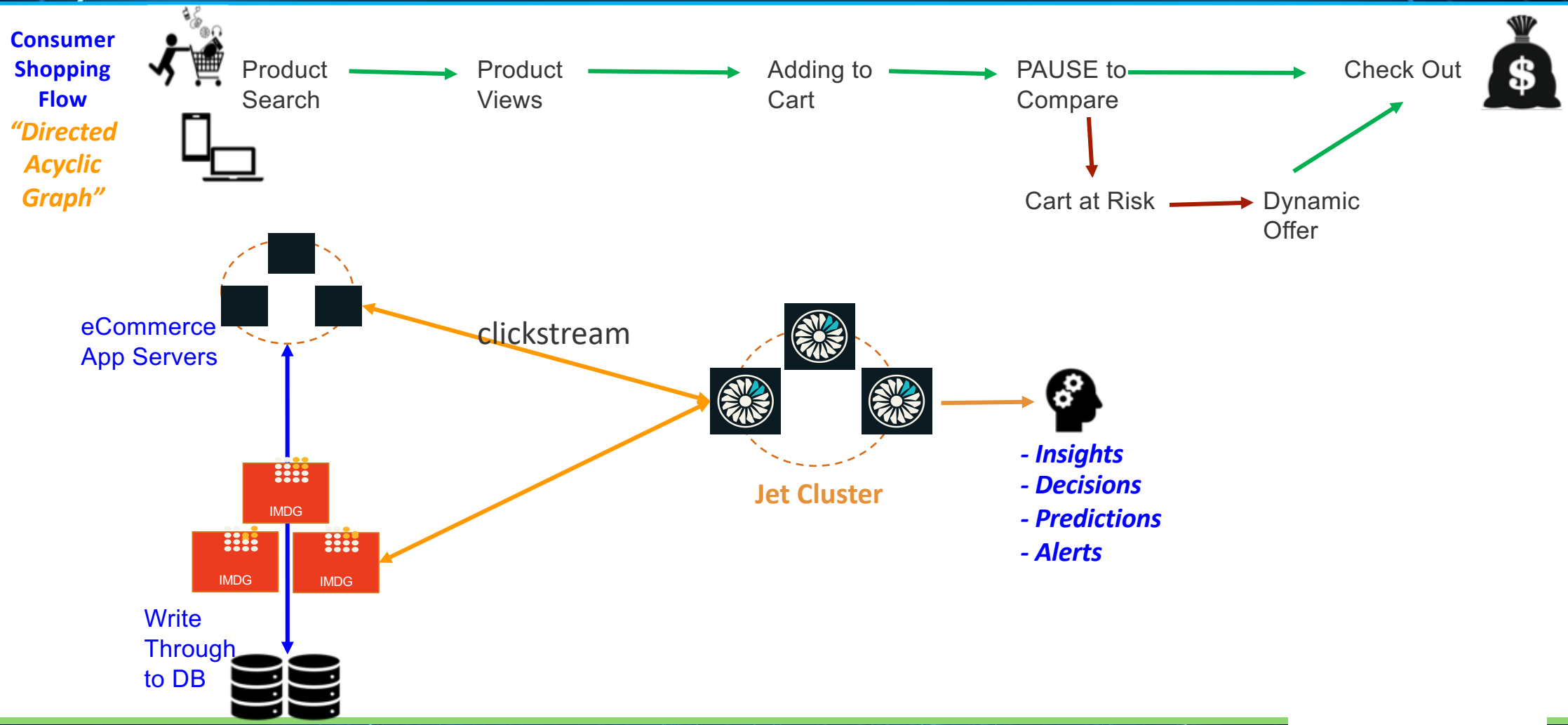
Eg. Credit Card fraud analysis



Payment “What Ifs?”

- What are their balances? - Risk > Payment > Identify fraud > Block payment
- What is their history? - Opportunity > Real-time Offers > Upsell

Eg. Real time offers in e-commerce



Demo time !

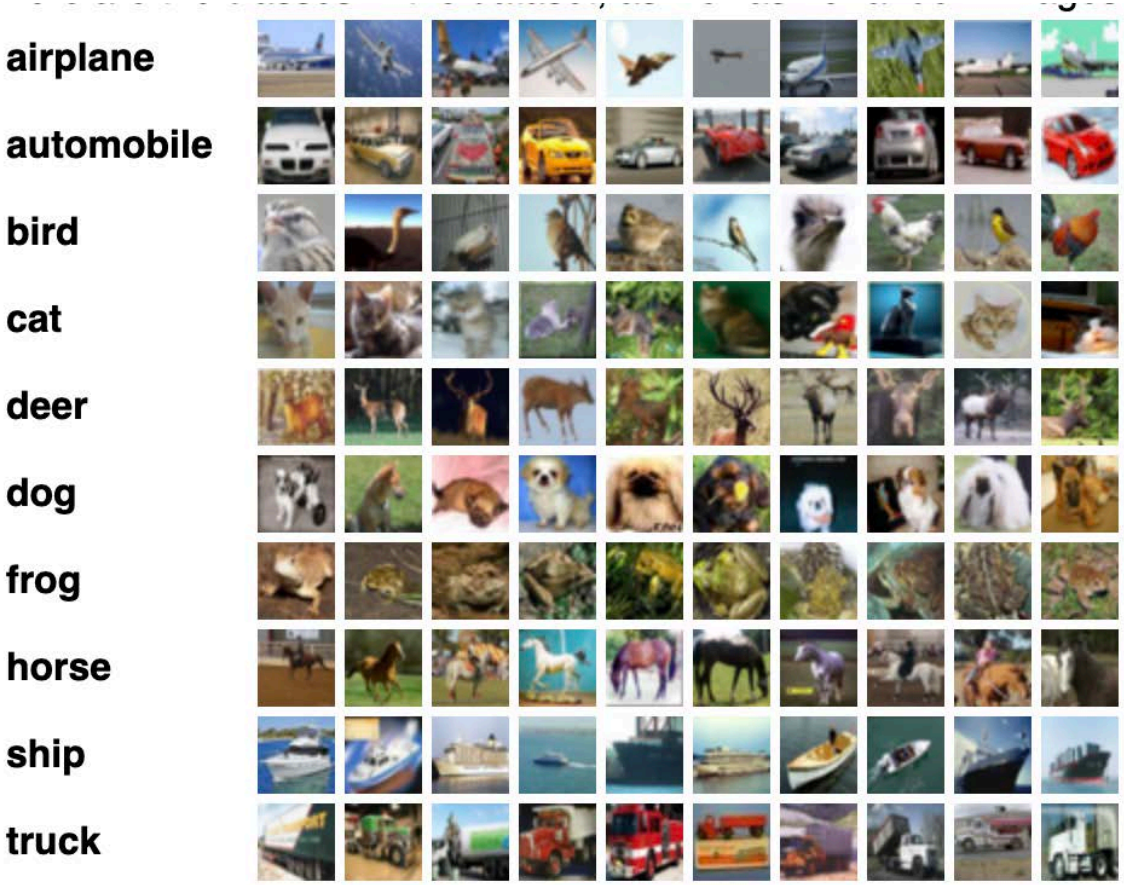
neil@hazelcast.com

Canadian Institute For Advanced Research

- CIFAR10
- https://www.cs.toronto.edu/~kriz/cifar.html

- 60,000 images, 10 classes
- (6000 of each ☺)

- A machine learning model



Is it a bird ? Is it a plane ?

Recognising animals

“I never expected all these cats”

Sir Tim Berners-Lee

Was it a bird ? Was it a plane ?

Did it work ?

…not perfectly, which is why we need to re-train and re-deploy the model

The End

neil@hazelcast.com

Summary

Wrong first time!

And every time… you will need to redeploy your ML

https://github.com/hazelcast/hazelcast-jet-demos

Questions ?

Thank You

neil@hazelcast.com