How Events Are Reshaping Modern Systems

Jonas Bonér
@jboner

Lightbend

WHY SHOULD YOU CARE ABOUT EVENTS?

1. Events DRIVE AUTONOMY
2. Events HELP REDUCE RISK
3. Events HELP YOU MOVE FASTER
4. Events INCREASE LOOSE COUPLING
5. Events INCREASE STABILITY
6. Events INCREASE SCALABILITY
7. Events INCREASE RESILIENCE
8. Events INCREASE TRACEABILITY
9. Events ALLOW FOR TIME-TRAVEL

WHY NOW?

1. Cloud and multicore architectures
2. Microservices and distributed systems
3. Data-centric applications
4. “We want more, of everything, and we want it now.” -Your Customers

WHAT IS AN EVENT?

The Nature of Events

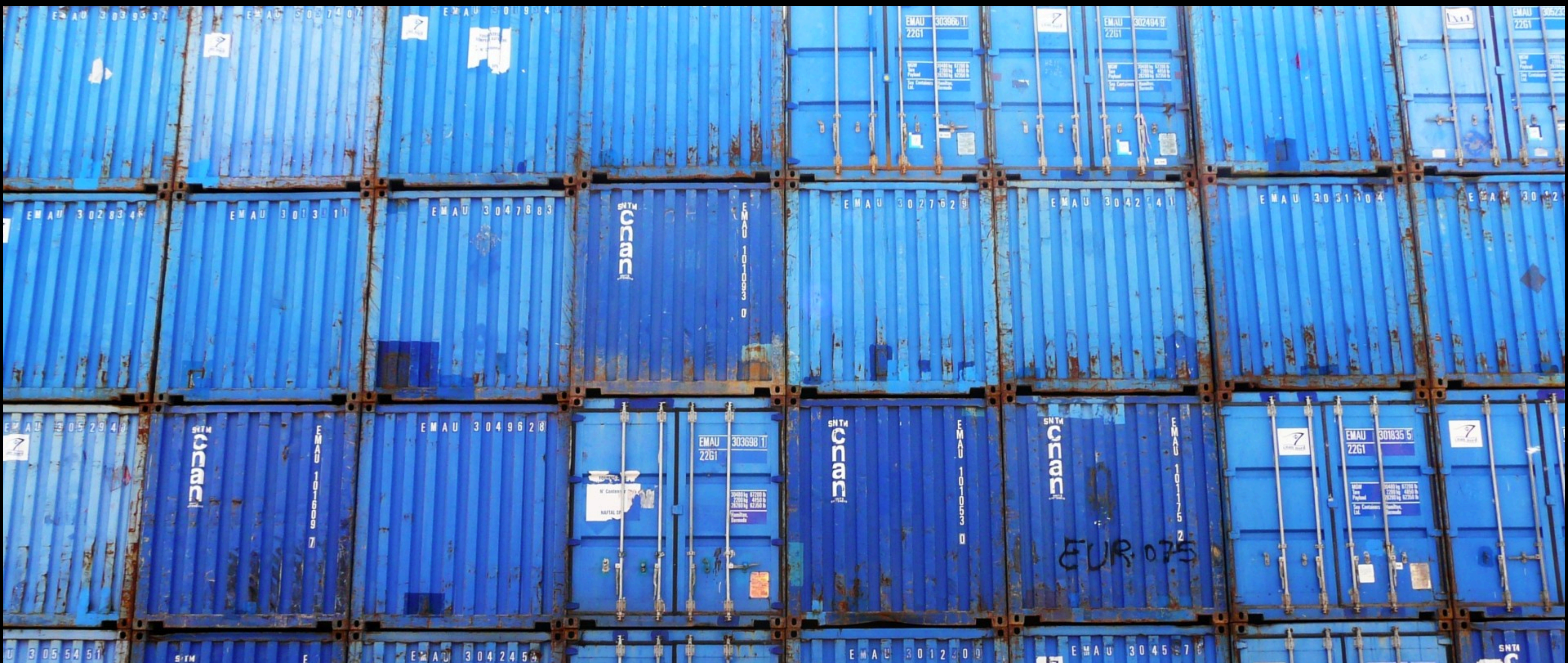
- Events represent FACTS OF INFORMATION
  - ➡ FACTS ARE IMMUTABLE
  - ➡ FACTS ACCRUE - KNOWLEDGE CAN ONLY GROW
- Events/Facts CAN BE DISREGARDED/IGNORED
- Events/Facts CAN NOT BE RETRACTED (once accepted)
- Events/Facts CAN NOT BE DELETED (once accepted)
  - ➡ Might be needed for LEGAL OR MORAL REASONS
- Events/Facts (new) CAN INVALIDATE existing Facts

Event Driven Services

1. RECEIVE and REACT (or not) TO FACTS, that are coming its way
2. PUBLISH NEW FACTS (immutable events) to the rest of the world
3. INVERT THE CONTROL FLOW to minimize coupling and increase autonomy

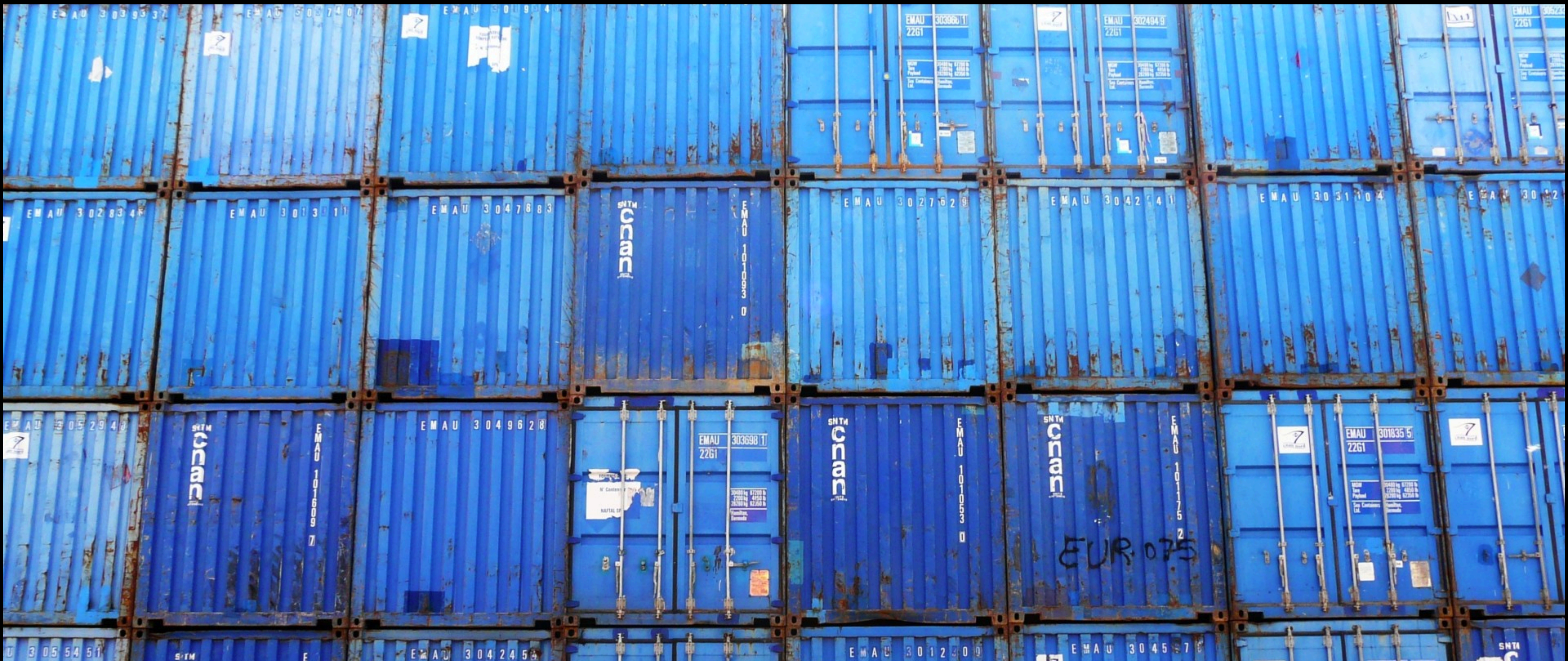
Mutable State Needs To Be Contained

And Non Observable

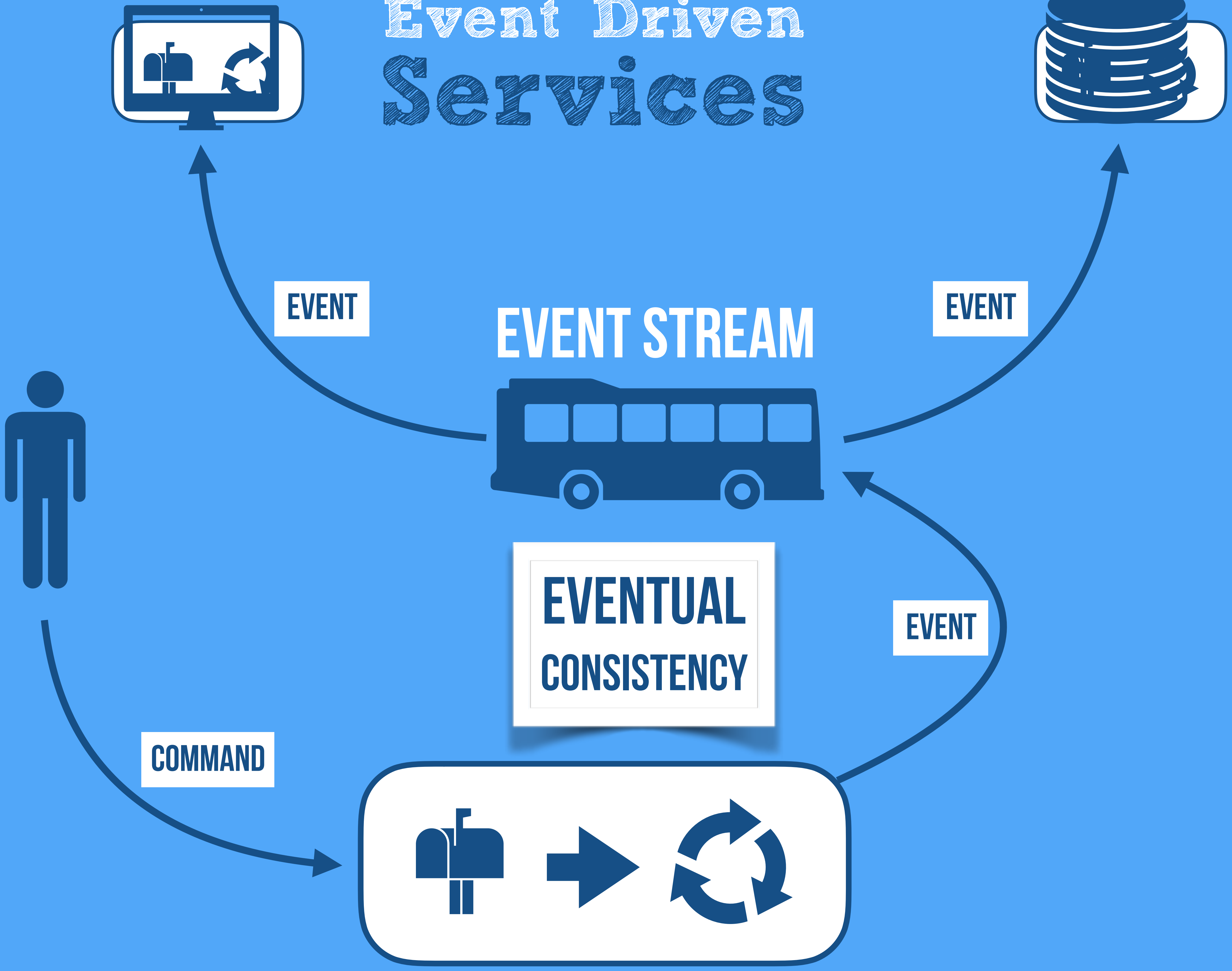


Publish Facts

To Outside World



Event Driven Services



USE THE Event Stream AS THE COMMUNICATION FABRIC

USE THE

Event Stream

AS THE INTEGRATION FABRIC

USE THE Event Stream AS THE REPLICATION FABRIC

USE THE Event Stream AS THE CONSENSUS FABRIC

USE THE Event Stream AS THE PERSISTENCE FABRIC

WE HAVE TO RELY ON

Eventual Consistency

BUT RELAX—IT'S HOW THE WORLD WORKS

Information Has Latency

Information Is Always From the Past

Welcome To The Wild Ocean Of Non Determinism Distributed Systems

We Need To Model Uncertainty

"In a system which cannot count on distributed transactions, the management of uncertainty must be implemented in the business logic."

- PAT HELLAND

Life Beyond Distributed Transactions, Pat Helland (2007)

Events Can Lead To Greater

Certainty

“An autonomus component can only promise its own behavior.”
“Autonomy makes information local, leading to greater certainty and stability.”

- MARK BURGESS

Events Can Help Us Craft Autonomous Islands Of Determinism

“Accidents come from relationships not broken parts.”

- SIDNEY DEKKER

“Complex systems run as broken systems.”

- RICHARD COOK

Resilience is by Design

Photo courtesy of FEMA/Joselyne Augustino

EVENTS CAN HELP US

Manage Failure

INSTEAD OF TRYING TO AVOID IT

Requirements for a Sane Failure Model

Failures need to be

1. Contained—Avoid cascading failures
2. Reified—as events
3. Signalled—Asynchronously
4. Observed—by 1-N
5. Managed—Outside failed context

But All This Stuff

- ASYNC?
- DISTRIBUTED SYSTEMS?
- EVENTUAL CONSISTENCY?
- UNCERTAINTY?
- FAILURE MODELS?

Is Hard

Think

In Terms Of

Workflow

Events First Domain Driven Design

"When you start modeling events, it forces you to think about the behaviour of the system. As opposed to thinking about the structure of the system."

- GREG YOUNG

A Decade of DDD, CQRS, Event Sourcing, Greg Young (Presentation from 2016)

- DON'T FOCUS ON THE THINGS

  The Nouns

  The Domain Objects

- FOCUS ON WHAT HAPPENS

  The Verbs

  The Events

Mine the Facts

Event Storming

Event Driven Design

- INTENTS
  - ➡ Communication

    Commands
  - ➡ Expectations
  - ➡ Contracts
  - ➡ Control Transfer
- FACTS
  - ➡ State

    Events
  - ➡ Causality
  - ➡ Notifications
  - ➡ State Transfer

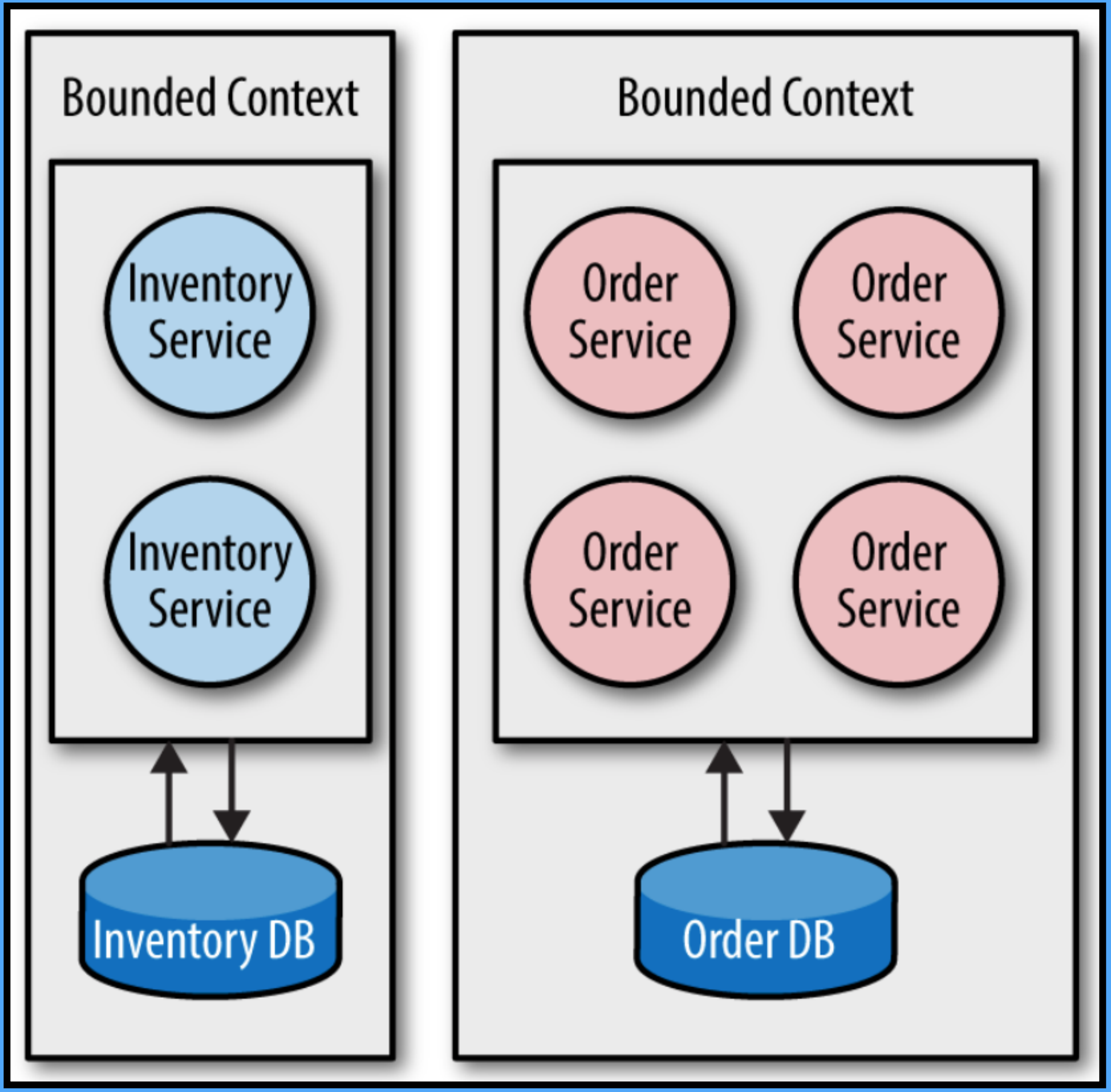
Event Driven Design

- COMMANDS
  - Object form of METHOD/ACTION REQUEST
  - IMPERATIVE: `CreateOrder`, `ShipProduct`
- REACTIONS
  - Represents SIDE-EFFECTS
- EVENTS
  - Represents something that HAS HAPPENED
  - PAST-TENSE: `OrderCreated`, `ProductShipped`

COMMANDS vs EVENTS

1. All about intent	1. Intentless
2. Directed	2. Anonymous
3. Single addressable destination	3. Just happens - for others (0-N) to observe
4. Models personal communication	4. Models broadcast (speakers corner)
5. Distributed focus	5. Local focus
6. Command & Control	6. Autonomy

Let the Events Define the Bounded Context



Inside Data

OUR CURRENT PRESENT——STATE

Outside Data

BLAST FROM THE PAST——EVENTS/FACTS

Between Services

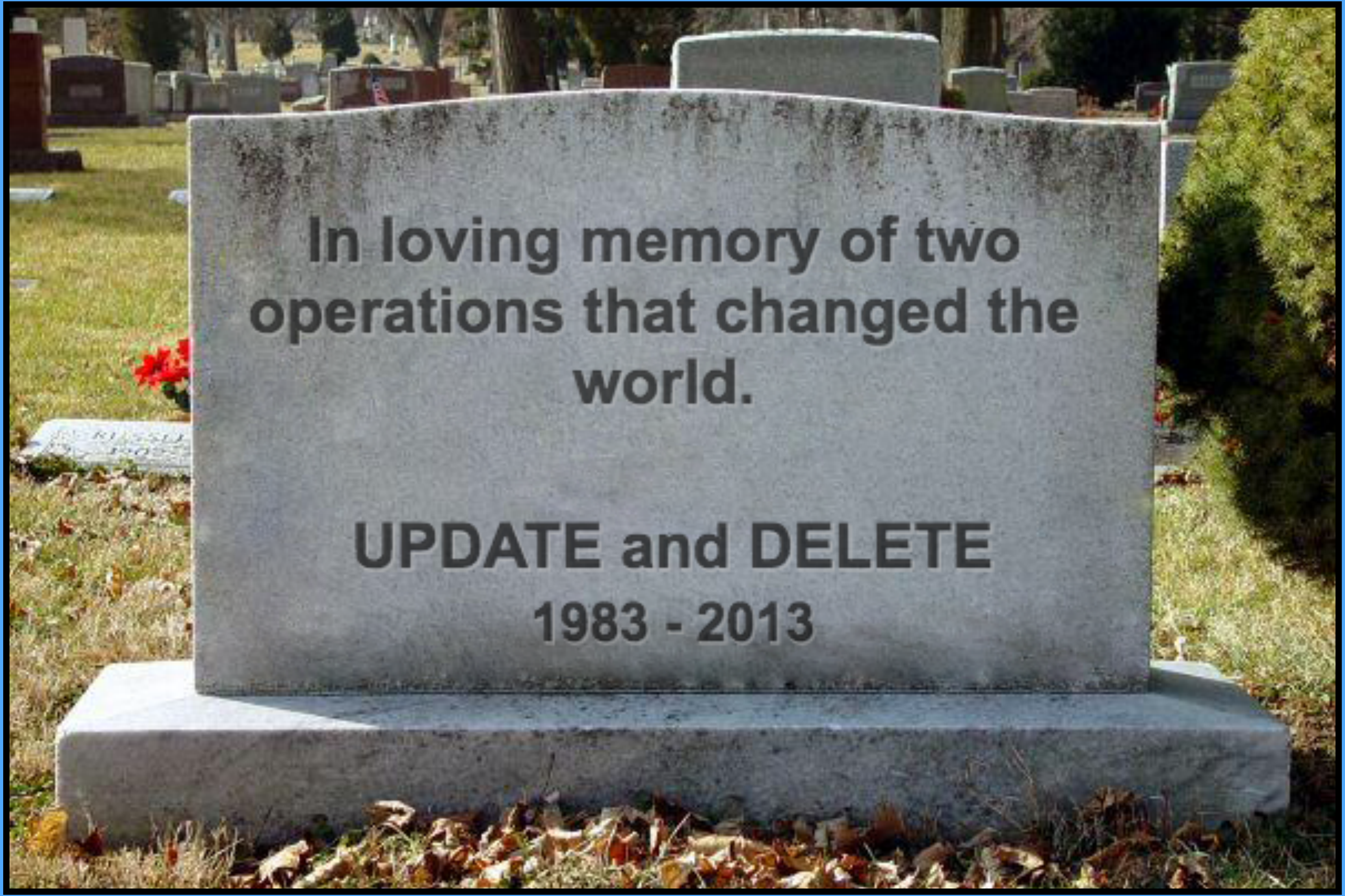
HOPE FOR THE FUTURE——COMMANDS

Event Based Persistence

The Aggregate

- Maintains INTEGRITY & CONSISTENCY
- Is our UNIT OF CONSISTENCY
- Is our UNIT OF FAILURE
- Is our UNIT OF DETERMINISM
- Is fully AUTONOMOUS

CRUD is DEAD

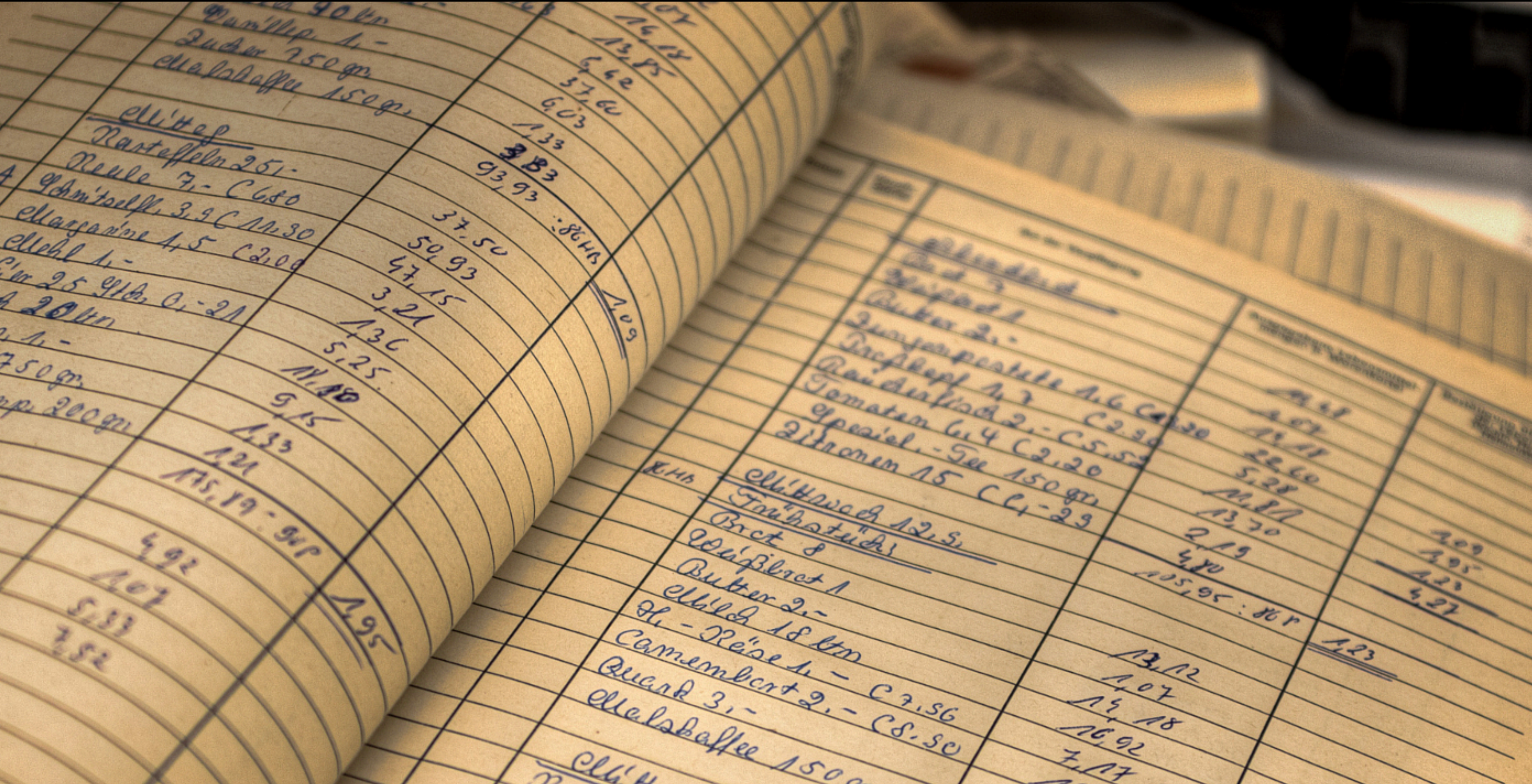

In loving memory of two operations that changed the world.
UPDATE and DELETE
1983 - 2013

“Update-in-place strikes systems designers as a cardinal sin: it violates traditional accounting practices that have been observed for hundreds of years.”

- JIM GRAY

The Transaction Concept, Jim Gray (1981)

Event Logging
The Bedrock



"The truth is the log.
The database is a cache
of a subset of the log."

- PAT HELLAND

Immutability Changes Everything, Pat Helland (2015)

Event Sourcing

A Cure For the Cardinal Sin

Event Sourced Services

Memory Image

HAPPY P

…aymentApproved")

…Append Event to Event Log

4) Update internal component state

5) Run side-effects (approve the payment)

SAD PATH - RECOVER FROM FAILURE

1) Rehydrate Events from Event Log

2) Update internal component state

Event Sourcing

- One single SOURCE OF TRUTH with ALL HISTORY
- Allows for MEMORY IMAGE (Durable In-Memory State)
- Avoids the OBJECT-RELATIONAL MISMATCH
- Allows others to SUBSCRIBE TO STATE CHANGES
- Has good MECHANICAL SYMPATHY (Single Writer Principle etc.)

Disadvantages

Of Using Event Sourcing

- UNFAMILIAR model
- VERSIONING of events
- DELETION of events (legal or moral reasons)

Events

Allow Us To Manage

Time

“Modelling events forces you to have a temporal focus on what’s going on in the system. Time becomes a crucial factor of the system.”

- GREG YOUNG

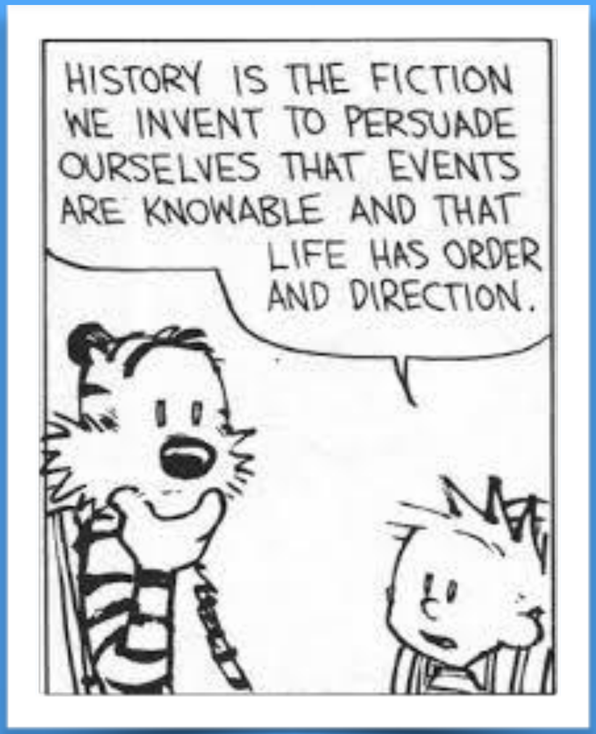
A Decade of DDD, CQRS, Event Sourcing, Greg Young (Presentation from 2016)

Event Sourcing Allows Us To Model Time

- ✳ Event is a SNAPSHOT IN TIME
- ✳ Event ID is an INDEX FOR TIME
- ✳ Event Log is our FULL HISTORY

The DATABASE OF OUR PAST

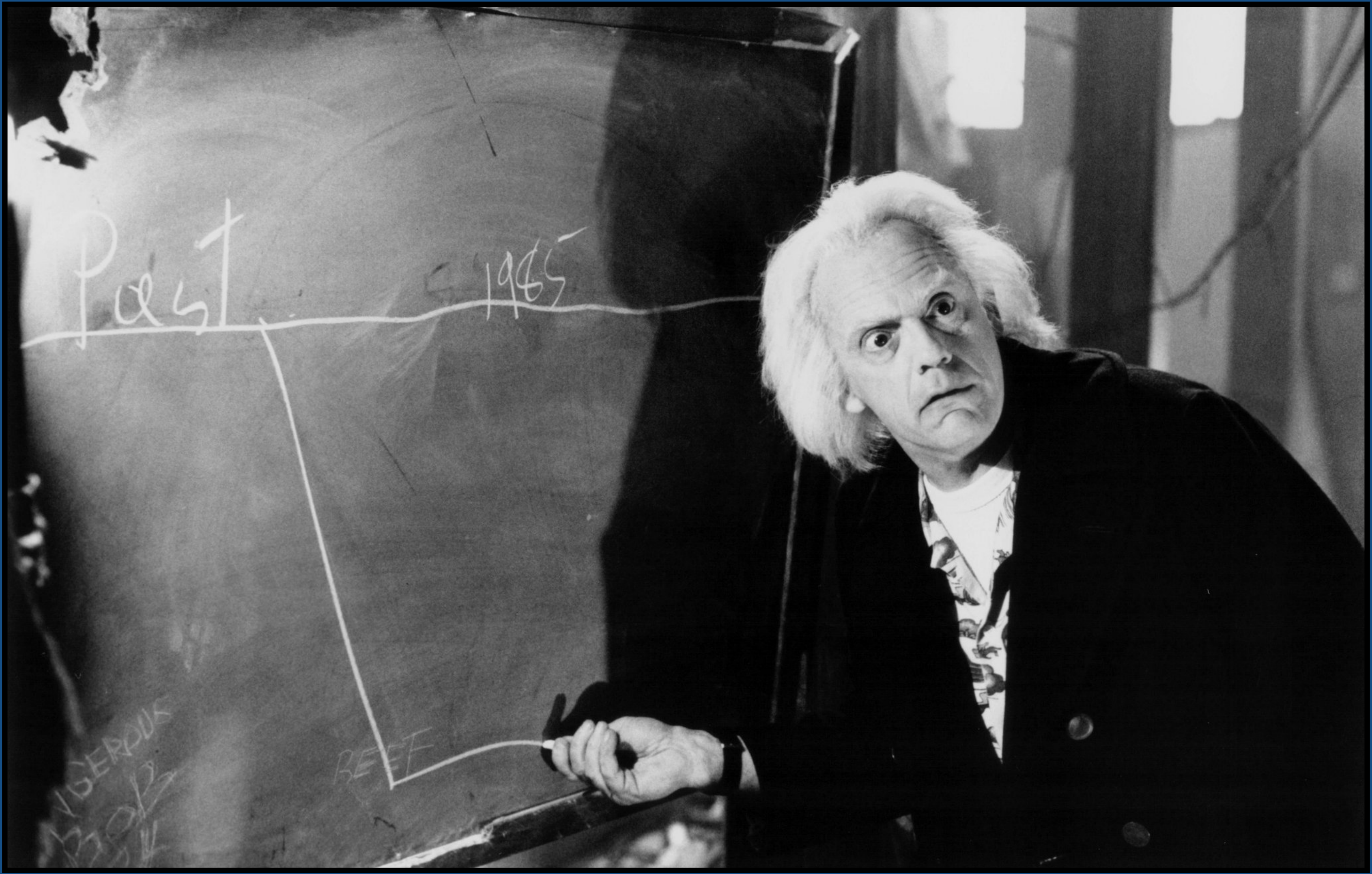
The PATH TO OUR PRESENT


HISTORY IS THE FICTION WE INVENT TO PERSUADE OURSELVES THAT EVENTS ARE KNOWABLE AND THAT LIFE HAS ORDER AND DIRECTION.

Event Sourcing Allows For Time Travel

- Replay the log FOR HISTORIC DEBUGGING
- Replay the log FOR AUDITING & TRACEABILITY
- Replay the log ON FAILURE
- Replay the log FOR REPLICATION

We Can Even Fork the Past



...Or Join Two Distinct Pasts

Key Takeaways

EVENTS-FIRST DESIGN helps you to:

- ✳ MOVE FASTER towards a RESILIENT architecture
- ✳ DESIGN AUTONOMOUS services
- ✳ BALANCE CERTAINTY and UNCERTAINTY
- ✳ REDUCE RISK when MODERNIZING applications

EVENT LOGGING allows you to:

- ✳ AVOID CRUD and ORM
- ✳ TAKE CONTROL of your system's HISTORY
- ✳ TIME-TRAVEL
- ✳ BALANCE STRONG and EVENTUAL consistency

akka

http://akka.io

Learn More

Download my latest book for free at:

bit.ly/reactive-microsystems

Lightbend