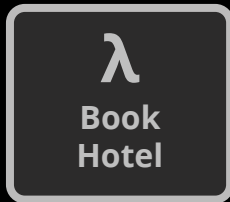
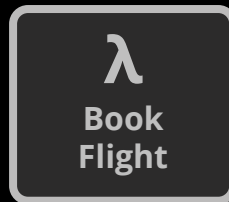
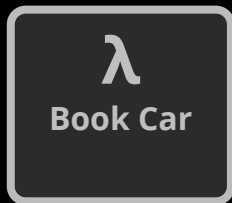


Coordination of Serverless Functions

@berndruecker



An example for today – the classical one

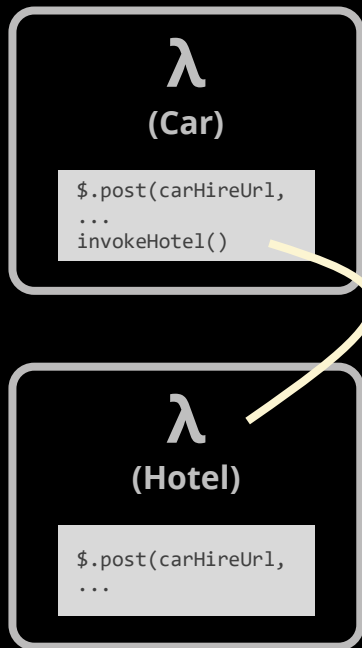


Function collaboration

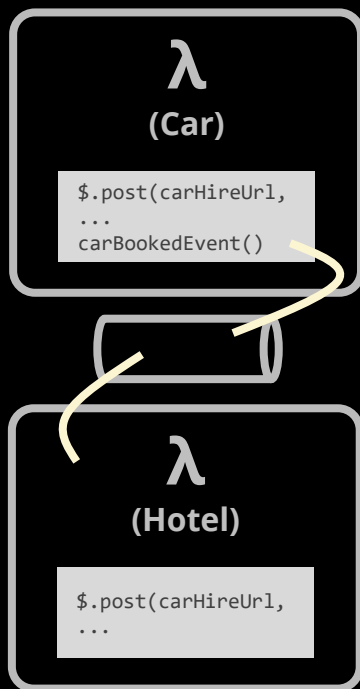


Choreography

Chaining



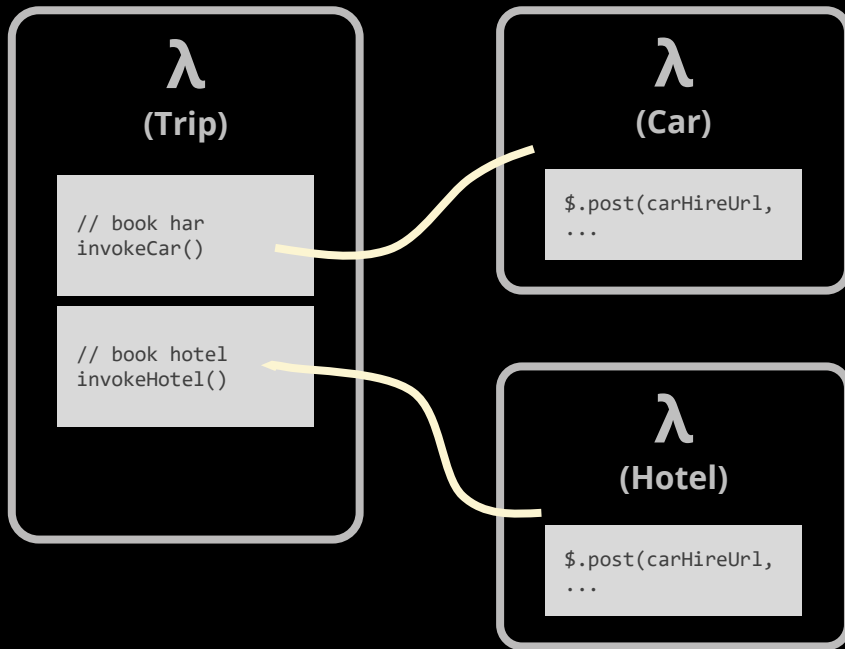
Event-Chain



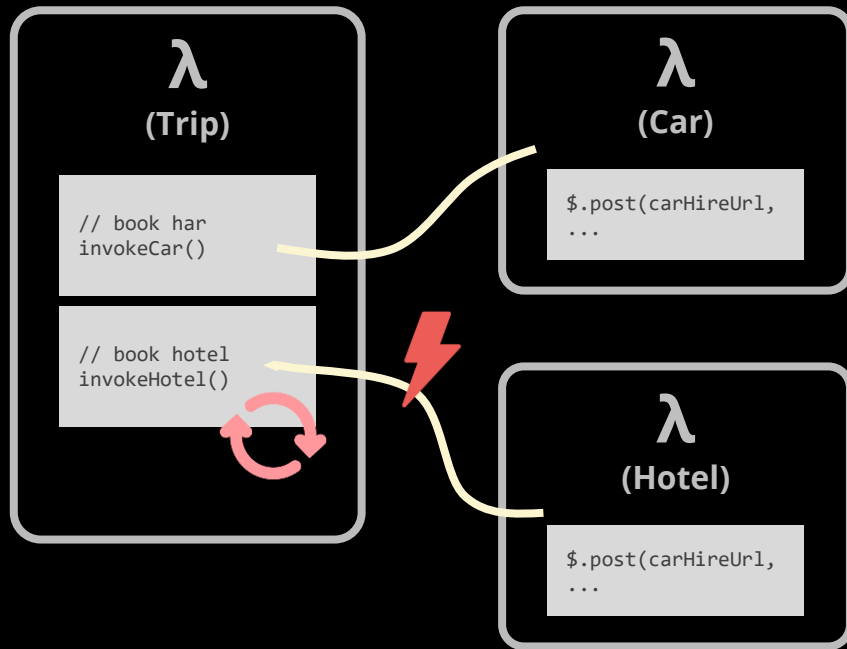
No extra framework needed

The flow is scattered across functions and thus invisible
(changing the sequence is hard)

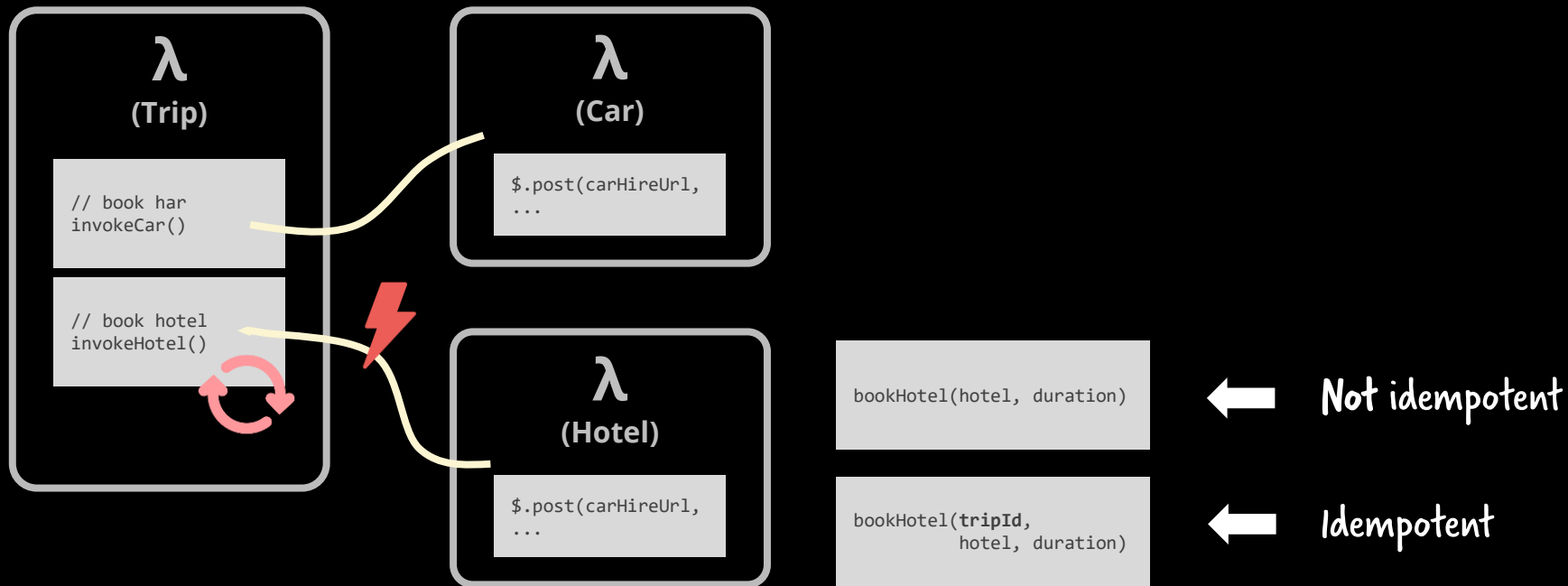
orchestration / coordinator function



orchestration / coordinator function

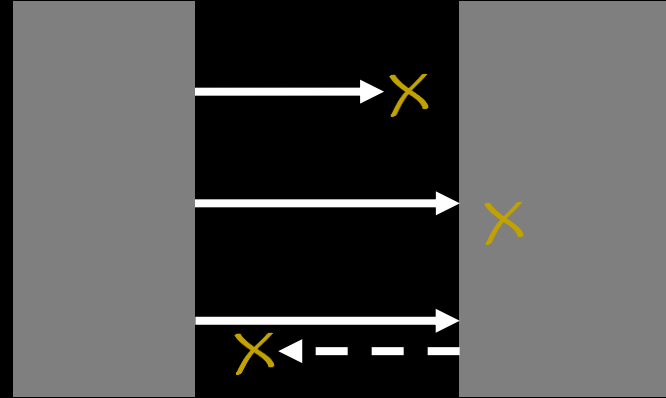


orchestration / coordinator function

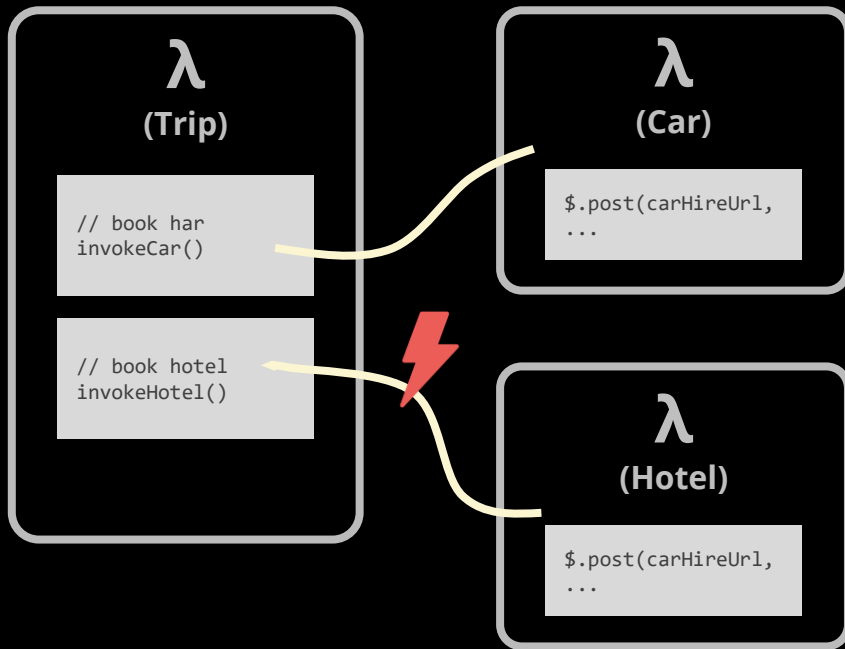


It is impossible to
differentiate certain
failure scenarios:

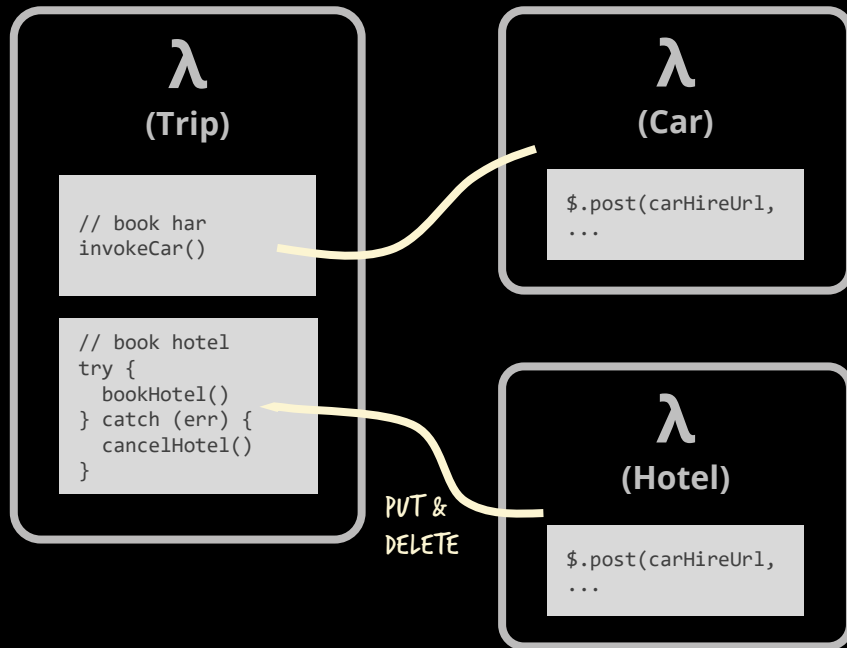
Independant of
communication style!



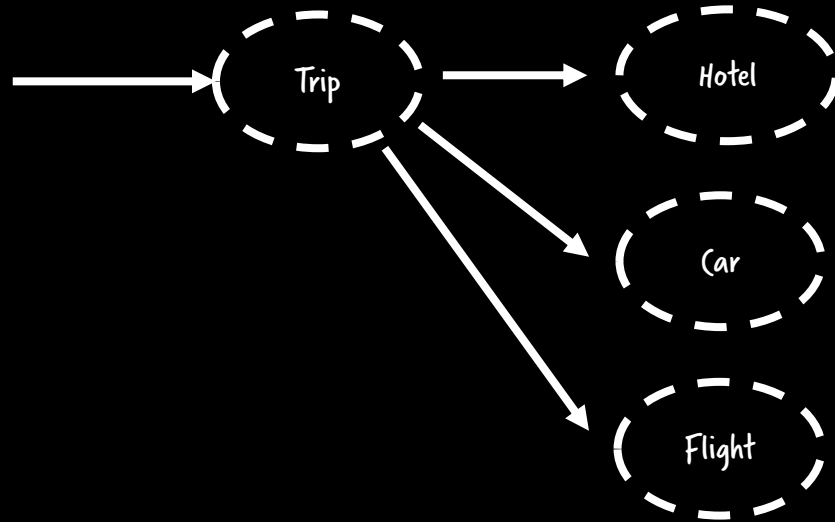
orchestration / coordinator function



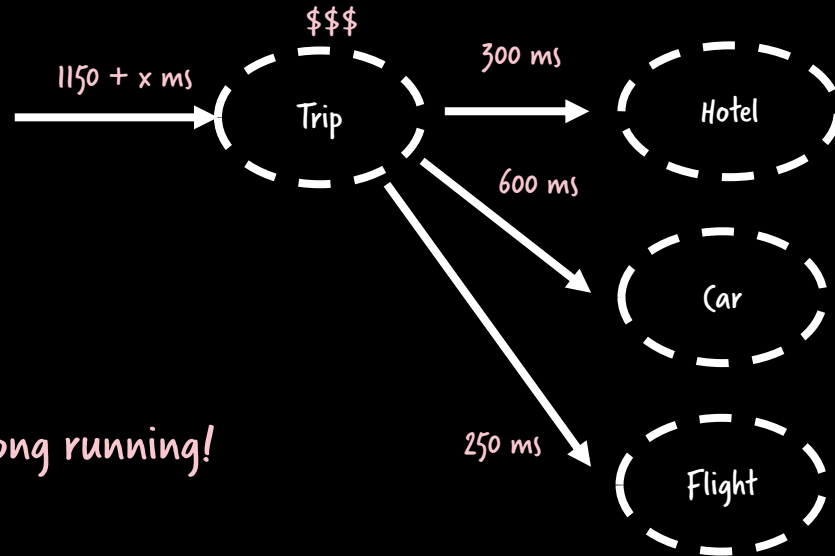
Cleanup!



Weaknesses

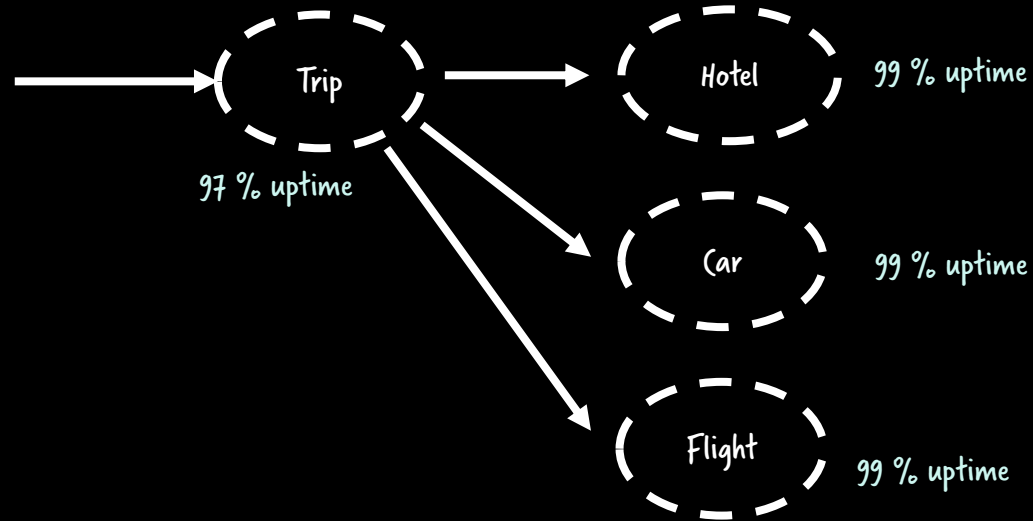


Weaknesses: Latency creep



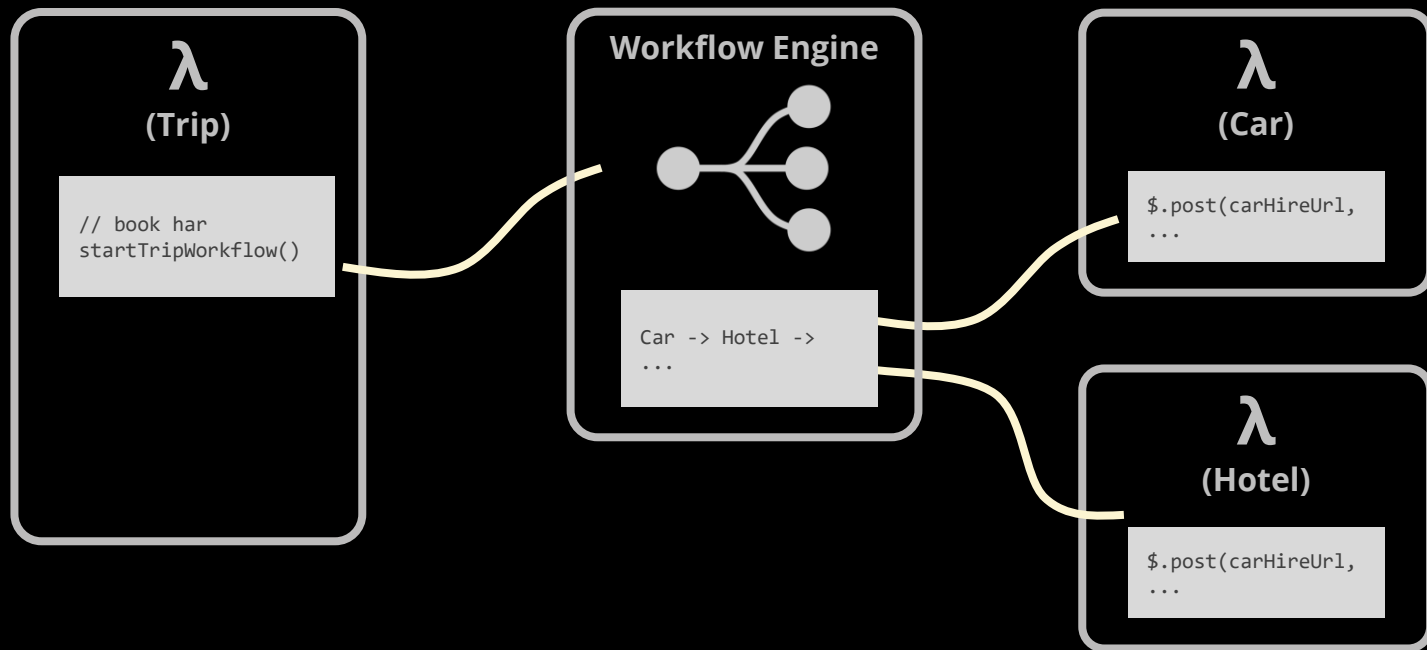
Coordinator get's long running!

Weaknesses: Availability erosion



orchestration with workflow

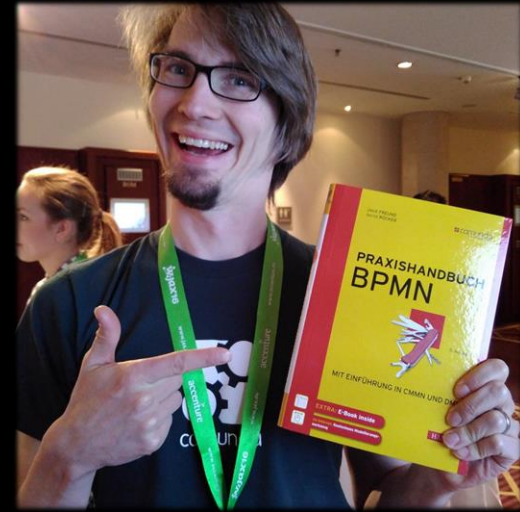
Functions keep independant
No long running function
Extra component involved



**Warning:
Contains Opinion**



Bernd Ruecker
Co-founder and
Chief Technologist of
Camunda

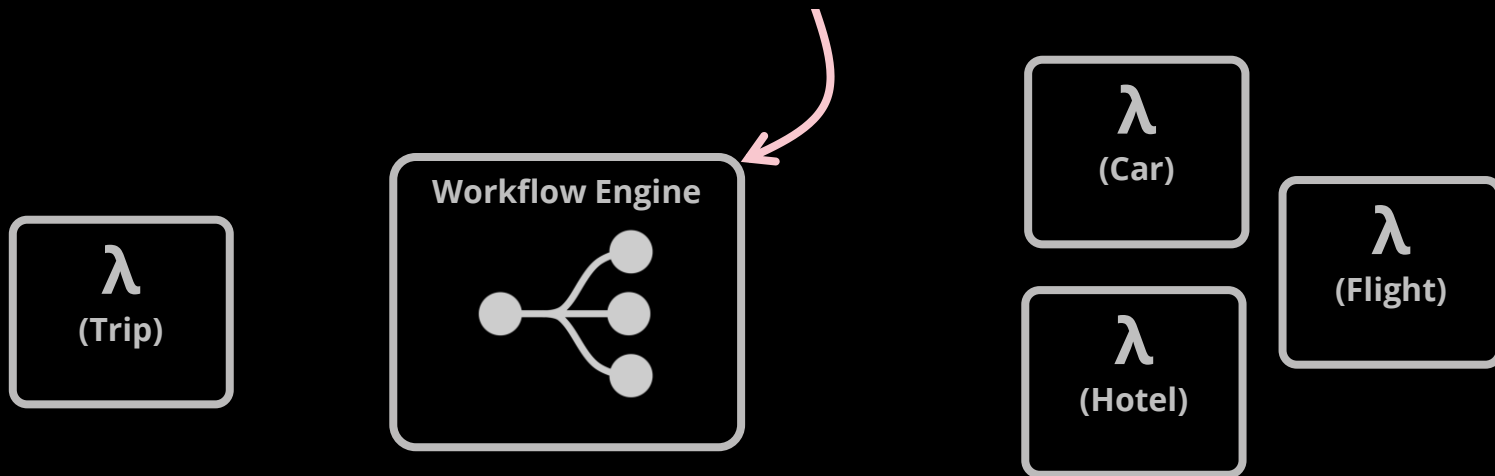
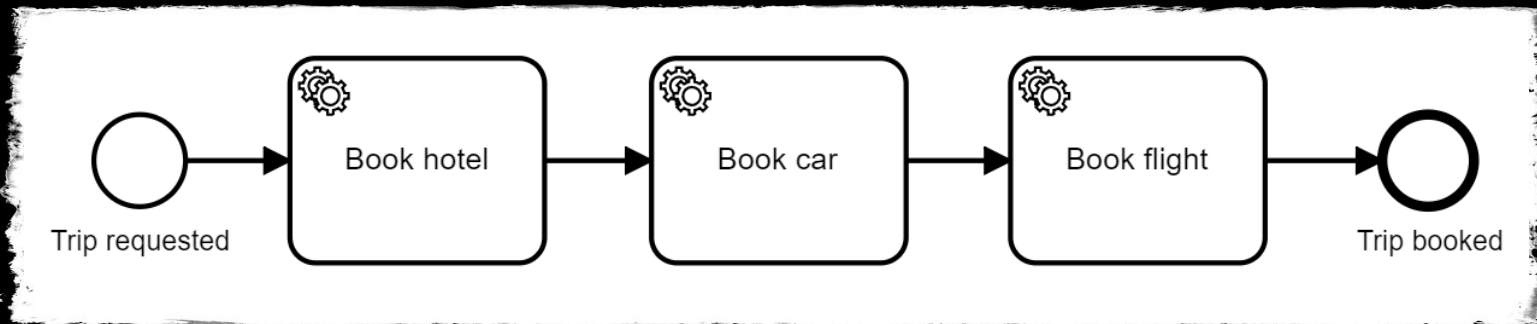


Berlin, Germany

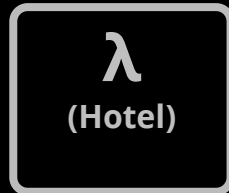
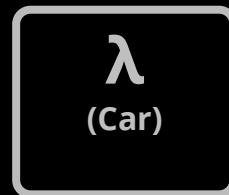
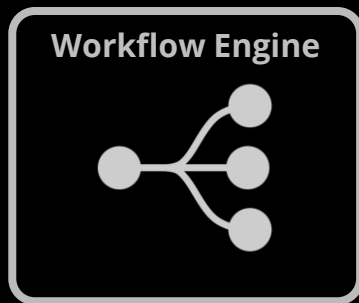
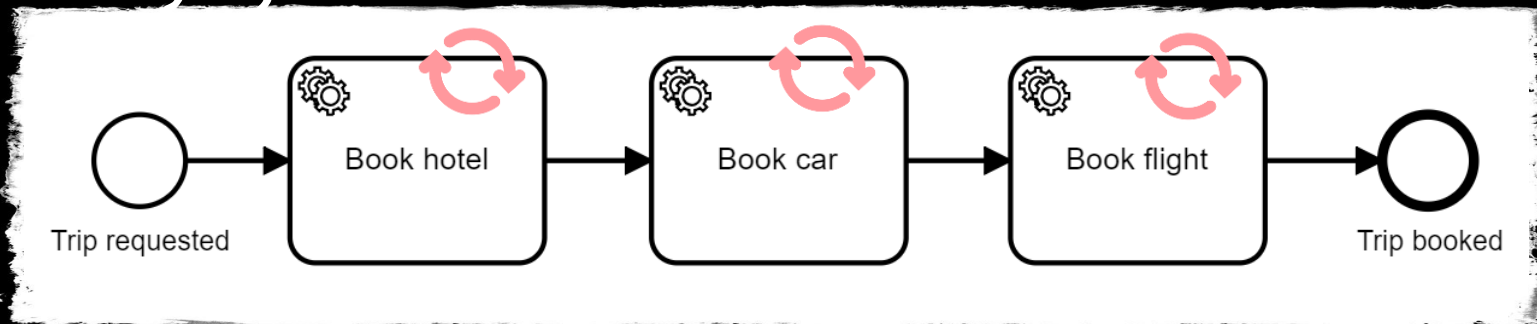


bernd.ruecker@camunda.com
@berndruecker

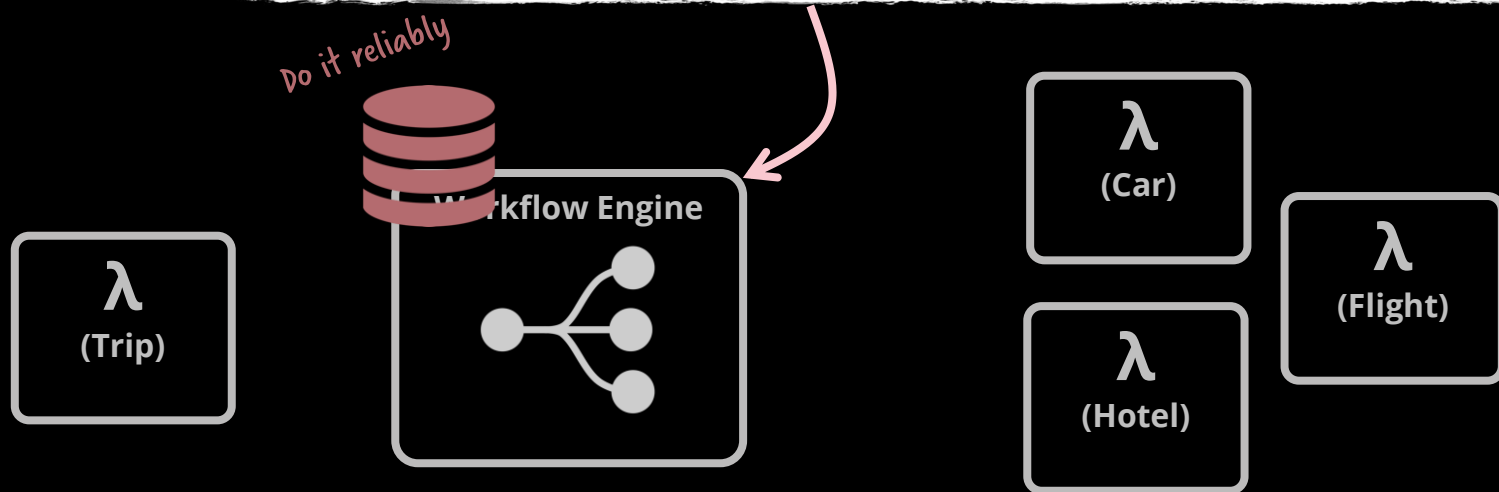
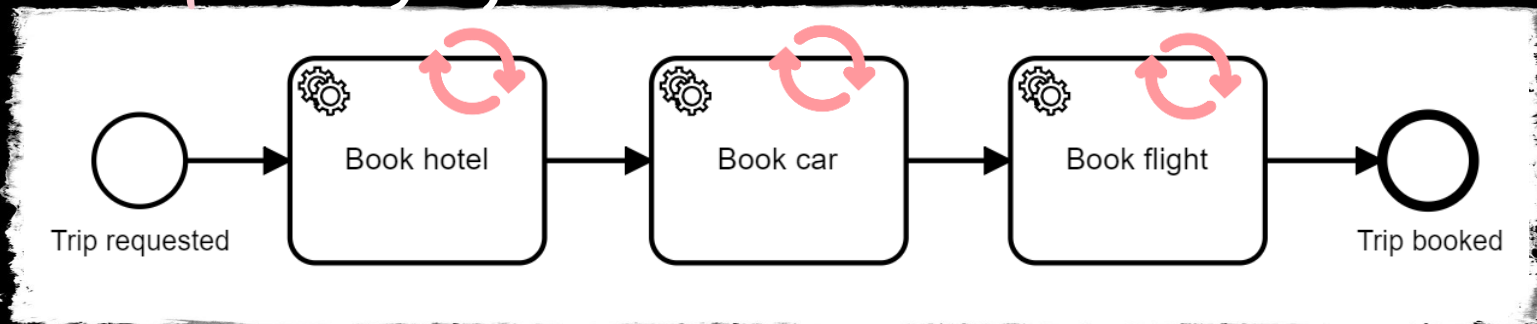
Example



Leverage retrying

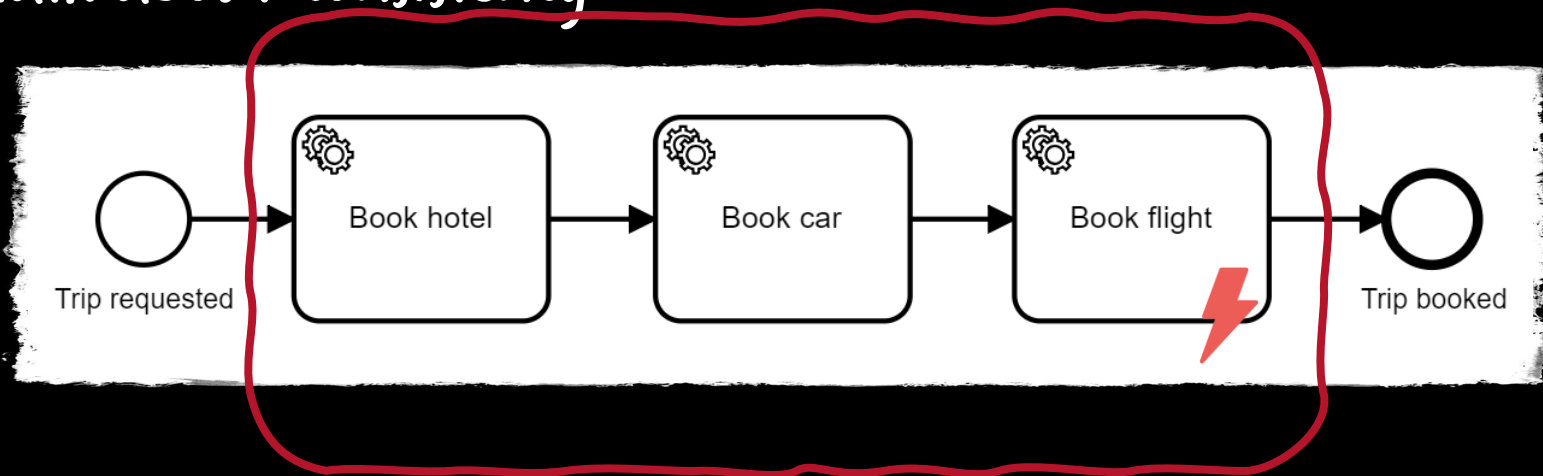


Leverage *stateful* retrying

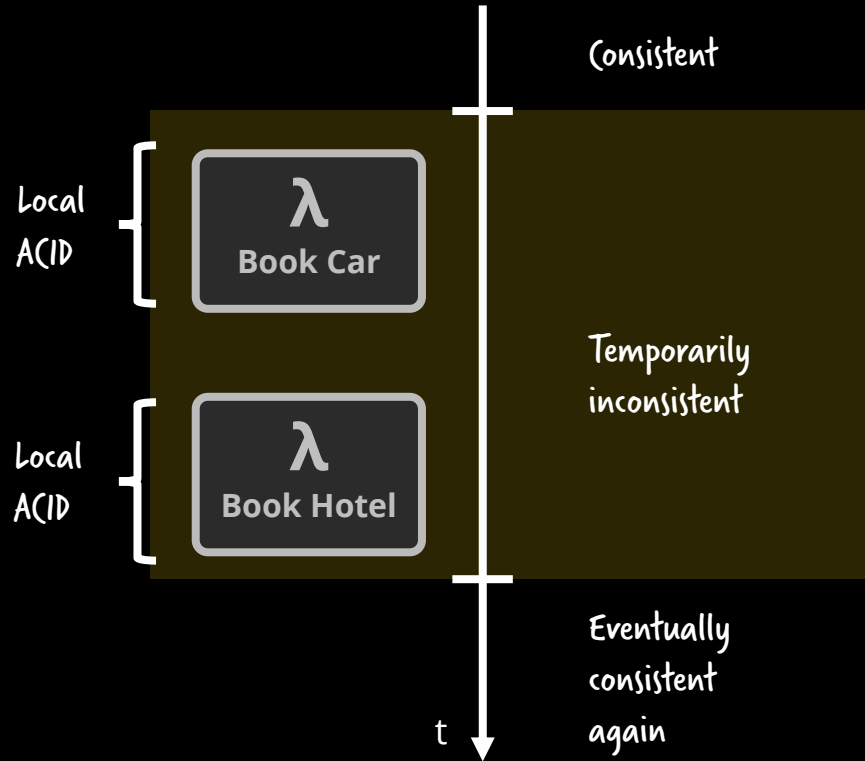


Let's talk about consistency

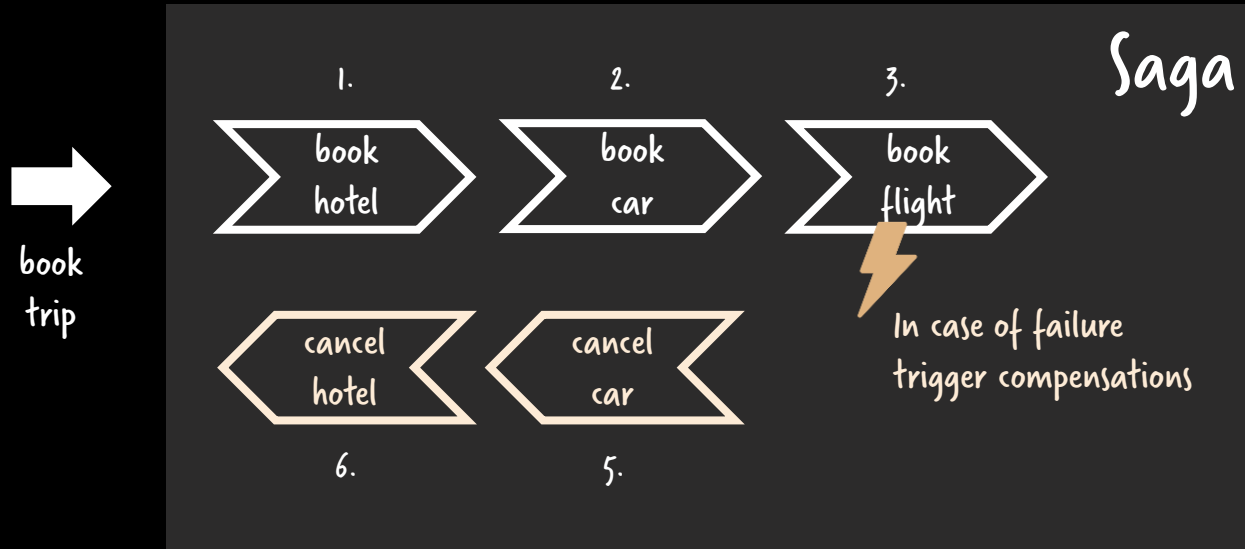
All or nothing



That means



Compensation – the classical example



CAMUNDA

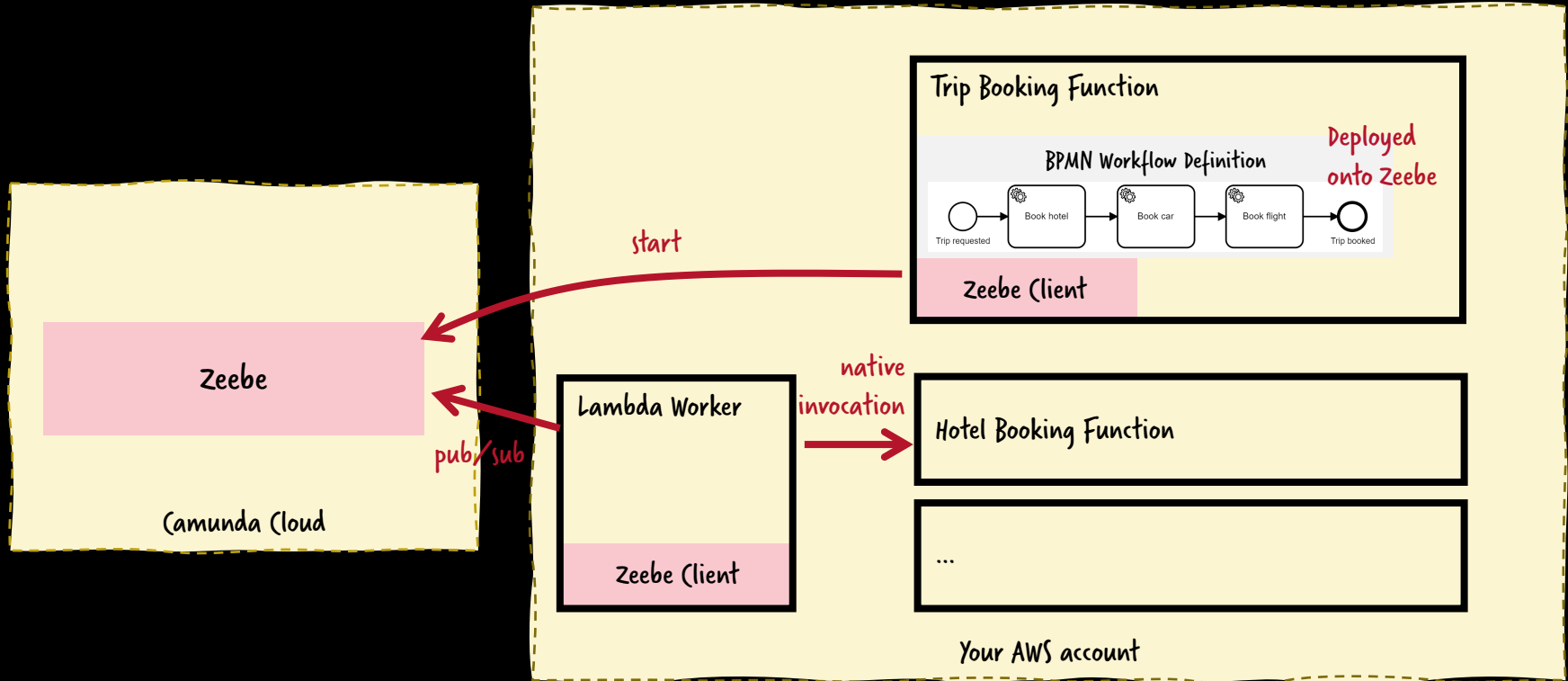
Camunda Cloud – Process Automation as a Service
Workflow engine based on BPMN
Cloud-native, Source-available, horizontally scalable

Live hacking

<https://github.com/berndruecker/trip-booking-saga-serverless>



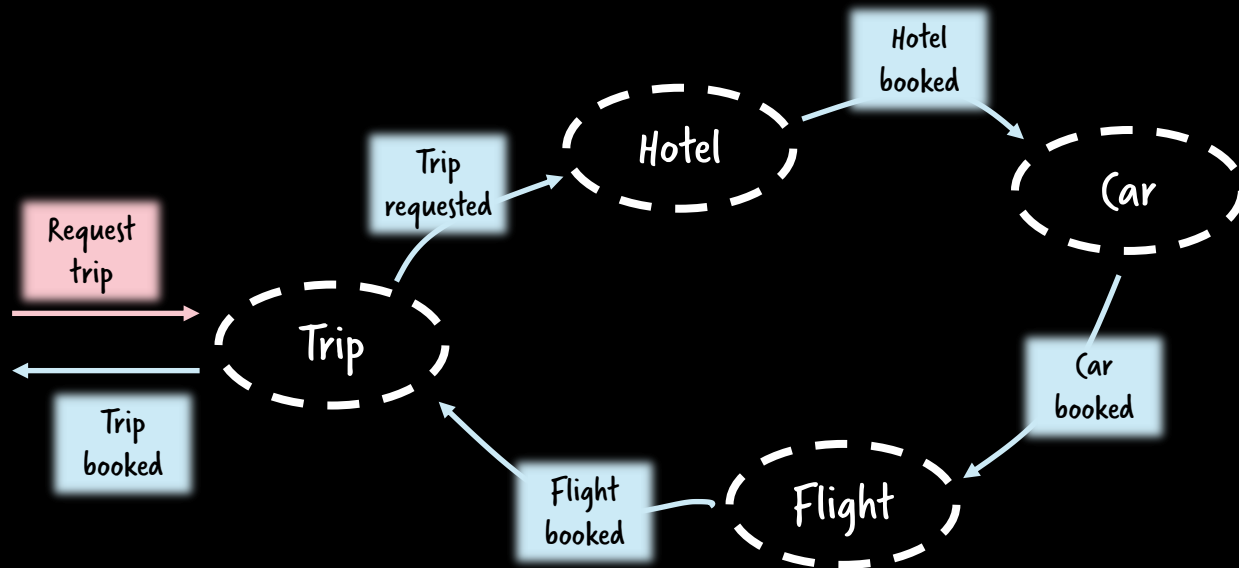
Demo Architecture



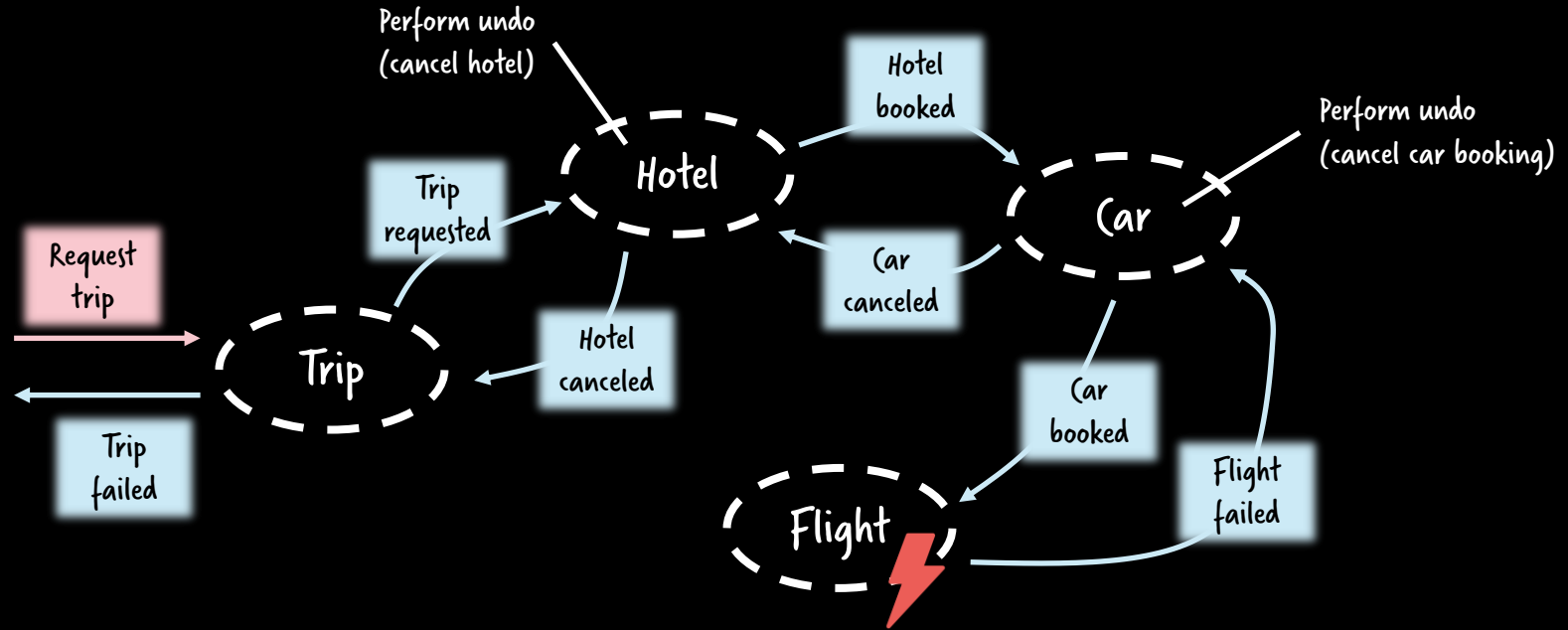
Choreography & orchestration



Event-driven choreography



Event-driven choreography





The danger is that it's very easy to make nicely decoupled systems with event notification, without realizing that you're losing sight of that larger-scale flow, and thus set yourself up for trouble in future years.

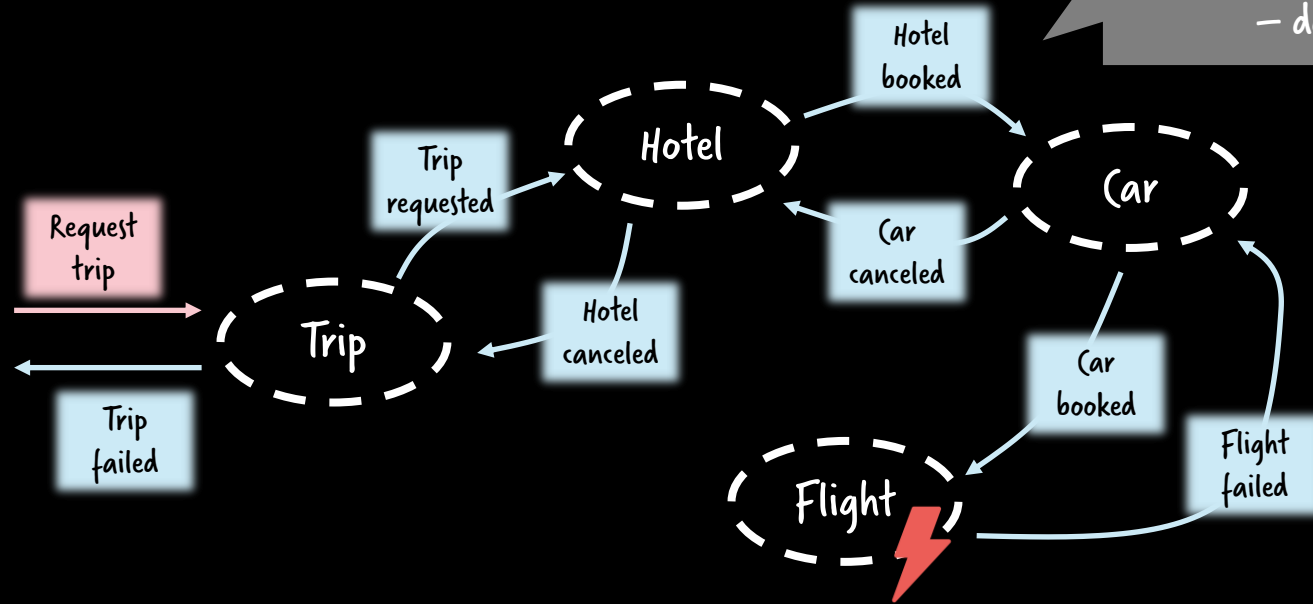


The danger is that it's very easy to make nicely decoupled systems with event notification, without realizing that you're losing sight of that larger-scale flow, and thus set yourself up for trouble in future years.



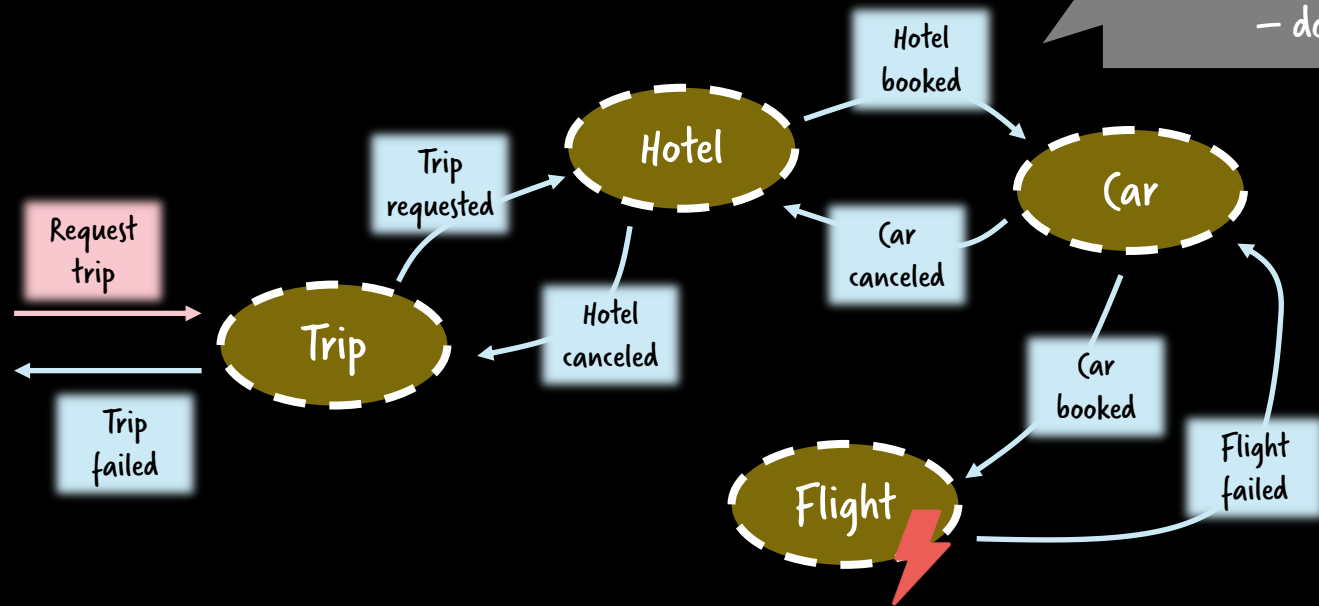
The danger is that it's very easy to make nicely decoupled systems with event notification, without realizing that you're losing sight of that larger-scale flow, and thus set yourself up for trouble in future years.

Implementing changes in the process



We have a new basic agreement with the car rental agency and can cancel for free within 1 hour – do that first!

Implementing changes in the process



We have a new basic agreement with the car rental agency and can cancel for free within 1 hour – do that first!

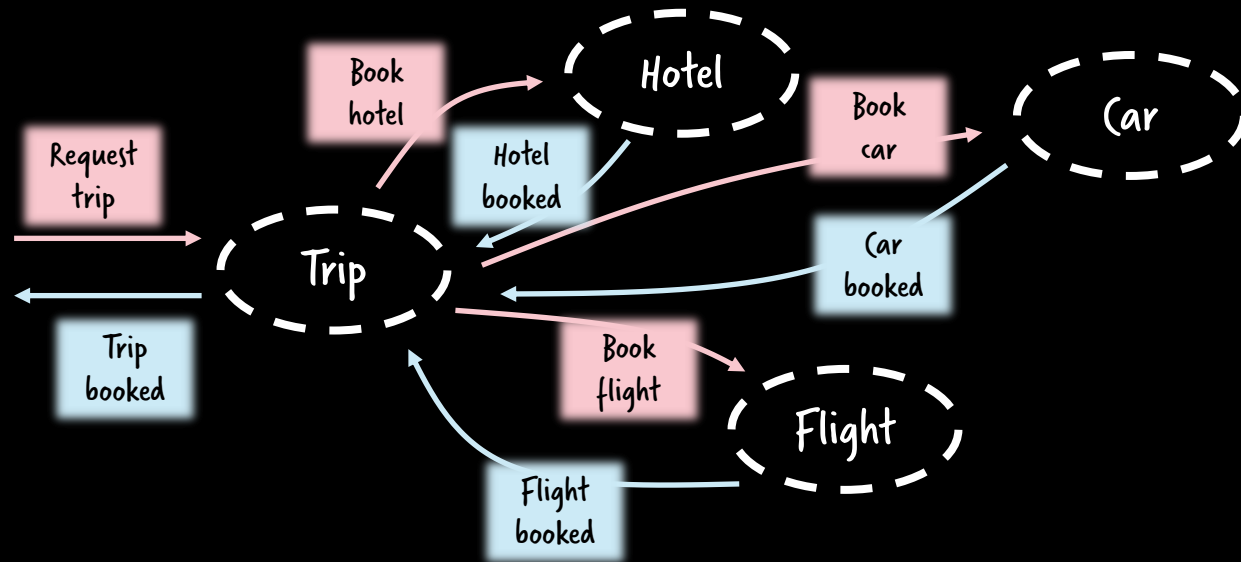
You have to adjust all services and redeploy at the same time!

What we wanted

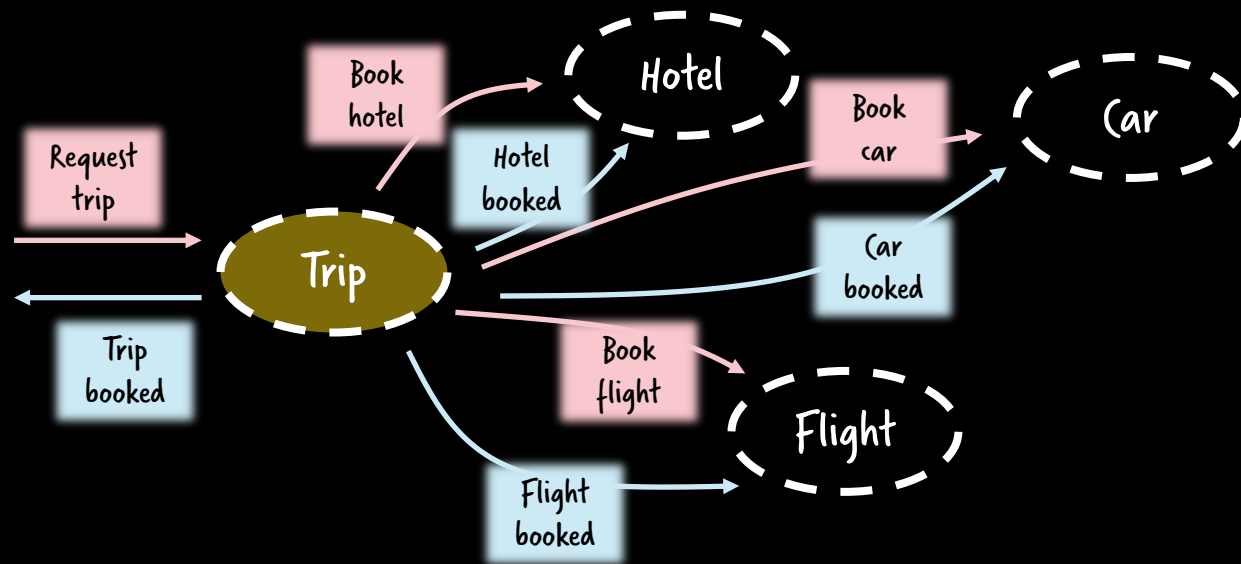


vs. what we got

orchestration



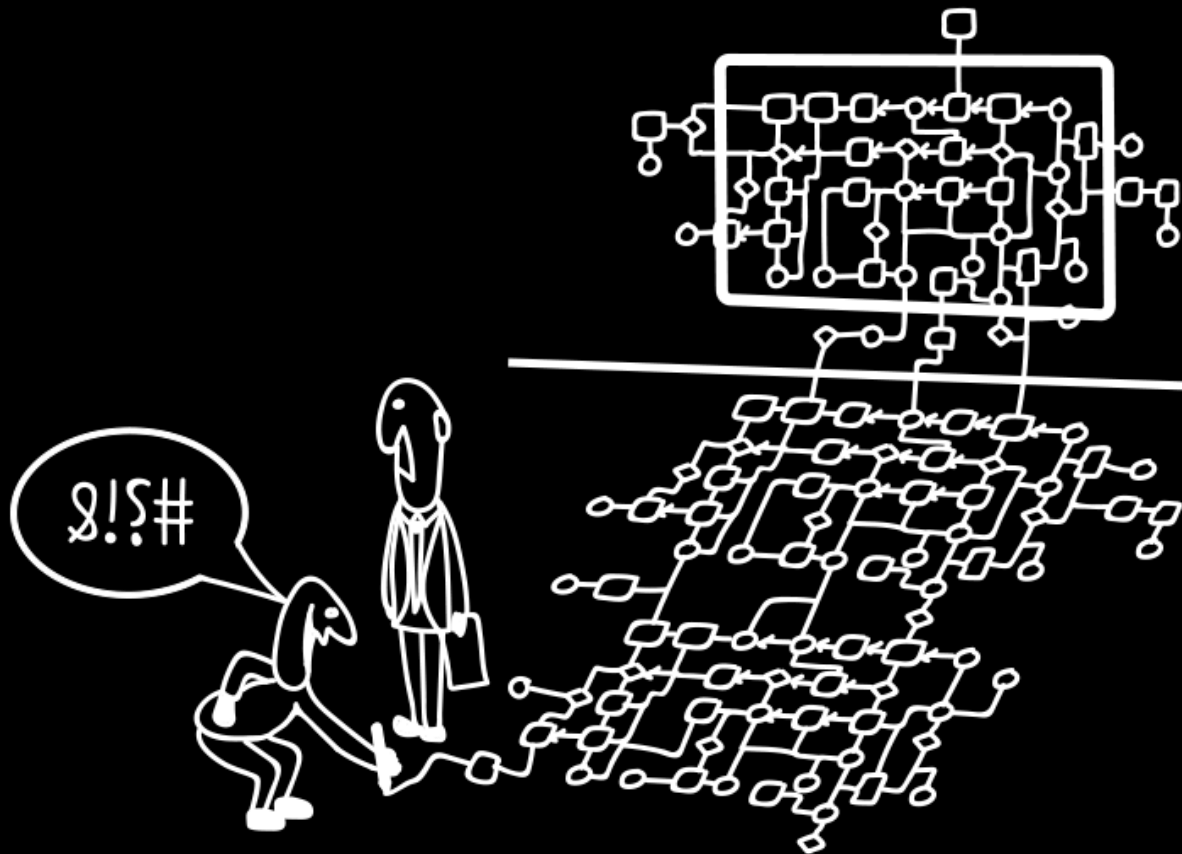
orchestration



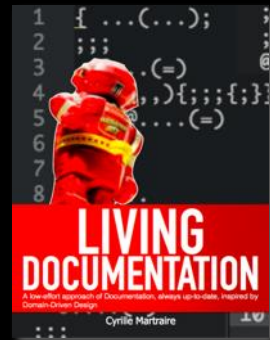
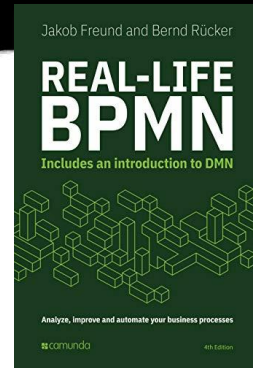
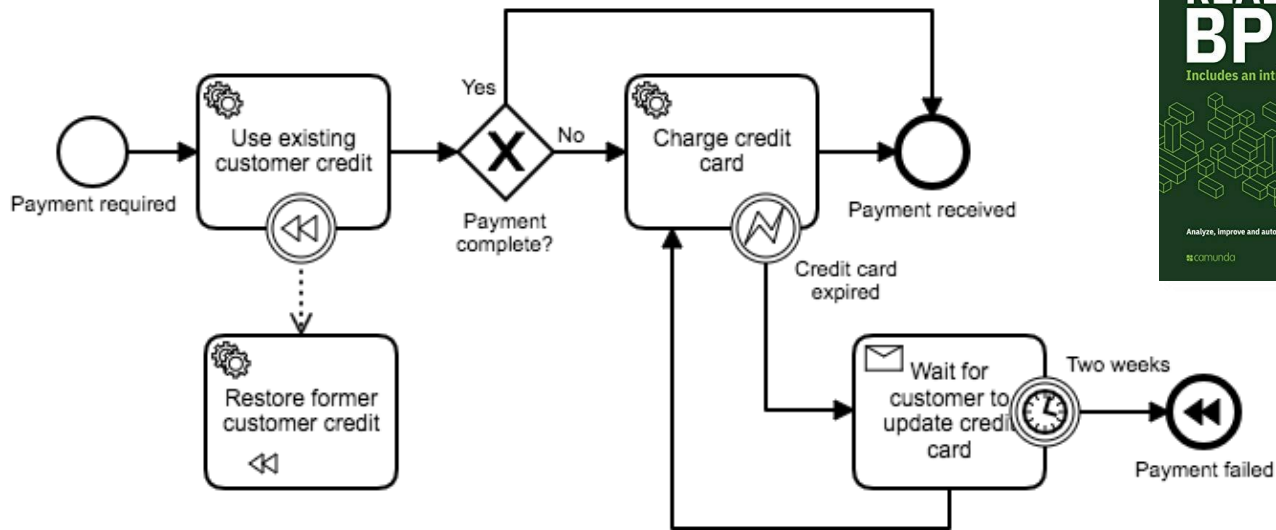
We have a new basic agreement with the car rental agency and can cancel for free within 1 hour – do that first!

You have to adjust one service and redeploy only this one!

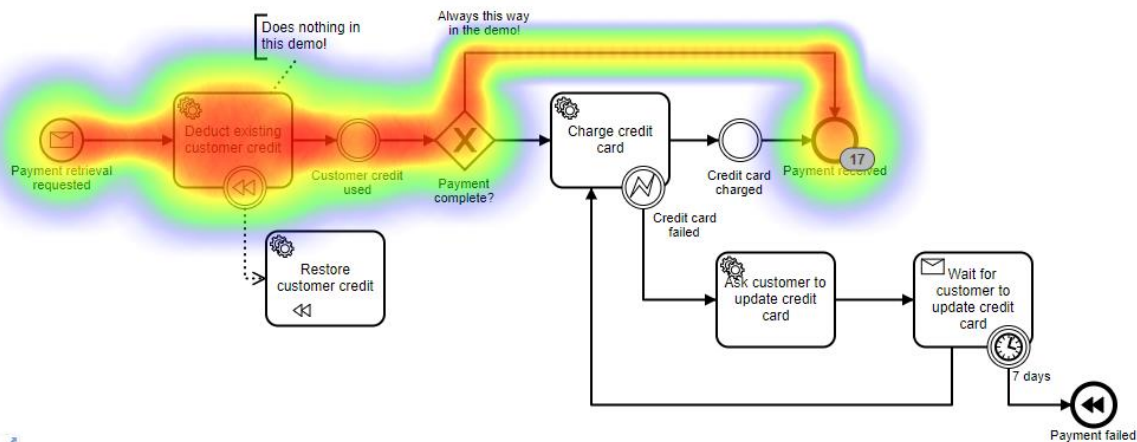
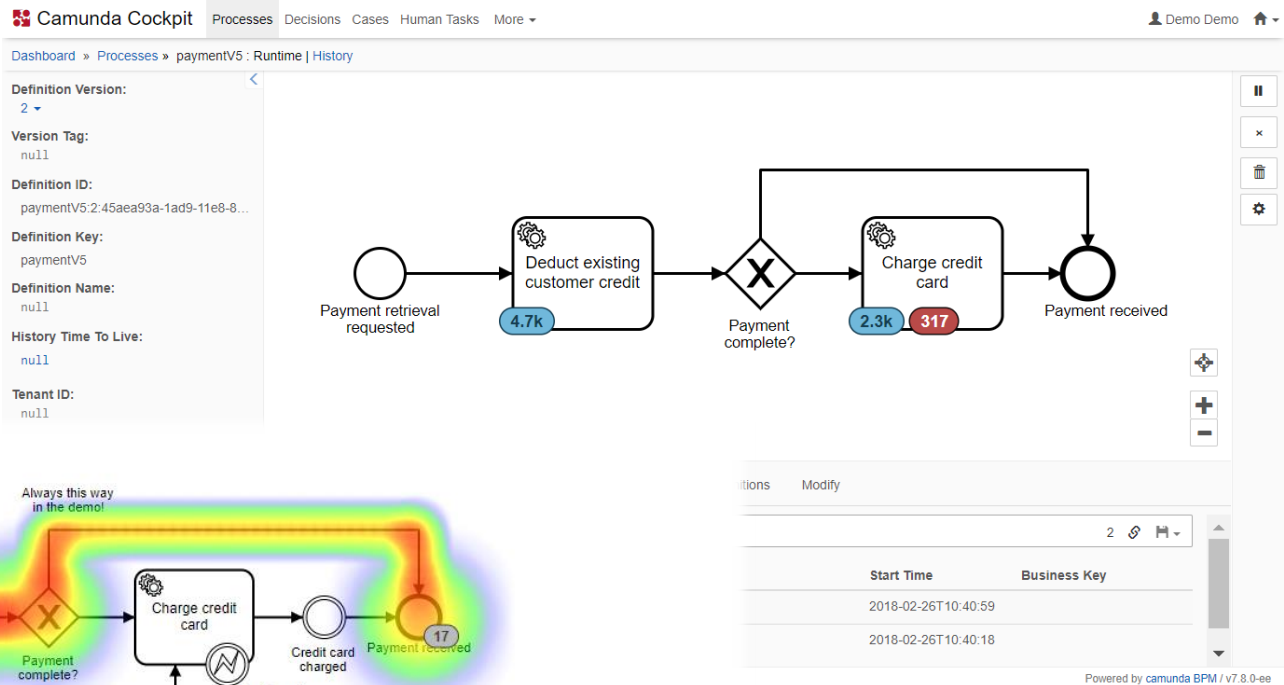
Graphical models?



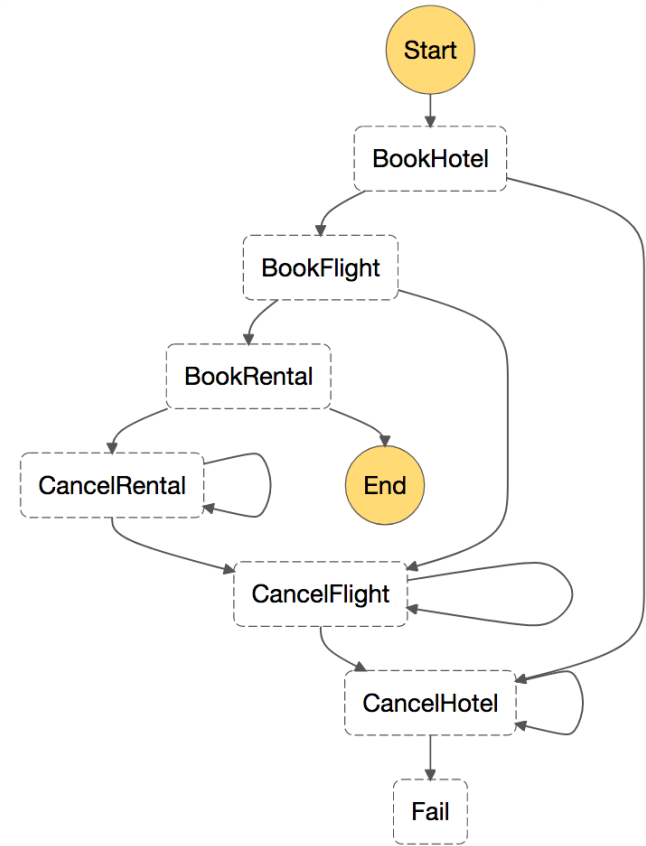
Living documentation for long-running behaviour



BizDevops

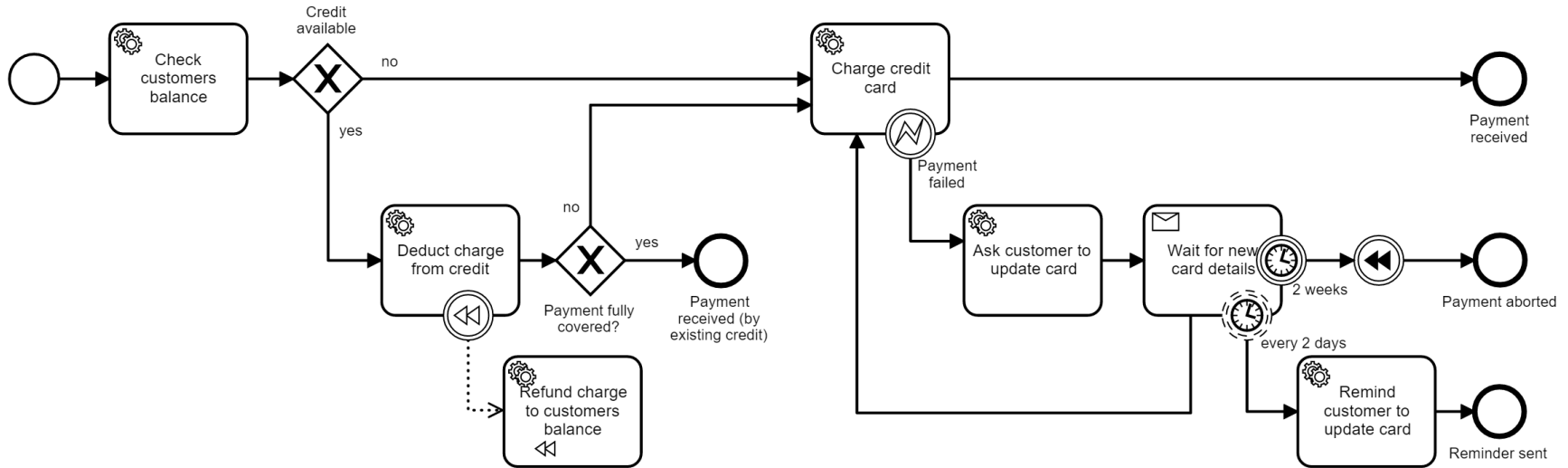


Step Functions



<https://read.acloud.guru/how-the-saga-pattern-manages-failures-with-aws-lambda-and-step-functions-bc8f7129f900>

Think of more complicated scenarios...



Recap

- Function coordination might include orchestration and workflow
- Carefully decide about choreography vs. orchestration
- Learn to handle eventual consistency and related business problems (e.g. transactions), e.g. by the Saga Pattern
- Idempotency is super important!

Thank you!



Contact: mail@berndruecker.io
[@berndruecker](#)

Slides: <https://berndruecker.io>

Blog: <https://medium.com/berndruecker>

Code: <https://github.com/berndruecker>

InfoWorld
FROM IDG

<https://www.infoworld.com/article/3254777/application-development/3-common-pitfalls-of-microservices-integration-and-how-to-avoid-them.html>

InfoQ
ueue

<https://www.infoq.com/articles/events-workflow-automation>

THE NEW STACK

<https://thenewstack.io/5-workflow-automation-use-cases-you-might-not-have-considered/>

