Reaktive Queries über verteilte Real-Time Data Streams

David Bach
Moritz Moxter

ETH zürich
Systems Group
clockworks

David Bach

david@clockworks.io

Moritz Moxter

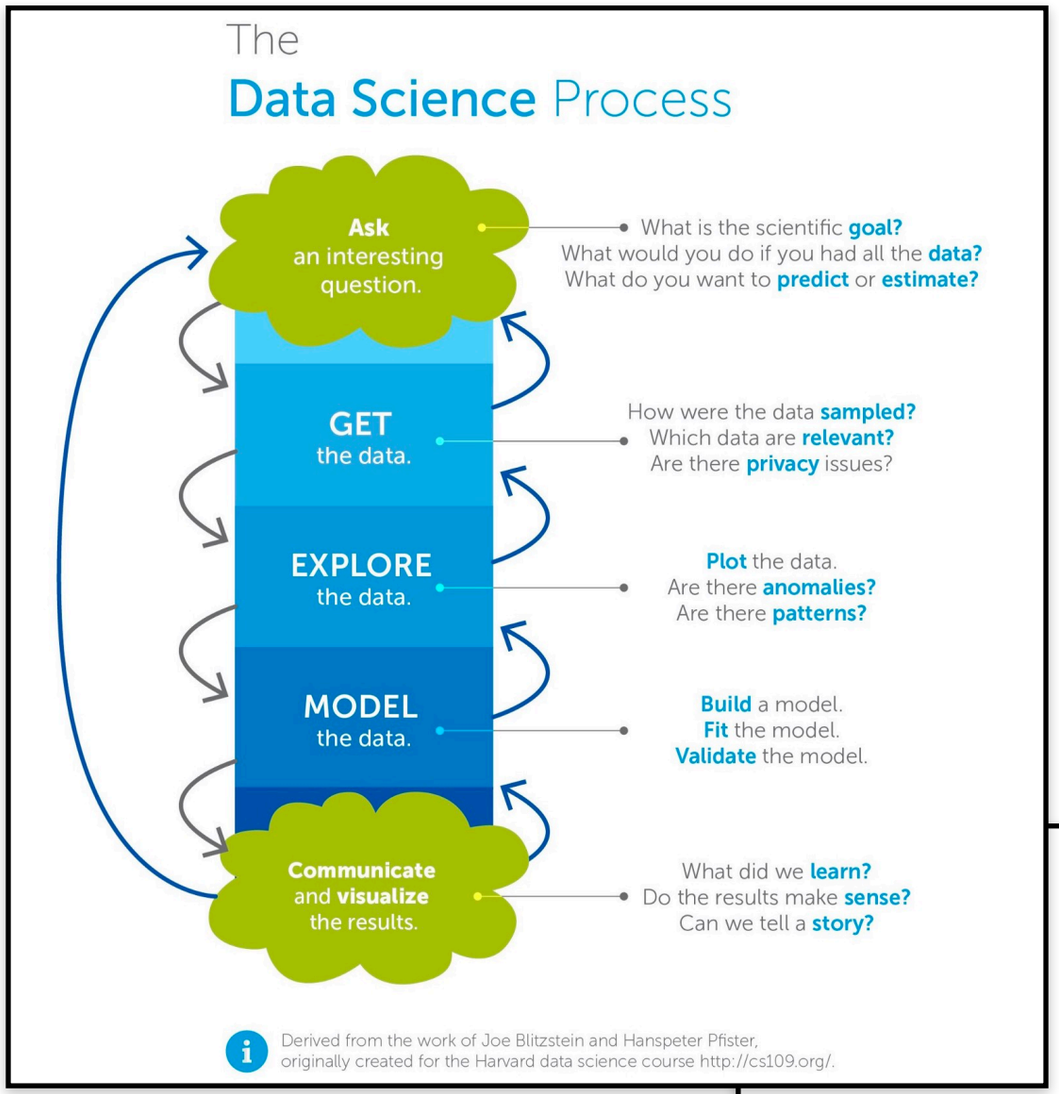
moritz@clockworks.io

in Kollaboration mit:

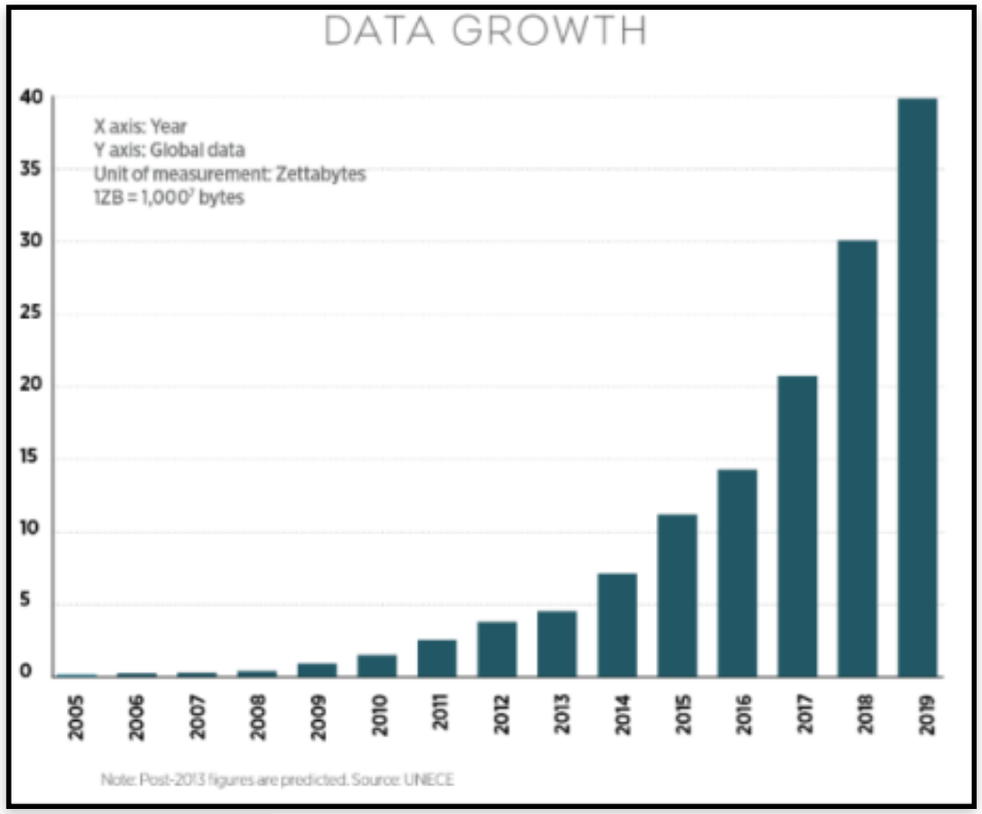
Frank McSherry (ETH)

Nikolas Göbel (Clockworks)

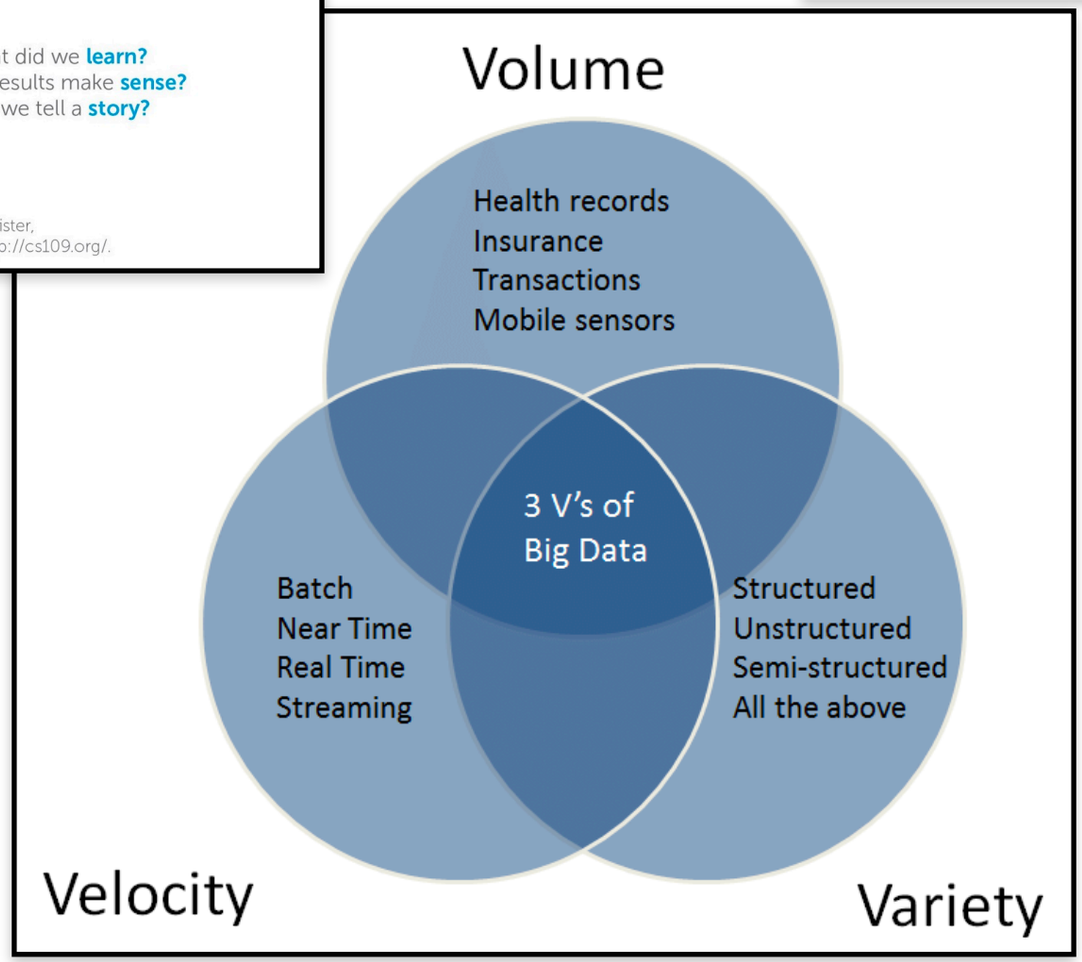
Malte Sandstede (Clockworks)



Typical Data Science Workflow. Source: Harvard Data Science Course - CS109



Data Growth Predictions. Source: UNECE. 2013



Austin, Christopher & Kusumoto, Fred. (2016). The application of Big Data in medicine: current implications and future directions. Journal of interventional cardiac electrophysiology : an international journal of arrhythmias and pacing.

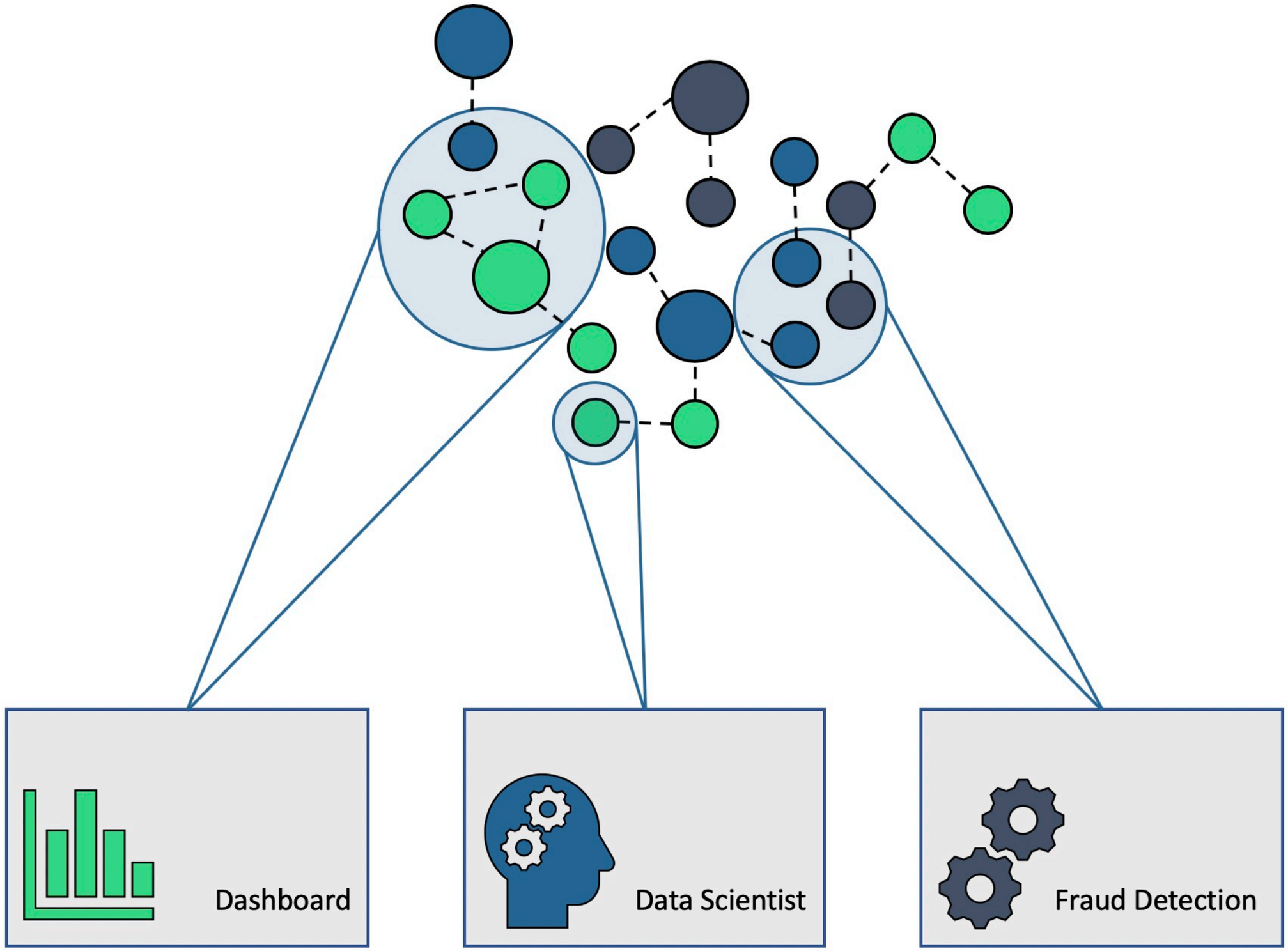
- Wir wollen eine gemeinsame und flexible Sprache finden, um über Informationen in einer Organisation zu sprechen.

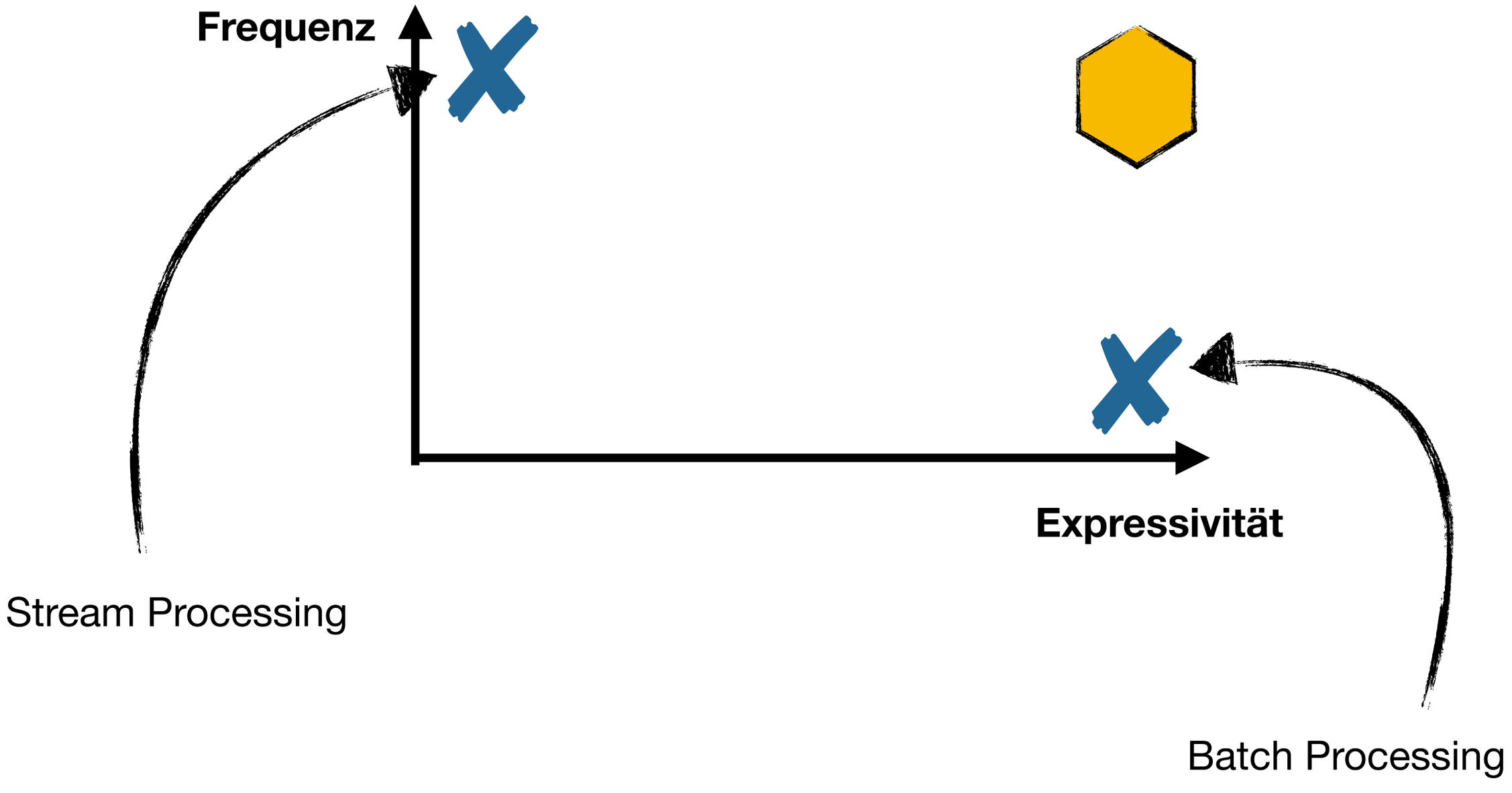
- Wir benötigen leistungsstarke und expressive Algorithmen, um diese Informationen zu analysieren.

- Wir wollen Akteuren eine konsistente und aktuelle Sicht auf ihre Daten bereitstellen.

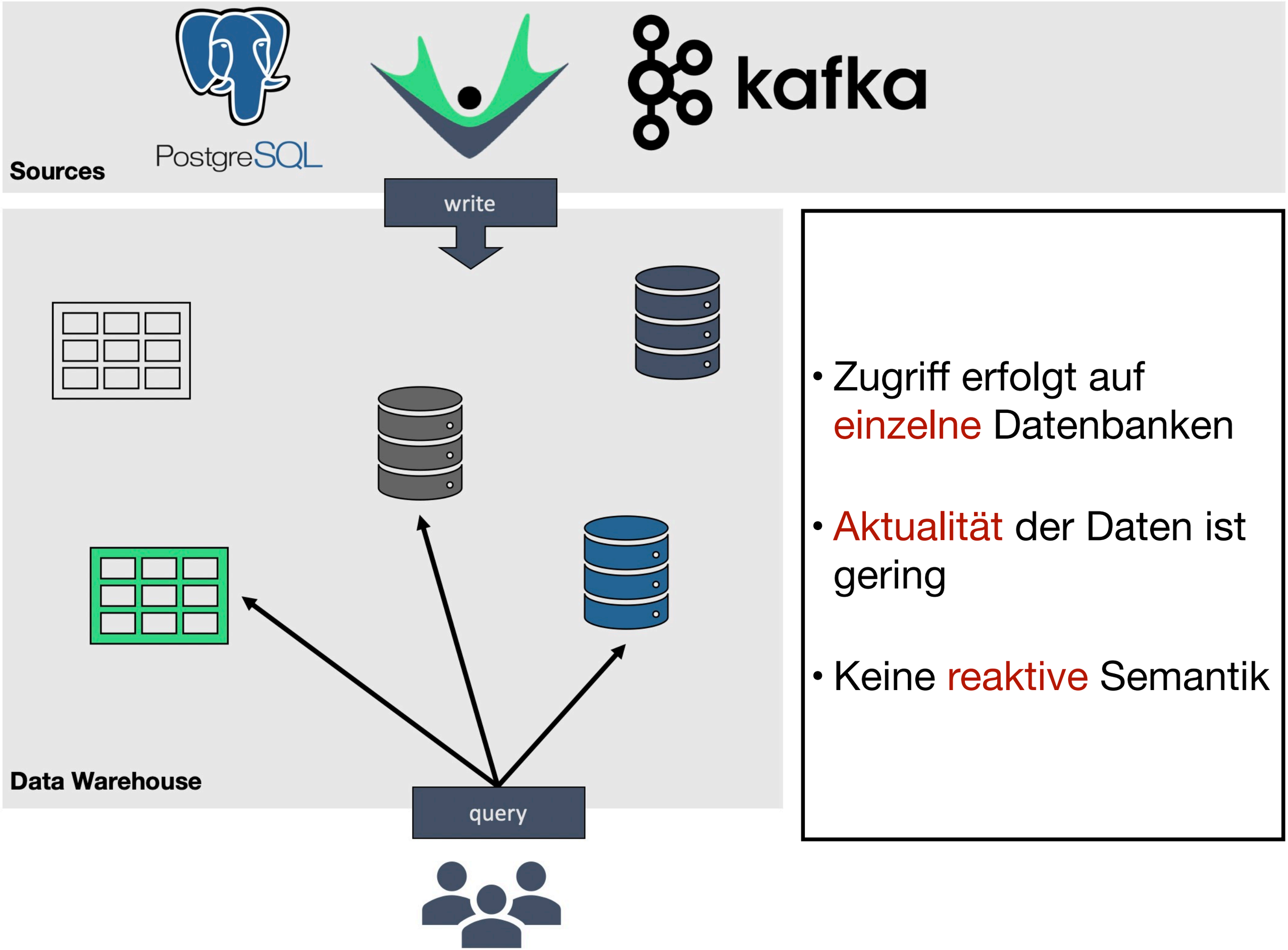
- Wir benötigen verteilte und effiziente Systeme, um diese Daten mit hohem Durchsatz zu verarbeiten.

Unsere Vision

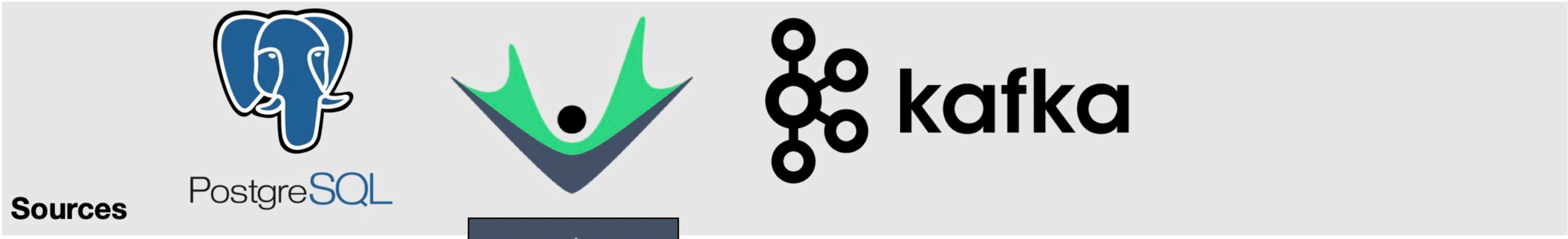

Dashboard
Data Scientist
Fraud Detection


Frequenz
Expressivität
Stream Processing
Batch Processing

Status Quo



- Zugriff erfolgt auf einzelne Datenbanken
- Aktualität der Daten ist gering
- Keine reaktive Semantik



- Limitierung der Expressivität der Algorithmen
- Iterationen und Graph Berechnungen nicht möglich
- Keine historischen Daten

- Komplexe Logik im Consumer
- Verteilen und Skalieren nicht inbegriffen
- Präziser Zugriff schwierig

Batch

- Hohe Expressivität
- Performance-Optimierung

Stream

- Ergebnisse mit hoher Frequenz
- Skalierendes und reaktives Dataflow-Modell

Event

- Einheitlicher Zugriff
- Verknüpfen verschiedener Berechnungen

Warum können wir diese Elemente nicht vereinen?

Warum können wir diese Elemente nicht vereinen?

Das volle Evaluieren von Berechnungen beim Eintreffen neuer Daten ist sehr aufwendig

Differential Computation

Ein Berechnungsmodell, welches selbst komplexe Algorithmen beim Eintreffen neuer Informationen inkrementell aktualisiert

Sources

PostgreSQL

kafka

write

query

query

query

Cluster

query

Dashboard

query

Data Scientist

query

Fraud Detection

3DF (github.com/comnik/declarative-dataflow)

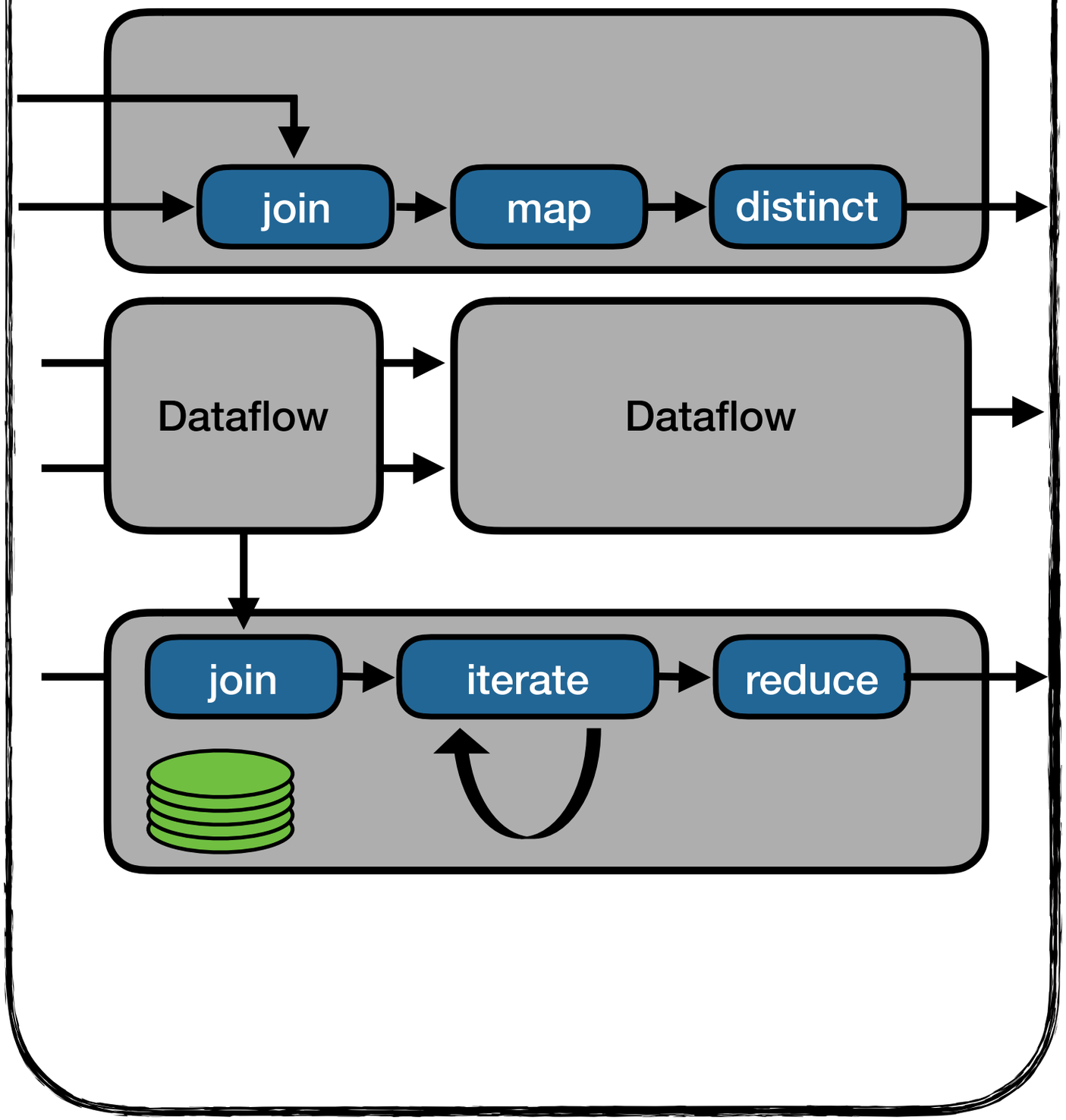
- Reaktive Datalog-Queries github.com/comnik/clj-3df
- Skaliert vom Browser bis zum Cluster
- Differential Computation als Berechnungsmodell

Client

```
[:find ?id
 :where
 [_ :building/id ?id]
 [_ :sensor/failure ?failure]
 [_ :backup/system "dead"]
 (> (count ?failure) 10)]


[:find (sum ?payload)
 :where
 [_ :shipment/payload payload?]
 [(completed? ?payload)]]
```

3DF

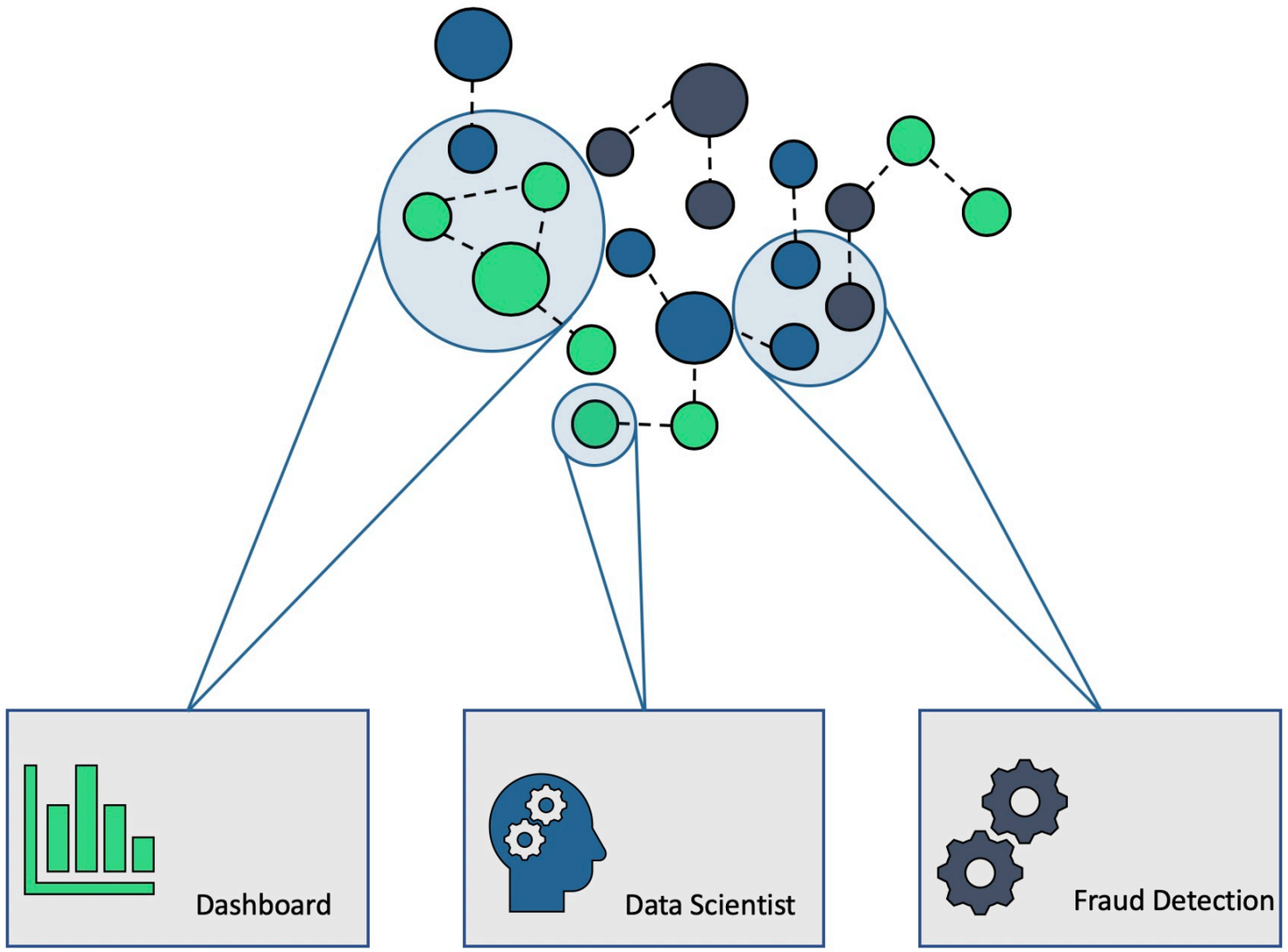


Differential Computation

Präzise Zeit-Semantik

Dataflow-Modell

Deklarativer Zugriff



Differential Computation

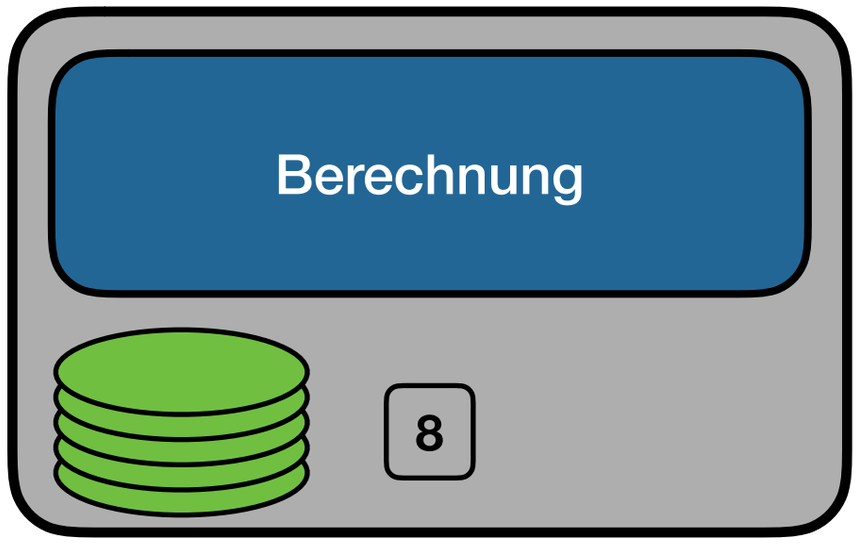
effizient auf neue Informationen reagieren

Präzise Zeit-Semantik

Dataflow-Modell

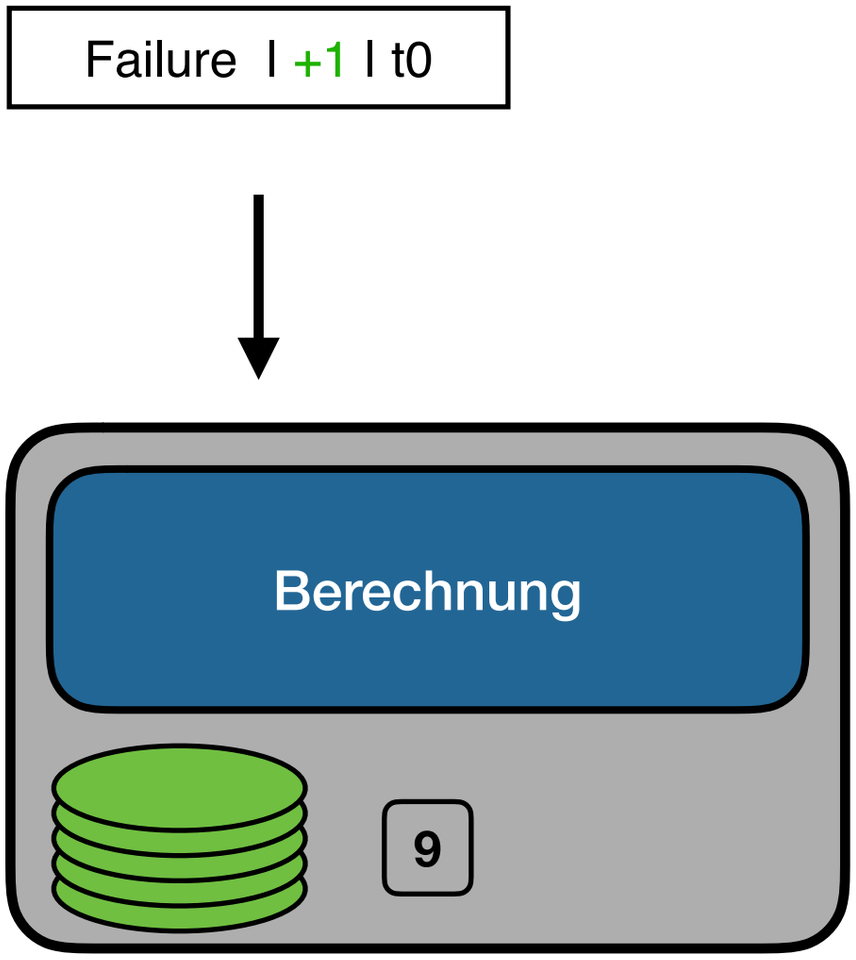
Deklarativer Zugriff

Differential Computation



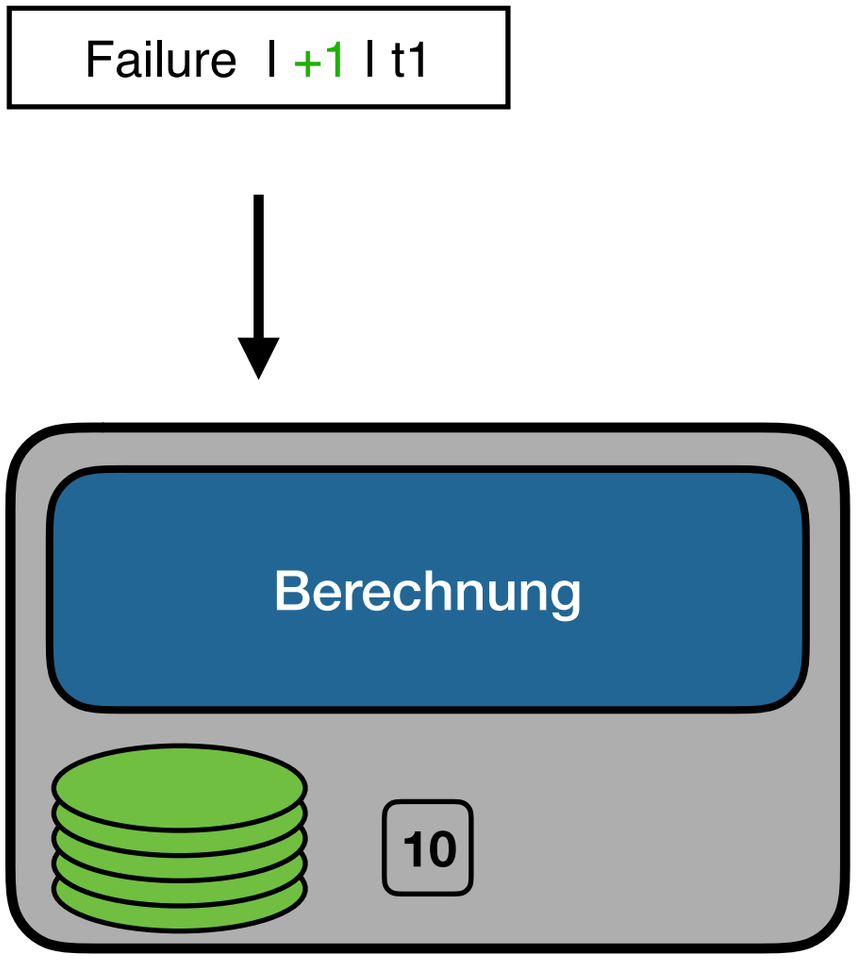

```
[:find ?id
 :where
 [_ :building/id ?id]
 [_ :sensor/failure ?failure]
 [_ :backup/system "dead"]
 (> (count ?failure) 10)]
```

Differential Computation



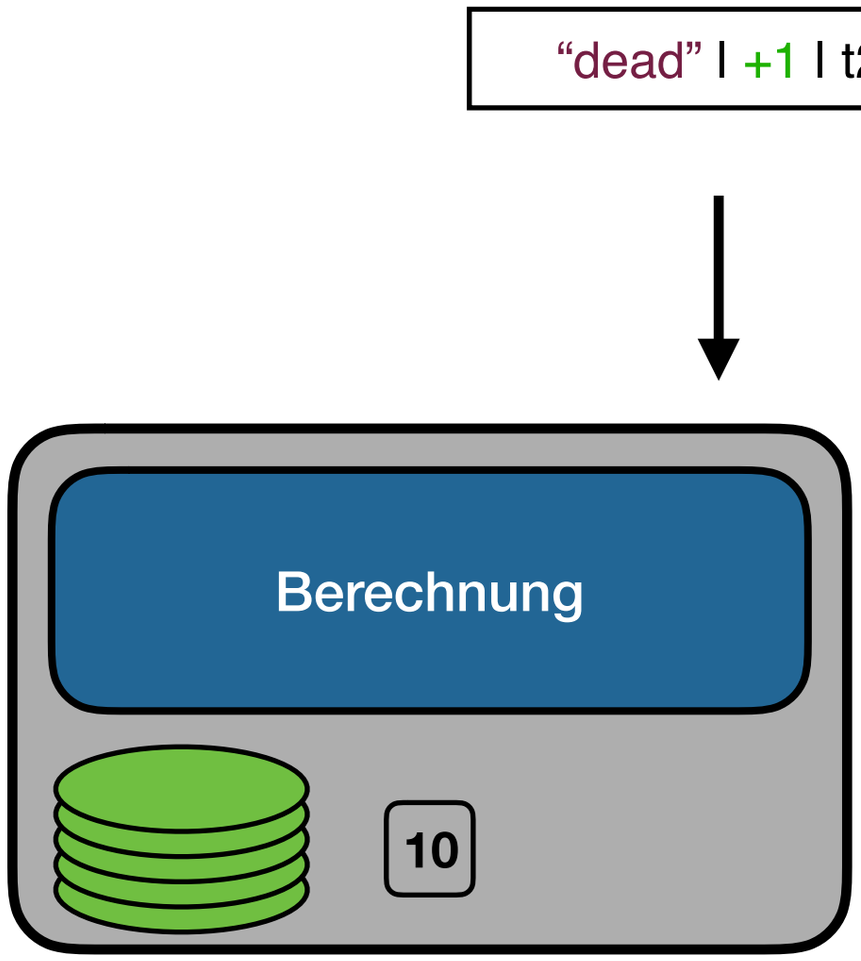

```
[:find ?id
 :where
 [_ :building/id ?id]
 [_ :sensor/failure ?failure]
 [_ :backup/system "dead"]
 (> (count ?failure) 10)]
```

Differential Computation



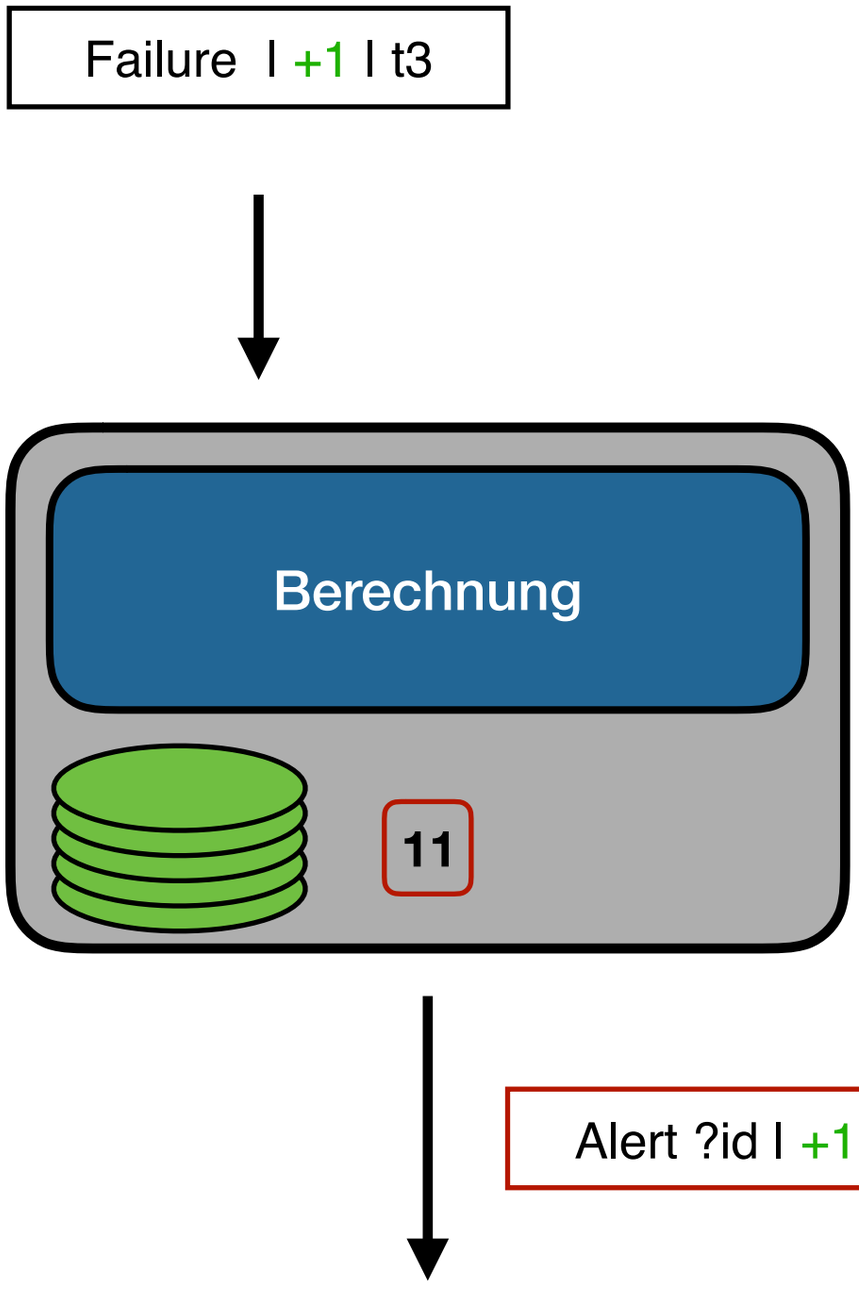

```
[:find ?id
 :where
 [_ :building/id ?id]
 [_ :sensor/failure ?failure]
 [_ :backup/system "dead"]
 (> (count ?failure) 10)]
```

Differential Computation



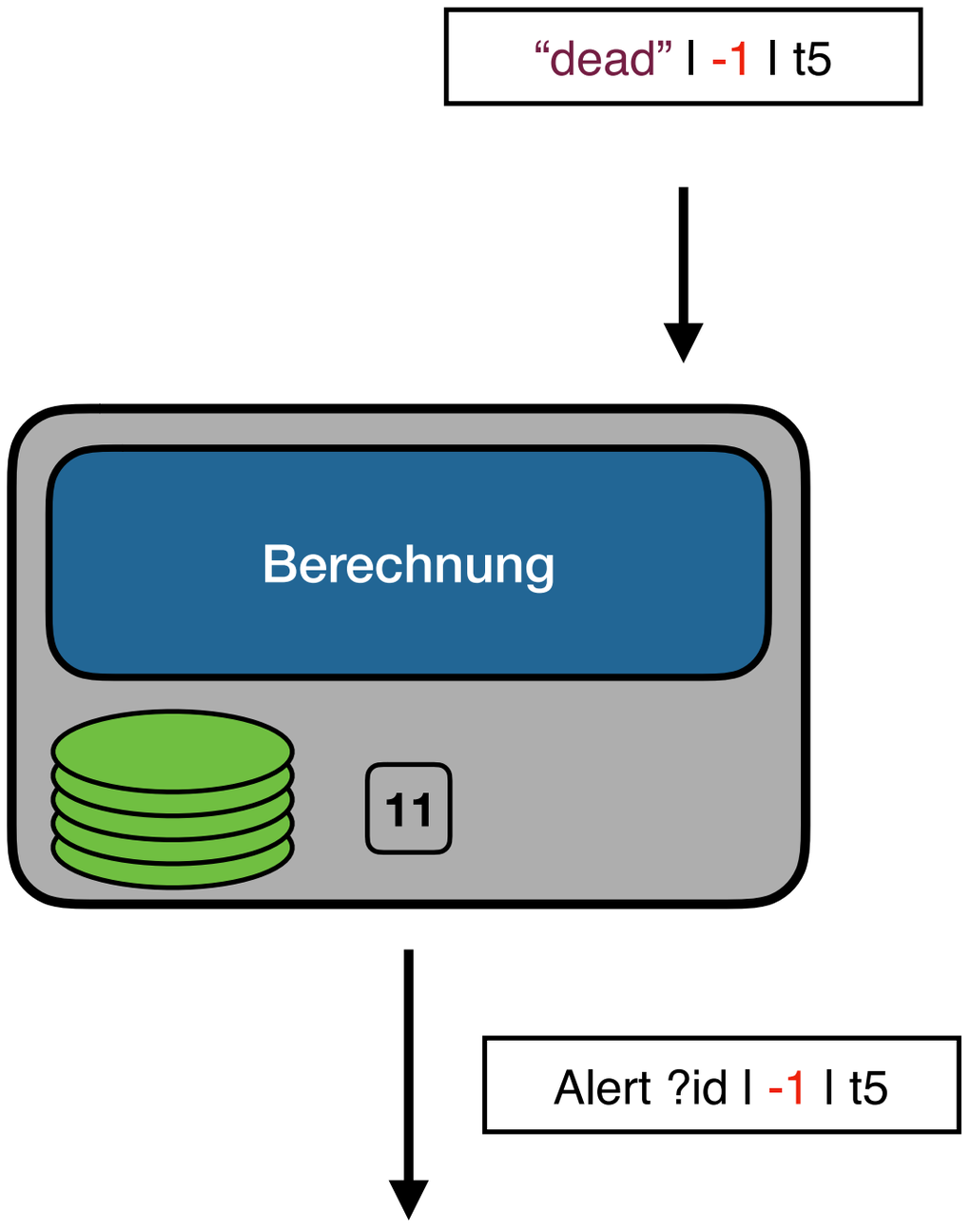

```
[:find ?id
 :where
 [_ :building/id ?id]
 [_ :sensor/failure ?failure]
 [_ :backup/system "dead"]
 (> (count ?failure) 10)]
```

Differential Computation




```
[:find ?id
 :where
 [_ :building/id ?id]
 [_ :sensor/failure ?failure]
 [_ :backup/system "dead"]
 (> (count ?failure) 10)]
```

Differential Computation

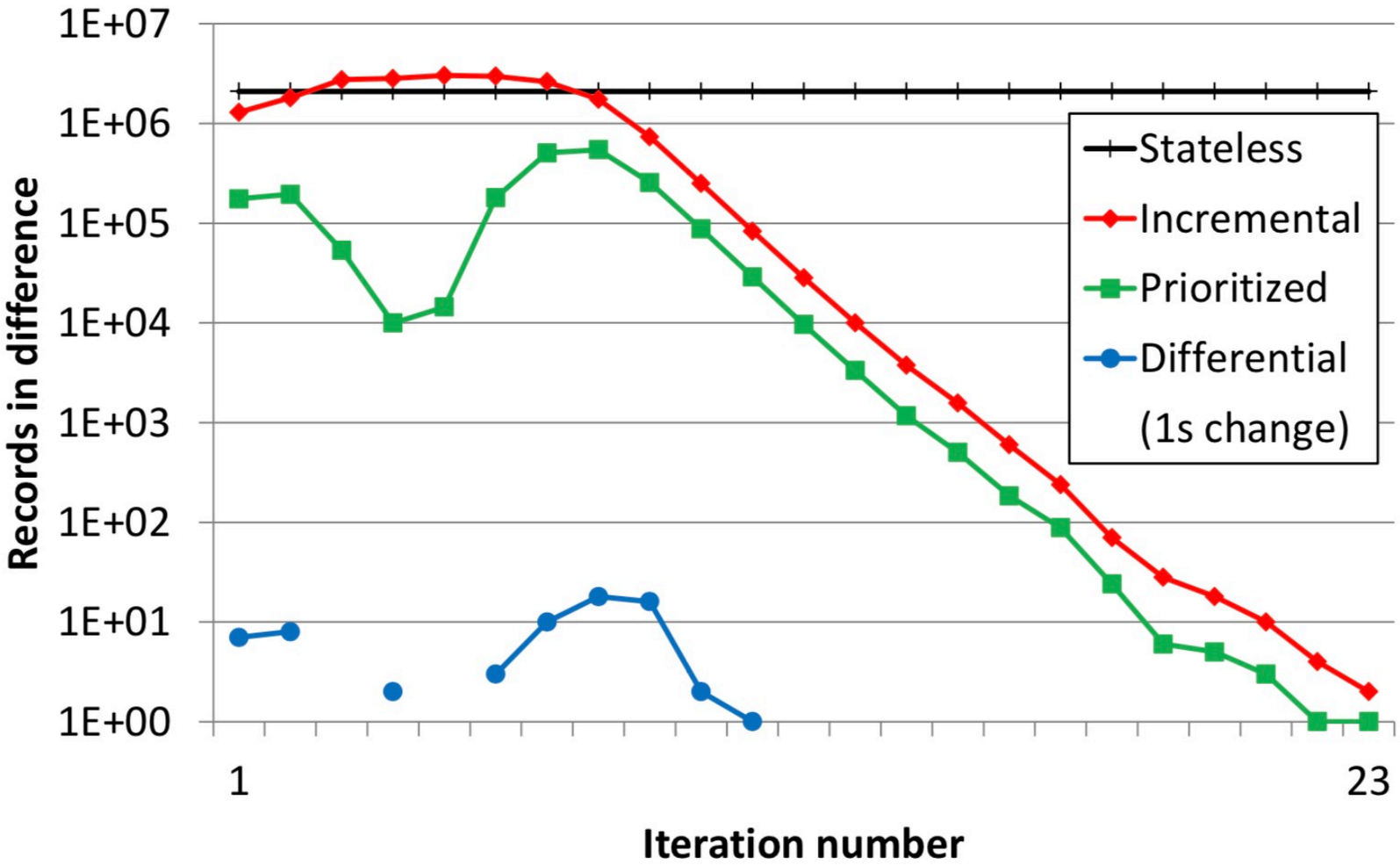



```
[:find ?id
 :where
 [_ :building/id ?id]
 [_ :sensor/failure ?failure]
 [_ :backup/system "dead"]
 (> (count ?failure) 10)]
```

Operatoren erwarten Deltas und produzieren Deltas

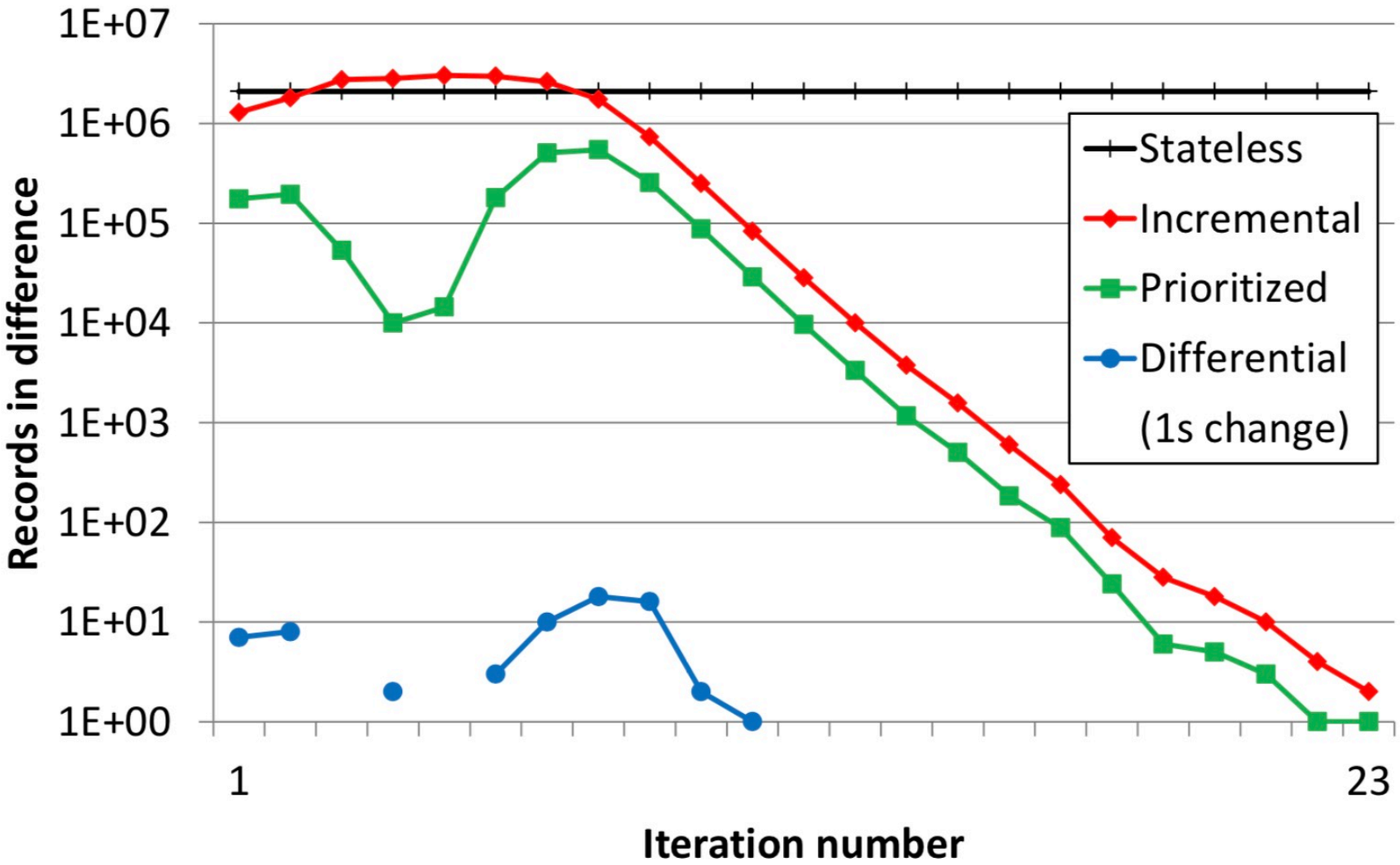
Der Arbeitsaufwand jeden Updates ist proportional zur Differenz und nicht zur Gesamtgröße des Datensets

Differential Computation



Mcsherry, F., Murray, D. G., Isaacs, R., & Isard, M., Differential dataflow

Differential Computation



Computation	Full	Update
gnp1	9.45s	18.29ms

Differential Computation

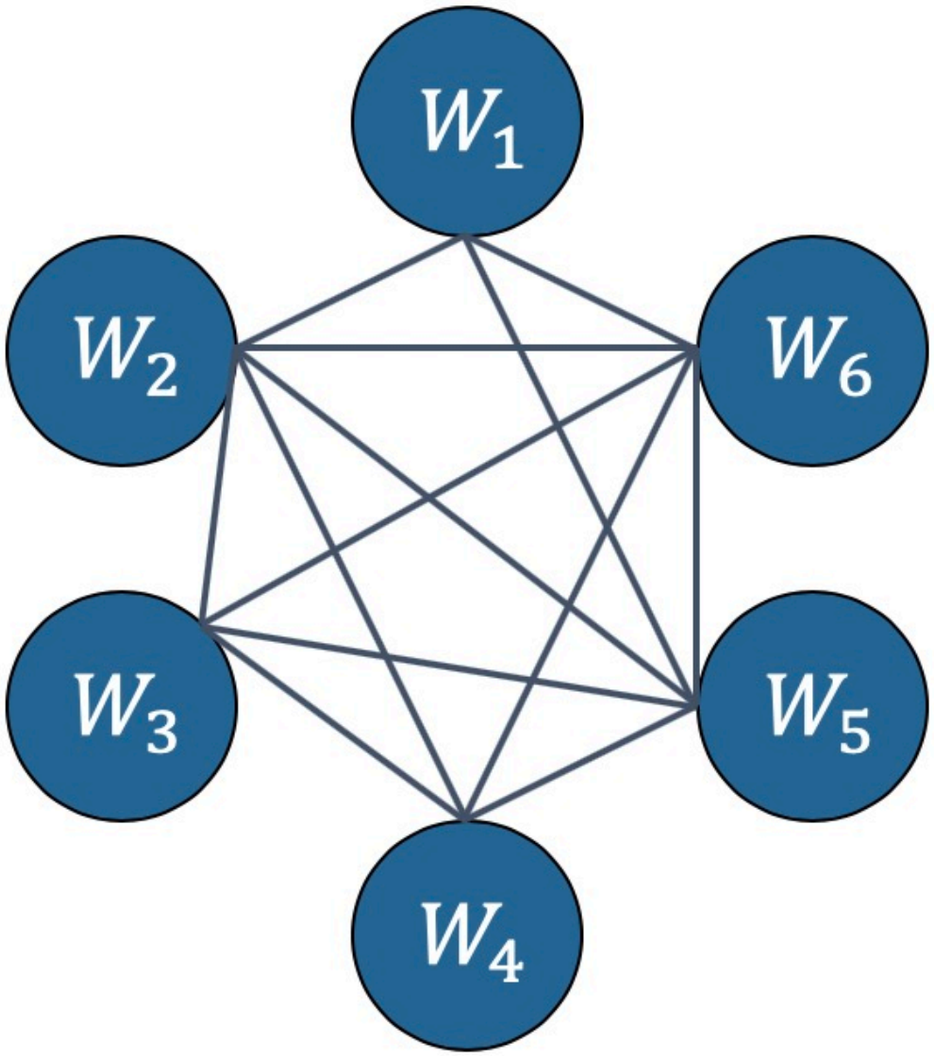
Präzise Zeit-Semantik

konsistente Ergebnisse liefern

Dataflow-Modell

Deklarativer Zugriff

Präzise Zeit-Semantik



- Starke Konsistenzgarantien
- Progress Tracking

Differential Computation

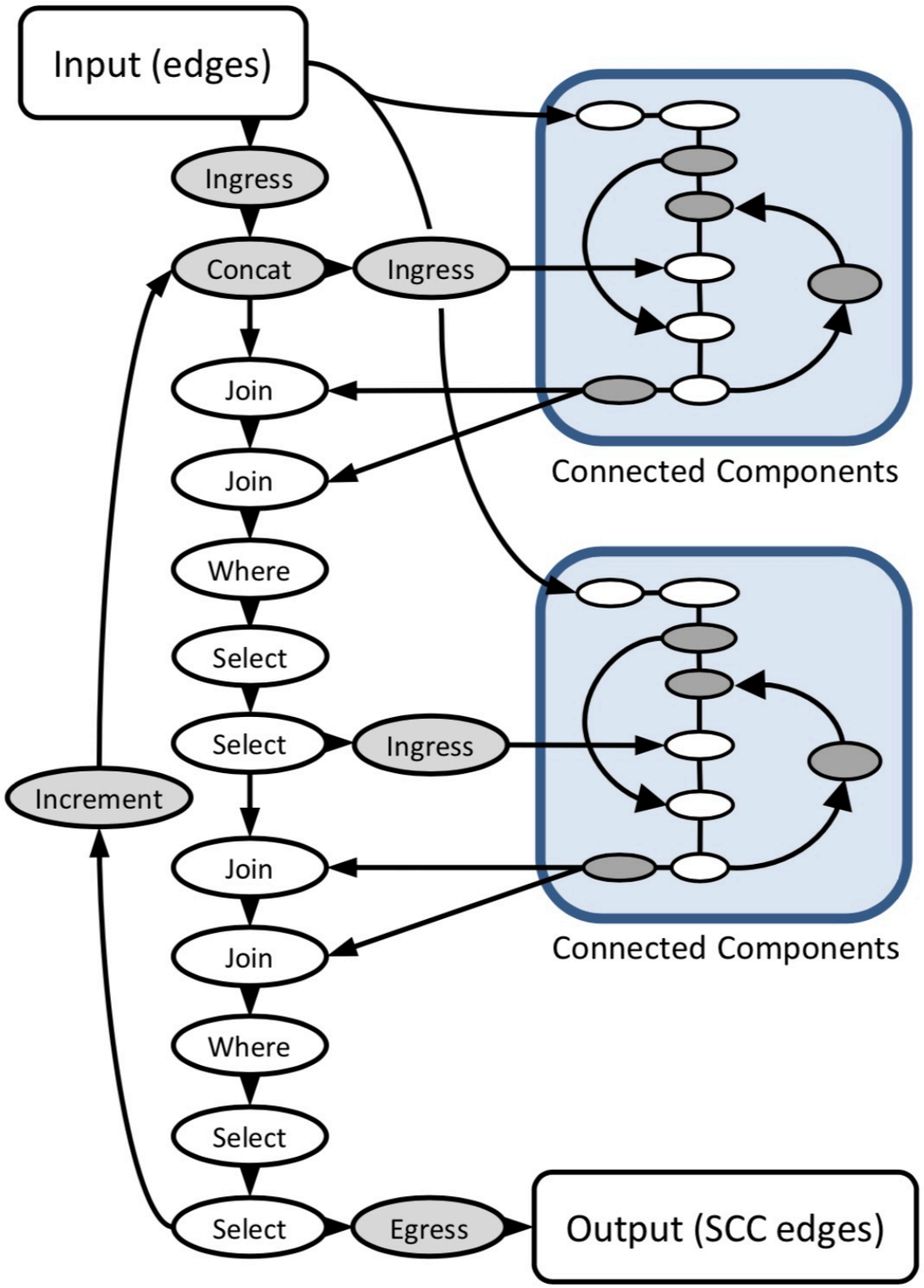
Präzise Zeit-Semantik

Dataflow-Modell

verteilt und iterativ operieren

Deklarativer Zugriff

Dataflow-Modell



- Iterative Algorithmen über Streams
- Temporale Joins

Mcsherry, F., Murray, D. G., Isaacs, R., & Isard, M., Differential dataflow

Differential Computation

Präzise Zeit-Semantik

Dataflow-Modell

Deklarativer Zugriff

gemeinsame Schnittstelle bieten

Deklarativer Zugriff

```
[:find ?id
 :where
 [_ :building/id ?id]
 [_ :sensor/failure ?failure]
 [_ :backup/system "dead"]
 (> (count ?failure) 10)]
```

Anomalie-Detektion

```
[(reach ?a ?b)
 [_ :edge/from ?a]
 [_ :edge/to ?b]]

[(reach ?a ?b)
 [_ :edge/from ?a]
 [_ :edge/to ?middle]
 (reach ?middle ?b)]
```

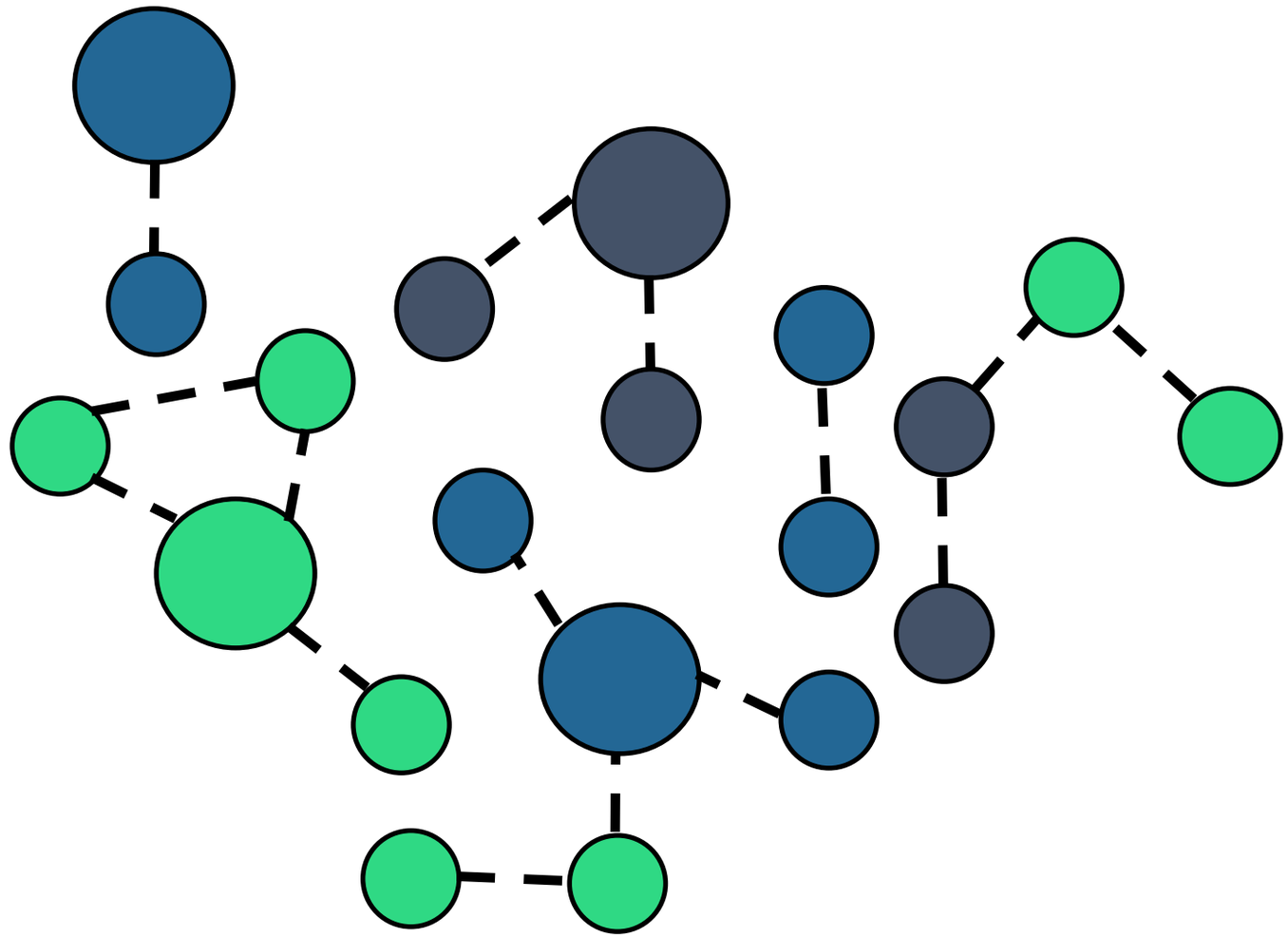
Recursive Reachability

eine gemeinsame und flexible Sprache

:sensor/location
:sensor/timestamp
:sensor/temperature
:sensor/voltage

:station/location
:station/timestamp
:station/temperature
:station/rain

:configuration/sensor
:configuration/settings



gemeinsame und flexible Sprache

leistungsstarke und expressive Algorithmen

konsistente und aktuelle Sicht

verteilte und effiziente Systeme

gemeinsame und flexible Sprache

- ✓ deklarativ und transparent
- ✓ Daten und Berechnungen werden einheitlich abgebildet

leistungsstarke und expressive Algorithmen

konsistente und aktuelle Sicht

verteilte und effiziente Systeme

gemeinsame und flexible Sprache

leistungsstarke und expressive Algorithmen

- ✓ Iteration und Rekursion
- ✓ Verknüpfen von Operationen

konsistente und aktuelle Sicht

verteilte und effiziente Systeme

gemeinsame und flexible Sprache

leistungsstarke und expressive Algorithmen

konsistente und aktuelle Sicht

- ✓ Starke Konsistenzgarantien
- ✓ Propagation von Resultaten durch das gesamte System

verteilte und effiziente Systeme

gemeinsame und flexible Sprache

leistungsstarke und expressive Algorithmen

konsistente und aktuelle Sicht

verteilte und effiziente Systeme

- ✓ Dataflow-Modell
- ✓ Differential Computation

gemeinsame und flexible Sprache

leistungsstarke und expressive Algorithmen

konsistente und aktuelle Sicht

verteilte und effiziente Systeme

Reaktive Queries über verteilte Real-Time Data Streams

@bachdavi | david@clockworks.io

clockworks.io

github.com/comnik/declarative-dataflow