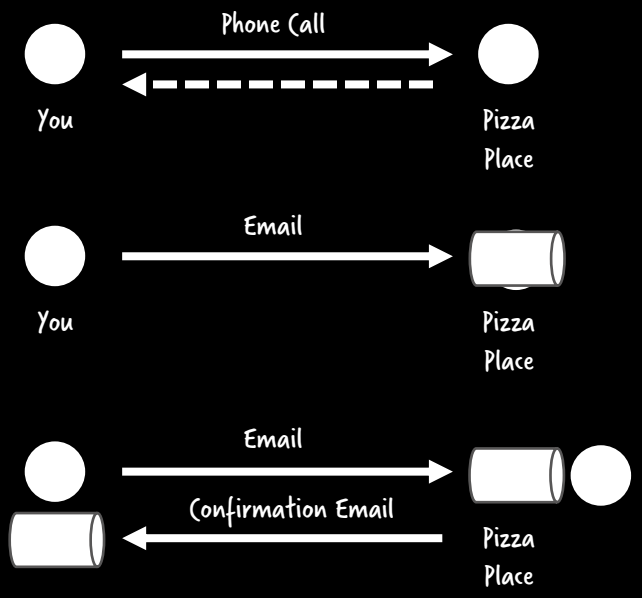
Are you done yet?
Mastering
long-running processes
in modern architectures

@berndruecker

Let's talk about food

How does ordering Pizza work?



Synchronous blocking communication
Feedback loop (ack, confirmation or rejection)
Temporal coupling (e.g. busy, not answering)

Asynchronous non-blocking communication
No temporal coupling

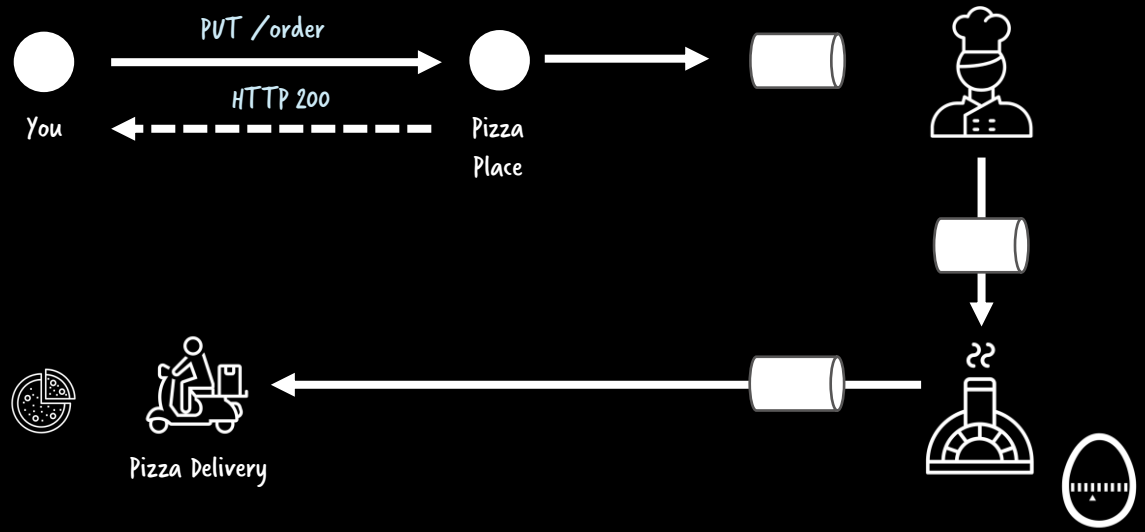
A feedback loop might make sense
(ack, confirmation or rejection)

Feedback loop != result

Feedback (ACK, confirmation, rejection)

Result

only the first communication step is synchronous / blocking



The task of Pizza making is long running

Synchronous blocking behavior for the result?

Bad user experience
Does not scale well

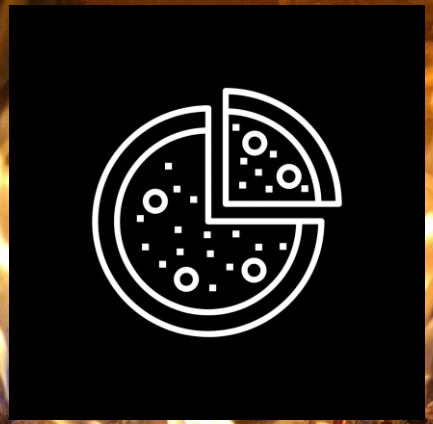
Scalable Coffee Making

https://www.enterpriseintegrationpatterns.com/ramblings/18_starbucks.html

Photo by John Ingle

Long running

Long running




Long running basically means waiting

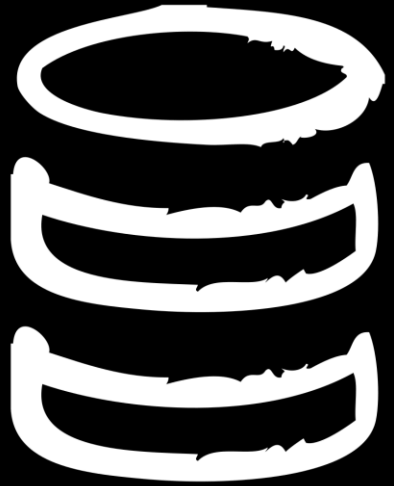
When do services want to wait? Some business reasons...

Human work

Waiting for response

Let some time pass

Why is waiting a pain?



Persistent state

Monitoring & Operations

Versioning

Scheduling & Timeouts

Visibility

Domain Logic

Scalability & Resilience

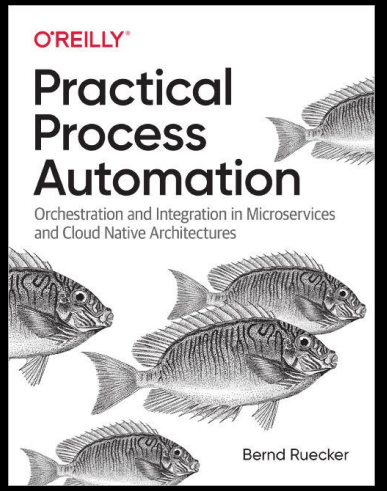
How to solve the technical challenges without adding accidential complexity?

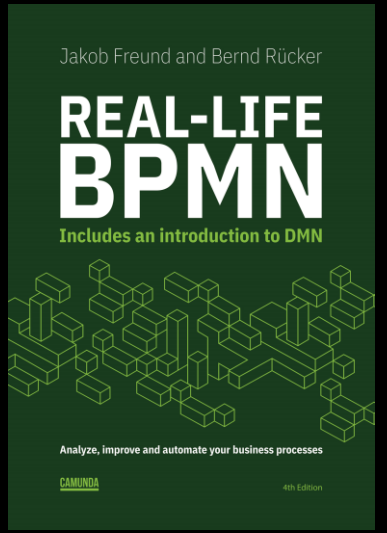
Warning:
Contains Opinion

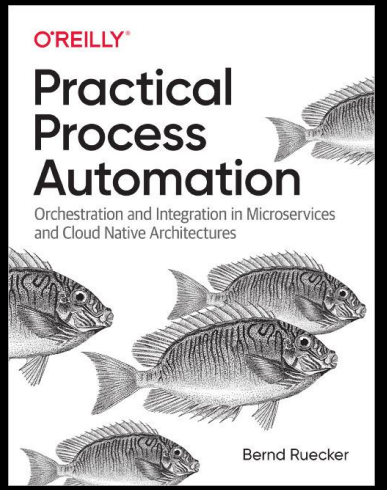
Bernd Ruecker

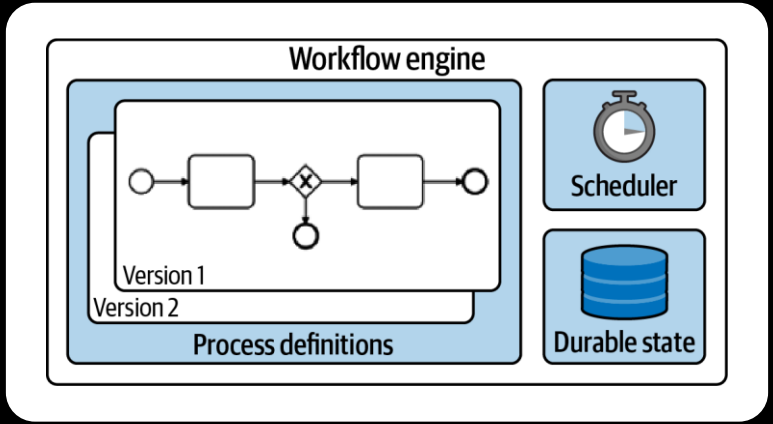
Co-founder and
Chief Technologist of
Camunda

bernd.ruecker@camunda.com
@berndruecker
http://berndruecker.io/





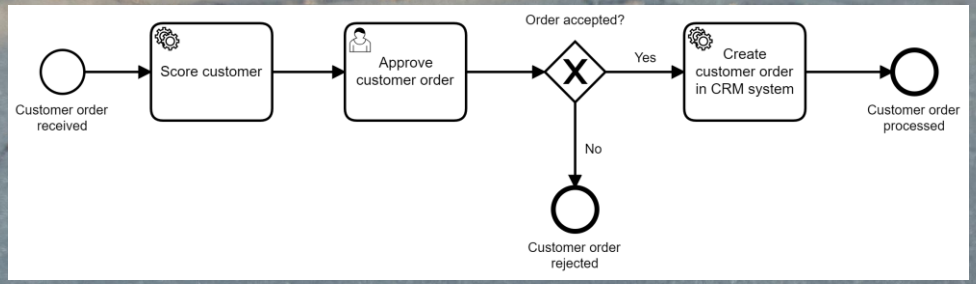




Workflow Engine aka Process Engine aka Orchestration Engine

Live hacking

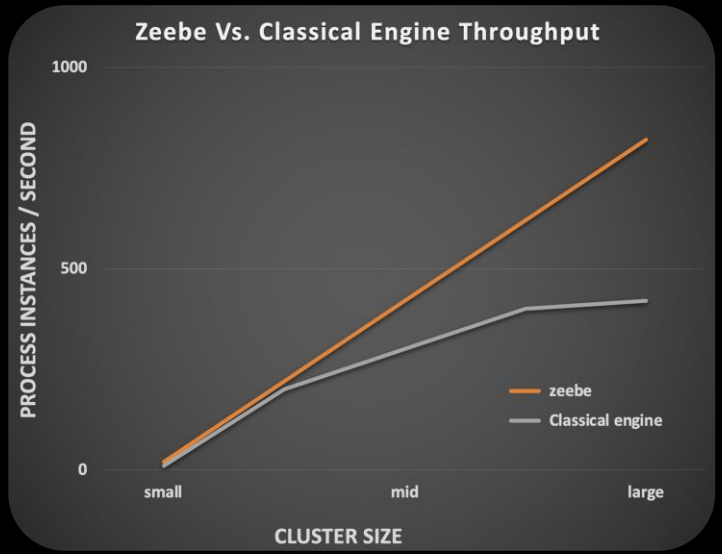
https://github.com/berndruecker/customer-onboarding-camunda-8-springboot



Customer onboarding

https://github.com/berndruecker/customer-onboarding-camunda-8-springboot

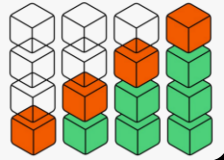
Scaling infinitely



CAMUNDA

Benchmark Performance with Camunda Platform's Zeebe Engine

February 2023



Workload Characteristics of Customers

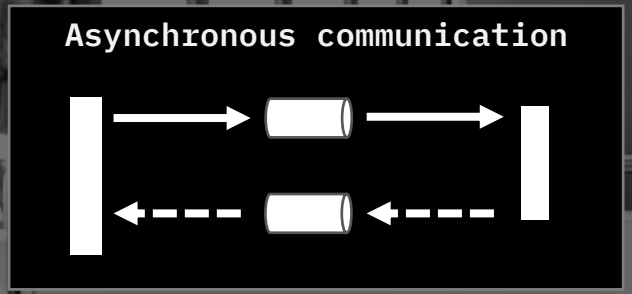
Throughput (PI/s)	Process size (#tasks)	Latency (ms)	Multi-Region Setup
10,000	8 tasks	500 ms	active-passive east-west 60ms
500	3 tasks + 2 messages + 2 call activities	1,000 ms	active-active 10ms avg / 35ms max
2,400	10 tasks	1,200 ms	active-passive 52ms one way
1,700	10 tasks	120,000 ms	active-active-passive 2x east coast + 1x central
800	8 tasks	200 ms	active-passive 62ms
3,000	3 tasks	300 ms	single-region replication factor = 1

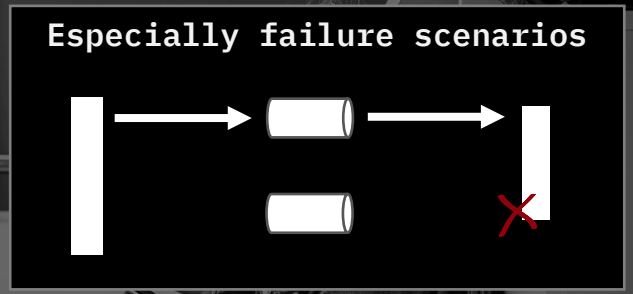
[Webinar] Benchmark Performance with Camunda Platform's Zeebe Engine


There are also technical reasons
why services need to wait...

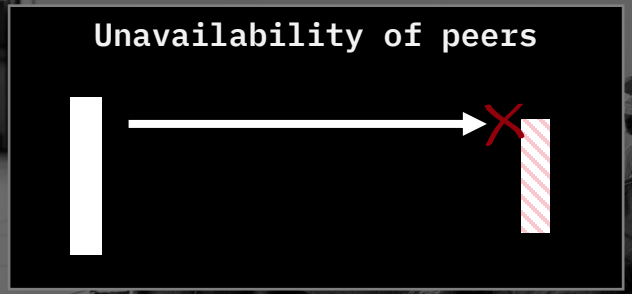
When do services need to wait? Some technical reasons...

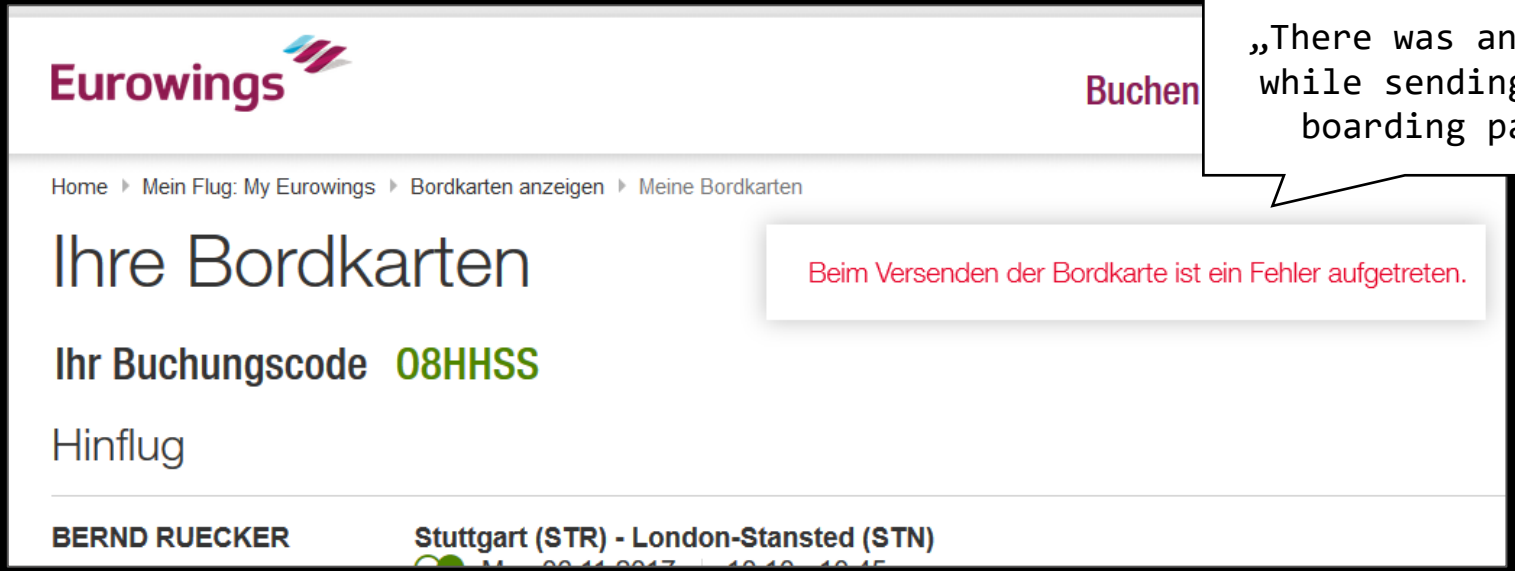
Wait for responses





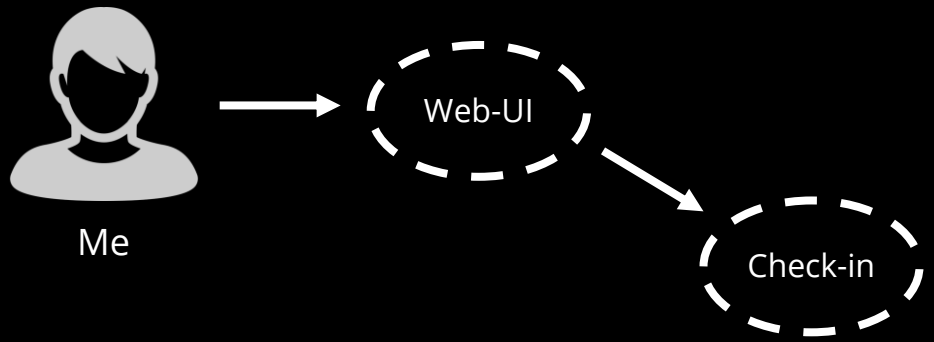
Wait for availability



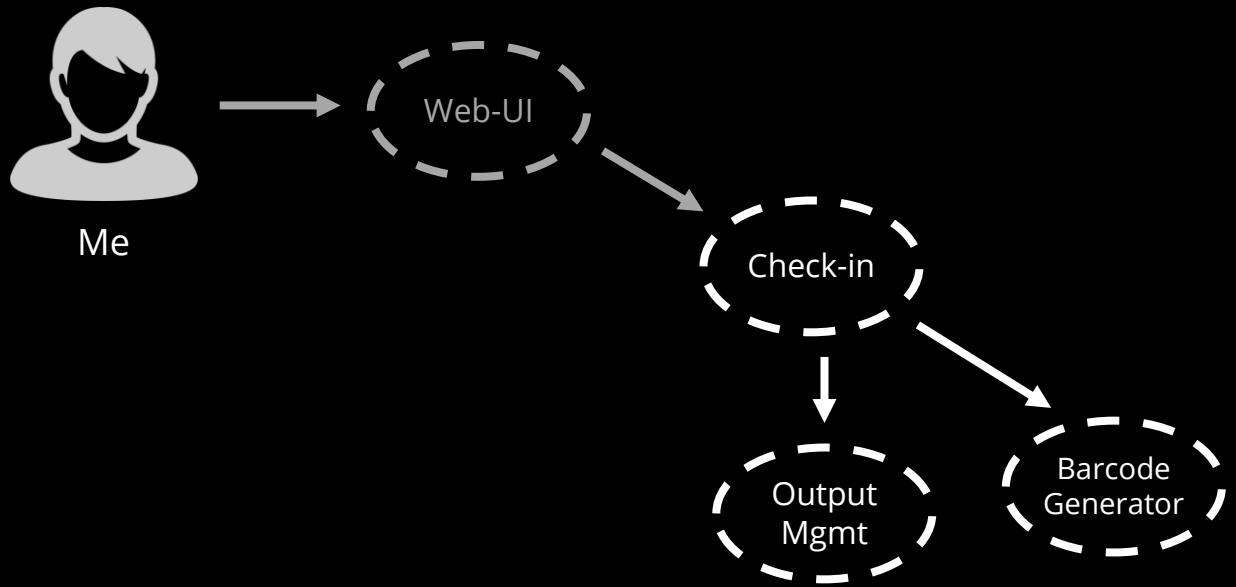


„There was an error while sending your boarding pass“

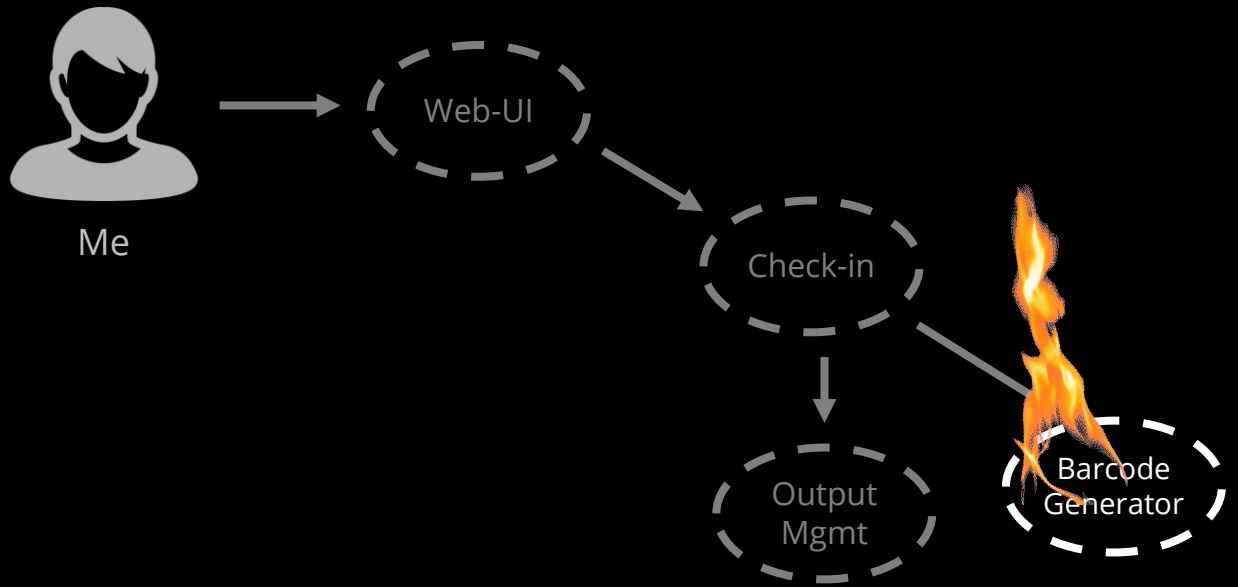
Current situation

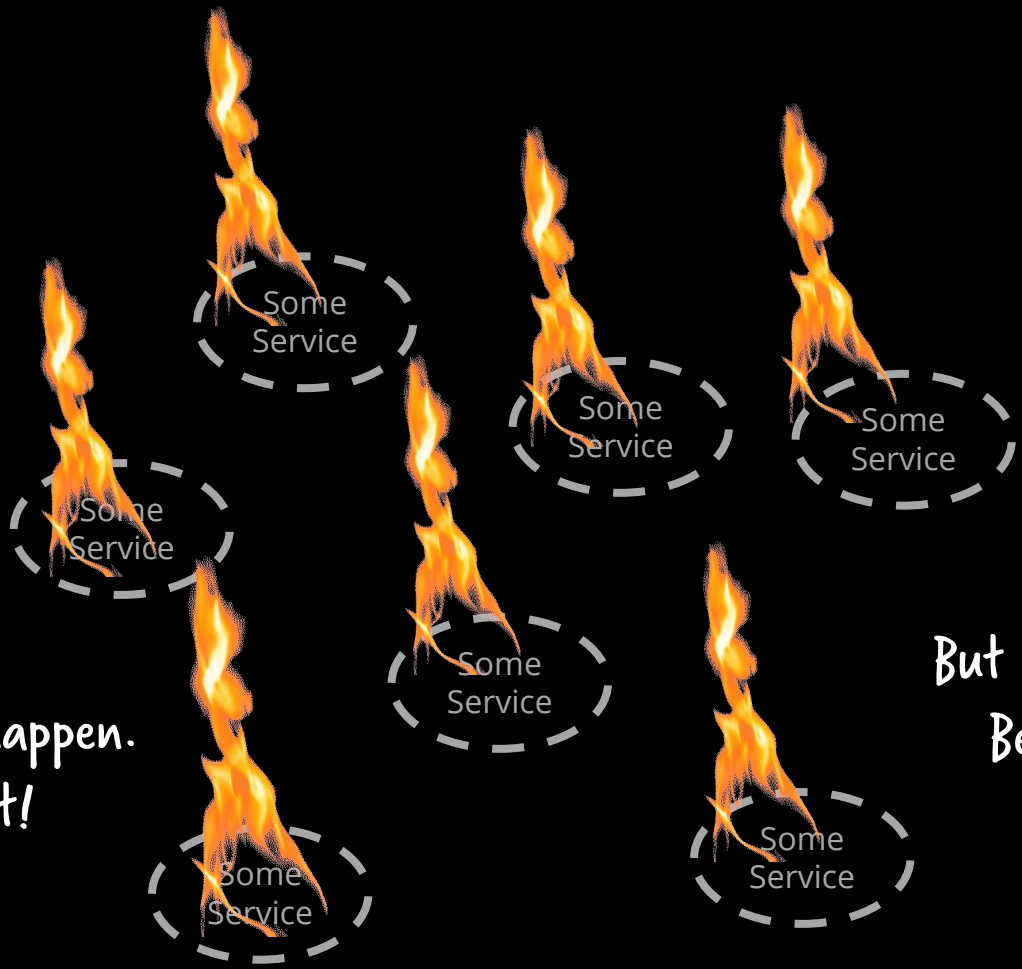


Current situation



Current situation

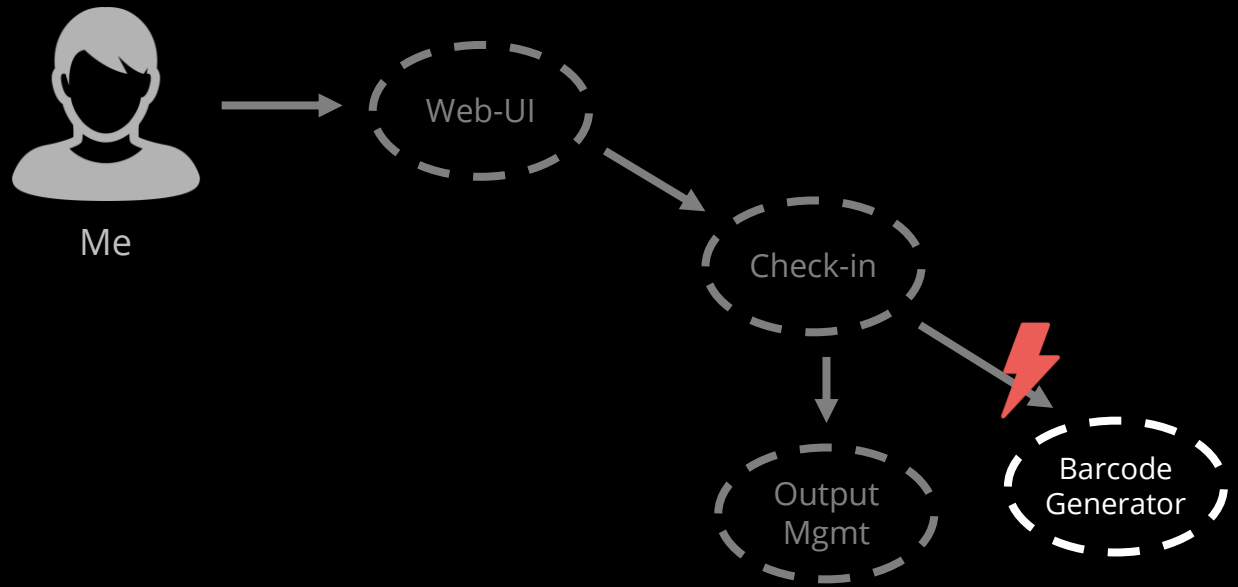




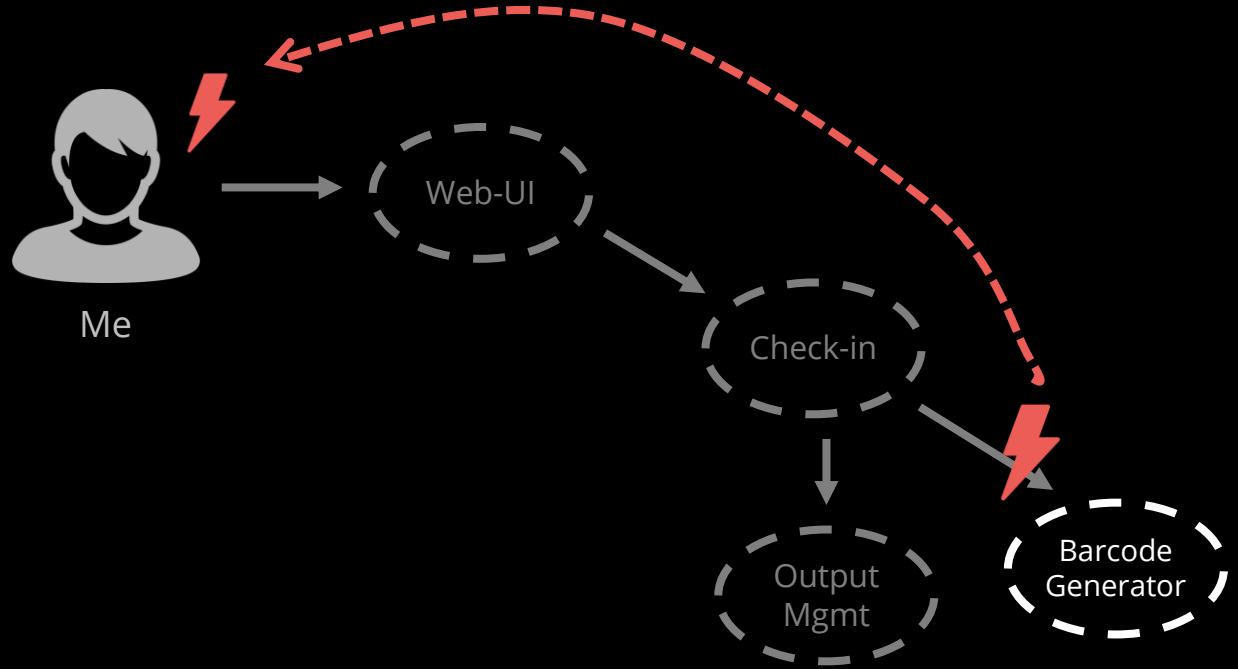
Failure will happen.
Accept it!

But keep it local!
Be resilient.

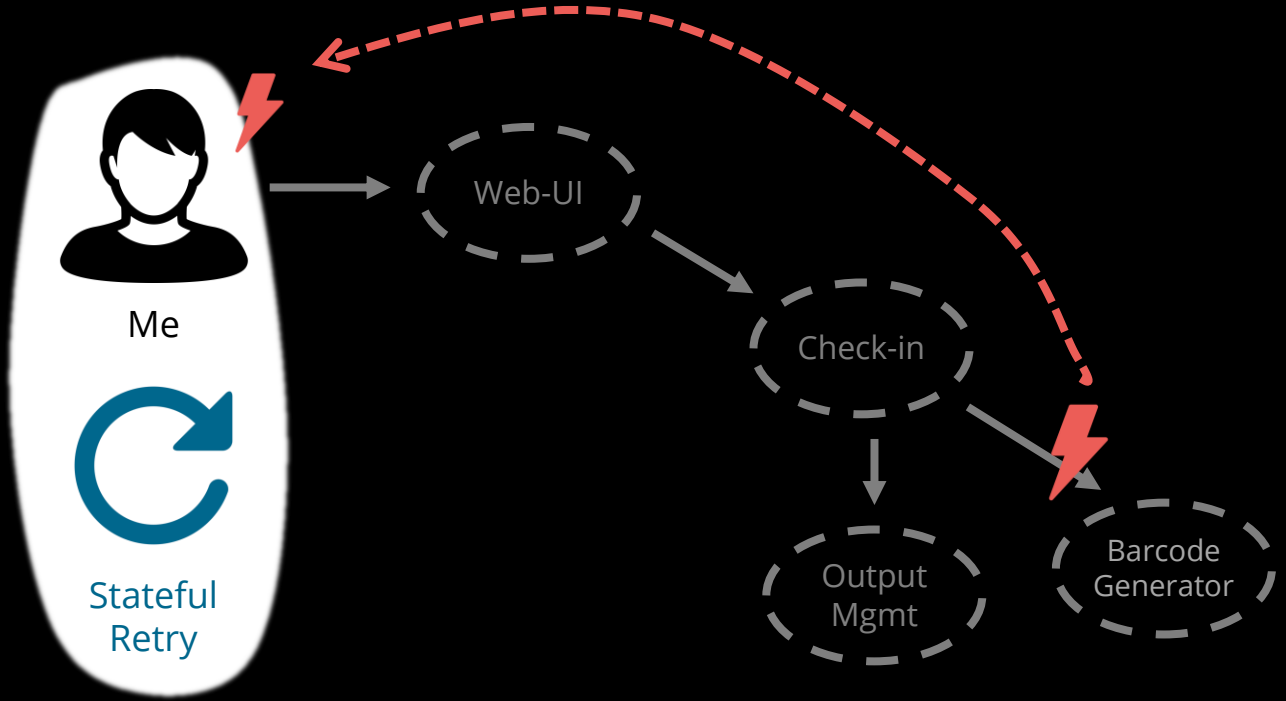
Current situation – the bad part

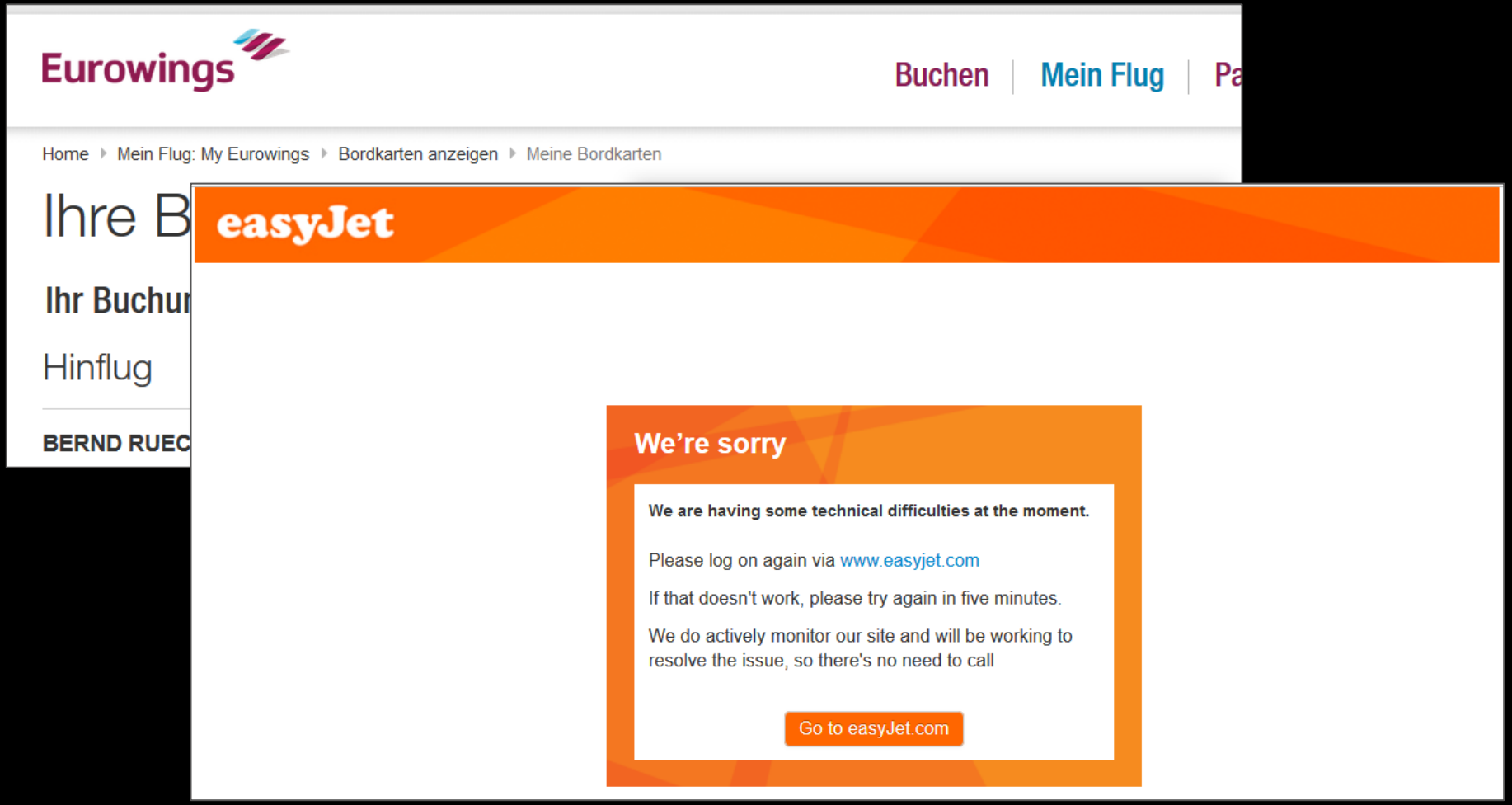


Current situation – the bad part

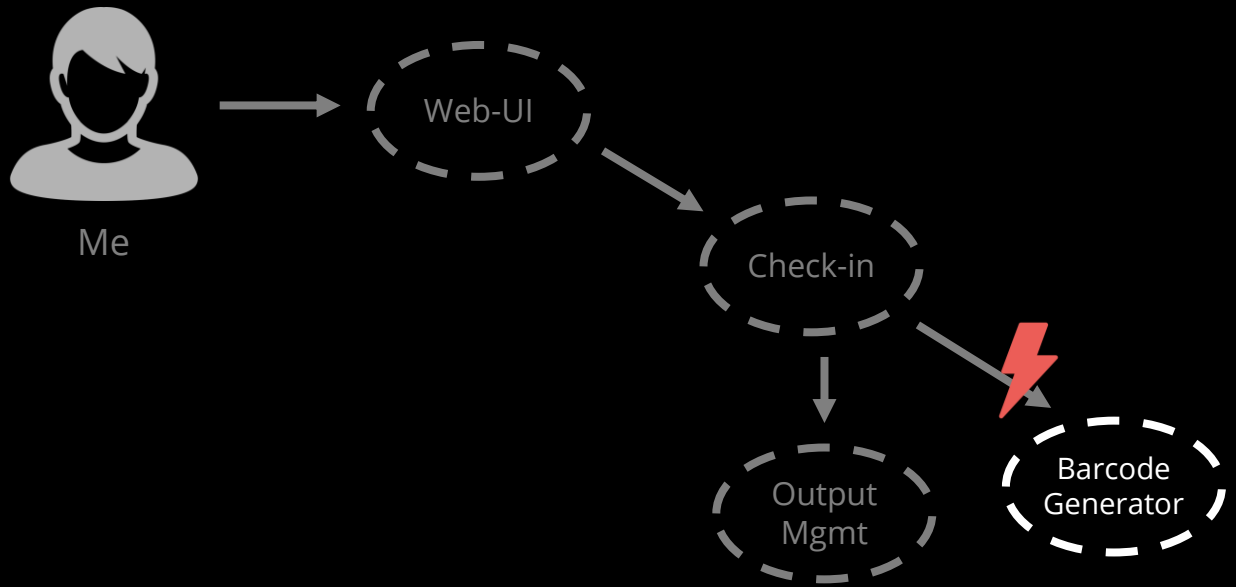


Current situation – the bad part

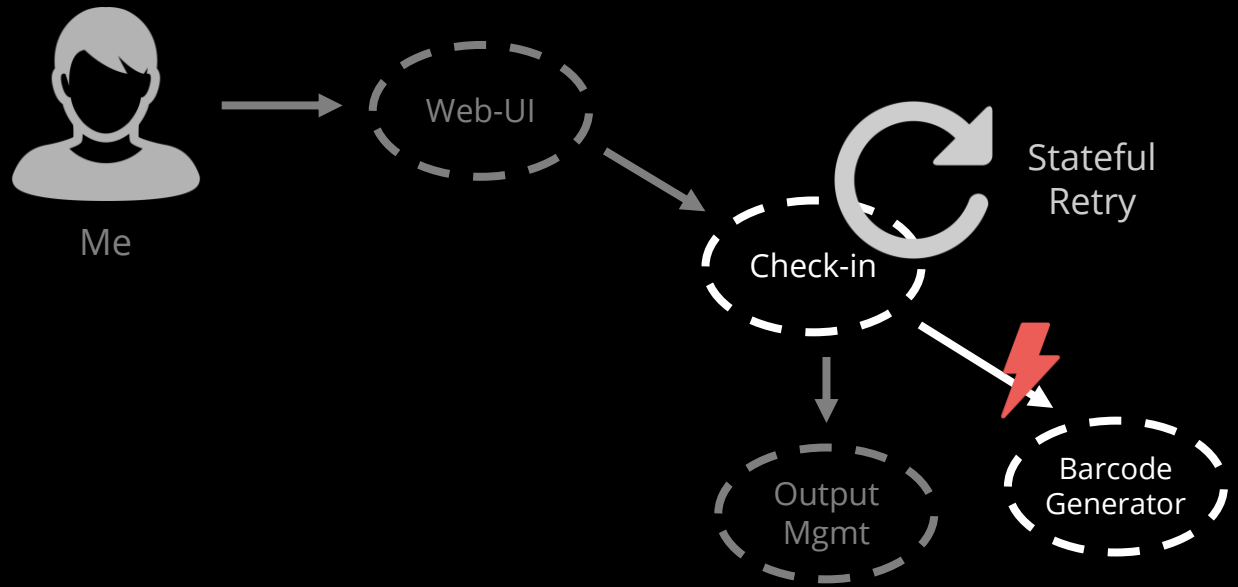



Eurowings
Buchen
Mein Flug
Home
Mein Flug: My Eurowings
Bordkarten anzeigen
Meine Bordkarten
Ihre B
Ihr Buchu
Hinflug
BERND RUEC
easyJet
We're sorry
We are having some technical difficulties at the moment.
Please log on again via www.easyjet.com
If that doesn't work, please try again in five minutes.
We do actively monitor our site and will be working to resolve the issue, so there's no need to call
Go to easyJet.com

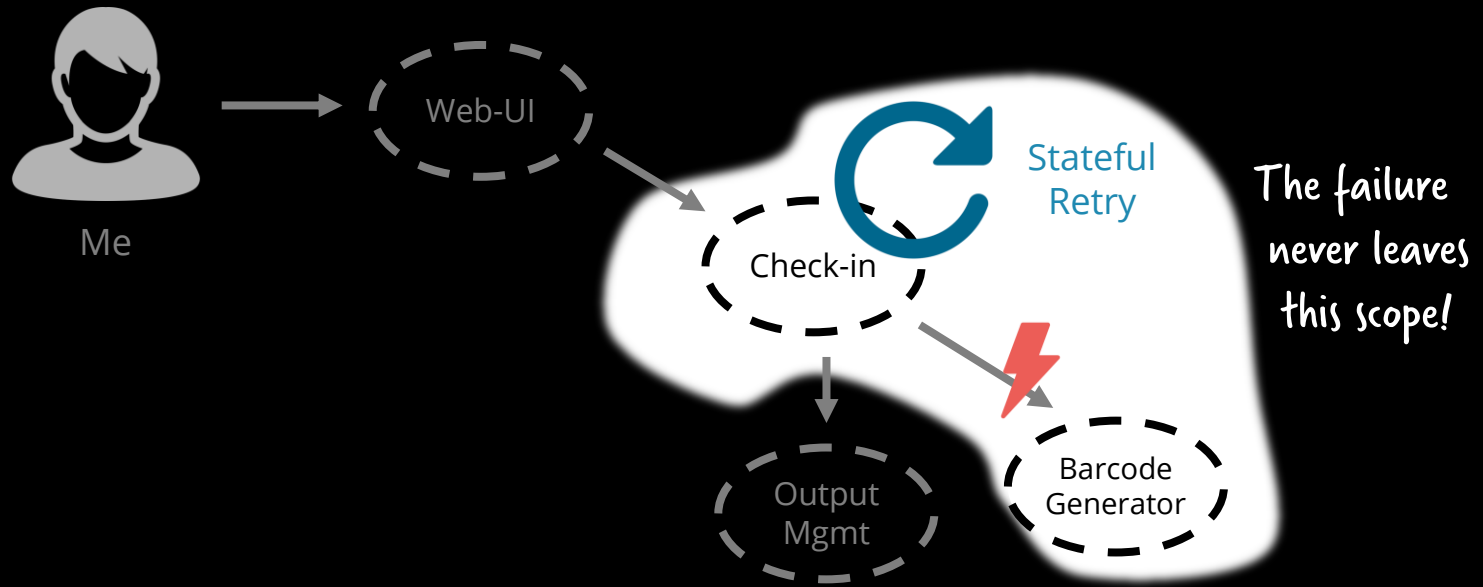
Possible situation – much better!



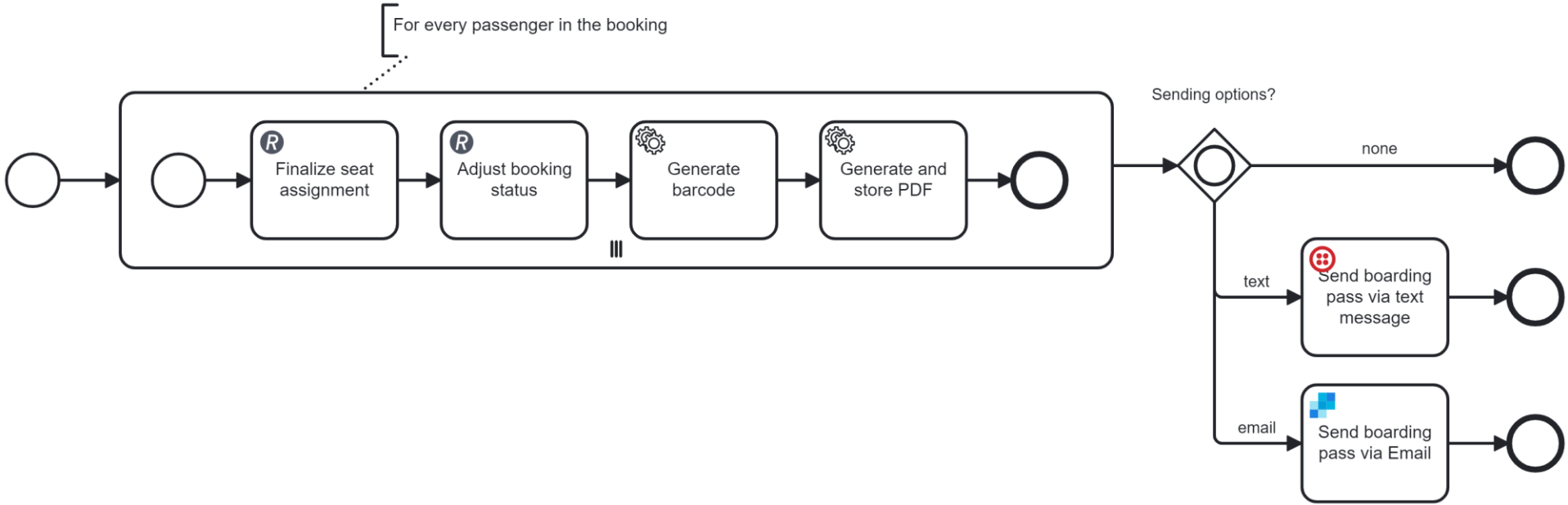
Possible situation – much better!



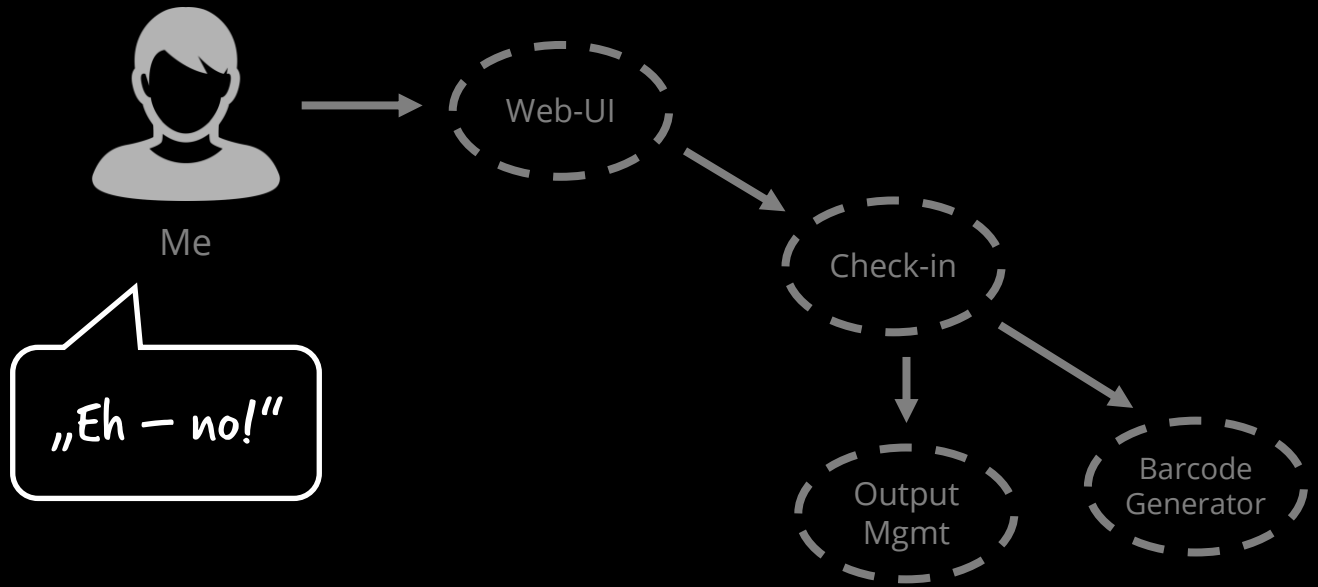
Possible situation – much better!



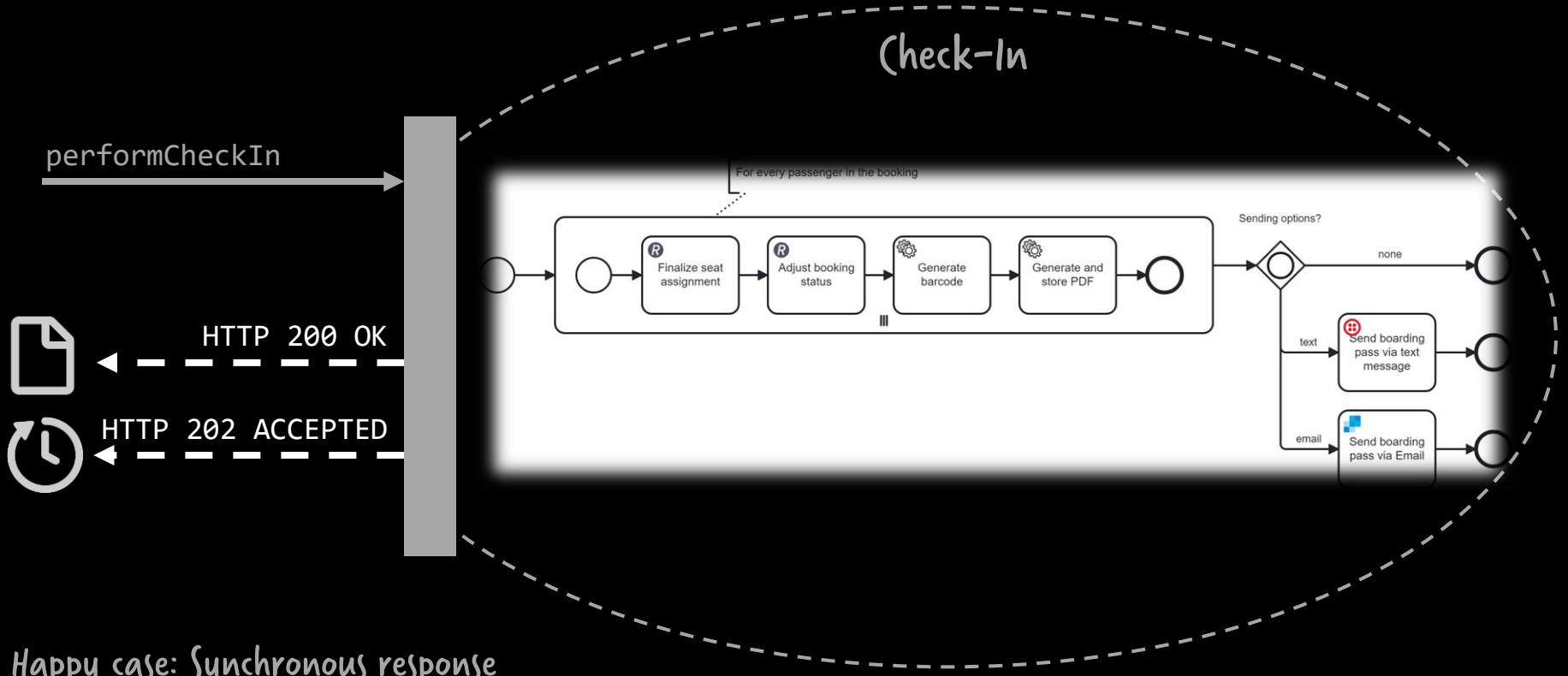
A long running process to check-in



„But the customer wants a synchronous response!“

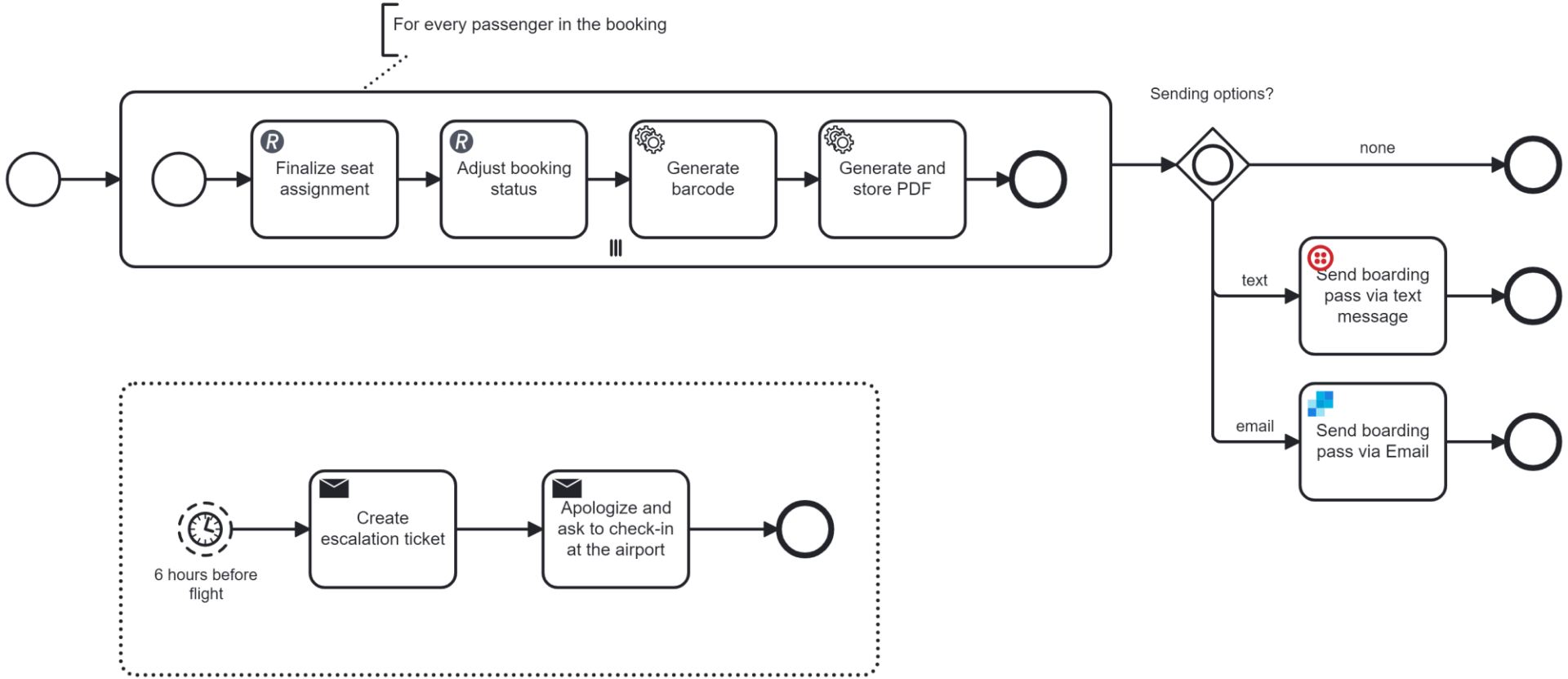


Embrace asynchronicity

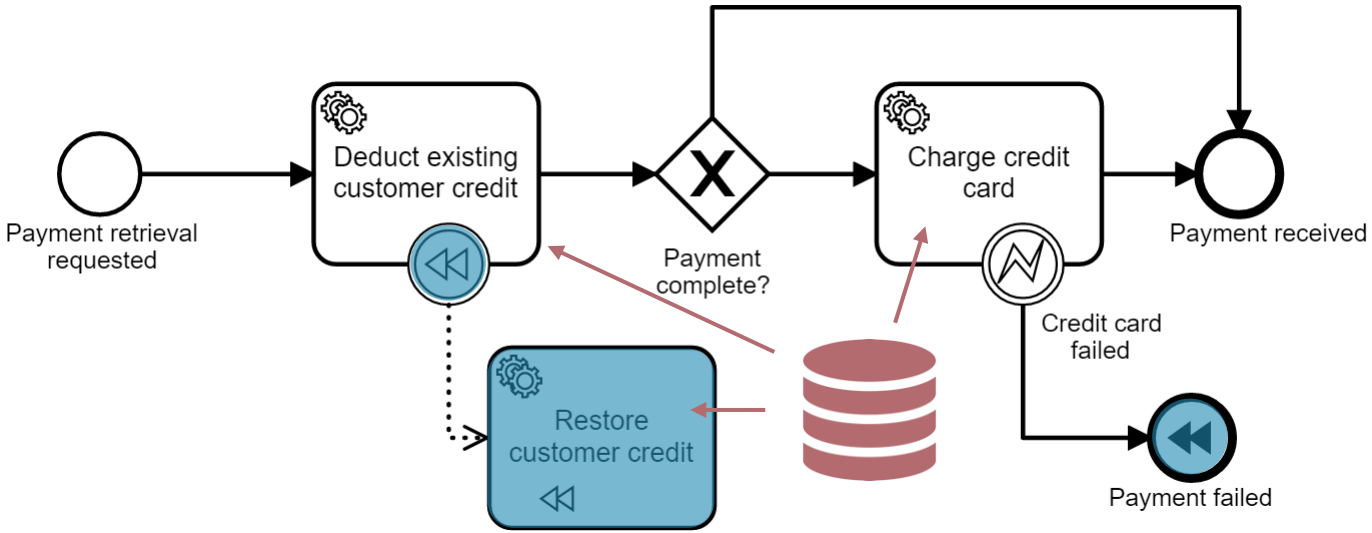


Happy case: Synchronous response
otherwise: asynchronous

Adding timeouts and alternative paths



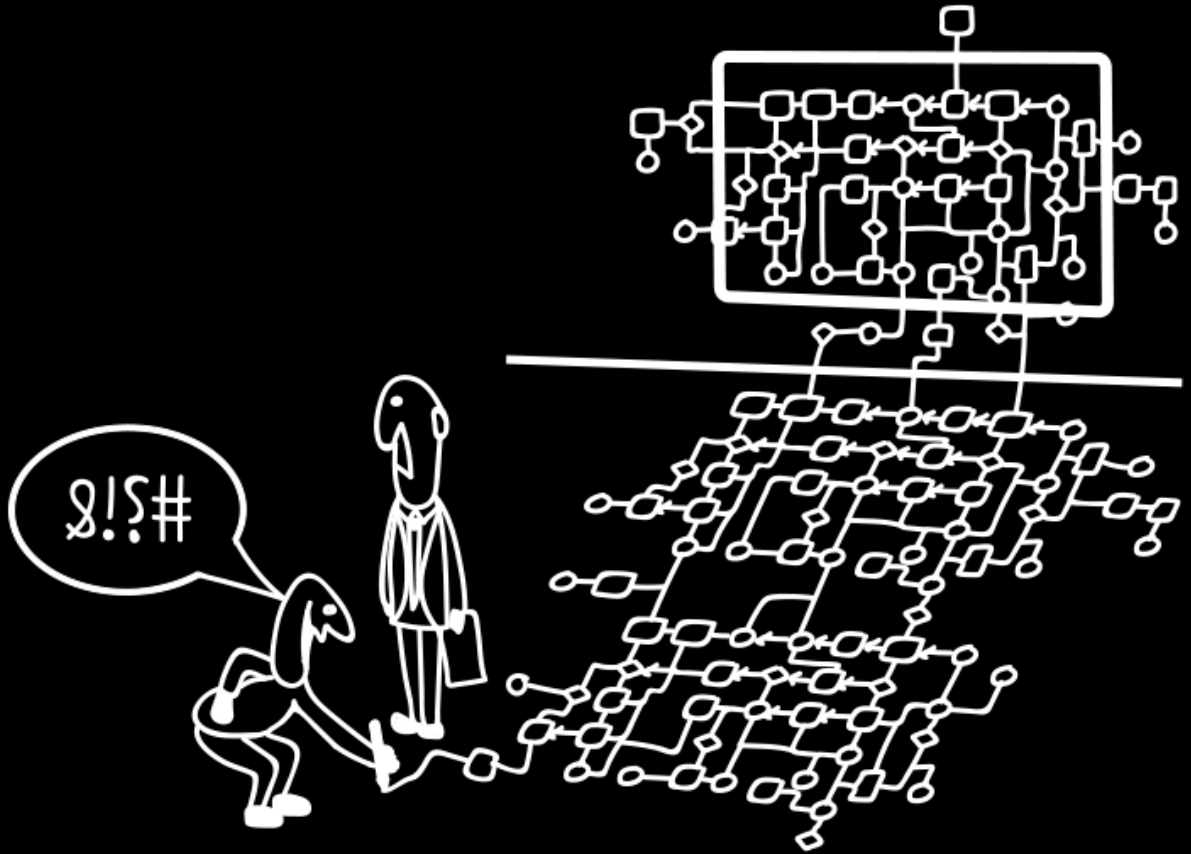
Advanced workflow patterns using BPMN



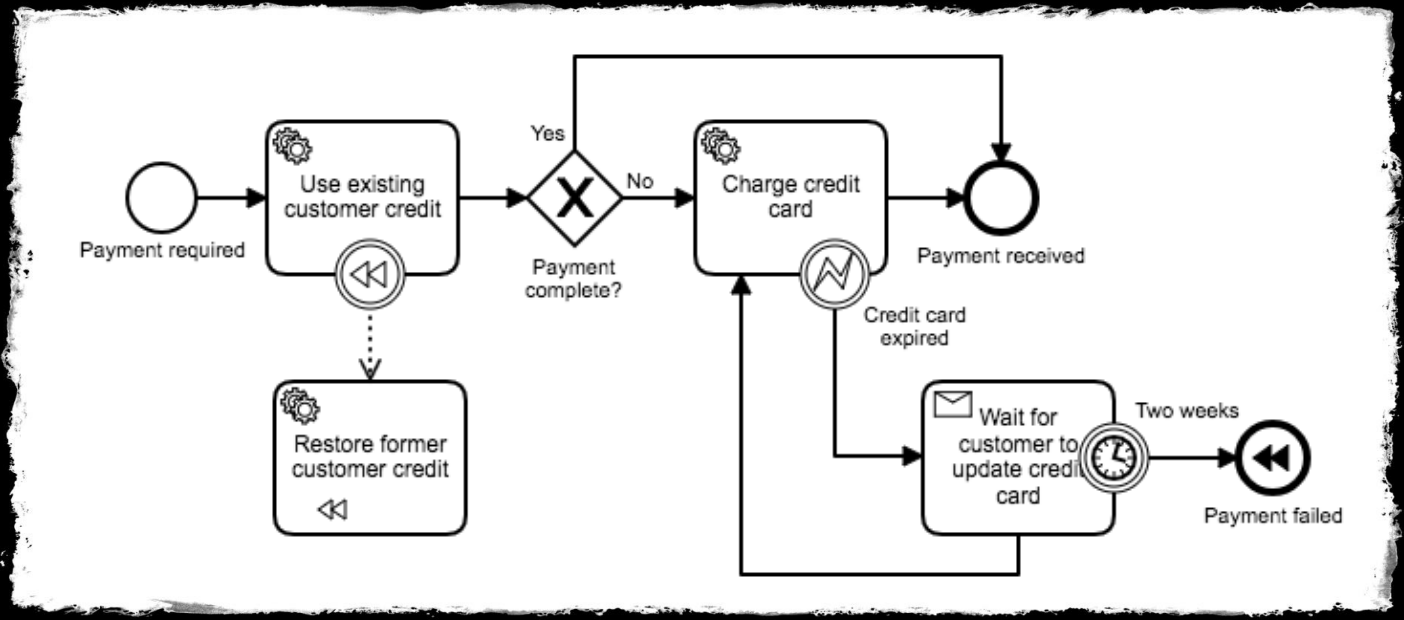
Compensation*

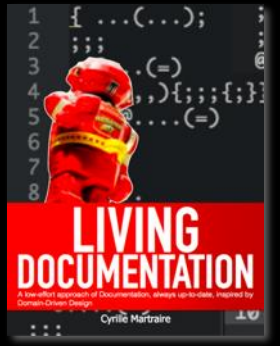
*aka Saga pattern

Graphical models?

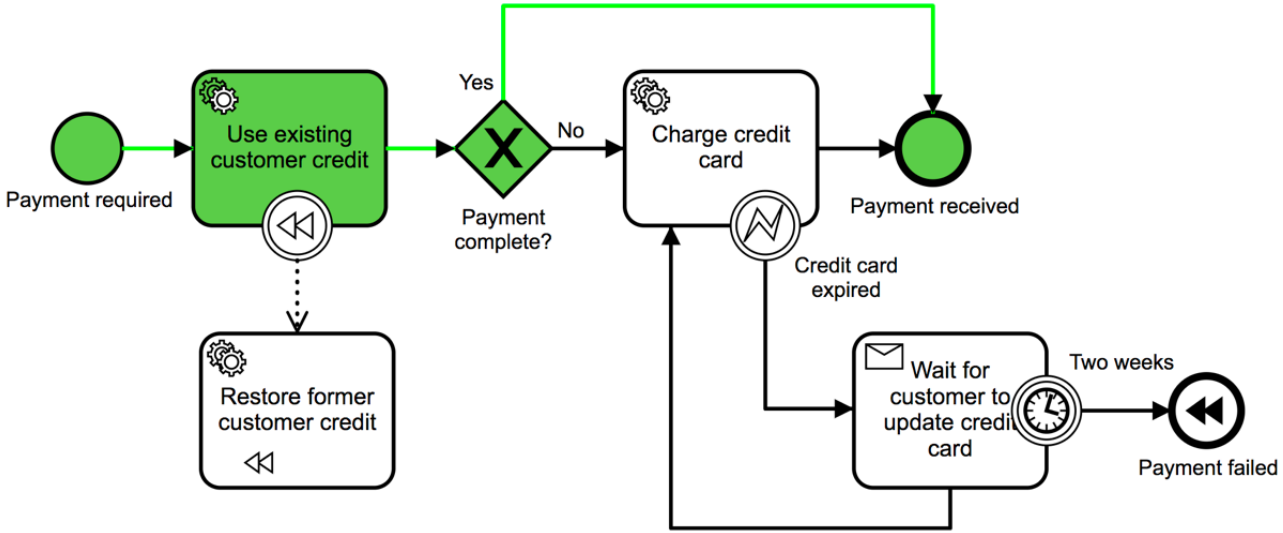


Living documentation for long-running behaviour

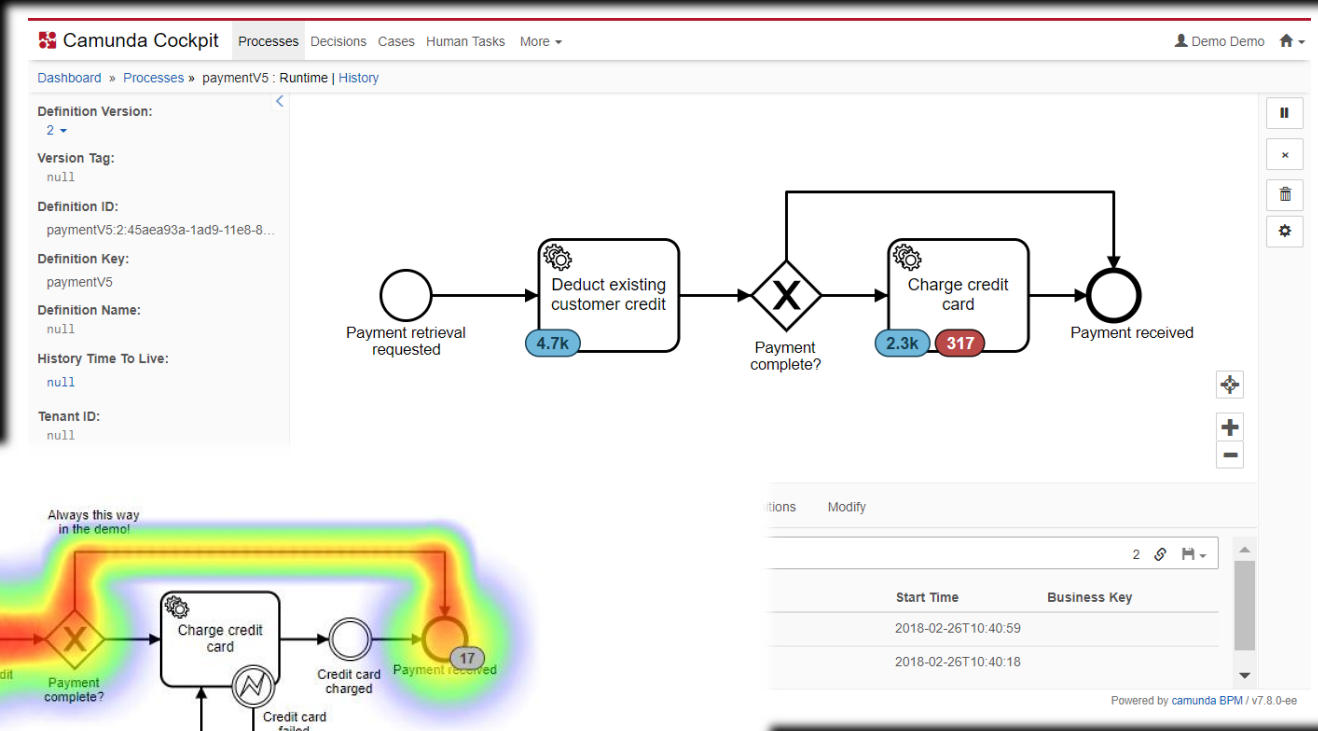


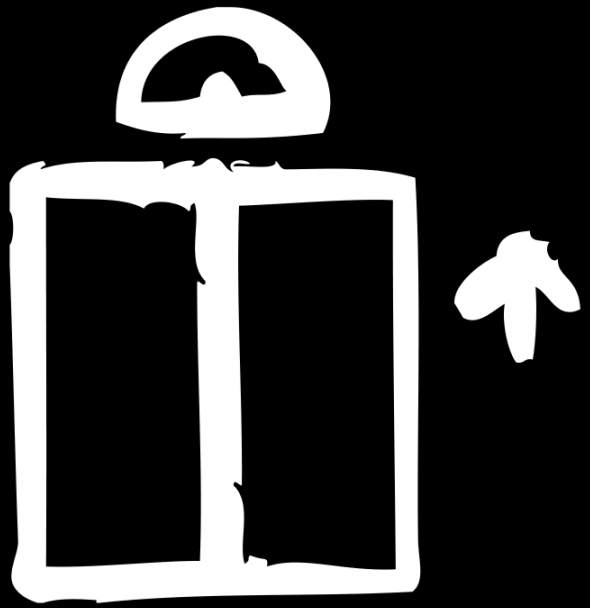


Living documentation for long-running behaviour



BizDevOps


Camunda Cockpit
Processes
Decisions
Cases
Human Tasks
More
Demo Demo
Dashboard » Processes » paymentV5 : Runtime | History
Definition Version:
2
Version Tag:
null
Definition ID:
paymentV5:2:45aea93a-1ad9-11e8-8...
Definition Key:
paymentV5
Definition Name:
null
History Time To Live:
null
Tenant ID:
null
Payment retrieval requested
Deduct existing customer credit
4.7k
Payment complete?
Charge credit card
2.3k
317
Payment received
Modify
Start Time
Business Key
2018-02-26T10:40:59
2018-02-26T10:40:18
Powered by camunda BPM / v7.8.0-ee
Does nothing in this demo!
Always this way in the demo!
Payment retrieval requested
Deduct existing customer credit
Customer credit used
Payment complete?
Charge credit card
Credit card charged
Payment received
17
Credit card failed
Restore customer credit
Ask customer to update credit card
Wait for customer to update credit card
7 days
Payment failed



Decisions about
long running behavior
need to be elevated
to the business level

Your mobile boarding pass always at hand.

You'll be automatically checked in for your flight

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verif...

Booking Reference 6YZJWW

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Dear Mr Bernd Ruecker,

Thank you for providing your details to be automatically checked in. You can now sit back and relax as we do the necessary when online check-in opens. A confirmation will be sent to you once your check-in is successful.

Manage Booking

Select / Change your seats and meals, update your personal details, and more.

Flights you'll be automatically checked-in for

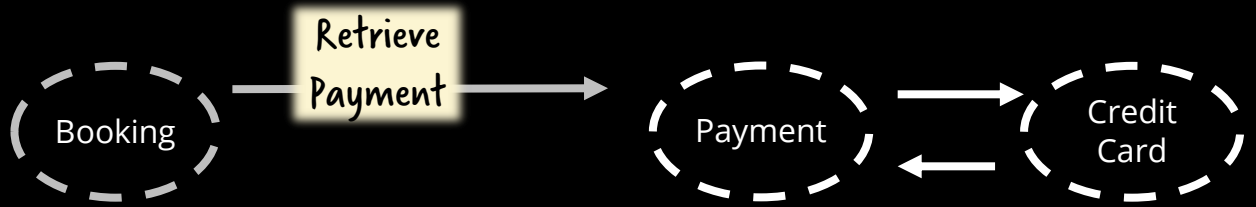
Flight	Departs	Arrives
1. Frankfurt to New York - Total travel time: 8hrs 35mins		Status: Confirmed

Long running capabilities are essential to design good service boundaries

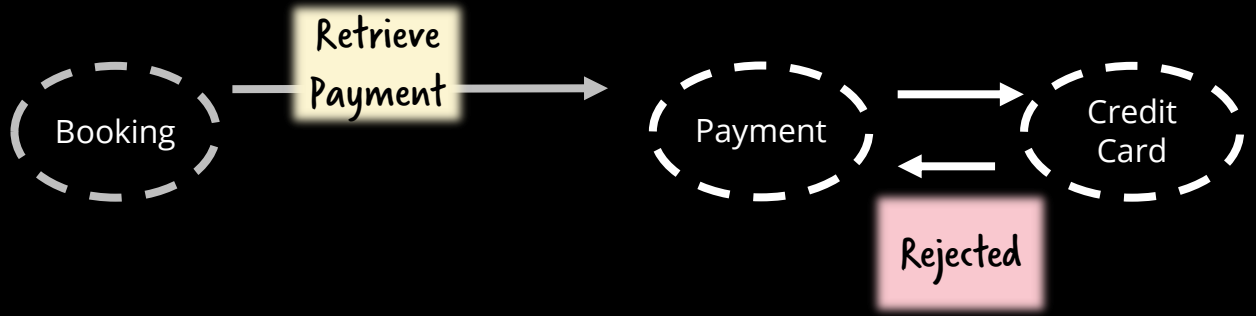
(= a good architecture)

Example

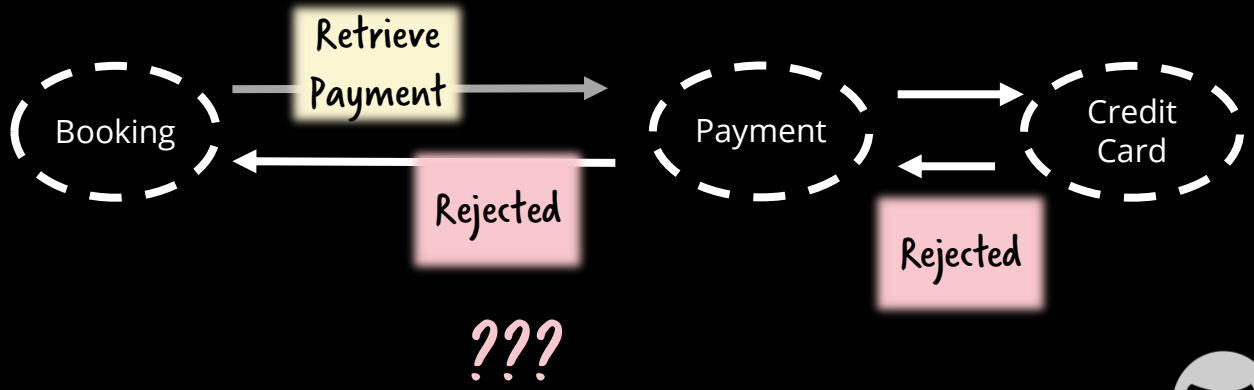
Example



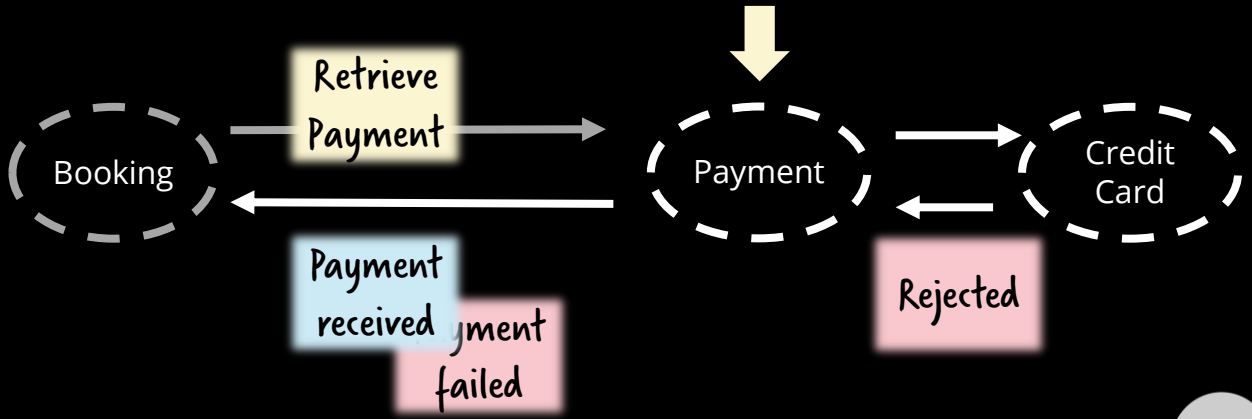
Example



Example

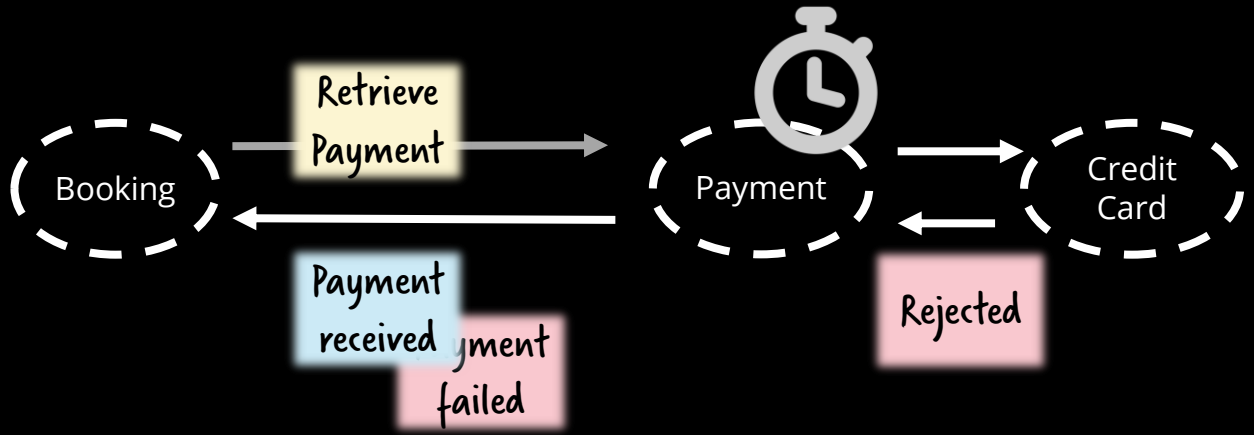

Booking
Retrieve Payment
Rejected
???
Payment
Credit Card
Rejected
If the credit card was rejected, the customer can provide new details

Who is responsible to deal with problems?

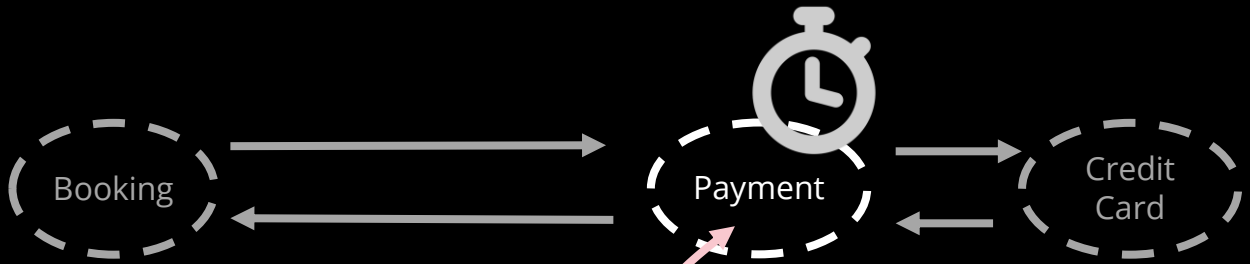


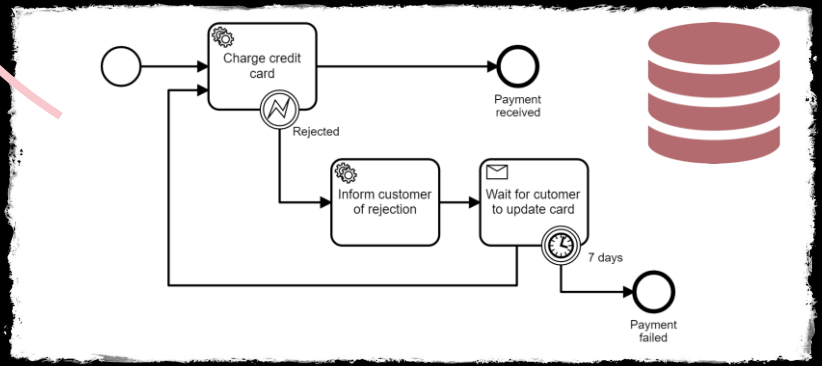
If the credit card was rejected, the customer can provide new details

(Potentially) long running services

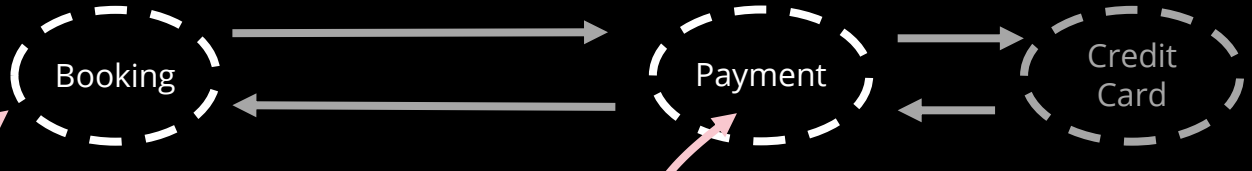


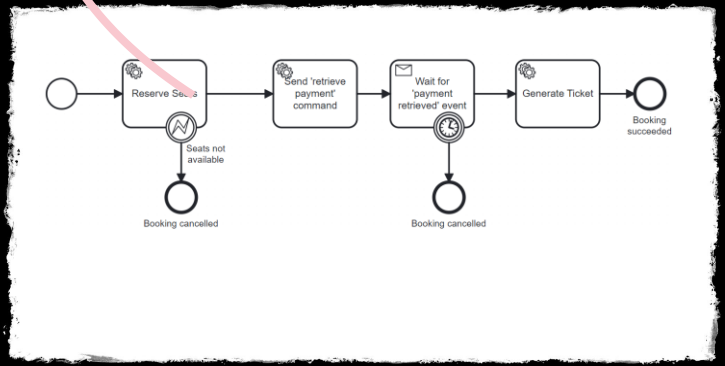
Long running services

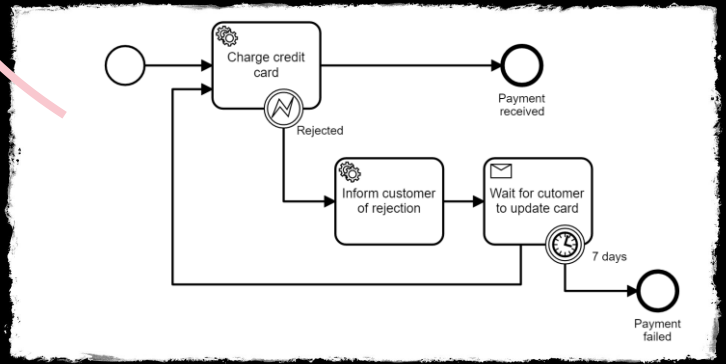




orchestration != monolithic processes



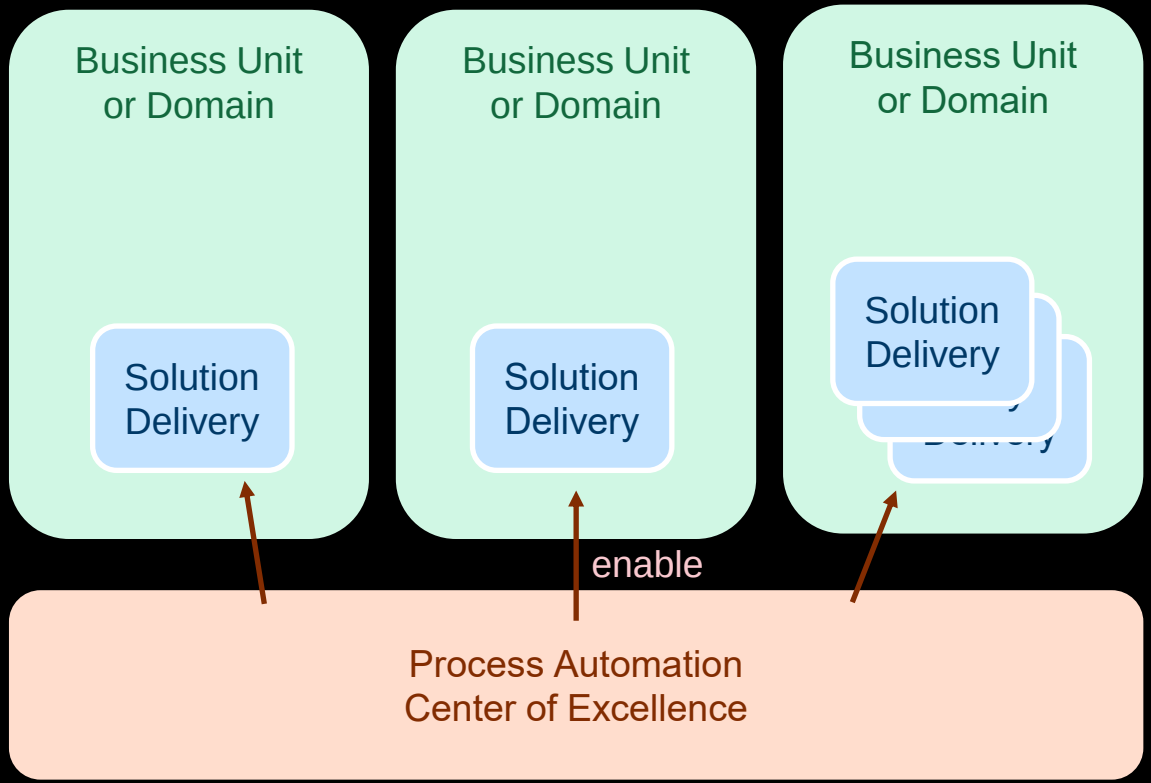


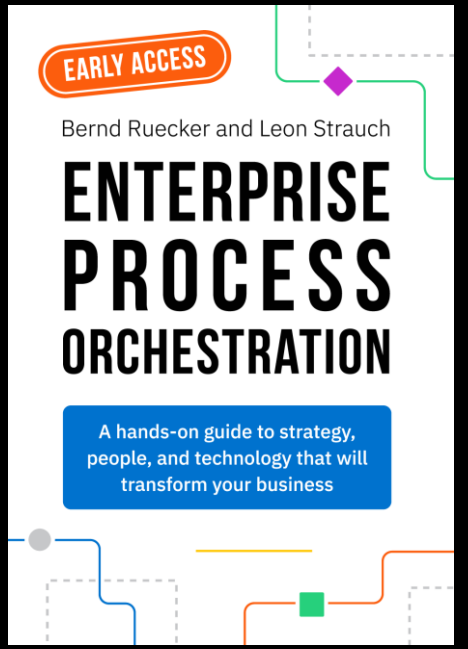


Ubiquitous Process orchestration

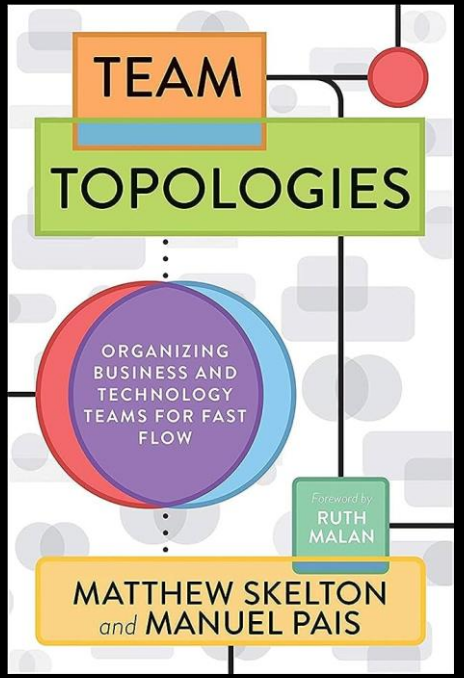
You need a
process orchestration capability
that is available as-a-service
within your organization
to create competitive architectures

Center of excellence can enable and provide a platform

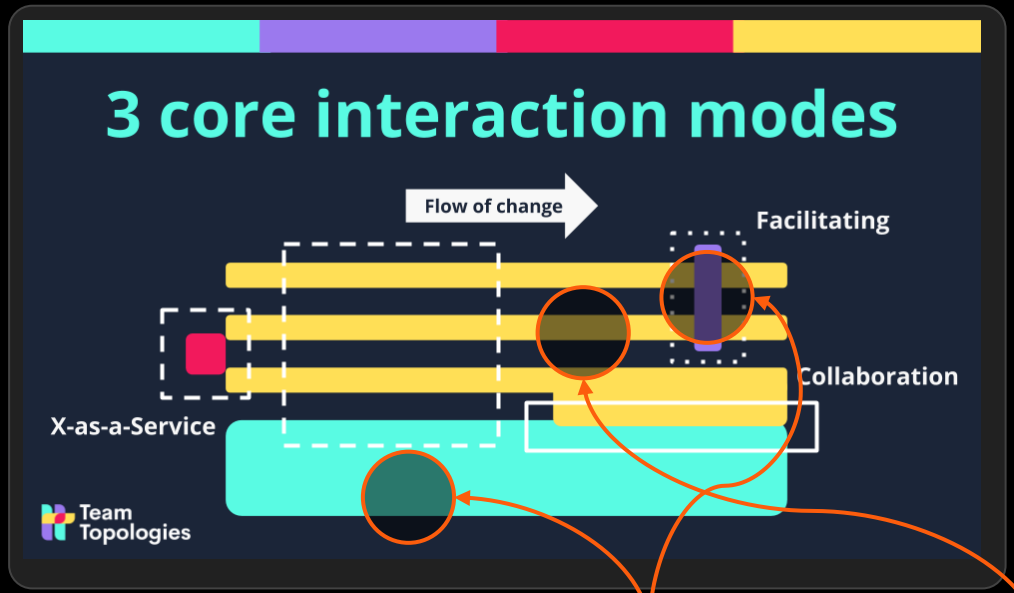




But isn't a central CoE harming team autonomy?

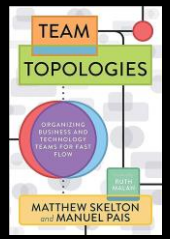


(This is a rhetorical question - the answer is of course No)



Center of Excellence

Domain



Stream-aligned team

Enabling team

Complicated Subsystem team

Platform team

Spotify R&D

Golden Paths

“We found that
rumour-driven development
simply wasn’t scalable”

https://engineering.atspotify.com/2020/08/how-we-use-golden-paths-to-solve-fragmentation-in-our-software-ecosystem/

The Speed Paradox

At Spotify, we've always believed in the speed and ingenuity that comes from having autonomous development teams. But as we learned firsthand, the faster you grow, the more fragmented and complex your software ecosystem becomes. And then everything slows down again.

The Standards Paradox

By centralizing services and standardizing your tooling, Backstage streamlines your development environment from end to end. Instead of restricting autonomy, standardization frees your engineers from infrastructure complexity. So you can return to building and scaling, quickly and safely.

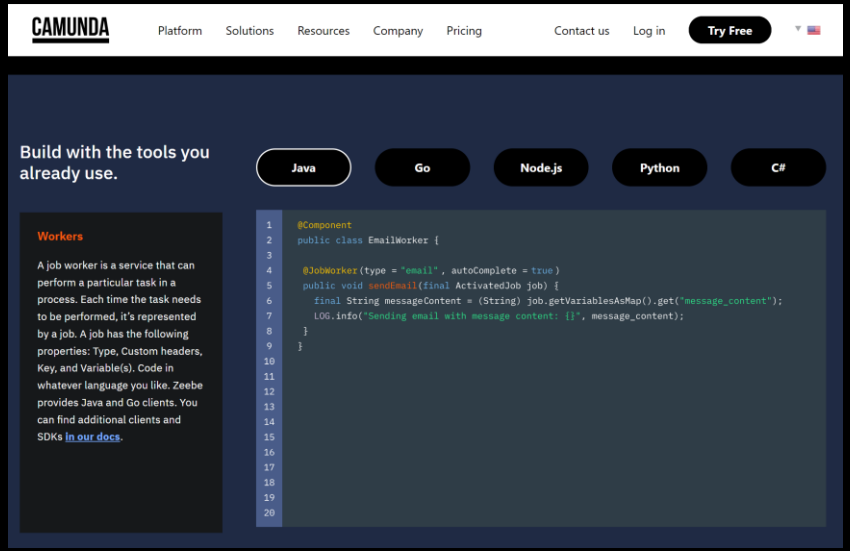
https://backstage.io/ (OSS, Made with 💔 at Spotify)

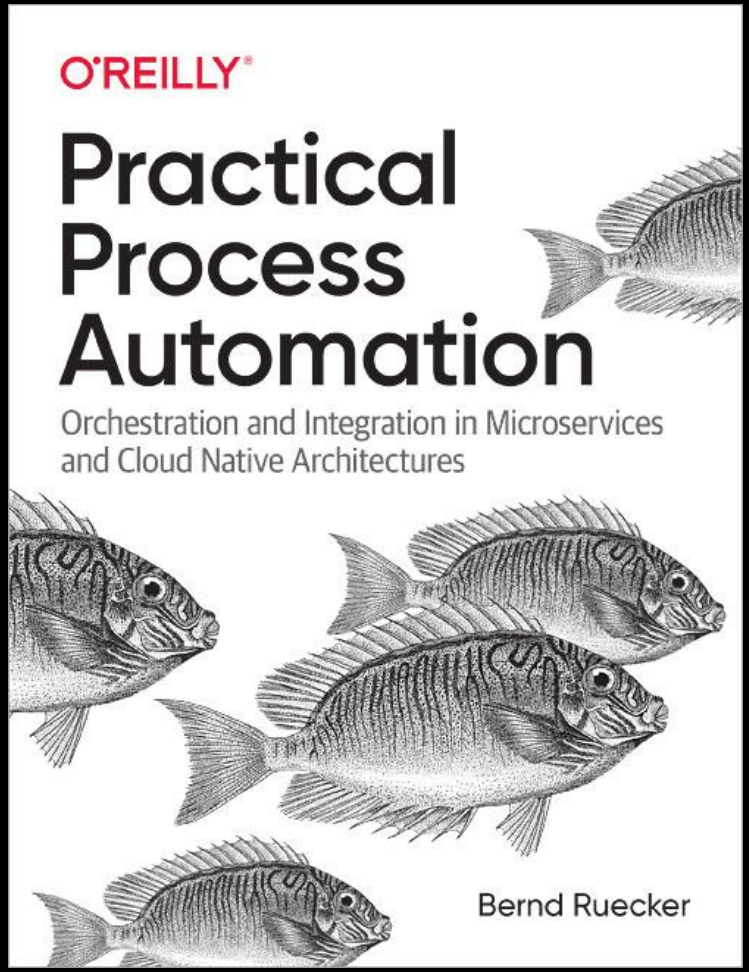
Recap

- You need capabilities for long running behavior for technical and business reasons
- Process orchestration platforms / workflow engines are a great fit
- This allows you to design better service boundaries, remove accidental complexity, implement quicker, and provide a better customer experience
- Organize central enablement (center of excellence, platform as-a-service) for easy adoption at scale

Want to learn more?

http://camunda.com/





THANK YOU

https://www.berndruecker.io/ | bernd.ruecker@camunda.com