THE IOT APPLICATION CHALLENGE

HANDLING MASSIVE STREAMING DATA

COLIN MACNAUGTHON

NEEVE RESEACH

WHO IS NEEVE RESEARCH?

- Headquartered in Silicon Valley
- Creators of the X Platform™ - Memory Oriented Application Platform.
- Passionate about high performance computing.
- Running in production at Fortune 100-300

AGENDA

- What is IoT … What are the Challenges?
- How The X Platform tackles Streaming
- Streaming Usecase: IoT Fleet Tracking

WHAT IS IOT

The "Internet of Things":
"real world" stuff (often augmented with sensors)
streaming data to a network

WHAT WE ARE REALLY TALKING ABOUT IS:
LARGE SCALE STREAMING

WHAT IS NEEDED FOR IOT

EVENT-DRIVEN
Its all about ***streaming*** lots of events

SCALABILITY
Lots of things ⟶ LOTS of events

SPEED
100s of thousands to millions of events/sec, response latency in microseconds or low millis.

RELIABILITY
CANNOT lose mission critical events No Dups / No Loss (Exactly Once)

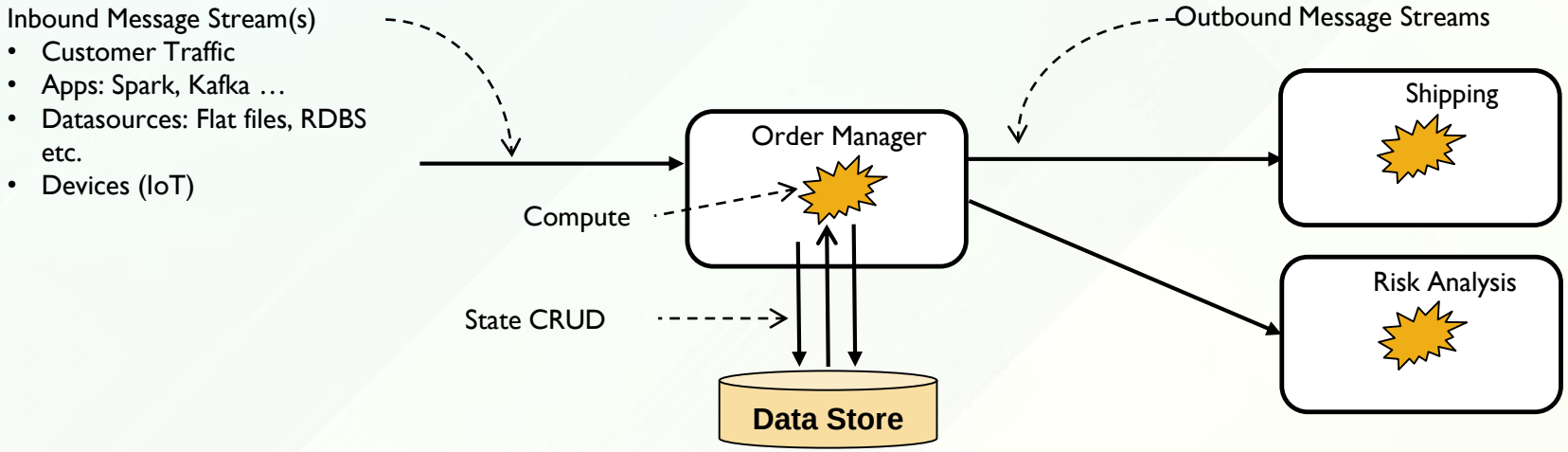
AVAILABILITY
Always On, Always available in the face of network/process/machine/data center failure

AGILITY/EASE
Applications are infinite need to be able to evolve organically

STREAMING APP CHARACTERISTICS

What do they do?

1. Consume Inbound Messages
2. Read / Update State
3. … and Produce Outbound Messages



MICROSECONDS MATTER

A processing time of 1ms limits your throughput to 1000 messages / sec.

Same applies to any synchronous callouts in the stream.

To achieve >10k Transactions/Second you must leverage In Memory technologies

MICROSECONDS MATTER

Memory	Latency
L1 Cache	~1ns
L2 Cache	~3ns
L3 Cache	~12ns
Remote NUMA Node	~40ns
Main Memory	~100ns
Network Read	100µs
Random SSD Read 4K	150µs
Data Center Read	500µs*
Mechanical Disk Seek	10ms

MEMORY ORIENTED COMPUTING!

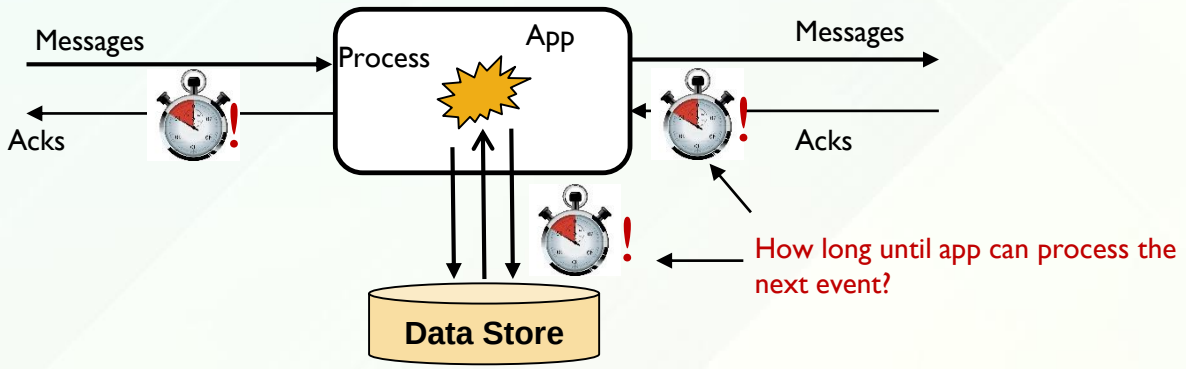
All State in Memory All The Time!

Non Starters For Performance We're Talking About!

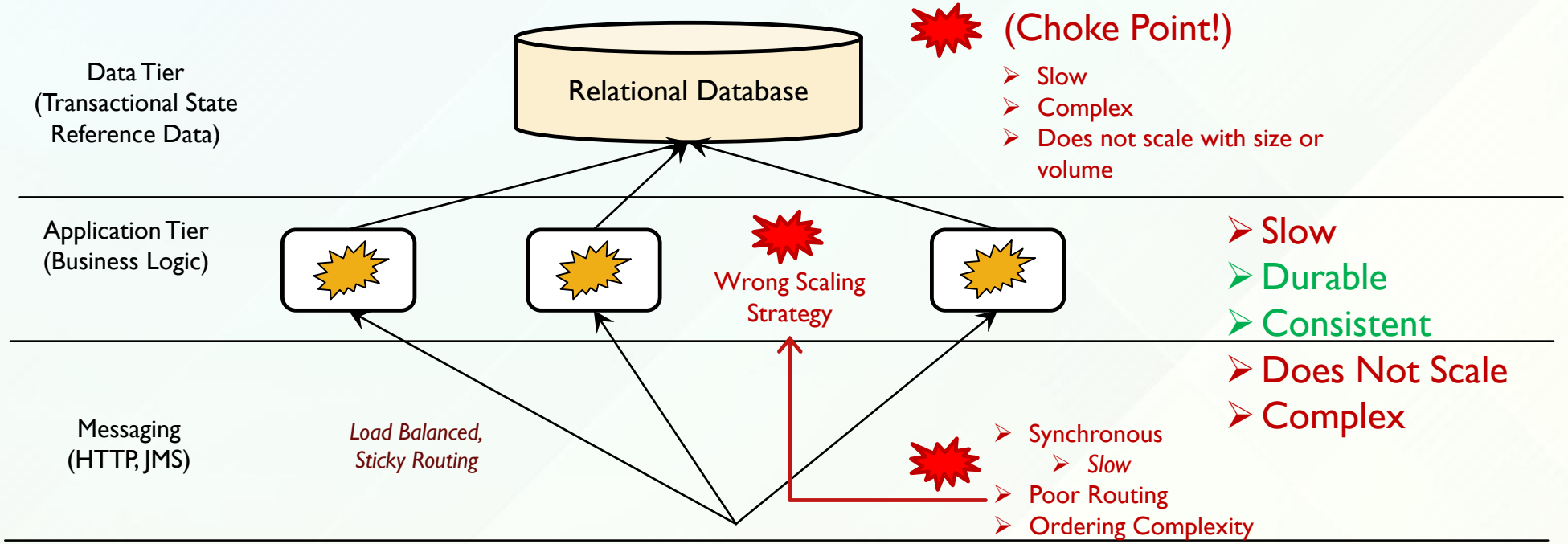
Sources: https://gist.github.com/jboner/2841832
http://mechanical-sympathy.blogspot.com/2013/02/cpu-cache-flushing-fallacy.html

THE CHALLENGES

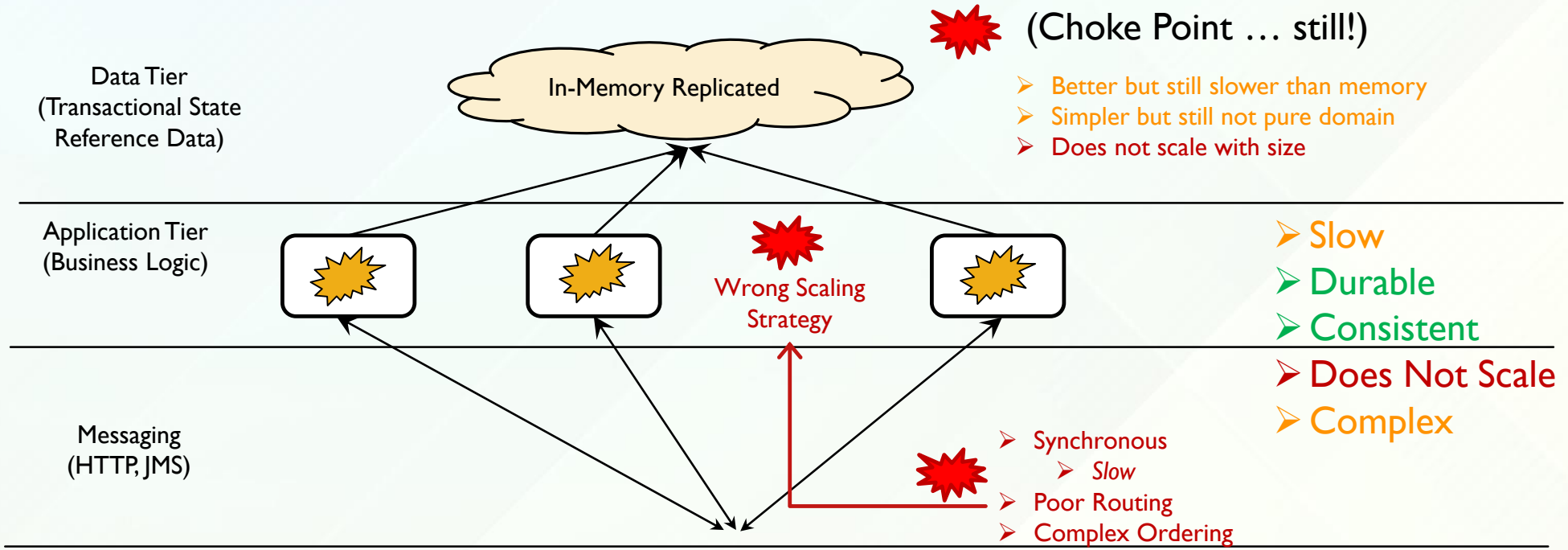
- Exactly Once Semantics
  - Messaging – No Loss / No Dups / Atomic
  - Storage and Access to State – No Loss / No Dups
- Atomicity between Message Streams and State Updates – Receive-Process-Send atomic



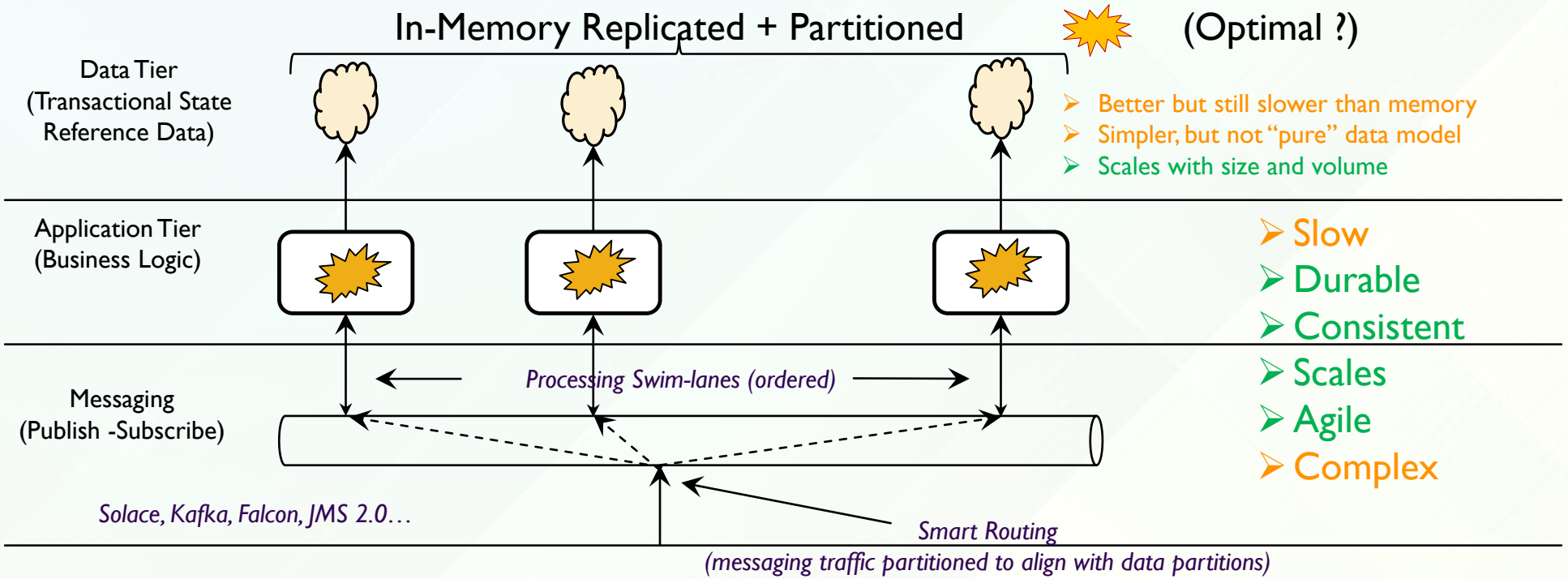
TRADITIONAL TP APPLICATION ARCHITECTURE



LAUNCH DATA INTO MEMORY



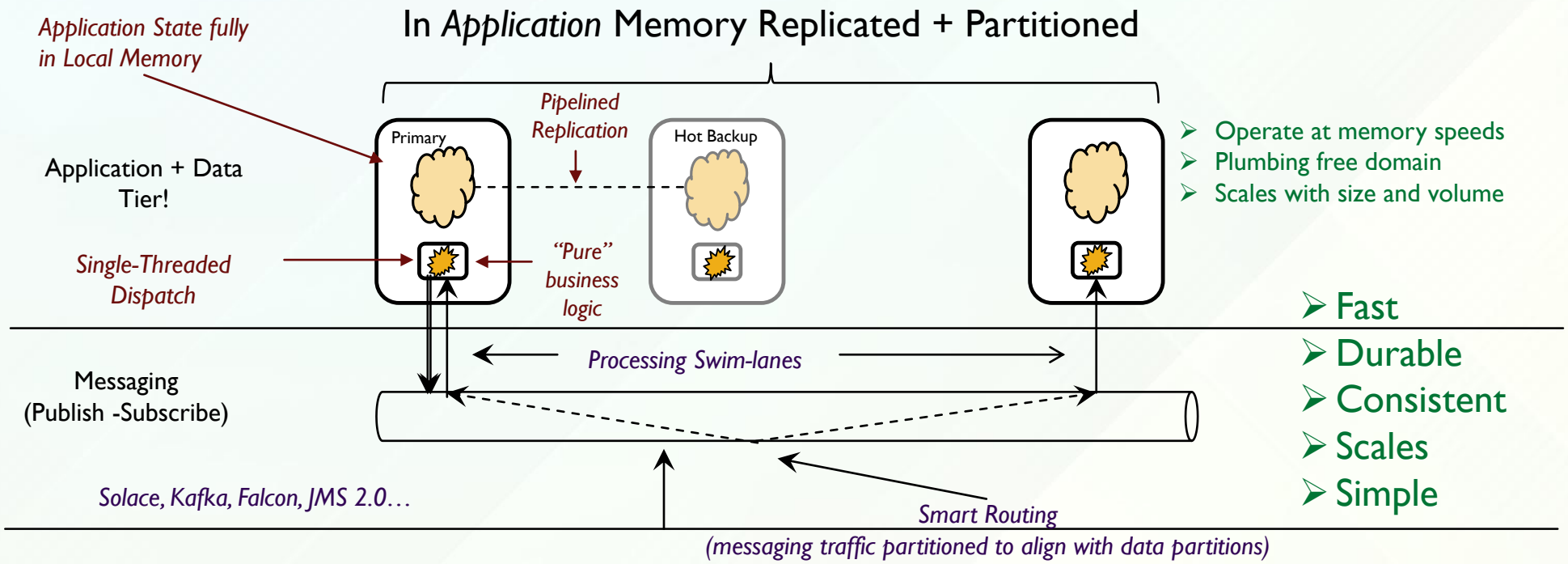
DATA GRAVITY
(DATA STRIPING + SMART ROUTING)



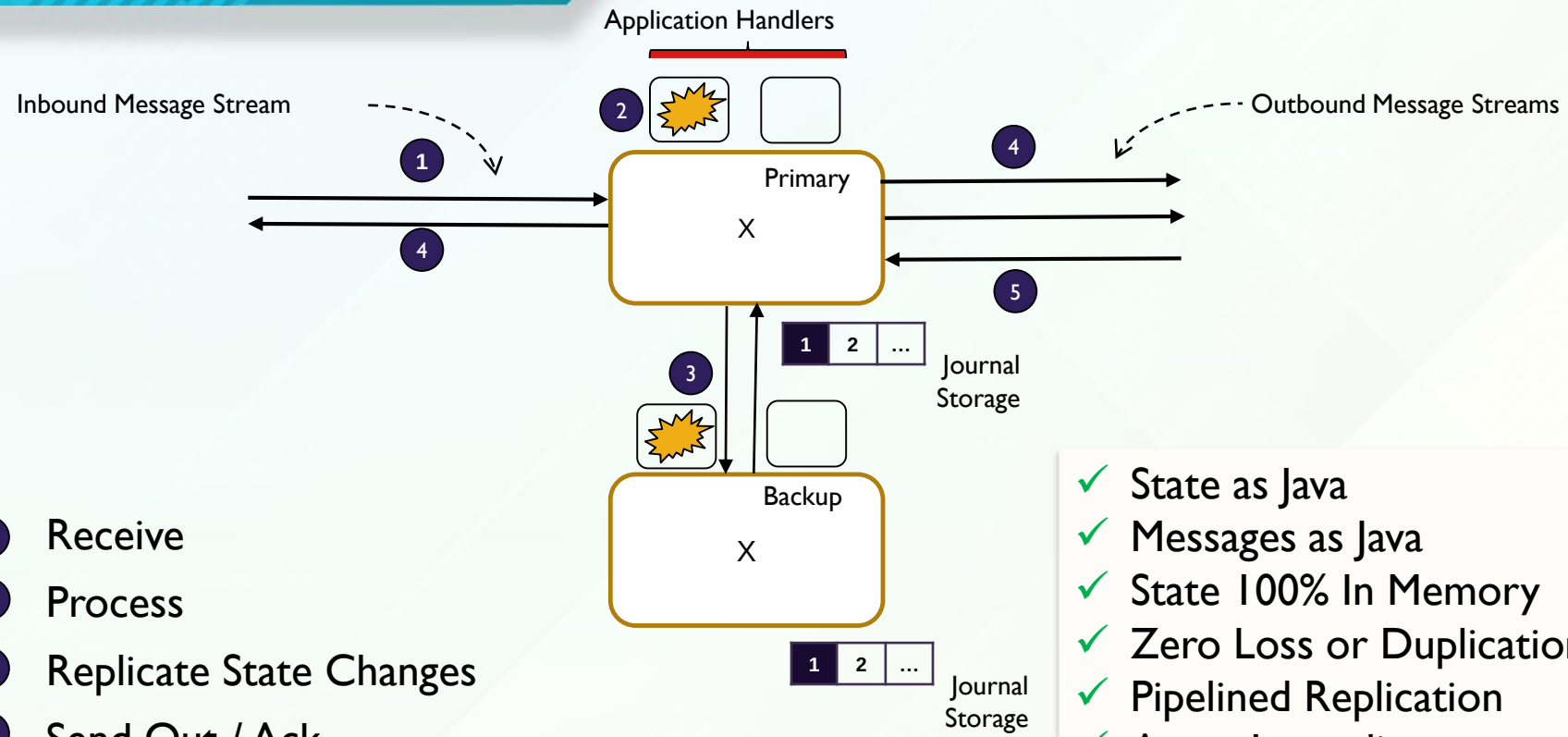
WHY STILL SLOW AND COMPLEX

- **How Slow?**
  - Latency
    - 10s to 100s of milliseconds
  - Throughput
    - Very low with single pipe
    - Few 1000s per second with high concurrency
- **Why Still Slow?**
  - Remoting out of process
  - Synchronous data management and stabilization
  - Concurrent transactions are not cheap!
- **Why Complex?**
  - Transaction Management still in business logic
  - Thread management for concurrency (only way to scale)
  - Data transformations due to lack of structured data models

THE X PLATFORM APPROACH



X PLATFORM TRANSACTION PIPELINING (HA)



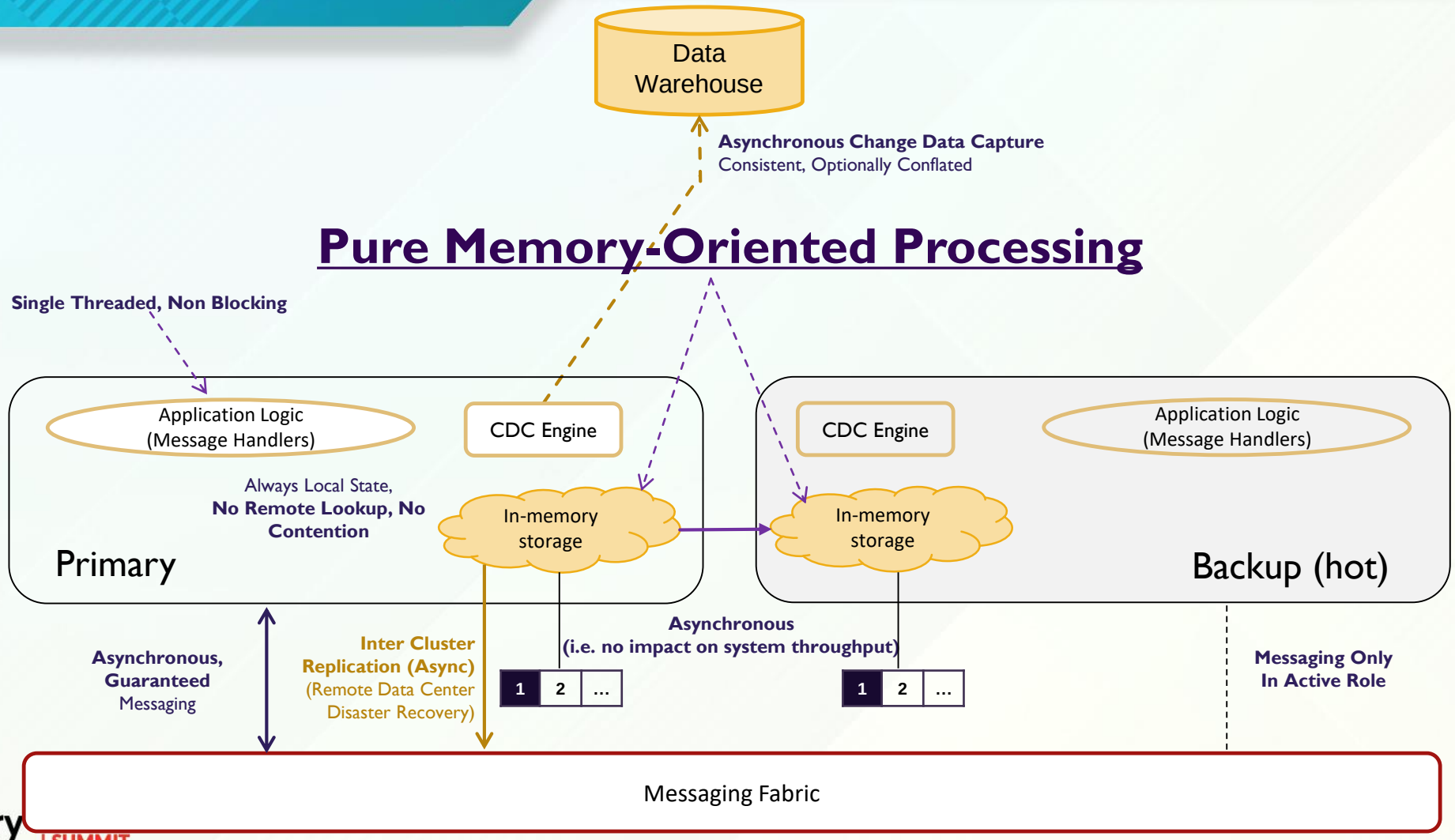
1. Receive
2. Process
3. Replicate State Changes
4. Send Out / Ack
5. Inbound Acks

- ✓ State as Java
- ✓ Messages as Java
- ✓ State 100% In Memory
- ✓ Zero Loss or Duplication
- ✓ Pipelined Replication
- ✓ Async Journaling
- ✓ Pipelined Messaging
- ✓ Pooling for Zero Garbage

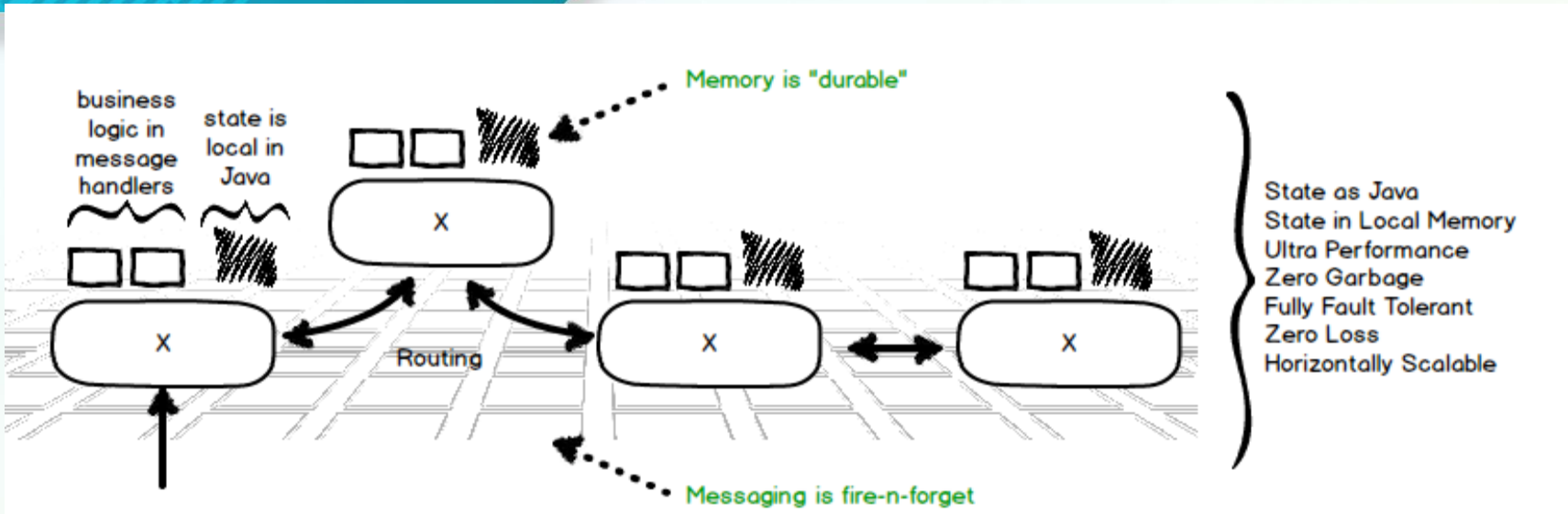
NOW WHAT IS THE PERFORMANCE?

- How Fast?
  - Latency
    - 10s of microseconds to low milliseconds
  - Throughput
    - 100s of thousands of transactions per second
- How Easy?
  - Model Objects and State in XML, generated into Java objects and collections.
  - Annotate methods as event handlers for message types.
  - Single threaded processing
  - Work with state objects treating memory as durable.
  - Send outbound messages as "Fire And Forget"
  - Shard applications by state, messages routed to right app.

RELIABILITY – EXTERNAL DATA STORES



STREAMING APPS ON THE X PLATFORM



- ✓ **Message Driven**
- ✓ **Stateful**
- ✓ Multi-Agent
- ✓ **Totally Available**
- ✓ **Horizontally Scalable**
- ✓ Ultra Performant

Building a Fleet Tracking System with The X Platform

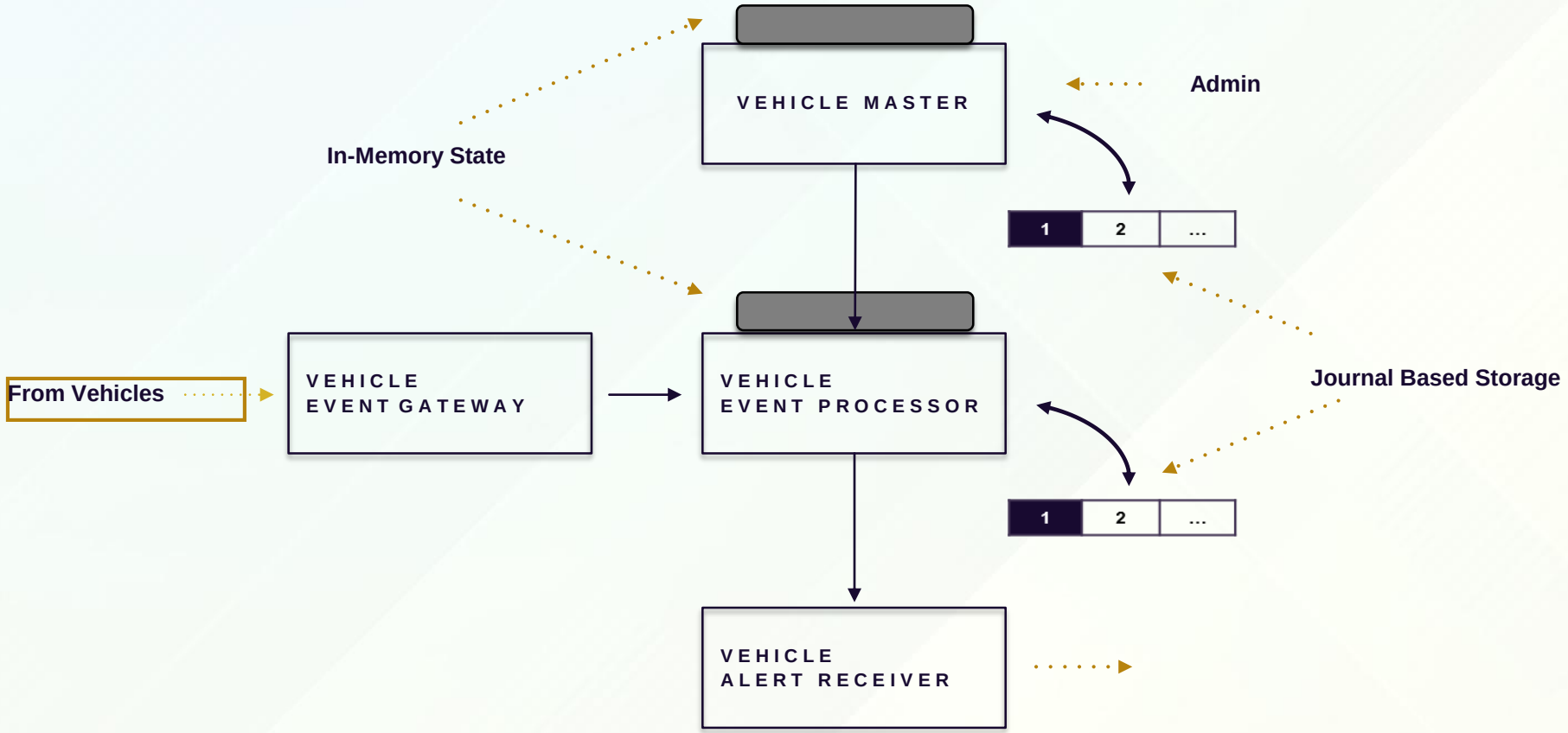
IMPLEMENTING GEOFENCING

- **We have a fleet of vehicles.**
  - (cars, trucks, whatever)

- **Each vehicle Should be following a route defined by Administrators**

- **Our Fleet Management System needs to:**
  - **Track location** of vehicles to ensure routes are being followed.
  - If a vehicle leaves its route, **trigger alerts**.

FLEET GEOFENCING



THE CODE

Message
Pkain Old Java Object
Generated from XML Model

State Management
Plain Old Java Objects
Generated from XML Model

Messaging
Annotation based handler discovery,
Single Threaded

```
@EventHandler
final public void onLocationEvent(final LocationEventMessage message, final Repository state) throws Exception {
    try {
        validate(message);
    }
    catch (Exception e) {
        invalidEventCount.increment();
        return;
    }

    // look up vehicle's route:
    final VehicleRoute route = _vehicleRoutes.get(message.getVehicleID());
    if (route == null) {
        droppedCount.increment();
        return;
    }

    // update state
    Vehicle vehicle = state.getVehicles().get(message.getVehicleID());
    if (vehicle == null) {
        vehicle = Vehicle.create();
        vehicle.setVehicleID(message.getVehicleID());
        state.getVehicles().put(message.getVehicleID(), vehicle);
    }

    // check if vehicle is within its route bounds, and alert if not
    if (!updateVehicleLocationIfInRouteBoundary(vehicle, route, message.getLocation(), message.getSpeed())) {
        RouteViolationEvent alert = populateRoutViolationEvent(vehicle, message.getLocation(), message.getSpeed());
        _messageSender.sendMessage("alerts", alert);
        alertCount.increment();
    }

    // update stats
    processedCount.increment();
}
```

State Management
Plain Old Java objects and Java Collections

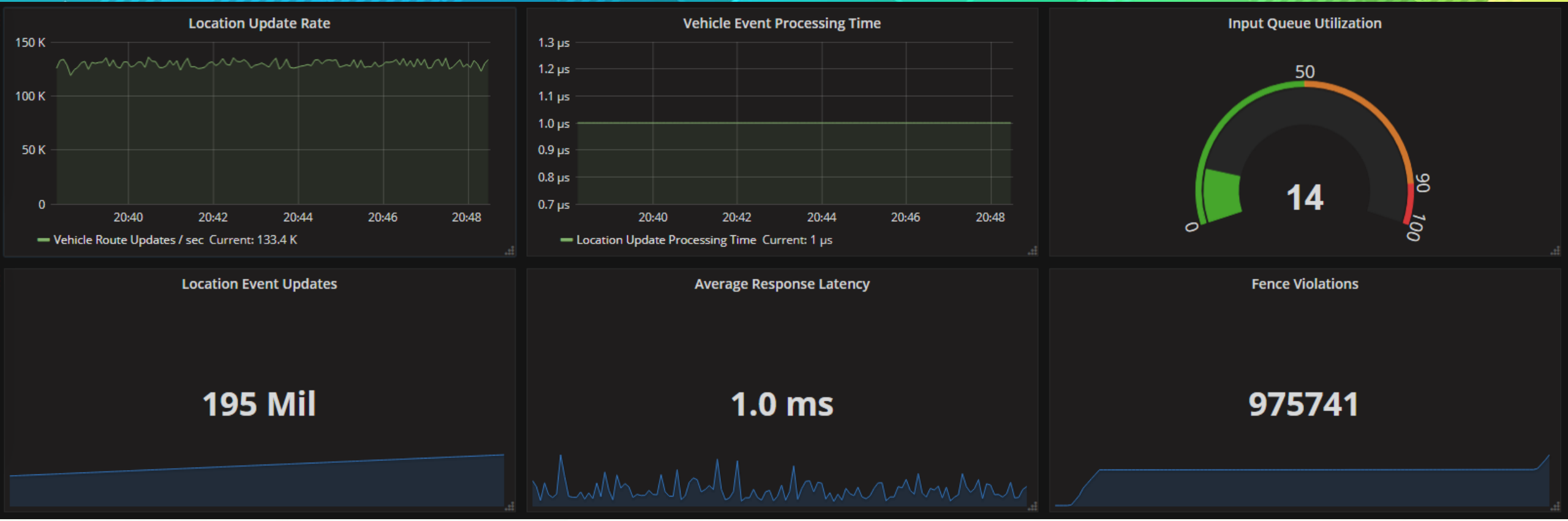
State Management
Object Pooling and Preallocation for Zero Garbage

State Management
State Changes transparently Replicated to Hot Backup and/or Disk Based Journal

Messaging
Create and populate "Fire and Forget"

Pure Business Logic – Exactly Once Processing

IOT FLEET GEOFENCING



Location Updates Events/sec: >130k

Single Shard, 1 Processor Core, Replicated.

1ms Response Time.

Full HA (Replicated), Exactly Once

In-Memory Computing SUMMIT 2017

WHY X?

- **Easy to Build**
  - Focus on domain
    - Pure Java
- **Easy to Maintain**
  - Pristine domain
    - No infrastructure bleed
- **Easy to Support**
  - Stock hardware
  - Small Footprint
  - Simple abstractions
  - Easy tools
- **Very, very fast**

✓ No Compromise

Agility, Availability, Scalability, Performance

GETTING STARTED WITH X PLATFORM™

Getting Started Guide

https://docs.neeveresearch.com

Get the Demo Source

https://github.com/neeveresearch/nvx-apps

We're Listening

contact@neeveresearch.com

QUESTIONS