Solace JCSMP
Apache Beam connector

Brought to you by Solace and Google

solace.

Matt Mays

Andrew MacKenzie

Google Cloud

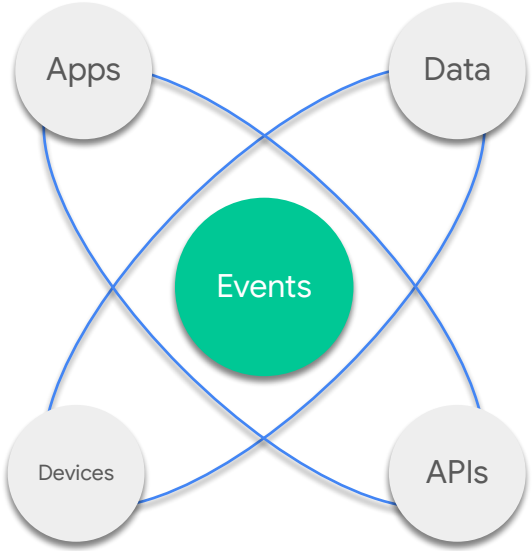
Bartosz Zabłocki

Israel Herraiz

01

What is Solace? Why this connector?

Powering your real-time, event-driven business



Event-driven has crossed the chasm

Capital Markets

Origin of Event-Driven

Faster data, faster trades, more money!

BARCLAYS

BSE EXPERIENCE THE NEW

CRÉDIT AGRICOLE

GBM GRUPO BURSÁTIL MEXICANO

KRX 한국거래소 KOREA EXCHANGE

London Stock Exchange

NSE

Standard Chartered

TMX

RBC

WeLab Bank

Telecom

airtel

Jio

Manufacturing

Unilever

HEINEKEN

SK hynix

Government

Canada

Singapore Government
Integrity · Service · Excellence

Retail

SCHWARZ

LIDL

Kaufland

EDEKA

Transportation

Federal Aviation Administration

Emirates

solace

BEAM SUMMIT

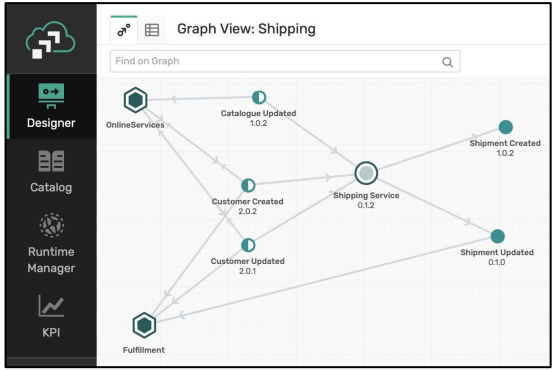
Solace Event Portal

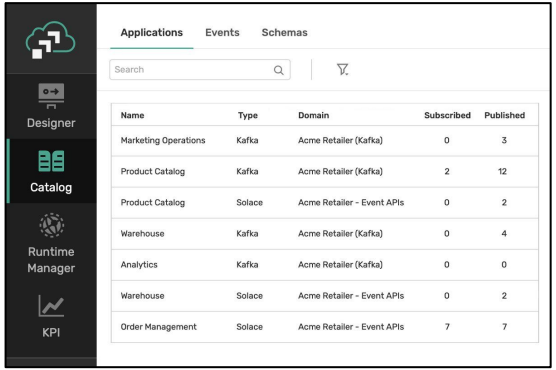
Find and understand all of your events

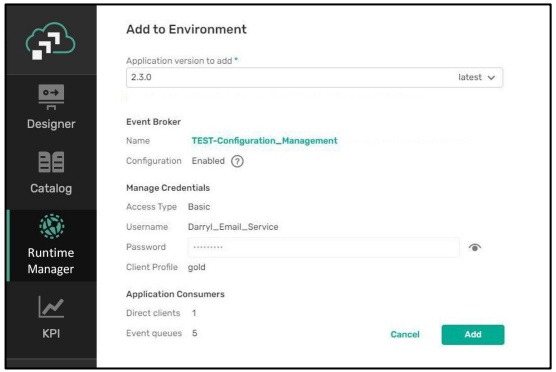
Graphically design and manage your event streams

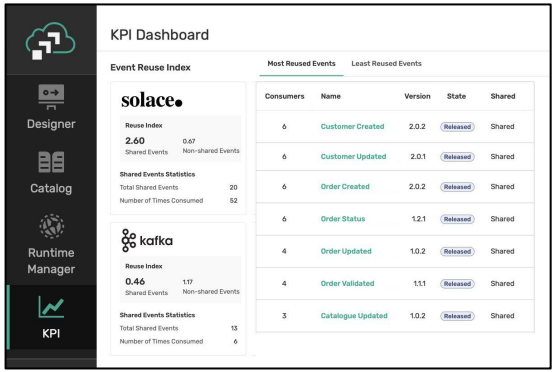
Launch your event-driven apps and push to Solace event brokers

Audit and Govern





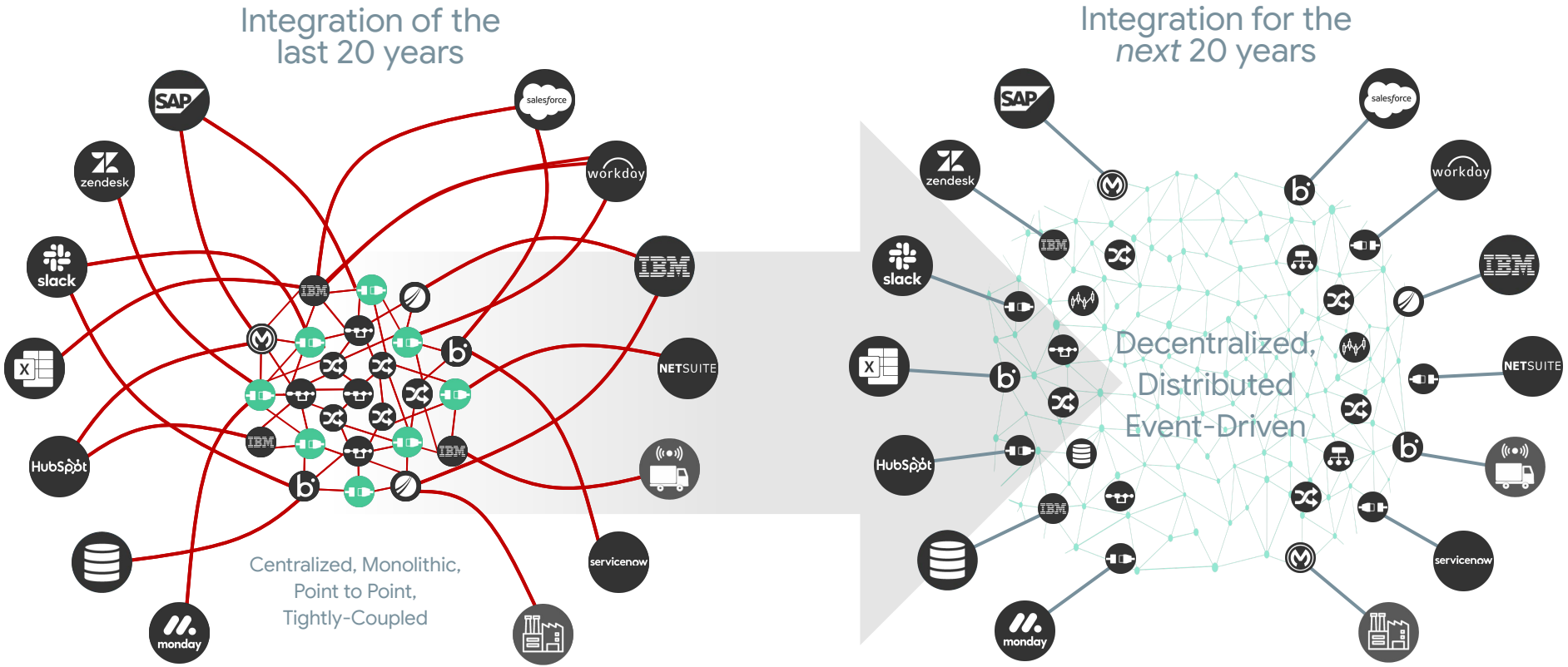




solace.

BEAM SUMMIT

Event-driven turns integration “inside out”

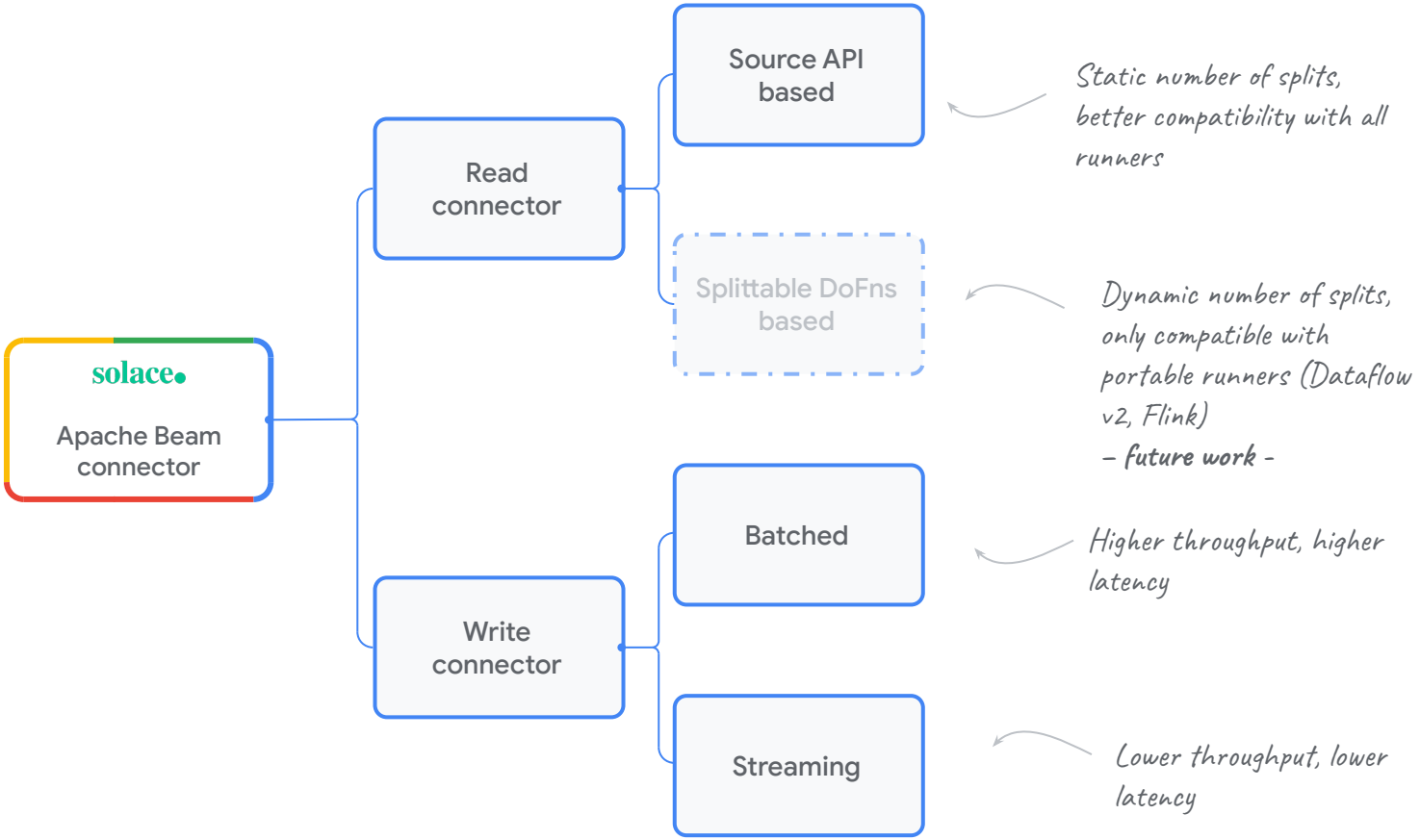


solace

BEAM SUMMIT

02

Overview of the Beam connectors


solace
Apache Beam connector
Read connector
Source API based
Splittable DoFns based
Write connector
Batched
Streaming
Static number of splits, better compatibility with all runners
Dynamic number of splits, only compatible with portable runners (Dataflow v2, Flink)
– future work –
Higher throughput, higher latency
Lower throughput, lower latency

Read connector

Scalability

Parallelize for a single topic or queue (*).

(*) Non-exclusive access queues. Exclusive queues use a single thread.

Queue created automatically for topics.

Inputs & Outputs

Inputs: none (initiator node)
Output: Record class, or custom data class

Summary of config

Optional deduplication.
Client pool size.

Requirements from Solace

SEMP API for tracking purposes, create queue
JCSMP API for receiving data

Write connector

Scalability

Shuffling right before writing.
Parallel clients in VM, number of used VMs.
No state consumption for batching.

Inputs & Outputs

Inputs: Record class
Output: Publish results, with latency data (for persistent msg)

Summary of config

Batched or streaming writers.
Higher throughput or lower latency.
Clients per worker and total workers.

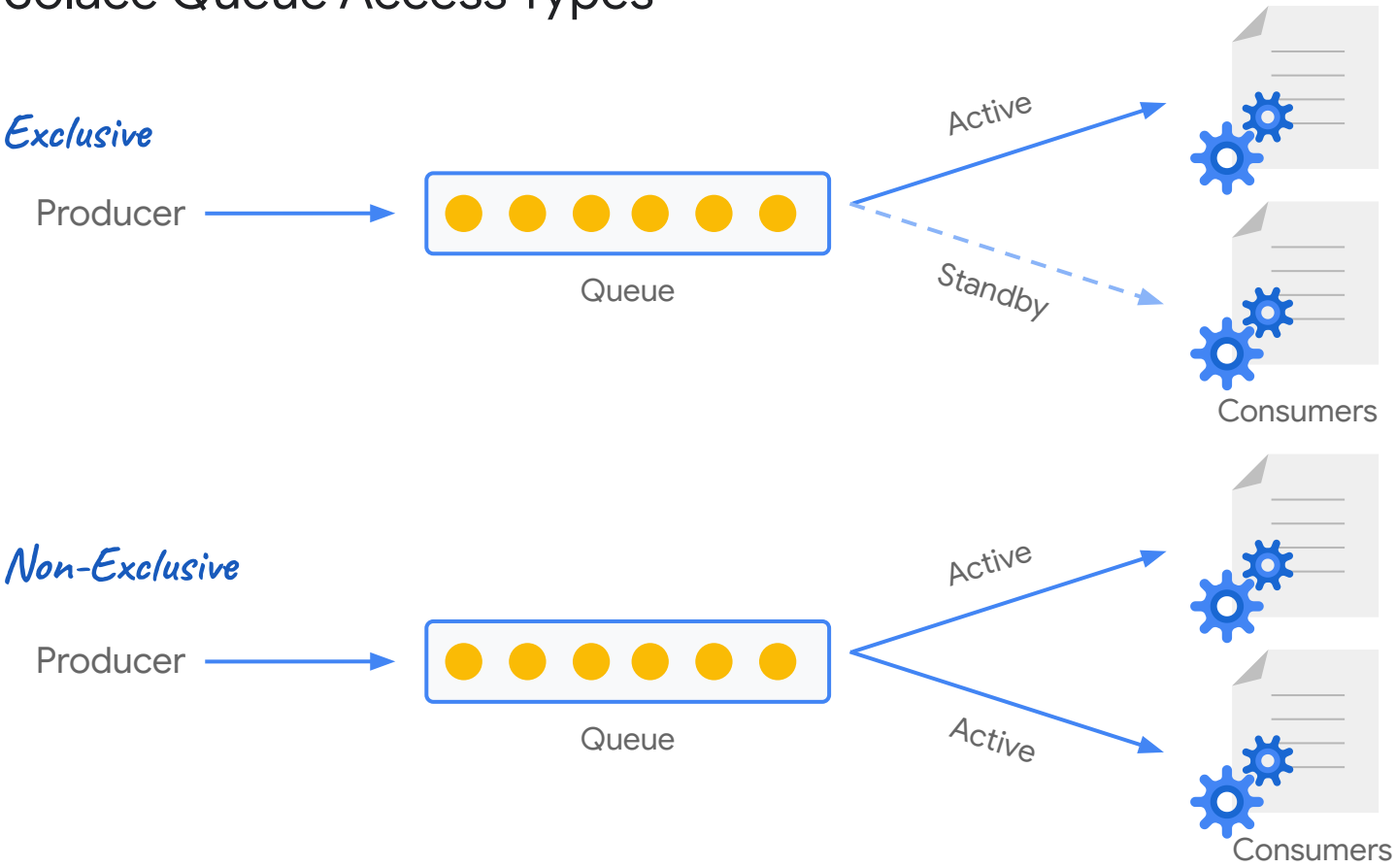
Requirements from Solace

JCSMP API
for publishing data (persistent or direct)
and receive acks (persistent msg only)

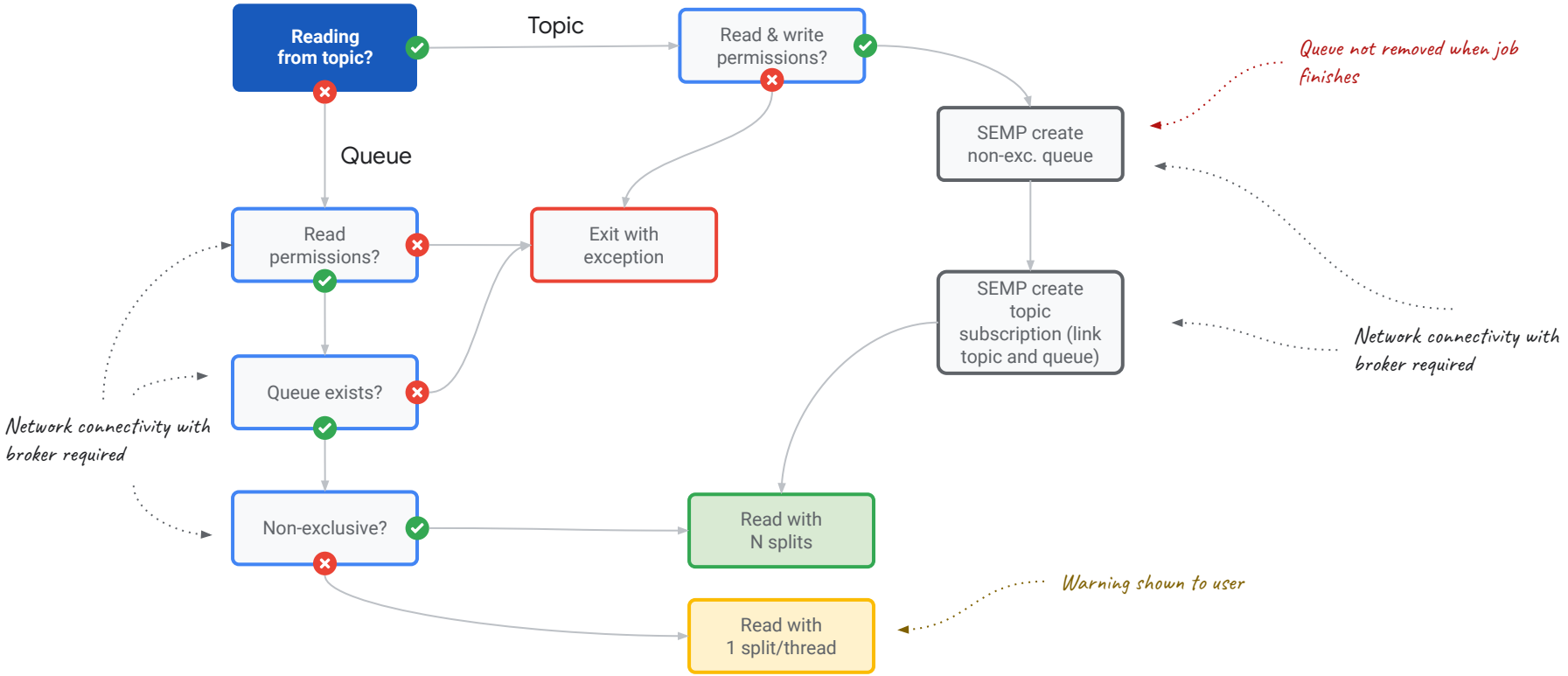
03

Read connector
Design principles

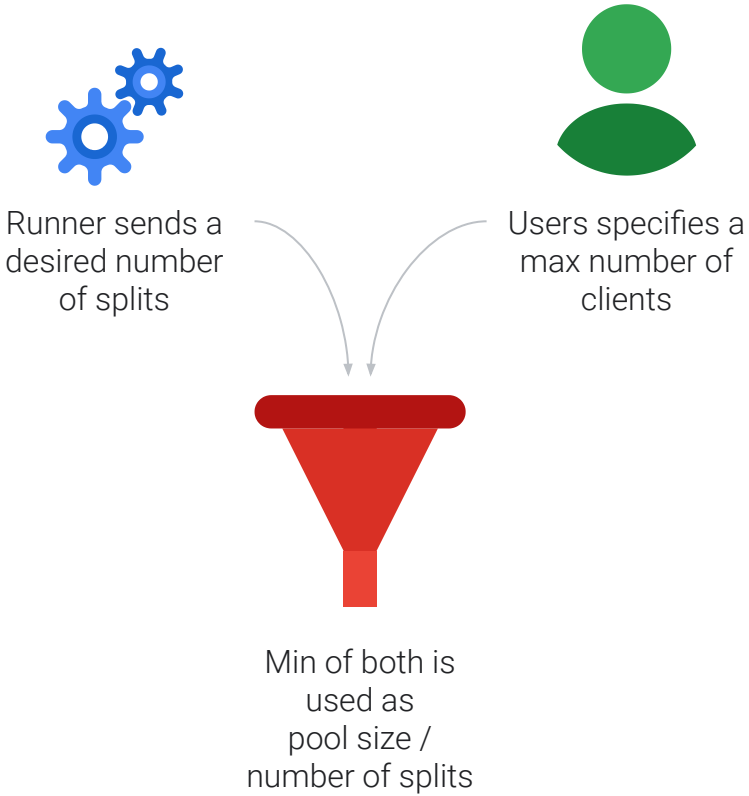
Solace Queue Access Types



Pipeline launch (driver program, creation of Solace resources)

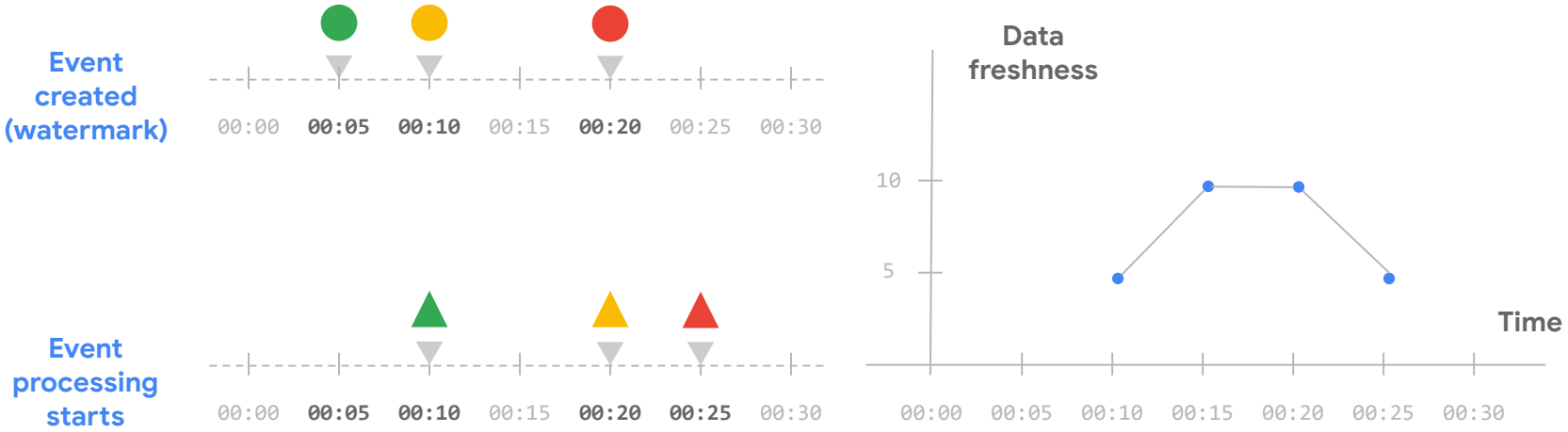


Parallelism: how is it decided?

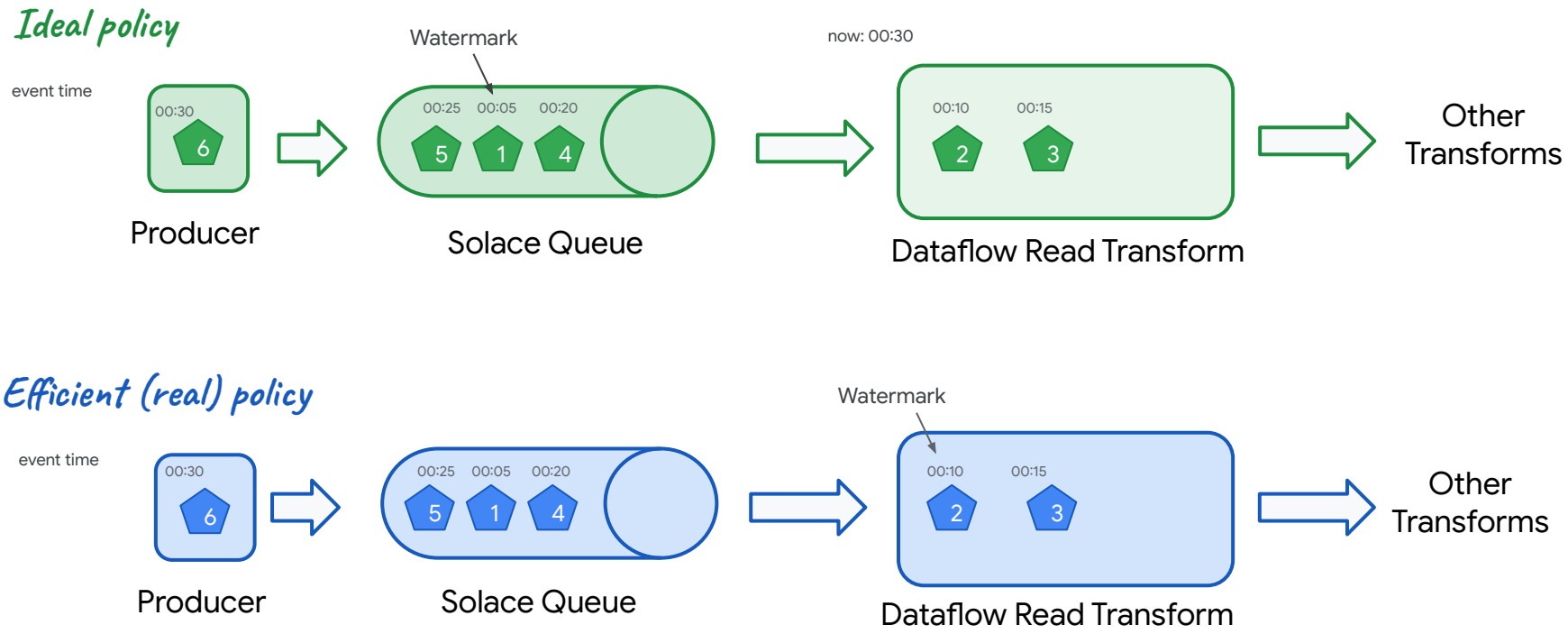


Watermark policy: data freshness

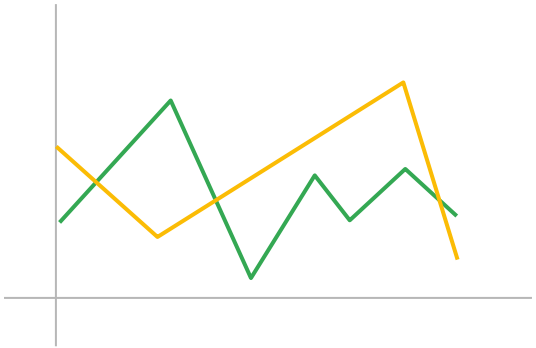
The **data freshness** measures at a point in time, the time that has elapsed between that moment and the time when the latest item fully processed by the pipeline was produced.



Watermark policy: ideal vs real situations with Solace



Backlog estimation using the SEMP API



- The SEMP API (**S**olace **E**lement **M**anagement **P**rotocol) offers backlog metrics
- Endpoint:
  - `/SEMP/v2/monitor/help/#/queue/getMsgVpnQueue`
- Property:
  - `msgSpoolUsage`
- The calls to this API are controlled by the runner.
  - The rates of calls to the SEMP API cannot be controlled.
  - With a streaming engine job, we have checked that this API gets called **once every 3-5 seconds**.
  - This rate is not correlated to the number of workers.

04

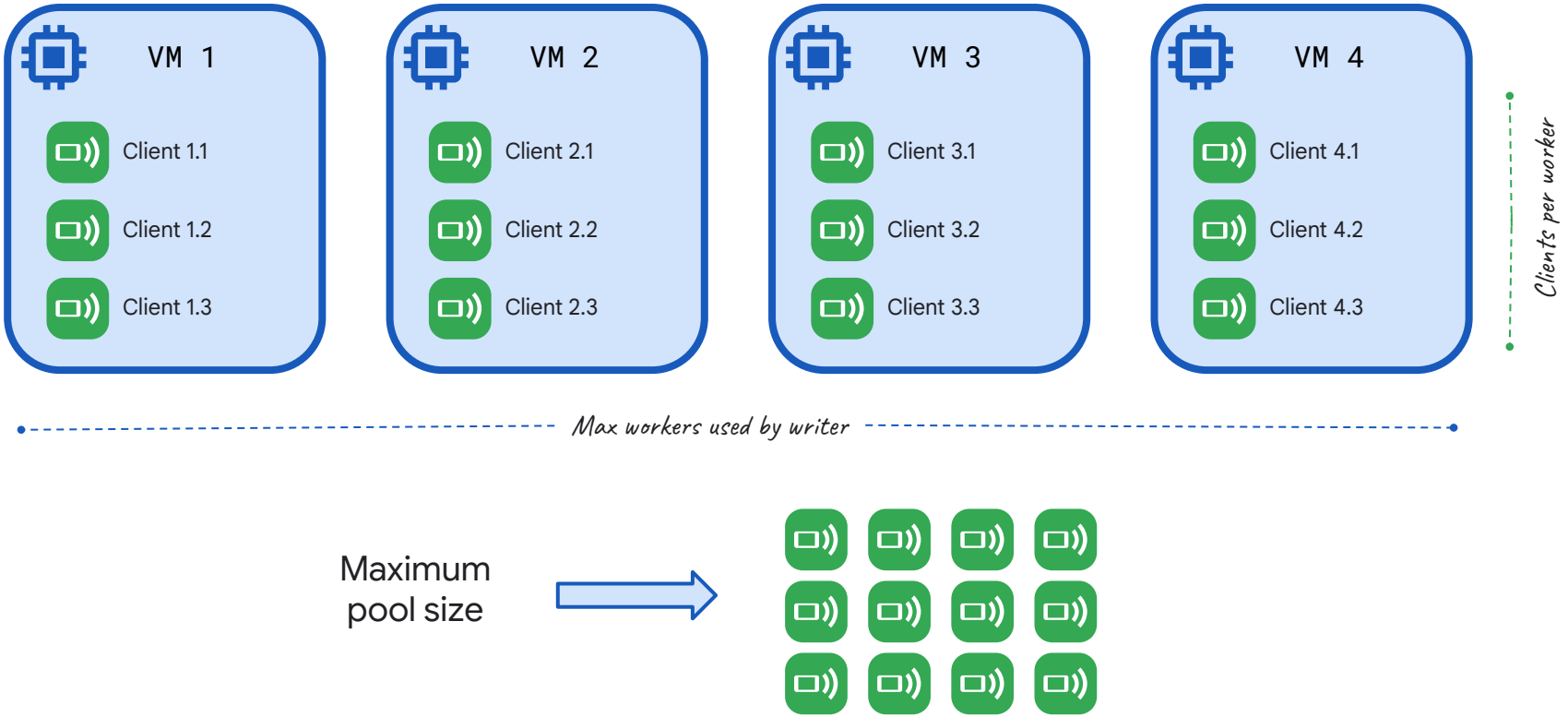
Write connector

Design principles

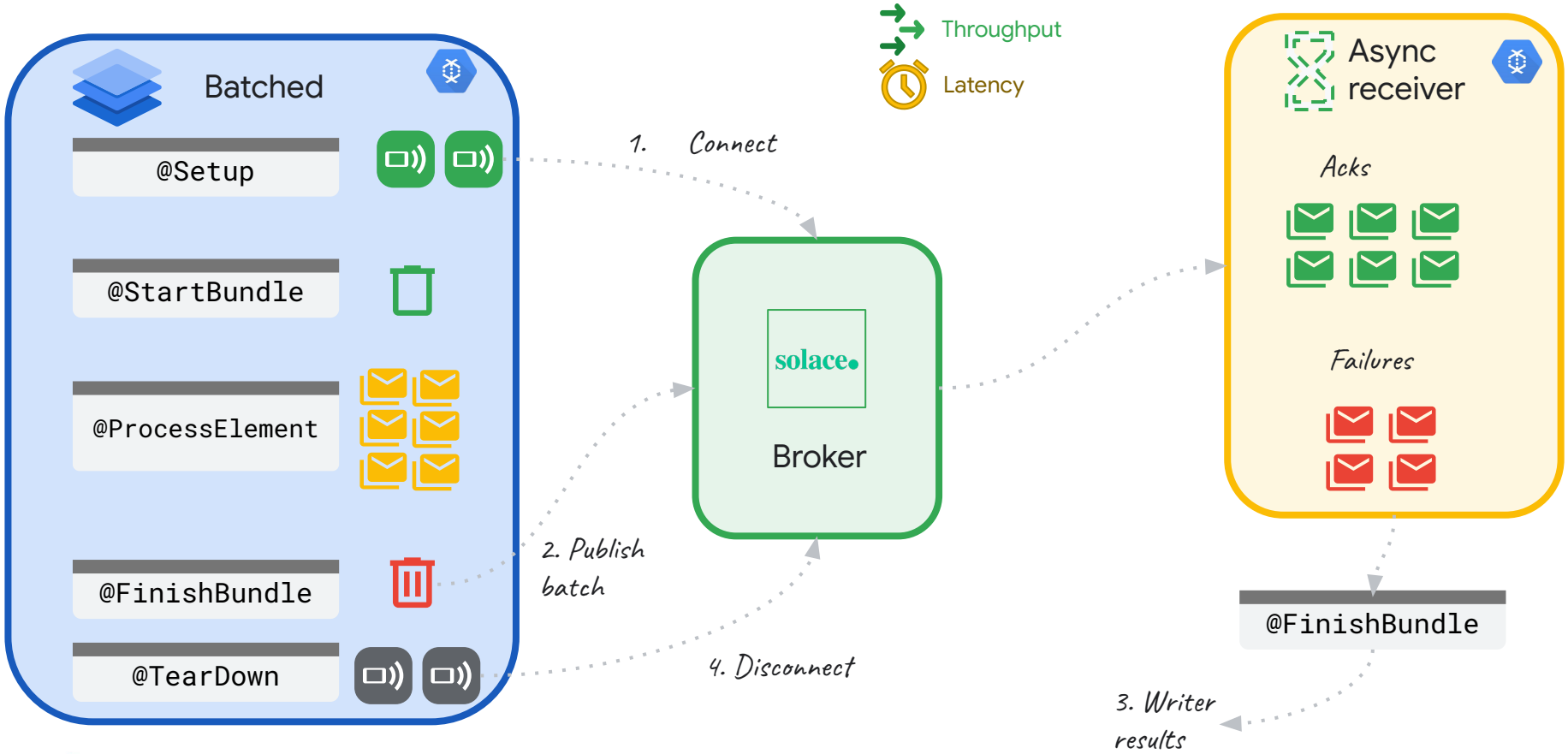
Pipeline launch (driver program)

- No resources are created by the Write connector in the driver program
  - Much simpler workflow. No interaction with Solace from the driver program
- There is no need for network connectivity when using the Write connector
- But bear in mind that:
  - If you are writing to a queue, it needs to exist prior to the job start
  - If it does not exist, you will have runtime errors in the job
  - All the runtime errors are recoverable if the queue gets eventually created
    - No need to stop the job if you forget to create the destination queue

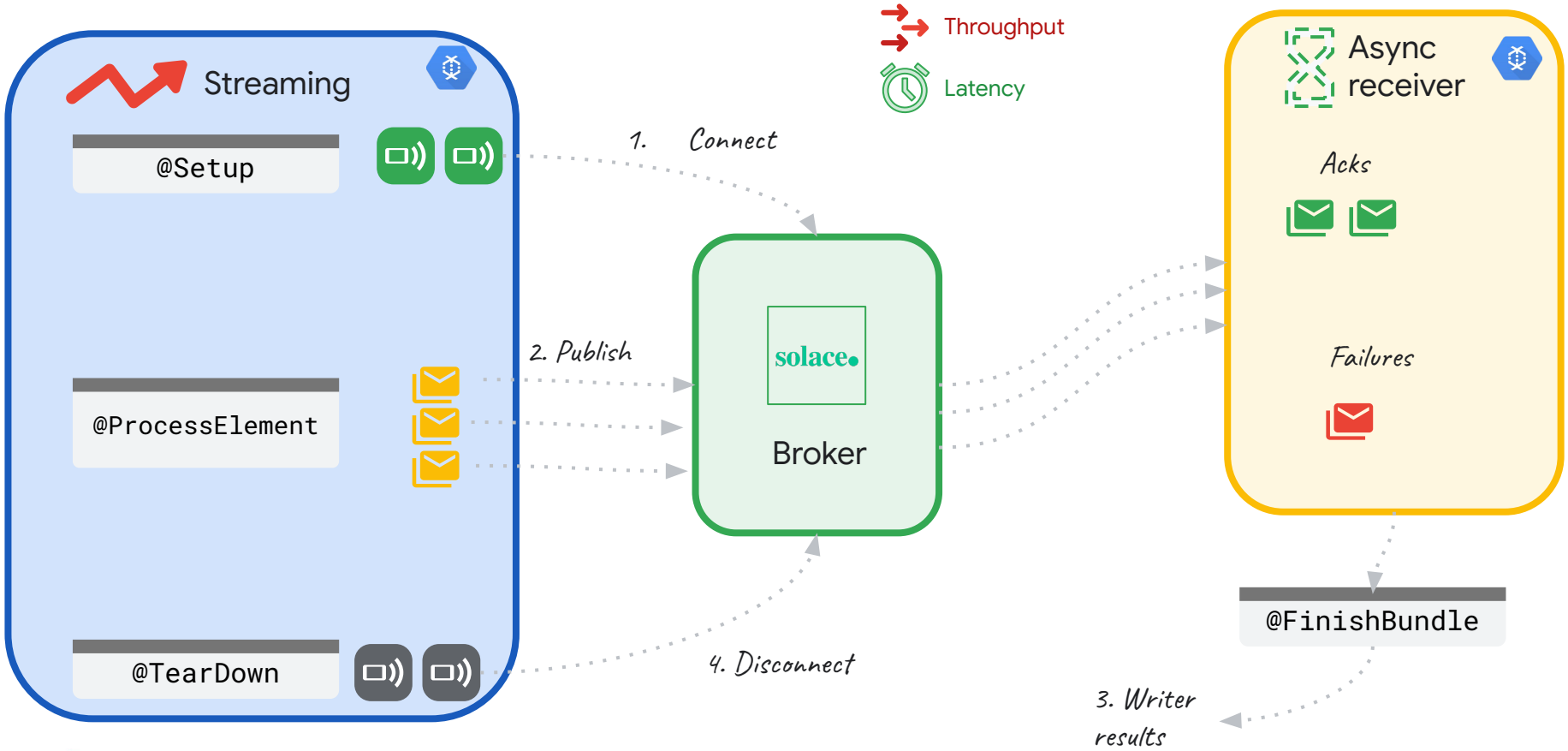
Parallelism and pool size: two parameters



Types of writers: batched



Types of writers: streaming



Design details of the connectors

Read connector

Design doc and all pull requests:

github.com/apache/beam/issues/31440

Available since Beam 2.58

Write connector

Design doc and all pull requests:

github.com/apache/beam/issues/31905

Last pull request still under review:

github.com/apache/beam/pull/32060

To appear in Beam 2.60 or 2.61

Tweaking the Solace session: dispatch mode

Solace session property	High Throughput Mode	Low Latency Mode
Pub Ack Window Size	255	1
Message callback on reactor	False	True

06

Conclusions

Using Solace with Beam is now a very smooth experience

- Read connector available since Beam 2.58
- Write connector to appear in Beam 2.60
  - Reviewers permitting :)
- The SolaceIO connector offers deep integration between Beam and Solace, for all runners
  - Accurate and lively estimation of backlog metrics
    - Better autoscaling for runners that support it
  - Accurate estimation of the watermark based on the Solace message timestamps
  - Efficient usage of Solace resources
    - Client multiplexing in multi-threaded runners

07

Backup Slides

Thank you!

solace.

Matt Mays

Andrew MacKenzie

Google Cloud

Bartosz Zabłocki

Israel Herraiz