
RI.SE

# WIP: Pods: Privacy Compliant Scalable Decentralized Data Services

**Jonas Spenger[12] <jspenger@kth.se>**, Paris Carbone[12], Philipp Haller[2]

[1]RISE Research Institutes of Sweden, Stockholm, Sweden
[2]Digital Futures and EECS, KTH Royal Institute of Technology, Stockholm, Sweden

Presented at Poly'21: Polystore systems for heterogeneous data in multiple databases with privacy and security assurances, August 20, 2021

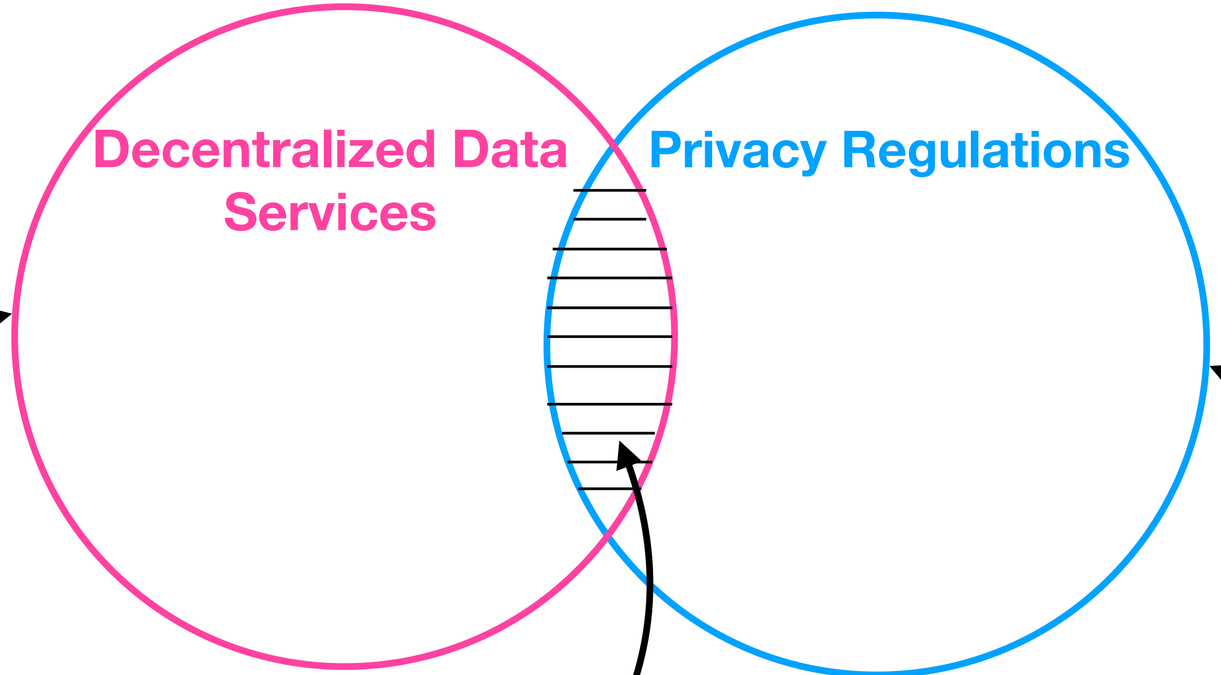
# Outline

1. Problem Scope
2. Pods: Our Approach
3. Open Questions / Research Directions

# Outline

1. **Problem Scope**
2. Pods: Our Approach
3. Open Questions / Research Directions

# Problem Scope



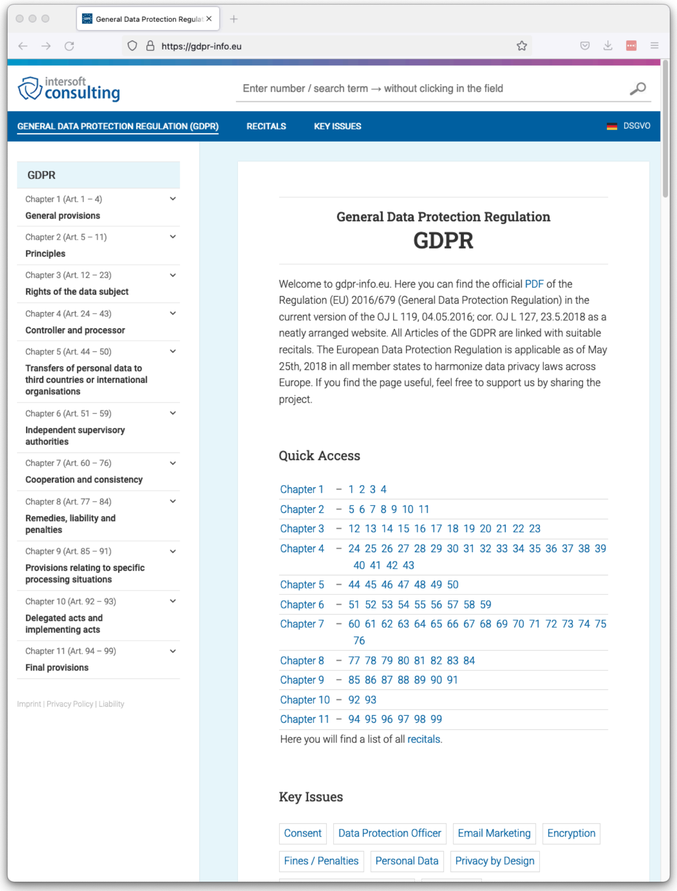

- Dataflow composition
- High-performance
- Elastically scalable
- Failure resilient
- Consistency
- Easy to program

- GDPR; CCPA
- Transparent

- Privacy compliant decentralized data services, **our work**

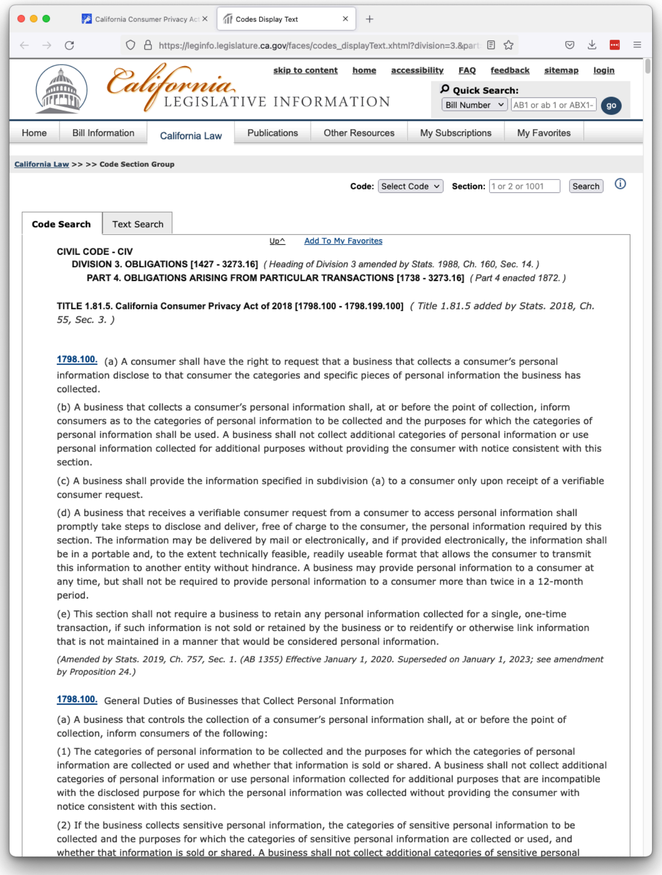
# Privacy Regulations

## GDPR: General Data Protection Regulation




https://gdpr-info.eu/

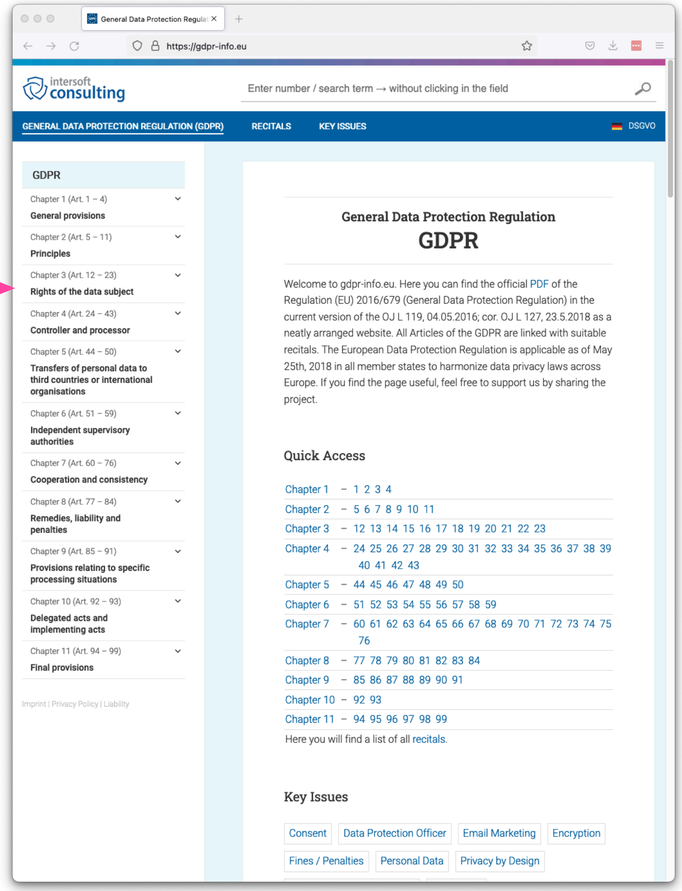
## CCPA: California Consumer Privacy Act




https://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?division=3.&part=4.&lawCode=CIV&title=1.81.5

# Privacy Regulations

## GDPR: General Data Protection Regulation




https://gdpr-info.eu/

### Chapter 3
### Rights of the data subject

| | | |
|---|---|---|
| **Section 1** | – | **Transparency and modalities** |
| Article 12 | – | Transparent information, communication and modalities for the exercise of the rights of the data subject |
| **Section 2** | – | **Information and access to personal data** |
| Article 13 | – | Information to be provided where personal data are collected from the data subject |
| Article 14 | – | Information to be provided where personal data have not been obtained from the data subject |
| Article 15 | – | Right of access by the data subject |
| **Section 3** | – | **Rectification and erasure** |
| Article 16 | – | Right to rectification |
| Article 17 | – | Right to erasure ('right to be forgotten') |
| Article 18 | – | Right to restriction of processing |
| Article 19 | – | Notification obligation regarding rectification or erasure of personal data or restriction of processing |
| Article 20 | – | Right to data portability |
| **Section 4** | – | **Right to object and automated individual decision-making** |
| Article 21 | – | Right to object |
| Article 22 | – | Automated individual decision-making, including profiling |
| **Section 5** | – | **Restrictions** |
| Article 23 | – | Restrictions |

# GDPR: Rights of the Data Subject

## Right of Access

- Grant access to “what”/“how” personal data is being processed
- Reply within one month

## Right to Erasure

- Erase all personal data concerning the data subject
- Within one month

## Right to Object

- Object to certain types of processing
- Within one month

Operations/Privacy Requests:

**Access-Request**
<$P_{access}$, datasubject>

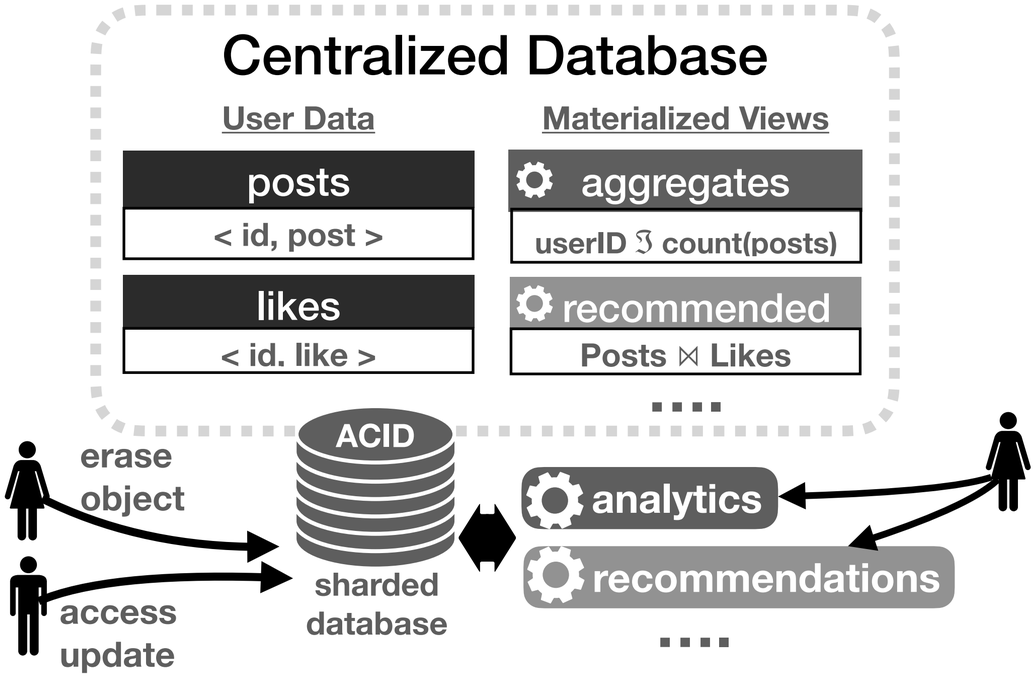
**Erasure-Request**
<$P_{erase}$, datasubject>

**Objection-Request**
<$P_{object}$, datasubject, purpose>

! Data service should implement these operations

# Consistent Privacy Requests

## Ideal Scenario




**Pros:**
Privacy requests are regular operations, ACID => consistent execution

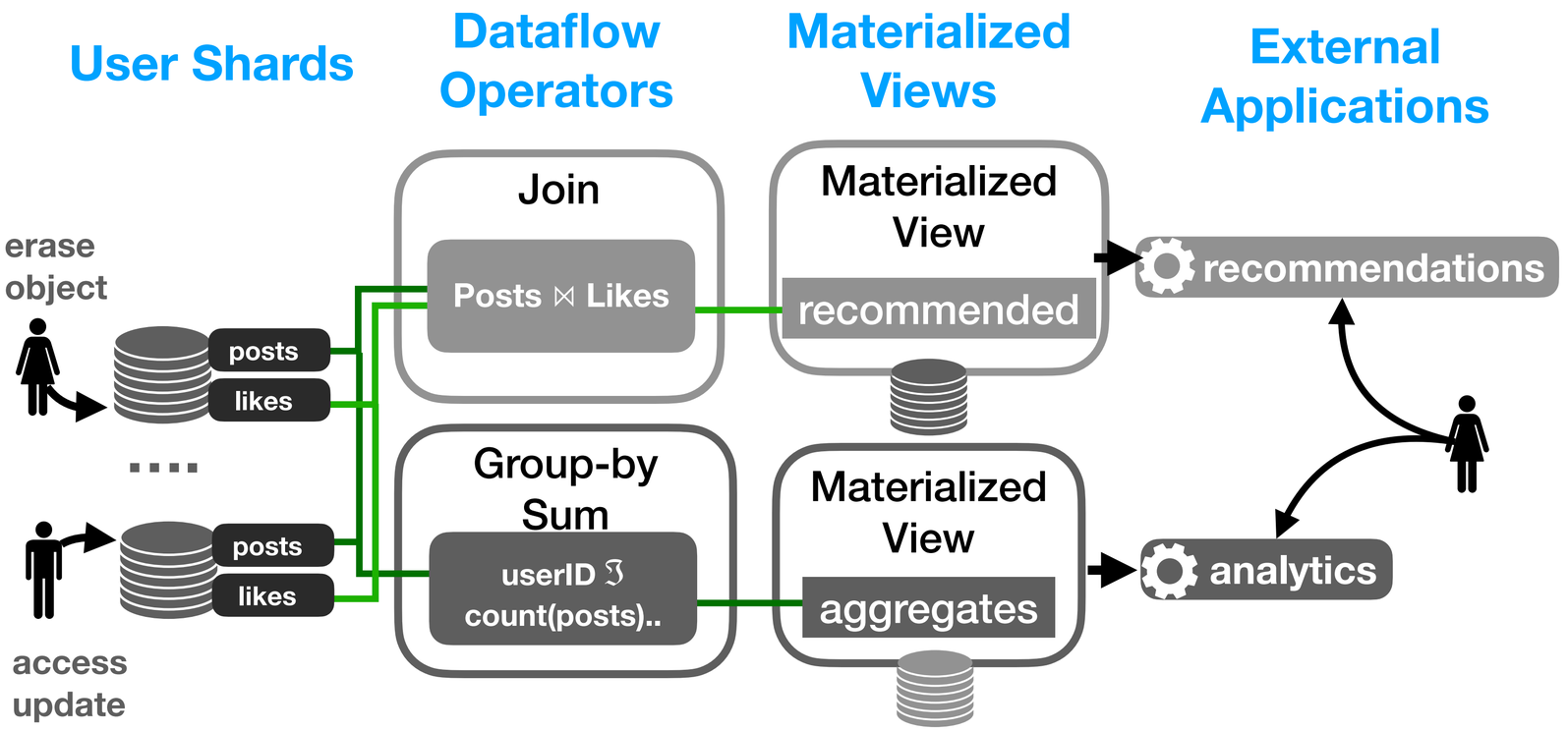
**Cons:**
Not scalable

# Consistent Privacy Requests

Pros: Simple, decentralized storage, distributed data centres, high-performance

Cons:
Eventual consistency, relational, coarse-grained

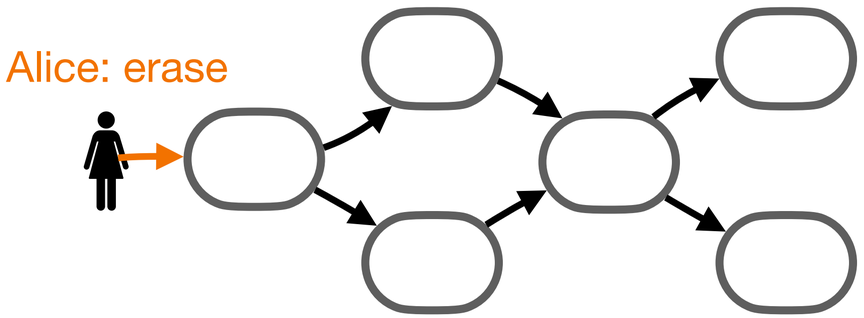
## Real-World Scenario




*Concepts from Schwarzkopf et al. "Position: GDPR compliance by construction" [16, 11]*

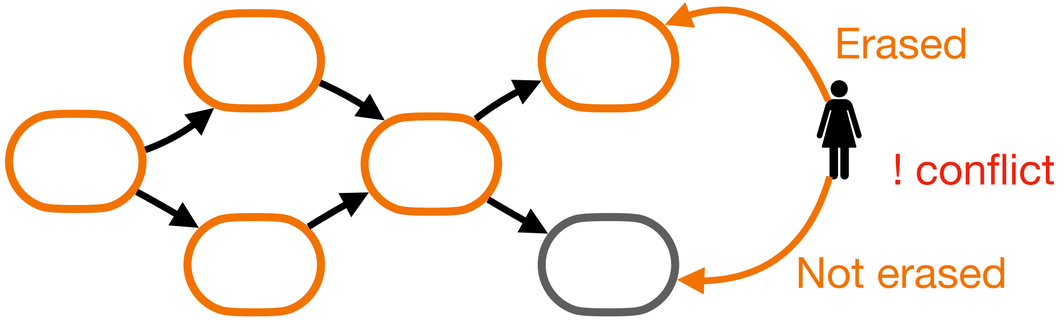
# Example of Issues

Eventually consistent dataflow system

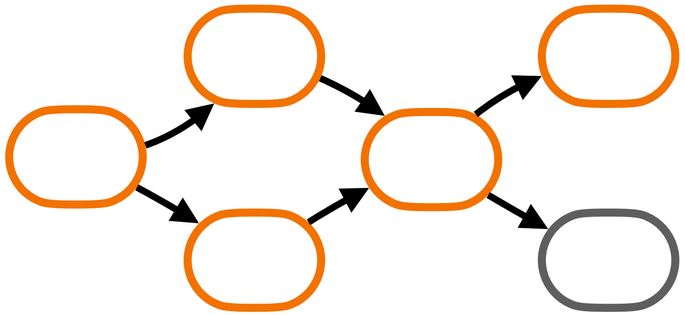


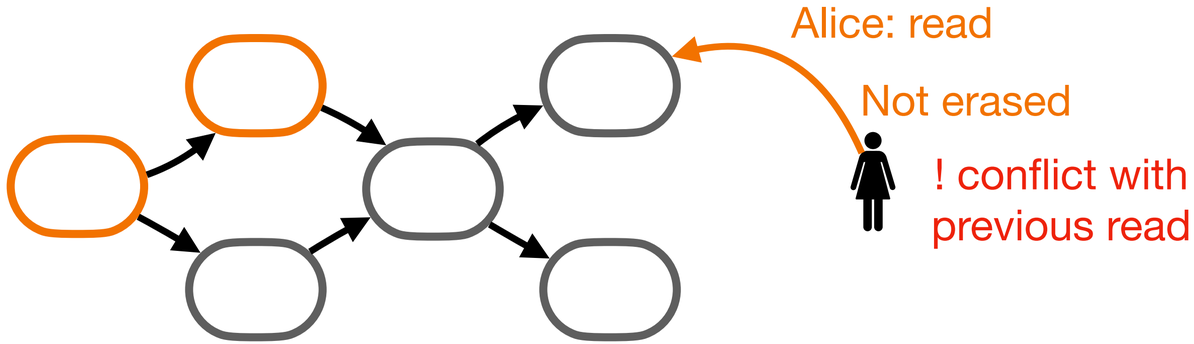

erasure request is propagated

Alice: read

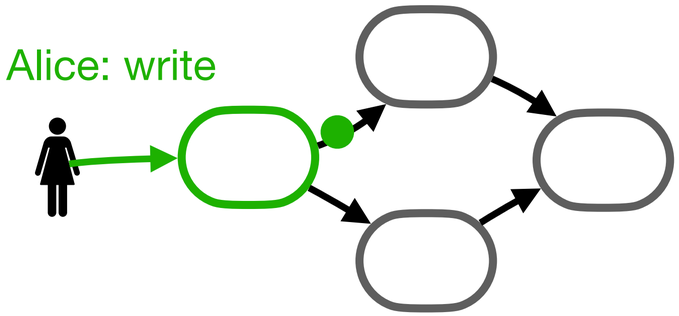



! Failure -> rollback failure recovery

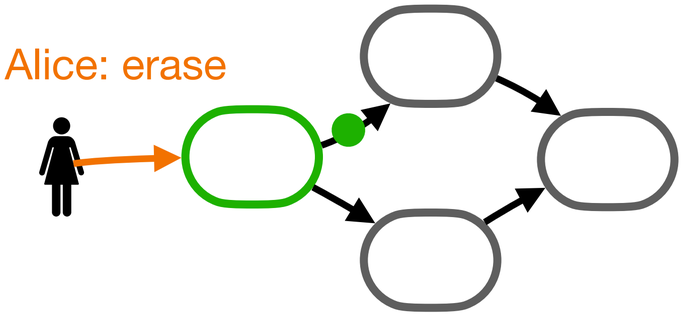




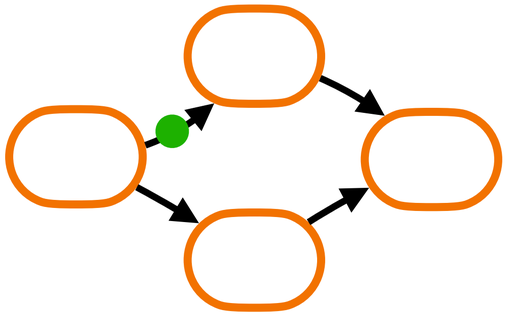
# Example of Issues



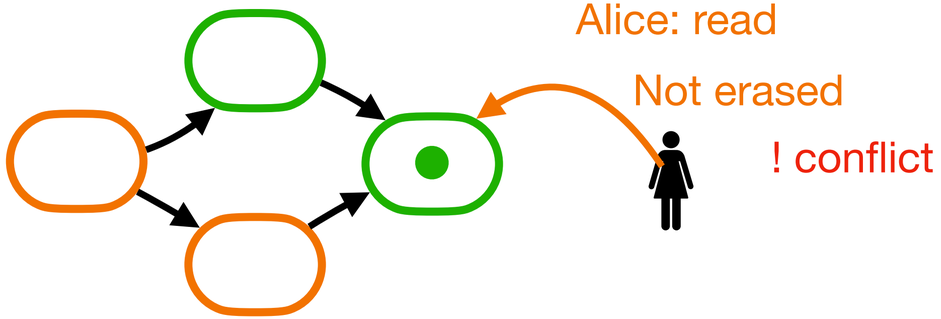

write request is propagated




! Erase on all operators atomically



In-flight message (green) not affected



# Our Approach to Consistent Privacy Requests

- **Causally consistent reads from materialized views**
  - Subsequent reads observe same or more updated state
- **Serializable privacy requests**
  - Effect of privacy request is as if it was executed atomically, with no concurrent operations on the system
- **Executing privacy requests**
  - Materialize and execute the request according to specification
  - UDFs
  - Fine-grained

# Outline

1. Problem Scope
2. **Pods: Our Approach**
3. Open Questions / Research Directions

# Pods: Our Approach

**Goal:**

- Scalable, failure-resilient, high-performance, dataflow composition
- Privacy compliance

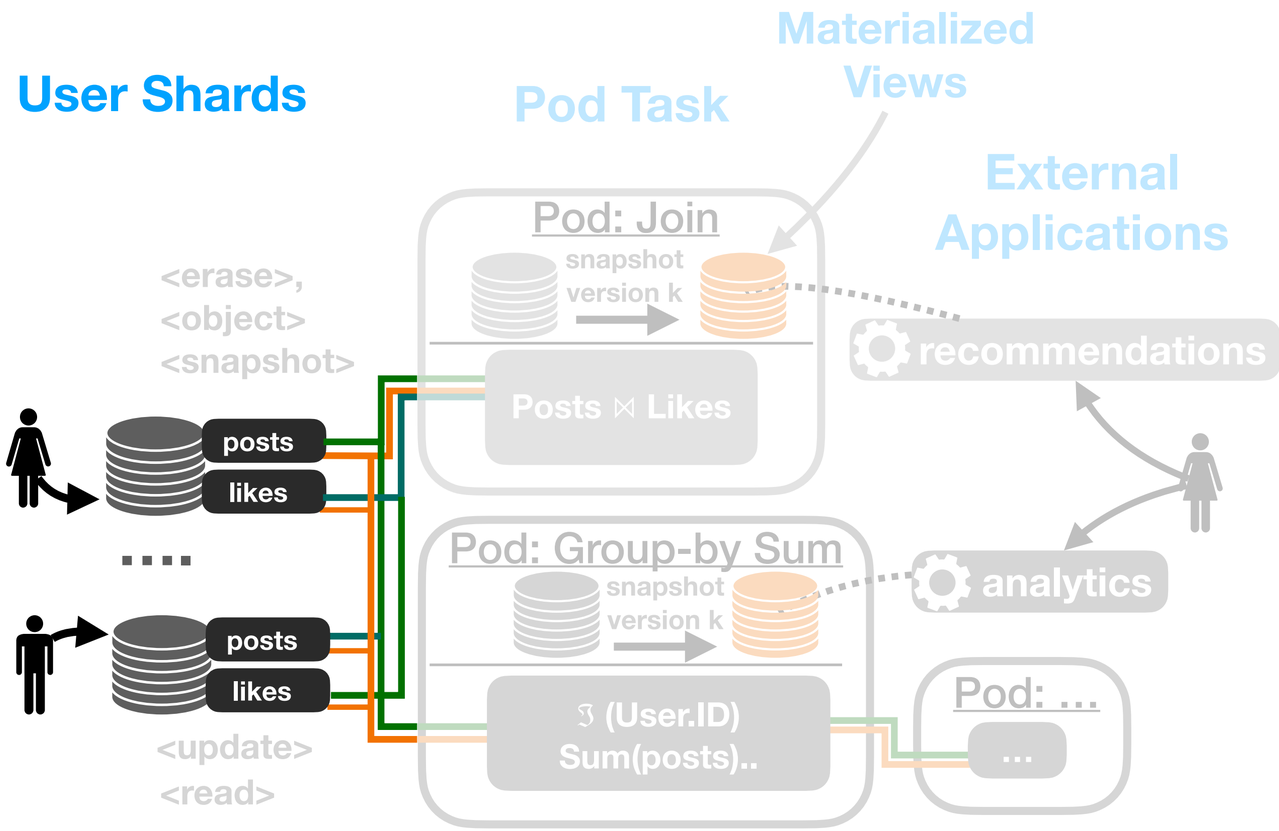
**High-level:**

- User-shards + dataflow composition [16]
- Couple data with policy-metadata ("Data Capsule") [22], fine-grained

# Pods: Our Approach

## User shards

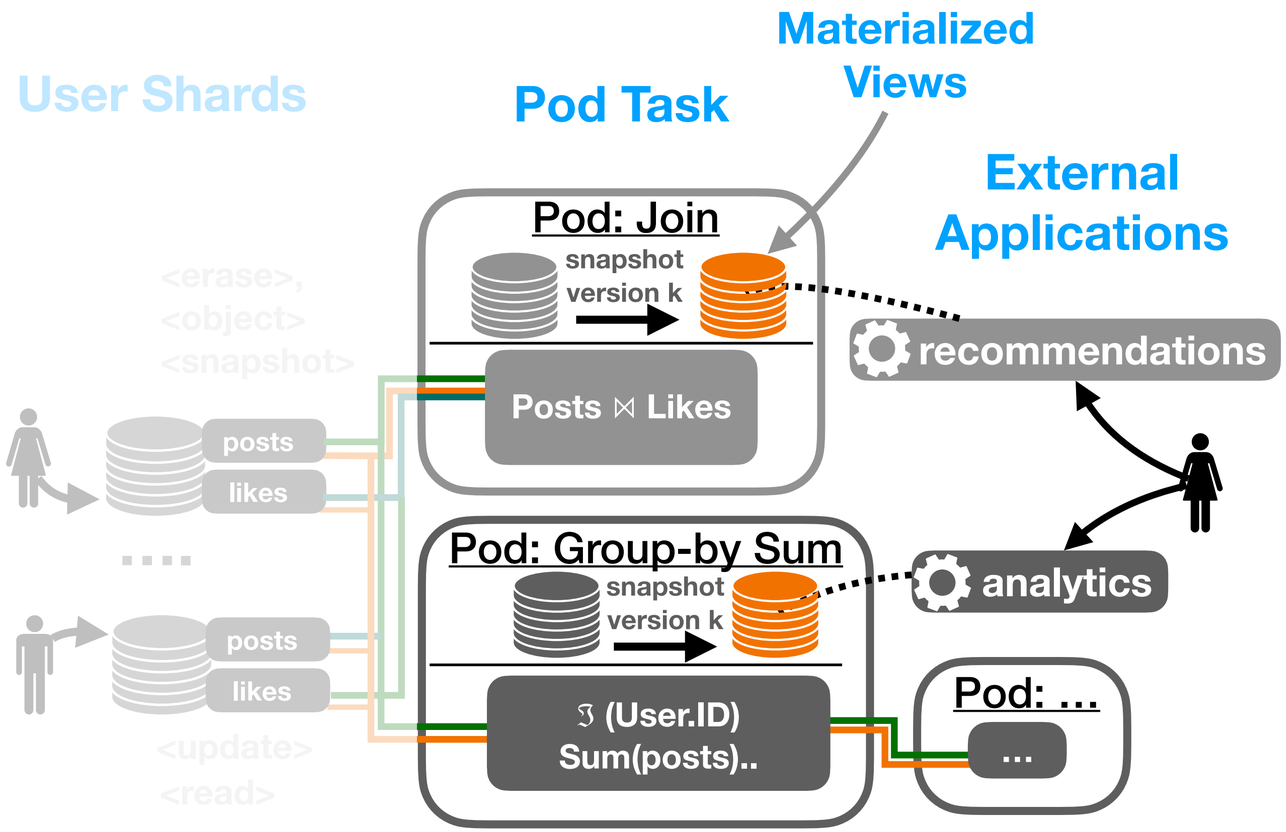
- User data is ingested in “user shards” [16]
- Persistently logged, replicated
- Emits events on update



# Pods: Our Approach

**Pod tasks**

- Subscribe to input streams, execute operations, generate output streams
- Snapshotted consistent state is externally queryable

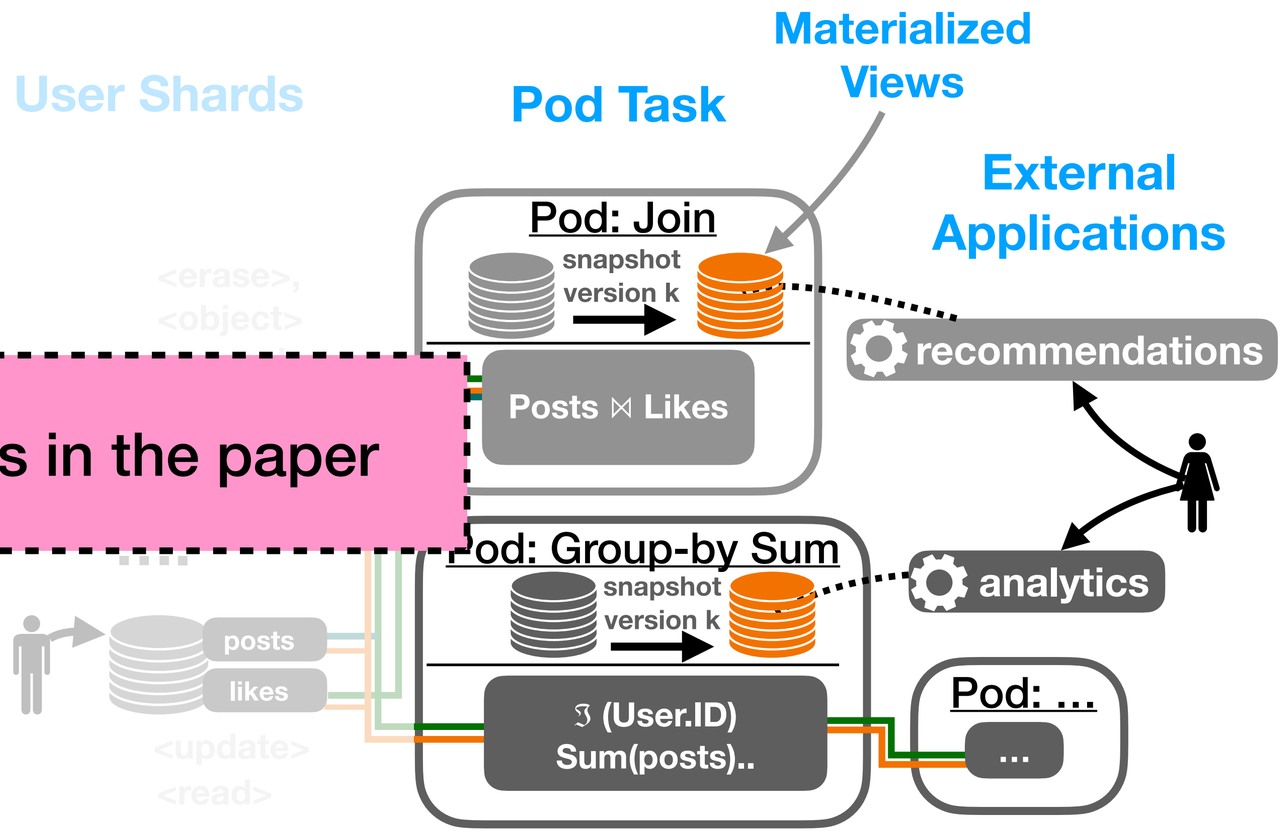


# Pods: Our Approach

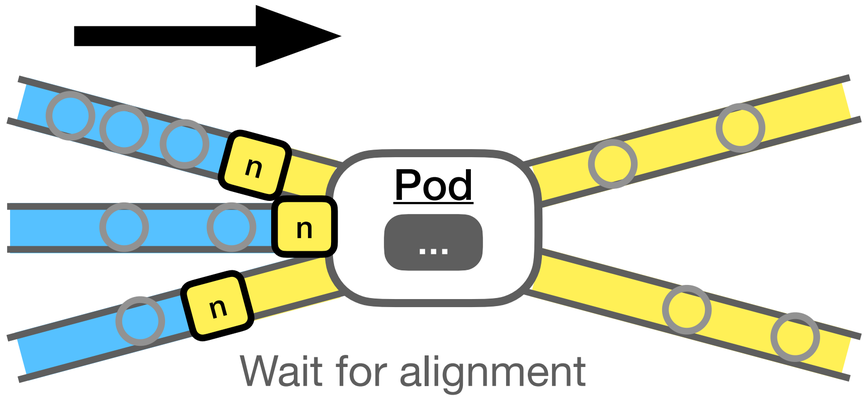
**Pod tasks**

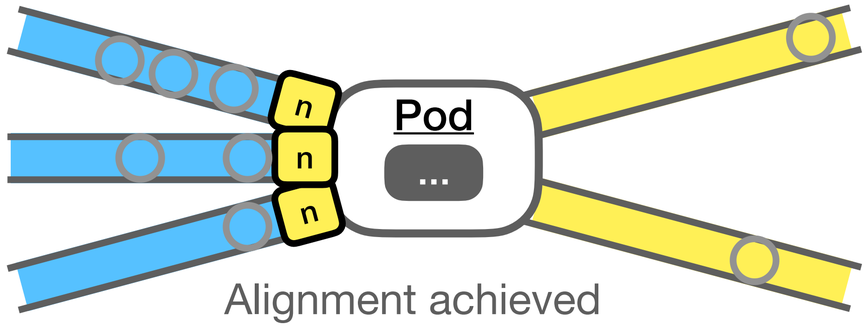
- Subscribe to input streams, execut
  generate output streams
- Snapshotted consistent state is ext

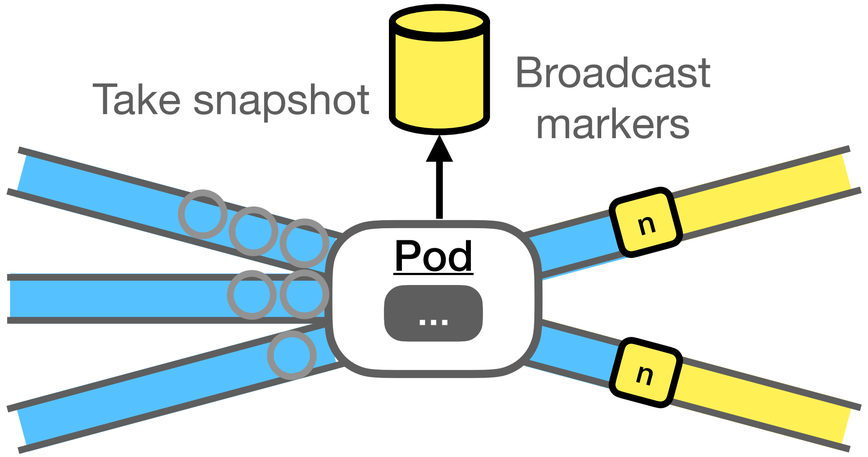
! More details in the paper

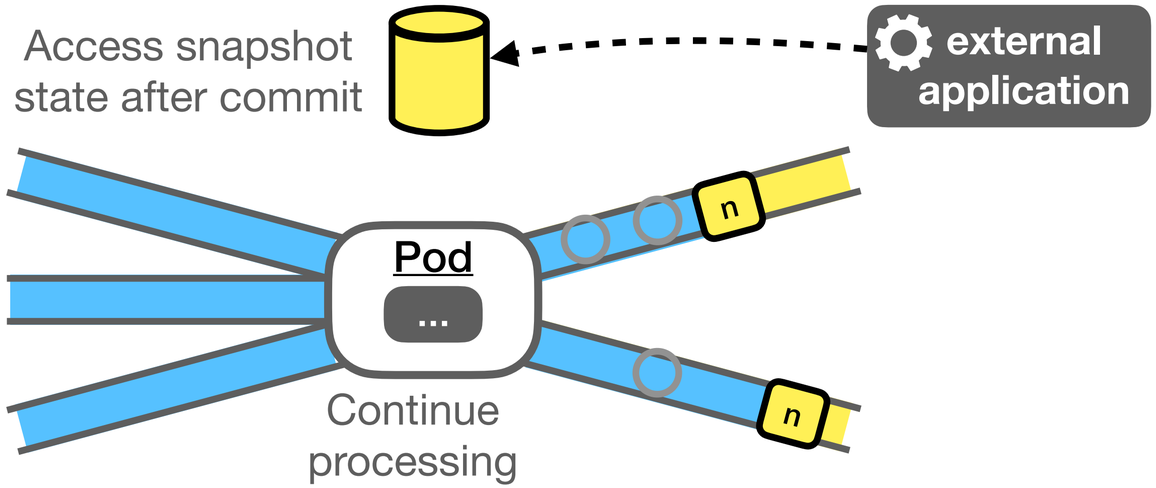


# Causally Consistent Reads from Materialized Views



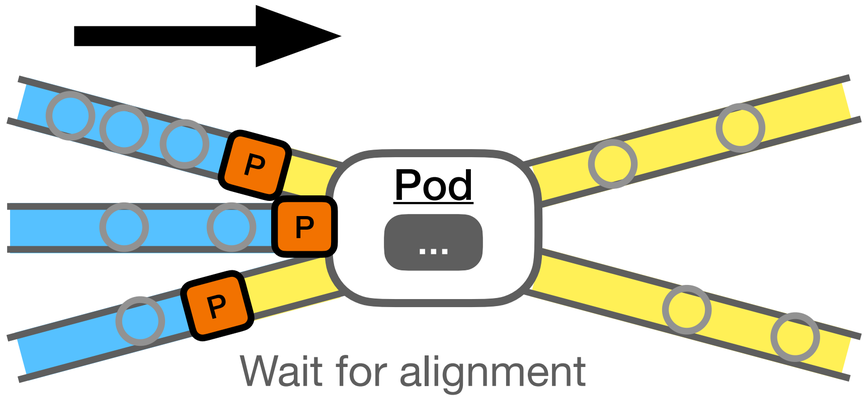



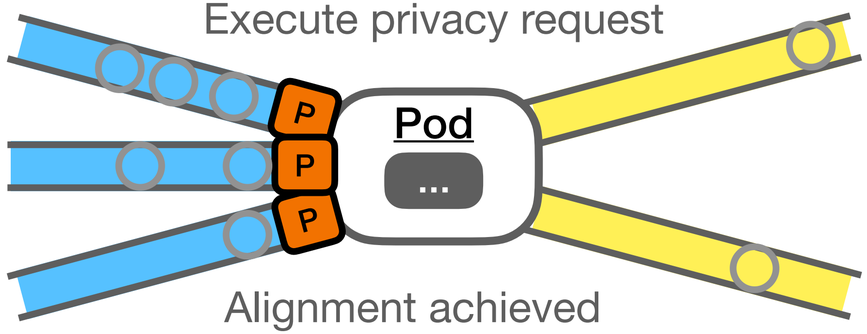

*Using asynchronous epoch commit from [6], reproduced/adapted*

# Serializable privacy requests



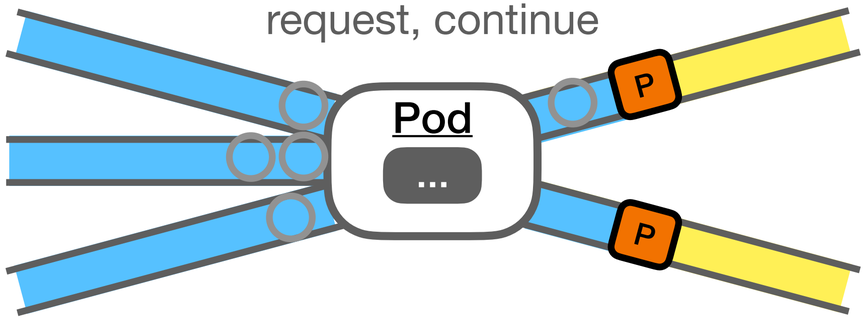

Wait for alignment

Execute privacy request




Alignment achieved

Broadcast privacy request, continue



# Fine-Grained Information Flow

Couple data with policy-metadata ("Data Capsule") [22], use fine-grained information flow [21, 15]

```
Event e1 = <a,   metadata={Alice}>
Event e2 = <b,   metadata={Bob  }>
e1 + e2  = <a+b, metadata={Alice, Bob}>
```

=> Compute the correct privacy policy of derived data

# Pods: Executing privacy requests

**Execution Strategies for Updates:**

- Static dataflow data dependencies, relational operators [16] =>
  1. **Differential updates**
  2. **Recompute state from replaying events**
     - (Ordering does not affect results)

- Fine-grained metadata [22], user-defined functions =>
  3. **Apply request on state**
     - Pro: always works
     - Con: may "erase" state that cannot be recomputed, e.g. joint state

# Outline

1. Problem Scope
2. Pods: Our Approach
3. **Open Questions / Research Directions**

# Open Questions / Research Directions

## Efficient information flow tracking

- Fine-grained information flow => efficiency challenge for aggregate data; look into declassification [14]

## Executing privacy requests

- Semantics of privacy requests are unclear [19]; UDFs; implement full specification may uncover more issues

## Consistent integration with external services

- Propagate privacy requests to external services with atomic consistency

## A more flexible programming model

- Supporting cycles; dynamic deployments; actors; and push-based updates (sources and sinks)

# References


[16] Schwarzkopf, Malte, et al. "Position: Gdpr compliance by construction." Heterogeneous Data Management, Polystores, and Analytics for Healthcare. Springer, Cham, 2019. 39-53.

[11] Gjengset, Jon, et al. "Noria: dynamic, partially-stateful data-flow for high-performance web applications." 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18). 2018.

[22] Wang, Lun, et al. "Data capsule: A new paradigm for automatic compliance with data privacy regulations." Heterogeneous Data Management, Polystores, and Analytics for Healthcare. Springer, Cham, 2019. 3-23.

[19] Stonebraker, Michael, et al. "Poly'19 Workshop Summary: GDPR." ACM SIGMOD Record 49.3 (2020): 55-58.

[6] Carbone, Paris, et al. "State management in Apache Flink®: consistent stateful distributed stream processing." Proceedings of the VLDB Endowment 10.12 (2017): 1718-1729.

[15] Salvaneschi, Guido, et al. "Language-integrated privacy-aware distributed queries." Proceedings of the ACM on Programming Languages OOPSLA (2019): 1-30.

[21] Volpano, Dennis, Cynthia Irvine, and Geoffrey Smith. "A sound type system for secure flow analysis." Journal of computer security 4.2-3 (1996): 167-187.


RI.SE

# WIP: Pods: Privacy Compliant Scalable Decentralized Data Services

**Jonas Spenger[12] <jspenger@kth.se>**, Paris Carbone[12], Philipp Haller[2]

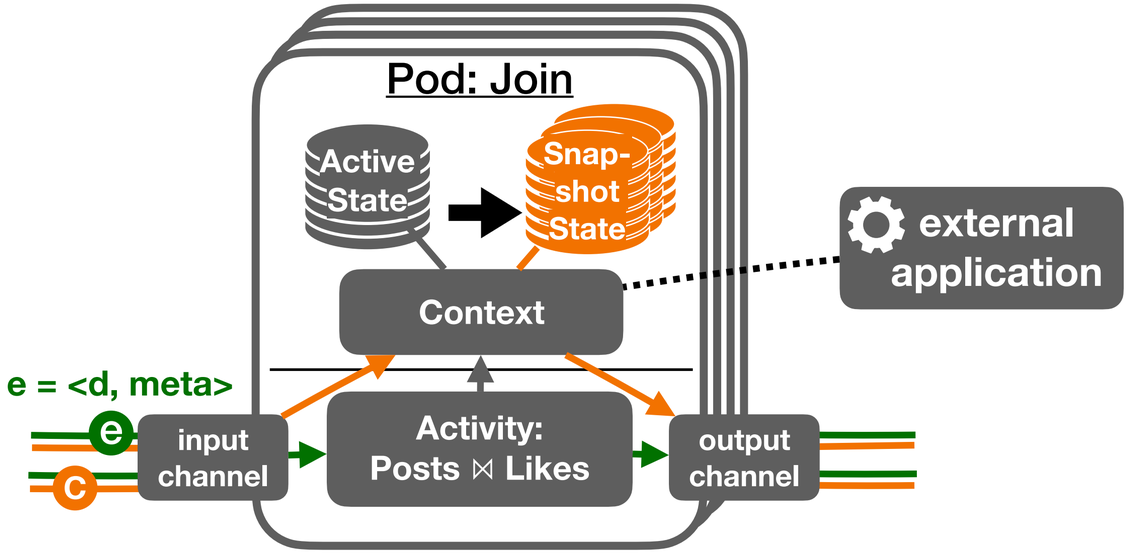
[1]RISE Research Institutes of Sweden, Stockholm, Sweden
[2]Digital Futures and EECS, KTH Royal Institute of Technology, Stockholm, Sweden

Presented at Poly'21: Polystore systems for heterogeneous data in multiple databases with privacy and security assurances, August 20, 2021

# Pod Task

## Pod task

- Dataflow composition, pods connected via channels
- Separate state and logic => serverless, elastic scaling
- Context handles privacy request "transparently"
- Distributed snapshotting => resilience to failures [6]
- External applications access snapshot consistent state
- Fine-grained information flow



# Anatomy of a Pod Task