Patterns for real-time stream processing

Martin Kleppmann

@martinkkl
Stream processing frameworks

- Storm
- Spark Streaming
- Samza
- Kafka Streams
- Concord
- Flink
- etc...
Event streams
“something happened”

Event streams

subscribe to it
"SOMETHING HAPPENED"

- User x clicked link y (activity event)
"SOMETHING HAPPENED"

- User $x$ clicked link $y$ (activity event)
- Sensor $x$ sent reading $y$ (time series tick)
FILTER (MAP)

\[\begin{align*}
5 & \rightarrow 12 & \rightarrow 19 & \rightarrow 15 \\
\end{align*}\]

\[X > 10?\]

\[\begin{align*}
12 & \rightarrow 19 & \rightarrow 15 \\
\end{align*}\]
FILTER (MAP)

5 12 8 19 15 → X>10? → 12 19 15

AGGREGATE (COUNT, SUM, AVG, ...)

A C B A B A B A → COUNT(X) → A:3 B:2 C:1
How to draw an owl

1. Draw some circles

2. Draw the rest of the fucking owl
PageView Event
eventType: PageViewEvent,
timestamp: 1413215518,
viewerId: 1234,
sessionId: fa1afe101234deadbeef,
pageKey: profile-view,
viewedProfileId: 4321,
trackingKey: invitation-email,
... etc. metadata about what content was displayed...
PageViewEvent

Who viewed your profile
PageViewEvent

People also viewed

Who viewed your profile
People Also Viewed

Rahul Vohra
LinkedIn Intro. CEO of Rapportive. Computer Scientist, Gamer, Entrepreneur.

Sam Stokes
Builder, technologist, mixologist

Jay Kreps
Principal Staff Engineer at LinkedIn

Rashmi Sinha
Cofounder and Head at SlideShare, a LinkedIn company

Lee Mallabone
Senior Engineer at LinkedIn

Jakob Homan
Hadoop Ecosystem Engineer
Who's viewed your profile

- **411** Profile views Last 90 days
- **98** Viewers found you from People similar to you
- **27** Viewers who work at LinkedIn
- **70** Viewers with the title Software Developer

Oct 13 – now

- **9** Profile views -- from the previous week
- **0** Actions taken -- from the previous week

Getting more profile views can help you get found for the right opportunity.

Get more profile views →
How you rank for profile views

27 Viewers who work at LinkedIn
70 Viewers with the title Software Developer

Oct 13 – now

9 Profile views
0 Actions taken

Getting more profile views can help you get found for the right opportunity.

Get more profile views:
eventType: PageViewEvent,
timestamp: 1413215518,
viewerId: 1234,
sessionId: fa1afe101234deadbeef,
pageKey: profile-view,
viewedProfileId: 4321,
trackingKey: invitation-email,
... etc. metadata about what content was displayed...
enrich with profile data

{viewerId: 1234, viewProfile: {jobTitle: "Software Engineer"}}
{viewerId: 1234}
(enrich with profile data)

{viewerId: 1234, viewerProfile: {
  jobTitle: "Software Engineer"
}}

indexing

70 Viewers with the title Software Developer
PageViewEvent stream

{viewerId: 12343}

Stream processor

PageViewEventWithViewerProfile
PageViewEvent stream

{viewerId: 1234}

Stream processor

get 1234 →

{jobTitle: "...", ...

Profiles

PageViewEventWithViewerProfile
STREAM-TABLE JOIN

Activity events

<table>
<thead>
<tr>
<th>key</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>A</td>
</tr>
<tr>
<td>234</td>
<td>B</td>
</tr>
<tr>
<td>345</td>
<td>C</td>
</tr>
<tr>
<td>456</td>
<td>D</td>
</tr>
</tbody>
</table>

Table / k-v store
STREAM-TABLE JOIN

Activity events

Lookup

Table / k-v store
PageViewEvent stream

~100-500k msg/sec

Stream processor

~1-5k req/sec

Profiles

PageViewEventWithViewerProfile
Too slow!
STREAM-TABLE JOIN

Activity events

Lookup

<table>
<thead>
<tr>
<th>key</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>A</td>
</tr>
<tr>
<td>234</td>
<td>B</td>
</tr>
<tr>
<td>345</td>
<td>C</td>
</tr>
<tr>
<td>456</td>
<td>D</td>
</tr>
</tbody>
</table>

Table / k-v store
STREAM-TABLE JOIN

Activity events

Lookup

Table / k-v store

Update

Table changelog

<table>
<thead>
<tr>
<th>key</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>A</td>
</tr>
<tr>
<td>234</td>
<td>B</td>
</tr>
<tr>
<td>345</td>
<td>C</td>
</tr>
<tr>
<td>456</td>
<td>D</td>
</tr>
</tbody>
</table>
"SOMETHING HAPPENED"

- User x clicked link y (activity event)
- Sensor x sent reading y (time series tick)
"SOMETHING HAPPENED"

- User x clicked link y (activity event)
- Sensor x sent reading y (time series tick)
- Database record x updated to y (data change event)

Immutable, point in time
{
    eventType: ProfileEditEvent,
    timestamp: 1413215518,
    profileId: 1234,
    old: {
        location: "London, UK",
        industry: "Financial Services"
    },
    new: {
        location: "Budapest, Hungary",
        industry: "Software"
    }
}
ProfileEditEvent

Samza
Samza → Profiles

Set 1234, {old: ..., new: ...} → ProfileEditEvent

Profiles (a copy just for us!)
PageViewEvent

ProfileEditEvent

{profileId: 1234, old: {...}, new: {...}}

Samza

Profiles (a copy just for us!)
PageView Event

{viewerId: 1234}

Samza

Profile Edit Event

Profiles

get 1234

{jobTitle: ""}

(a copy just for us!)
JOINZ

NOT DEAD
SAMZA
RETURN OF THE JOIN
STREAM-STREAM (WINDOW) JOIN

Search queries

Result clicks
STREAM-STREAM (WINDOW) JOIN

Search queries

Result clicks

<table>
<thead>
<tr>
<th>Value</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>foo</td>
<td>19</td>
</tr>
<tr>
<td>bar</td>
<td>24</td>
</tr>
<tr>
<td>baz</td>
<td>28</td>
</tr>
<tr>
<td>qux</td>
<td>30</td>
</tr>
<tr>
<td>hello</td>
<td>39</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>foo</td>
</tr>
<tr>
<td>29</td>
<td>bar</td>
</tr>
<tr>
<td>32</td>
<td>baz</td>
</tr>
<tr>
<td>42</td>
<td>qux</td>
</tr>
</tbody>
</table>
STREAM-STREAM (WINDOW) JOIN

Search queries

Result clicks
STREAM-STREAM (WINDOW) JOIN

Search queries

Result clicks

emit joined event
Kafka
stream

oldest events

most recent events
stream

new events added here

← oldest events

most recent events →
stream

new events added here

oldest events

most recent events

real-time consumer position (close to head of stream)
Scaling by co-partitioning
Kafka

Samza

Embedded database
(Level DB/Rocks DB)
Kafka

Samza

Data lost 😞
stream new events added here

oldest events most recent events

complete history
(using compaction to collect garbage)
Kafka changelog compaction
Kafka

Samza

Data lost 😞
stream

new events added here

← oldest events

most recent events →

↑

re-processing historical events starts here

real-time consumer position

(close to head of stream)
2. Data standardization

(s13u?)
software engineer
computer programmer

developer

software engineer
ProfileEditEvent

{ jobTitle: "Code magician", ... }

update search indexes
ProfileEditEvent

{ jobTitle: "Code magician", ... }

Standardization job

{ jobTitle: "Code magician", StandardTitle: "Software Engineer", ... }

StandardizedProfile Event

update search indexes
**ProfileEdit Event**

```json
{ jobTitle: "Code magician", ... }
```

**Standardization job**

```json
{ jobTitle: "Code magician", standardTitle: "Software Engineer", ... }
```

**Standardized Profile Event**

- update search indexes
- other
Profile Edit Event

{ jobTitle: "Code magician", ...

Standardization job

{ jobTitle: "Code magician", standardTitle: "Software Engineer", ...

Standardized Profile Event

update search indexes

other
Near real-time index updates

⇒ Stream process

Standardization rules change

⇒ Re-classify everything

⇒ Batch process
ProfileEditEvent → HDFS

- Stream process
- Batch (M/R) process

Merge

update search indexes

other
architecture

“implement the same job twice”
K architecture

“process real-time data and re-process historical data in the same framework”
Profile Edit Event

Standardization job version 1

Standardization job version 2
ProfileEditEvent

Standardization job version 1

Standardization job version 2

Standardized profiles v1 v2
ProfileEditEvent

- Standardization job version 1
- Standardization job version 2

Standardized profiles

Clients
ProfileEditEvent

Standardization job version 1

Standardization job version 2

Standardized profiles

clients

switch over when ready
How to draw an owl

1. Draw some circles

2. Draw the rest of the fucking owl
STREAM PROCESSING PATTERNS.

- Filter/map (stateless)
- Aggregate
- Stream/table join (changelog streams)
- Window join (message buffer, co-partitioning)
- Reprocessing (batch & streaming converge)
References

Designing Data-Intensive Applications

Martín Kleppmann