Architecting Tomorrow: a Vision on Event-Driven Architecture

Smals Research

Koen Vanderkimpen

Nov 9 2023

Smals
ICT for society

Smals Research 2023

Innovation with new technologies

Consultancy & expertise

Internal & external knowledge transfer

Support for going live

AI-Augmented Software Development

Workshops AI

Language Model for social security

Data protection in Legacy

Pseudonymization Service

Customer Service Automation

Recommender System Portal

Innovation Smals & OISZ

GIS Analytics

Confidential Computing

Custom Assistant

Causal AI

Event Driven Architecture

Graph Machine Learning

Synthetic Data in practice

On-prem Voice2voice Translation

Diffusion Models

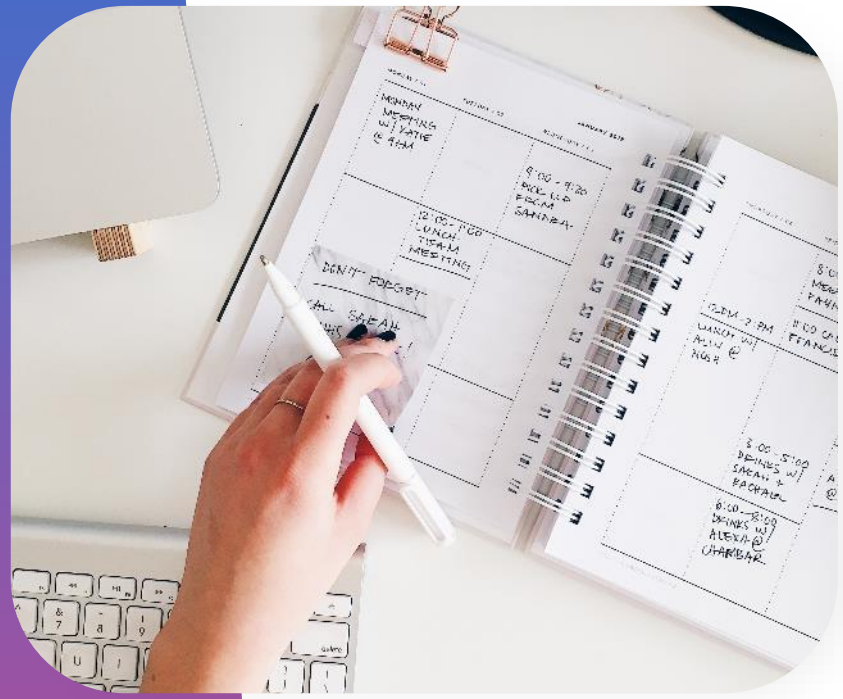
Digital EU Wallet

Linkurious in practice

Best Practices in Cryptography

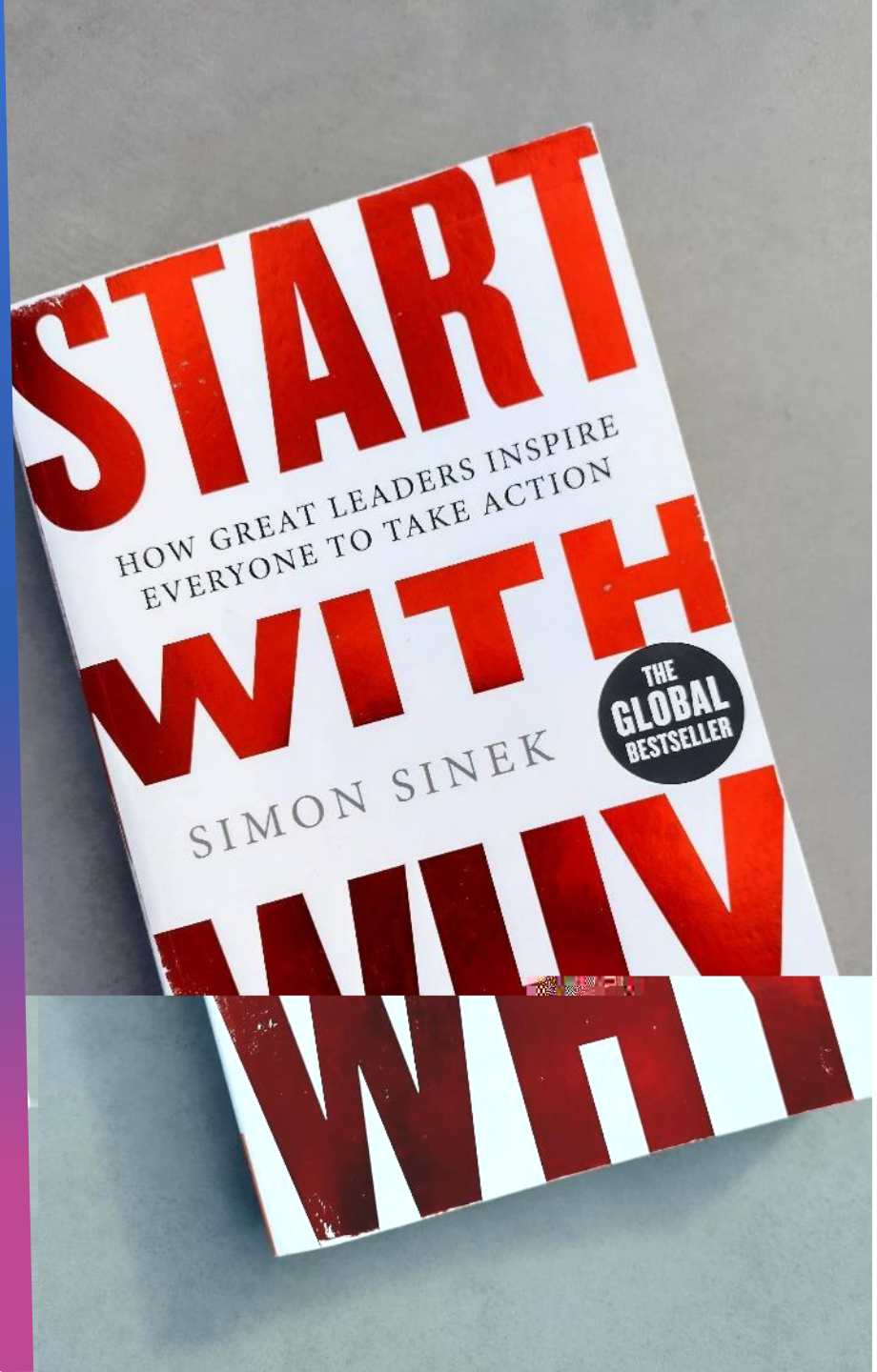
www.smalsresearch.be

Agenda



- Start with: WHY?
- EDA 101: What is Event Driven Architecture?
- An Evolving IT Architecture for the Future
- EDA 201: Some Advanced Concepts
- EDA benefits for Business and IT
- How to start using EDA: Approach and Mind-Set Shift
- Conclusions

Smals ICT for society



Let’s imagine our WHY

Some men see things as they are, and say why.

I dream of things that never were, and say why not.

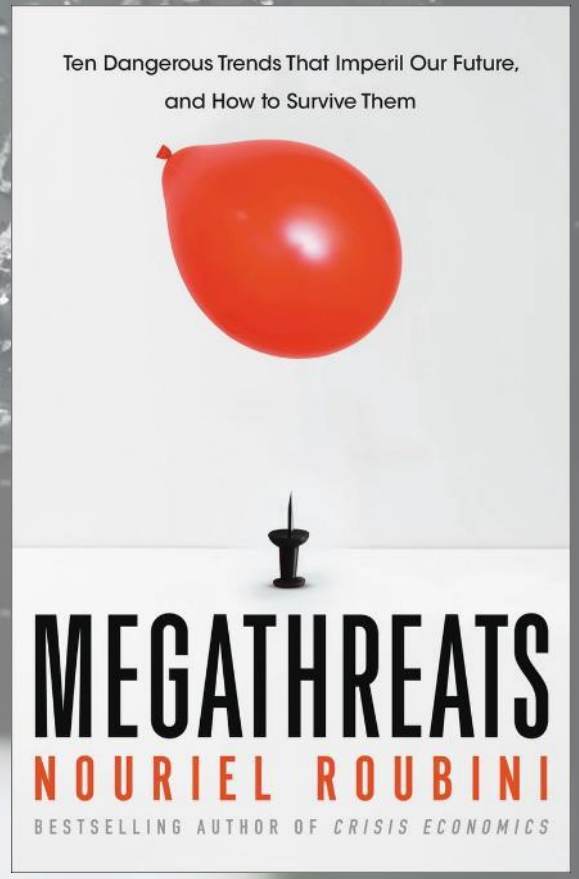
– Robert Kennedy

- My Vision
- eGov 3.0
- Why EDA; Why Now ?

Smals ICT for society

A Perfect Storm

- Debt, Inflation, Stagflation, Financial Crises, …
- Income Inequality
- Globalization/ Mobility of Workers & Capital
- Deglobalization / Protectionism / Political Extremism
- Aging Populace, Retiring Baby Boomers
- Global Health Emergencies, Pandemics
- Transformative Climate Change
- the AI revolution, Workplace Automation



(Book for sale in most book stores; Not the exact same threats)

Smals ICT for society

What's it to Us?

What can we do, as the architects and builders of the eGovernment?

More specifically:

Let's build a machine, capable of efficiently automating what we know and quickly adapting to what we don't know.

Smals ICT for society

My vision: Imagine… IT as the Agile Automator

In my vision, the administration, and its IT systems, get out of citizens' and companies' ways, as an **invisible force** that handles all the data and acts accordingly, with minimal human intervention. Whenever something changes in the real world, through actual events in people's lives, or by changing the rules by which the administration has to play, this system **easily adapts**, with minimal maintenance.

Key Principles: agility, automation, resiliency, once only, no wrong door

Main take-away: **EDA will be an important part of the puzzle ;-)**

Smals ICT for society

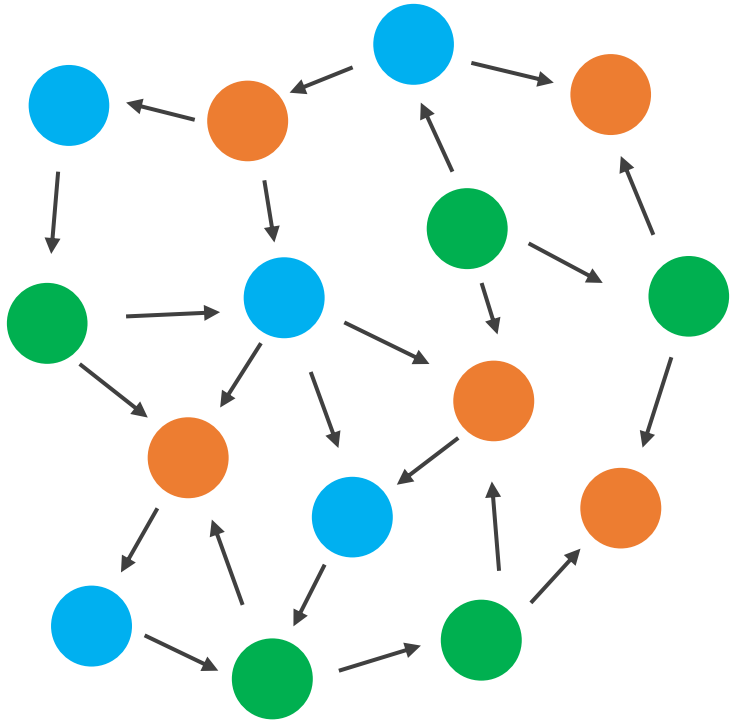
What does that mean, Practically? (1)

- **Resiliency**: (in the strict sense of the word) continuity (and self repair) in the presence of failures, even stronger than High Availability. But one could also include: very agile.
- **Agility**: Resiliency to required changes at the functional level. The greater the ease with which we can adapt the system's behaviour, and the smaller the impact of required change, the higher the agility.
- **Automation**: At business level (e.g. getting certain rights automatically) and at IT level (DevOps, CI/CD).

What does that mean, Practically? (2)

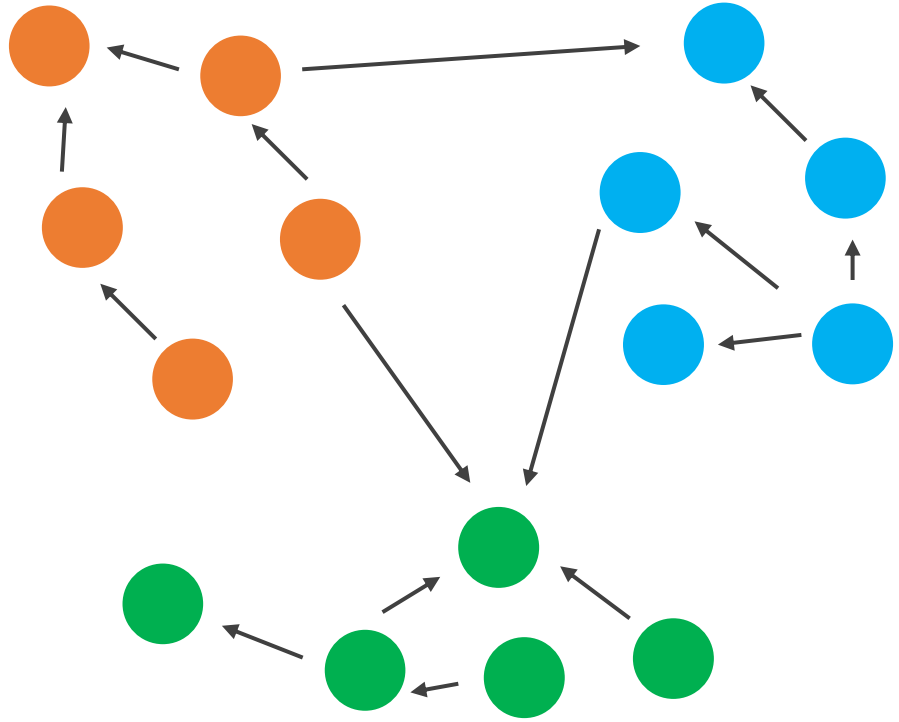
- **Once only**: you give your data to the government only once. It then travels wherever it is needed. : this does not mean we retain it only once! We are allowed to make copies (for technical or legal reasons).
- **No wrong door**: it does not matter which channel is used by the citizen/company to give the necessary data to the government.

Looking ahead to How: *Modularity*



LOW COHESION

HIGH COUPLING



HIGH COHESION

LOW COUPLING

Smals ICT for society

The eGov 3.0 Vision

Pure Pragmatism can't imagine a bold future.

Pure idealism can't get anything done.

It's when the two cooperate that magic happens.

– Simon Sinek

eGov 3.0: the Groundwork for the near and far future

Created by the leaders of our institutions of Social Security

Comparable to my own vision ☺ (but more practical)

Smals ICT for society

The eGov 3.0 Vision

A joint strategy for the future, driven by the joint institutions of Social Security (OISZ/IPSS), together with a number of stakeholders.

- Short Explanation
- Why EDA?
- eGov 3.0 EDA Exploration Workgroup

Smals ICT for society

What does it entail, in short?

Some statements:

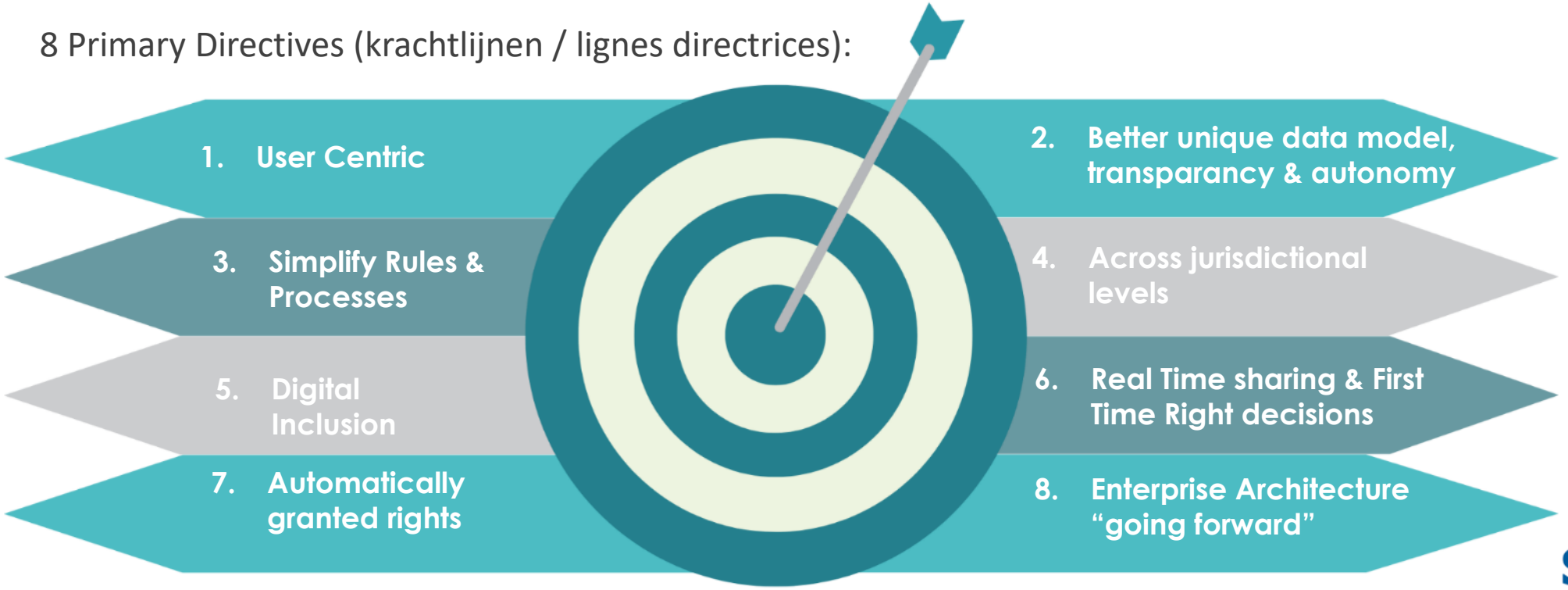
future proof

ecosystem context (adaptable, expandable)

Agility and flexibility are a conditio sine qua non

main goal

8 Primary Directives (krachtlijnen / lignes directrices):



Smals ICT for society

Why EDA for eGov 3.0?

- EDA can help with several of the directives: ***real-time, automatically granted rights, future-proof architecture, across jurisdictional levels***. Perhaps even more?
- EDA is referred to in the document:

real-time

maximally Event-Driven

a more event-driven approach

asynchronous communication and APIs, EDA, eventual consistency, reactive

Smals ICT for society

eGov 3.0 Exploration Workgroup: EDA

Elementary questions (WHY , WHAT, WHEN, WHO) about using EDA in OISZ

Justification

Current architecture paradigms are at their limits for the ambitions of eGov 3.0, so the event-driven architecture becomes necessary

Goal

Map what needs to be done for a shift towards using event-driven architecture

Scope

- *Map EDA impact on*
  - *architecture*
  - *business/organisation/processes*
- *SWOT analysis EDA*
- *Requirements to start using EDA*

Output

- *Guidelines for using EDA*
  - *SWOT EDA*
  - *When to use*
  - *When not*
- *Guidelines to analyse projects*
- *Approach on improving EDA knowledge @ OISZ*

OISZ: Openbare Instelling van de Sociale Zekerheid

Why EDA, in general?

- APIs are really good, but they don’t cover all types of **Integration**
- EDA offers increased **Agility, Modularity & Resiliency**
- **Real-time** Interactions & Insigths: Value of Information decreases with Time!

After Critical Mass attained:

- Cost Savings
- Faster Time to Market

Gartner:

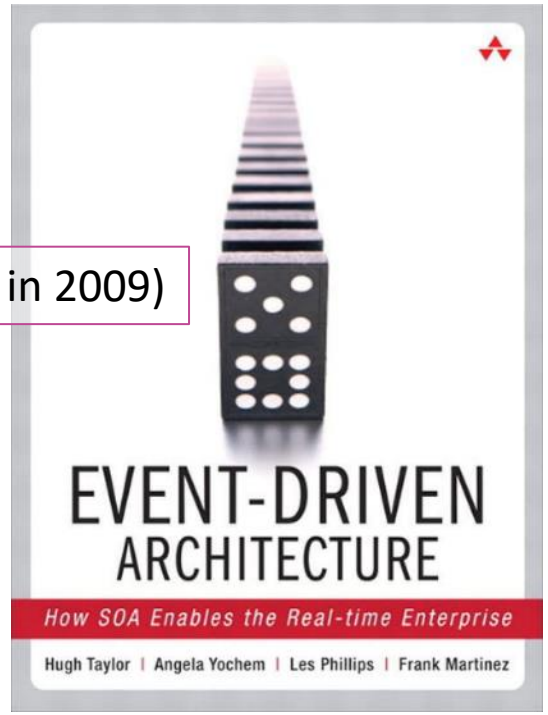
“The World is Event Driven

Are your Business Systems Event-Aware?

Smals ICT for society

Why EDA *Now*?

(Published in 2009)



EDA is **nothing new**, really

- Publish/Subscribe mechanism and Message queues have existed for decades

And : IT industry has really started focusing on EDA the **last 5-7 years**

- Might be, in part, a **reaction to overly connected** microservices architecture, and also the increase in coupling between systems caused by RESTful APIs

According to Gartner, this year, 2023, “over 50% of business organizations will participate in event-driven digital business ecosystems” and “most leading providers of application platforms will include high-productivity tools for event-driven design”

Recent reply by Gartner’s leading EDA analyst: “There is a lot more adoption and the **need for continuous intelligence** is more important. Also the popularity of Kafka.” – Yefim Natis

Smals ICT for society

EDA 101: What is Event Driven Architecture?

Life is more than sunglasses and hit movies.

Reality - that's the main Event.

– Sylvester Stallone

Smals
ICT for society

EDA 101

- A Different way to Communicate / Integrate
- A Different way to Represent Reality
- Definitions
- Technology, Community

Smals ICT for society

A different mode of Communication

Synchronous Communication

Two-way

waiting for immediate reply

Asynchronous Communication

One way

not waiting for reply

Smals ICT for society

Add ***one-to-many*** and you have… EDA

Direct Call

direct, synchronous

two-directional, ***1 receiver***

Event Send and Receive

indirect, asynchronous

unidirectional, ***many receivers***

Smals ICT for society

APIs: the Golden Hammer

The **RESTful API standard** remains the standard way of Integrating systems… ***synchronously***

But they are so useful they blind us to other options

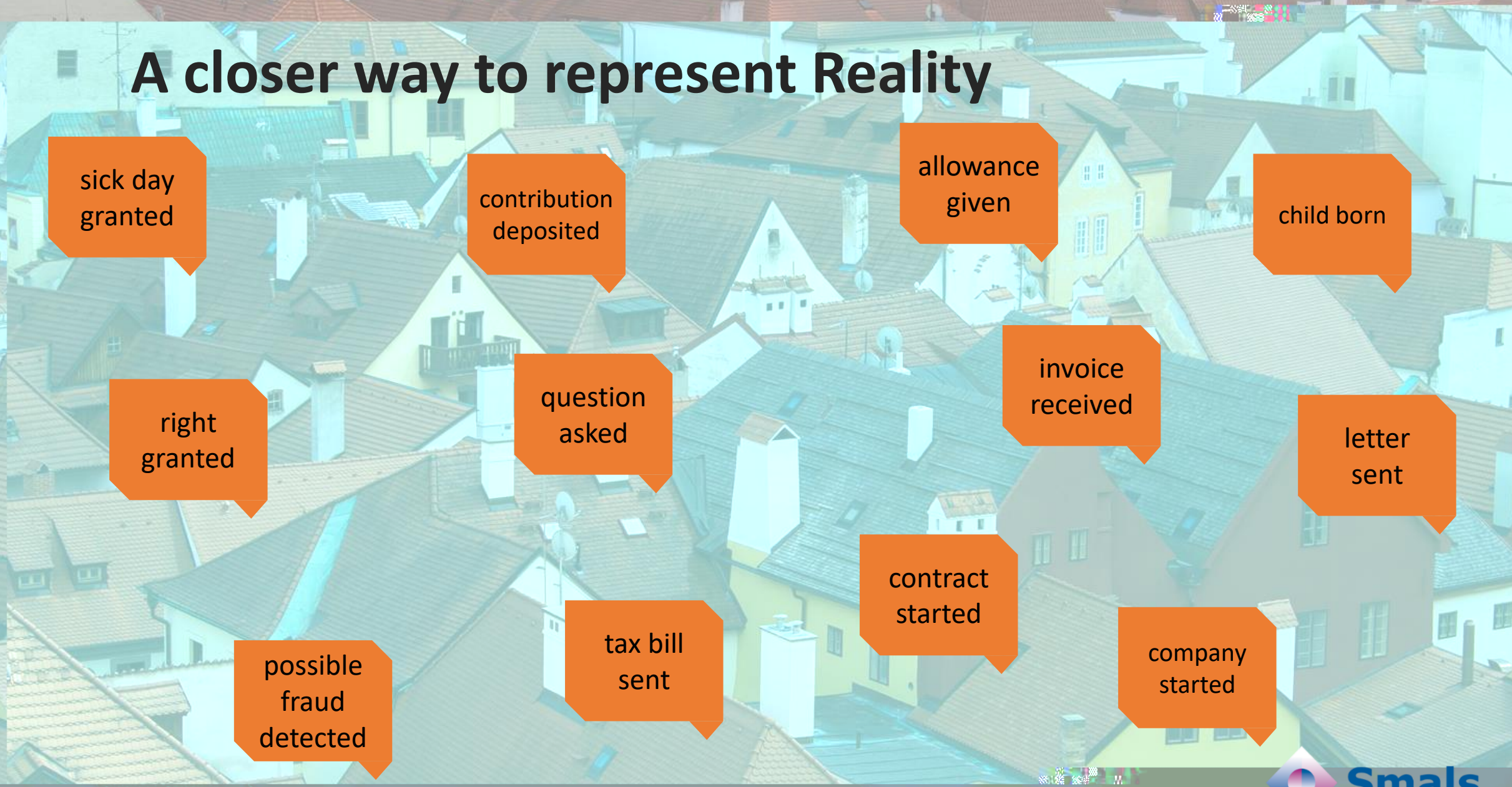
"With a good hammer, every problem looks like a nail…"

Often, a screw driver would be better

Let's consider a **FULL integration strategy**, giving equal importance to EDA

Smals ICT for society

A closer way to represent Reality



Smals ICT for society

Business Events: the prime drivers of Data

Currently, all **data is derived from Events in the real world**.

Representing that data as the actual Event brings the IT system closer to the real world.

I believe we should keep it in that form as long as possible, and only **transform it to "regular" data when a business process effectively needs to** act on it.

Keep these Events around… you never know when it would be useful to have a look at where your data came from.

Smals ICT for society

Event: a Definition

- An Event in the real world (technical events also exist)
- … captured by an IT system
- Given a type and a timestamp
- Often with extra data about the Event (“payload”)

begov.rsz.work.salary.SalaryPayed

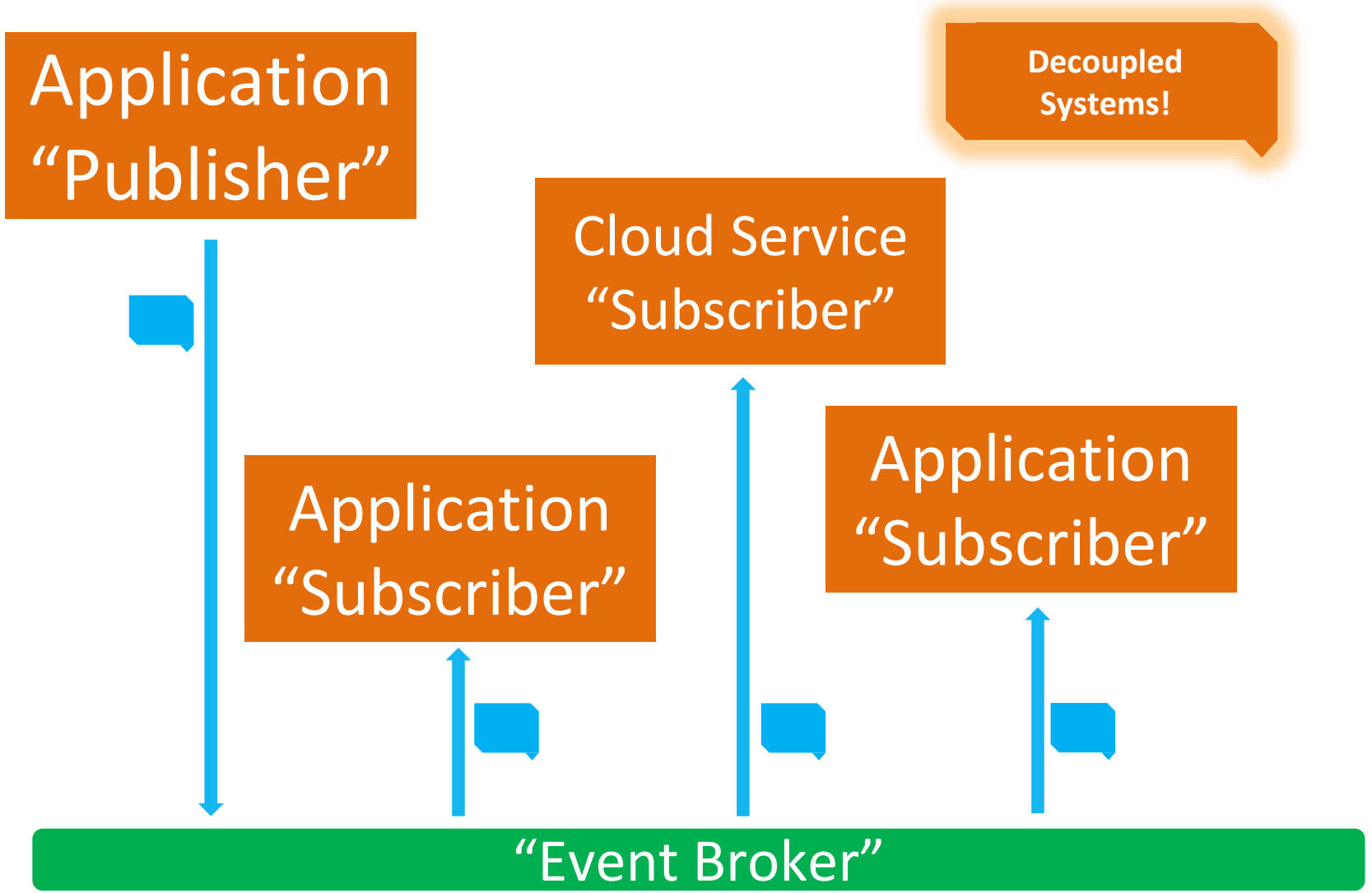
- **Time:2022.04.01-15:43:21:321456**
- **Employer: 0406.798.006**
- **Employee: 881122.555.77**
- **PaymentDate: 2022.03.31**
- **Bruto: 8000**
- **Netto: 4000**
- **WorkPeriod: {…}**
- **…**

Smals ICT for society

Event Driven Architecture

Software Architecture Paradigm, centering on the creation, transmission, and handling of business Events

Basics: Publish/Subscribe mechanism



Events are:

- Sent by an IT system as "publisher" of the event
- Received by other, interested IT systems: "subscribers"
- Via an "Event Broker"; a middleware IT communication system with high availability

Smals ICT for society

EDA Technology

- Event Broker (Apache Kafka, Azure Event Grid, …)
- Event Definition Standards (AsyncAPI, CloudEvents)
- Libraries & Frameworks (AxonIQ, Dapr, …)
- Overarching Integration Suites (Solace PubSub+ “event mesh”)
- API Gateways with EDA support: Event Driven APIs (Gravitee, Axway Amplify Streams, …)
- Auxiliary concepts (webhooks, SSE, …)
- …

AxonIQ

gravitee.io

Smals ICT for society

EDA Community

- https://www.eda.community/

The Event Driven Manifesto

- https://cymo.eu/blog/eda-manifesto

1. Think Business First
4. Topics and Event schemas define the contracts
8. Stay Focused on the end-to-end Process

Smals ICT for society

Zooming out again… Let's talk About IT Architecture!

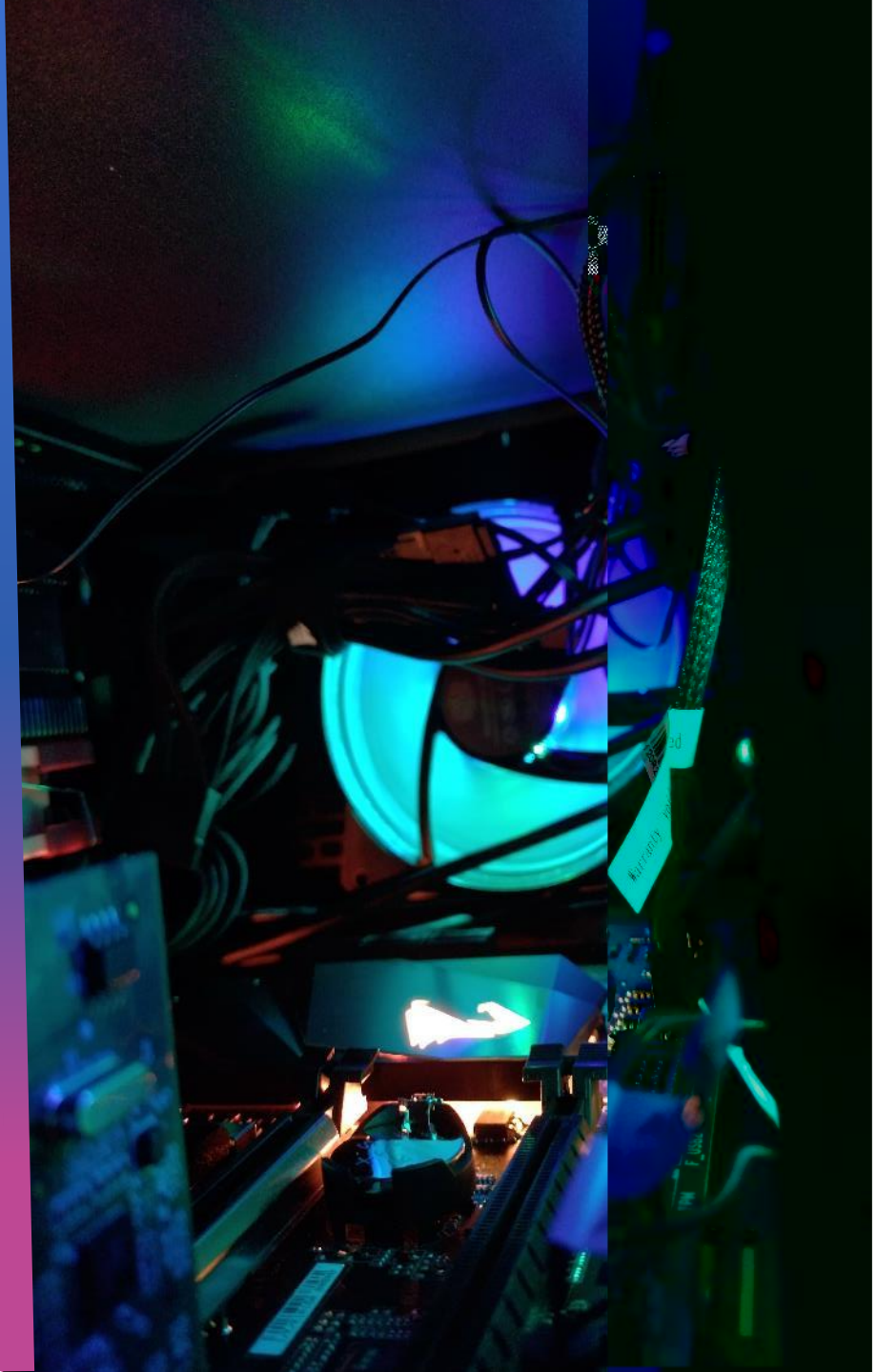
There's a way to do it better.

Find it.

– Thomas Edison

How can EDA help to modularize and make IT more future-proof?

Smals ICT for society

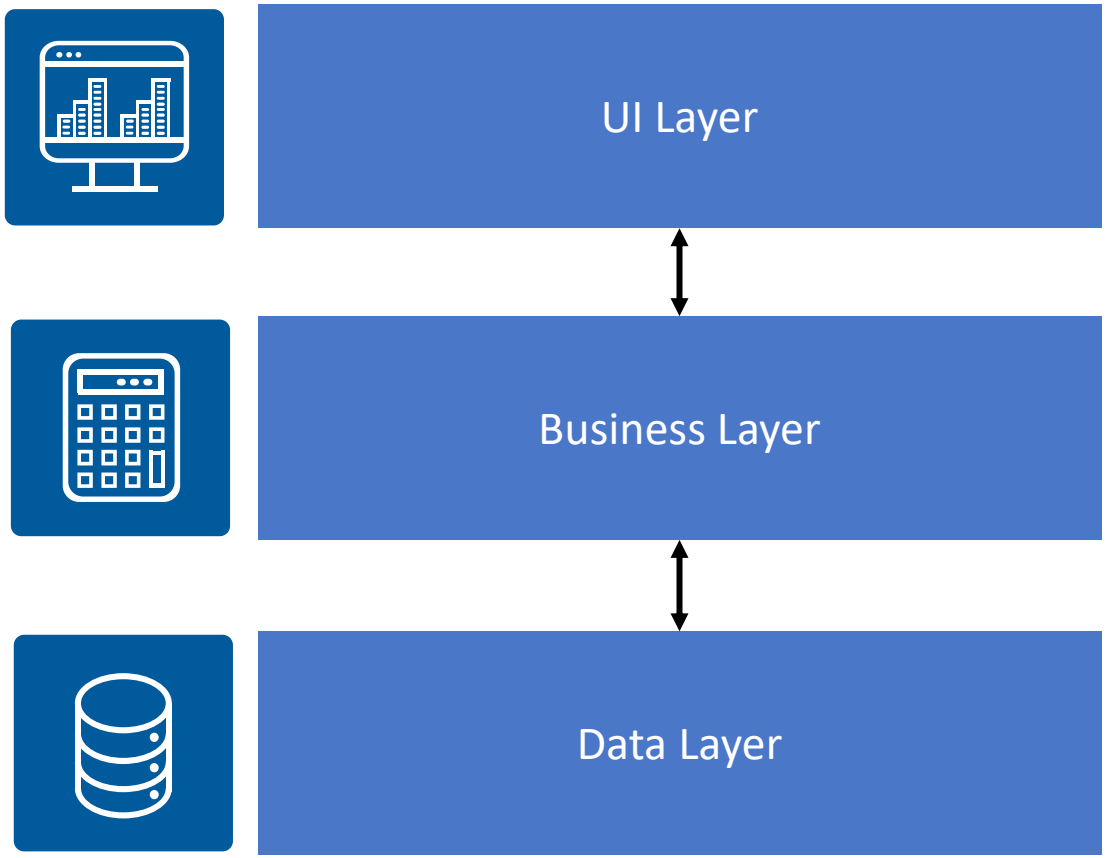


IT architecture: The “HOW”

- 3 Layer Architecture and the introduction of APIs
- Distributed Systems and the CAP theorem
- Layered Ecosystems
- EDA in the Service Layer, Eventual Consistency
- Hexagonal Design, Domain Driven Design and Event Storming
- What about microservices?

Smals ICT for society

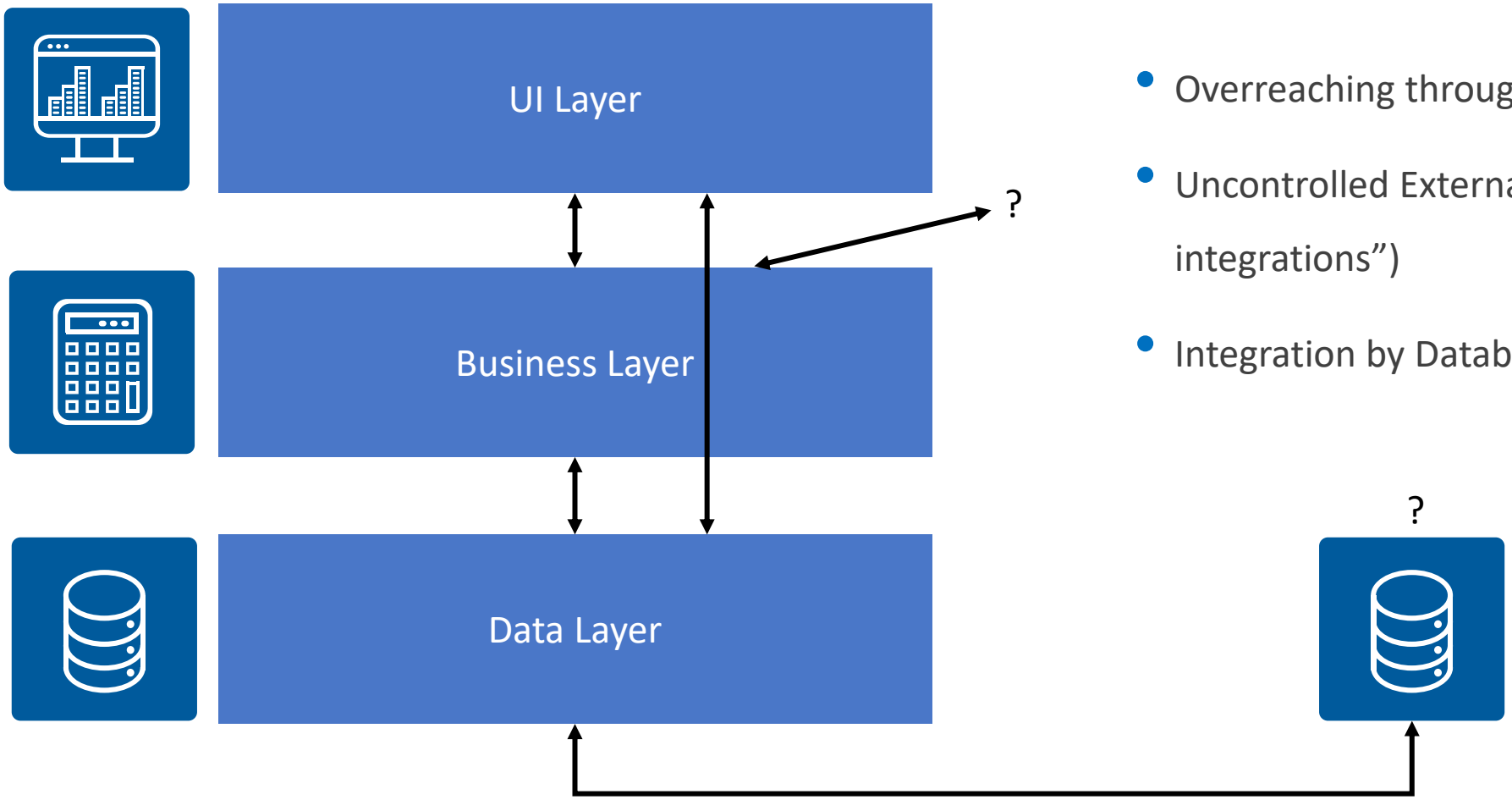
Are you building with 3 tiers?



- Increasing **modularity**
- The traditional architecture that came to be when people were sick of spaghetti code
- A good start: these **layers have very different responsabilities** (concerns)
- “Separation of Concerns”
- Higher cohesion, coupling a bit lower

Smals ICT for society

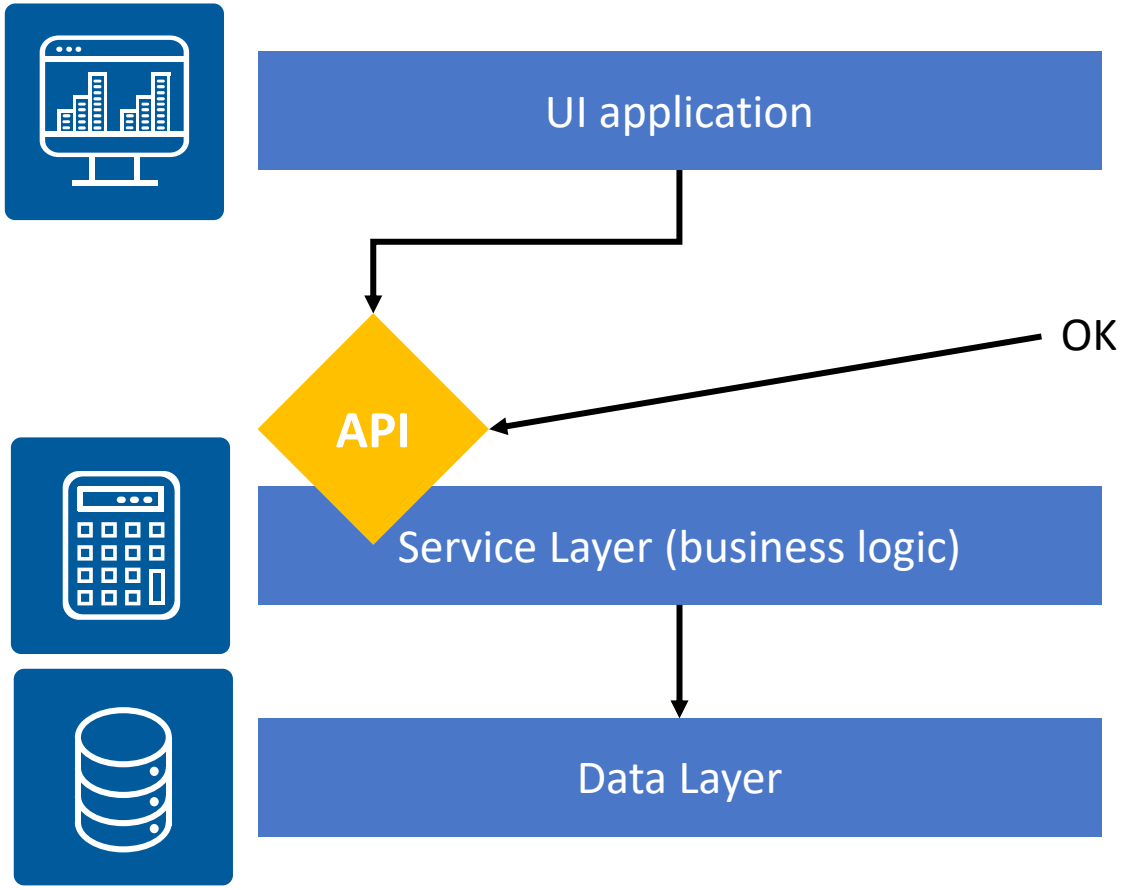
Some problems with 3 tiers…



- Overreaching through the layers
- Uncontrolled External Access (“wild integrations”)
- Integration by Database (that includes batch !)

Smals ICT for society

Enter the API



- Rule: applications can only be **accessed by their API**
- Only higher layers can initiate the interaction
- Only an application can use its own database
- They are now called "**services**"
- APIs become **reusable** entities (if cataloged)

- → Service Oriented Architecture

Smals ICT for society

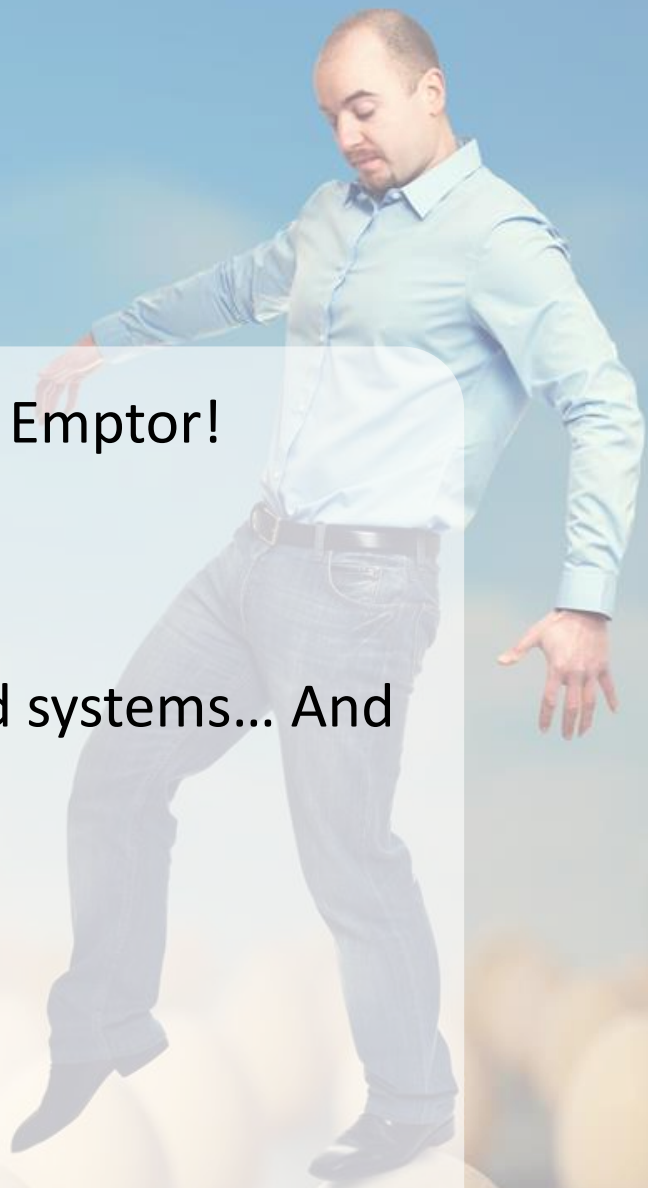
What are the Hardest Things in Software Design?

There are only two hard things in Computer Science: cache invalidation and naming things – Phil Karlton

There are only two hard things in Computer Science: cache invalidation and naming things and one-off errors and cache invalidation – K. Alan Bates

Smals
ICT for society

And in IT EcoSystems?



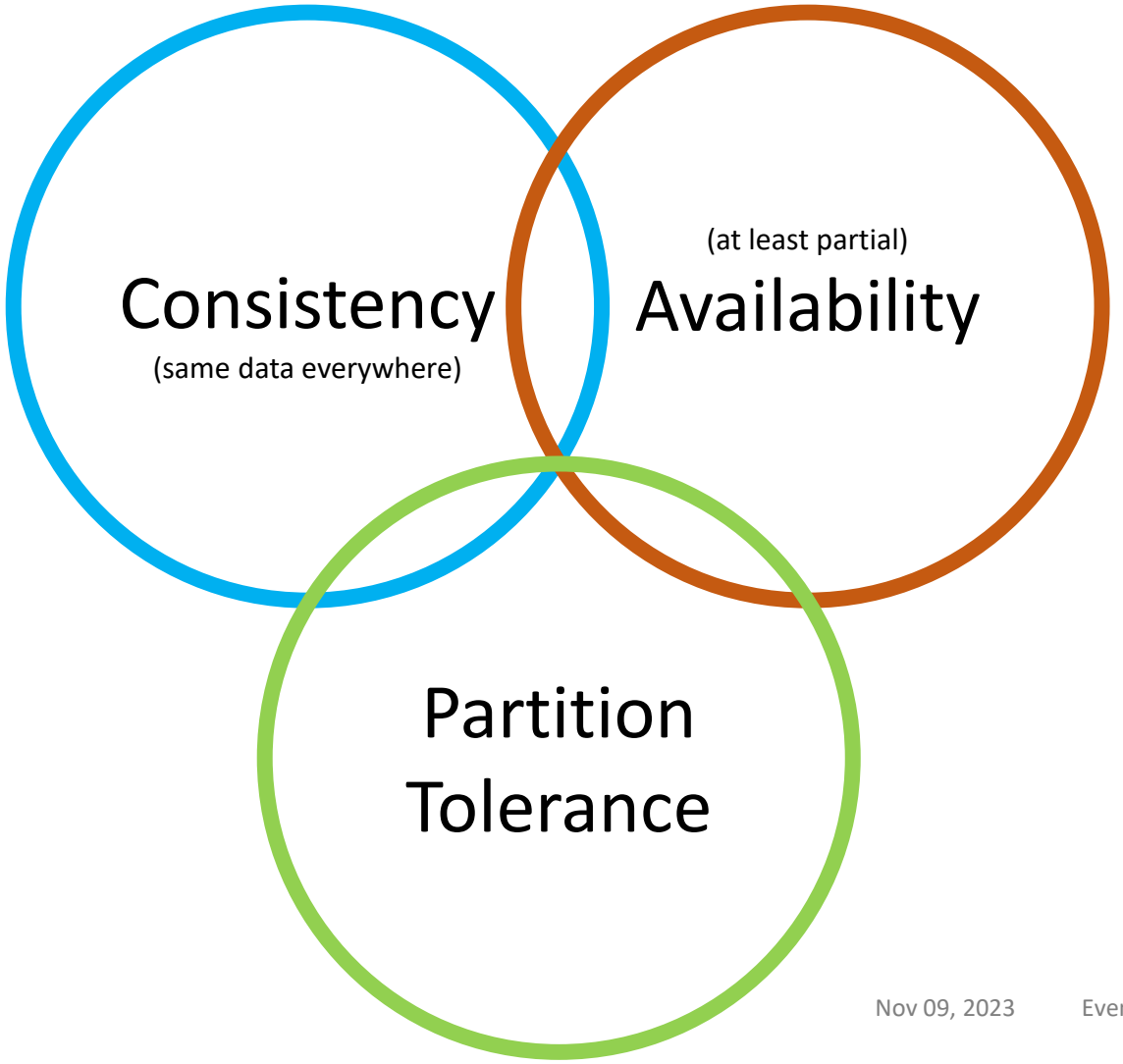
… Caveat Emptor!

Lots of things are hard, especially distributed systems… And we have a giant ecosystem…

→ **Integration** is often the biggest problem

Smals ICT for society

Why are distributed systems hard? The CAP theorem



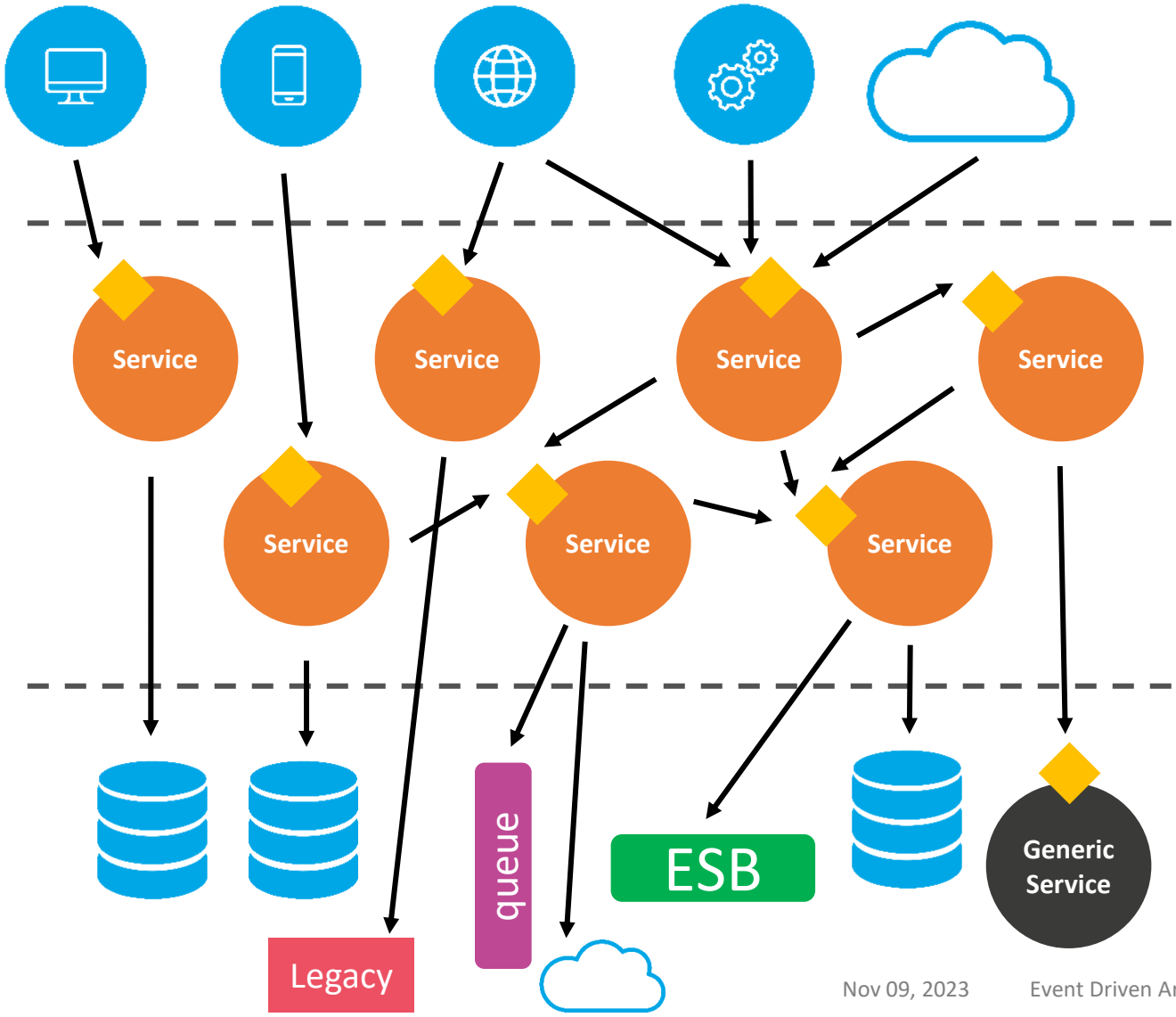
- You can't have all 3. Pick 2
- When Integrating Systems, you **always need "Partition Tolerance"** = tolerance to systems being unable to communicate (e.g. network failure)
- This means we have to make a choice:

Consistency or Availability ?

(typically too much focus on consistency…)

Smals ICT for society

A Layered Ecosystem! (quite similar to the application)



Client Layer: Frontend & External Systems

Web, mobile, speech, IoT, services, partners

Service Layer

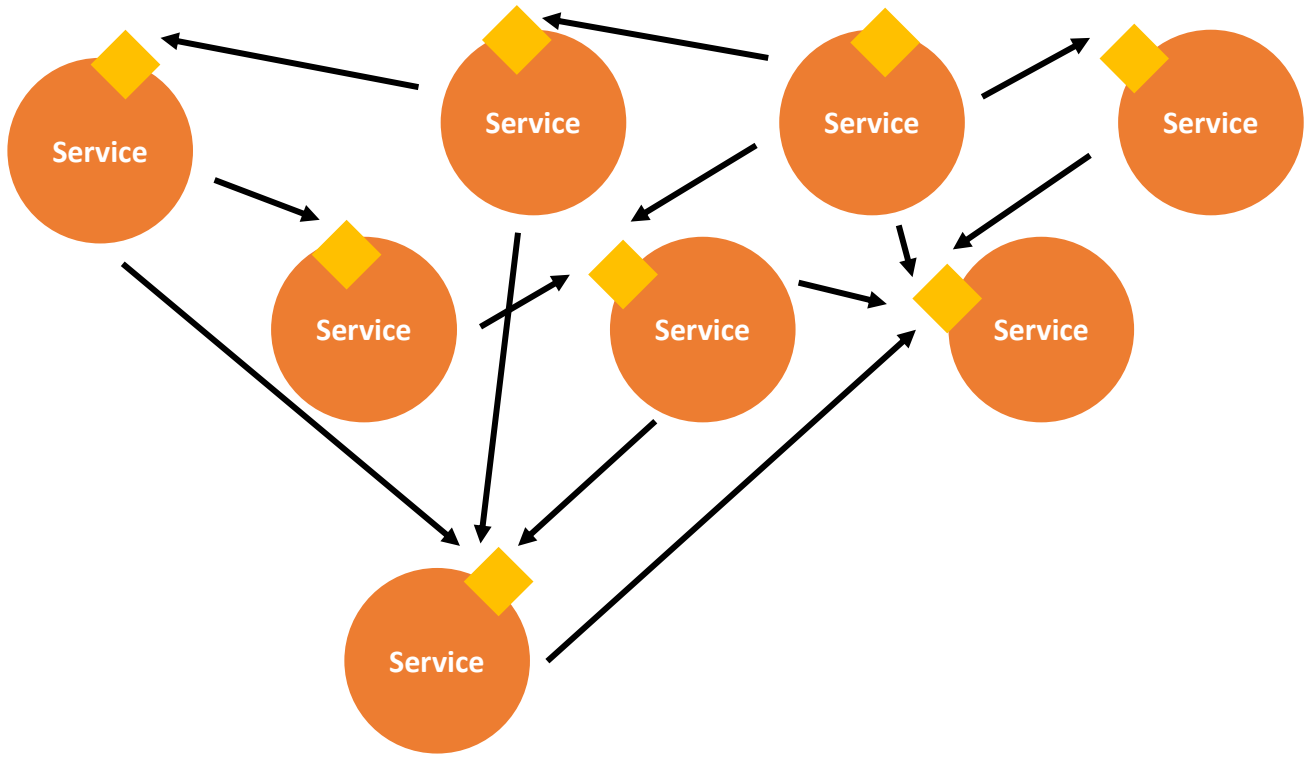
Business Services: added value !

Infrastructure Layer

Generic Services (e.g. archival, pdf creation, …),
Databases, Middleware, Integration Solutions, …

Smals ICT for society

Let’s focus on the Service Layer

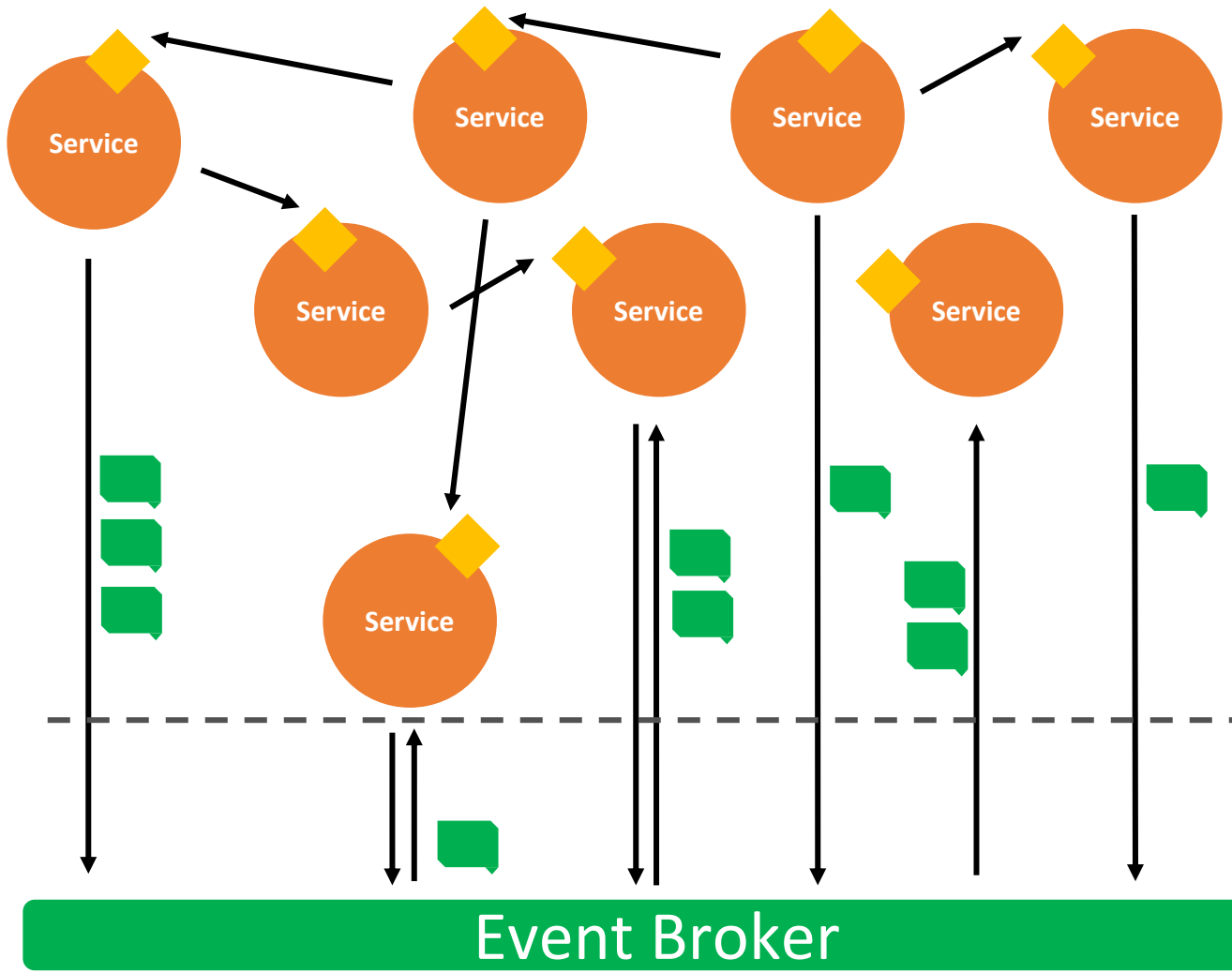


- Modularization → many small services
- High cohesion
- (Business) **need for integration** → many interlinks
- High coupling
- Possible cascading failures
- Need for (central control)

This is where EDA can take things a step further…

Smals ICT for society

EDA for Lower Coupling



- Start replacing synchronous, direct calls, with asynchronous, indirect Events
- Replace interdependencies with the use of a new **infrastructure**: the Event Broker
- lower coupling
- Events become **reusable** entities (if cataloged)

replace orchestration with
(decentralized control)

Infrastructure Layer

Smals ICT for society

Revisit CAP: "Eventual" Consistency

- EDA reduces coupling; **increasing independency** of services and reducing cascading failures

→ increased availability (and other advantages)

- We don't wait for an answer when we send an event: we **no longer perform a *transaction***

→ consistency is ***delayed*** (but reached)

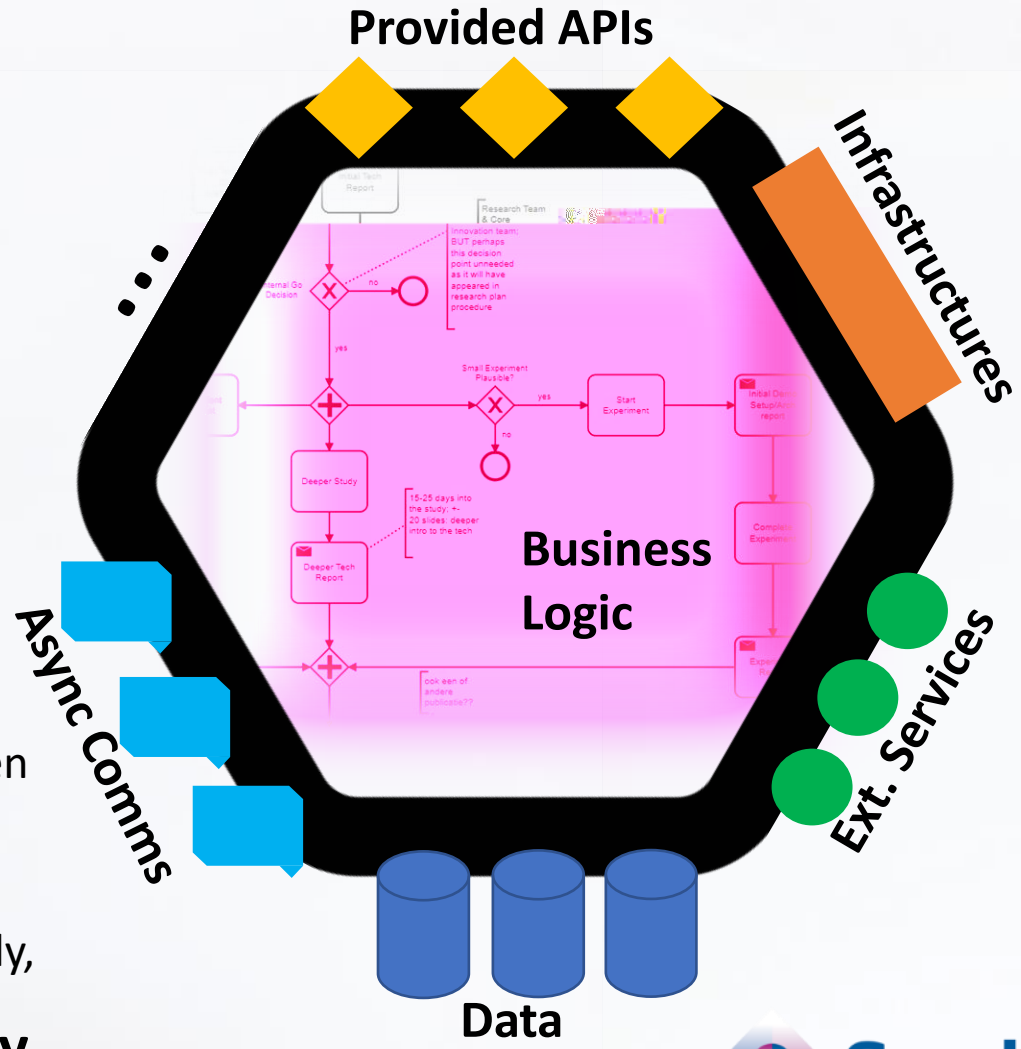
Typically, a lot of business processes can handle a small delay in consistency. Data is **propagated via Events behind the scenes to ALL** interested systems/parties, potentially triggering numerous other automations, and reaching ***consistency across the ecosystem*** in potentially minutes (if not mere seconds)

Smals ICT for society

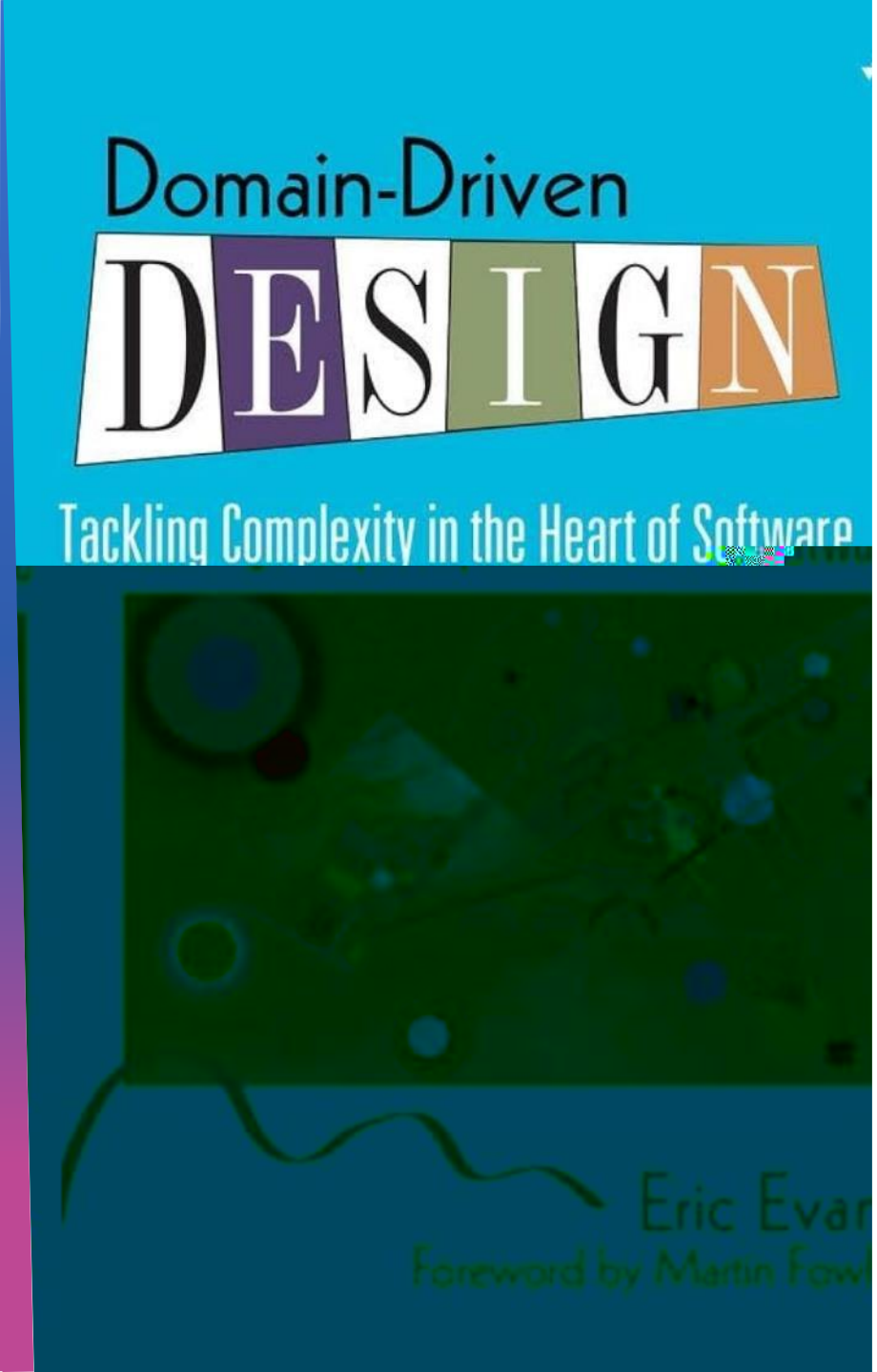
Let’s get back to the design of ONE service

“HEXAGONAL” design

- A.k.a.
- Goal: **keep business logic independent** from all externalities / other concerns
- The hexagon can act as an “anticorruption” layer between all the involved systems
- Ultimately, everything should be able to evolve separately, and plugged in/out easily → **Increased Modularity**



Smals ICT for society



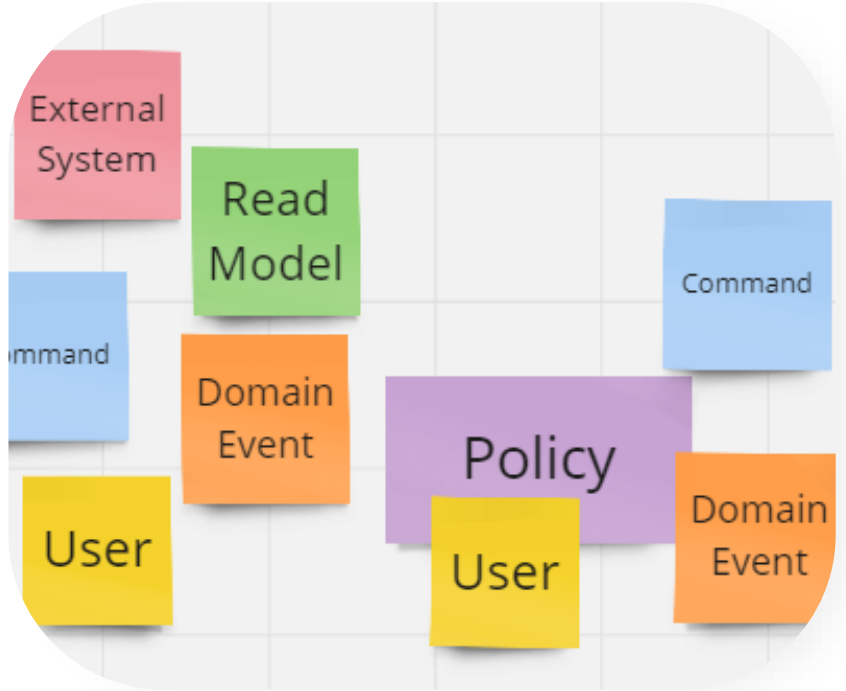
Domain Driven Design

- Design method; book published in 2003
- Focus on : how to organize large business **domains into separate subdomains**, called Bounded Contexts; these are typically quite small; not the entire business logic of an application
- **Bounded Context**: Logical, cohesive subset of the business domain that should be considered a single module
- Also about the **ubiquitous language**: a common language between business and IT, embedded into everything from analysis to the programming code (can be different in each bounded context)

Smals ICT for society

Event Storming

- Method for finding the **domain Events** in your business, **as a Lingua Franca** for describing the business and implementing its supporting software
- Focus on ***behaviour* first**! → Not data, which risks your design being centered around an ER (database) scheme
- Eventually, it also helps you to identify commands, users, … and altogether the **bounded contexts** within the business domain
- Starting with this method @analysis can lay a lot of the groundwork for a **good, modular, future-proof architecture**,

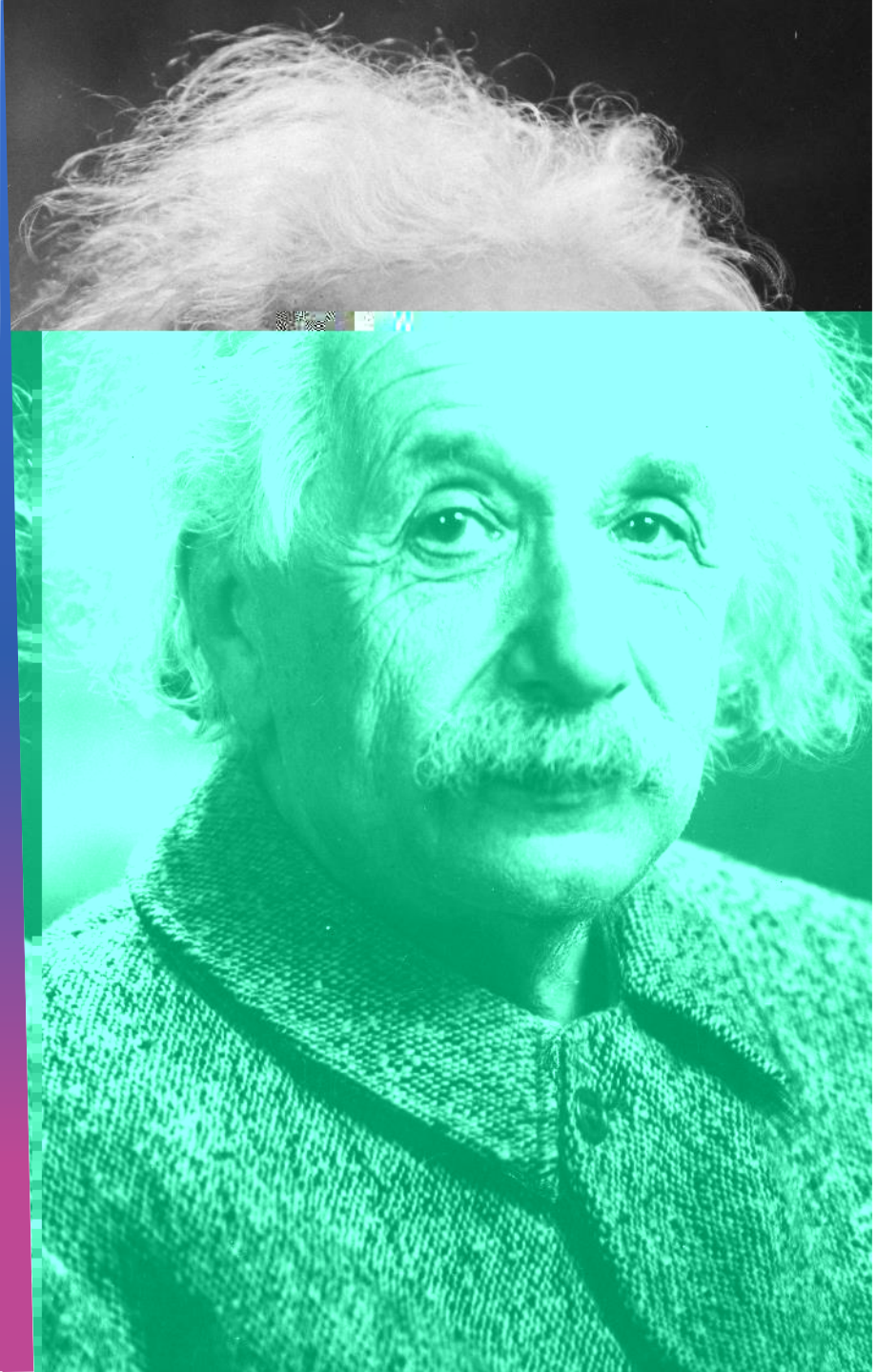


Smals ICT for society

Does Complete Modularization = Microservices?

NO (we should be practical)

- We can design ***a good monolith*** and respecting all principles (high cohesion, low coupling, …)
- The choice between monolith and microservice is not made @design level, but **@deployment** level, i.e. where do we effectively run which part of our application ?
- Separating a part of the system out of the monolith and deploying it independently, can be done as needed for scalability and performance reasons. If the part we take out is **really small (and very independent), we call it a microservice**
- Too many microservices can make an ecosystem **harder to manage**



EDA 201

Everything should be made as simple as possible,

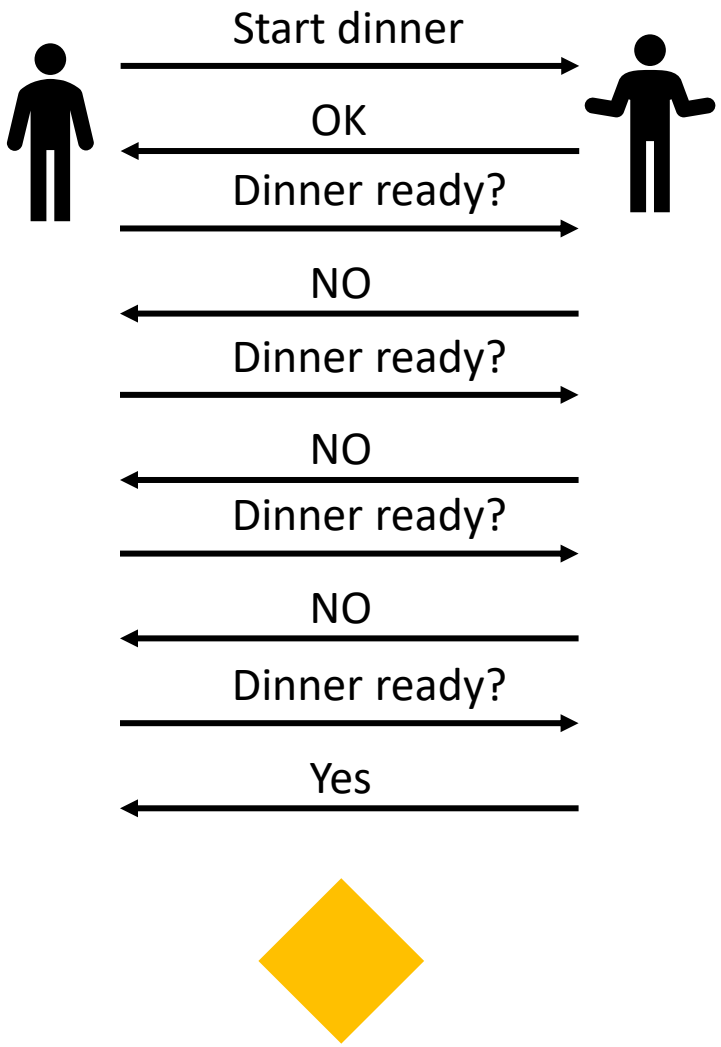
but not simpler

– Albert Einstein

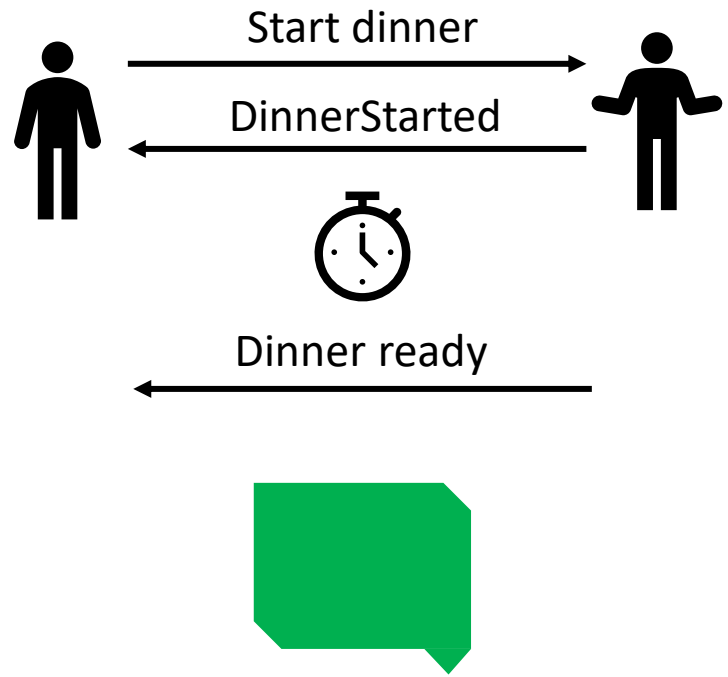
4 Different styles of EDA usage

Smals ICT for society

1. Event Notification



- Subscribers are being made aware of certain Events and state changes by publishers
- Very useful to **stop polling** style of communication
- Event only has type and time-stamp (and perhaps publisher ID)
- **No payload** with other information: subscriber still has to ask (less decoupling)



Smals ICT for society

2. Event Carried State Transfer

begov.podmi.ocmw.**LivingWageGranted**

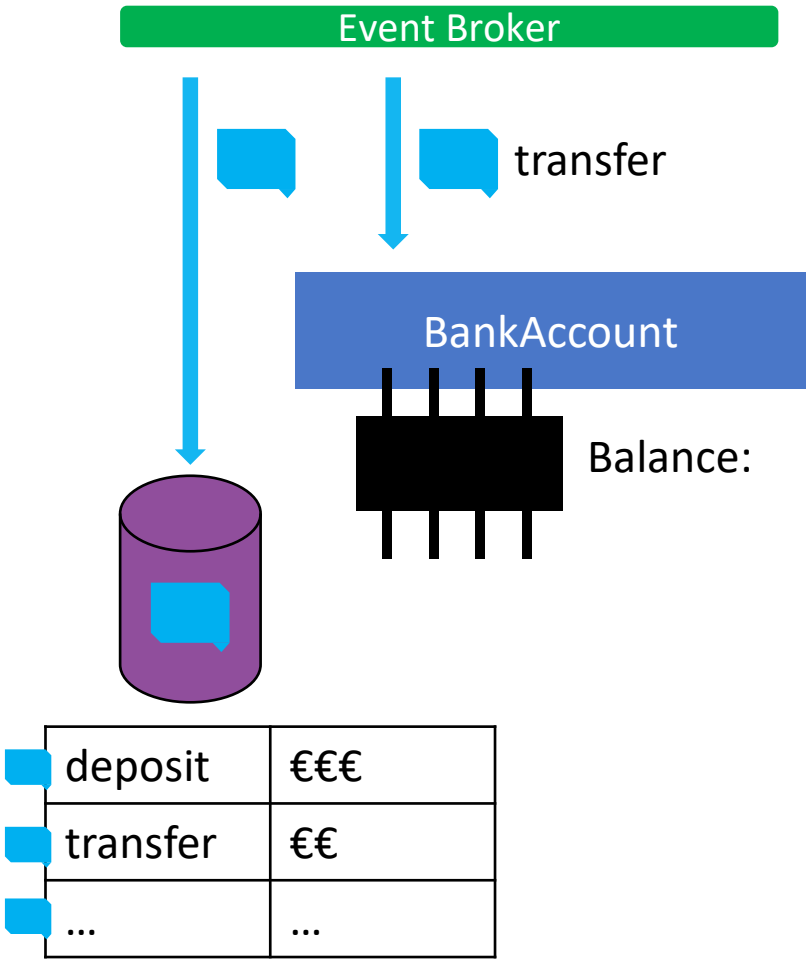
- ❑ **Time:2023.10.21-15:43:21:321456**
- ❑ **OCMW: 3300**
- ❑ **Grantee: 881122.555.77**
- ❑ **StartDate: 2023.10.31**
- ❑ **Amount: {EUR; 809,42}**
- ❑ **…**

- Lots of relevant information added to **Event Payload**
- **Distribution** of timely, useful data, across many interested stakeholders in the **Ecosystem**
- **Decoupling** between publishers and subscribers
- : allow all stakeholders to construct their own view of reality

untapped potential

Smals ICT for society

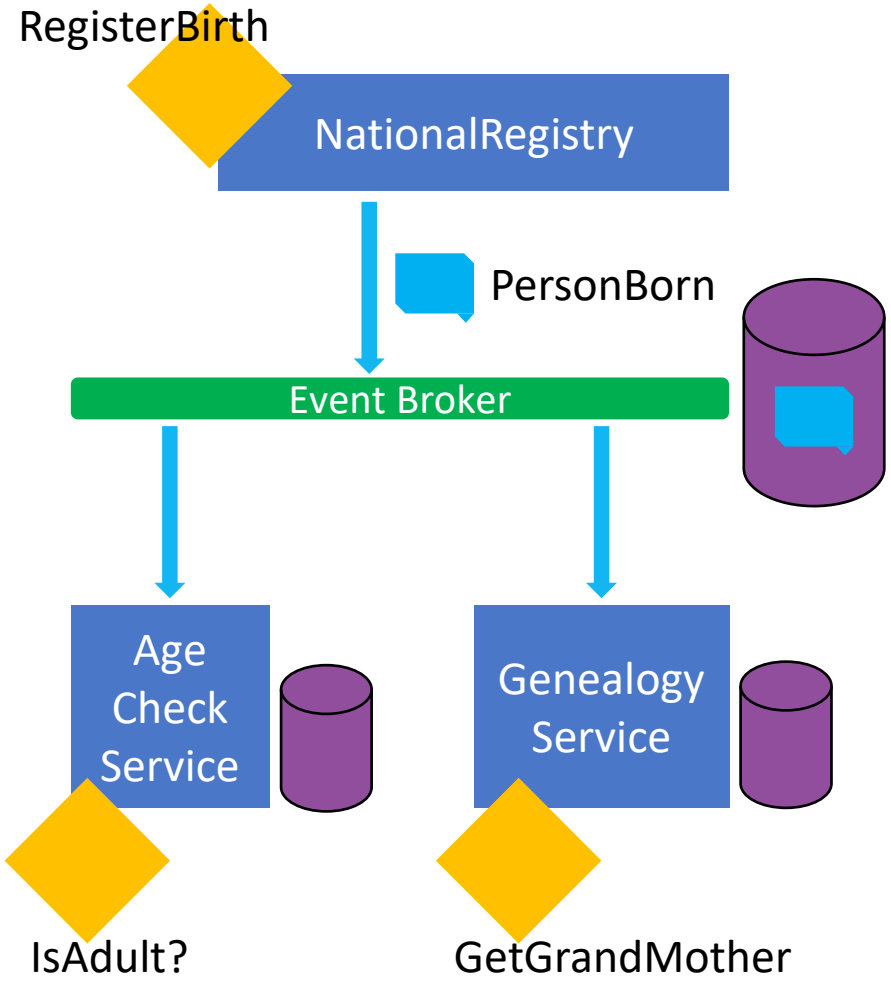
3. Event Sourcing



- Application **storing all Events as its main source of truth** (instead of only its current state to a database)
- Current state is “calculated” from the Event history
- Strictly implemented, the **Event store only grows**: corrective events may be necessary
- Useful for history reconstruction, audit logging, testing new versions, analytics, ...

Smals ICT for society

4. Command & Query Responsibility Segregation



- **CQRS**
- Receiving Commands (**causing state changes**) is done by a different module than handling queries (**only asking for information**)
- Modules for answering queries can use their own internal state
- If deployed independently, query services can scale separately as needed

Smals ICT for society

EDA Advantages, Examples

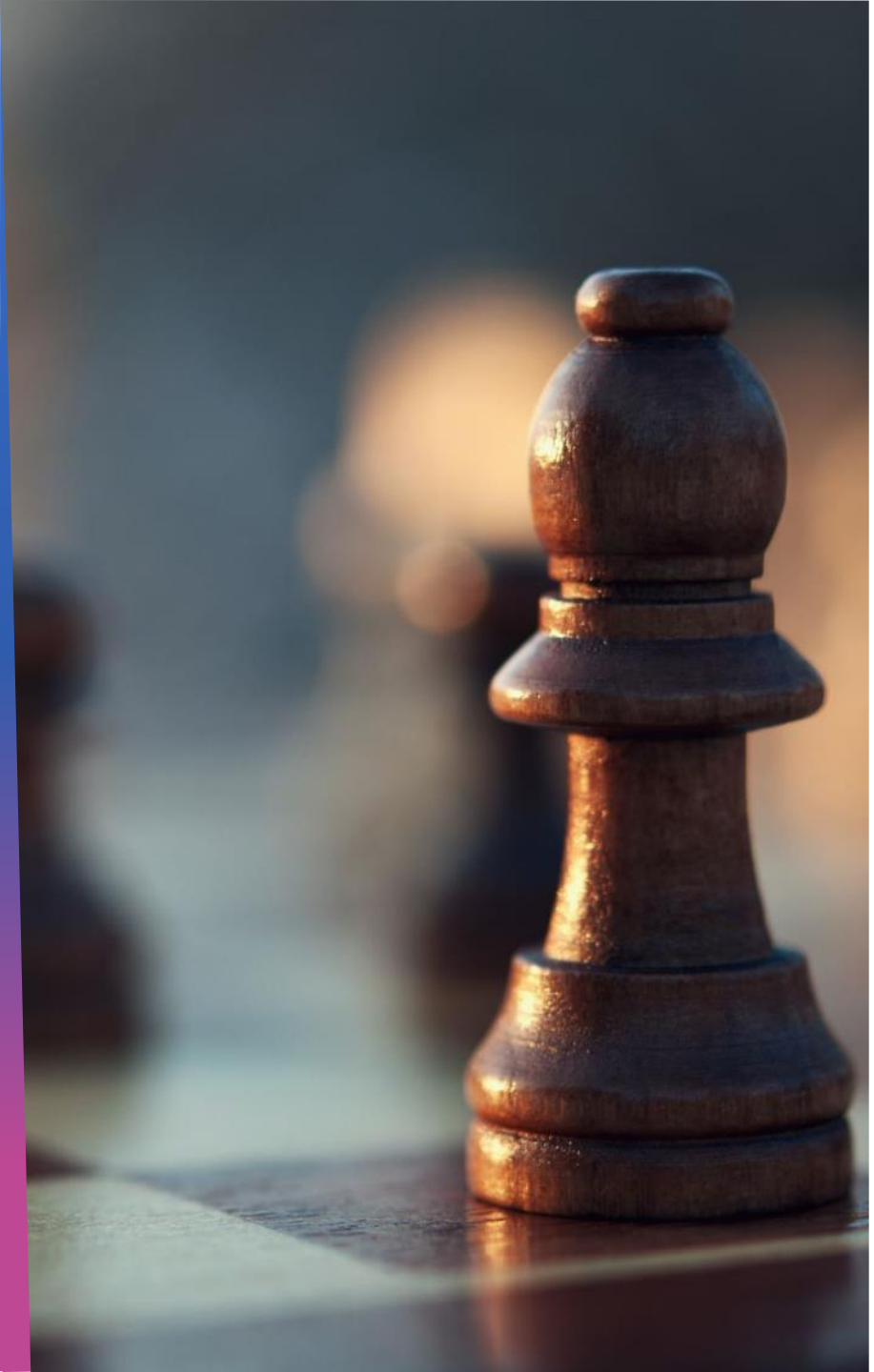
Don't be afraid to give up the good to go for the great.

– John D. Rockefeller

What are the advantages of EDA for **resilient, agile** automation ?

How does EDA get the data everywhere in real-time, so we have **once only** and **no wrong door** ?

Smals ICT for society



EDA advantages & examples

- Real-time eGovernment
- Agility & ReUse
- Resilience & other IT advantages

Disclaimer – high impact examples

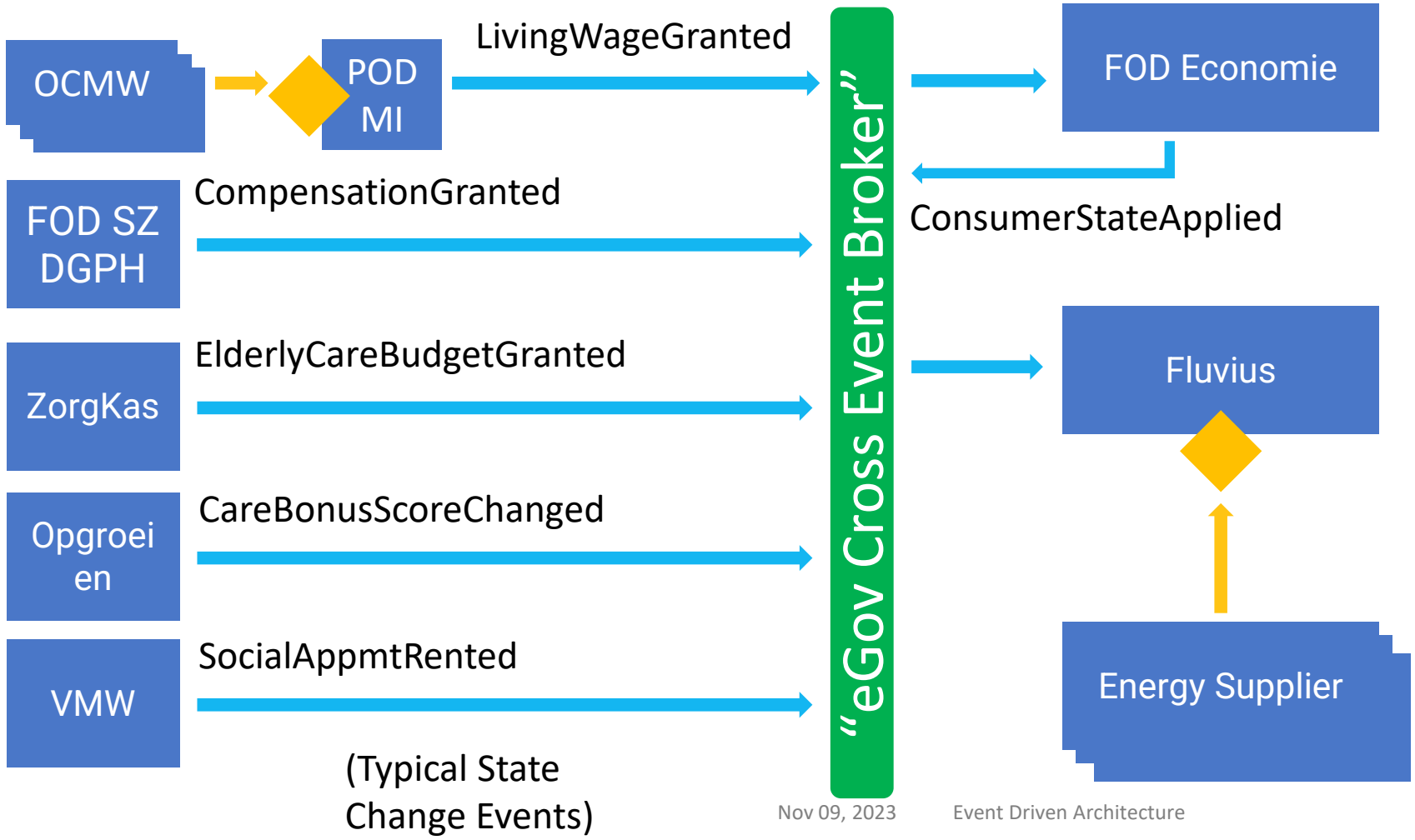
The examples are based mostly on how we could do things if we had an **ecosystem with a lot of EDA technology** and existing events in place. I realize the effort it would take to first create the necessary preconditions of such an event-driven ecosystem.

Smals ICT for society

The Real-time eGovernment

- Real-time Ecosystem, with events coming in from everywhere, representing **all important business events**
- Events going everywhere a.s.a.p. , to all systems requiring them, making **high-speed decisions and automations** feasible
- Real-time insights in the data, using **on-the-fly analytics** on the stream of events
- Pushing our own events to 3rd parties, delivering immediate feedback to companies, citizens and social benefit organizations

Example in Automatic Benefits: the Protected (Energy) Consumer (hypothetical, not exhaustive)



(Currently: Trimestrial Updates by FPS Economy in an incredibly shredded social landscape)

Smals ICT for society

Agility & ReUse

- **Minimize the impact** of changing requirements! Event Consumers can be added or removed without impacting Event producers**. Decoupled systems** can evolve and adapt independently to meet business requirements.
- Use Events to communicate between teams/domains, creating **natural boundaries based on separate business contexts**. (Think about Conway's Law.)
- Store your events, and retain all information with **potential future use**, "replay" Events to aid in simulating/testing.
- **ReUse existing Events** in new applications. If an Event is valuable to the ecosystem, publish it; even if nothing will consume it now.

Events are ReUsable Building Blocks (like APIs)

- Systems/Applications can send a number of Events of certain types. Many other systems can subscribe to get some of these types of Events.
- Standards exist to describe Events (e.g. AsyncAPI, same organization as OpenAPI for REST)
- API Management Software now often supports Event Driven APIs
- **Events ARE data**. APIs can offer data, but also functionality; their data is usually at least already partially processed. Events can be closer to the uninterpreted truth. So they fill a different niche.

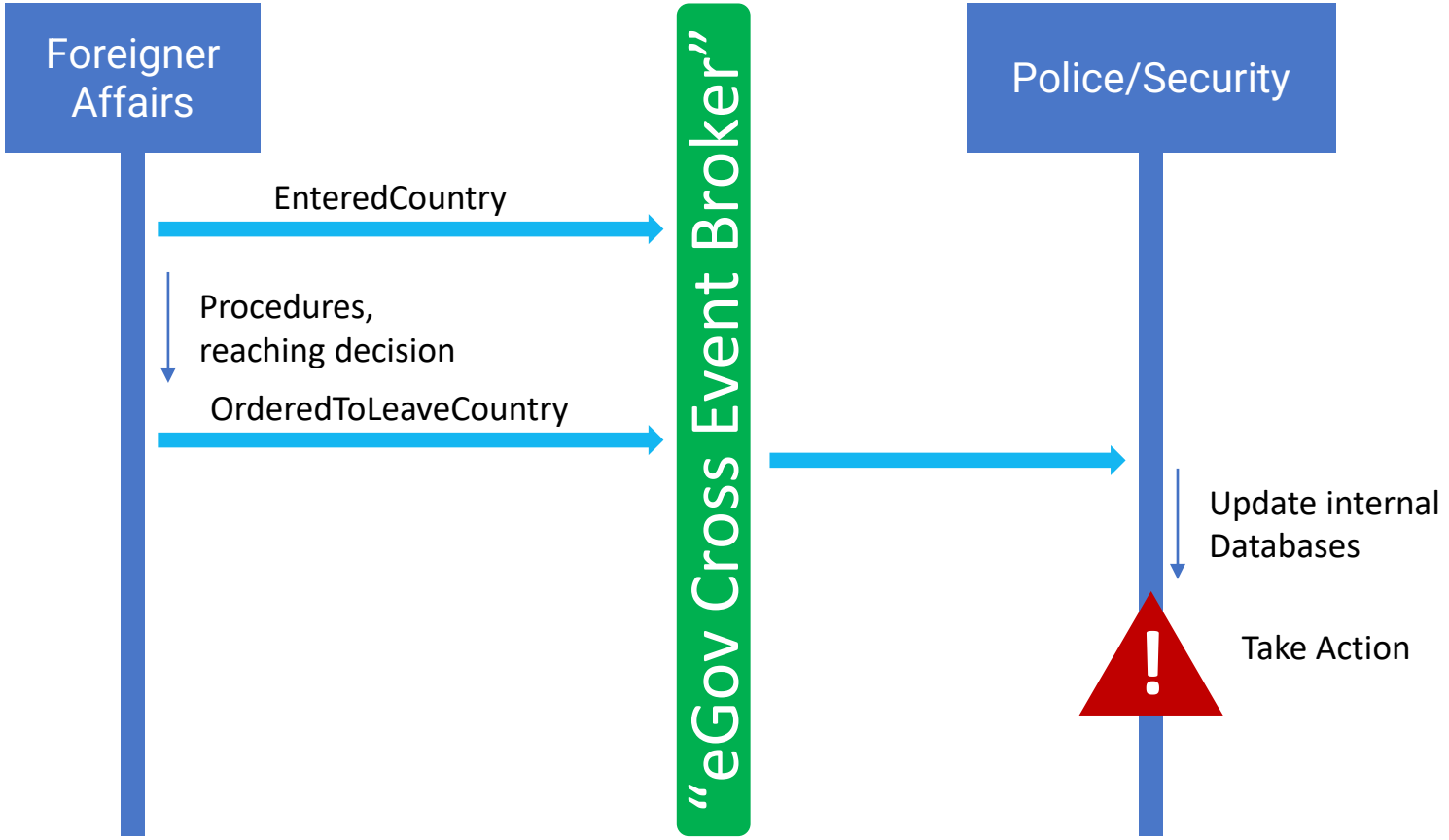
→**Many-to-Many** model of **reusable, timely Events to subscribe to** with your new shiny application

We should add these to the **ReUse Catalog** !  https://ict-reuse.be/

Smals ICT for society

Example: knowing about Banished Foreigners

(hypothetical, many assumptions)



Smals ICT for society

EDA: IT Drivers

- Increased **Resiliency** from decoupling and eventual consistency (CAP theorem)
- Stop “dis-architectures” using polling, or full batch transfer to update minor details: **performance** improvement and **green IT**
- EDA decoupling: Escape Service Interdependence, **avoid cascading failures**
- EDA decoupling: increase **modularity, scalability, maintainability**
- Flexibility, Composability (agility): easily add and remove Event Consumers without breaking producers → Increase **future-proofing**
- Align IT with the business using the Event as **representation of reality**

Let’s make it Happen!!

If I had asked people what they wanted, they would have said faster horses.

– Henry Ford

This is not business as usual. The vision (including certain examples) is very ambitious, and will take a lot of work to come true. How can we take some first steps towards it?

Smals
ICT for society

Where do we start? How to evolve EDA usage

- What are some practical options to do first?
  - Let’s discuss together in the workgroup!
- **But Most Importantly: The Mind-Set Shift**

Smals ICT for society

What first? Some options…

- PUBLISH your business events!
- Acquire a **full Event broker (HA, with persistence)** ?
- Get **Event-Driven API** support
- Get serious about Events in the entire software lifecycle (e.g. Event Storming)

→ Let's talk (find out who in your organization is involved in the workgroup ?)

Smals ICT for society

Most Importantly: the Mindset Shift

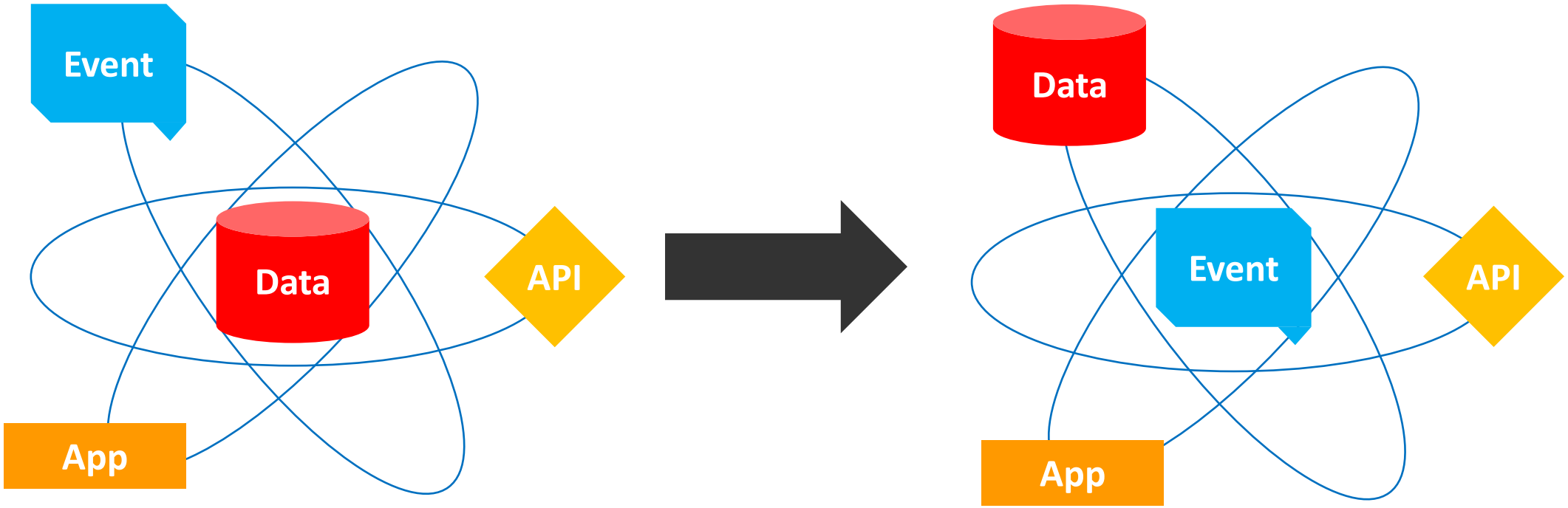
I have been impressed with the urgency of doing. Knowing is not enough; we must apply. Being willing is not enough; we must do.

– Leonardo da Vinci

- Using EDA makes a lot of sense, but we're not used to thinking about our business, our IT systems in terms of Events
- Traditionally: government **gathers and manages data, and then** uses it to implement policy
- Historically, eGov IT has evolved to simply make the above workflow digital, with minor automations

Smals ICT for society

Mindset & Role Shift



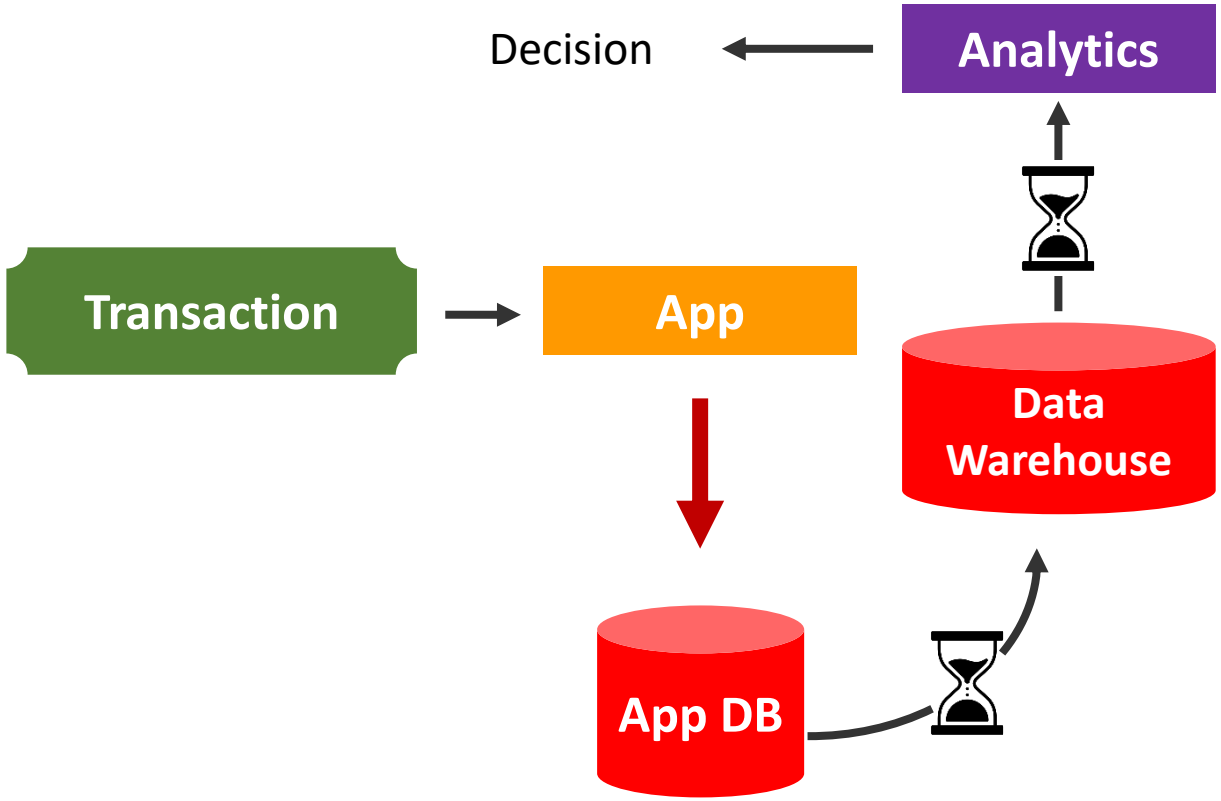
IT as a Guardian of Data
“Data Custodian”

IT as Digital Business nervous system
“Progress Engine”

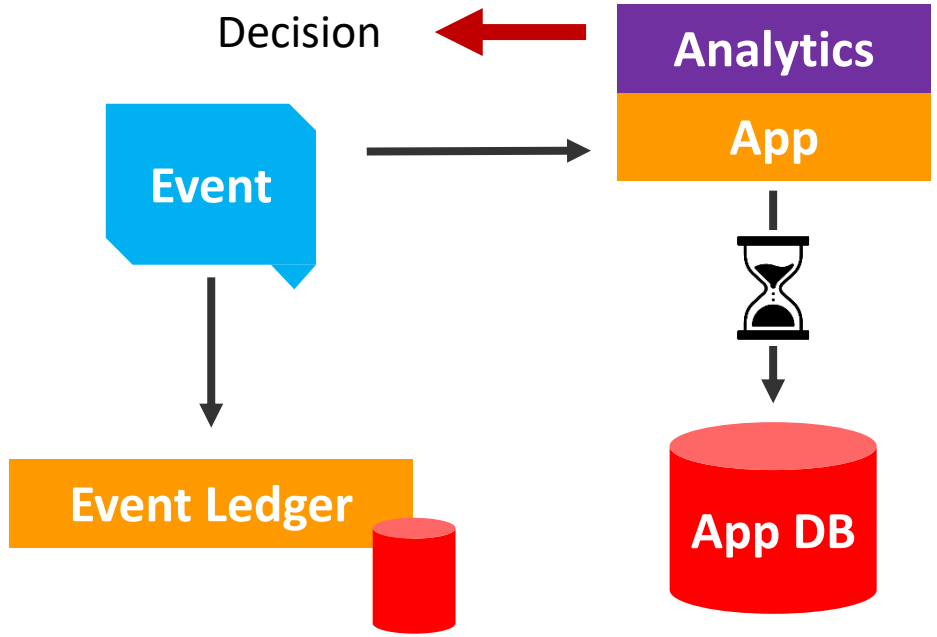
This culture shift is THE greatest challenge

Smals ICT for society

Mindset & Priority Shift



Continuous State; Eventual Intelligence



Continuous Intelligence; Eventual State

Smals ICT for society

Conclusion

The best time to plant a tree was 20 years ago. The second best time is now

– Chinese Proverb

To eventually achieve the vision of IT as the agile automator, using EDA, we should start with three things…

Smals ICT for society

Main Take-Aways:

- Keep learning about EDA (ask me!) and **Change your Mindset!** Don't be a custodian of data, be a **Proactive** engine of progress!
- **Prioritize EDA** at the same level as API for a Complete Integration **Strategy**
- Get **a sense of urgency**. Most of us are behind in our adoption of EDA. Start Now!

Smals ICT for society

Thank you !

If you have any questions, do not hesitate to contact me

koen.vanderkimpen@smals.be

Smals
ICT for society