DDS + Apache Flink

Real-Time Stream Processing Redefined!

Angelo Corsaro, PhD

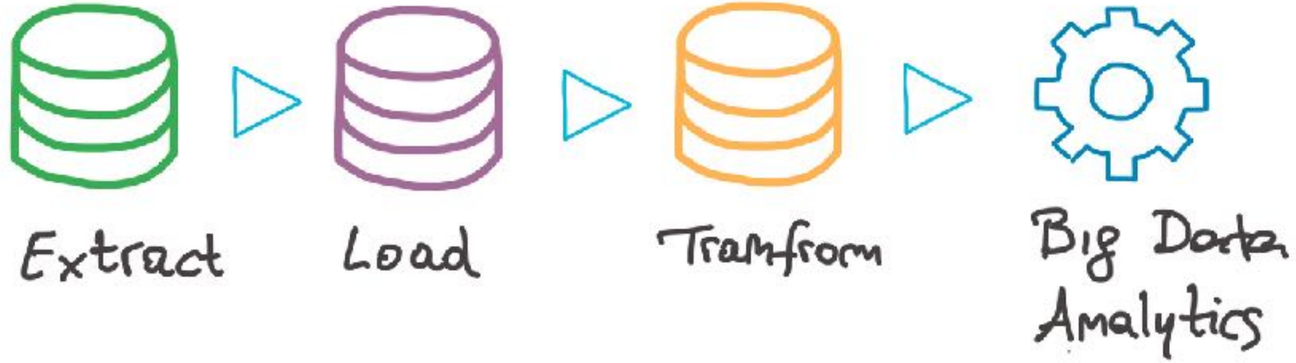
CTO, ADLINK Tech. Inc.
Co-Chair, OMG DDS-SIG
angelo.corsaro@adlinktech.com

PRISMTECH™
AN ADLINK COMPANY

Data Analytics Converge

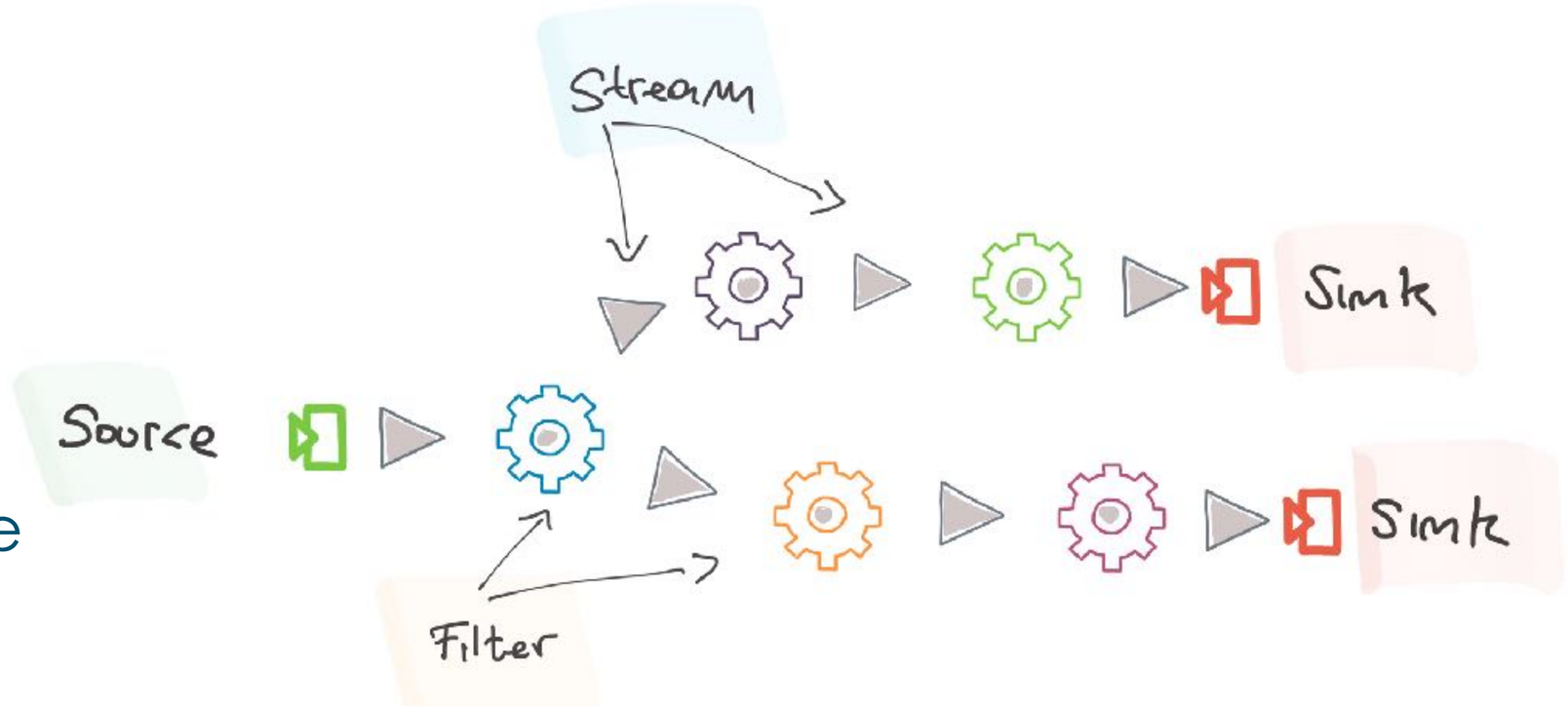
Data Analytics in the IT World

Data Analytics in the IT and Internet world has traditionally been the synonym of **ETL** and batch processing



Data Analytics in the OT World

Data Analytics in the OT world has traditionally about real-time stream processing where stages of **filters** (analytics) operate on real-time **streams** of data



Convergence

The IT and OT requirements for analytics are now converging toward the need of operating on real-time data as well as on data at rest

The operational needs driving this convergence is unveiling limitations on technologies traditionally used for Big Data Analytics and creating the opportunity for innovation and cross contamination

Apache Flink

Apache Flink is the first Open Source platform that **unifies real-time** and **batch analytics**

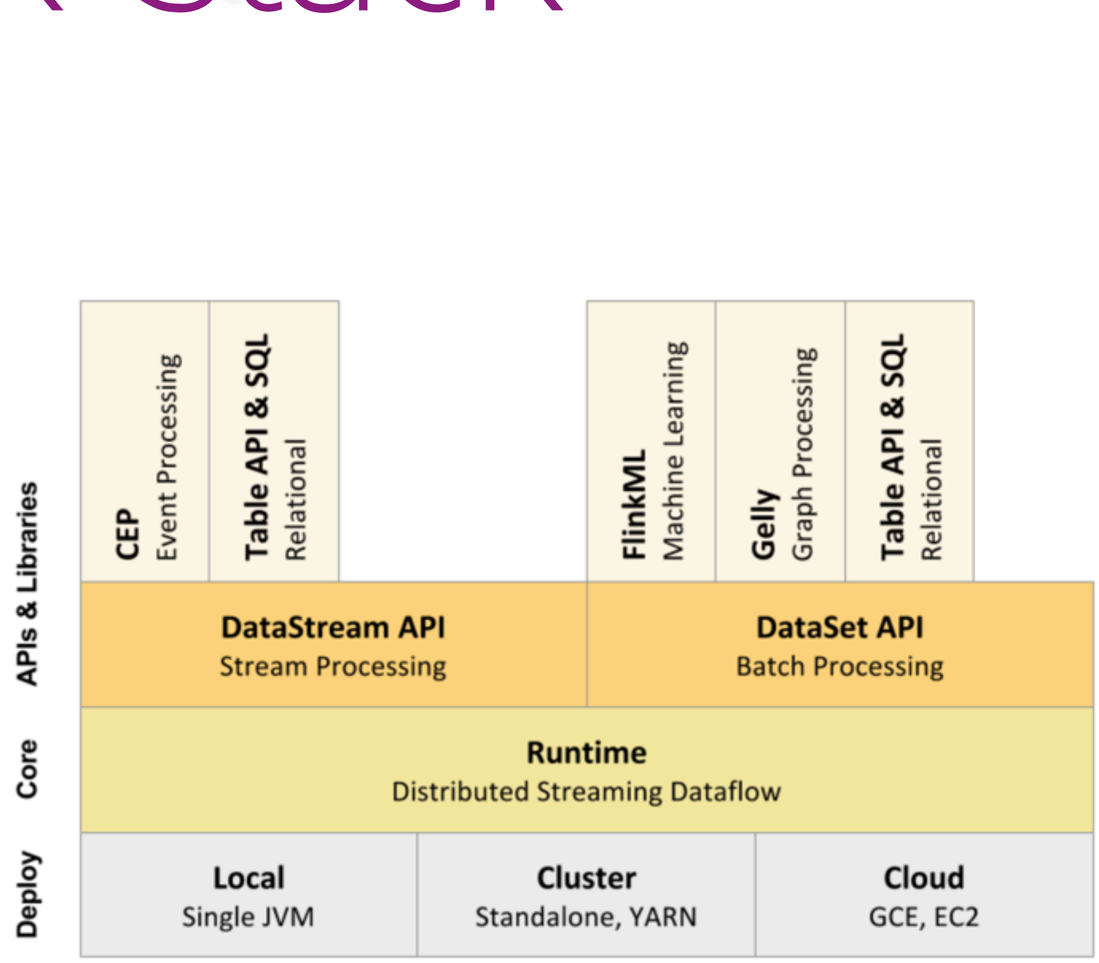
Yet most of the technologies used to ingest data in Apache Flink were designed with IT/Internet applications in mind, not necessarily high performance distributed analytics

Apache Flink Stack

Apache Flink provides API for Data Streaming and Batch processing starting from a single runtime distributed streaming data flow

The processing engine is fault-tolerant and can guarantee exactly once semantics

Exactly once semantics leverages checkpoints and the ability to resume consuming data from a given point in the past (*rewind*)



Flink Connectors: Kafka

Kafka is probably one of the most commonly used connector for Flink's **sources** and **sinks**

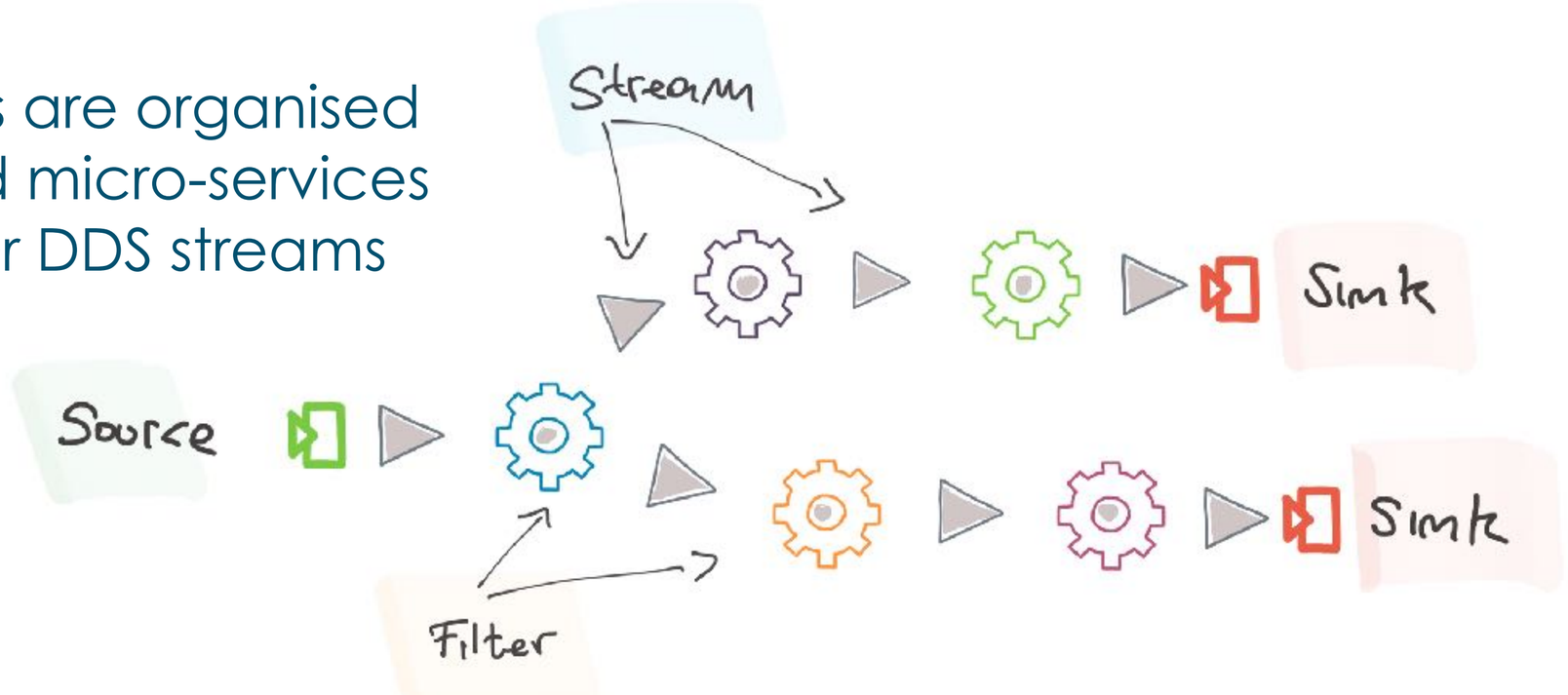
The reminder of this talk will motivate why **DDS** can be a very interesting alternative for Kafka when looking for better **scalability**, **performance** and **predictability**

DDS
Stream Processing

Real-Time Stream Processing

DDS is used today extensively in Military and Aerospace, Medical, IIoT, etc. as the data sharing infrastructure to build high-performance streaming applications

These applications are organised as loosely coupled micro-services that compute over DDS streams



DDS as Flink Connector

DDS Streams for Flink

DDS features, performance and scalability characteristics align very well with Flink needs

DDS was designed for real-time stream processing and has been used very successfully in some of the most challenging operational environments

DDS vs. Others

Table 1

Connector	Real Time	Exactly once	Durability	Storage Capacity	Notes
HDFS	No	Yes	Yes	Years	
Kafka	Yes	Source only	Yes* (Flushed but not synced)	Days	*Writes are replicated but may not persisted to durable media. (flush.messages=1 bounds this but is not recommended)*
RabbitMQ	Yes	Source only	Yes* (slowly)	Days	*Durability can be added with a performance hit*
Cassandra	No	Yes* (If updates are idempotent)	Yes	Years	*App developers need to write custom logic to handle duplicate writes.*
Sockets	Yes	No	No	None	
DDS	Yes	Yes	Yes	Depends on Back-End (e.g. File-system, RDBMS, HDFS, etc.)	*High-Performance distributed durability* *Built-in support for circuit-breakers as well as batching* *Content awareness (filters and queries natively supported)* *Dynamic Discovery facilitates deployment and shields from topology changes* *Support for IP multicast give unparalleled scalability*

*Expanded Comparison Table available at http://bit.ly/2q4ctYG to include **DDS***

DDS inside Flink

DDSizing Flink

Our initial analysis of Flink runtime reveals that it could take advantage of DDS it is core too

One obvious example is using DDS as a State Back-end
(see http://bit.ly/2p1Vx3y). This would provide Flink with a very high performance high availability storage for storing its snapshots.

Additionally, Flink implements in its runtime space and time dependent batching. This feature is built-in in DDS

Overall the use of DDS within the Flink core could reduce the code, simplify the deployment and improve scalability and performance, especially in large scale systems and when IP-Multicast is available (DDS implements a very reliable multicast over IP-Multicast)