



**Event-Driven
vs
Event Sourcing**



Cédric Pontet

CTO @ Agile Partner, Founder @ #play14, Agile Coach, Software & Cloud Architect, DDDer, EventStormer, and Sketchnoter

Luxembourg · [Contact info](#)

[2,915 followers](#) · [500+ connections](#)



Agile Partner S.A.



Université Henri Poincaré
(Nancy I)

Learning by playing



SHALL WE PLAY A GAME?

I will need 4 volunteers





Event-Driven

Focused on reacting to events in real-time

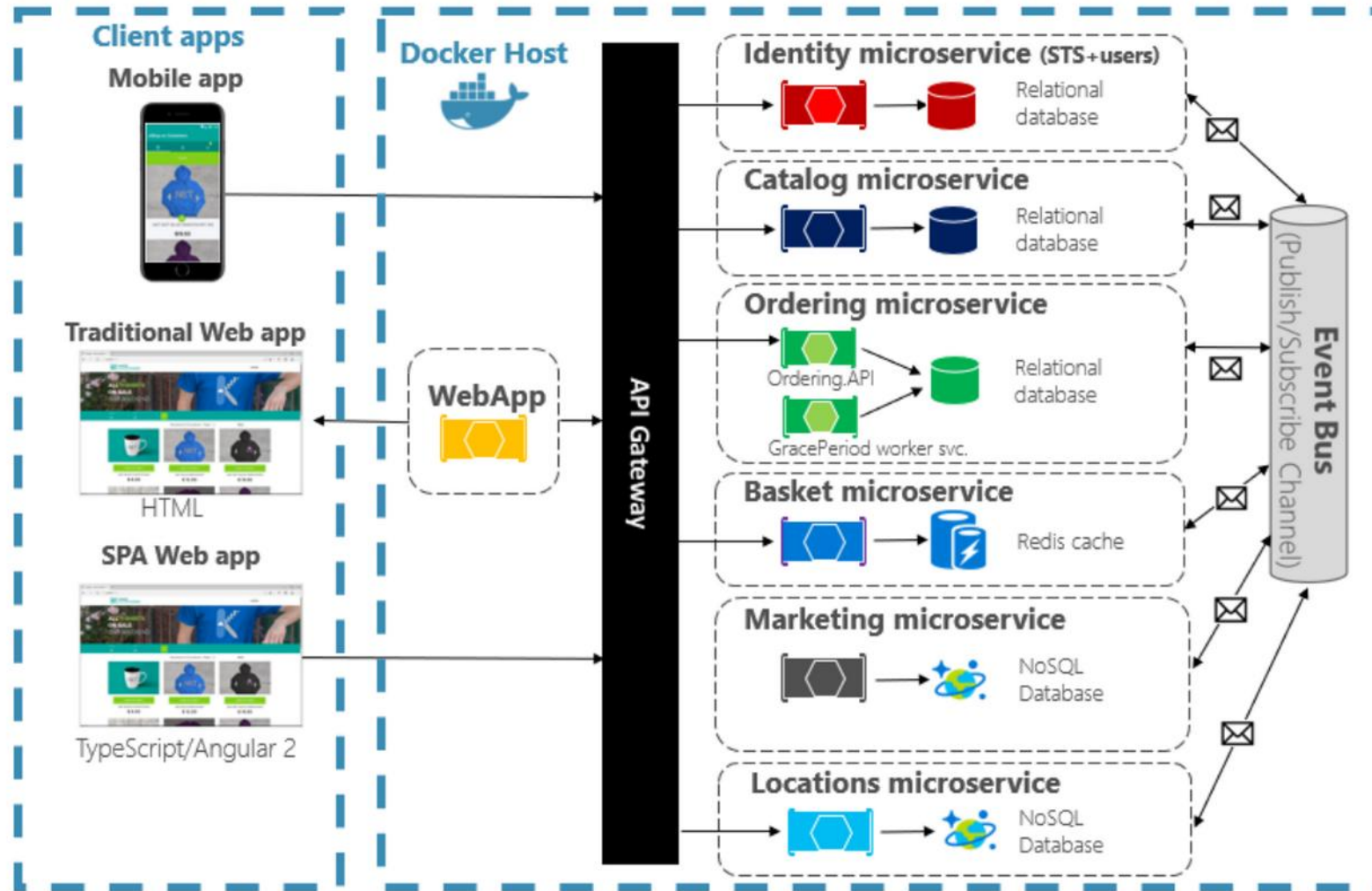


Event Sourcing

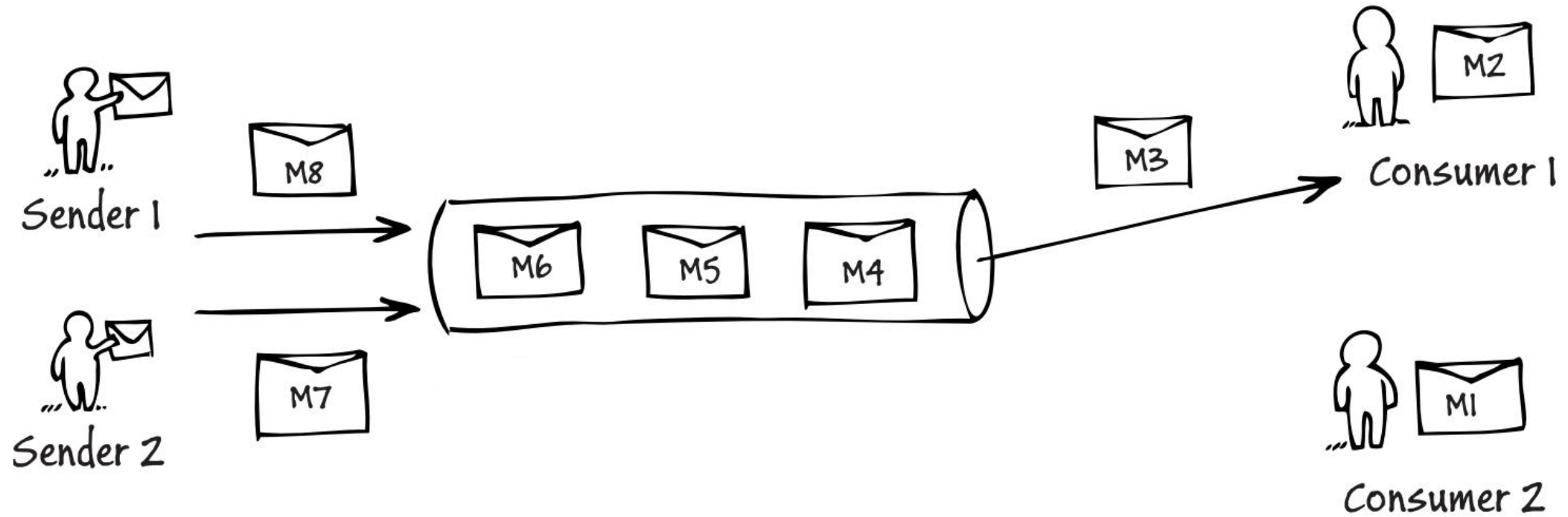
Persisting state changes as a sequence of events



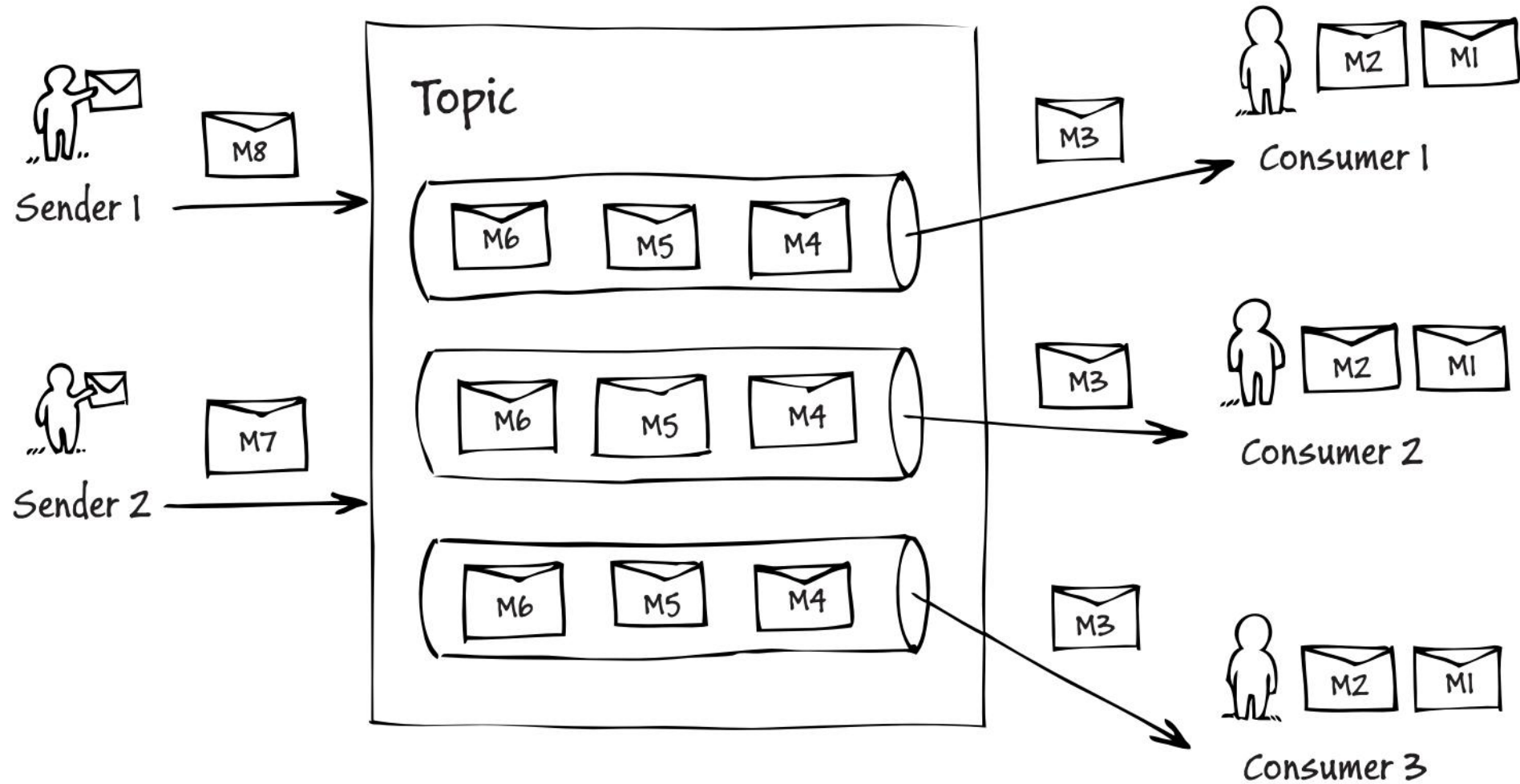
Events with microservices



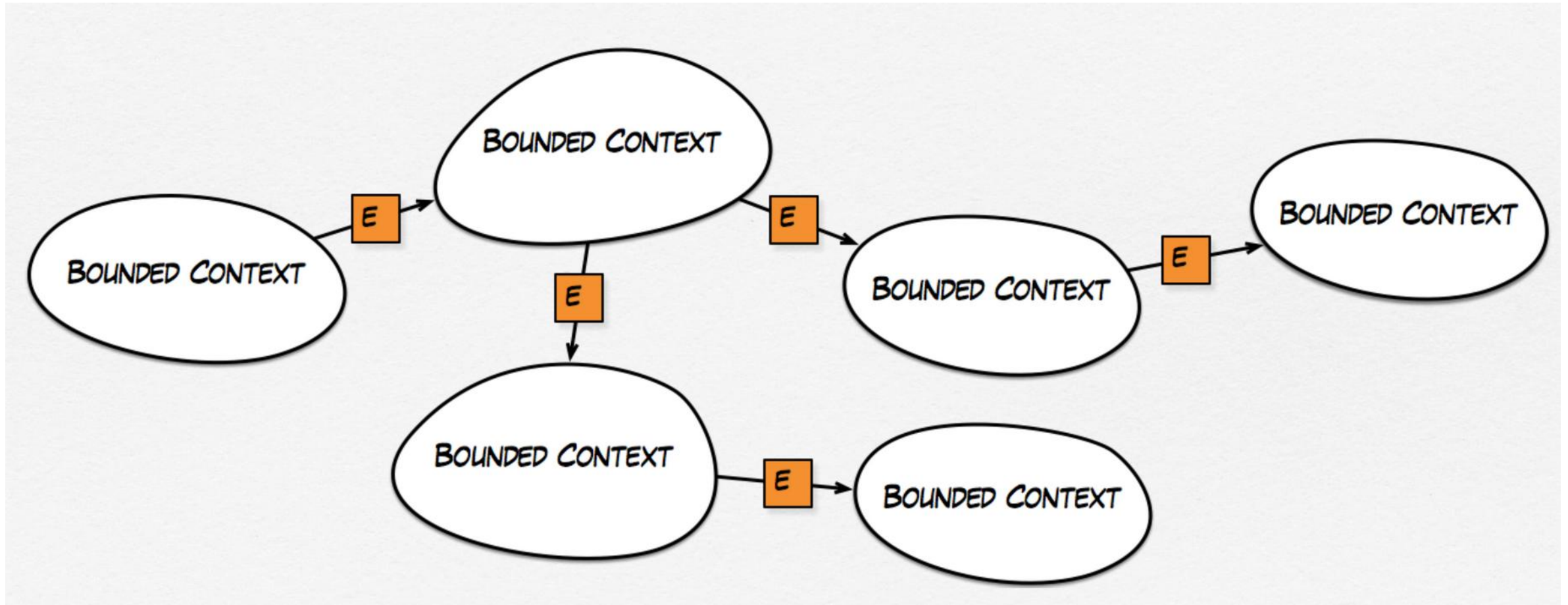
Queue



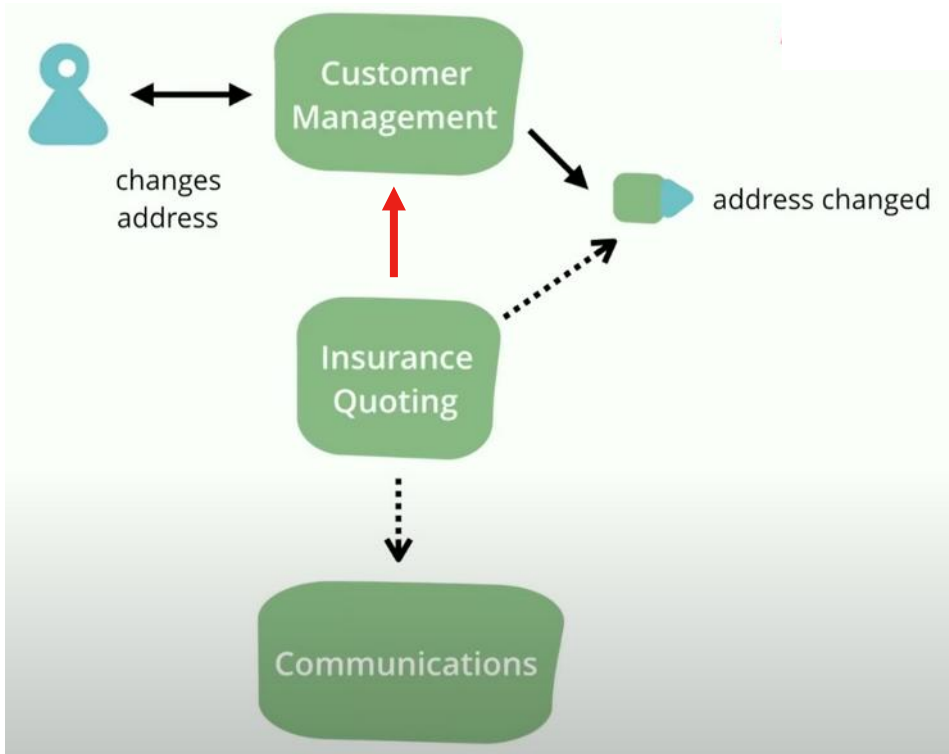
Topic (pub/sub)



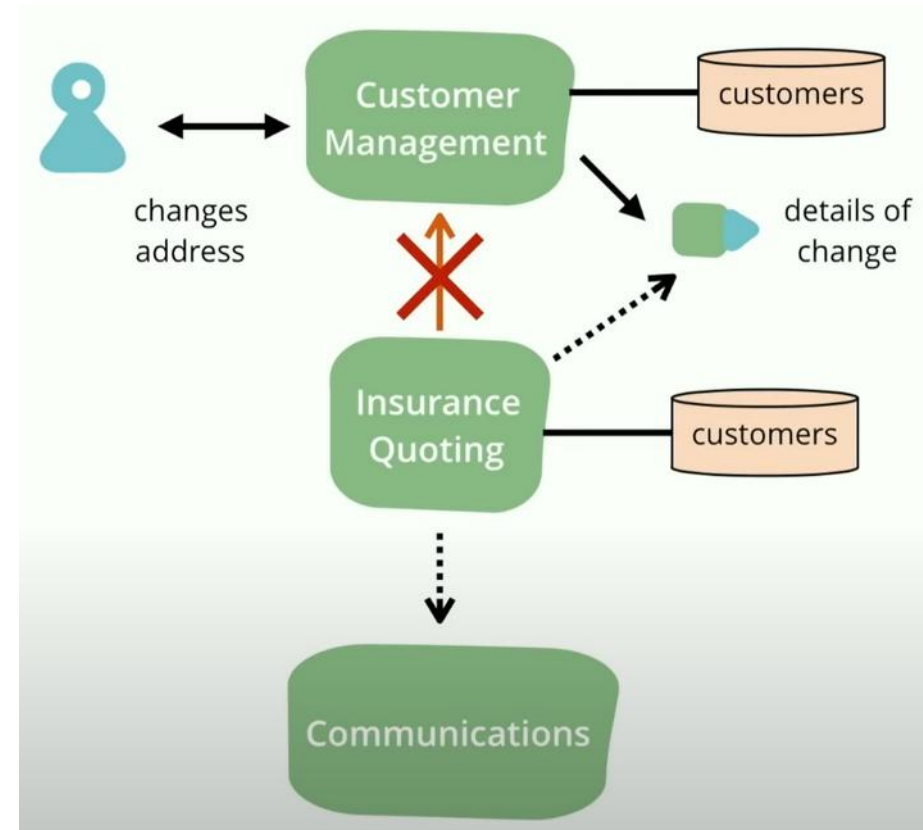
Event-Driven



Styles & patterns

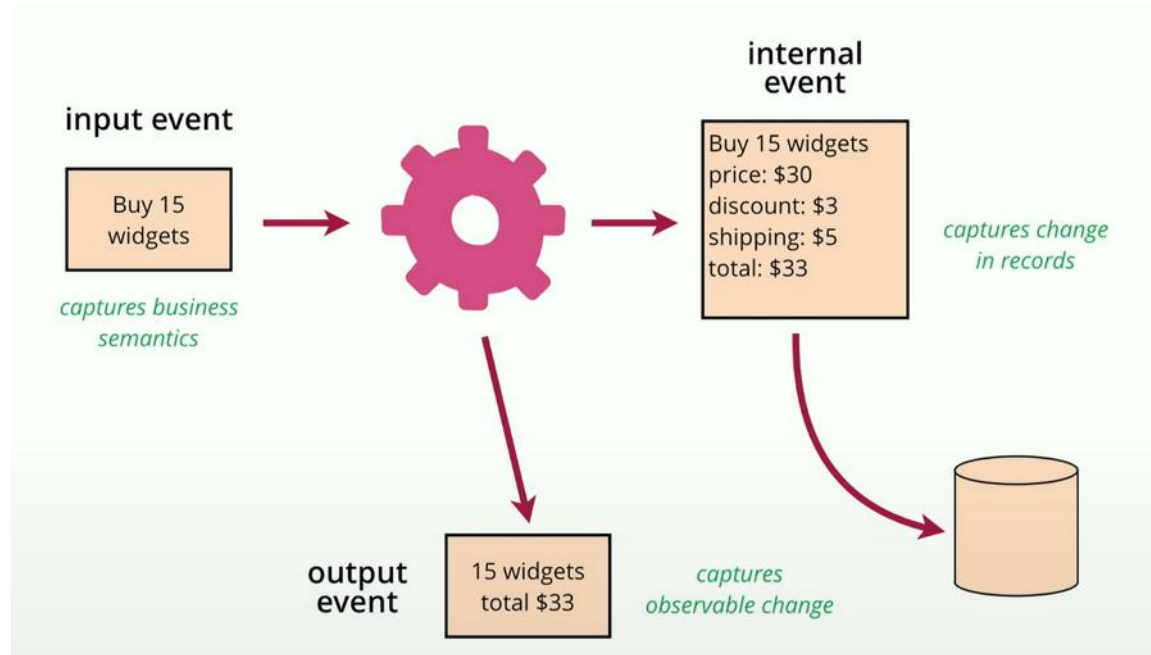


Event Notification

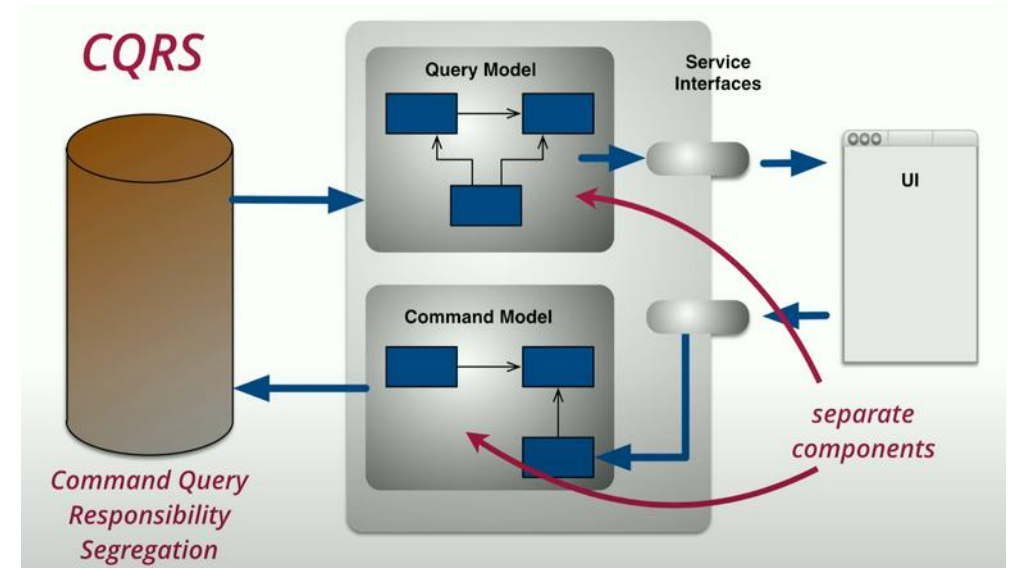


Event-carried State Transfer

Styles & patterns

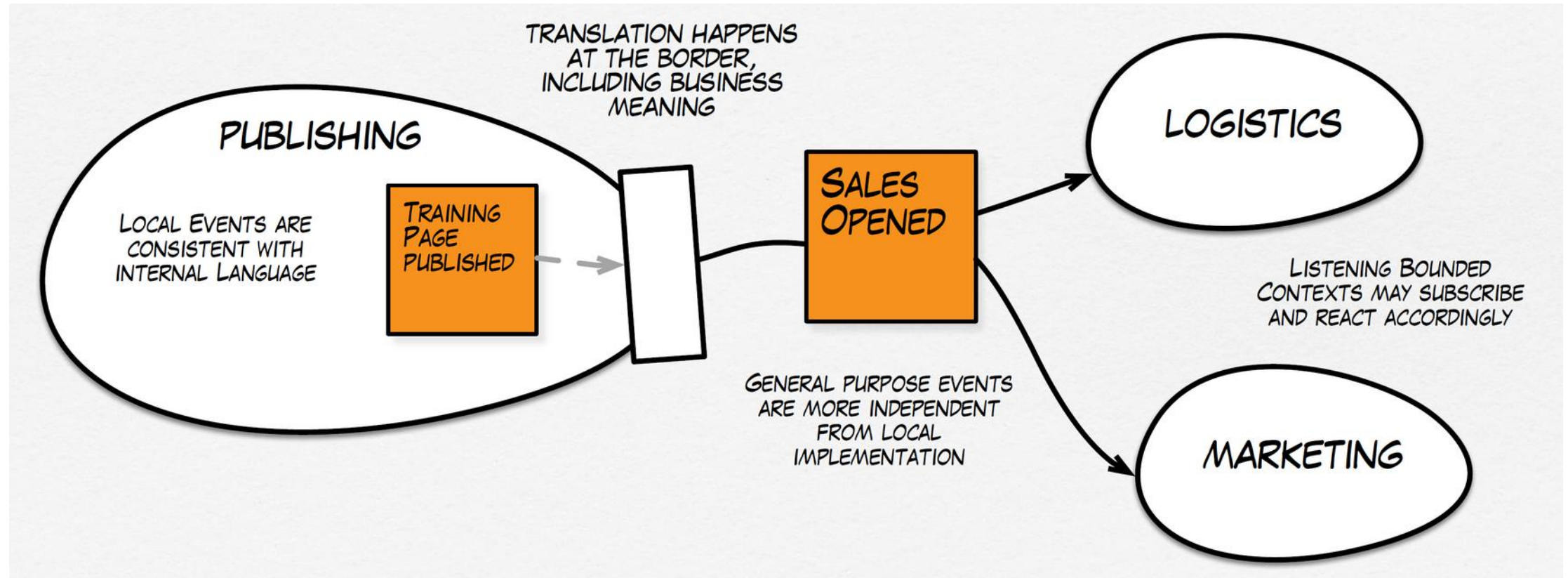


Event Sourcing



CQRS

Domain events vs Integration events



**Have you ever designed an
event sourced system?**

**Have you ever used an
event sourced system?**

Accounting



git

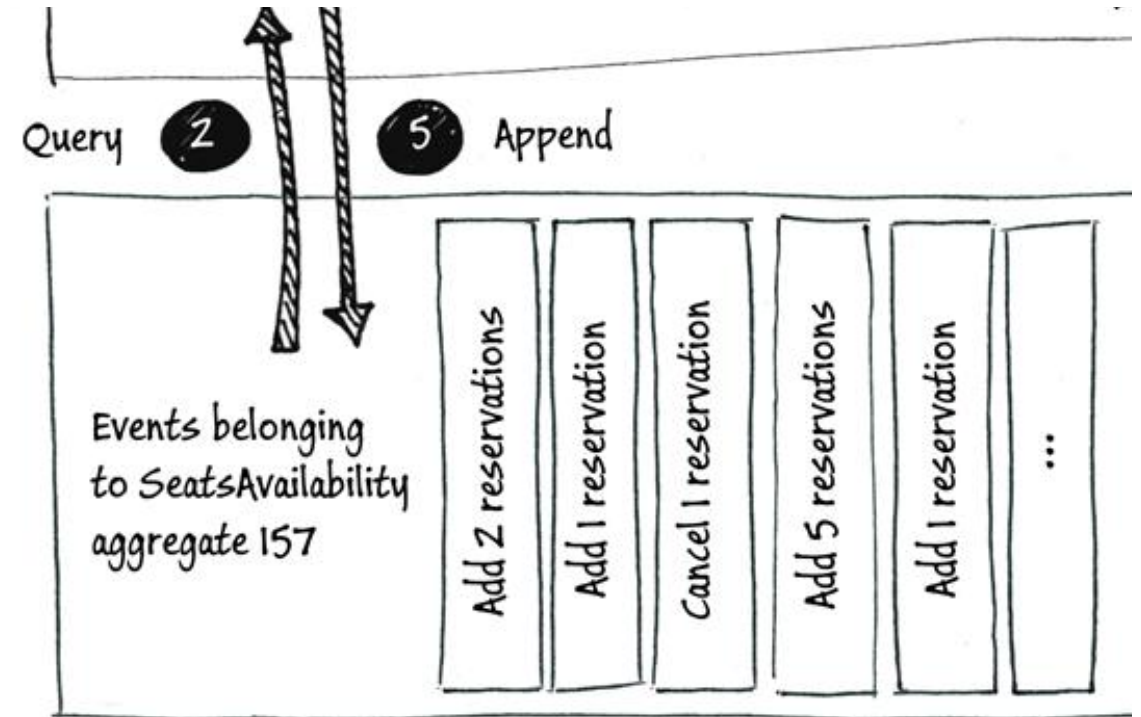


Post

LUXEMBOURG

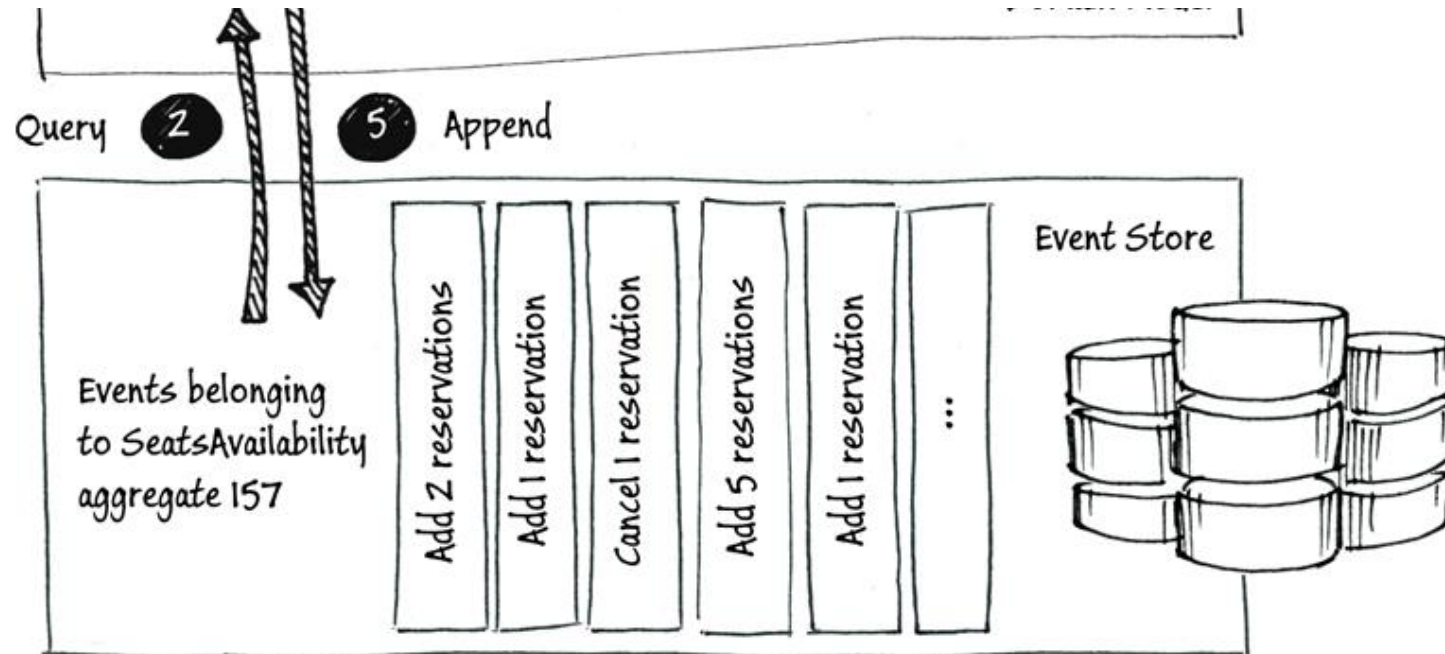
Events as Primary Source of Truth

Instead of storing the current state of the application, every state change is stored as an immutable event. These events are the single source of truth.



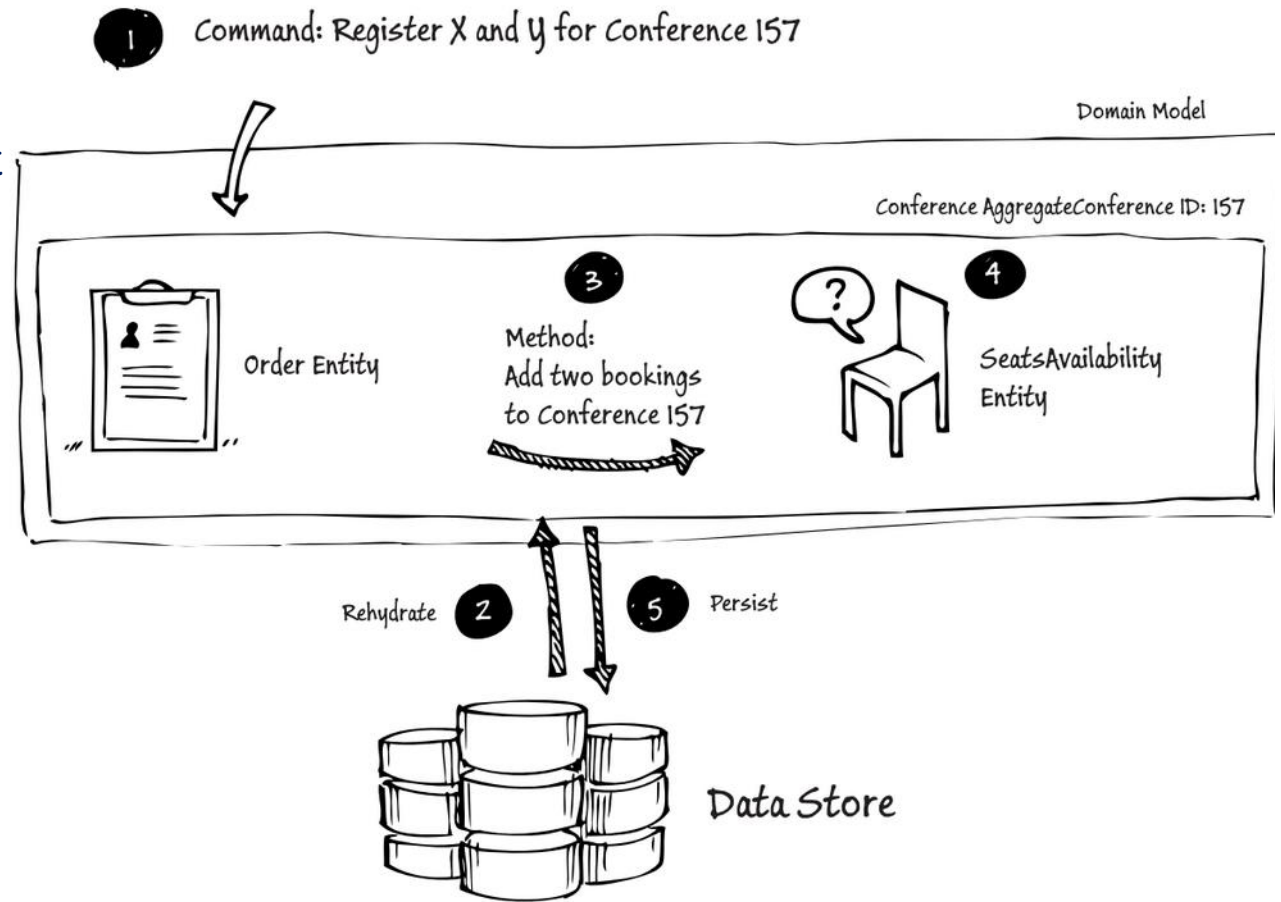
Event Store

A specialized database or storage mechanism that saves all the events. This store can be queried to rebuild the current state by replaying the events.



Rebuilding State

To get the current state of an object, the application replays all the events related to that object from the event store. This allows for reconstructing the state at any given time.



Audit Trail

Event sourcing provides a complete audit trail of all changes, as every state change is recorded as an event. This makes it easy to track the history of changes for debugging, auditing, or other purposes.

Time	Exch	Contract	Message Type	ExecType	Status
13:49:01.074	CME	ZB Sep15	ExecutionReport	New	Working
13:49:01.013	CME	ZB Sep15	NewOrderSingle		
00:00:00.064	CME	ZN Dec15	PositionModificati		Credit Res
00:00:00.055	CME	ZN Dec15	PositionModificati		Credit Res
16:00:00.069	CME	ZN Sep15	ExecutionReport	Expired	Expired
16:00:00.069	CME	ZN Sep15	ExecutionReport	Expired	Expired
12:01:37.299	CME	ZN Dec15	ExecutionReport	Trade	Filled
12:01:37.177	CME	ZN Dec15	ExecutionReport	Trade	Partially F
12:01:37.176	CME	ZN Dec15	ExecutionReport	Trade	Partially F
12:01:37.106	CME	ZN Dec15	ExecutionReport	Trade	Partially F
12:01:37.104	CME	ZN Dec15	ExecutionReport	Trade	Partially F
12:01:37.096	CME	ZN Dec15	ExecutionReport	Trade	Partially F
12:01:37.096	CME	ZN Dec15	ExecutionReport	Trade	Partially F

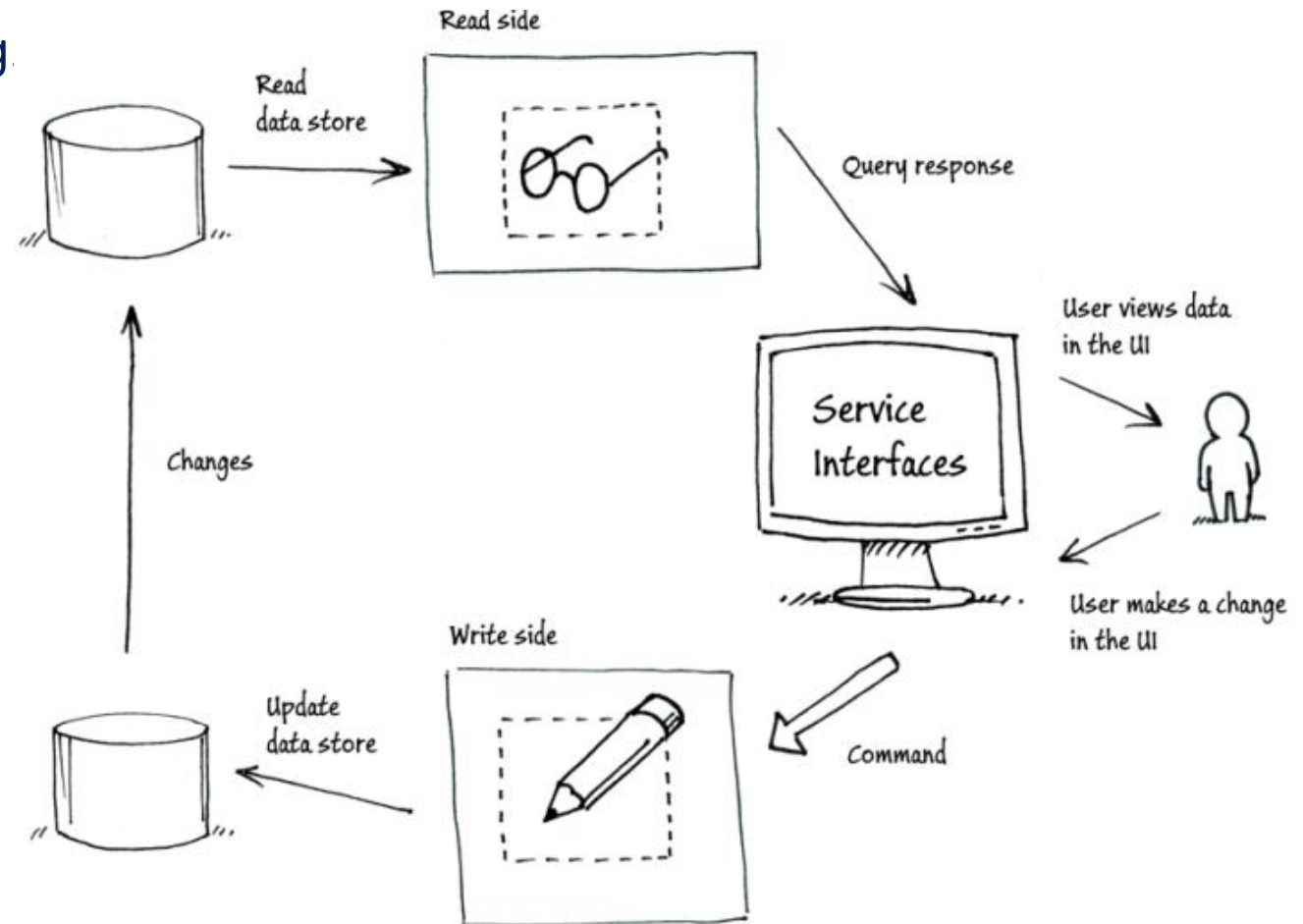
Temporal Queries

With a sequence of events, it's possible to reconstruct the state of the application at any point in the past. This allows for historical queries and debugging.

```
AS OF <date_time>
FROM <start_date_time> TO <end_date_time>
BETWEEN <start_date_time> AND <end_date_time>
CONTAINED IN (<start_date_time> , <end_date_time>)
ALL
```

Command Query Responsibility Segregation

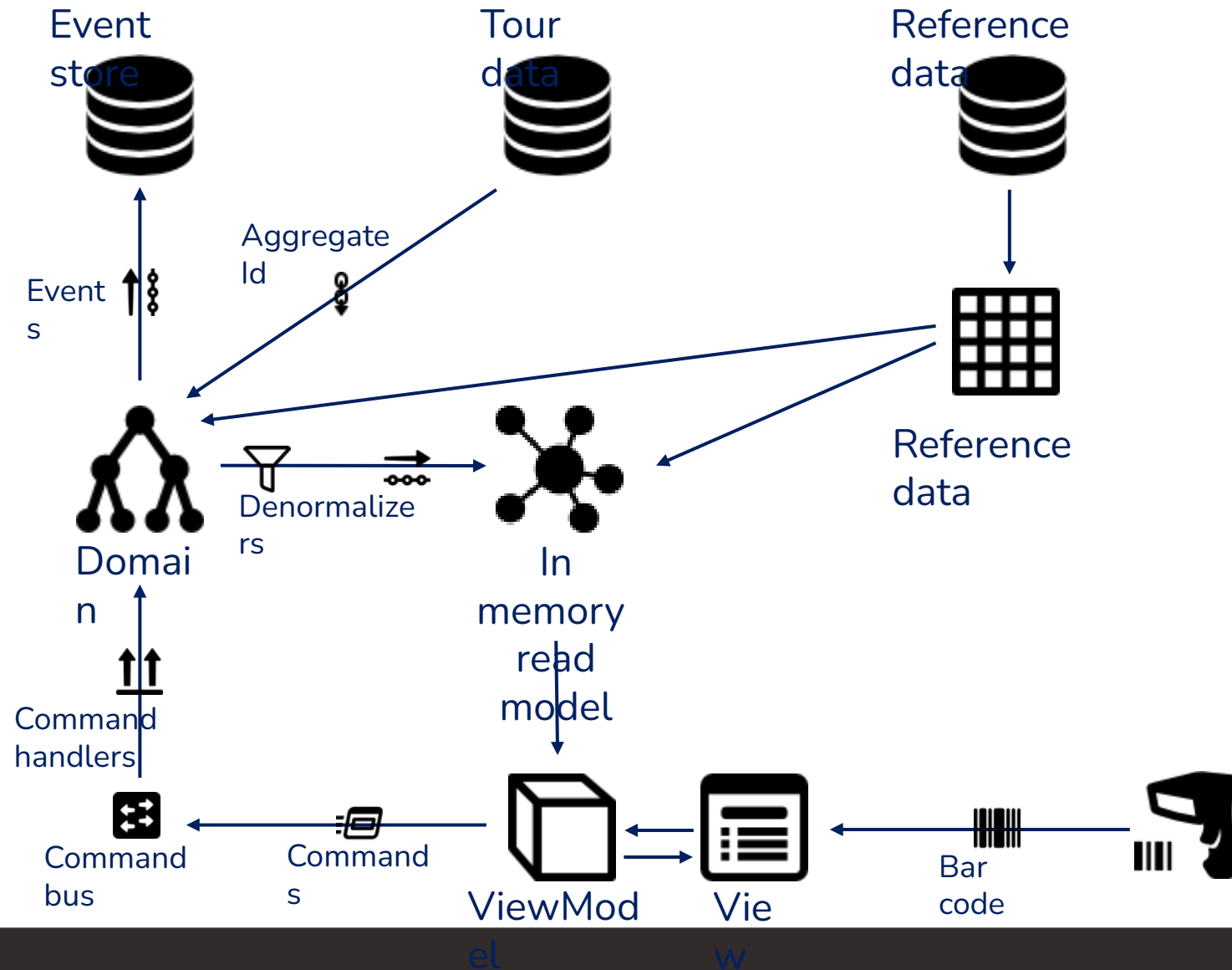
Often used together with event sourcing, CQRS separates the read and write operations of the data. Commands change the state and are handled by writing events, while queries read the state by replaying events.



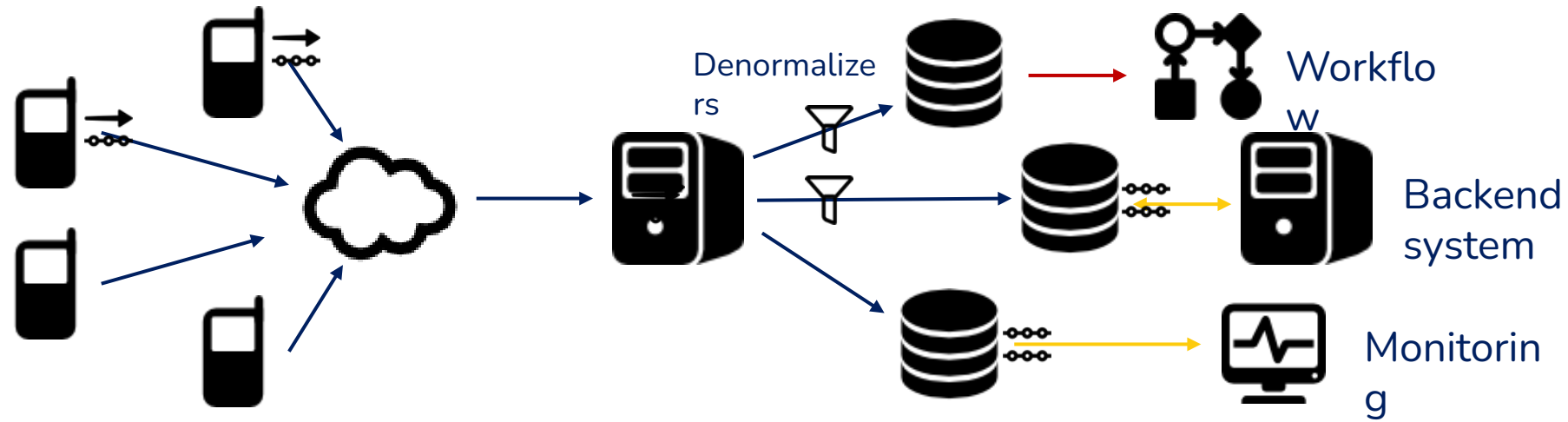
Once upon a time at POST Luxembourg



On device



Event sourced mail delivery system



Let's create our very own restaurant

Waiter

- Takes the client's order
- Brings the order to the cook
- Manages drinks stocks
- Serves plates when ready

Cook

- Receives order from waiter
- Prepares the dishes
- Manages food stocks
- Sends cooked order to accountant

Accountant

- Receives cooked order from cook
- Calculates the client's bill
- Calculates daily revenue
- Sends calculated bill to cashier

Cashier

- Receives calculated bill from accountant
- Gets payment from customer
- Calculates total incoming payment

Our menu

Food

1. Mettwurst – 5 €
2. Gromperekichelcher – 7 €
3. Bouchée à la reine – 15 €
4. Tartare de bœuf – 18 €



Drinks

- A. Viva (eau plate) – 4 €
- B. Rosport bleue (eau gazeuse) – 4 €
- C. Battin (bière) – 5 €
- D. Vin blanc de la Moselle – 7 €



$$[z(z)] \left[\left(\frac{F}{f(z)} \right) P \right] f(z)$$

$$(1+z)^2 \sqrt{1+2q_0^2}$$

$$\vec{p} + \vec{\nabla} r \cdot \left(p + \frac{p_{th}}{cr} \right) \leq 0$$

$$\vec{u} = -\vec{\nabla} r \phi \sim r^3$$

PER DENSITY OF THE UNIVERSE

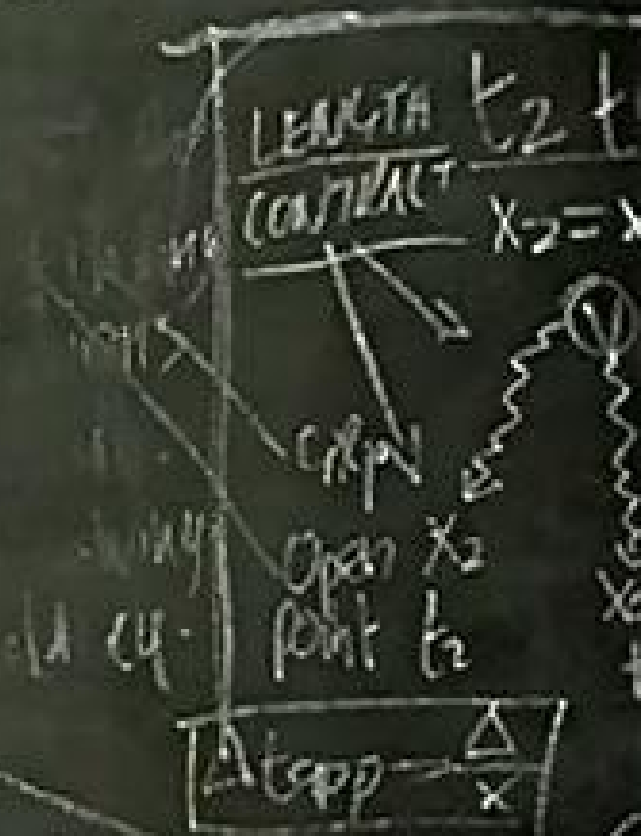


$$H = \frac{1}{R}$$

✓ ETE - photo - spectra (a)

THE SOLUTION IS: $x = \frac{c}{g}$

What if?



- circle of light

- R tensor PROPORTIONAL



INTE
AP
LO

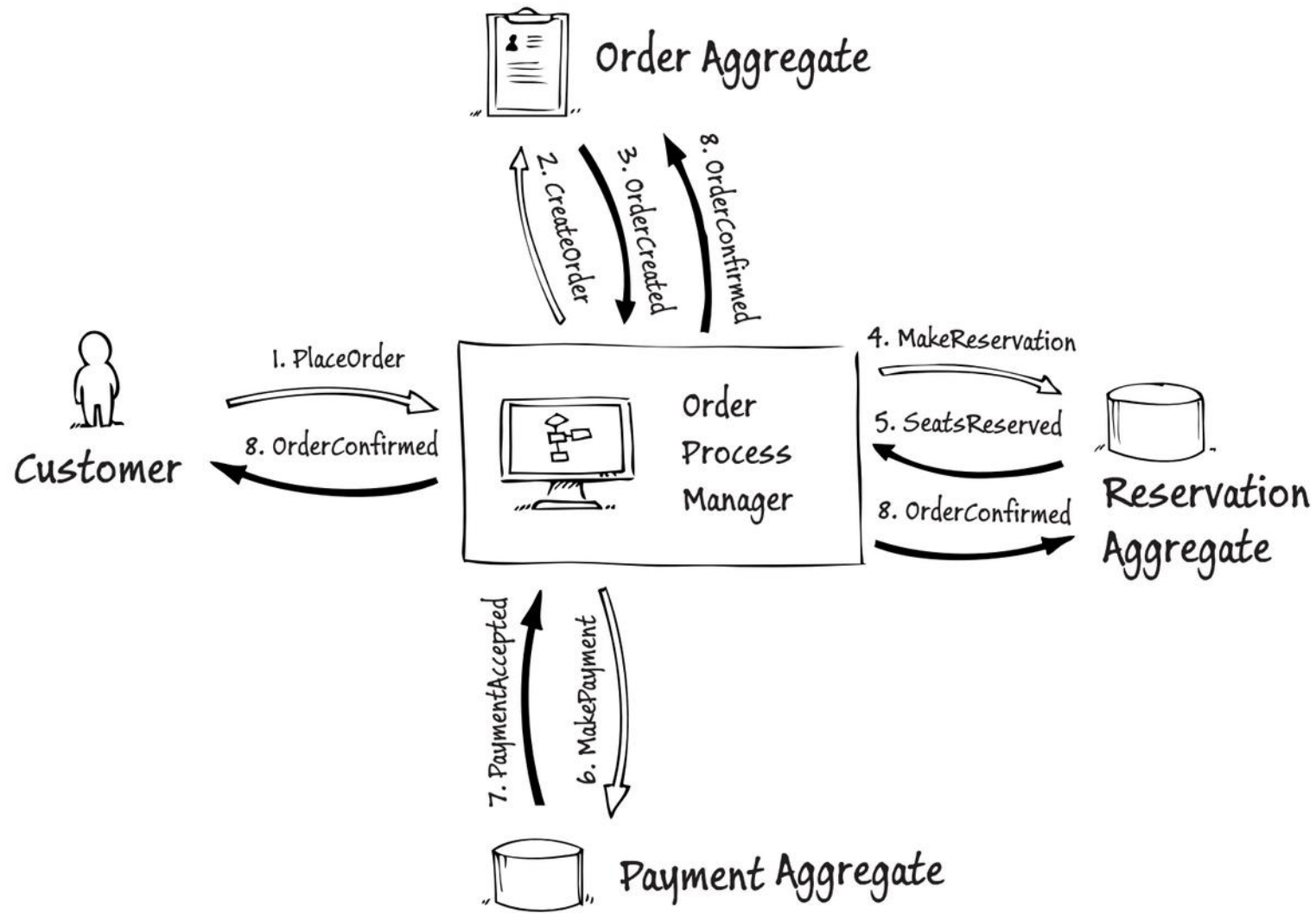
Choreography



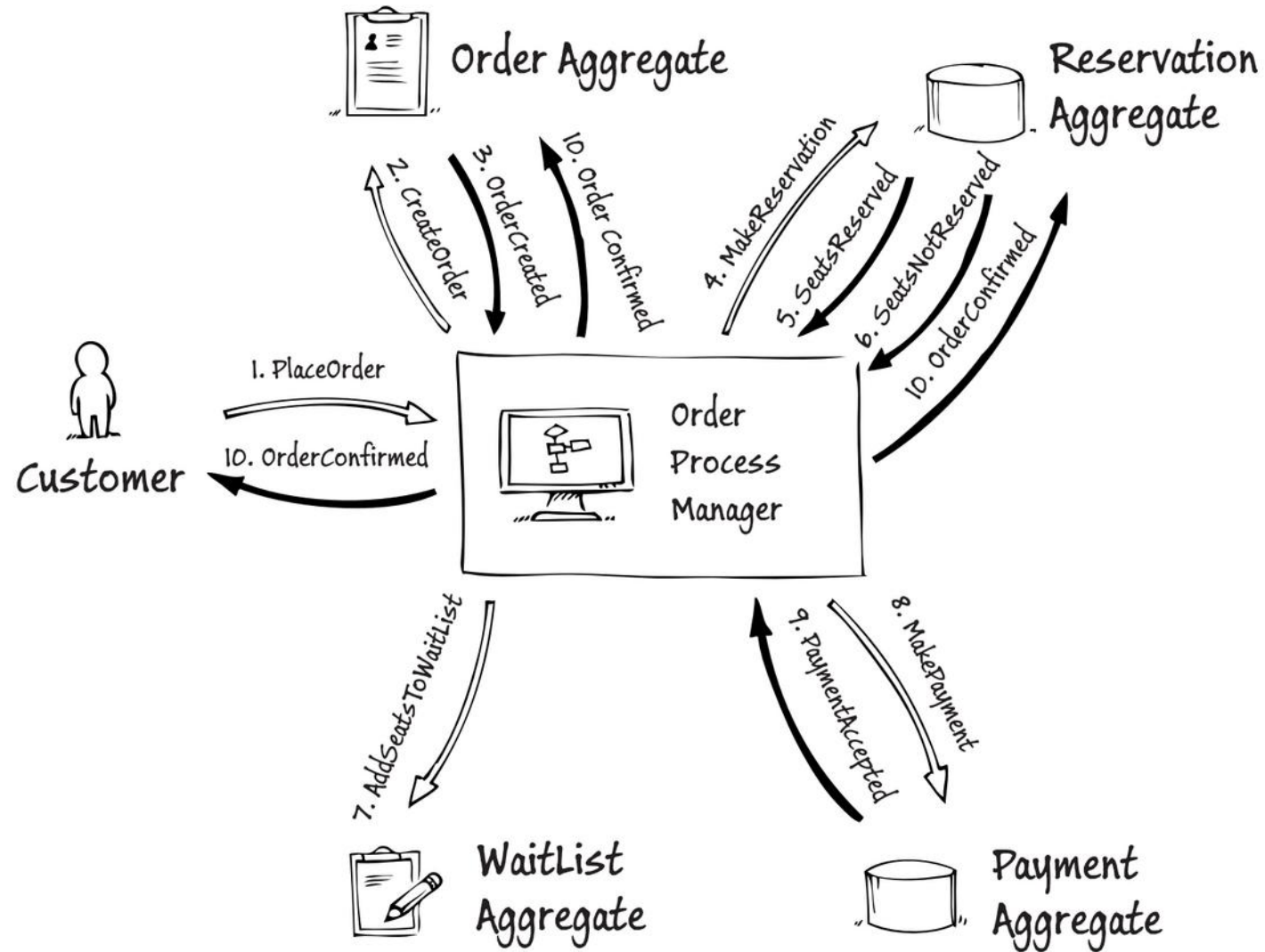
Orchestration



Process manager

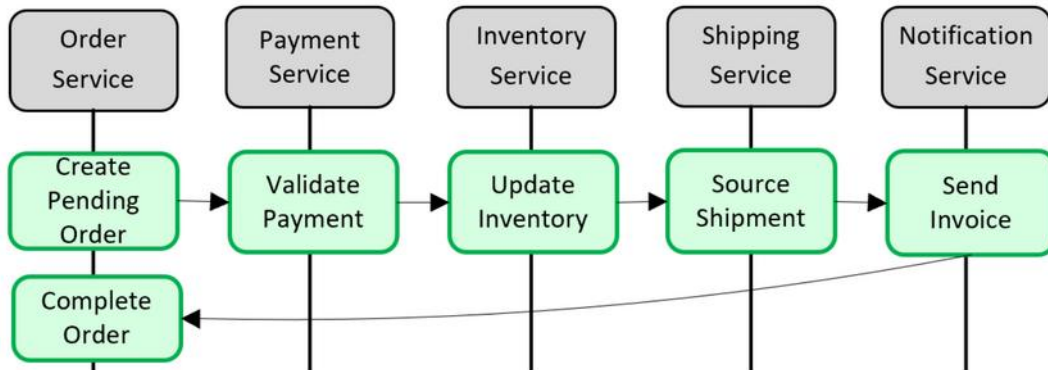


Process manager

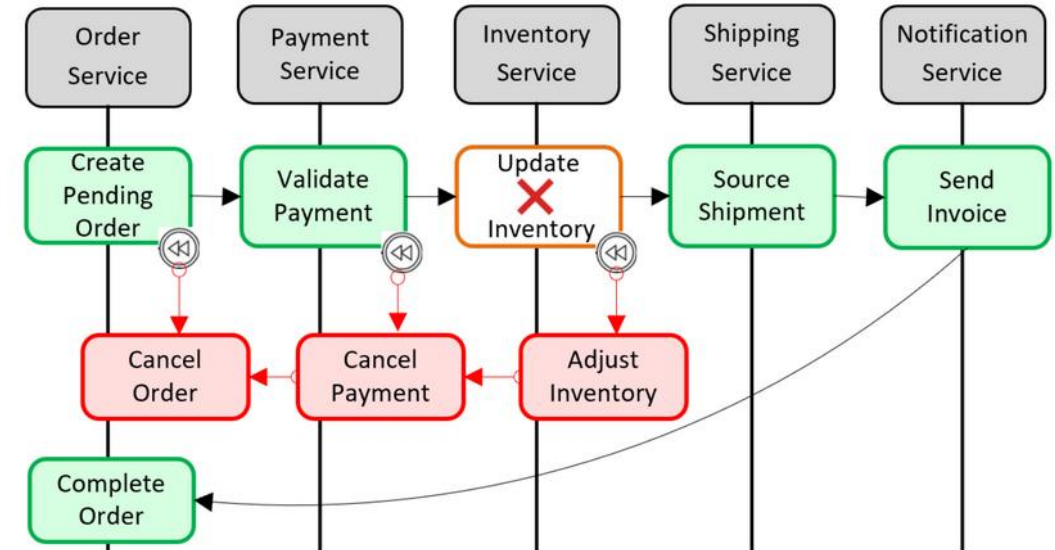


Distributed transaction

Happy path



Compensating transaction



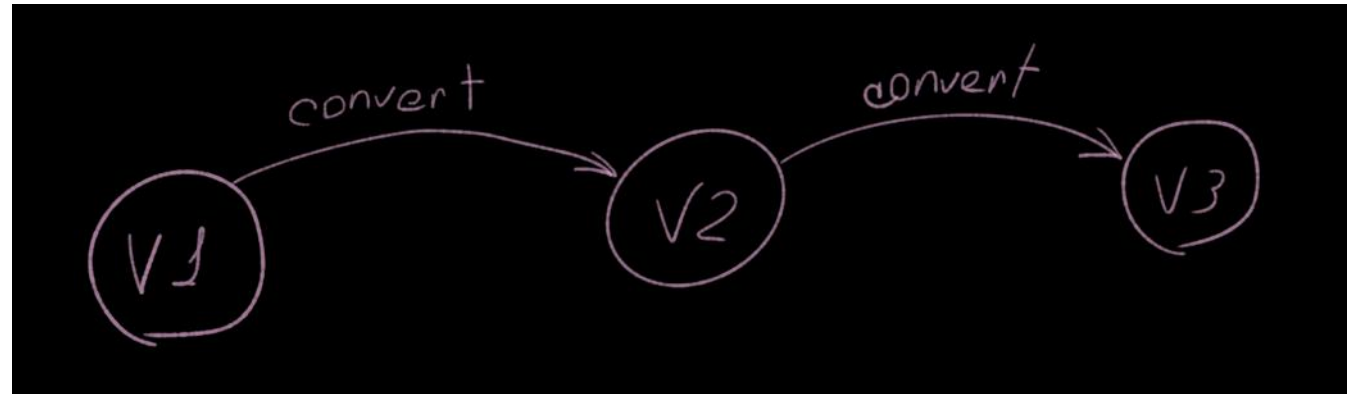
Which is better?

	Choreography	Orchestration
Control	Decentralized	Centralized
Workflow Logic	Implicit, emerging from the interactions between services	Explicitly defined and managed by the orchestrator
Complexity Management	Hard (Distributed)	Easier (Centralized)
Scalability	Typically more scalable due to decentralized decision-making	May face scalability issues due to the central controller
Fault Tolerance	More fault-tolerant due to the absence of a single controlling entity	Single point of failure at the orchestrator

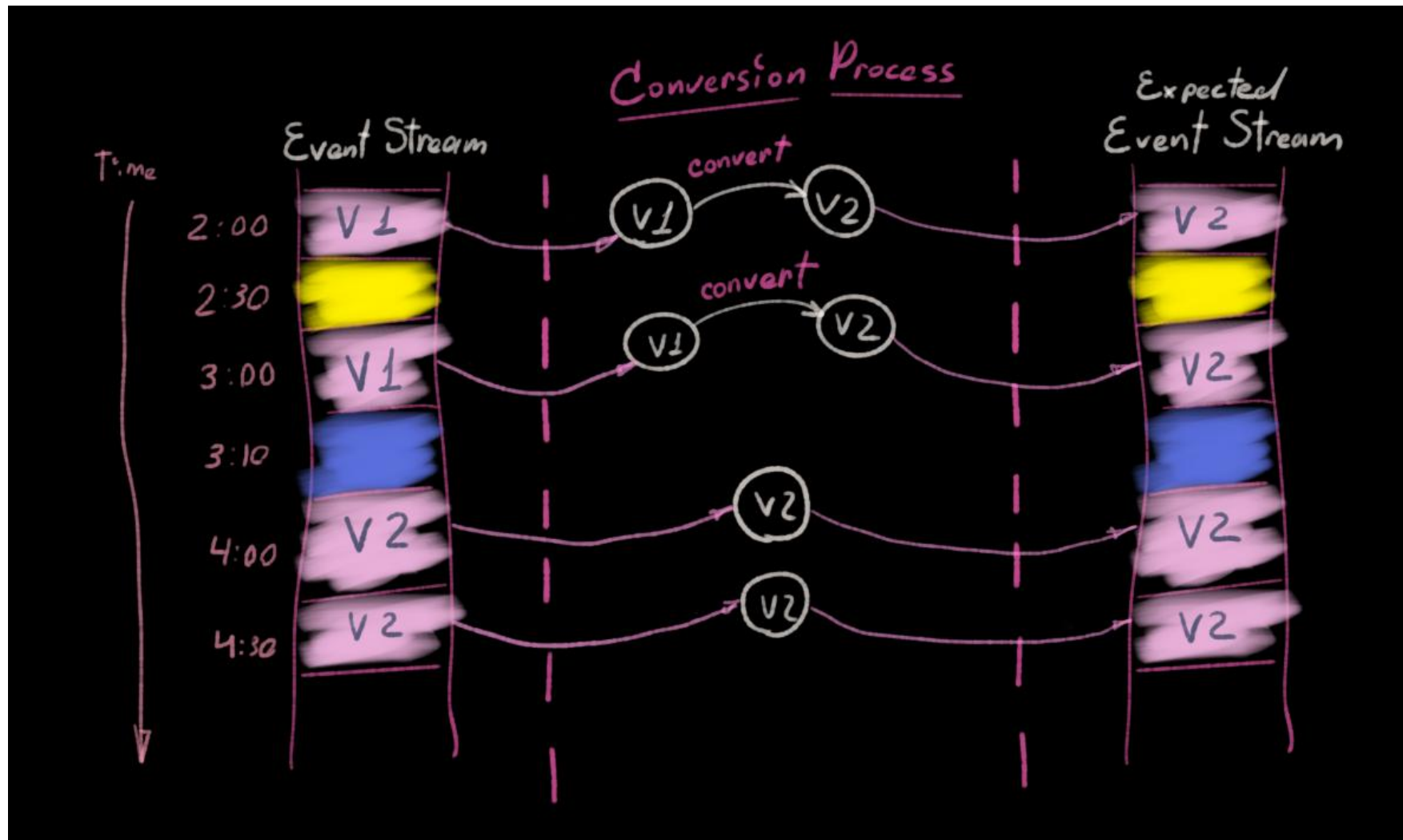
Event versioning



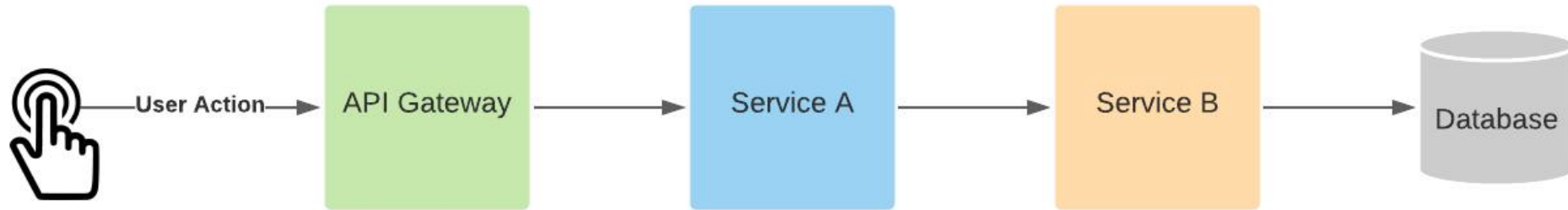
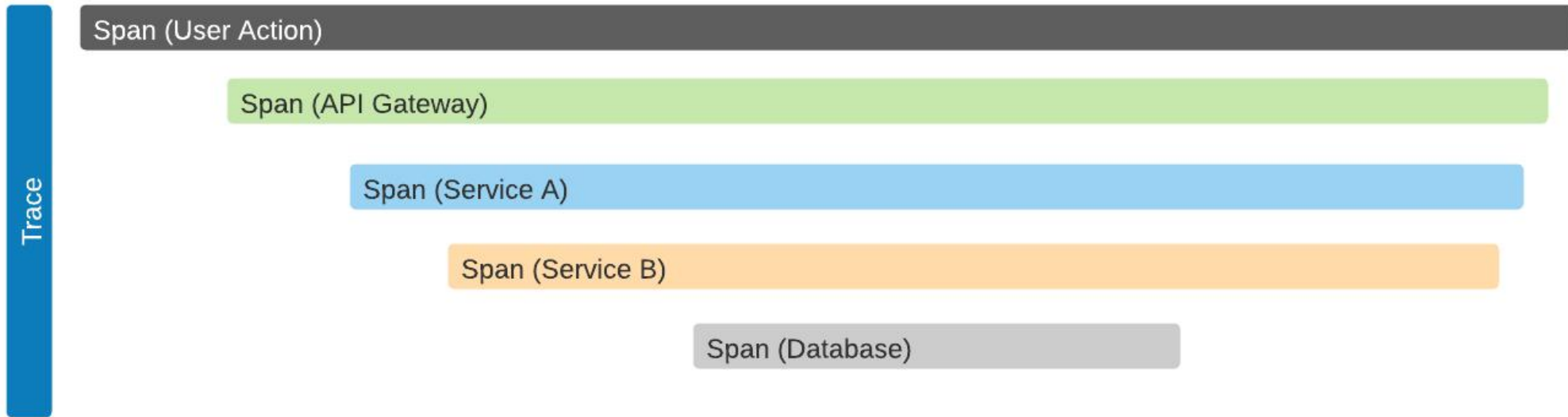
- Events are by definition immutable
- Weak schema (tolerant reader)
 - Mapper
 - Wrapper
- Content negotiation
- Strategies
 - Simple mapping
 - Upcasting
 - Downcasting
 - Event stream migration
- Snapshotting



Event versioning



Tracing





Merci !

