

Streaming Data Engineering

A Tale of Three Streams

Emanuele Della Valle

prof @ Politecnico di Milano

founder @ Quantia Consulting

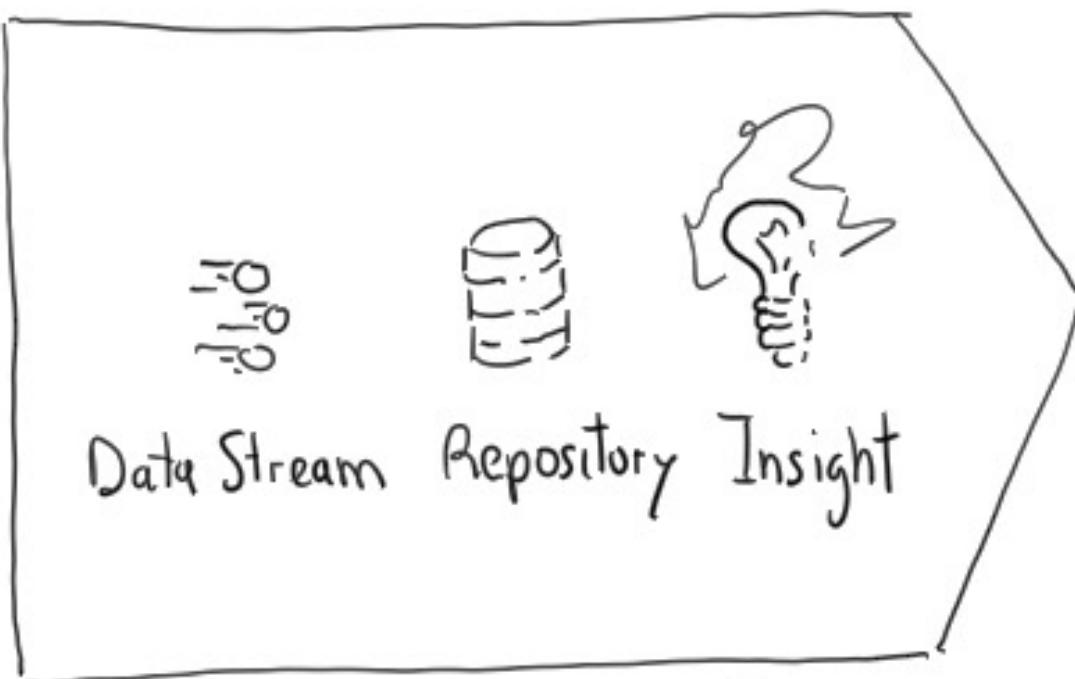
founder & CRO @ motus ml

1st premises: continuity matters



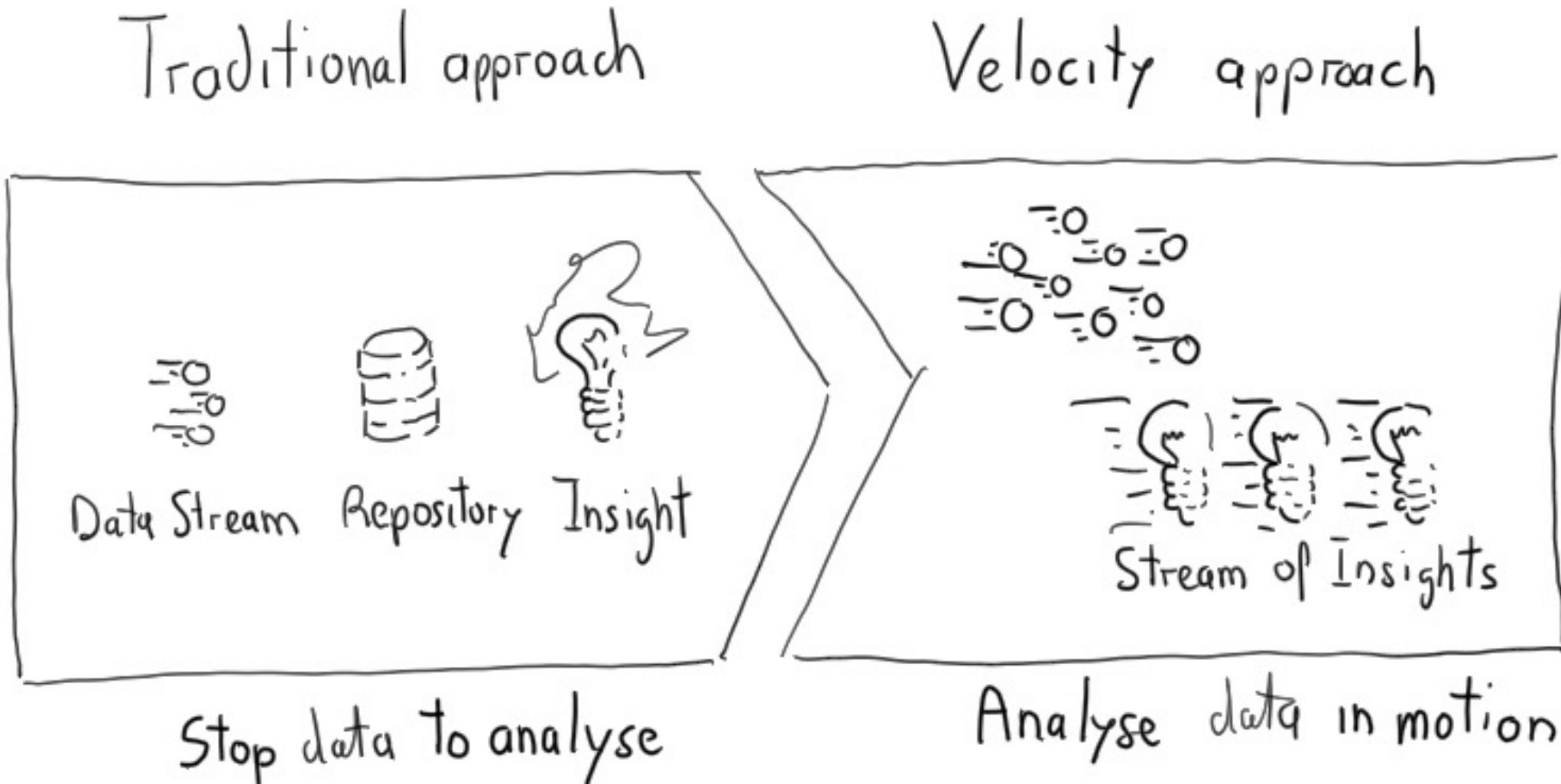
1st premises: continuity matters

Traditional approach



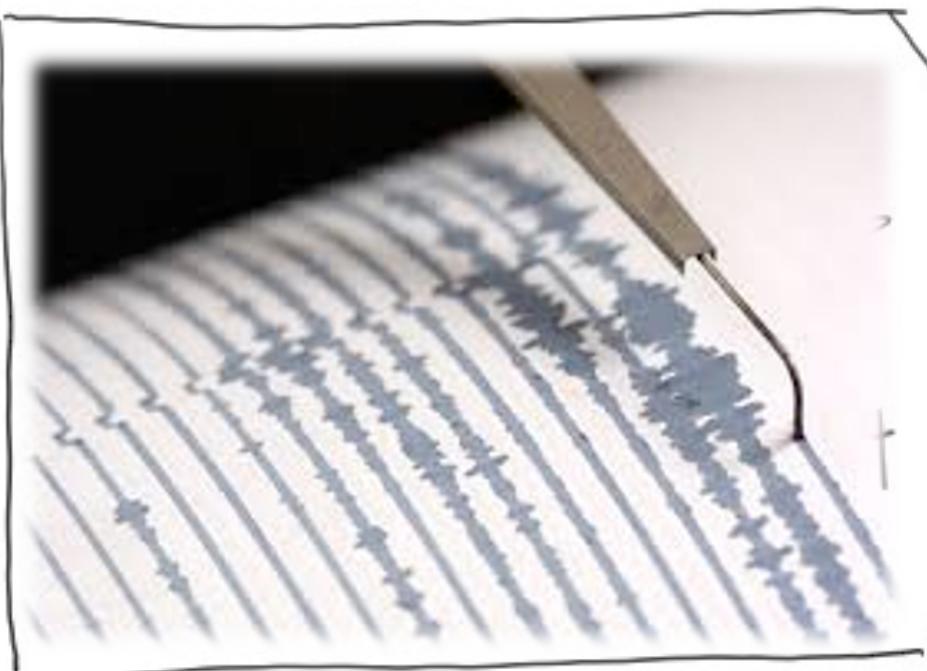
Stop data to analyse

1st premises: continuity matters



1st premises: continuity matters

Traditional approach



Stop data to analyse

Velocity approach

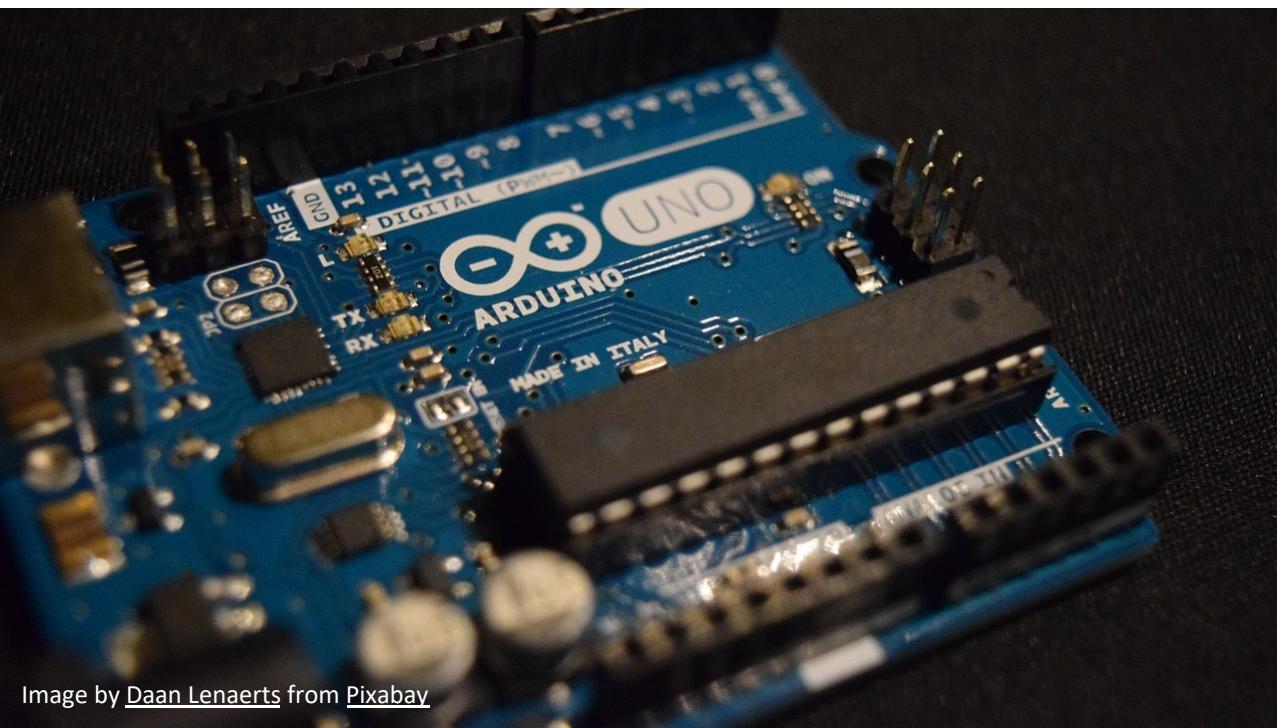


Analyse data in motion

**Stream: a useful metaphor to
explore the solution space**

Streams take many forms ...

**Myriads of tiny flows
that you can collect**



**e.g., from physical or
software sensors**



e.g., social media,
telcos' and utilities'
monitoring

@manudellavalle - <http://emanueledellavalle.org>

**Continuous massive flows
than you cannot stop**

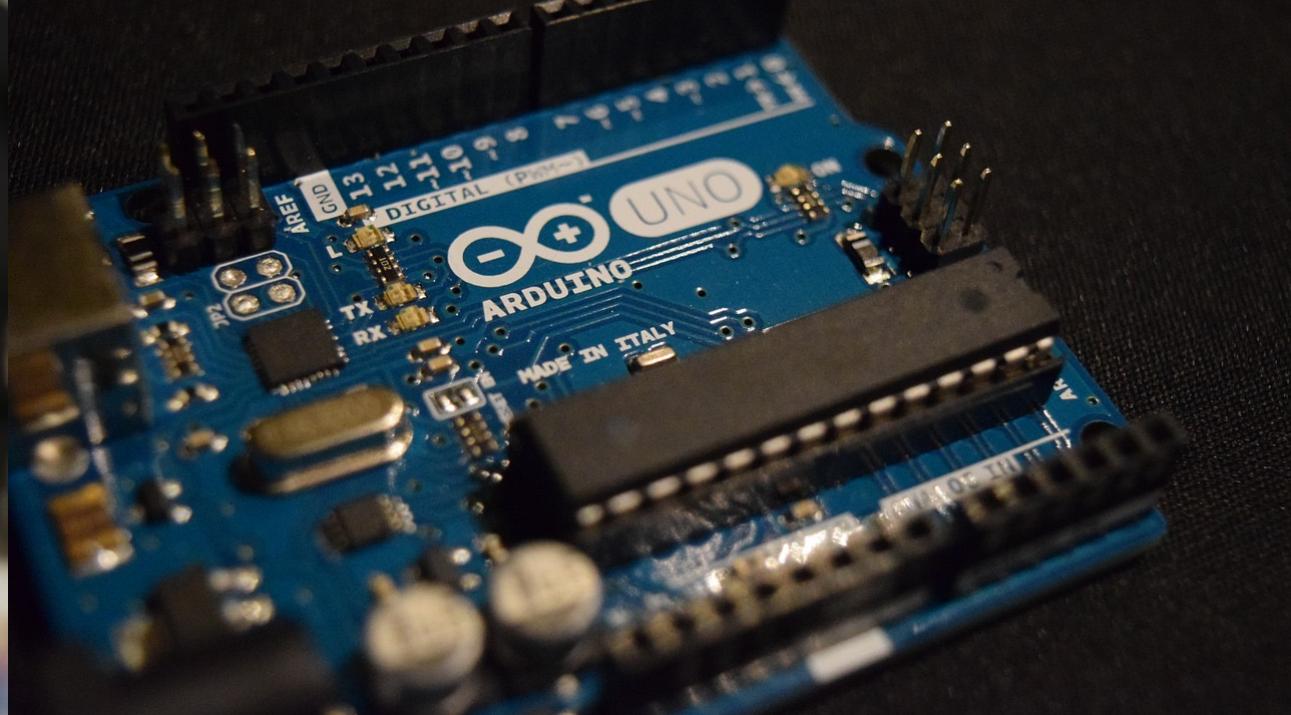
Image by Bruno /Germany from [Pixabay](#)

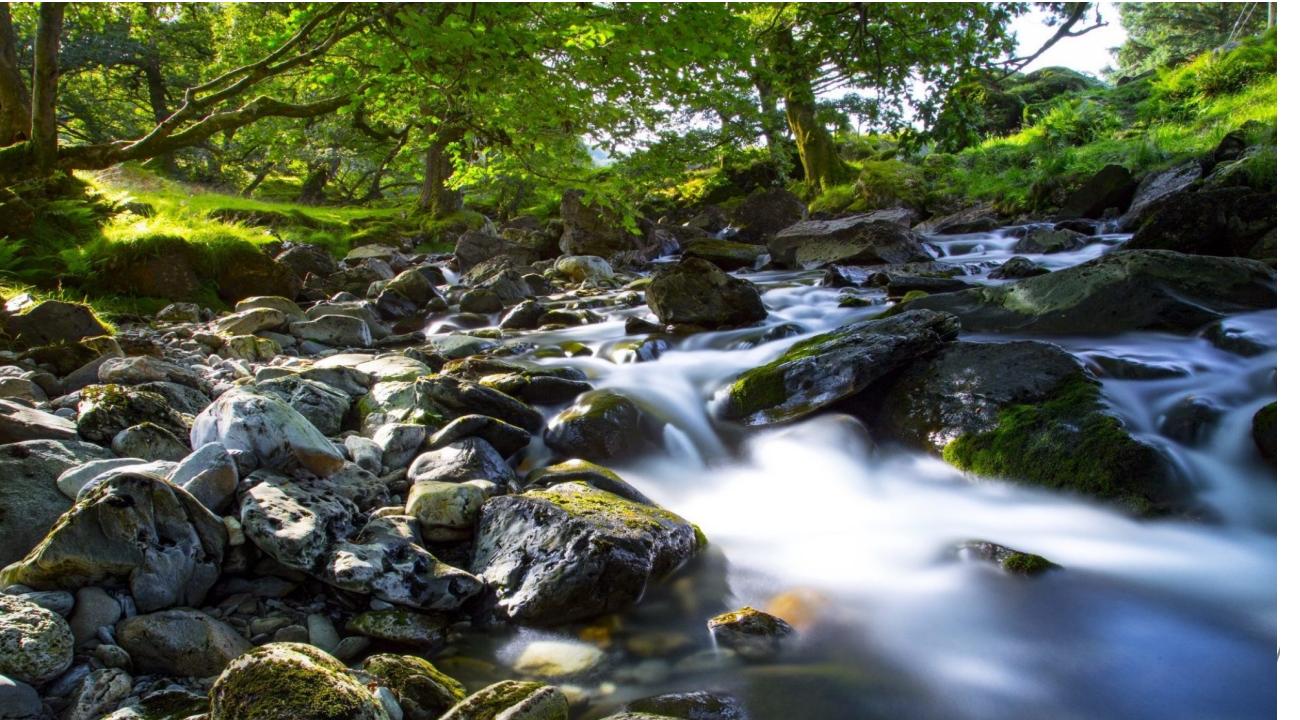


e.g., physical or cyber
alarms

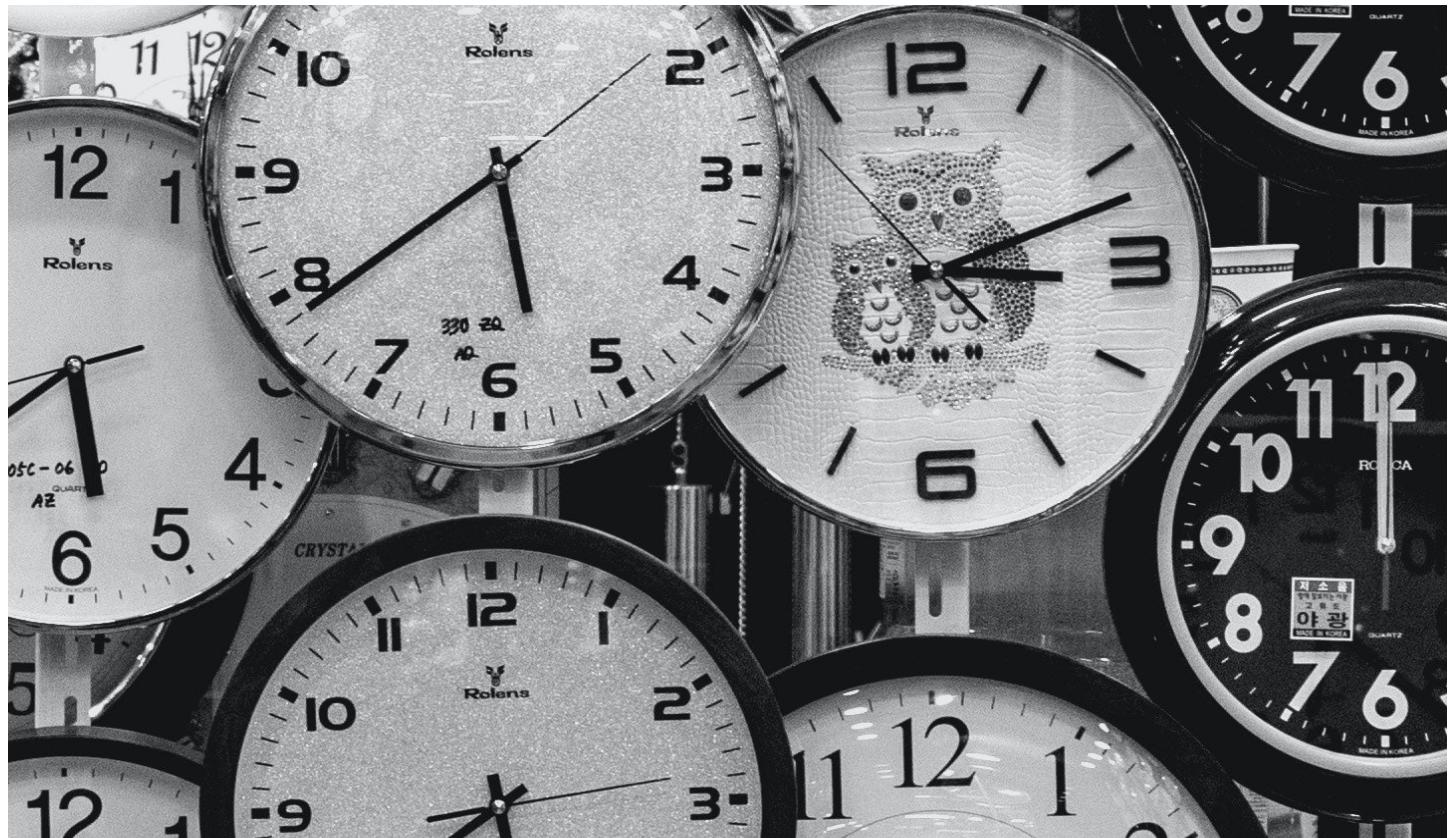


**Continuous numerous flows
that can turn into a torrent**





2nd premises: time (models) matters!



Stream-only Time Model

too little



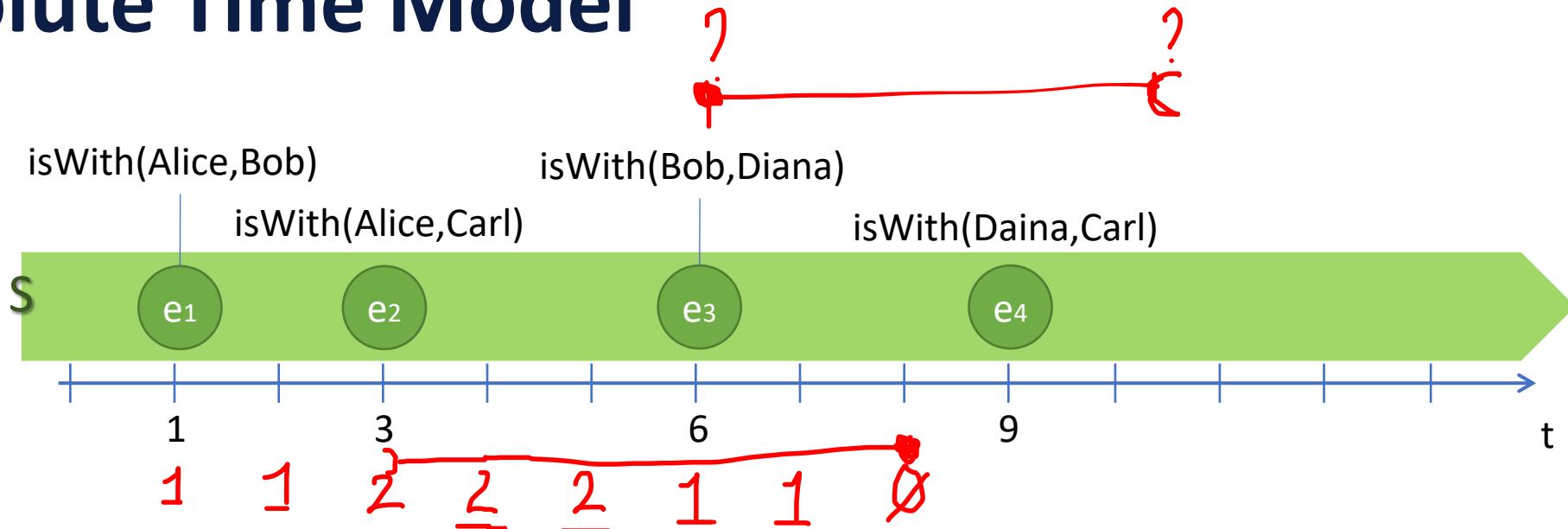
- The order can be exploited to perform queries

- Does Alice meet Bob **before** Carl?
- Who does Carl meet **first**?

forget when ans.

cnt (Alice) | no forget
last

Absolute Time Model

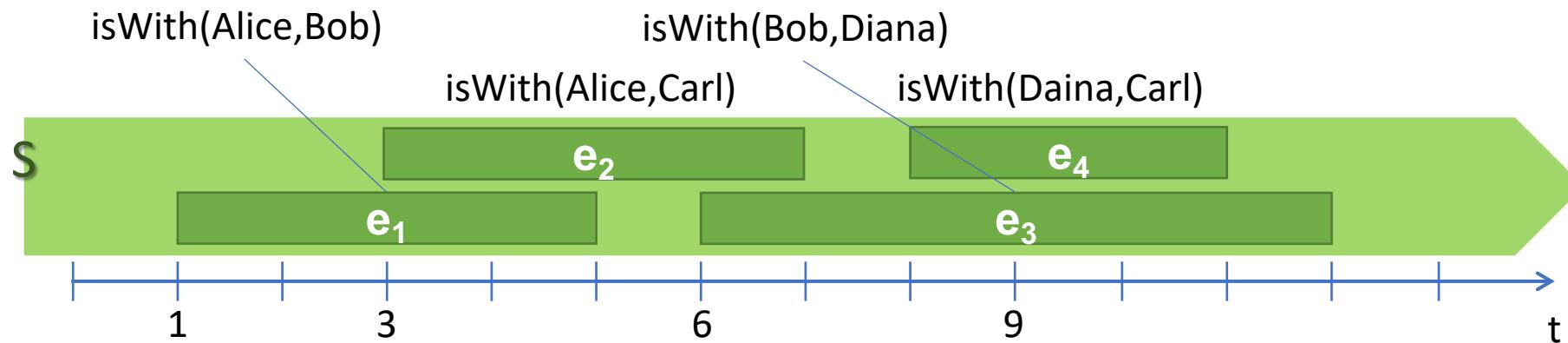


- We can ask the queries in the previous slide
- We can start to compose queries taking into account the time
 - How many people has Alice met in the last 5m? *window*
 - Does Diana meet Bob and then Carl within 5m? *I forgot after ans.*

every | no Forget
↑
every | no Forget

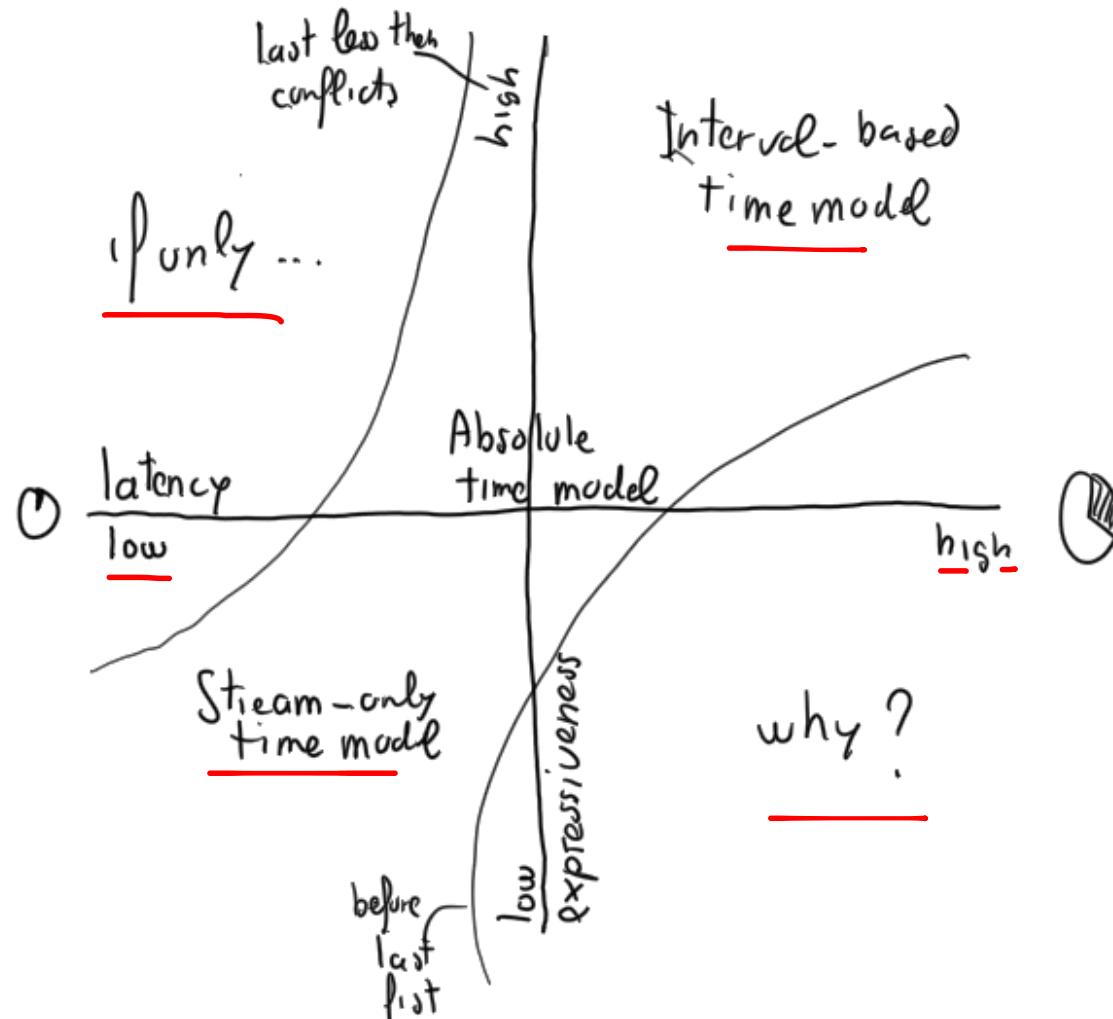
Interval-base Time Model

too much



- We can ask the queries in the previous two slides
- It is possible to write even more complex queries:
 - Which are the meetings the **last less than 5m**?
 - Which are the meetings with **conflicts**?

A first trade-off: latency vs. expressiveness



Can we tame data velocity?

A long-exposure photograph of a waterfall cascading down a mossy rock face into a pool of water. The water appears as a bright, blurred stream against the dark, textured rock and lush green foliage. The scene is peaceful and natural.

Taming
*Myriads of tiny flows
that you can collect*
with
Event-based Systems

On Events and decoupling

Events

an immutable and append-only stream of “business facts”

Decoupling

means decentralizing the freedom to act, adapt, and change

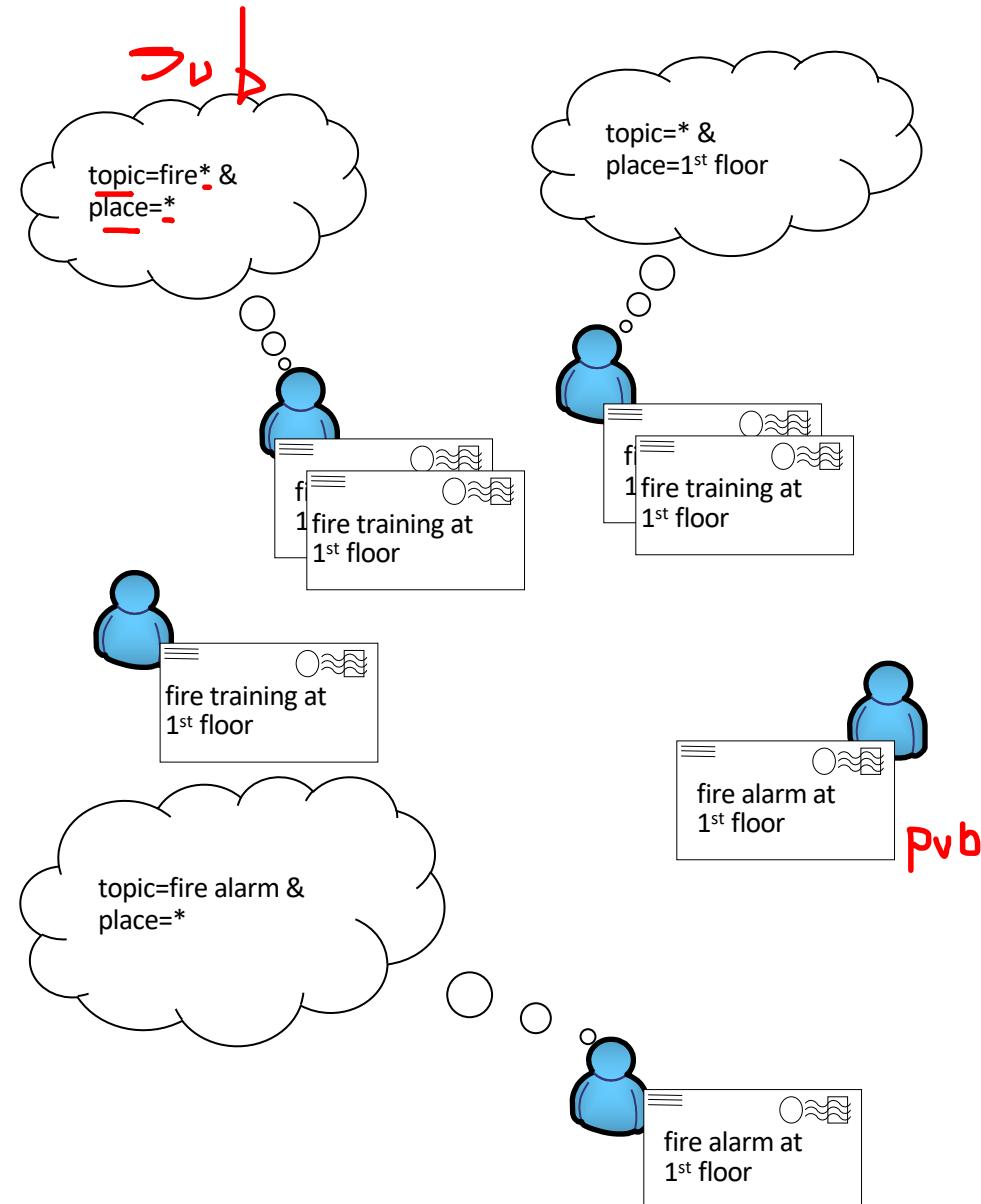
Event-based systems

- Events as “first-class” entities
- Services emit and consume events asynchronously
- Better decoupling with respect to APIs (no temporal coupling)

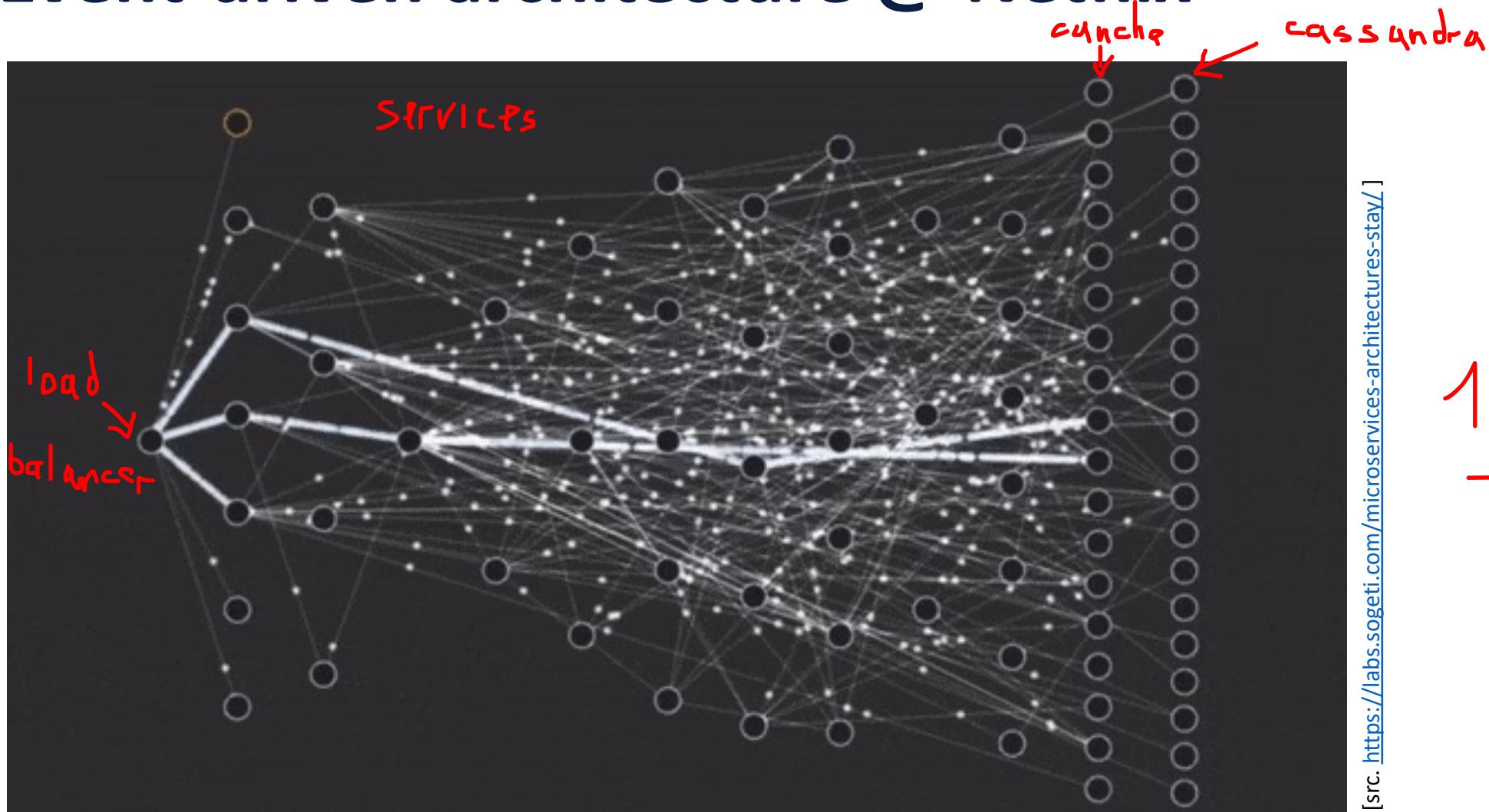
Eugster, P. T., Felber, P. A., Guerraoui, R., & Kermarrec, A. M. (2003). **The many faces of publish/subscribe.** *ACM computing surveys (CSUR)*, 35(2), 114-131.

Publish/subscribe

- Components
 - collaborate by exchanging events
 - publish events they observe
 - subscribe to the events they are interested in
- Communication is:
 - Purely message based
 - Asynchronous
 - Multicast
 - Implicit
 - Anonymous



Event-driven architecture @ Netflix



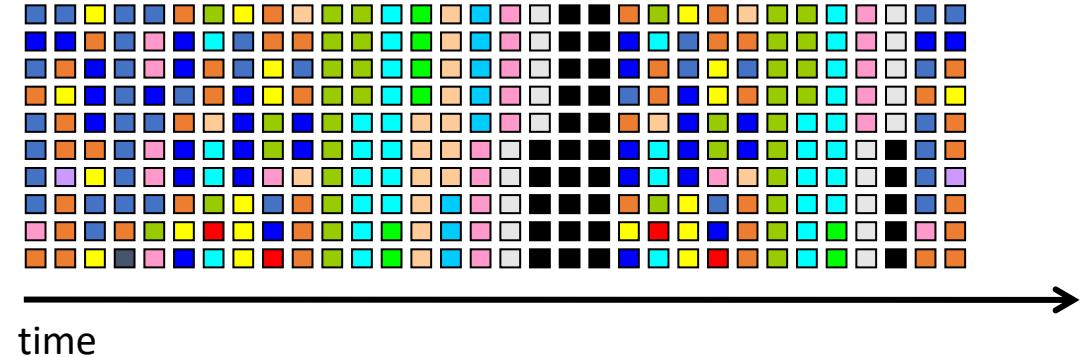
10:45

An aerial photograph of a river flowing through a green landscape. A bridge spans the river in the background. Two red annotations are present: a small red 'D' is located on the dark blue water near a small island, and a larger red circle surrounds a turbulent, white-capped wave on the lighter green water.

Taming
Continuous massive flows
than you cannot stop
with
Data Stream Mng. Systems

Data Stream Management Systems (DSMS)

- Data streams are (**unbounded**) sequences of time-varying data elements
- Represent:
 - an (almost) “continuous” flow of information
 - with the recent information being more relevant as it describes the current state of a dynamic system



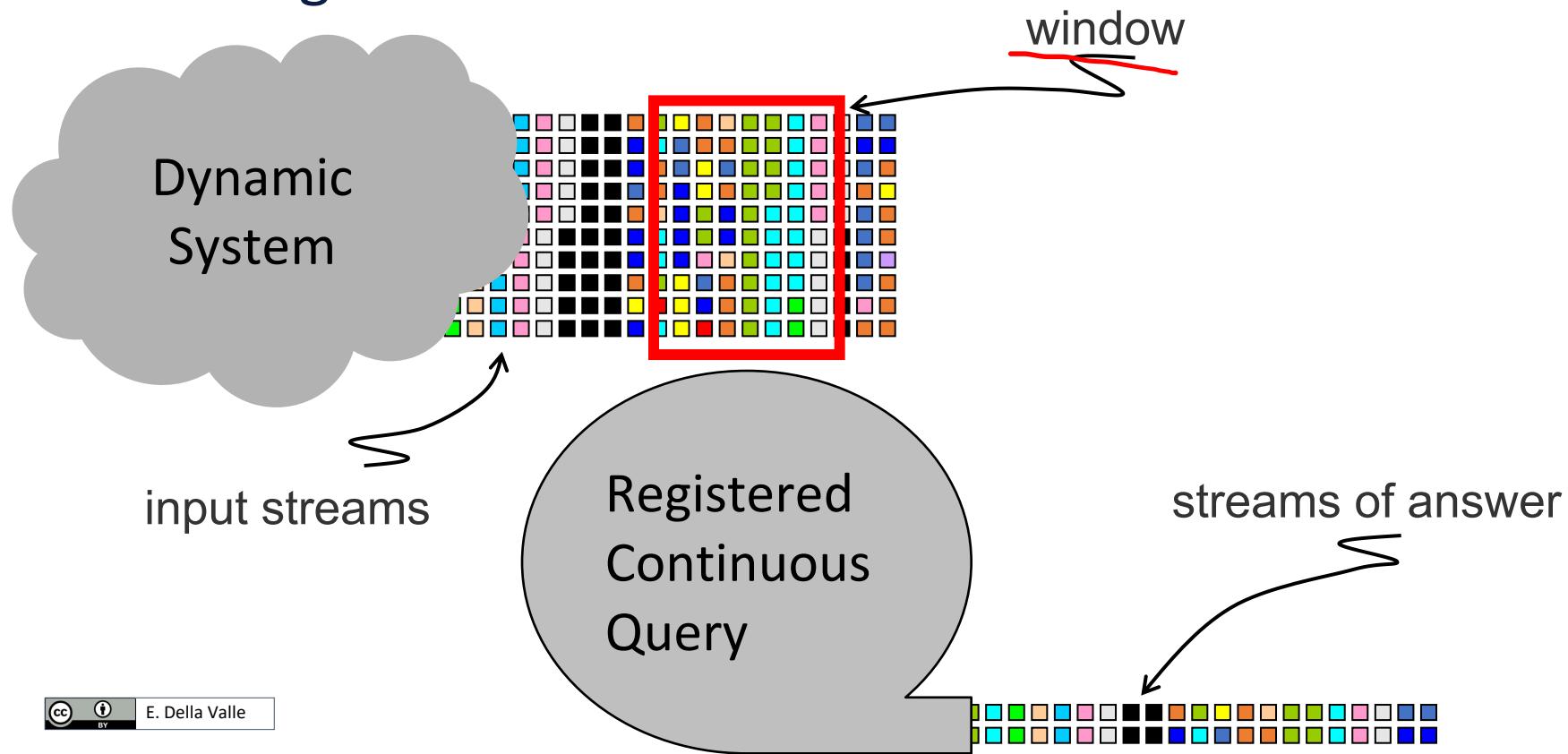
Data Stream Management Systems (DSMS)

- The nature of streams requires a paradigmatic change*
 - **from persistent data**
 - one time semantics
 - **to transient data**
 - continuous semantics

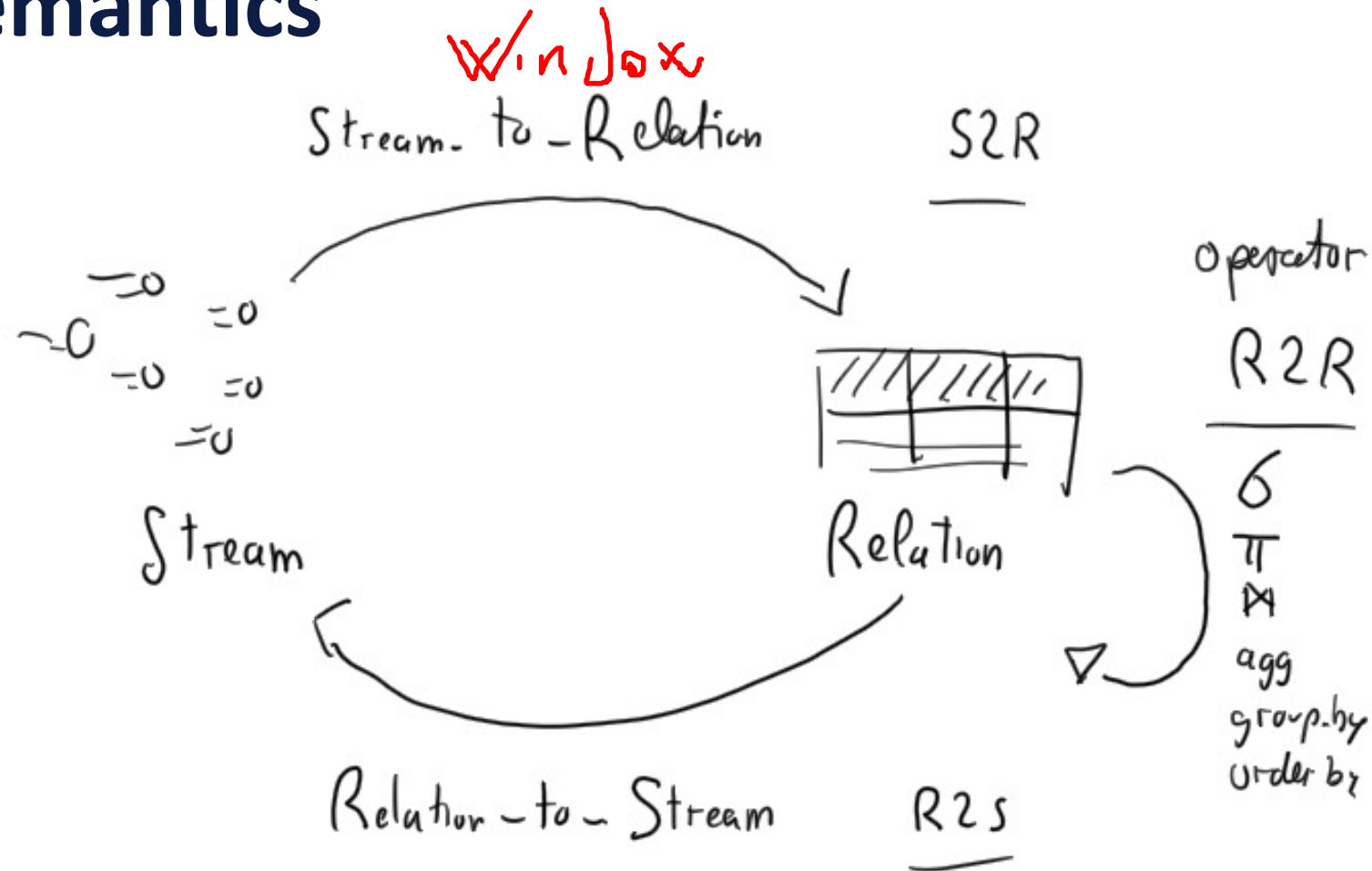
* This paradigmatic change first arose in DB community in **the late '90s**

Continuous Semantics

- Continuous **queries registered** over streams that are observed through **windows**



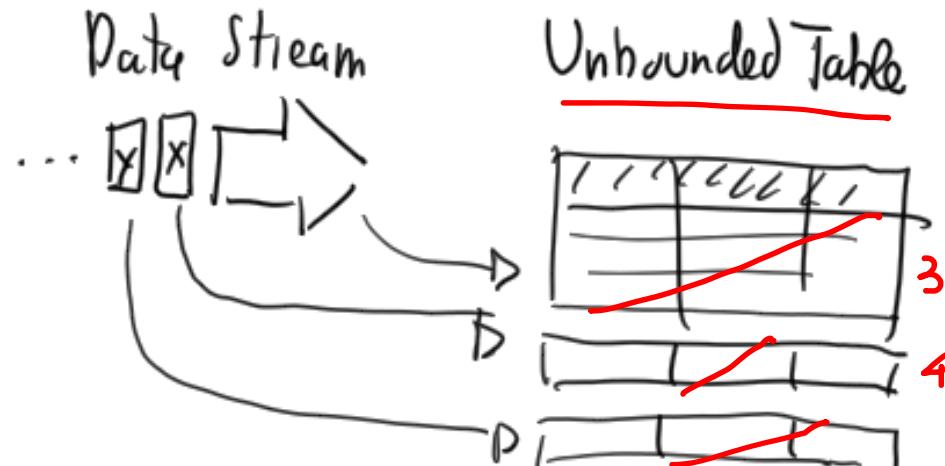
DSMS Semantics



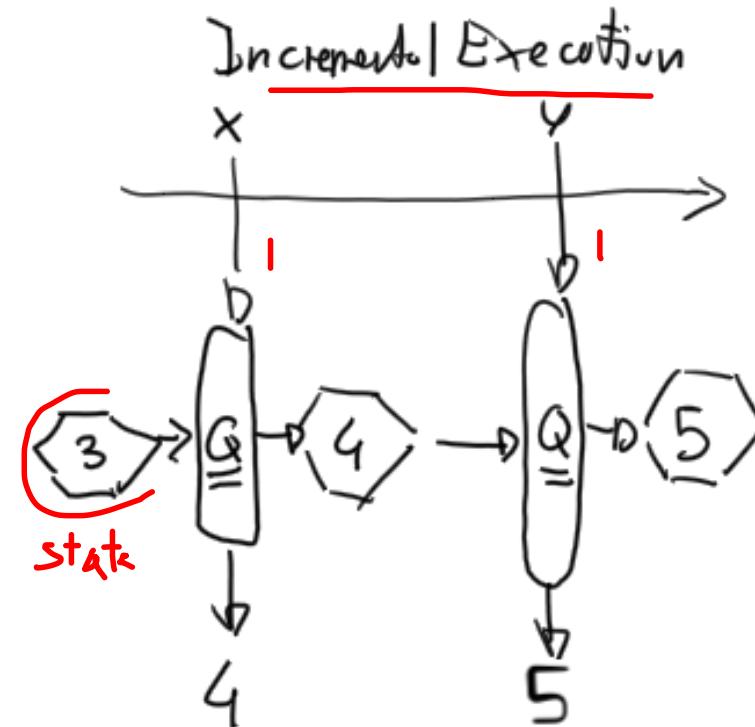
Arasu, A., Babu, S. and Widom, J., 2006. **The CQL continuous query language: semantic foundations and query execution.** *The VLDB Journal*, 15(2), pp.121-142.

Modern DSMS

Logic View



Physical View



$Q = \text{Count}$

Registered

SQL Extensions

Windowed aggregation

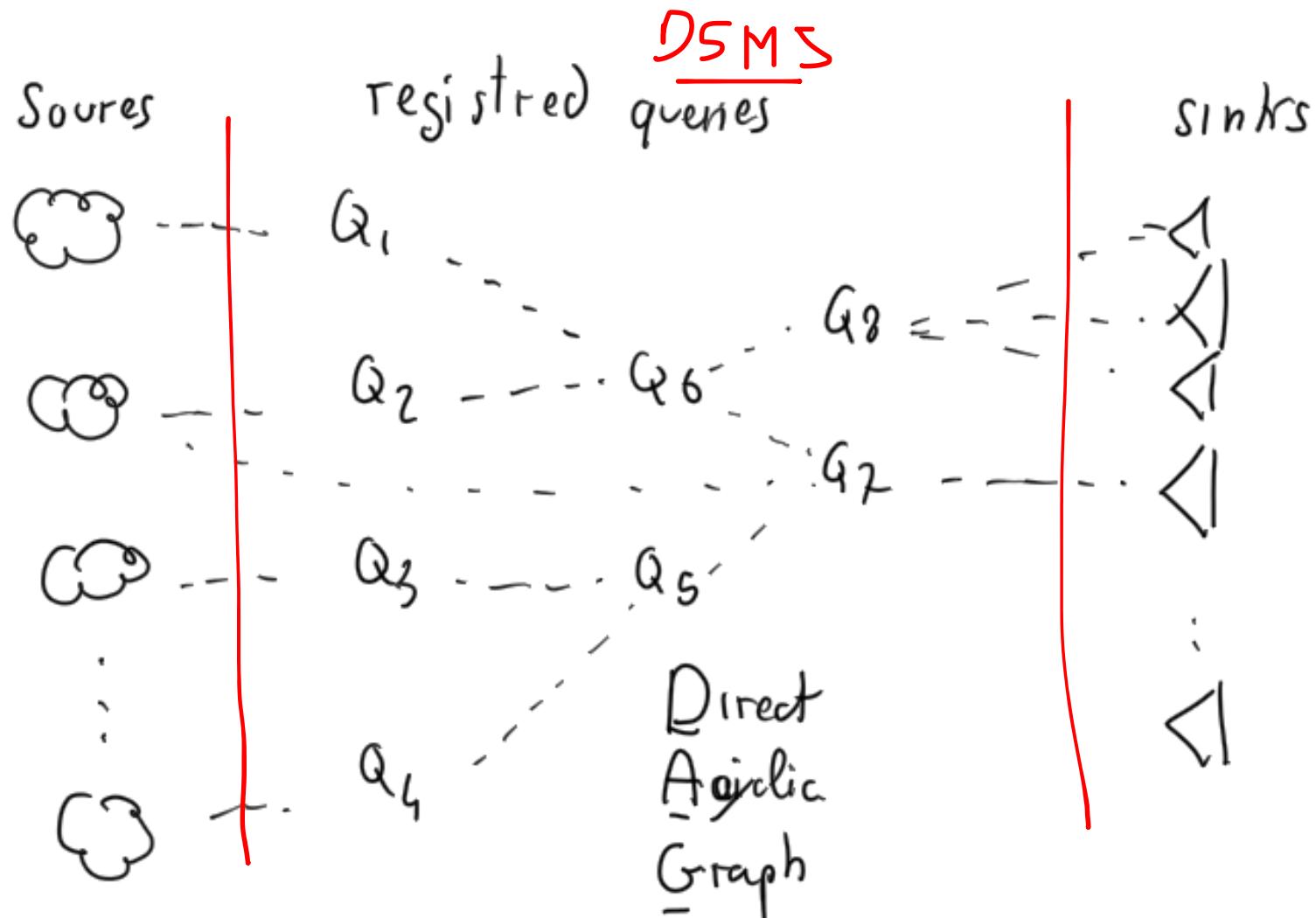
```
SELECT x, COUNT(*), SUM(y)  
FROM s1 WINDOW  
      TUMBLING (SIZE 1 MINUTE)  
GROUP BY x  
EMIT CHANGES;
```



Join two streams together

```
CREATE STREAM s3 AS  
SELECT s1.c1, s2.c2  
FROM s1 JOIN s2  
      WITHIN 5 MINUTES  
ON s1.c1 = s2.c1  
EMIT CHANGES;
```

Typically queries are placed in a network



Typically queries are placed in a network

[src: <https://docs.confluent.io/platform/current/control-center/ksql.html>]

HOME > CONTROLCENTER.CLUSTER > KSQLDB

ksqlDB

Editor Flow Streams Tables Running queries

Search

KSQl_PROCESSING_LOG STREAM

USERS TABLE

PAGEVIEWS_ORIGINAL STREAM

CREATE-STREAM

PAGEVIEWS_ENRICHED STREAM

CF

The screenshot shows the ksqlDB interface within Confluent Control Center. On the left, a sidebar lists cluster management options like Cluster overview, Brokers, Topics, Connect, and Consumers. The 'ksqlDB' option is selected and highlighted in blue. The main area is titled 'ksqlDB' and contains tabs for Editor, Flow, Streams, Tables, and Running queries. Below these tabs is a search bar. The 'Flow' tab is active, displaying a diagram of stream processing. It starts with a 'USERS TABLE' and a 'PAGEVIEWS_ORIGINAL STREAM'. An arrow labeled 'CREATE-STREAM' points from these two sources to a final 'PAGEVIEWS_ENRICHED STREAM'. A red question mark is drawn over the 'CREATE-STREAM' node. To the right of the diagram is a detailed view of the 'PAGEVIEWS_ENRICHED' stream, including its data structure (STREAM), total rows (15938), and updates per second (5.01). The 'Messages' tab is selected, showing a list of messages with columns for USERID, PAGEID, and REGION. The first few entries are: User_6, Page_36, Region_4; User_6, Page_64, Region_4; User_4, Page_81, Region_6.

USERID	PAGEID	REGION
User_6	Page_36	Region_4
User_6	Page_64	Region_4
User_4	Page_81	Region_6
User_7	Page_83	Region_2
User_6	Page_43	Region_4

PAGEVIEWS_ENRICHED

Data structure STREAM
Total rows 15938
Updates /sec 5.01

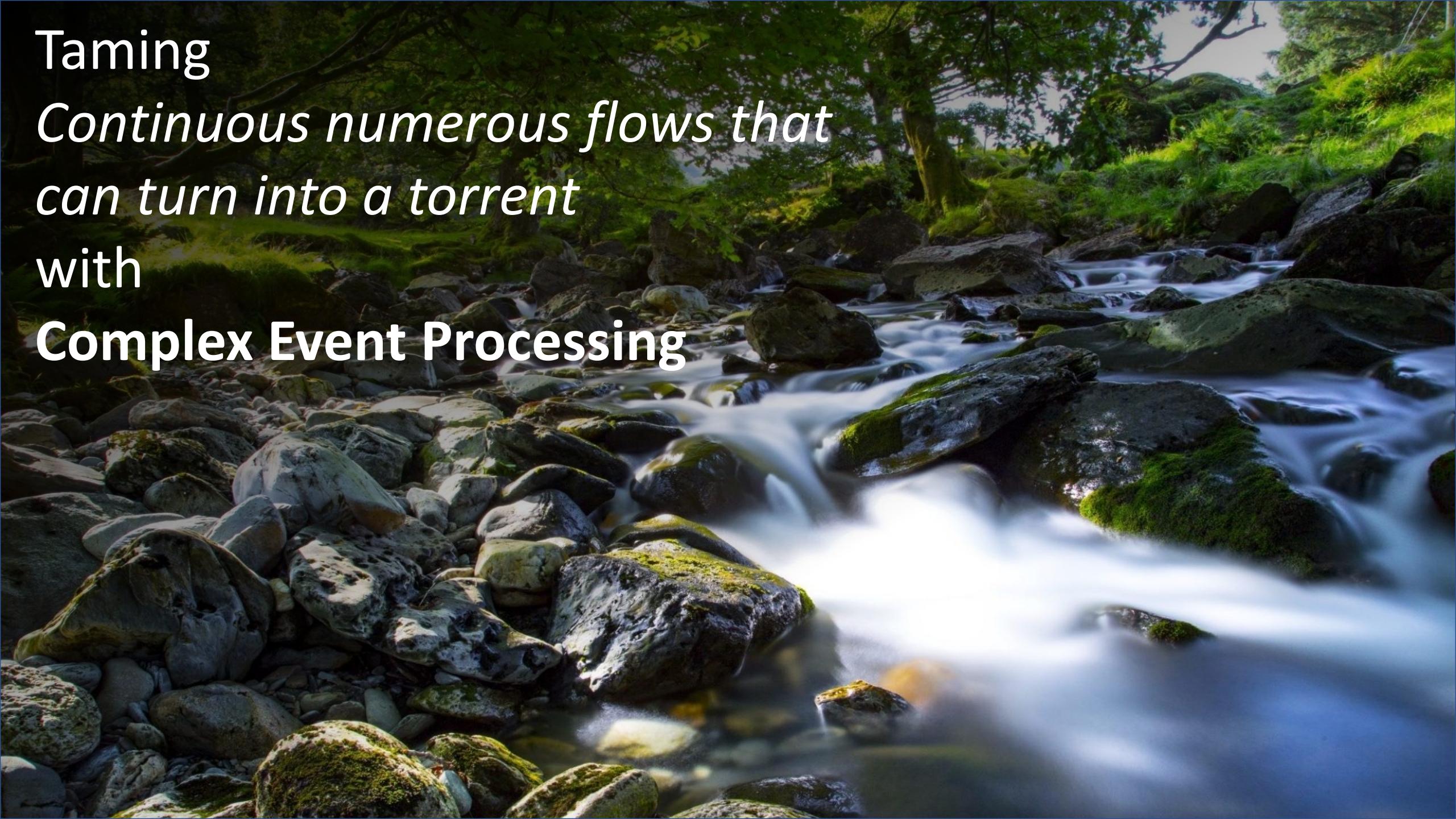
Messages Schema

Filter by keyword

USERID	PAGEID	REGION
User_6	Page_36	Region_4
User_6	Page_64	Region_4
User_4	Page_81	Region_6
User_7	Page_83	Region_2
User_6	Page_43	Region_4

@manudellavalle - <http://emanuele.dellavalle.org>

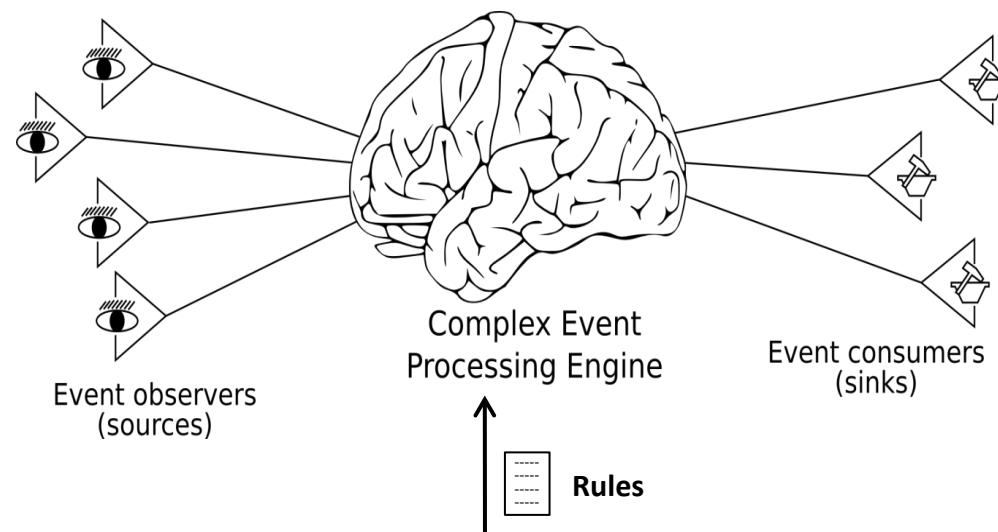
32



Taming
*Continuous numerous flows that
can turn into a torrent*
with
Complex Event Processing

~~Complex~~ Event Processing (CEP)

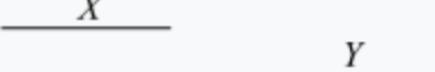
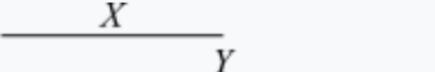
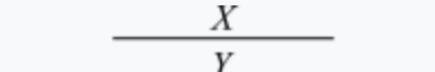
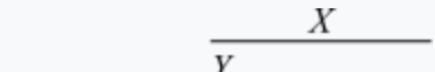
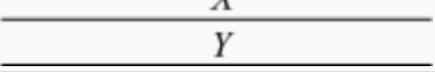
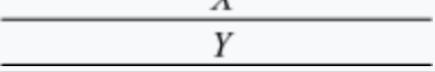
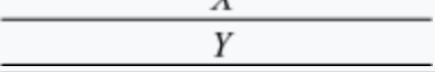
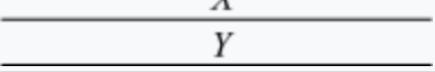
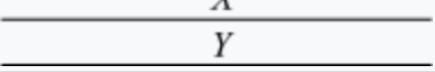
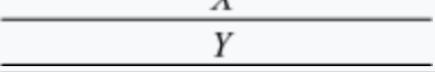
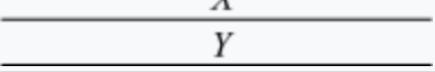
- CEP systems adds the ability to deploy *rules* that describe how composite events can be generated from primitive (or composite) ones
- Typical CEP rules search for *sequences of events*
 - Raise C if A → B
 - Time is a key aspect in CEP



Cugola, G., & Margara, A. (2012). **Processing flows of information: From data stream to complex event processing**. *ACM Computing Surveys (CSUR)*, 44(3), 1-62.

CEP semantics

- subsets of Allen's Interval Algebra

Relation	Illustration	Interpretation
$X < Y$		
$Y > X$		X takes place <u>before</u> Y
$X \mathbf{m} Y$		
$Y \mathbf{mi} X$		X meets Y (<i>i</i> stands for <i>inverse</i>)
$X \mathbf{o} Y$		
$Y \mathbf{oi} X$		X overlaps with Y
$X \mathbf{s} Y$		
$Y \mathbf{si} X$		X starts Y
$X \mathbf{d} Y$		
$Y \mathbf{di} X$		X during Y
$X \mathbf{f} Y$		
$Y \mathbf{fi} X$		X finishes Y
$X = Y$		X is equal to Y

[src: https://en.wikipedia.org/wiki/Allen%27s_interval_algebra]

Detecting Languages



- Event Processing Language (EPL)

```
insert into alertIBM
select *
from pattern [
    every (
        StockTick(name="IBM", price>100)
        ->
        (StockTick(name="IBM", price<100)
            where timer:within(60 seconds))
    );
]
```

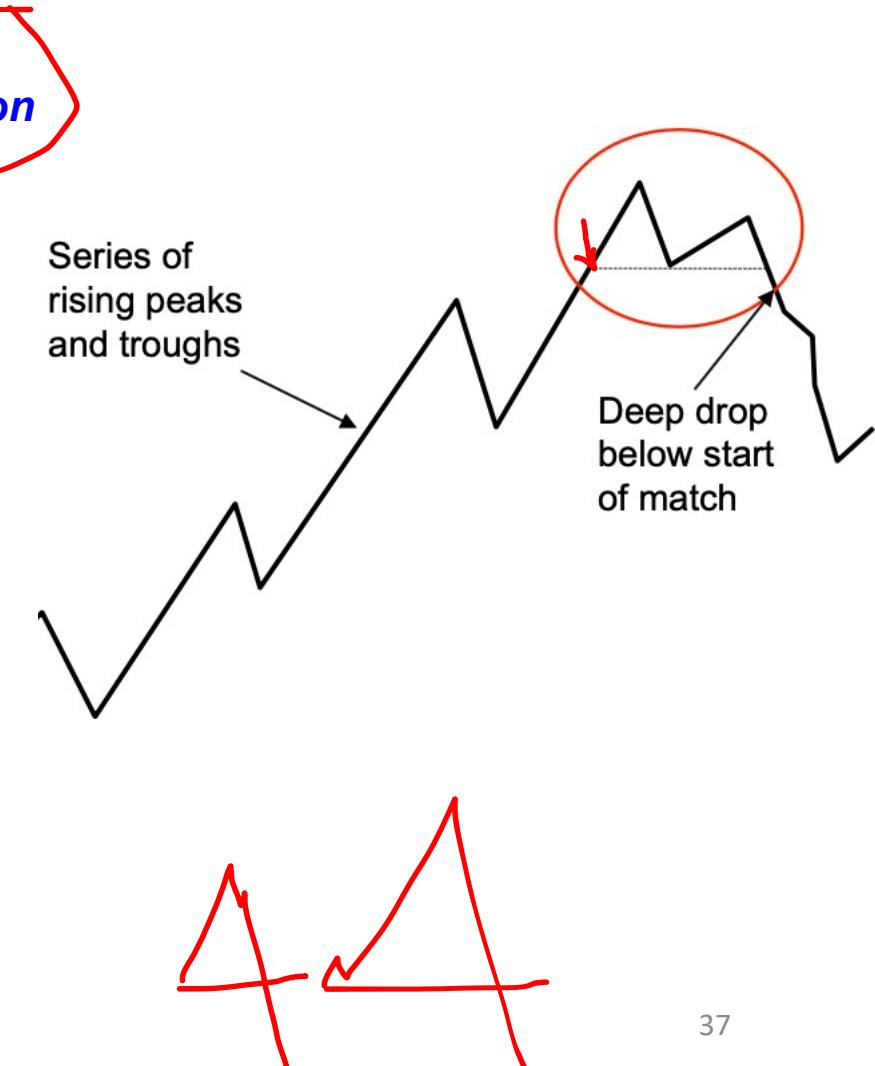
Write a query which alerts on each IBM stock tick with a price greater than 100 followed by an IBM stock tick lower than 100 within 60 seconds

- Supported since 2006 Esper by EsperTech Inc. and Oracle CEP

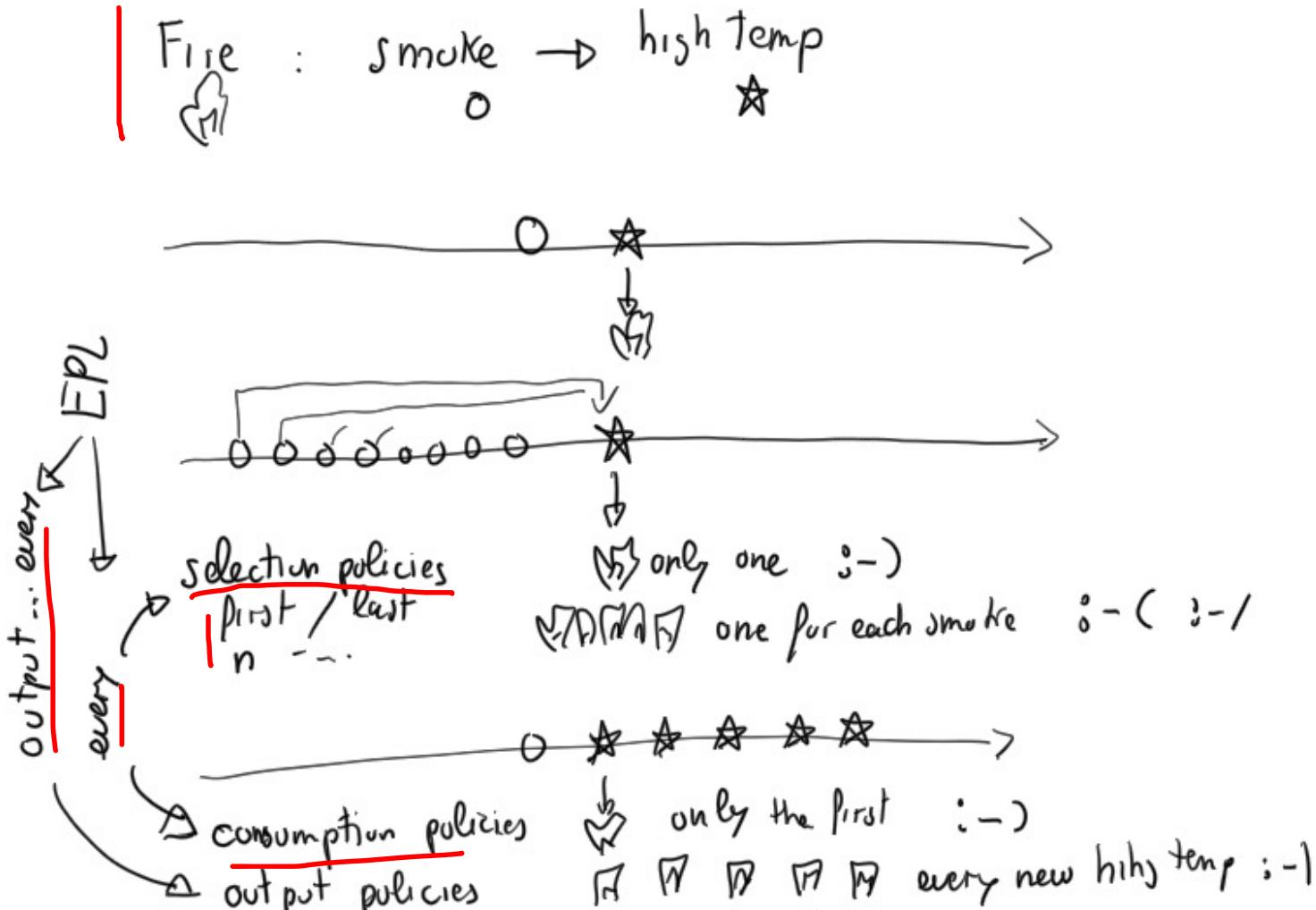
More Expressive Detecting Languages

```
1 stream<MatchT> Matches = MatchRegex(Quotes) {  
2     param  
3         pattern : ". rise+ drop+ rise+ drop* deep";  
4         partitionBy : symbol;  
5         predicates : {  
6             rise = price>First(price) && price>=Last(price),  
7             drop = price>=First(price) && price<Last(price),  
8             deep = price<First(price) && price<Last(price) };  
9     output  
10        Matches : symbol=symbol, seqNum=First(seqNum),  
11                         count=Count(), maxPrice=Max(price);  
12 }
```

→ Operator only, no extensions to SPL syntax



Why can event-based systems fight the torrent effect?



Wrapping up



Thank you for your attention!

Questions?

Credits

- These slides are partially based on
 - "A modeling framework for DSMS and CEP" by G. Cugola and A. Margara presented in the PhD course on "Stream and complex event processing" offered by Politecnico di Milano in 2015.
 - <http://www.streamreasoning.org/courses/scep2015>
 - "Tutorial: Stream Processing Languages" by Martin Hirzel, IBM Research AI. 1 November 2017 Dagstuhl Seminar on Big Stream Processing Systems
 - <https://materials.dagstuhl.de/files/17/17441/17441.MartinHirzel1.Slides.pdf>
 - "The Death and Rebirth of the Event Driven Architecture" by Jay Kreps, Confluent, Kafka Summit 2018
 - <https://www.confluent.io/kafka-summit-london18/keynote-the-death-and-rebirth-of-the-event-driven-architecture>
 - "Time Series Databases" by Dmitry Namiot
 - <https://www.slideshare.net/coldbeans/on-timeseries-databases>
 - "Fractal Tree Indexes : From Theory to Practice" by Tim Callaghan
 - <https://www.slideshare.net/tmcallaghan/20131112-plukfractaltreestheorytopractice>

Streaming Data Engineering

A Tale of Three Streams

Emanuele Della Valle

prof @ Politecnico di Milano

founder @ Quantia Consulting

Founder & CRO @ motus ml

12 : 00
