GOTO
Guide

LET US HELP YOU

Ask questions through **the app**

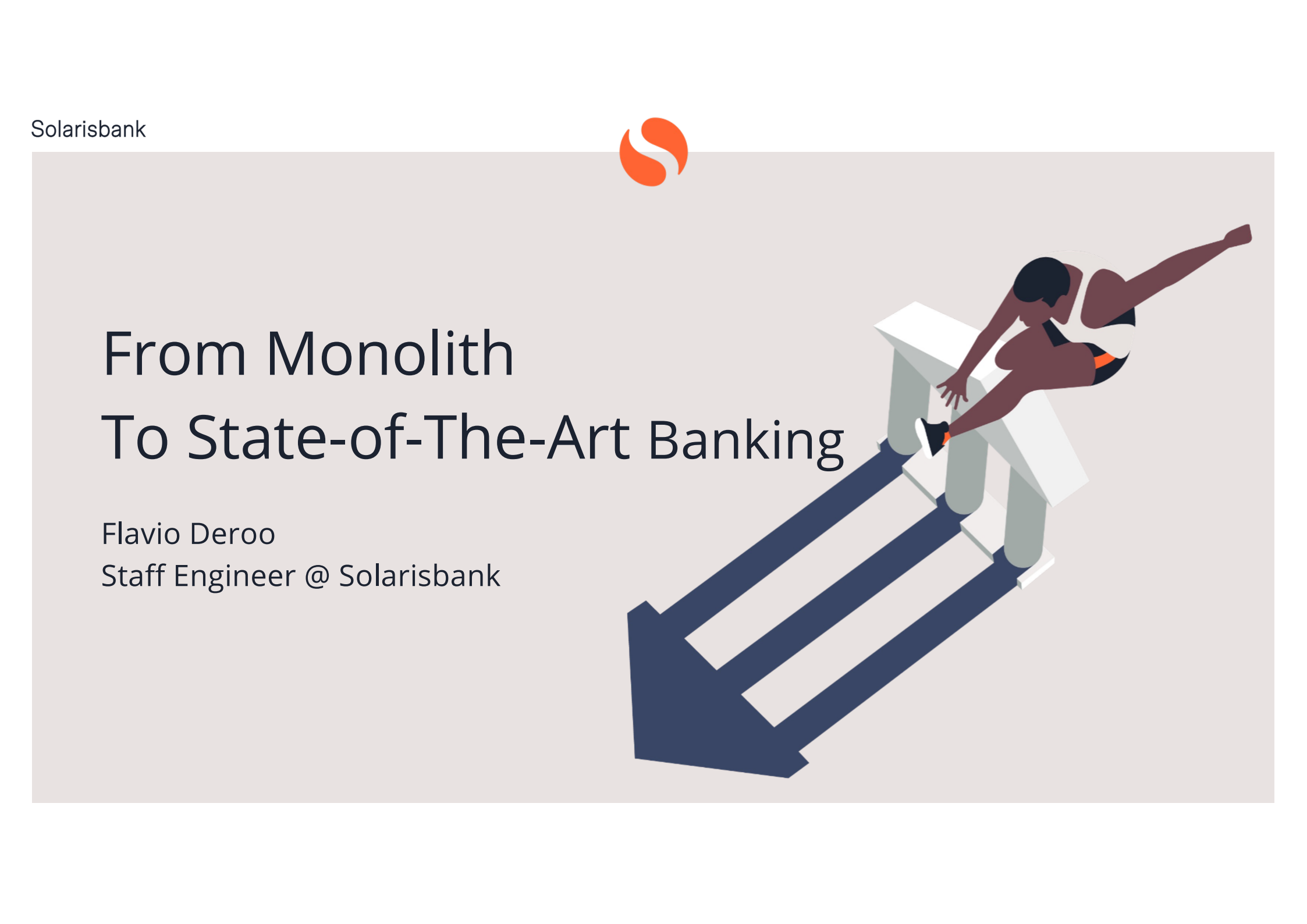
★★★★★

also remember to rate session

THANK YOU!

#GOTOams

Solarisbank



From Monolith To State-of-The-Art Banking

Flavio Deroo
Staff Engineer @ Solarisbank

Tech Company

with a Banking License

Europe Market Leader

5+ Millions accounts

450 APIs

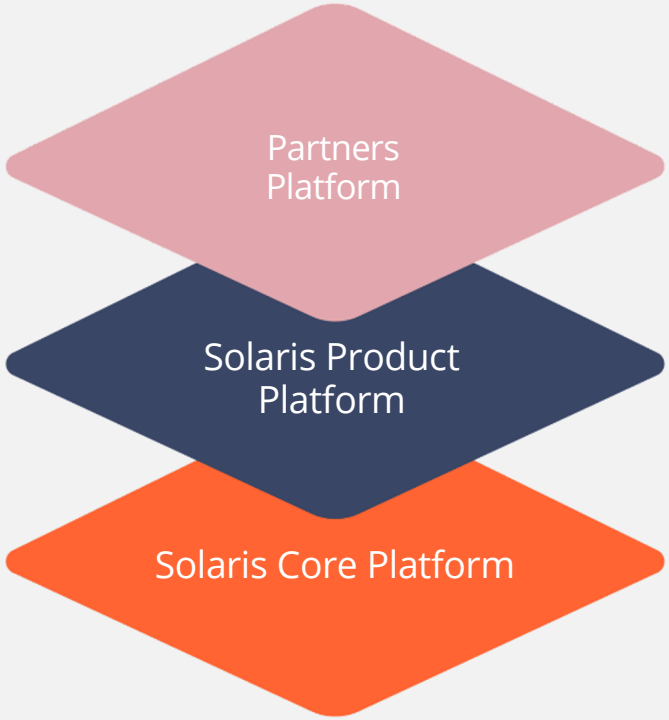
SAMSUNG

vivid

TRADE REPUBLIC

PENTA

Services Layers



Authentication
API Integration
SDK
WebUI

KYC
Payments
Cards
Lending
Regulatory
Backoffice

Accounts
Ledger
Banking Transactions
AML Detection
Reporting
Banking Messages

01. Core Banking

History of Banking Systems

1970	1980	1990	Present
Networked Branches	Mainframes	Online Banking	Banking-as-a-Service

Don't take my word for it

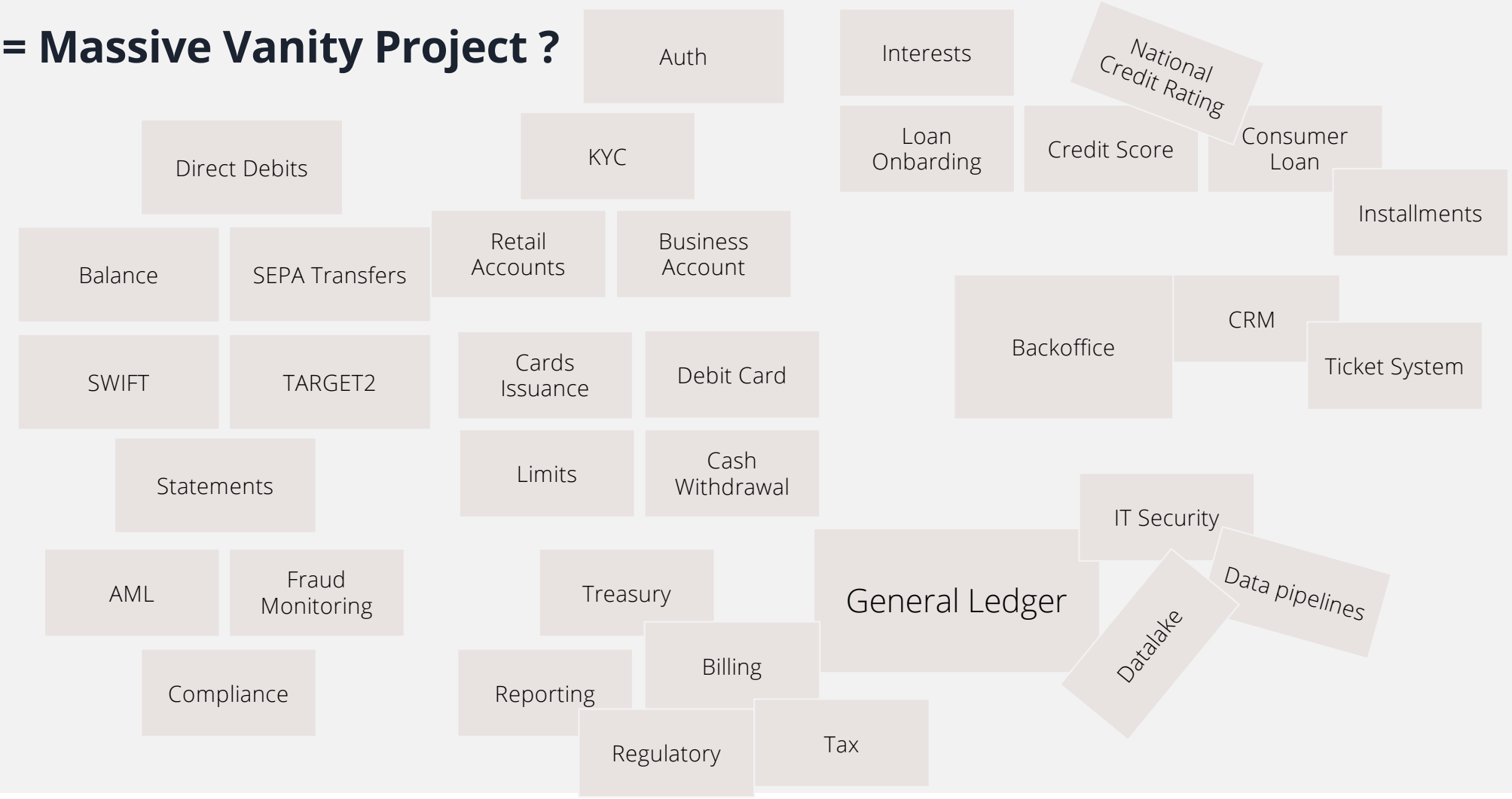
IBM COBOL

```
//COBUCLG  JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//HELOWRLD EXEC COBUCLG,PARM.COB='MAP,LIST,LET'
//COB.SYSIN DD *
  001  IDENTIFICATION DIVISION.
  002  PROGRAM-ID.  'HELLO'.
  003  ENVIRONMENT DIVISION.
  004  CONFIGURATION SECTION.
  005  SOURCE-COMPUTER.  IBM-360.
  006  OBJECT-COMPUTER.  IBM-360.
  0065 SPECIAL-NAMES.
  0066     CONSOLE IS CNSL.
  007  DATA DIVISION.
  008  WORKING-STORAGE SECTION.
  009  77  HELLO-CONST   PIC X(12) VALUE 'HELLO, WORLD'.
  075  PROCEDURE DIVISION.
  090  000-DISPLAY.
  100      DISPLAY HELLO-CONST UPON CNSL.
  110      STOP RUN.
//LKED.SYSLIB DD DSNAME=SYS1.COBLIB,DISP=SHR
//           DD DSNAME=SYS1.LINKLIB,DISP=SHR
//GO.SYSPRINT DD SYSOUT=A
//
```

From The Worst Programming Language Ever - Mark Rendle - NDC Oslo 2021

Let’s build an MVP

= Massive Vanity Project ?



Building a Bank From Scratch

in less than a year

TECH

SolarisBank Receives Banking License

March 21, 2016 By: Mike Dautner

Business bank account startup Penta hitches a ride on solarisBank

“Principal Membership”: Mastercard und solarisBank schließen strategische Partnerschaft

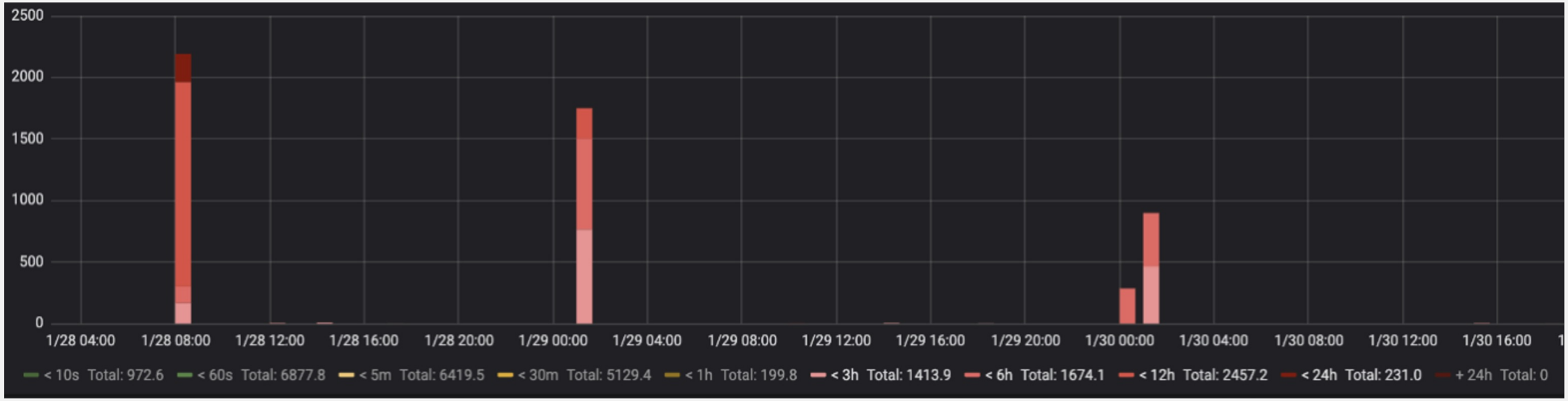
Im Rahmen der FinTech-Week Hamburg verkünden Mastercard und solarisBank (FinLeap) heute die künftige intensive Zusammenarbeit, um Innovationen im Bereich des digitalen Bankings voranzutreiben. Durch eine “Principal Membership” agiert die solarisBank zukünftig in der höchsten Mastercard-Emittentenstufe und erhält Zugang zur ganzen Bandbreite des Mastercard Produktportfolios.

Instant credit via API: AutoScout24 uses a new module from SolarisBank

The cost of moving fast

Performance abyss

All your grafana dashboards are red



Grafana archives, colorized, 2017

Ticking time bomb

Performance reports from an Engineer, 2017

Observations: It takes between 10 sec. - 30 min. to book SCT during the business day

Internal memo from the CPO

Problem statement:

The Core-banking performance is stalling into an unstable state, [redacted] Since TR indicates increased growth (500k new vIBANs [redacted]) we need to take pre-cautionary measures to prevent it from collapsing.

Solarisbank

02. You cannot optimize your way out of a bad design

A World With Constraints

Invariant constraints in the banking world

Strict(er) SLOs

Frequent patches

Cost-efficiency

Scalable

Build for Scalability

Invariant concepts in our tech stack

Event sourcing

Golang

Kubernetes

Cloud Native

DynamoDB & PostgreSQL

Solarisbank

Event Sourcing

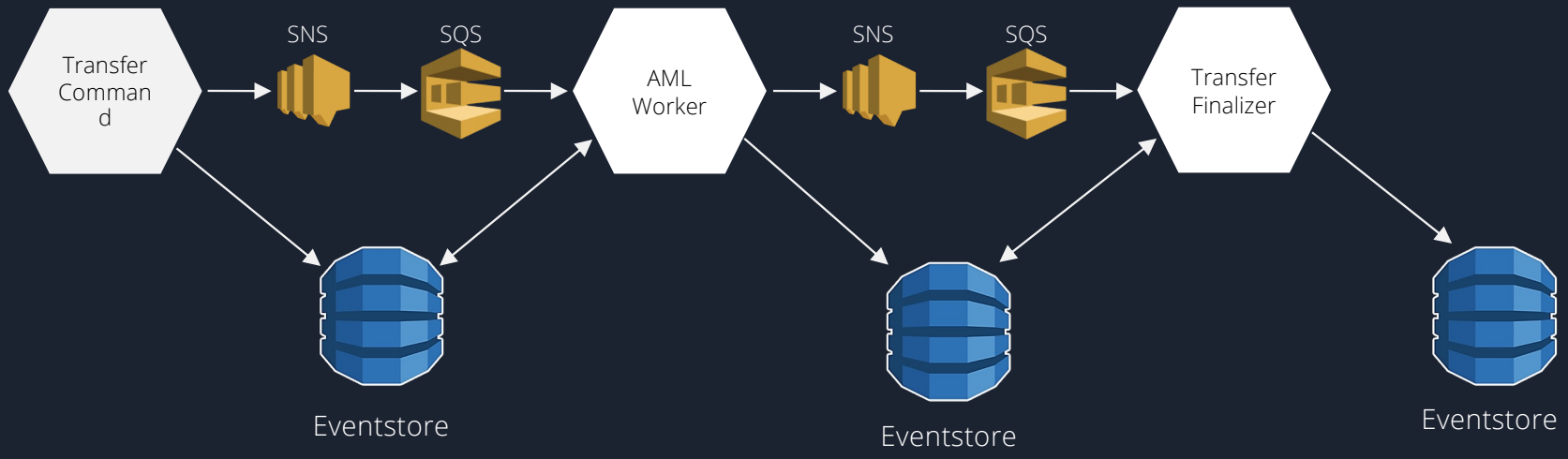
"Capture all changes to an application state as a sequence of events"

"Create a new bank transfer !"

```
POST /transactions/credit_transfer
```

"Is it compliant ?"

"Finalize the transfer."



Solarisbank

Aggregating Events

Each steps produces events, which needs to be aggregated to figure out the next step

1 Transfer API

```
{
  "uuid" : "294992d2-7fbe-4eef-a42f-b9ef72861552",
  "aggregated_uuid": "1234abcd-0000-1111-2222-dcba43214321",
  "sequence": "1",
  "action": "credit_transfer.initiation"
  "metadata": {
    "created_at":1618322847,
    ...
  },
  "payload":{
    "amount": 50000,
    "creditor_iban": "FR45100110109432010",
    "debtor_iban": "DE45100110103421939",
    "settlement_date": "2021-05-03"
  }
}
```

2 AML Service

```
{
  "uuid" : "5bdd202c-1c8c-4d65-8f4d-7080f1a6cc66",
  "aggregated_uuid": "1234abcd-0000-1111-2222-dcba43214321",
  "sequence": "2",
  "action": "aml_check.ok"
  "metadata": {
    "created_at":1618323126,
    ...
  },
  "payload":{}
}
```

3 Finalizer Service

```
{
  "uuid" : "a1230047-40bf-4c29-adcd-d0aa9b92fedc",
  "aggregated_uuid": "1234abcd-0000-1111-2222-dcba43214321",
  "sequence": "3",
  "action": "credit_transfer.ready_for_export"
  "metadata": {
    "created_at":1618323126,
    ...
  },
  "payload":{}
}
```

Append Only

Immutable

Not commutative

uuid	aggregated_uuid	sequence	action	metadata	payload
294992d2-...	**1234abcd**-...	1	credit_transfer.initiation	{...}	{...}
5bdd202c-...	**1234abcd**-...	2	aml_check.ok	{...}	{...}
a1230047-...	**1234abcd**-...	3	credit_transfer.finalized	{...}	{...}

All events of an entity are visible in a sequence, aggregating each events gives us all information regarding the amount, IBANs, checks, status...

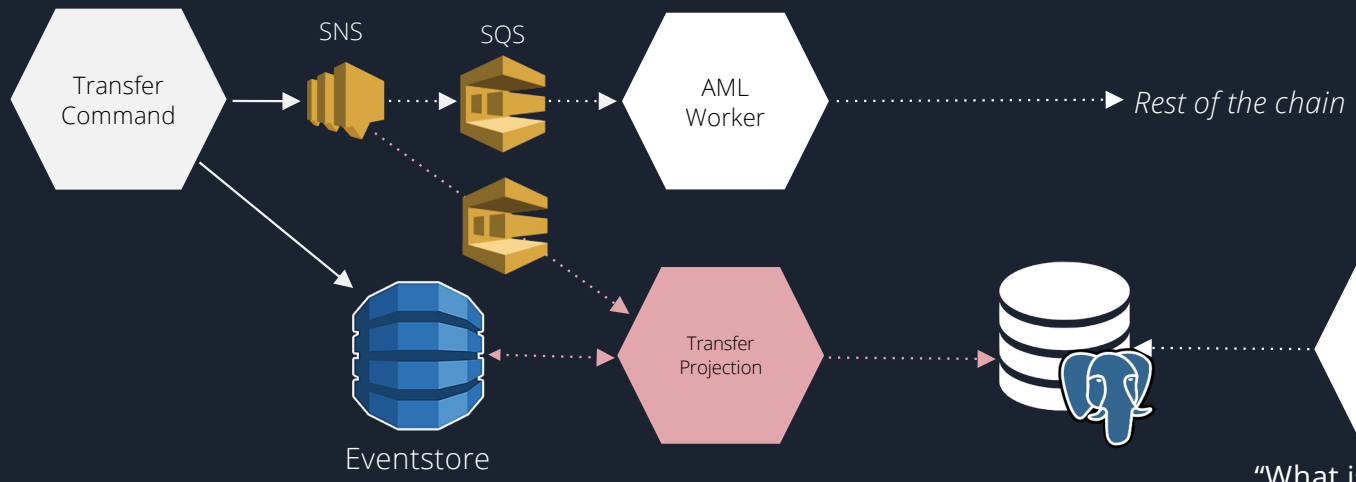
Solarisbank

CQRS

Projection & Queries

"Create a new bank transfer !"

```
POST /transactions/credit_transfer
```



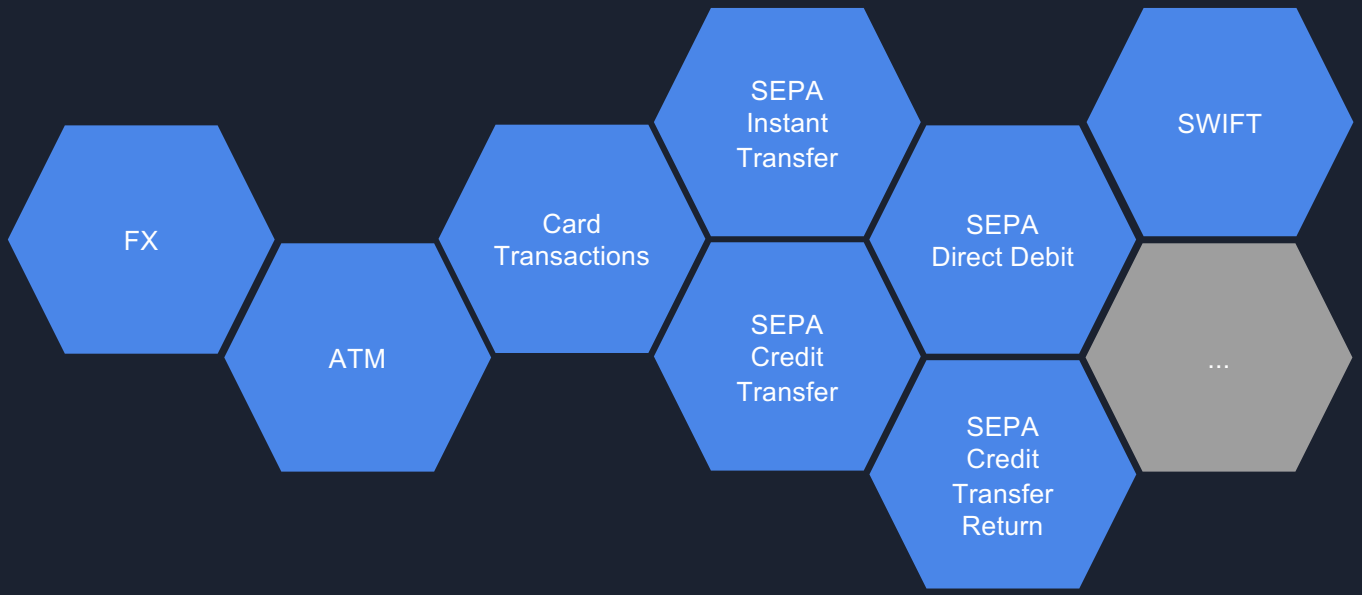
"What is the status of this transfer?"

```
GET /transactions/{uuid}
```

Solarisbank

The Ledger

It’s just another Projection



What does all those events have in common ? **They move money**

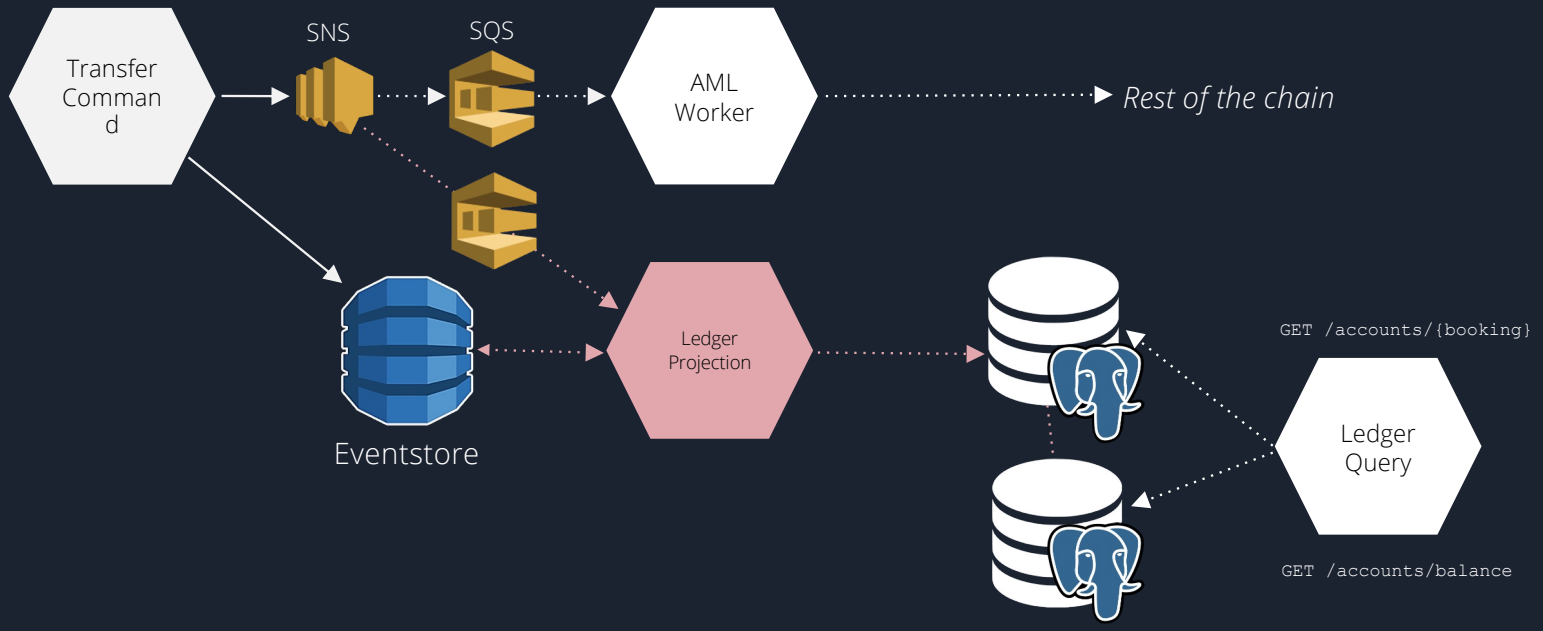
Solarisbank

Ledger Projection

Projection & Queries

"Create a new bank transfer !"

```
POST /transactions/credit_transfer
```



Solarisbank

Double Entry

Concept & Challenges

transaction_uuid	amount	iban	transfer_type	settlement_date	last_update
awsfinte-...	50000	FR45100110109432010	SEPA_CREDIT_TRANSFER	“2021-05-03”	“2021-05-03T01:44:18+2318”
awsfinte-...	-50000	DE45100110103421939	SEPA_CREDIT_TRANSFER	“2021-05-03”	“2021-05-03T01:44:18+2318”

Σ {entries} = 0

The sum of all ledger entries will always equal to 0

Balance = **Σ {entries(iban)}**

The balance of a customer is equal to the sum of all of its entries

Solarisbank

03. “Distributed software has a major disadvantage, the fact that it's distributed”

mandatory Martin Fowler quote

Dealing With Eventual consistency

Be prepared to handle timing attacks

Solarisbank

Can be eventually consistent

Transfers*

Imports

Direct Debits

Overdraft

Cannot be eventually consistent

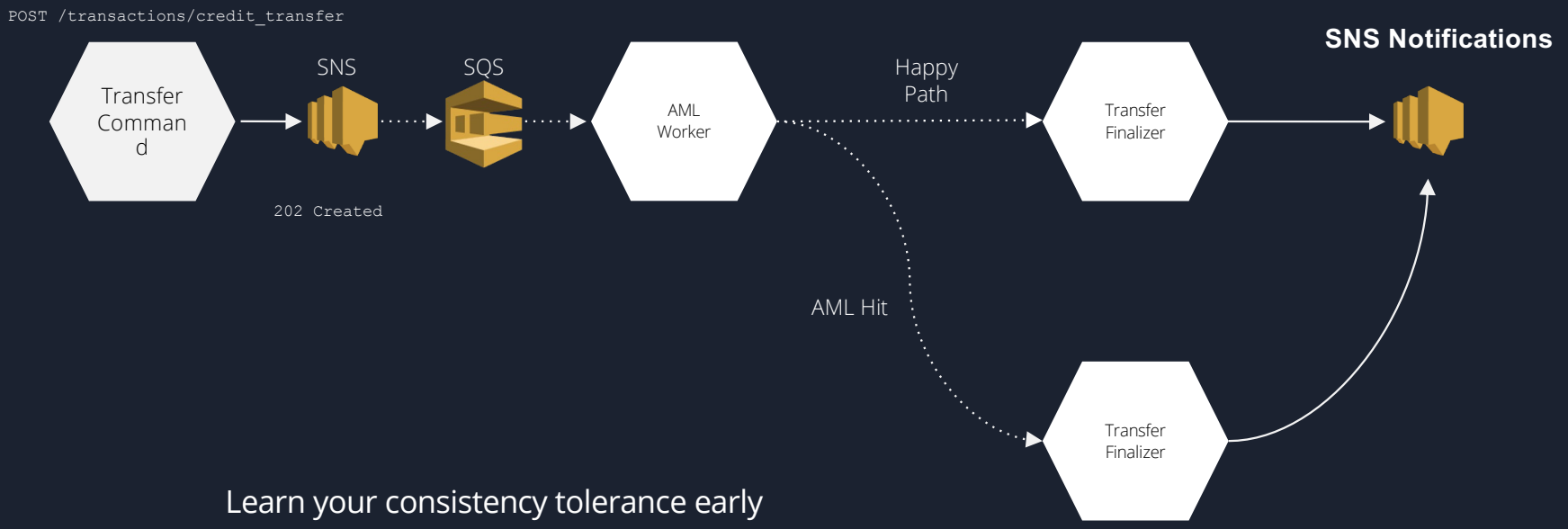
Account Blocks Checks

Account Closures Checks

Transfer Authorization

Dealing With Async

Build sensible business process around your architecture



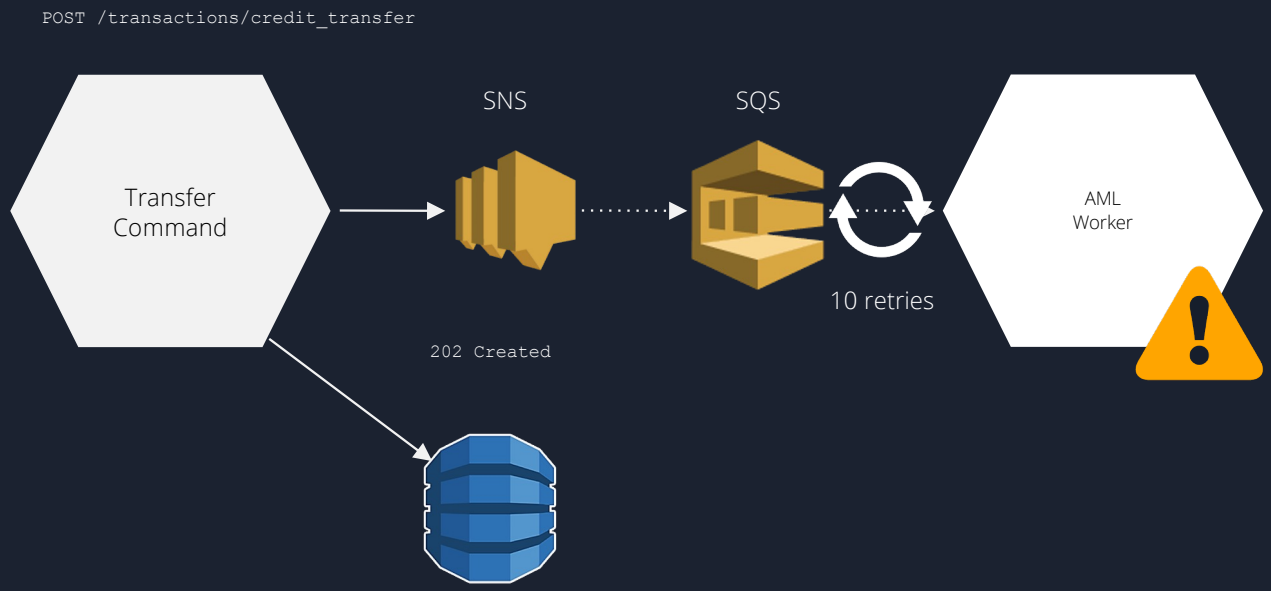
Learn your consistency tolerance early

Be crystal clear in your API about codes meaning

Solarisbank

Dealing With Event Chain Faults

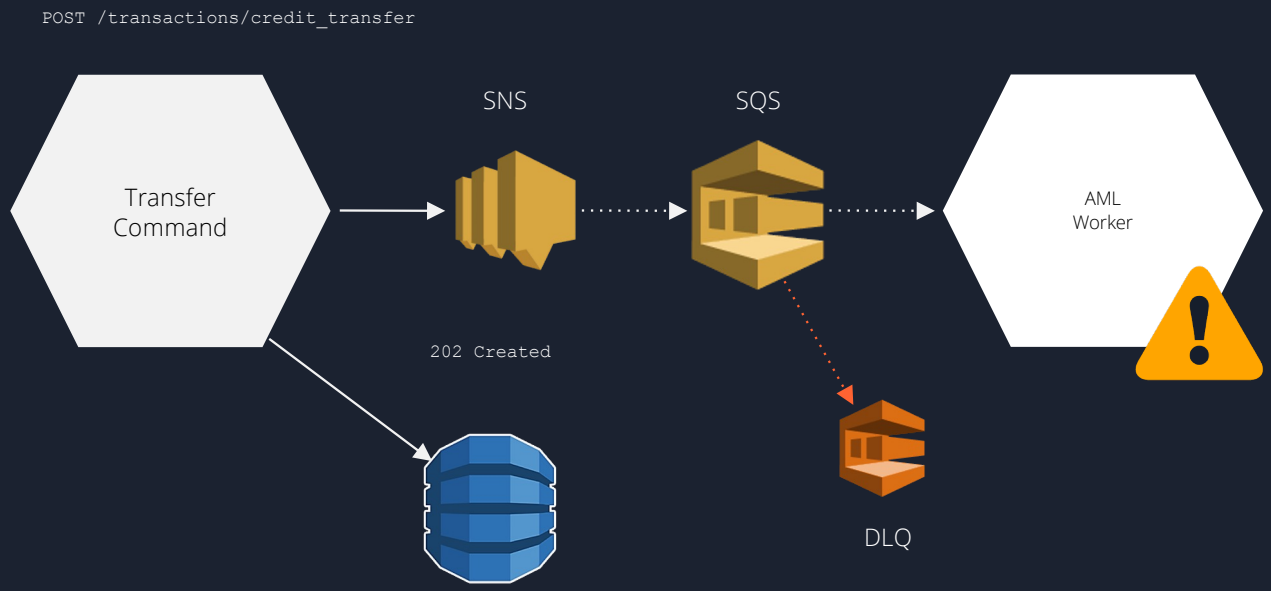
Everything will break, at least some of the time



Solarisbank

Dealing With Event Chain Faults

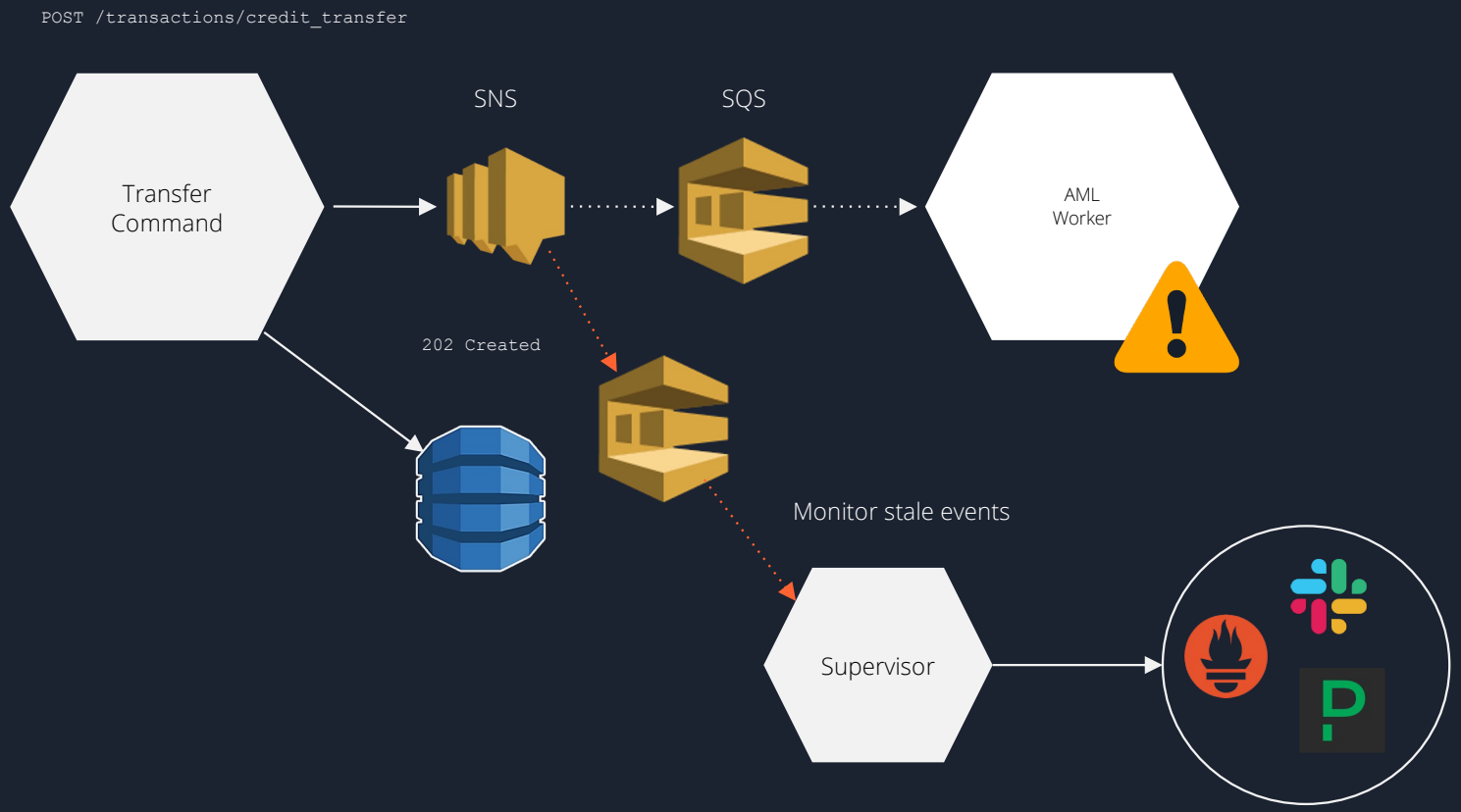
Everything will break, at least some of the time



Solarisbank

Dealing With Event Chain Faults

Everything will break, at least some of the time



Solarisbank

Dealing with Infrastructure Faults

Load Balancing & Autoscaling

Every Service must have multiple replicas, and horizontal autoscaling

Solarisbank

Dealing with Infrastructure Faults

Distribute your instances across multiple nodes

Solarisbank

Dealing with Infrastructure Faults

Predictable builds, progressive deployments, easy rollbacks

Multiple
deployments a day

Progressive
deployment strategies

Everything
as a code

Solarisbank

04. Migration Plan

Migration Timeline

Solarisbank

Kickstart

Architecture bases, tooling, hiring, goal setting

Q2 2018

Central Bank Registration

First productive transfer using the new platform.

Greenlight by the Central bank

Q3 2019

All Partners Broadcast

All new account runs on the new system.

All customers are notified about the migration plan.

Q4 2020






Q1 2019

Accounts Wrapper

New BIC opening

Sync of accounts with the Legacy

Q2 2020

Beta

Onboarding of a new partner using the new Platform with a large feature set

Q2 2021

Migration

All customers are moved to the new platform, all dependencies switch their host.

Shutdown of the legacy.

Migration

Challenges

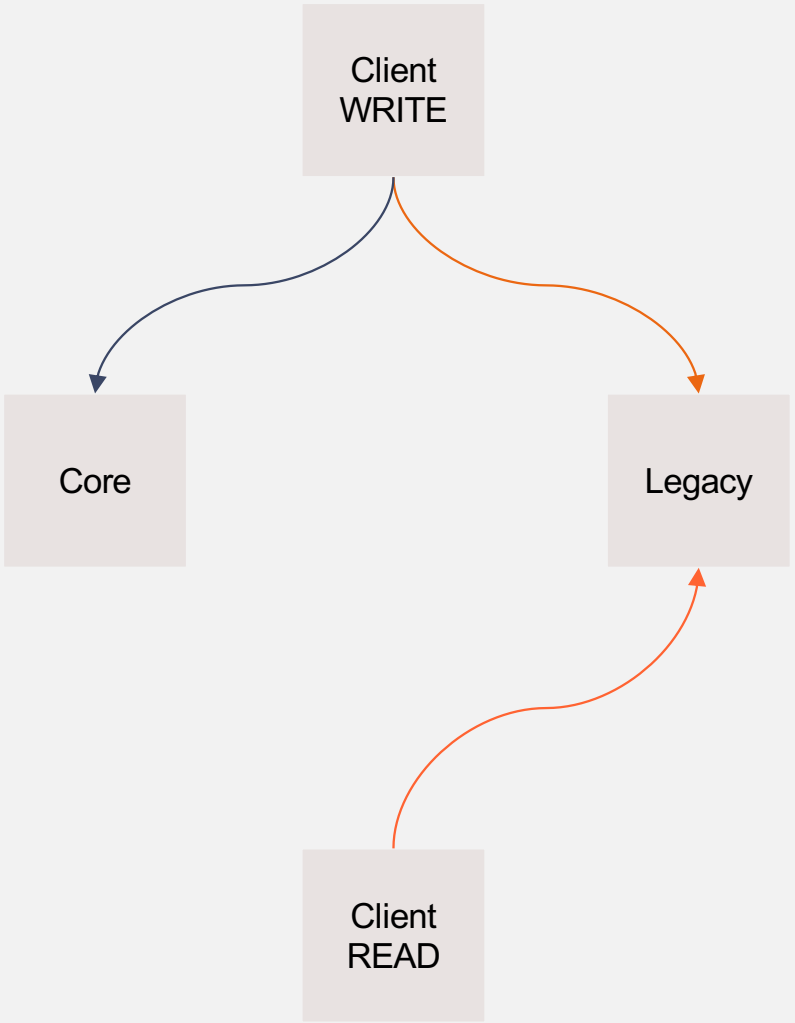
- Structuration of data different between the platforms
- 0% consistency failure tolerance
- (Ideally) No downtime
- Feature Parity
- Strict Deadline
- Must handle triple the load overnight
- Processes and Contingency for ongoing banking operations (how to handle cross-platform returns ?)

Migration - D-60

Double Writing

- Idempotency
- Sync vs Async copies

Inevitably, the transition period produces incorrect balances

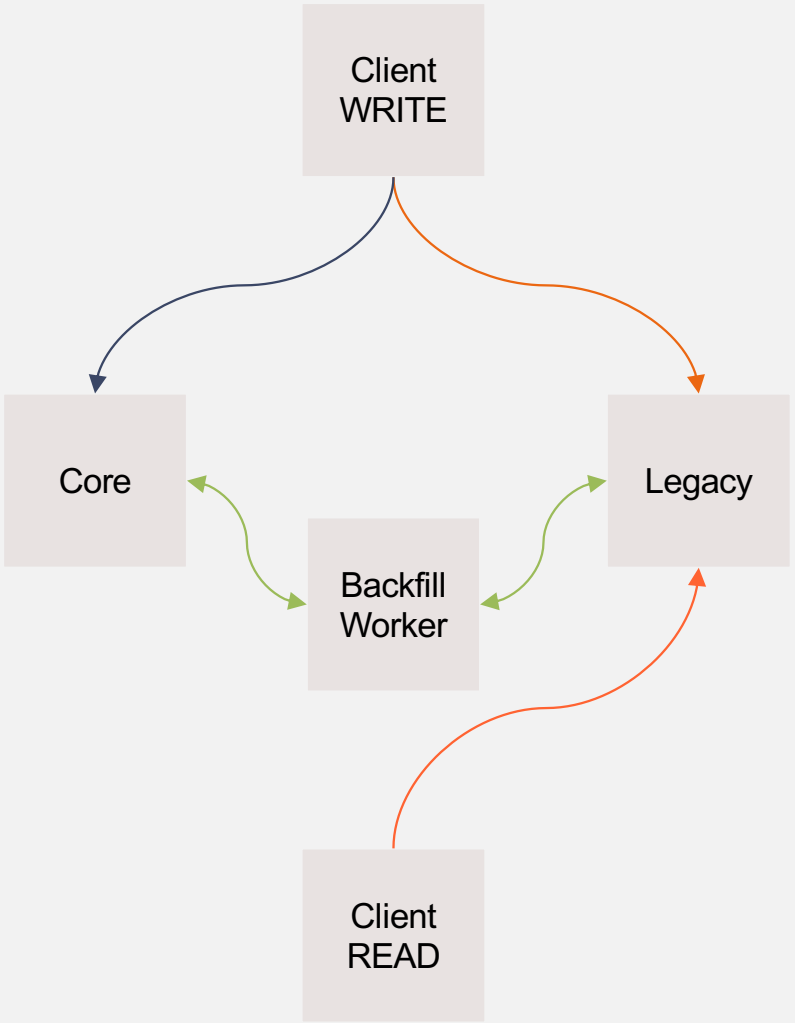


Solarisbank

Migration - D-30

Backfill Data

- Start from the oldest entries
- Make your backfill endpoint idempotent
- Copy - Verify - Mark as done



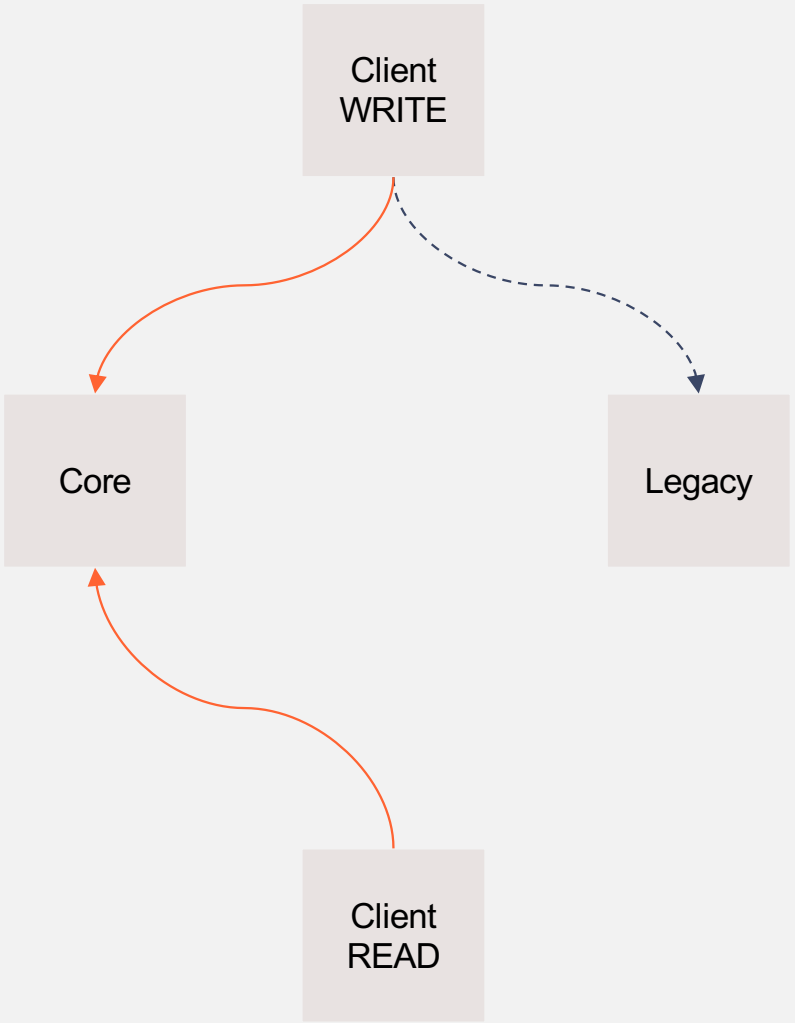
Solarisbank

Migration - D-0

Switch to the new system

Most if not all of the data should be in sync. Reconciliation checks returns balance equality all across.

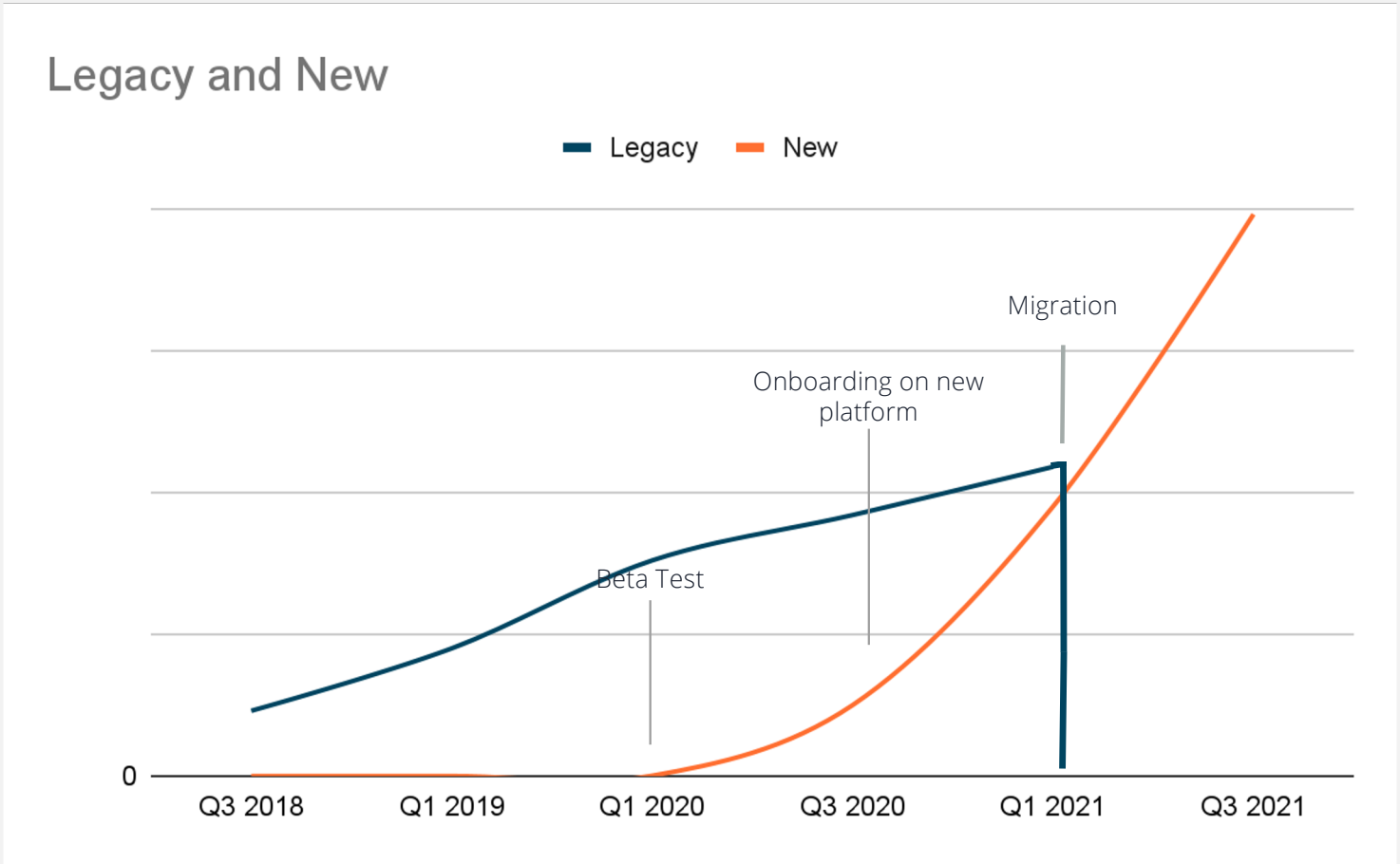
At this point, the legacy system is being written for rollback purposes only



Solarisbank

No such thing as Big Bang

Amount of new bank account created each quarter



What we take with us moving forward

Event sourcing is a big upfront investment

Understand your domains boundaries early

If availability is your goal, embrace eventual consistency

Use managed services

Thanks


goto;
GOTO
Guide
Remember to
rate this session
THANK YOU!
Download on the
App Store
GET IT ON
Google Play
#GOTOams