Why and When To Consider Stream Processing Frameworks In Our Solutions

Soroosh Khodami - Software Architect at Code Nomads

Utrecht JUG Meetup

12 Feb , 2024 @ KeyLane Office

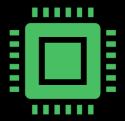
Agenda

- What is Stream Processing?
- Frameworks & Platforms
- Basic Concepts & Patterns
- Preview/Demo
- Benefits & Drawbacks + Considerations
- Use Cases For Different Industries
- How to start ?

This Talk is For

Software Developers

Tech Leads / Software Architects



Data Engineers / Data Scientist / AI Engineers

Product Owners / Product Managers / Business Analysts

beam

Flink

Spark Streaming

samza

akka

Slides & Code Repository Link Will Be Shared At The End

$ whoami

- I'm Soroosh Khodami
- Solution Architect @ Rabobank via Code Nomads
- Worked with Stream Processing at Scale in Bol.com
- Software Architecture Enthusiastic

@SorooshKh

linkedin.com/in/sorooshkhodami/

RIGHT TOOL FOR THE JOB

What is Stream Processing?

Event Processing?

Event Driven?

Wikipedia Definition

Stream processing

13 languages

Article Talk Read Edit View history Tools

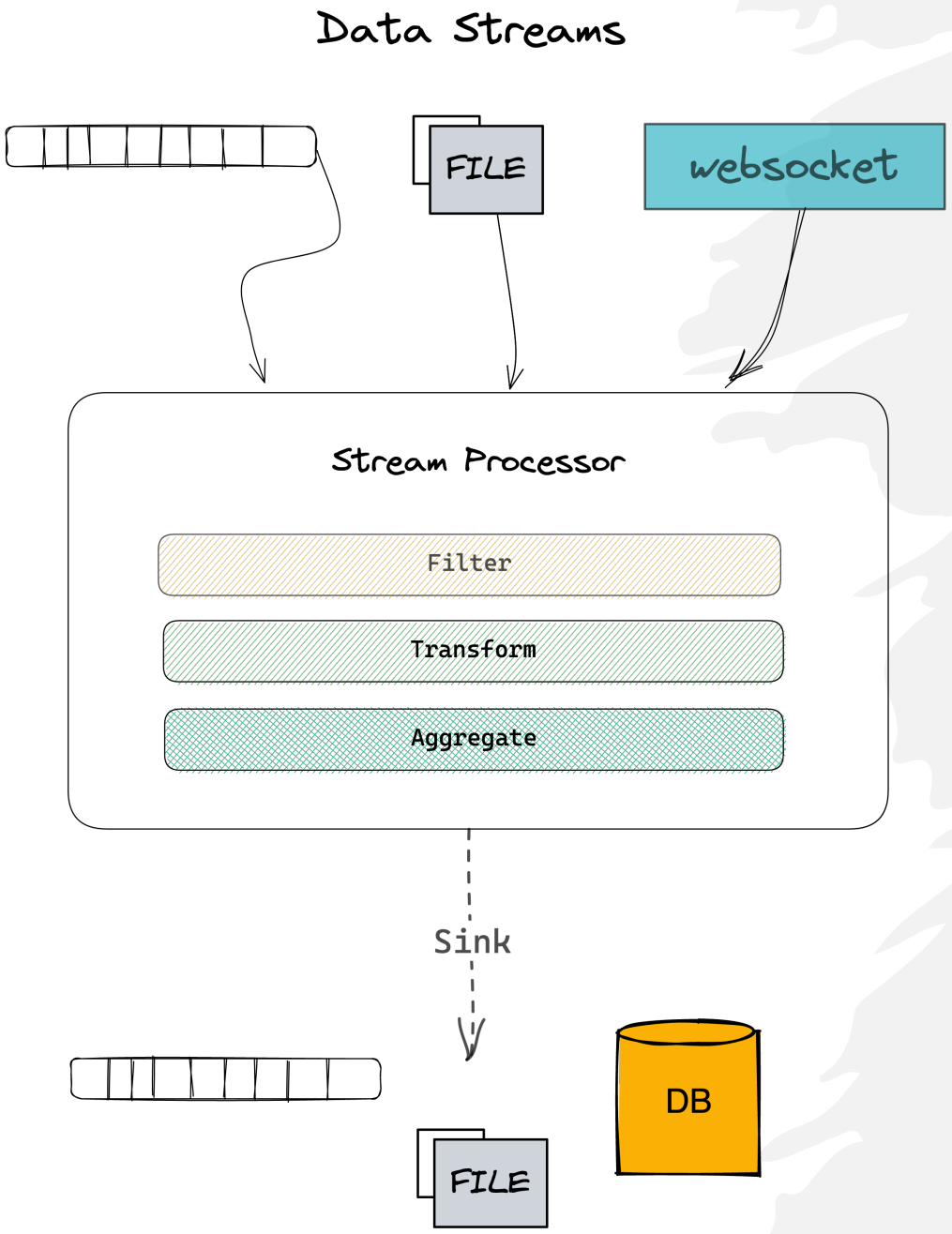
From Wikipedia, the free encyclopedia

In computer science, **stream processing** (also known as **event stream processing**, **data stream processing**, or **distributed stream processing**) is a programming paradigm which views data streams, or sequences of events in time, as the central input and output objects of computation. Stream processing encompasses dataflow programming, reactive programming, and distributed data processing.[1] Stream processing systems aim to expose parallel processing for data streams and rely on streaming algorithms for efficient implementation. The software stack for these systems includes components such as programming models and query languages, for expressing computation; stream management systems, for distribution and scheduling; and hardware components for acceleration including floating-point units, graphics processing units, and field-programmable gate arrays.[2]

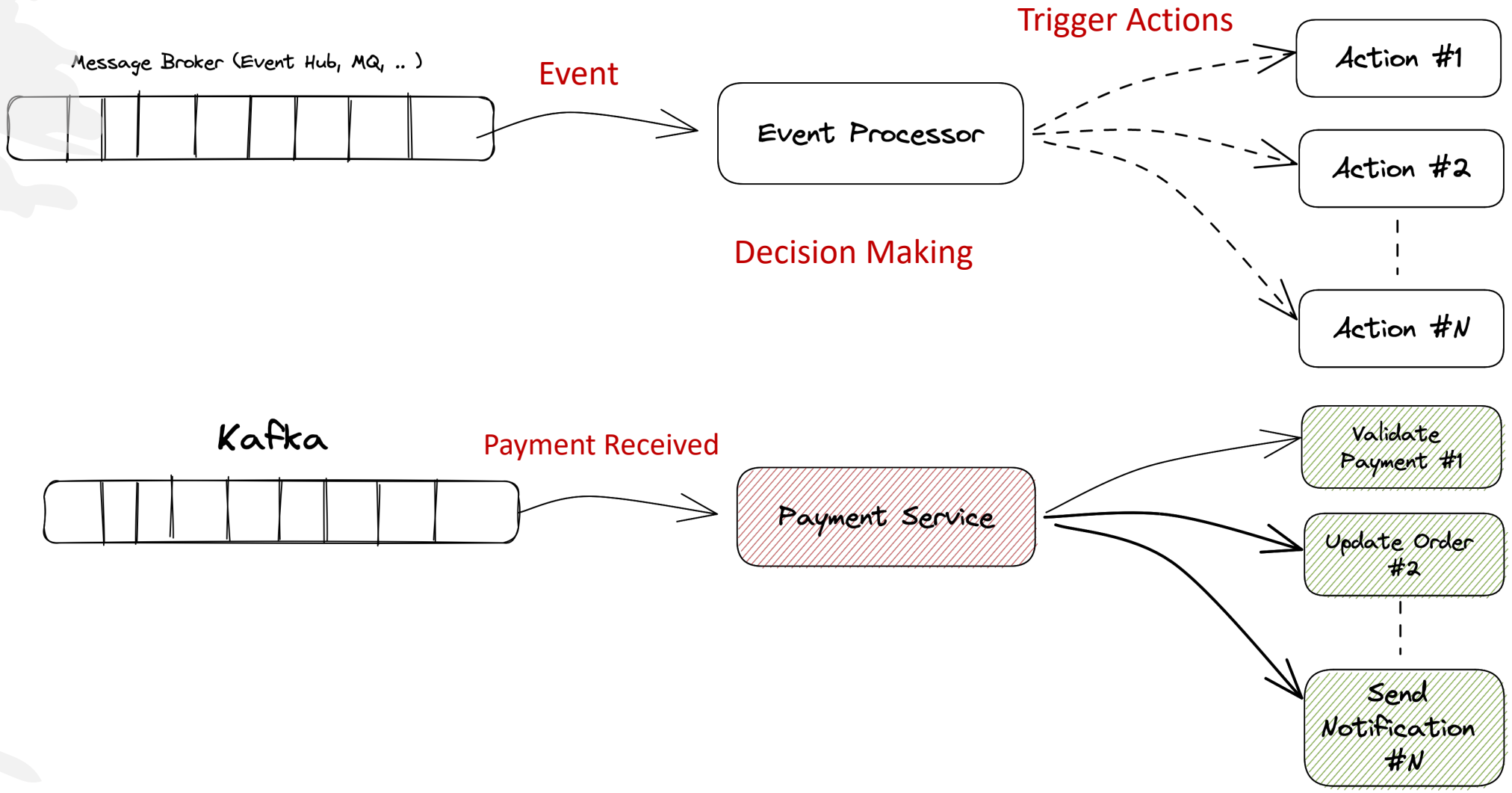
Ref: https://en.wikipedia.org/wiki/Stream_processing

Stream (Data) Processing

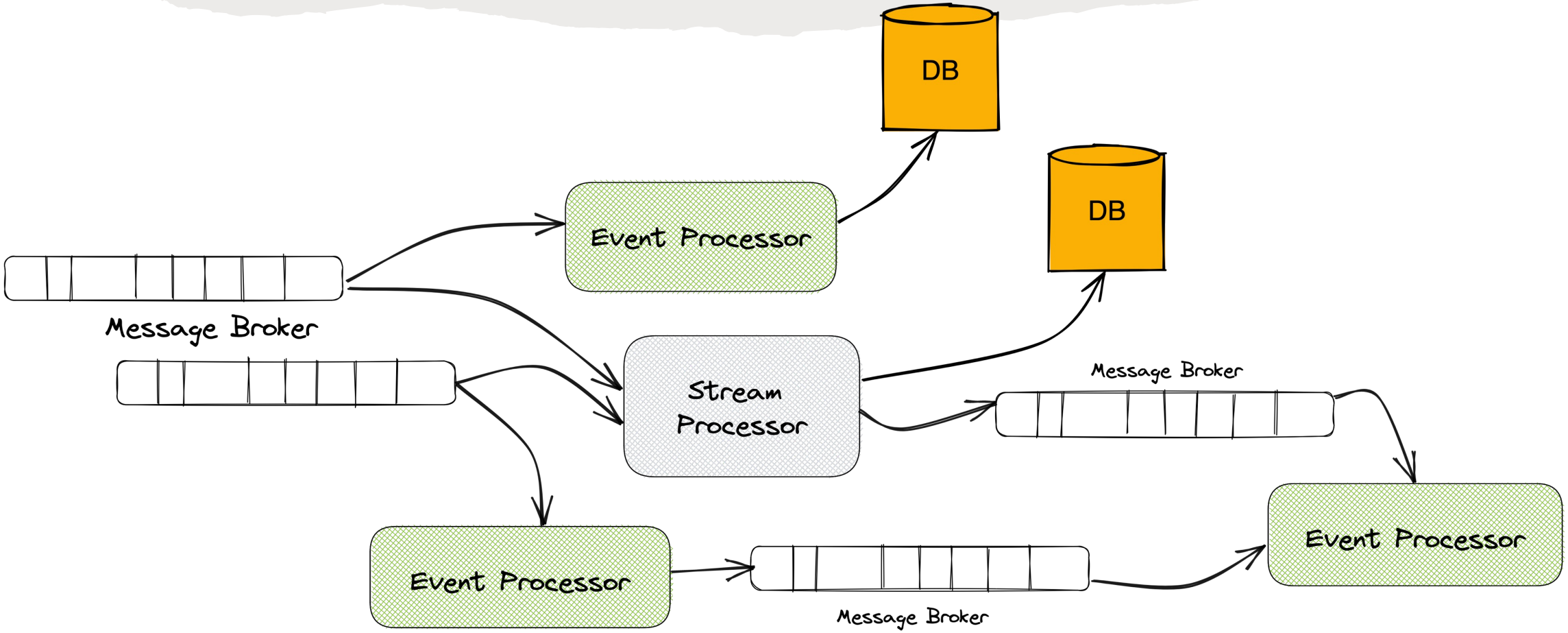
Stream data processing is a big data technique that focuses on continuously reading data, processing the data individually or joining it with related data sets in real-time or near real-time, and then sending the output to other applications, data-stores, or systems.



Event (Stream) Processing



Event Driven Architecture

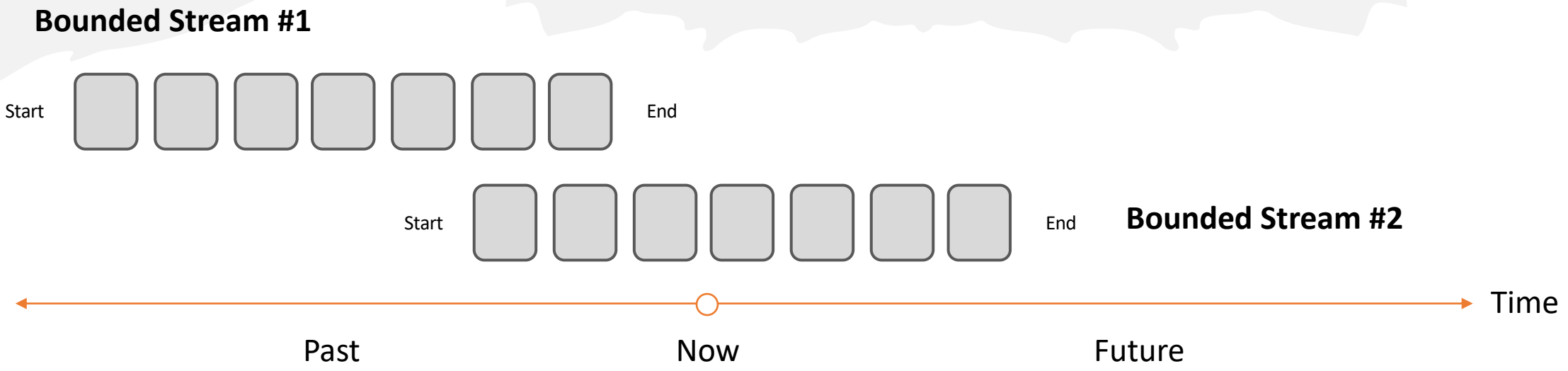


Stream Processing

Basic Concepts & Patterns

+ Examples

Bounded Stream / Unbounded Stream



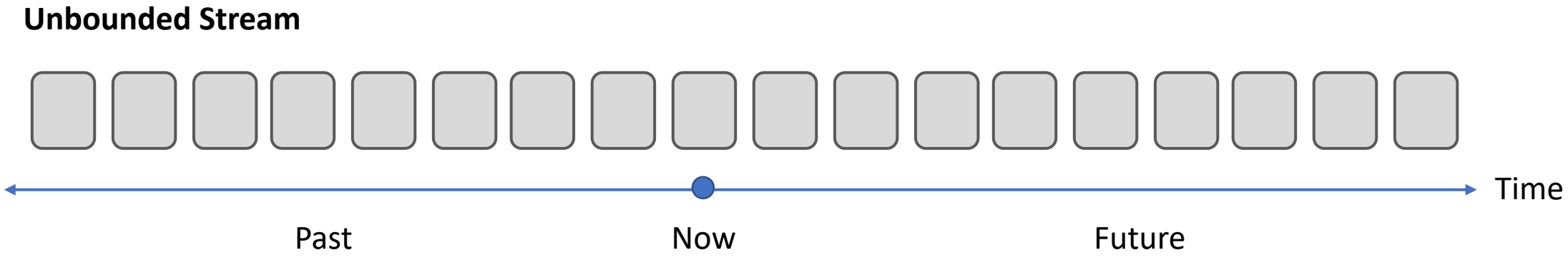
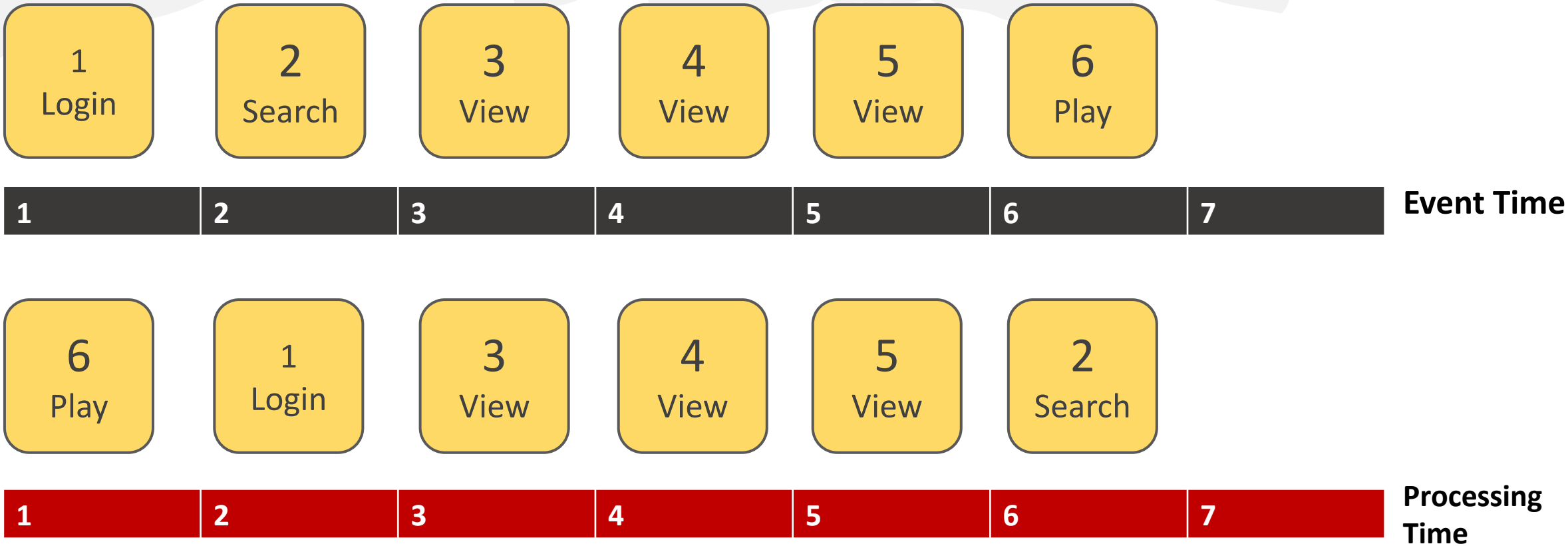


Photo: https://tclelectronics.co.nz/best-streaming-apps-for-new-zealand-2022/

Event Time & Processing Time



Delivery Guarantees

At Most Once

Messages can be lost, but never duplicated (Fire & Forget)

At Least Once

Messages can be duplicated

Exactly Once

Messages are delivered & processed exactly once

Learn More (Important)
Streaming Concepts - Exactly Once Fault Tolerance Guarantees youtube.com/watch?v=9pRsewtSPkQ
Rundown of Flink's Checkpoints - youtube.com/watch?v=hoLeQjoGBkQ
Understanding exactly-once processing and windowing in streaming pipelines - youtube.com/watch?v=DraQGkARegE

IoT Farm

Context

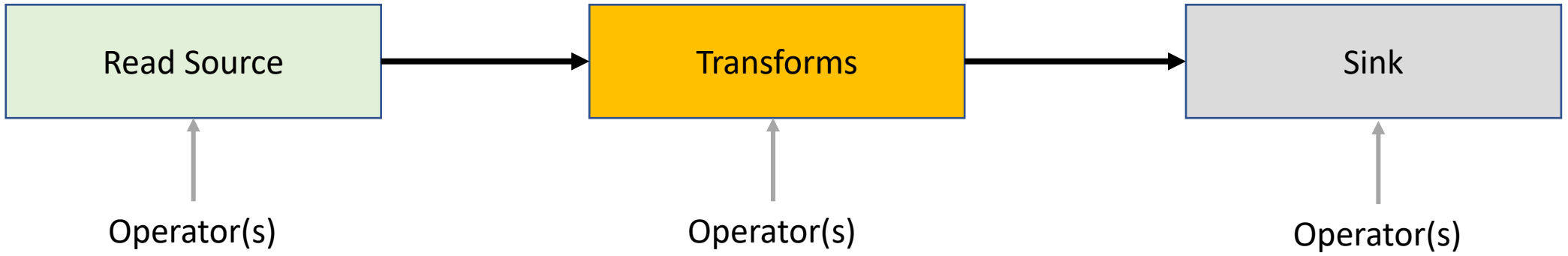
- +1000 Sensors
- Multiple Sensors per location
- Not reliable internet connection
- Large amount of continuous sensors data

Requirements

- Aggregated Sensors Data Per Location
- Correct Order Of Data
- No Duplicates/Double Processing

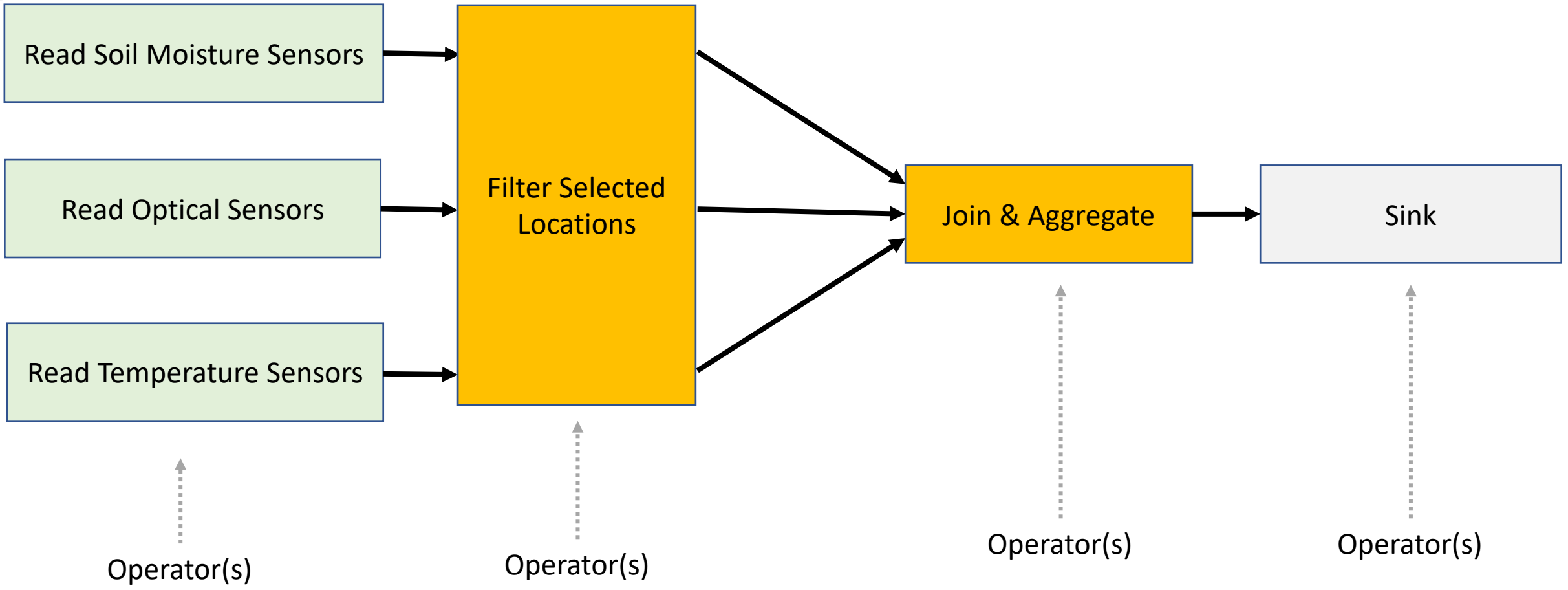
Operators & Transform

Basic Building Blocks

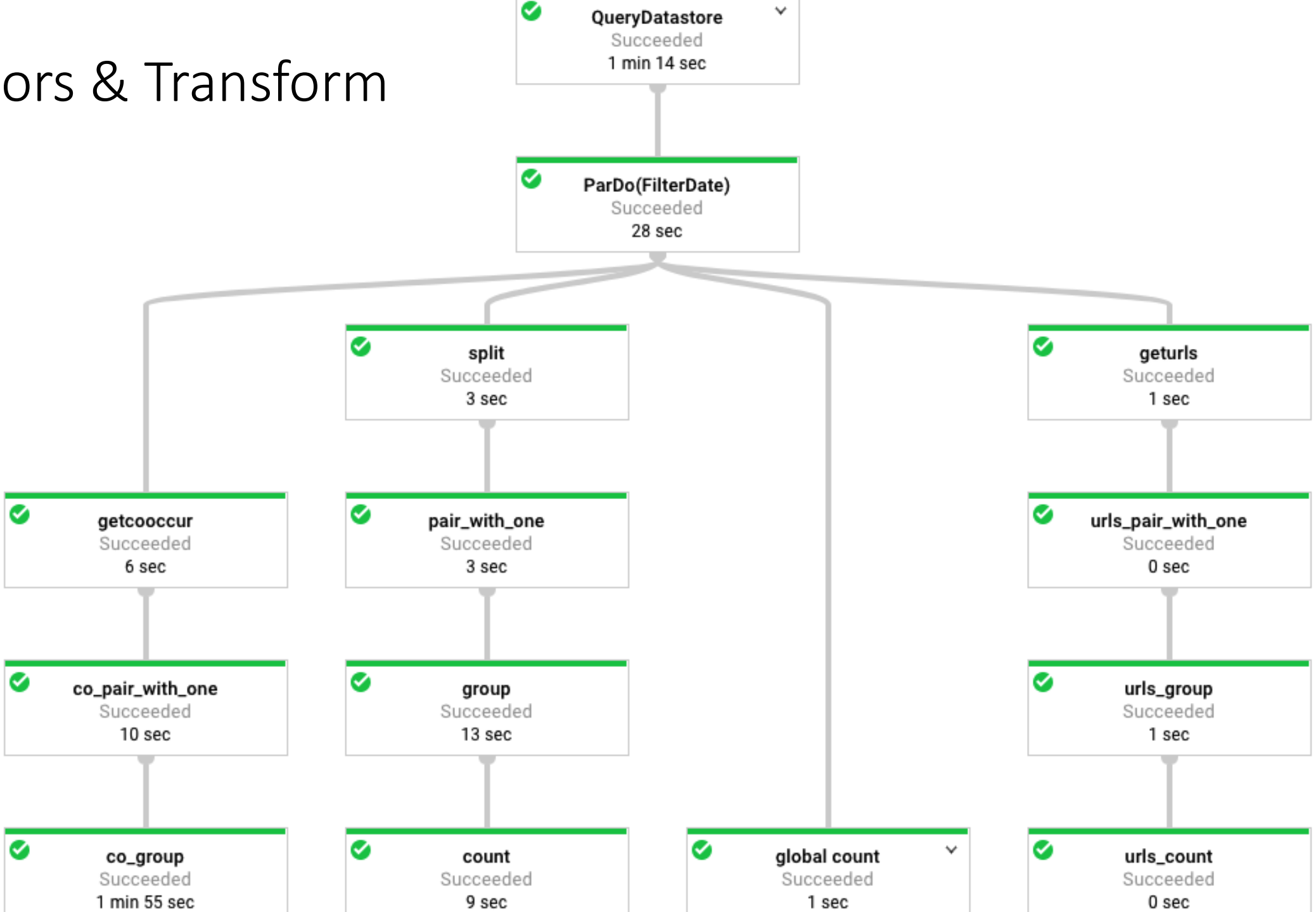


Operators & Transform

IOT Farm Example

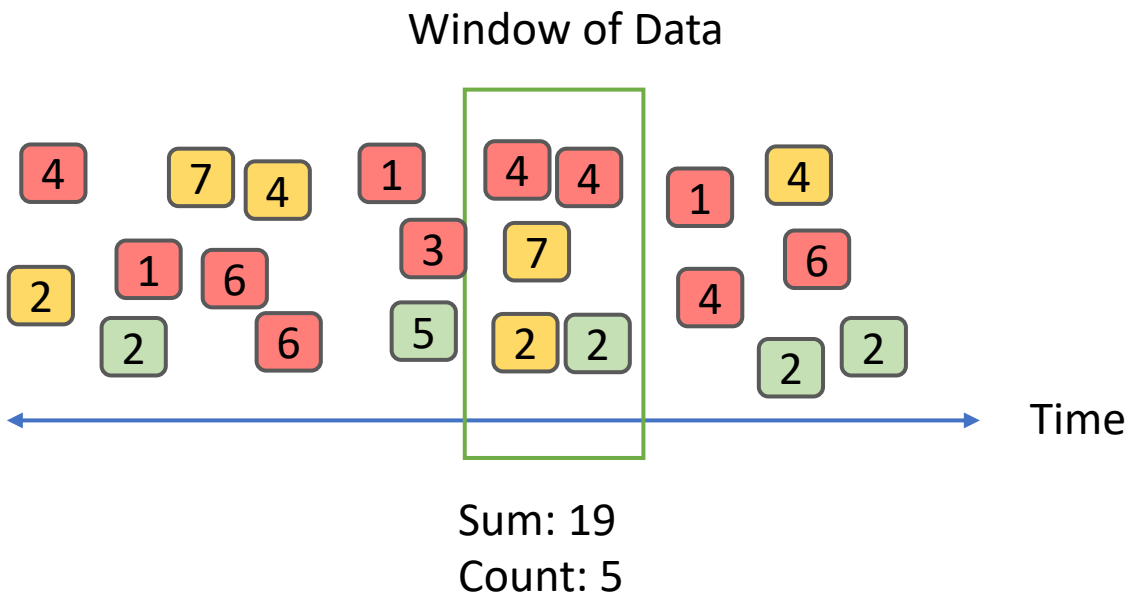


Operators & Transform



Windowing

- **Divides** an unbounded, continuous data stream **into smaller, finite segments**
- Allows to **perform operations** and calculations on manageable **chunks of data.**
- It's not feasible to load/keep entire stream into memory
- Useful for analyzing data over specific time periods or fixed numbers of events.

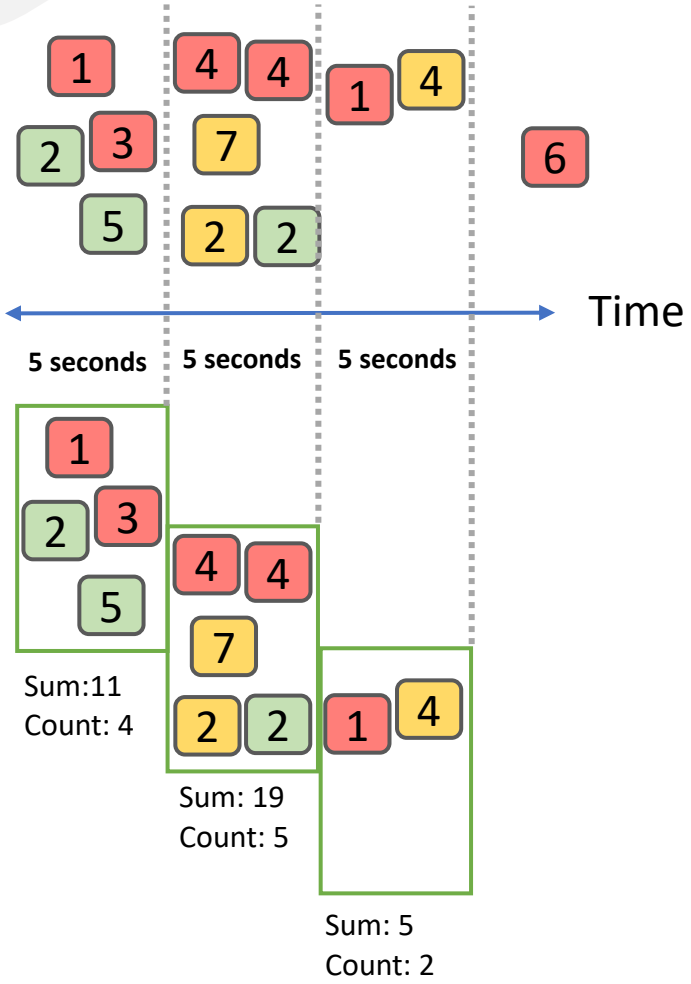


Learn More
Basics of Windowing - https://www.youtube.com/watch?v=oJ-LueBvOcM&t=1s
Advanced Windowing Concepts - https://www.youtube.com/watch?v=MuFA6CSti6M

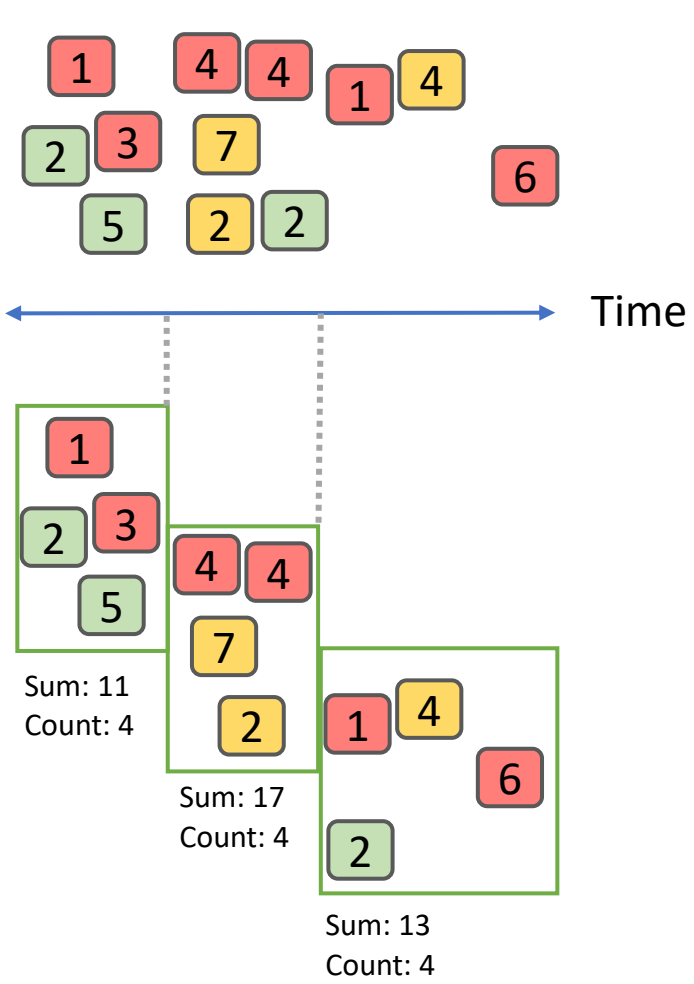
Tumbling/Fixed Window

No Overlaps between windows elements

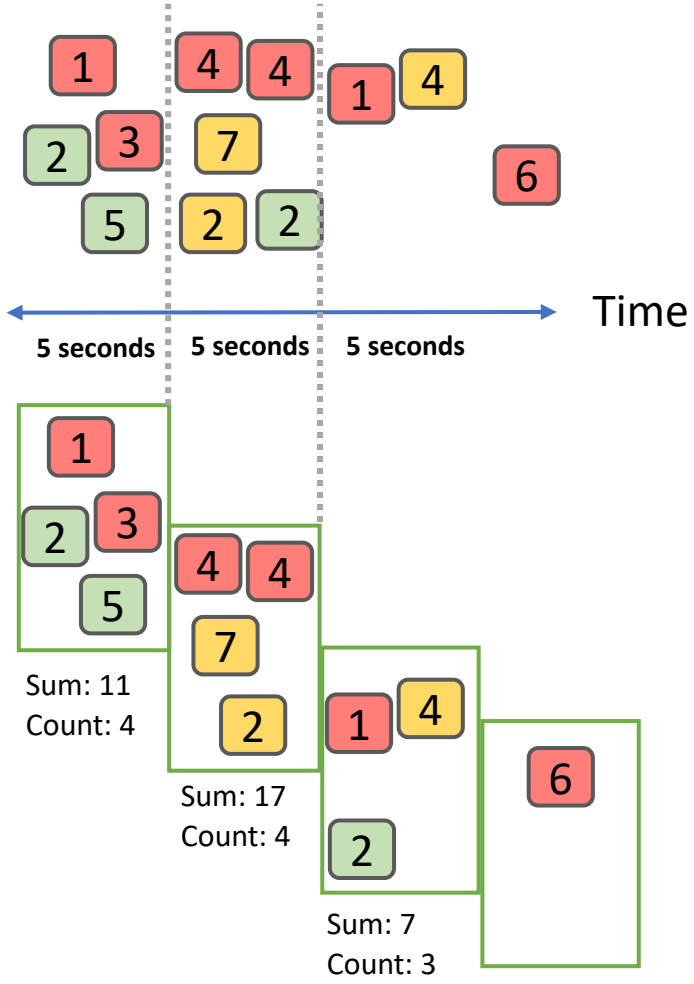
Time Based Windows



Size Based Windows



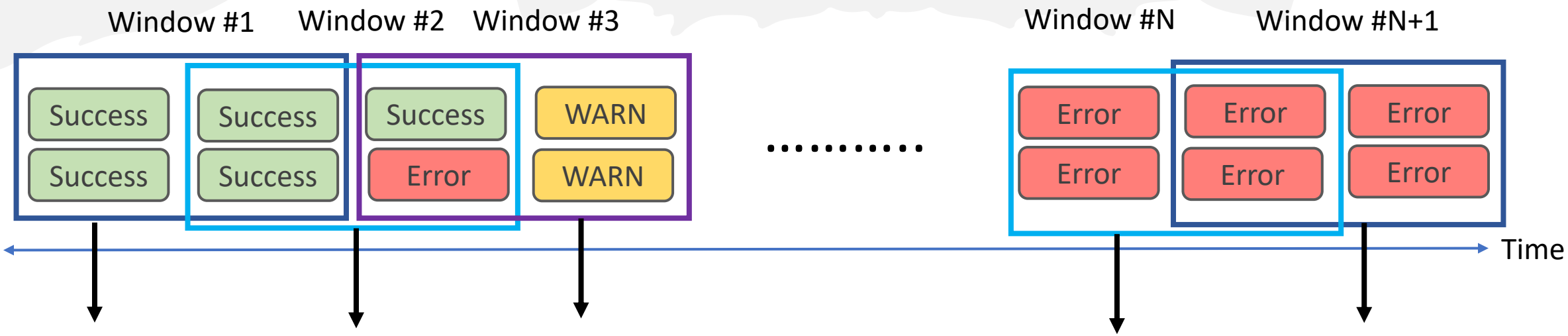
Time & Size Based Windows



Sliding Window

Last 10 Second Every 5 Seconds + Overlaps Between Windows

Time Based



Success : 4
Warn : 0
Error : 0

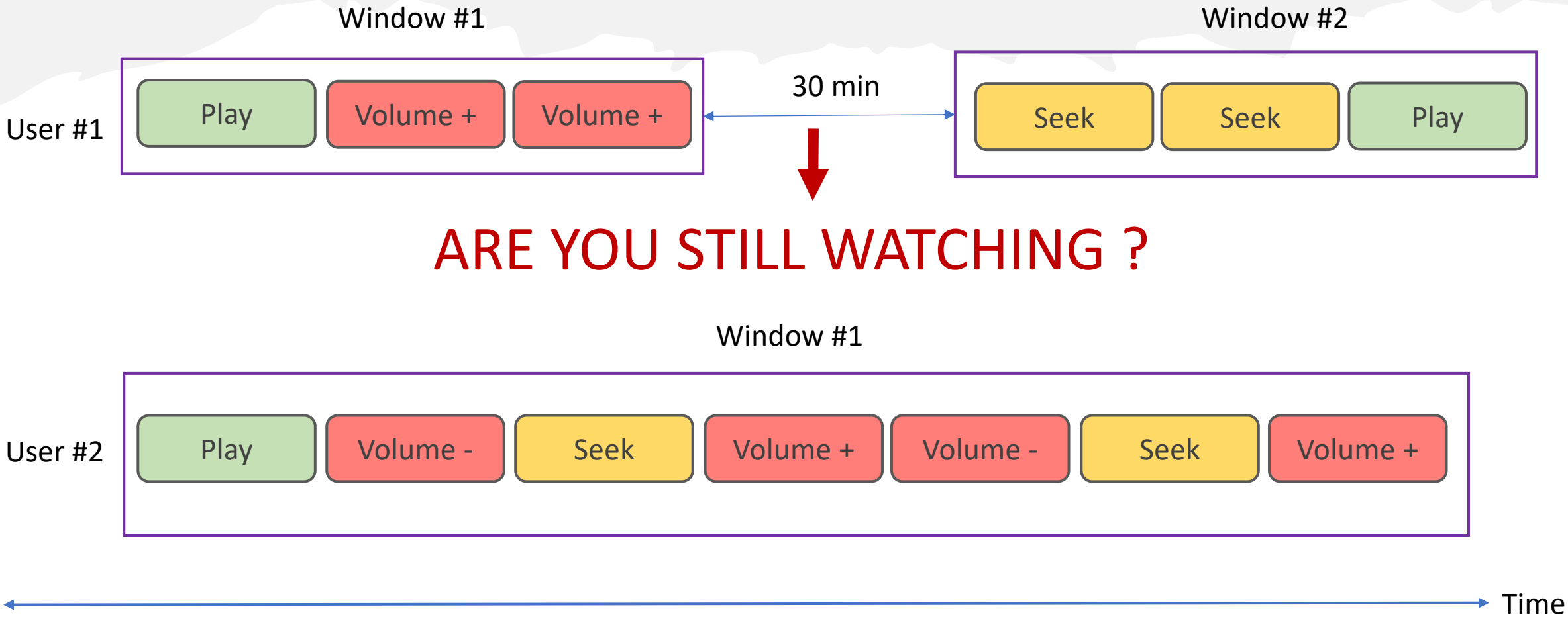
Success : 3
Warn : 0
Error : 1

Success : 1
Warn : 2
Error : 1

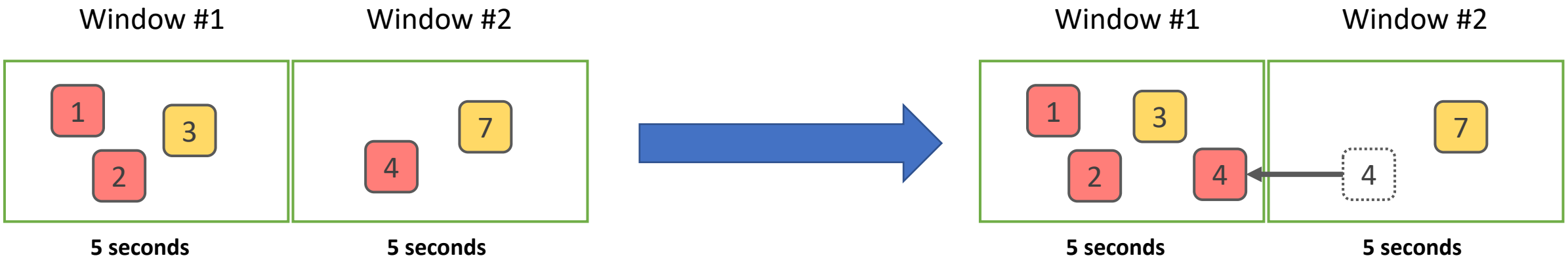
Success : 0
Warn : 0
Error : 4

Session Window

Close the window based on GAP Duration > 10 min



Watermarks



Learn More

Basics of Windowing - https://www.youtube.com/watch?v=oJ-LueBvOcM&t=1s

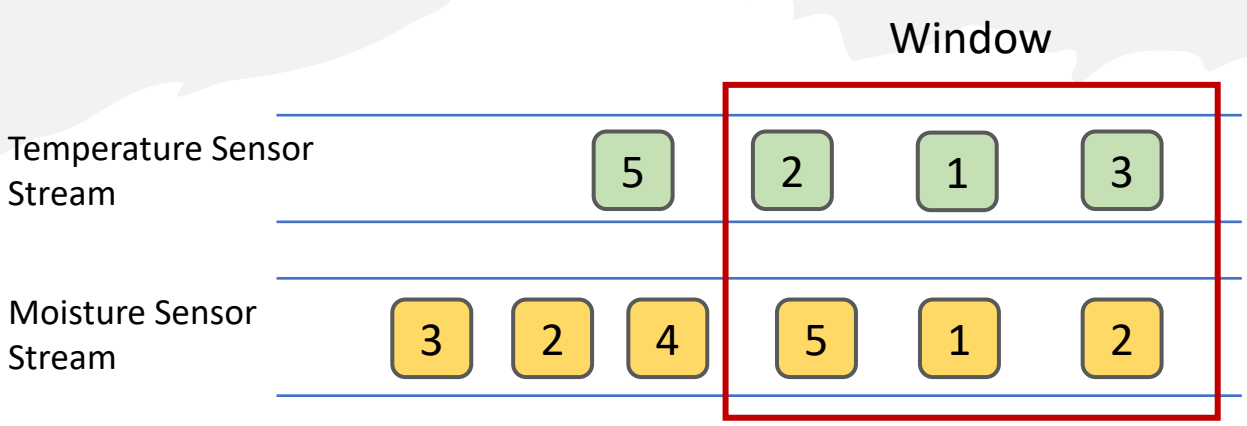
Advanced Windowing Concepts - https://www.youtube.com/watch?v=MuFA6CSti6M

Basic Concepts & Patterns

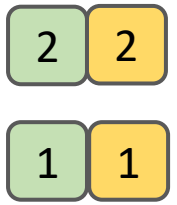
- ✓ Bounded Stream / Unbounded Stream
- ✓ Operators & Transforms
- ✓ Event Time & Processing Time
- ✓ Event Delivery Guarantee
- ✓ Windowing (Fixed , Sliding, Session, Watermark)
- ❑ States & Stateful Stream Processing
- ❑ Joining Streams & Enrichment Pattern

Apache Spark
kafka Streams
beam
amazon KINESIS
Google Dataflow
Stream Analytics
Flink

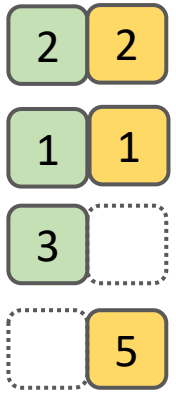
Joining Streams & Enrichment Pattern



Window Inner Join



Window Cross Join
(CoGroup)

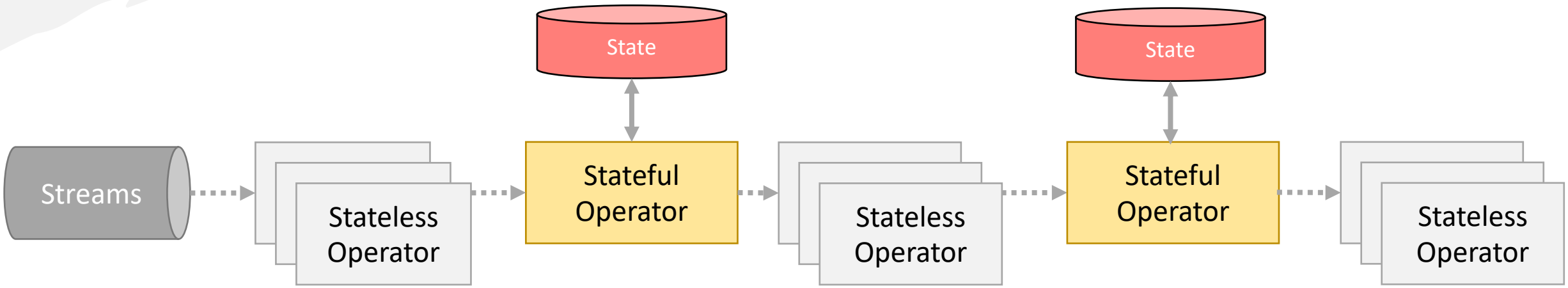


Learn More

Stream Join in Flink: from Discrete to Continuous - Xingcan Cui https://www.youtube.com/watch?v=3YVRIuJUKIw

Webinar: 99 Ways to Enrich Streaming Data with Apache Flink - Konstantin Knauf - https://www.youtube.com/watch?v=cJS18iKLUIY

States & Stateful Stream Processing



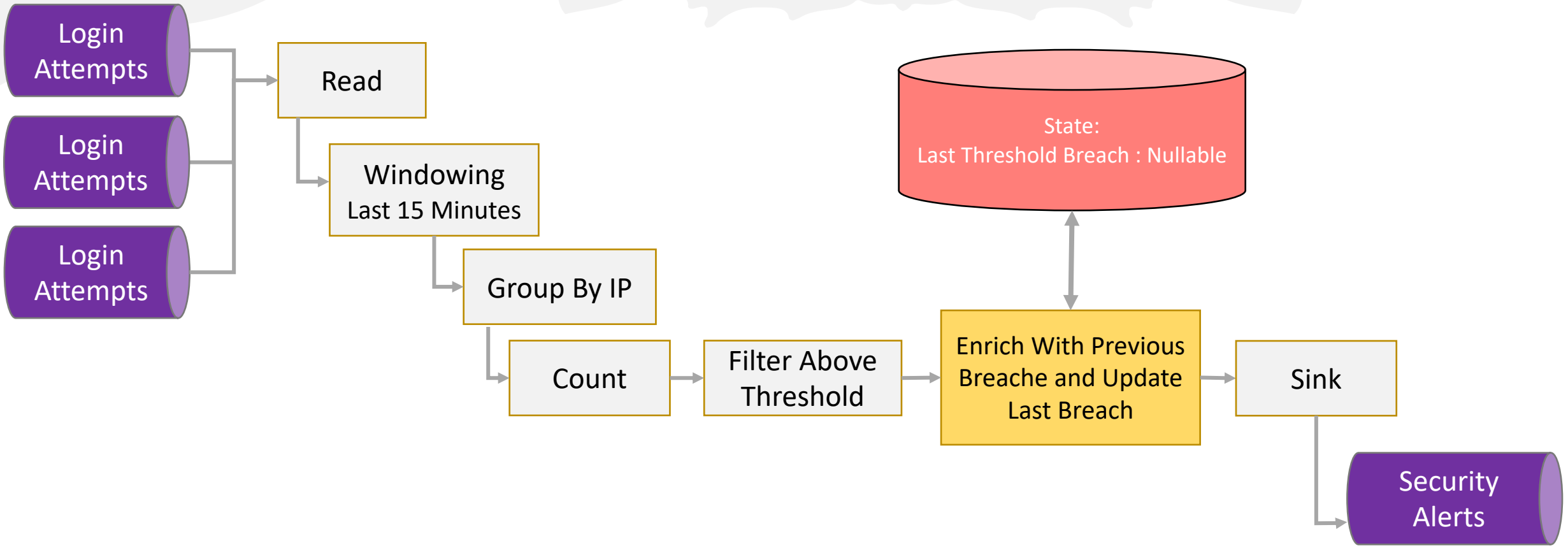
Learn More

Introduction to Stateful Stream Processing with Apache Flink - Robert Metzger https://www.youtube.com/watch?v=DkNeyCW-eH0

Webinar: Deep Dive on Apache Flink State - Seth Wiesman - https://www.youtube.com/watch?v=9GF8Hwqzwnk

States & Stateful Stream Processing

Brute Force Login Monitoring

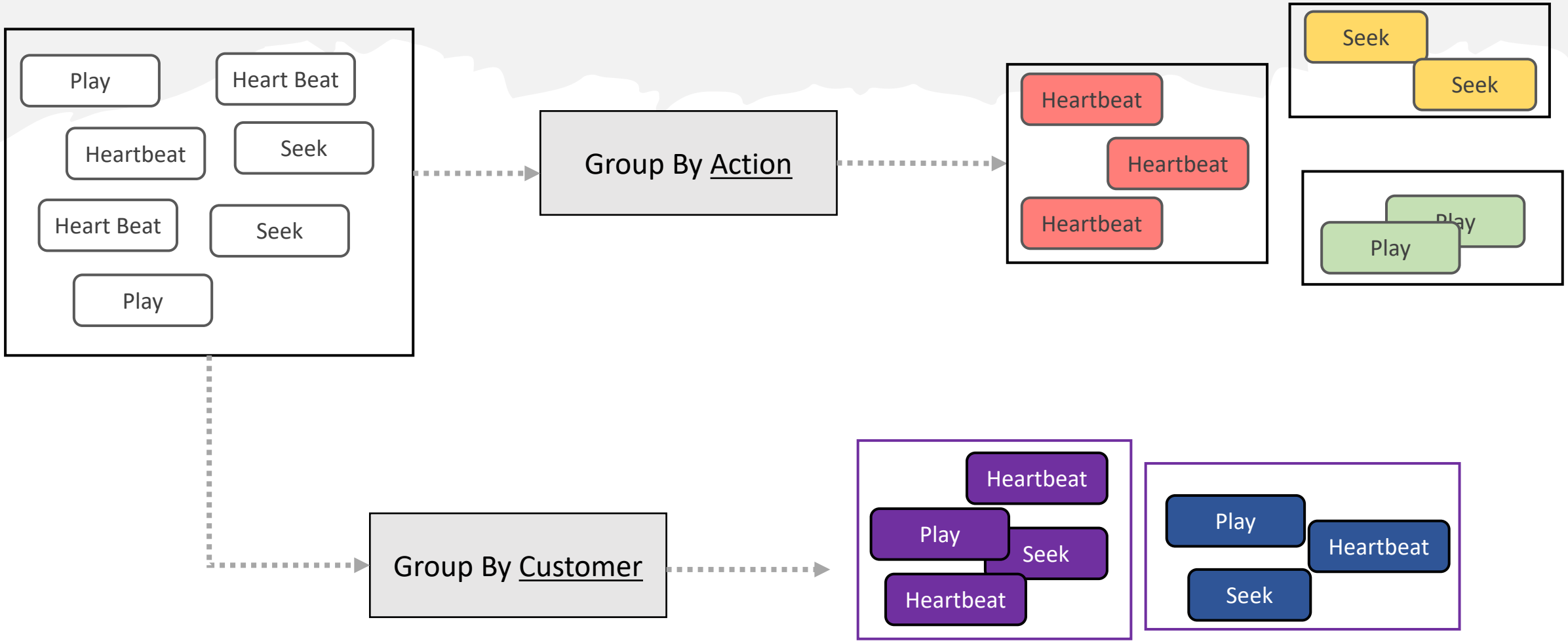


Learn More

Introduction to Stateful Stream Processing with Apache Flink - Robert Metzger https://www.youtube.com/watch?v=DkNeyCW-eH0

Webinar: Deep Dive on Apache Flink State - Seth Wiesman - https://www.youtube.com/watch?v=9GF8Hwqzwnk

Group By Key / KeyBy [4Geeks]



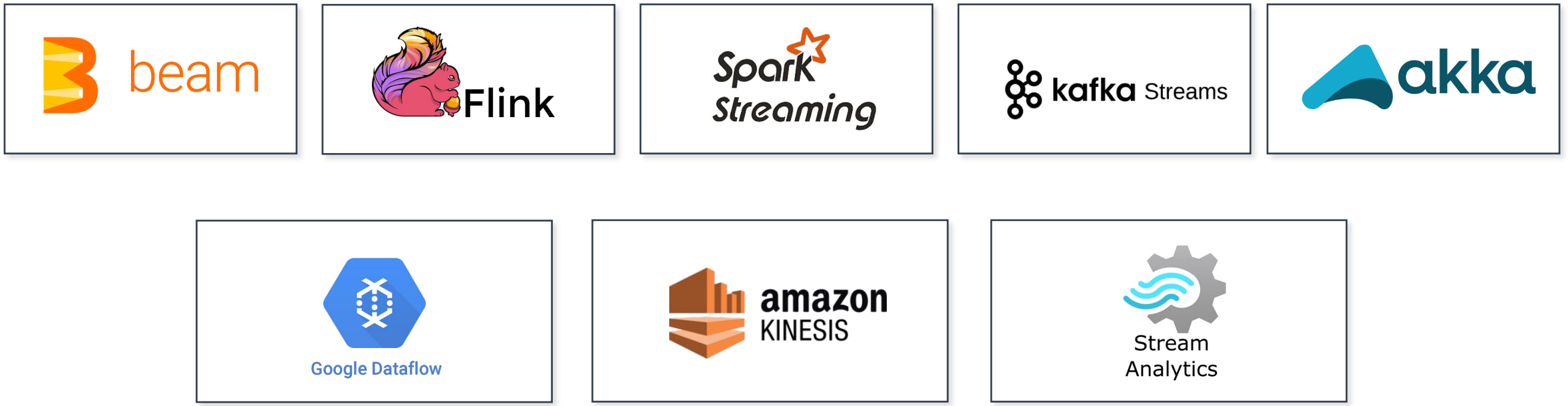
Learn More

Apache Flink Specifying Keys https://medium.com/big-data-processing/apache-flink-specifying-keys-81b3b651469

Branching & merging PCollections with Apache Beam - https://youtu.be/RYD40js20a4

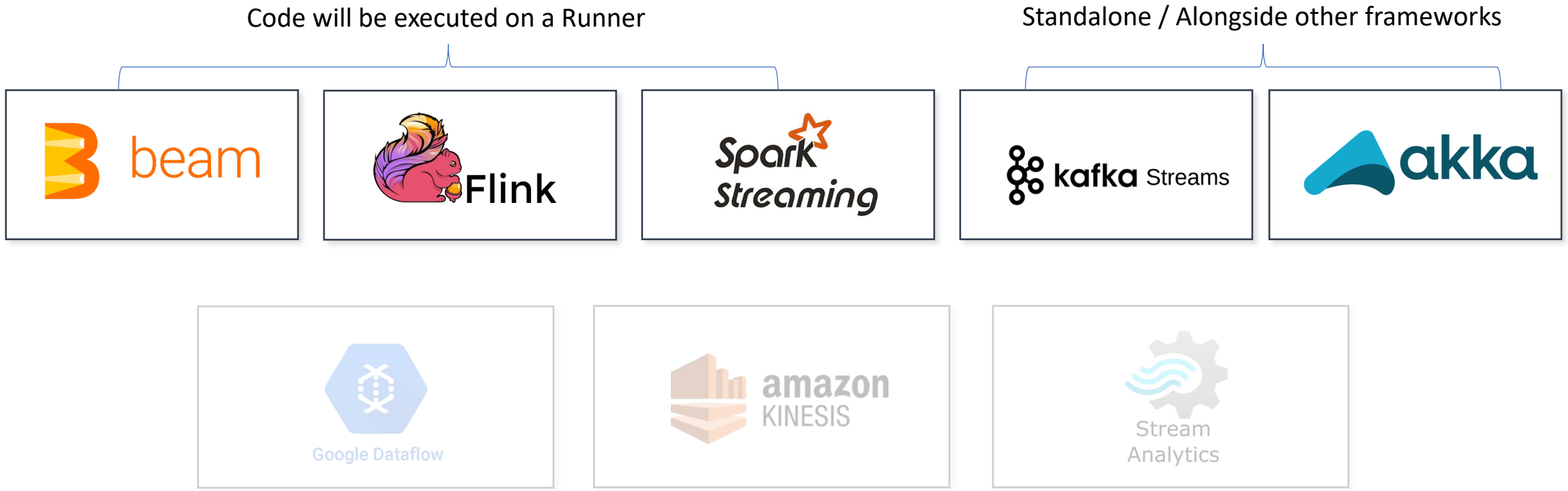
Stream Processing

Frameworks & Platforms



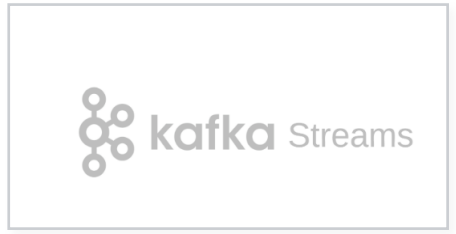
Stream Processing Universe

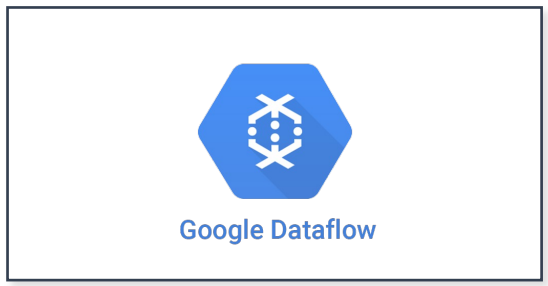
2023



Stream Processing Universe

2023





Cloud Platforms

Stream Processing Universe

2023

Hardened at Scale

Powered By Flink https://flink.apache.org/powered-by/

DEMO TIME

Apache Beam Code

“ Talk is cheap. Show me the code. ”

Linus Torvalds

IP Monitoring (Apache Beam)

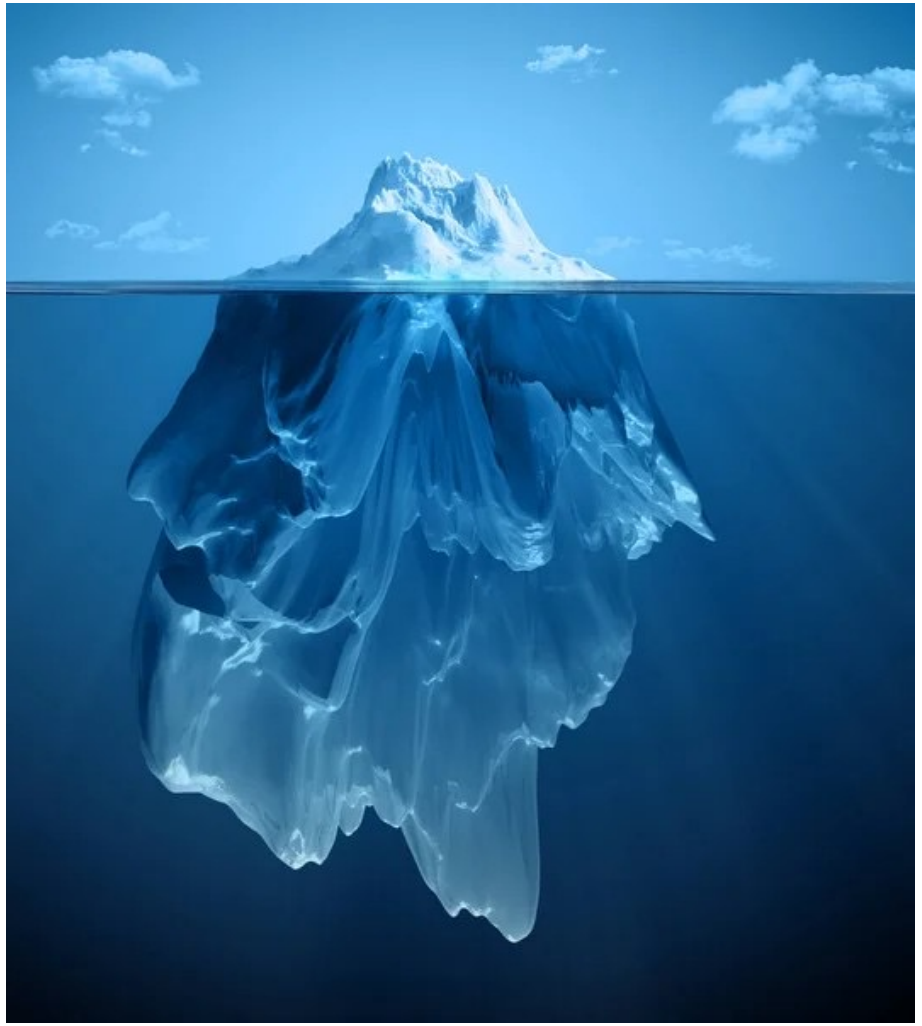
```
val failedLogins = p.apply("Read PubSub Messages", readFromPubSubSubscription())

val ipCounts = failedLogins
    .apply("Window", failedLoginWindowingStrategy())
    .apply("Map to KV <IP,MSG>", mapToKVIPAddr())
    .apply("Group by Key IP-Addr", GroupByKey.create())
    .apply("Count per IP", countNumberOfAttempts())

val alerts = ipCounts
    .apply("Filter by Threshold", isCountOfAttempAboveThresholdFilter())
    .apply("Enrich with Old Breaches Last Month", enrichWithOldBreachesLastMonth())

alerts.apply("Write Alerts to PubSub", publishToPubSubTopic())
```

IP Monitoring (Apache Beam)

```
val failedLogins = p.apply("Read PubSub Messages", readFromPubSubSubscription())

val ipCounts = failedLogins
    .apply("Window", failedLoginWindowingStrategy())
    .apply("Map to KV <IP,MSG>", mapToKVIPAddr())
    .apply("Group by Key IP-Addr", GroupByKey.create())
    .apply("Count per IP", countNumberOfAttempts())

val alerts = ipCounts
    .apply("Filter by Threshold", isCountOfAttempAboveThresholdFilter())
    .apply("Enrich with Old Breaches Last Month", enrichWithOldBreachesLastMonth())

alerts.apply("Write Alerts to PubSub", publishToPubSubTopic())
```

What You Just Saw

Hidden Code Behind
The Functions



Order Enrichment With Customer Data [4Geeks]

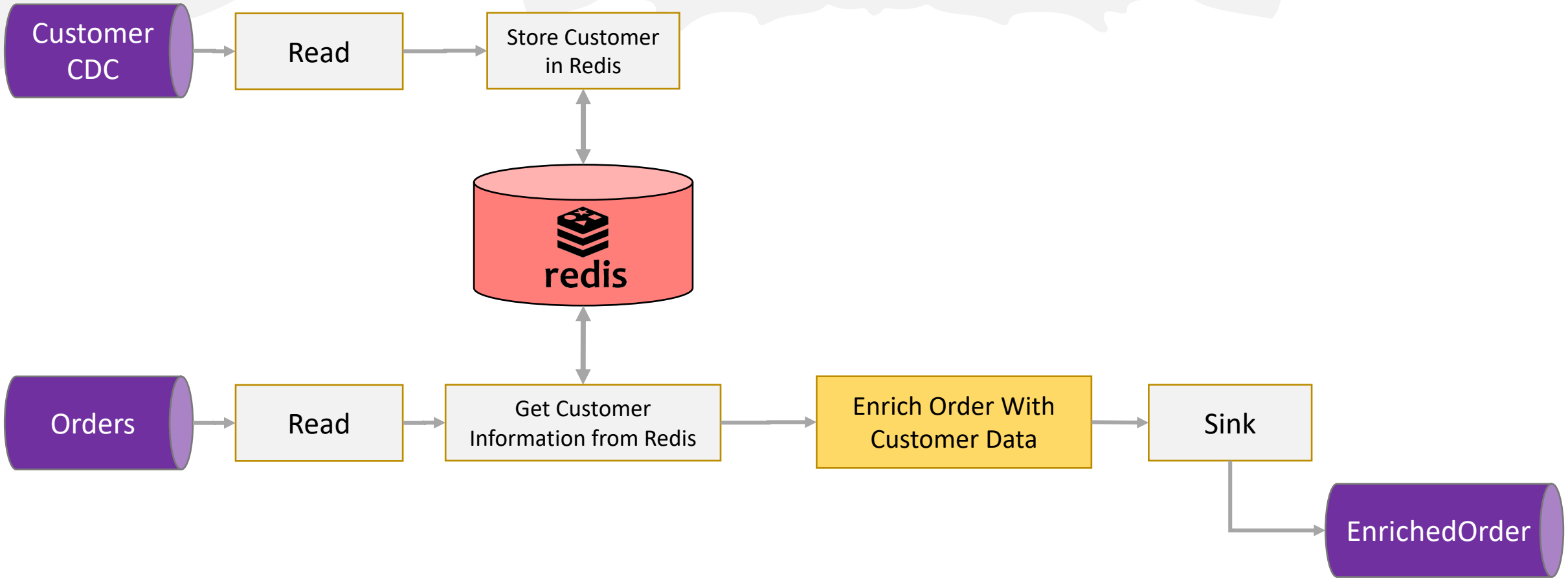
Apache Beam + Dataflow vs Spring Boot + Redis

@SorooshKh

Code Repository & Slides

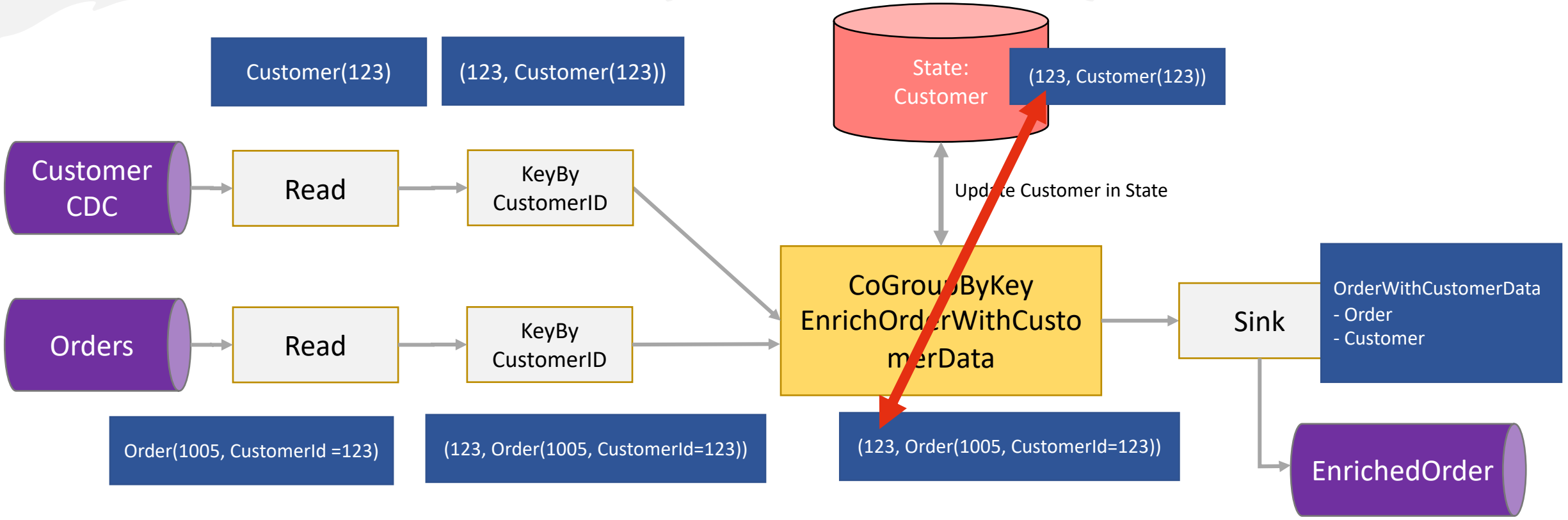
Order Enrichment With Customer Data [4Geeks]

Spring Boot + Redis



Order Enrichment With Customer Data [4Geeks]

Apache Beam + Dataflow



Learn More

Stream Join in Flink: from Discrete to Continuous - Xingcan Cui https://www.youtube.com/watch?v=3YVRluJUKIw

Webinar: 99 Ways to Enrich Streaming Data with Apache Flink - Konstantin Knauf - https://www.youtube.com/watch?v=cJS18iKLUIY

Order Enrichment (Beam) Code

```
PCollection<Customer> customersStream = readCustomers(pipeline)
        .apply("Windowing Customer", myDefaultWindowStrategy());


PCollection<Order> ordersStream = readOrdersStream(pipeline)
        .apply("Windowing Order", myDefaultWindowStrategy());

PCollection<KV<Long, Customer>> kvCustomerIdCustomer = customersStream
        .apply(
                WithKeys.of(Customer::getId)
                .withKeyType(TypeDescriptor.of(Long.class)));

PCollection<KV<Long, Order>> kvCustomerIdOrder = ordersStream
        .apply(
                WithKeys.of(Order::getCustomerId)
                .withKeyType(TypeDescriptor.of(Long.class)));


PCollection<KV<Long, CoGbkResult>> result =
                 KeyedPCollectionTuple.of(CUSTOMER_TUPLE_TAG, kvCustomerIdCustomer)
                .and(ORDER_TUPLE_TAG, kvCustomerIdOrder)
                .apply(CoGroupByKey.create());

PCollectionTuple enrichedOrdersAndErrors = result.apply(ParDo.of(new EnrichmentDoFn())
        .withOutputTags(ENRICHED_ORDER_WITH_CUSTOMER_DATA_TUPLE_TAG, TupleTagList.of(ENRICHMENT_ERROR_TUPLE_TAG)));

enrichedOrdersAndErrors.get(ENRICHED_ORDER_WITH_CUSTOMER_DATA_TUPLE_TAG)
        .apply("Map to String", ParDo.of(new MapToString()))
        .apply("Sending Pub/sub", PubsubIO.writeStrings().to("projects/stream-proc/topics/enriched-orders-dataflow"));

return pipeline.run();
```

Order Enrichment (Beam) Code

```
PCollection<Customer> customersStream = readCustomers(pipeline)
        .apply("Windowing Customer", myDefaultWindowStrategy());


PCollection<Order> ordersStream = readOrdersStream(pipeline)
        .apply("Windowing Order", myDefaultWindowStrategy());

PCollection<KV<Long, Customer>> kvCustomerIdCustomer = customersStream
        .apply(
                WithKeys.of(Customer::getId)
                .withKeyType(TypeDescriptor.of(Long.class)));

PCollection<KV<Long, Order>> kvCustomerIdOrder = ordersStream
        .apply(
                WithKeys.of(Order::getCustomerId)
                .withKeyType(TypeDescriptor.of(Long.class)));


PCollection<KV<Long, CoGbkResult>> result =
                 KeyedPCollectionTuple.of(CUSTOMER_TUPLE_TAG, kvCustomerIdCustomer)
                .and(ORDER_TUPLE_TAG, kvCustomerIdOrder)
                .apply(CoGroupByKey.create());

PCollectionTuple enrichedOrdersAndErrors = result.apply(ParDo.of(new EnrichmentDoFn())
        .withOutputTags(ENRICHED_ORDER_WITH_CUSTOMER_DATA_TUPLE_TAG, TupleTagList.of(ENRICHMENT_ERROR_TUPLE_TAG)));

enrichedOrdersAndErrors.get(ENRICHED_ORDER_WITH_CUSTOMER_DATA_TUPLE_TAG)
        .apply("Map to String", ParDo.of(new MapToString()))
        .apply("Sending Pub/sub", PubsubIO.writeStrings().to("projects/stream-proc/topics/enriched-orders-dataflow"));

return pipeline.run();
```

Insights

Order Enrichment Test Results

Input : 1 million Customer Msg + 120k Order Msg

Expected Output: 120k CustomerEnriched message

Spring Boot

Tested on Kubernetes Pod on GCP (2 GB Ram , 2 x GCP CPU Core)
120k message processed in 5 minutes

~ 400 msg/second
Lower Costs
For Keeping Job Running

Apache Beam + Dataflow

1 Dataflow Worker with Default Spec (4 vCPU, 15 GB memory)
120k message processed in 3 minutes

~ 700 msg/second
Higher Costs
For Keeping Job Running

Note: Please note that the insights provided above are not derived from a fully accurate benchmark.

Why Should We Consider It

Benefits, Drawbacks & Considerations

Stream Processing Frameworks

Benefits & Drawbacks

Benefits

- ✓ Fast & High-Throughput
- ✓ Easy to Scale
- ✓ Exactly Once Processing / Fault Tolerant
- ✓ Customizable
- ✓ Advanced features in scale: Windowing, Watermarks, Stateful Functions and ..

Drawbacks

- Complexity
- Implementation & Maintenance
- Testing & Debugging is challenging
- Changing the data pipelines are hard
- Error handling is not simple

Stream Data Integration vs Stream Analytics

Stream Data Integration

- Reading Input
- Map
- Filter
- Simple Enrich

Stream Analytics

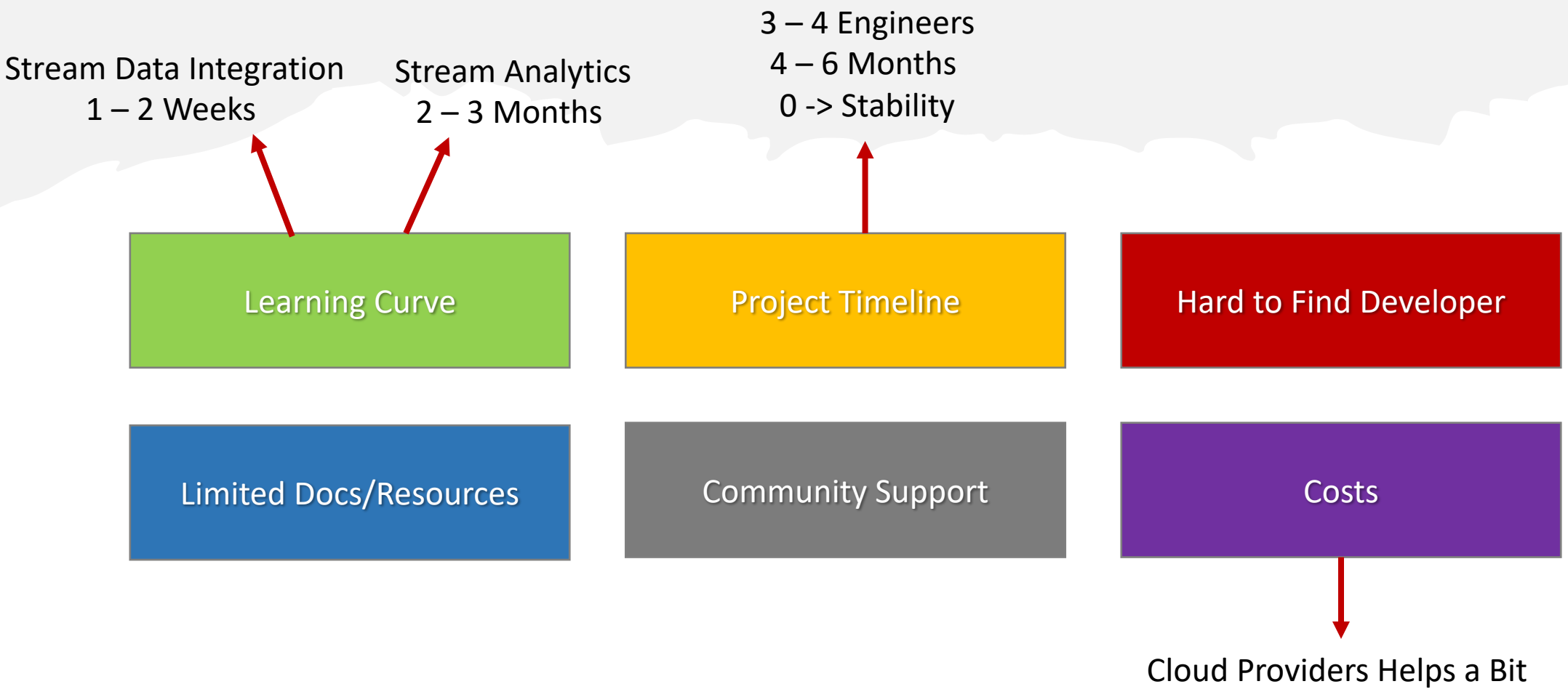
- Stateful Processing
- Pattern Matching
- Complex Joins / Aggregations

Learn More

Stream Processing – Concepts and Frameworks (Guido Schmutz, Switzerland)

https://www.youtube.com/watch?v=vFshGQ2ndeg | https://www.slideshare.net/gschmutz/introduction-to-stream-processing-132881199

Considerations



Learn More (Important)

Apache Flink Worst Practices - Konstantin Knauf - https://www.youtube.com/watch?v=F7HQd3KX2TQ

Stream Processing

When should we consider it in our solutions?

DECISION MAKING FACTORS

Requirements (FRs + NFRs + Roadmap)

Development Cost (Capex)

Maintenance Cost (Opex)

Complexity

Limitations

Industry Best Practices

When should we consider it in our solutions?

Case: Stream Data Integration

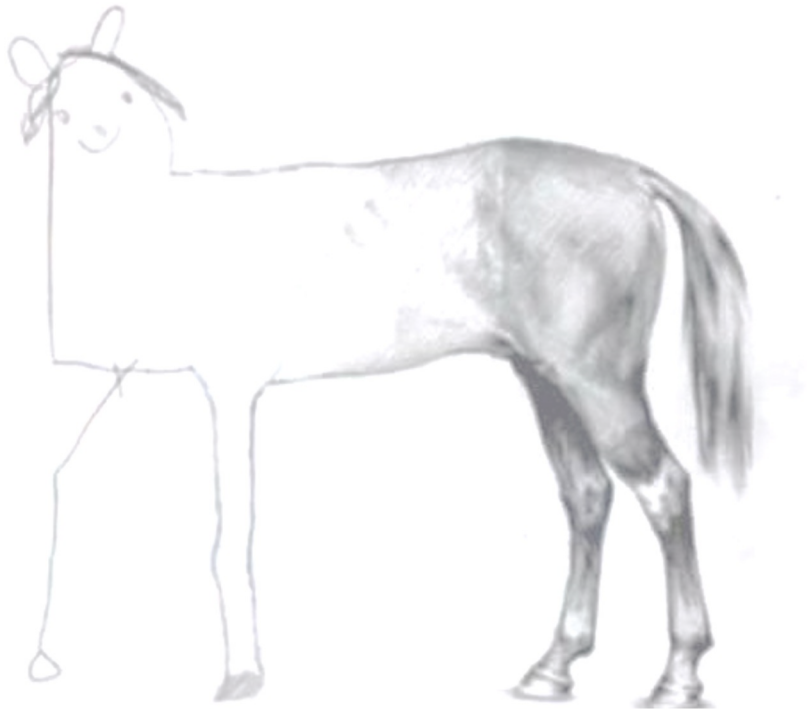
Context / Conditions

When should we consider it in our solutions?

Case: Stream Data Integration

Context / Conditions

- Events / second < 1K
- 3 – 4 Mid-Senior Developers
- Experience of Stream processing : No
- Business queries are changing frequently
- Time to market : Very tight



Note: The cases incorporated within this presentation are designed to demonstrate the reasoning process.

Learn More

Apache Flink Worst Practices - Konstantin Knauf https://www.youtube.com/watch?v=F7HQd3KX2TQ

When should we consider it in our solutions?

Case: Stream Analytics

Context / Conditions

- Events / second > 10K
- 3 – 4 Mid-Senior Developers
- Experience of Stream processing : No
- Business queries are clear and not changing frequently
- Real time/near real time insights are crucial ? Yes

Note: The cases incorporated within this presentation are designed to demonstrate the reasoning process.

Learn More

Apache Flink Worst Practices - Konstantin Knauf https://www.youtube.com/watch?v=F7HQd3KX2TQ

Quick Look On

Stream Processing Use Cases

Usecases

Video Streaming
Playback Analytics

Gaming Industry
Anti-Cheat

Telecom
Billing / Charging System

IOT
GPS Tracking

Finance
Fraud Detection

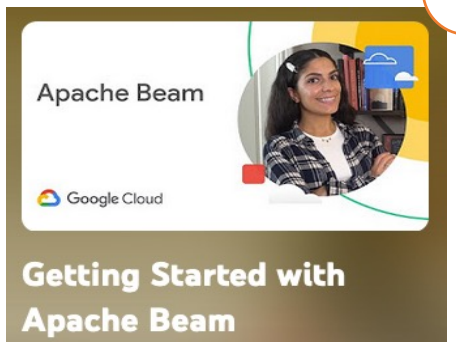
E-Commerce
User Analytics

Stream Processing

How to start learning ?

How to start learning?

1



Google Cloud Apache Beam
Debi Cabrera

2

Apache Beam Step By Step
Atul Raina

3

BEAM SUMMIT

FLINK FORWARD

BEAM SUMMIT & FLINK FORWARD

4

beam

Flink

Official Documentation

IMPORTANT NOTE

Creating a Stream Processing service isn't as straightforward as crafting CRUD APIs. Relying solely on Google, development tools, Stackoverflow, and copy-pasting won't get you far. It's crucial to dedicate ample time to thoroughly learn and understand the underlying concepts.

[1] https://youtu.be/65lmwL7rSy4
[2] https://youtube.com/playlist?list=PL8bzd7vku-WhVHzJgmXoCxx3aB4PxTQLP
[3] https://beamsummit.org/
[3] https://www.flink-forward.org/
[4] https://beam.apache.org/documentation/
[4] https://nightlies.apache.org/flink/flink-docs-stable/

One Last Thing ..

Thanks for your Attention!

Slides & Code Repository

Any Question ?

Send me a message on twitter or Linkedin

@SorooshKh

linkedin.com/in/sorooshkhodami/

[4 Geeks]

Stream Processing Use Cases For Different Industries + Case Studies

Video Platforms Use cases

- Playback Analytics
- Content Provider Shares
- Pay Per Minute
- Fraud Detection
- Personalized Recommendation

Learn More

Massive Scale Data Processing at Netflix using Flink - Snehal Nagmote & Pallavi Phadnis youtube.com/watch?v=lC0d3gAPXaI
Custom, Complex Windows at Scale using Apache Flink - Matt Zimmer (Netflix) youtube.com/watch?v=XUvqnsWm8yo
SF 2017: Monal Daxini - Stream Processing with Flink at Netflix youtube.com/watch?v=sPB8w-YXX1s
Real-time Processing with Flink for Machine Learning at Netflix - Elliot Chow youtube.com/watch?v=o4C7TDneH00

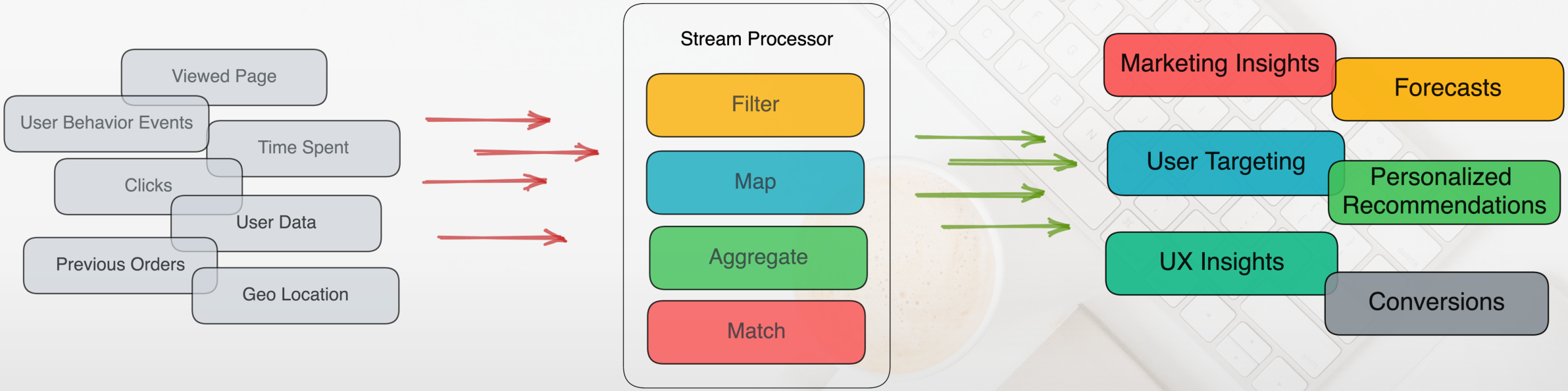
Gaming Industry Use cases

Learn More

Kafka and Big Data Streaming Use Cases in the Gaming Industry
https://www.confluent.io/online-talks/kafka-and-big-data-streaming-use-cases-in-the-gaming-industry/

Let's Play Flink – Fun with Streaming in a Gaming Company
https://www.youtube.com/watch?v=8BNKEmt47UM

Application Analytics Use cases



Learn More

Implementing Google Analytics: A Case Study - Making Sense of Stream Processing by Martin Kleppmann https://www.oreilly.com/library/view/making-sense-of/9781492042563/ch01.html

Martin Kleppmann — Event Sourcing and Stream Processing at Scale https://www.youtube.com/watch?v=avi-TZI9t2I

Singles Day 2018: Data in a Flink of an eye https://www.ververica.com/blog/singles-day-2018-data-in-a-flink-of-an-eye

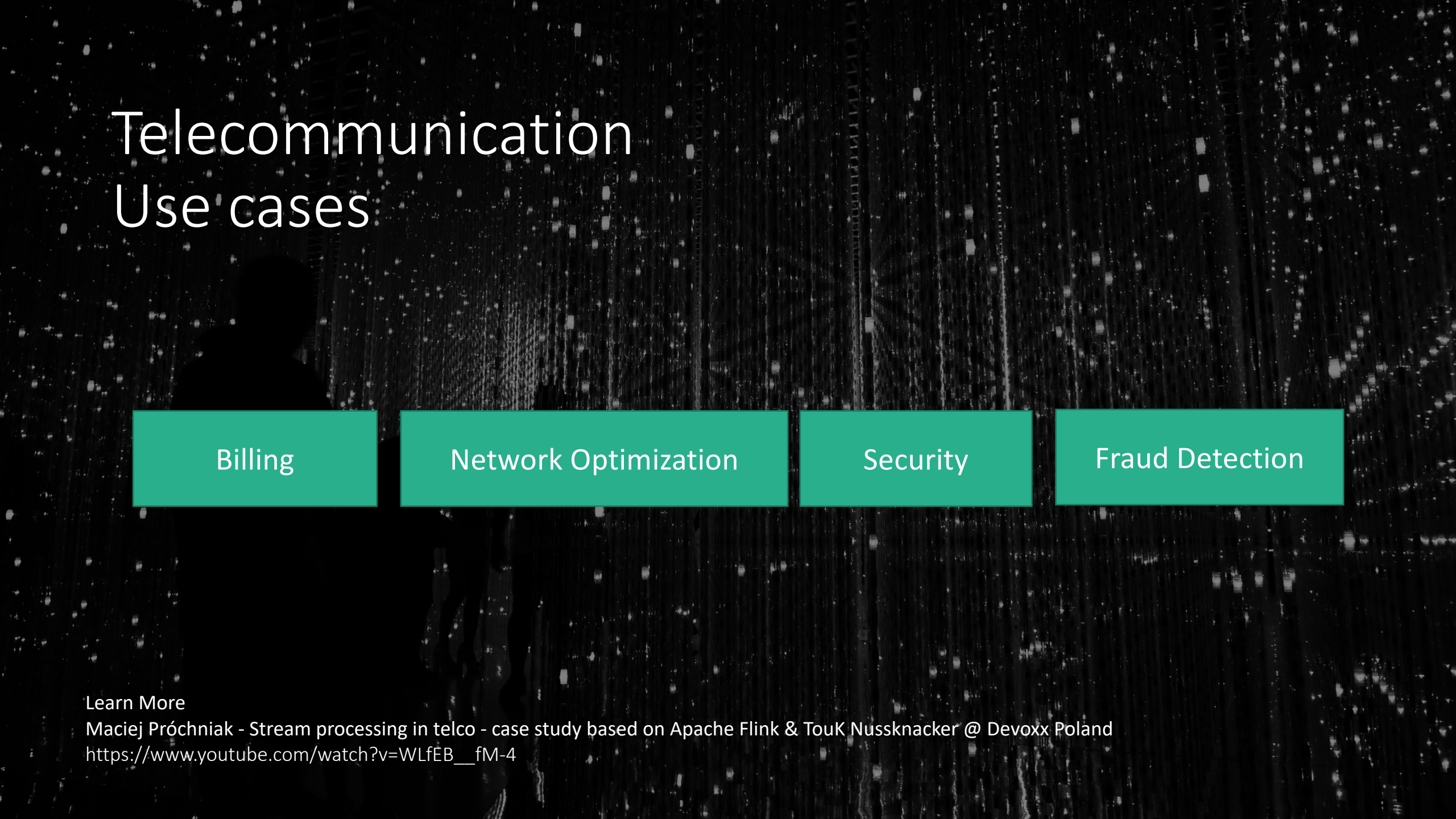
Internet Of Things Use cases

- Fleet management / GPS Tracking
- Anomaly detection
- Smart home automation
- Energy management
- Environmental monitoring
- Predictive maintenance
- Self-Driving Cars

Learn More

7 Reasons to use Apache Flink for your IoT Project
https://www.youtube.com/watch?v=Q0LBTmT4W9o

Telecommunication Use cases



Learn More
Maciej Próchniak - Stream processing in telco - case study based on Apache Flink & TouK Nussknacker @ Devoxx Poland
https://www.youtube.com/watch?v=WLfEB__fM-4

Financial Systems
Use cases

- Risk management
- Algorithmic trading
- Fraud detection
- Real-time portfolio analysis
- Customer analytics
- Profit & Lost Insights
- Regulatory compliance

Learn More

Real Time Fraud Detection with Stateful Functions https://www.youtube.com/watch?v=RxDlksbsdQ0

Fast Data at ING - Martijn Visser & Bas Geerdink (ING) https://www.youtube.com/watch?v=e-_6gijUGAw

Stream ING Models – Real time model deployment of ML Capabilities https://www.youtube.com/watch?v=Do7C4UJyWCM