Industrial Experiences with Runtime Verification of Financial Transaction Systems:
Lessons Learnt and Standing Challenges

Christian Colombo

Joint work with Gordon Pace

L-Università ta' Malta

Marrying Runtime Verification with the Financial Transaction Industry

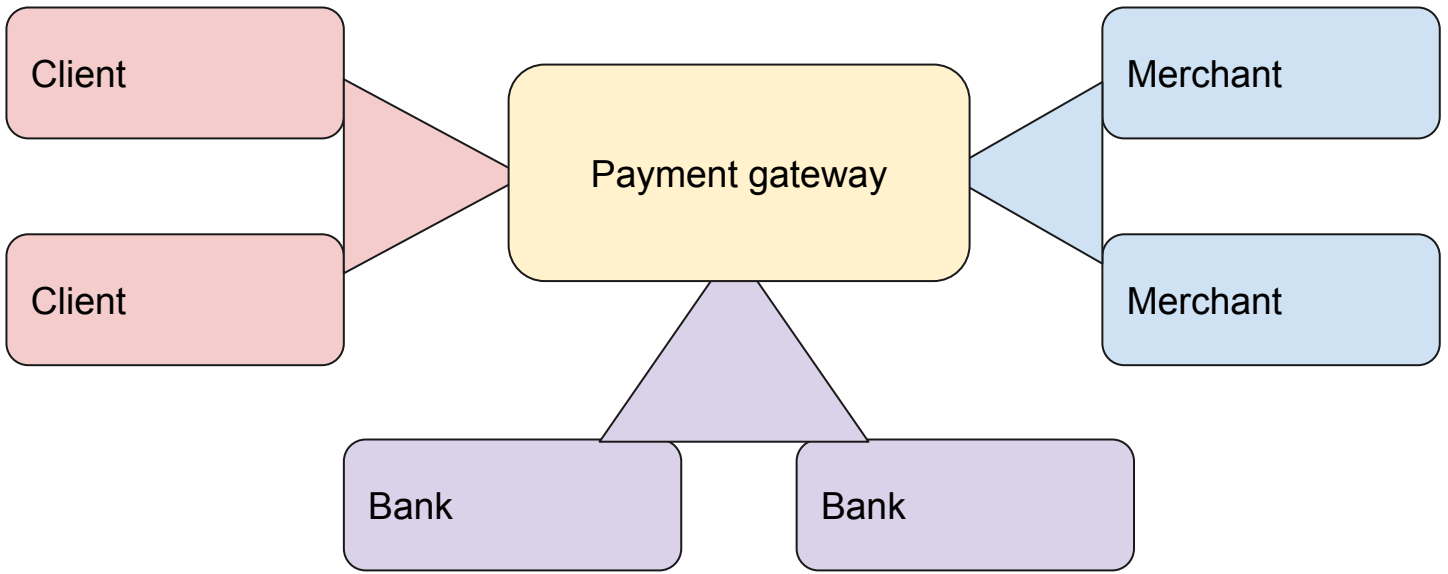
Christian Colombo

Joint work with Gordon Pace

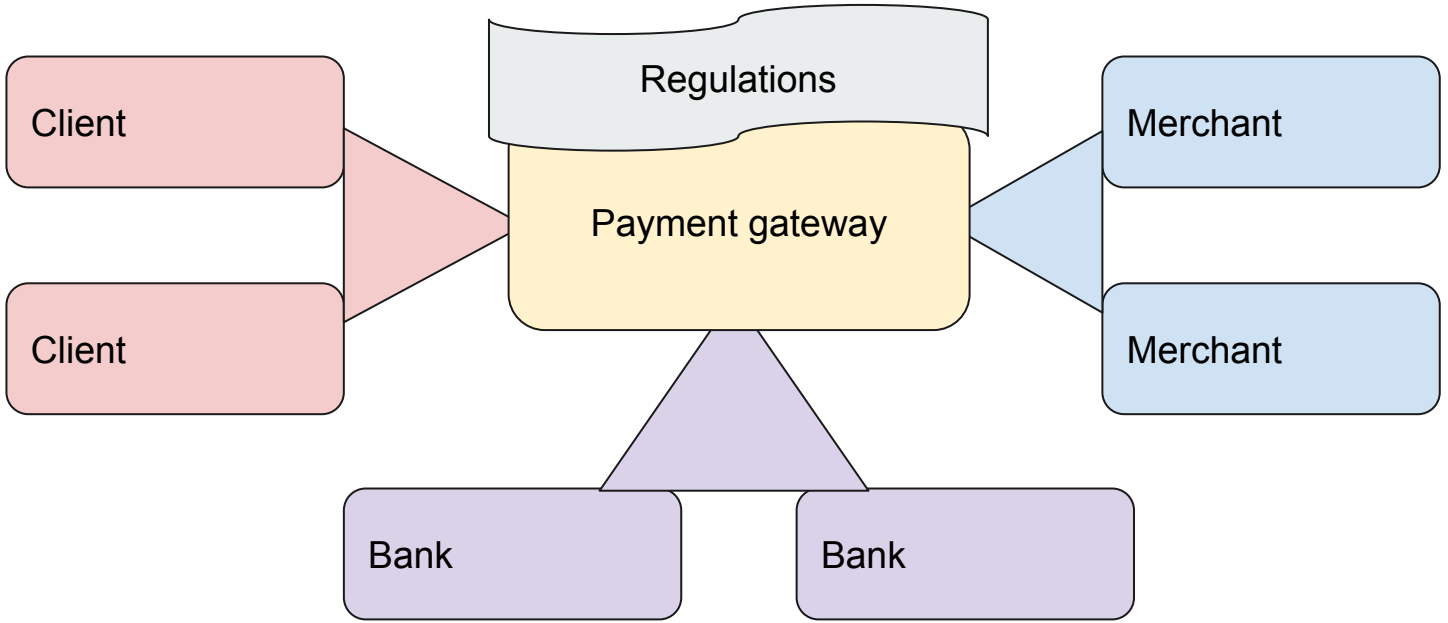
L-Università ta' Malta

Financial transaction systems

Taking a closer look



Taking a closer look



What kind of regulations?

E-money funds can be redeemed at par value and without delay

No interest can be given unless the institution is a bank

An unregistered user cannot spend more than €xxx/€xxx a week/month.

Marrying RV and the Financial Industry

How does Fintech see RV?

How does RV see Fintech?

Why is RV different for the industry?

Unlike model checking and testing, RV works live!

Why is RV different for the industry?

Unlike model checking and testing, RV works live!

They need to **trust** the RV code!

- Will it scale up?
- Is it safe?
- Is it correct?

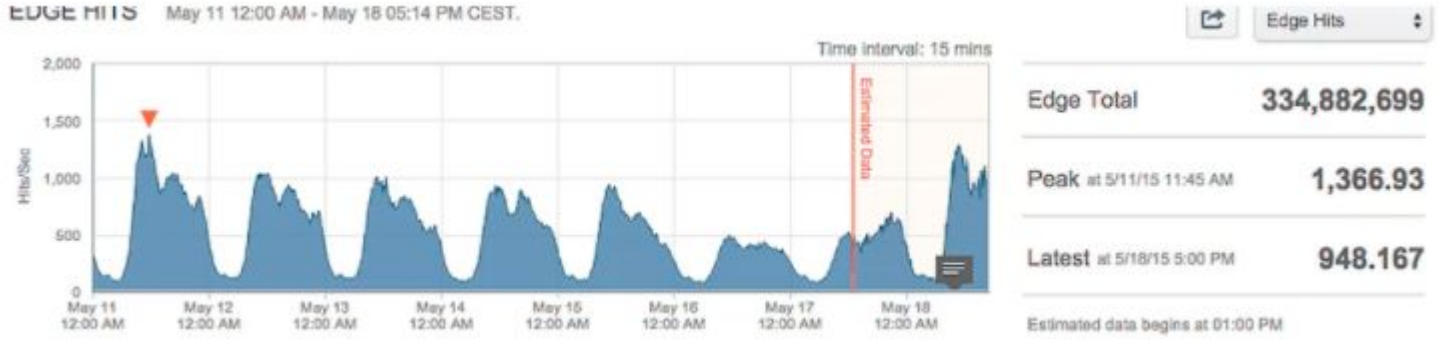
The attraction

Compliance is a huge headache for Fintech

Their testing is not enough

How is Fintech challenging for RV?

Spikes in traffic



How is Fintech challenging for RV?

Spikes in traffic

Reliability requirements

How is Fintech challenging for RV?

Spikes in traffic

Reliability requirements

The human side!!

What next?

Challenges and options:

- The process (human) side of things
- The engineering challenges

Our experiences

- Our decisions
- Lessons learnt
- A success story

Process (human) side of things

The human side of it

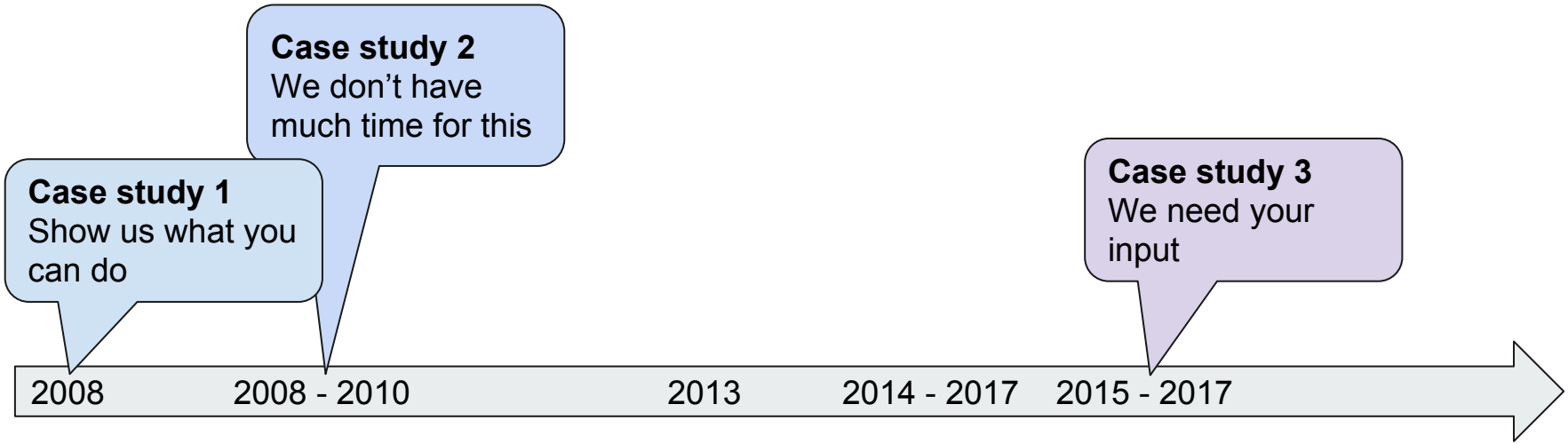
Communicating the idea

Acceptance of the project

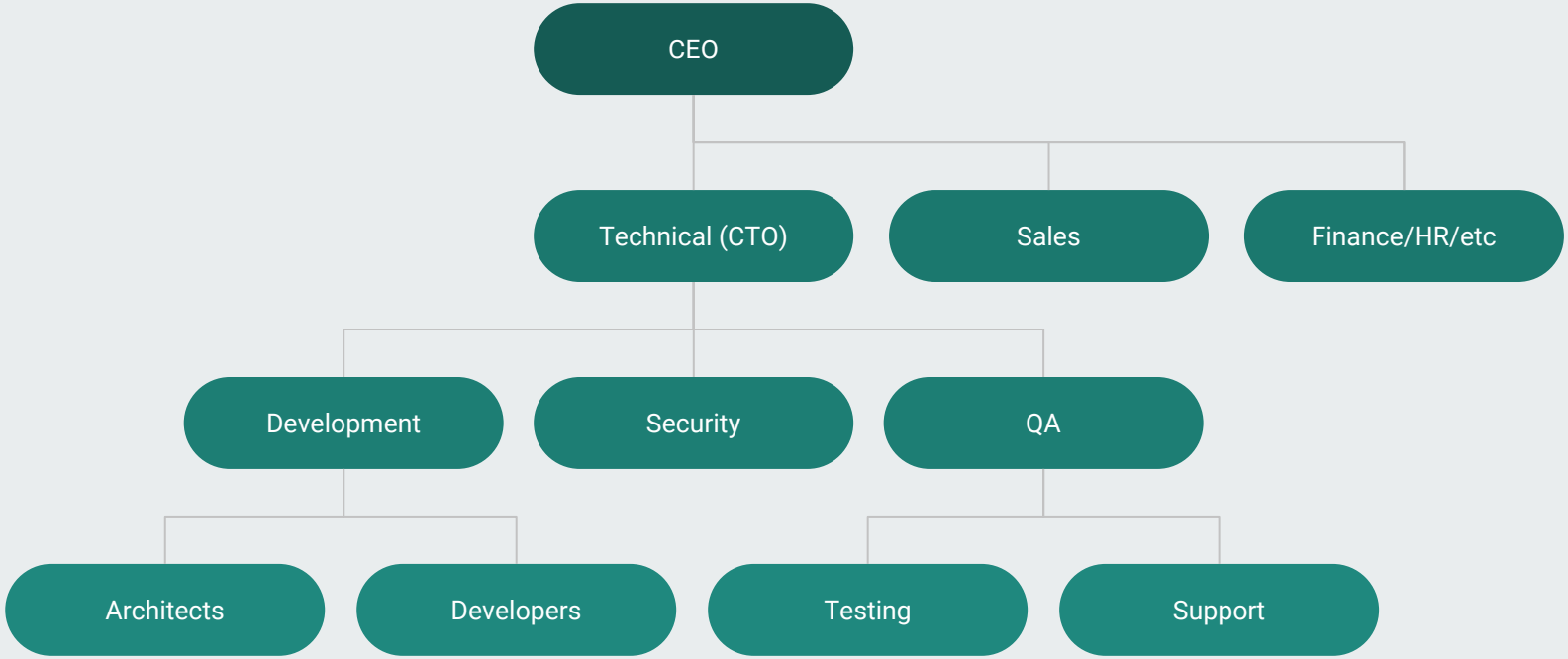
Identifying properties

Who does what when

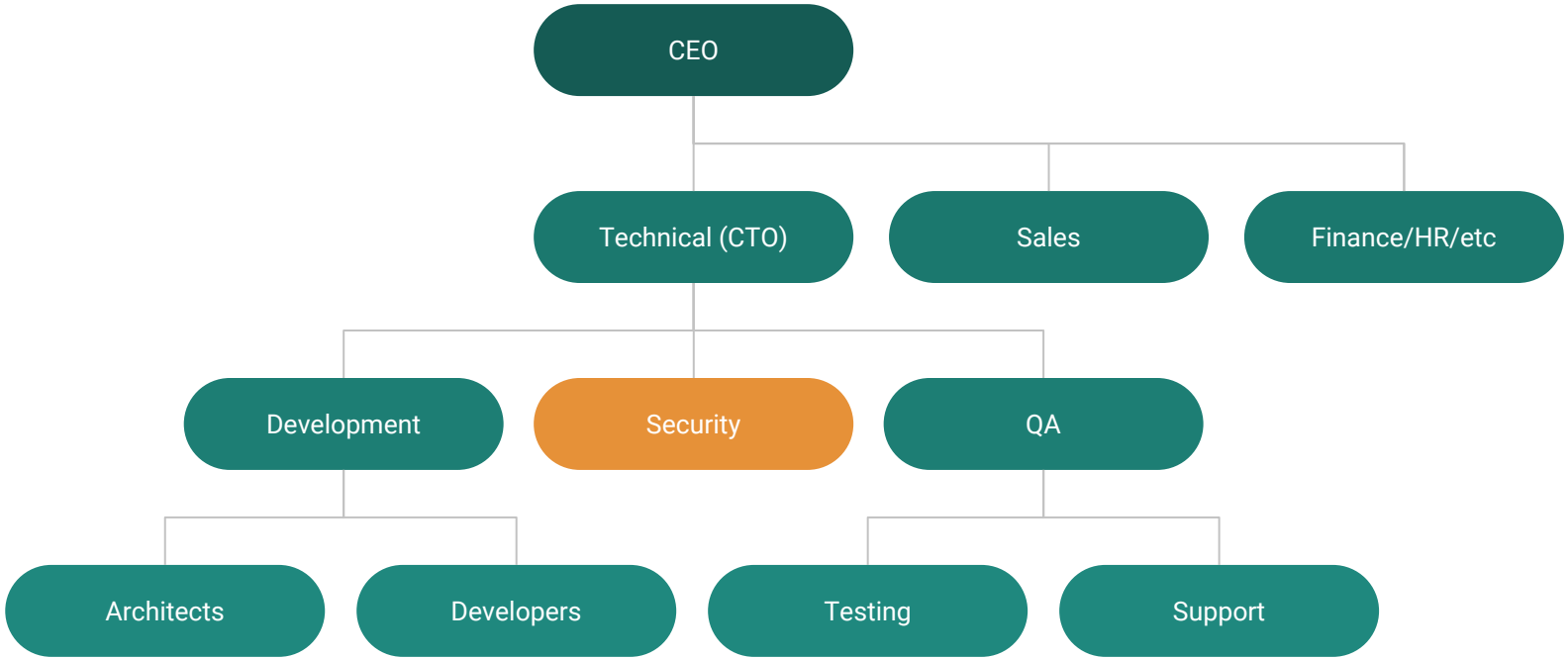
Case studies



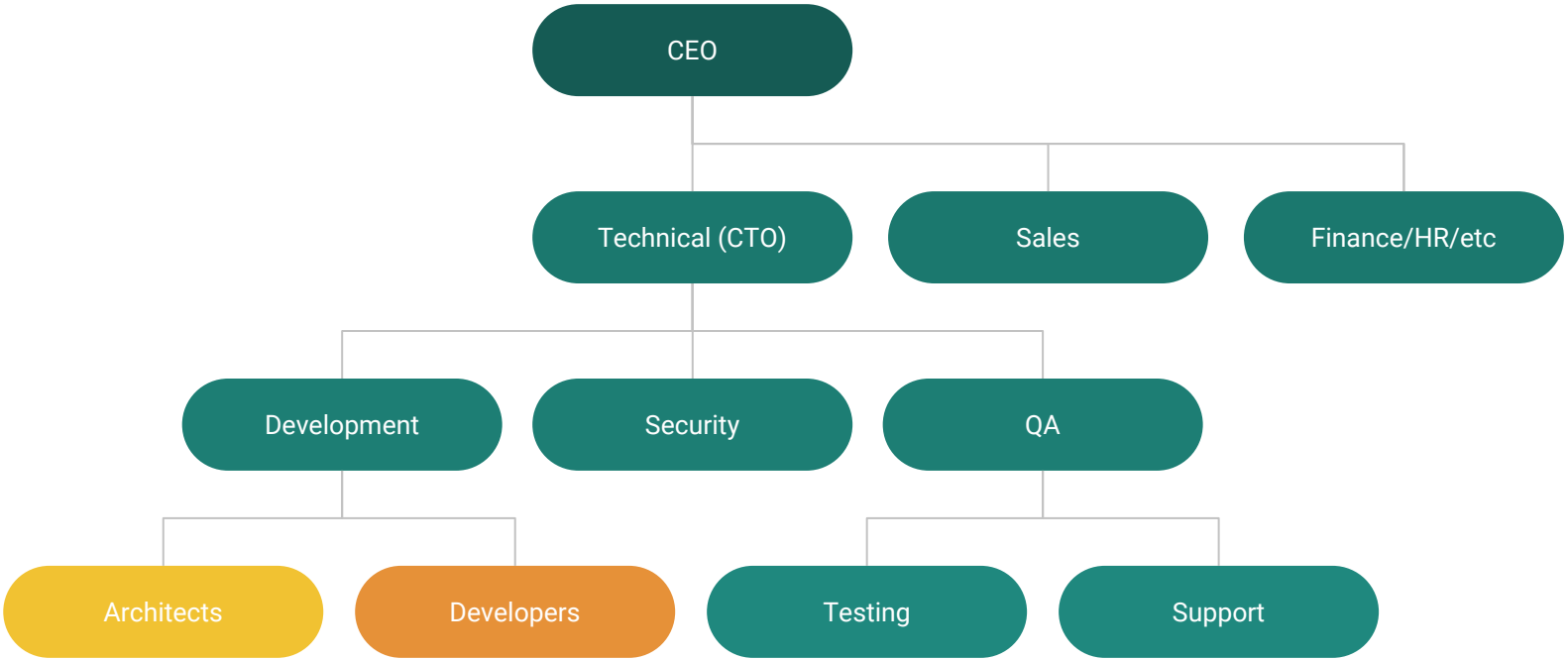
Exercise - Process (CS1)



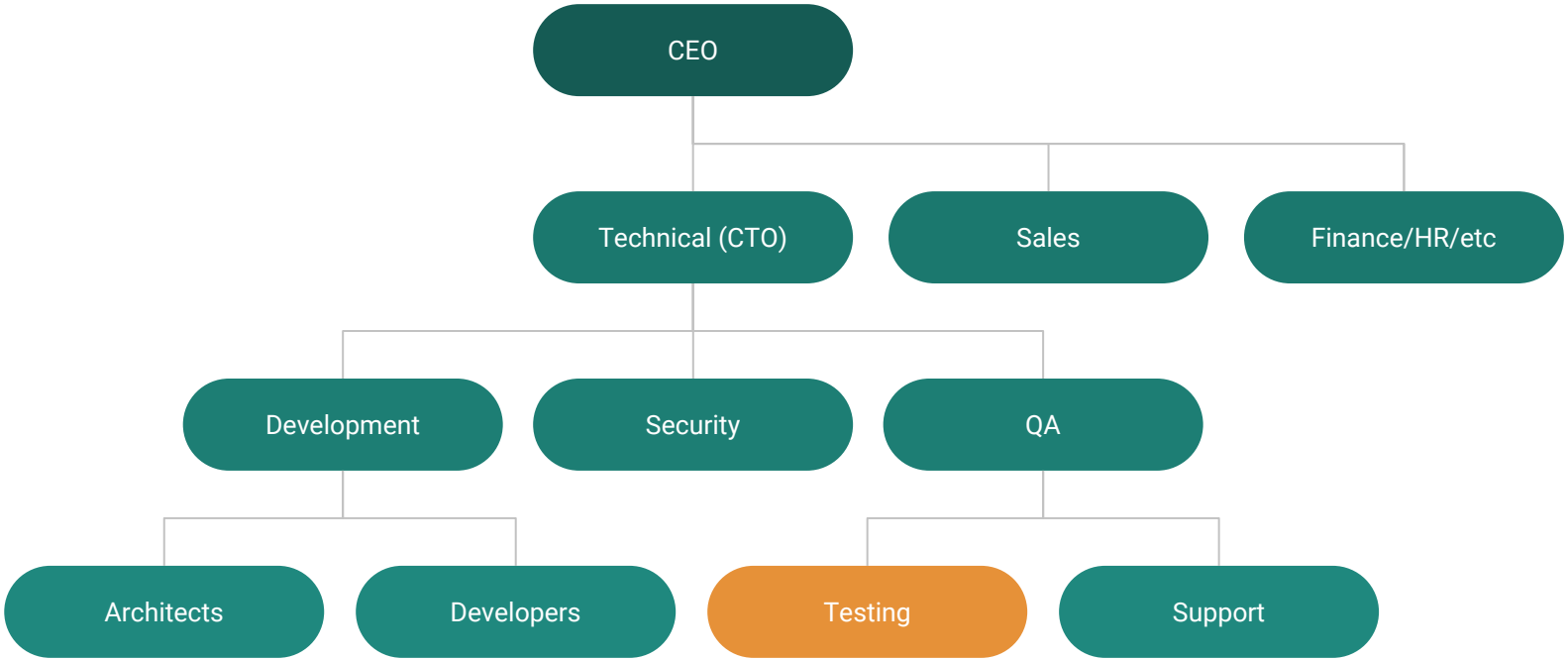
Case study 1


CEO
Technical (CTO)
Sales
Finance/HR/etc
Development
Security
QA
Architects
Developers
Testing
Support

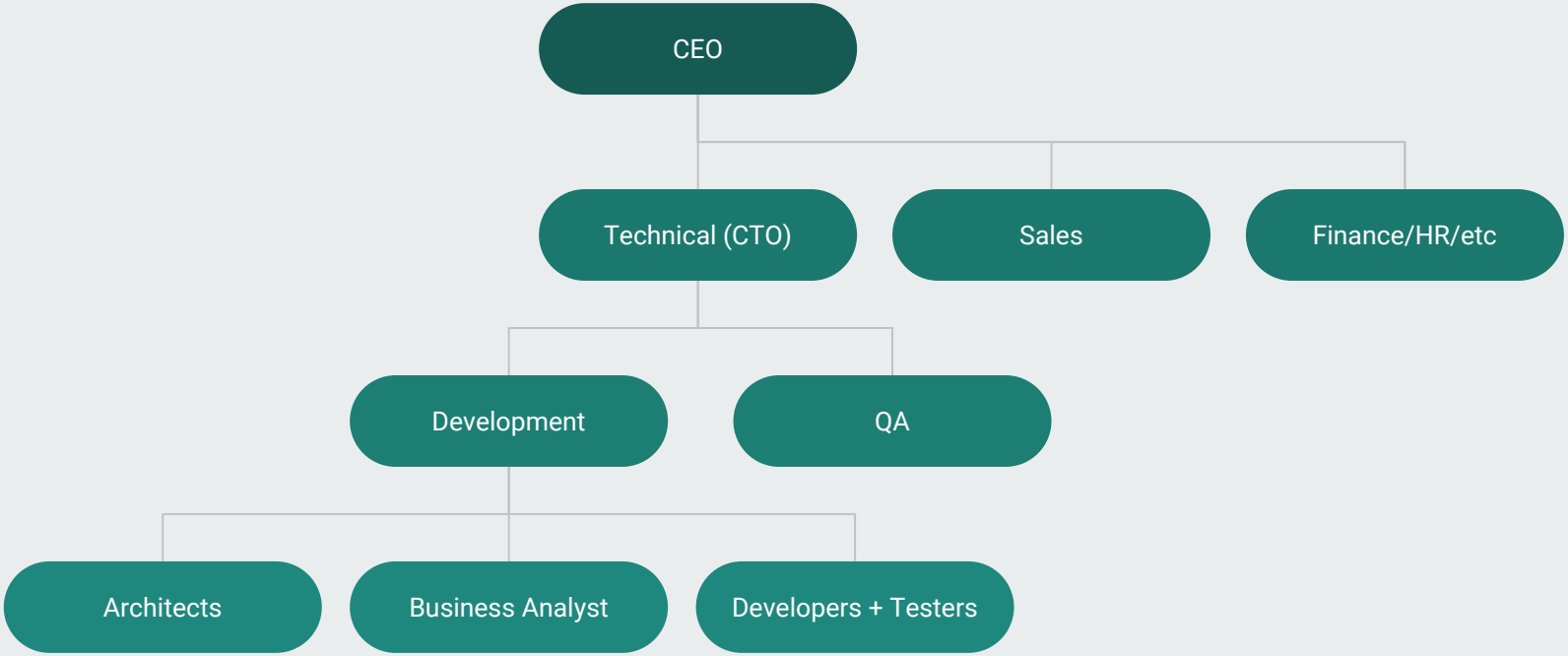
Case study 1


CEO
Technical (CTO)
Sales
Finance/HR/etc
Development
Security
QA
Architects
Developers
Testing
Support

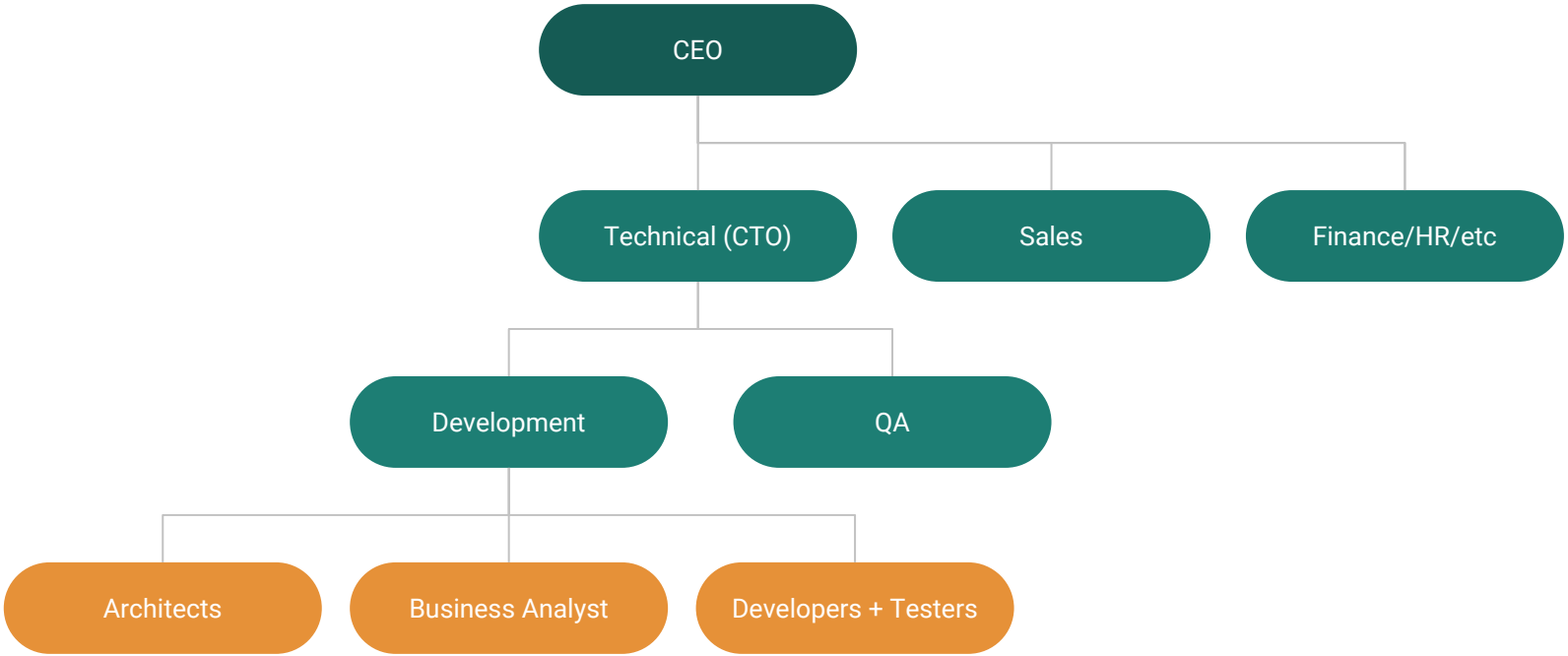
Case study 1


CEO
Technical (CTO)
Sales
Finance/HR/etc
Development
Security
QA
Architects
Developers
Testing
Support

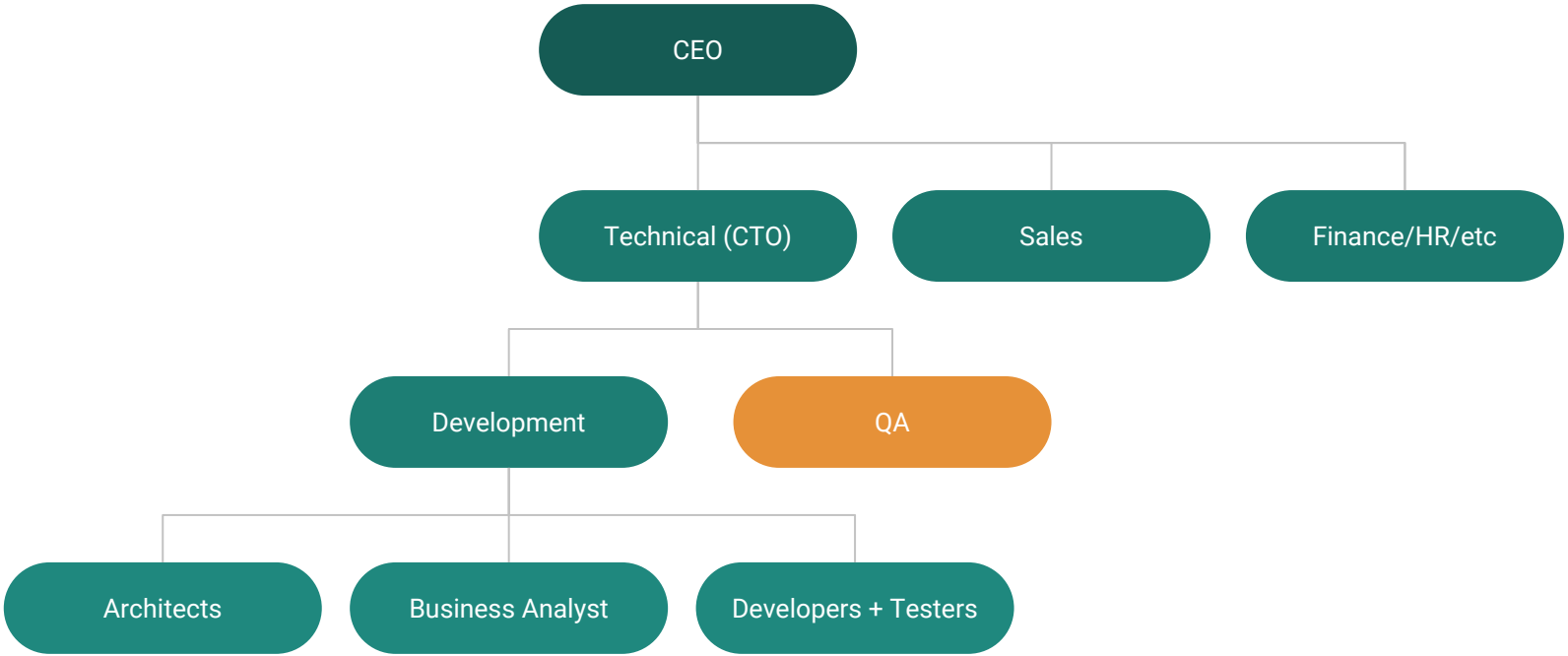
Exercise - Process (CS2)



Case study 2


CEO
Technical (CTO)
Sales
Finance/HR/etc
Development
QA
Architects
Business Analyst
Developers + Testers

Case study 2

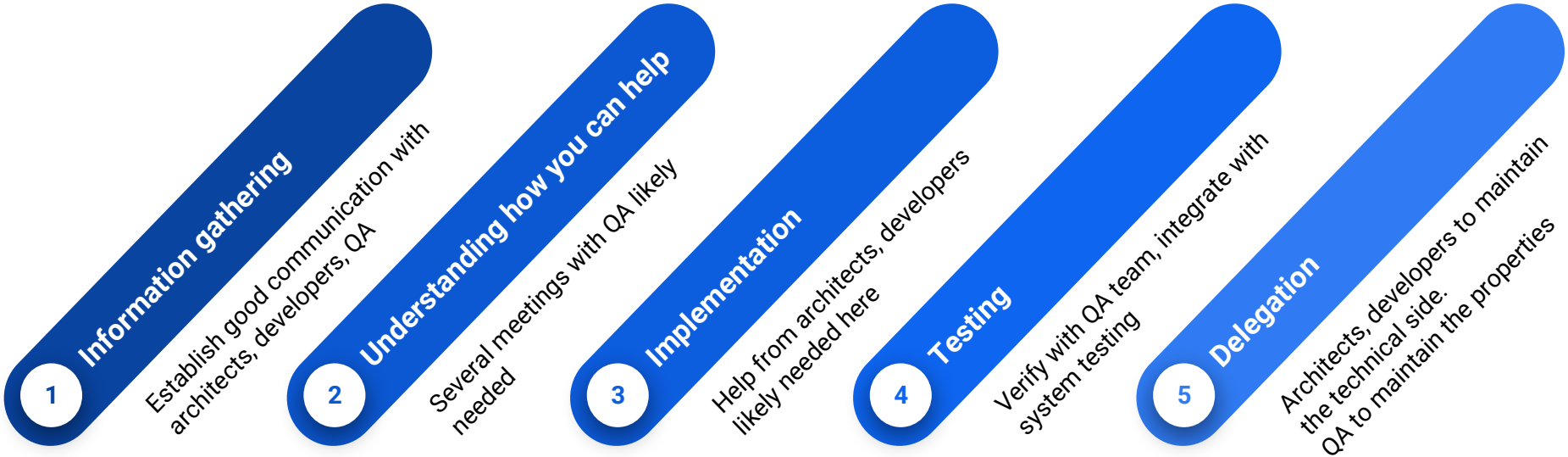

CEO
Technical (CTO)
Sales
Finance/HR/etc
Development
QA
Architects
Business Analyst
Developers + Testers

Lessons Learnt

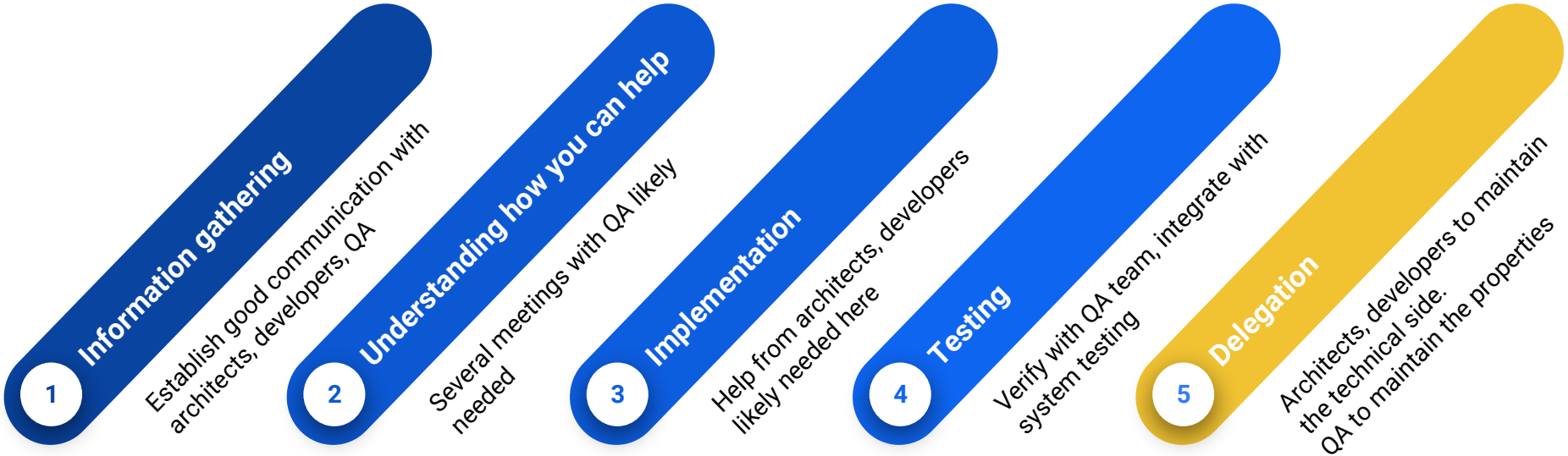
What benefit will these guys get?

Do these guys have the right level of abstraction of the system?

Summary for academia-led projects



Summary for academia-led projects



Who will maintain the monitors?

Those who benefit

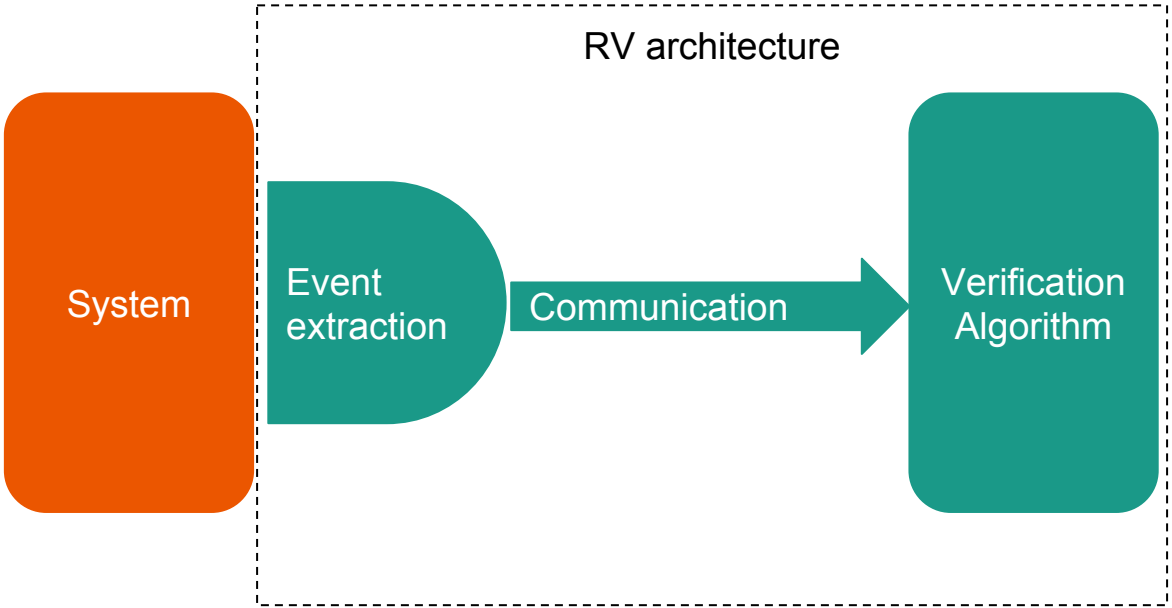
Who will maintain the monitors?

What if those who benefit are not technical?

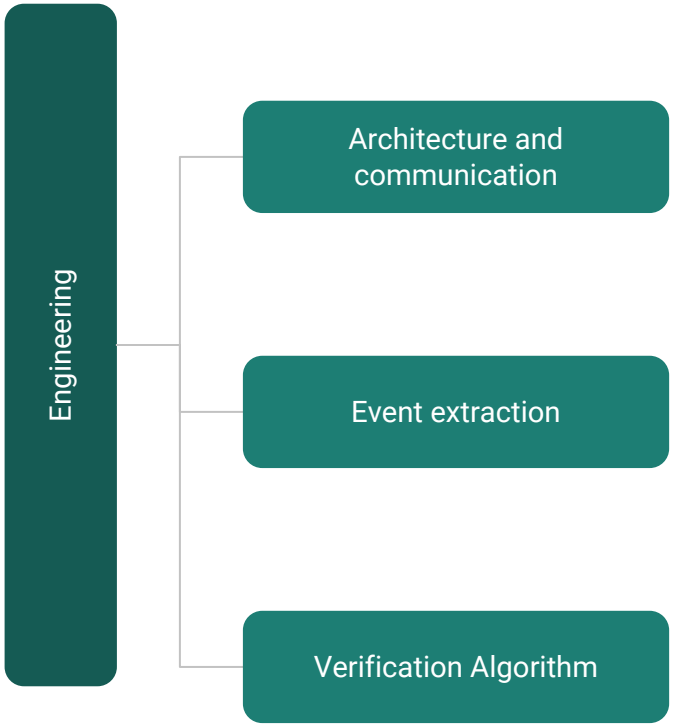
- Controlled natural language

Engineering side of things

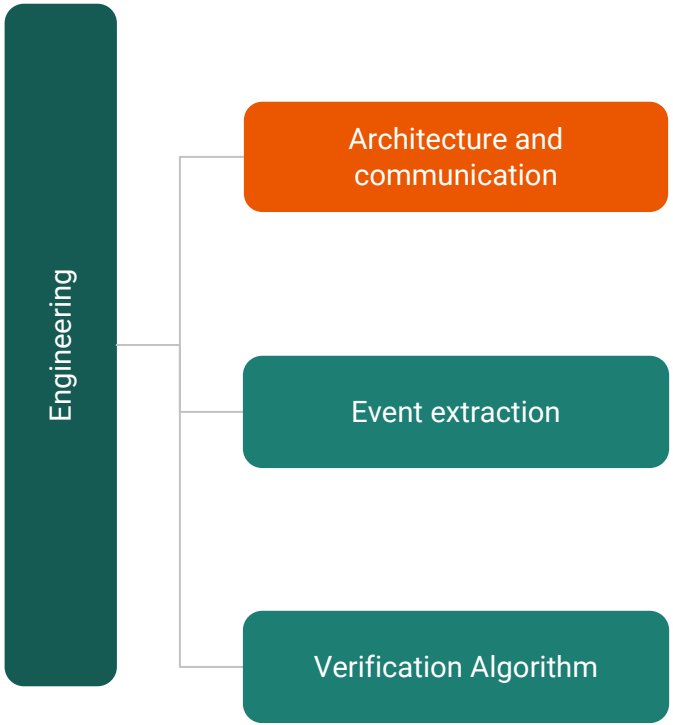
Engineering



Engineering design options



Engineering design options



System-Monitor Organisation

System Execution
Monitor Execution

Online

Synchronous

Asynchronous

Offline

System-Monitor Organisation

System Execution
Monitor Execution

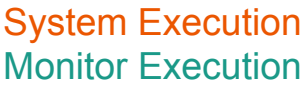
Online

Synchronous

Asynchronous

Offline

System-Monitor Organisation



Online

Synchronous

Asynchronous

Offline

Communication

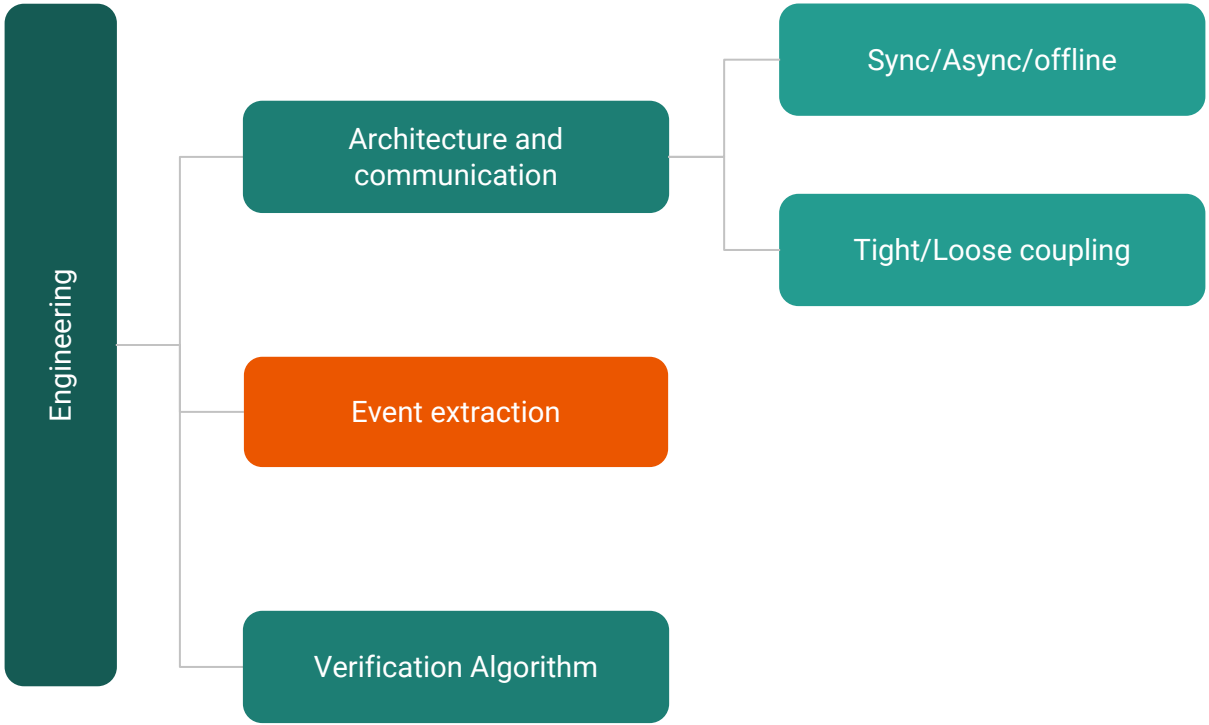
- Method call
- Communication protocol

  TCP/IP

  Jason
- Database logs

Tight coupling

Loose coupling


Engineering
Architecture and communication
Sync/Async/offline
Tight/Loose coupling
Event extraction
Verification Algorithm

Event extraction

Method call interception (aspect-oriented programming)

Event extraction

Method call interception (aspect-oriented programming)

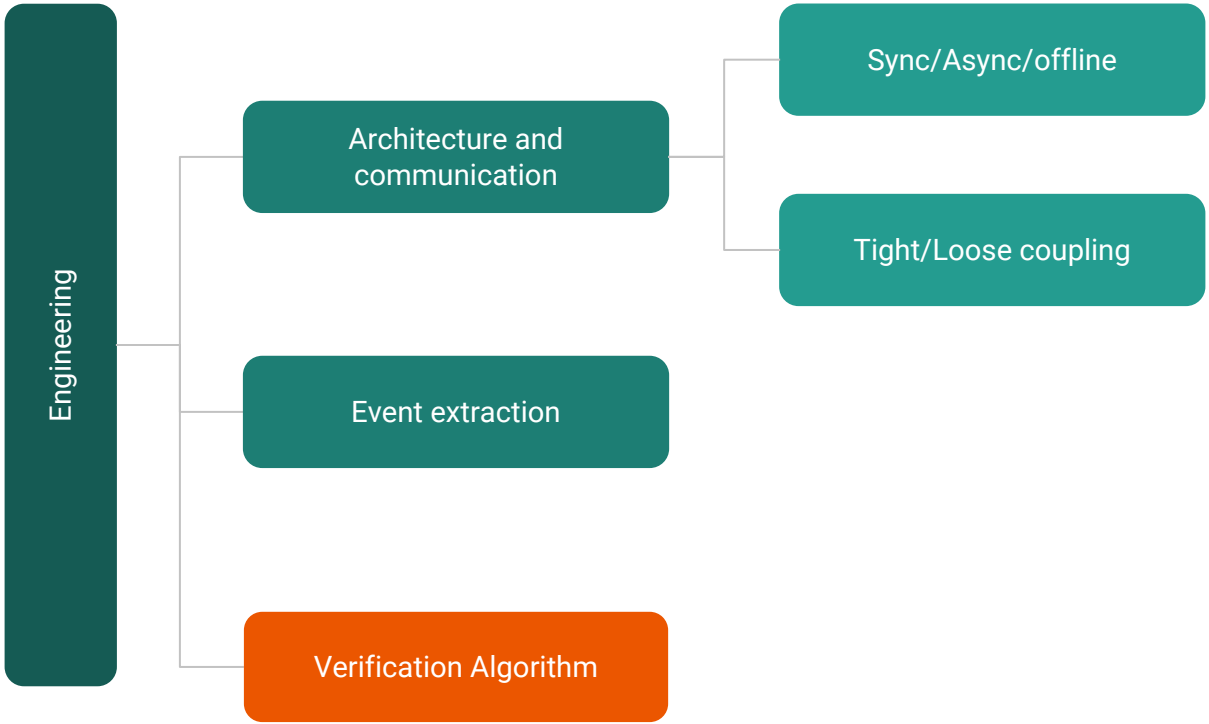
Communication interception (monitor acts as a proxy or sniffer)

Event extraction

Method call interception (aspect-oriented programming)

Communication interception (monitor acts as a proxy or sniffer)

Events by design


Engineering
Architecture and communication
Sync/Async/offline
Tight/Loose coupling
Event extraction
Verification Algorithm

Implicit vs explicit state space

Monitoring logics using rewriting rules

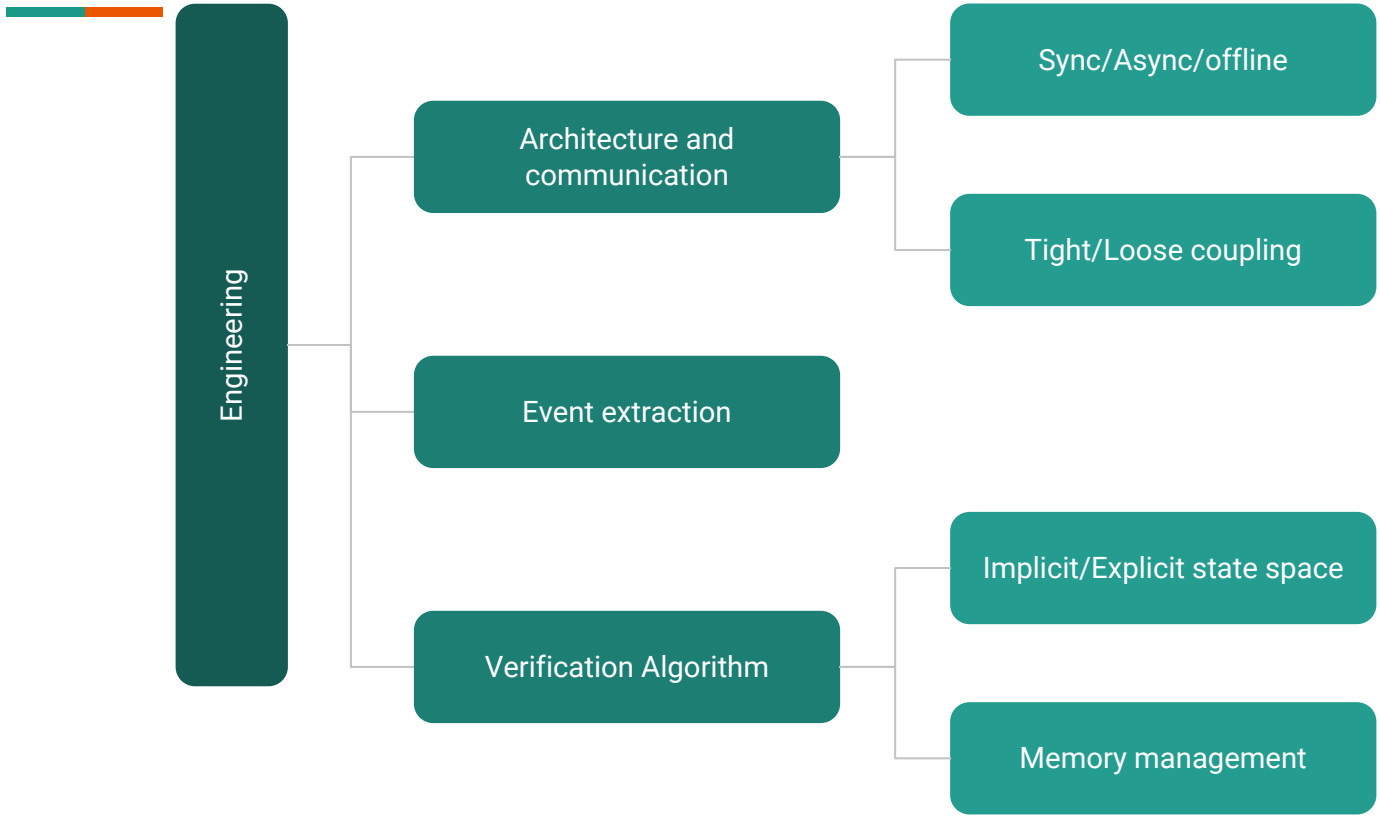
vs

Traversing automata

Memory management

Garbage collection

Setting a maximum memory size per monitored entity

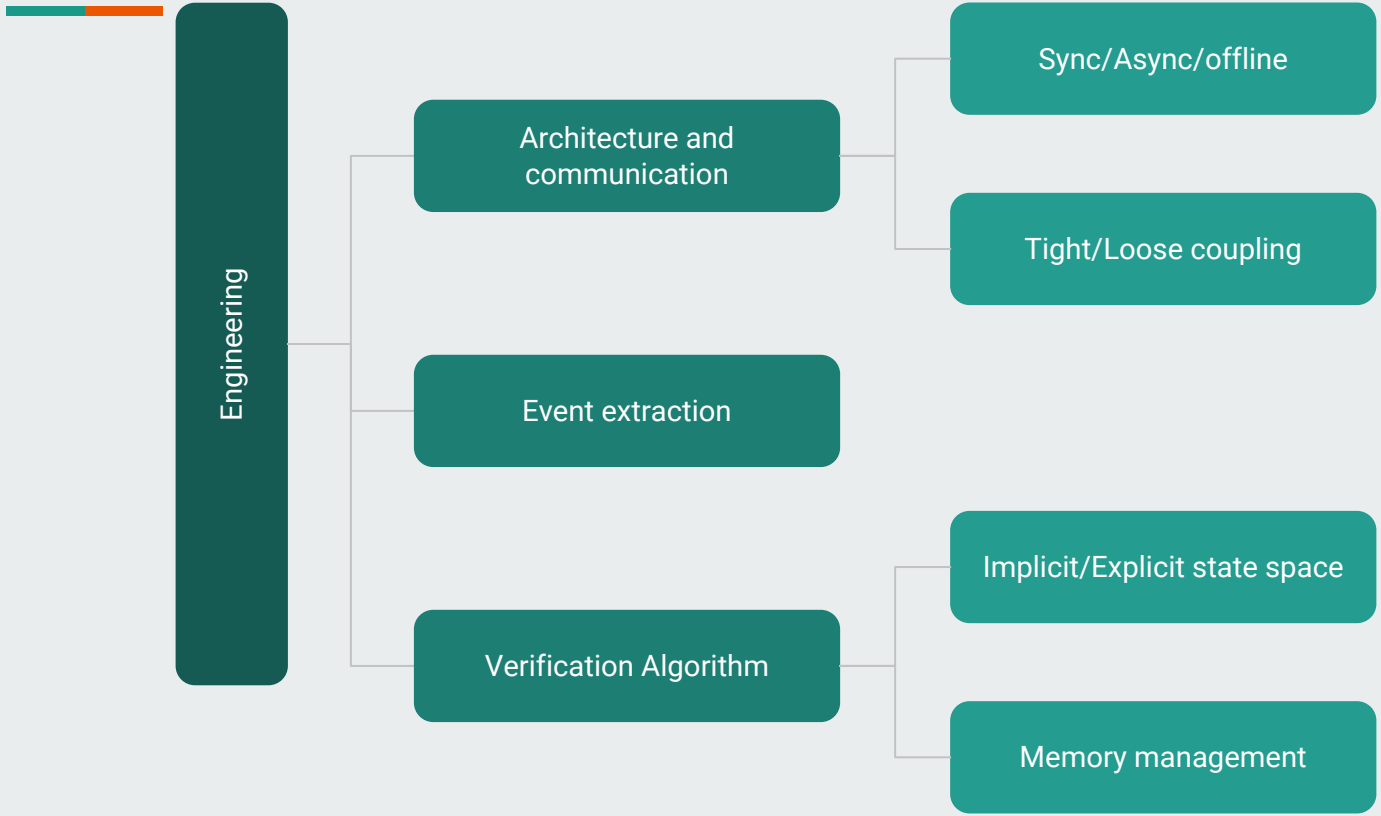

Engineering
Architecture and communication
Sync/Async/offline
Tight/Loose coupling
Event extraction
Verification Algorithm
Implicit/Explicit state space
Memory management

Exercise - Engineering (CS2)

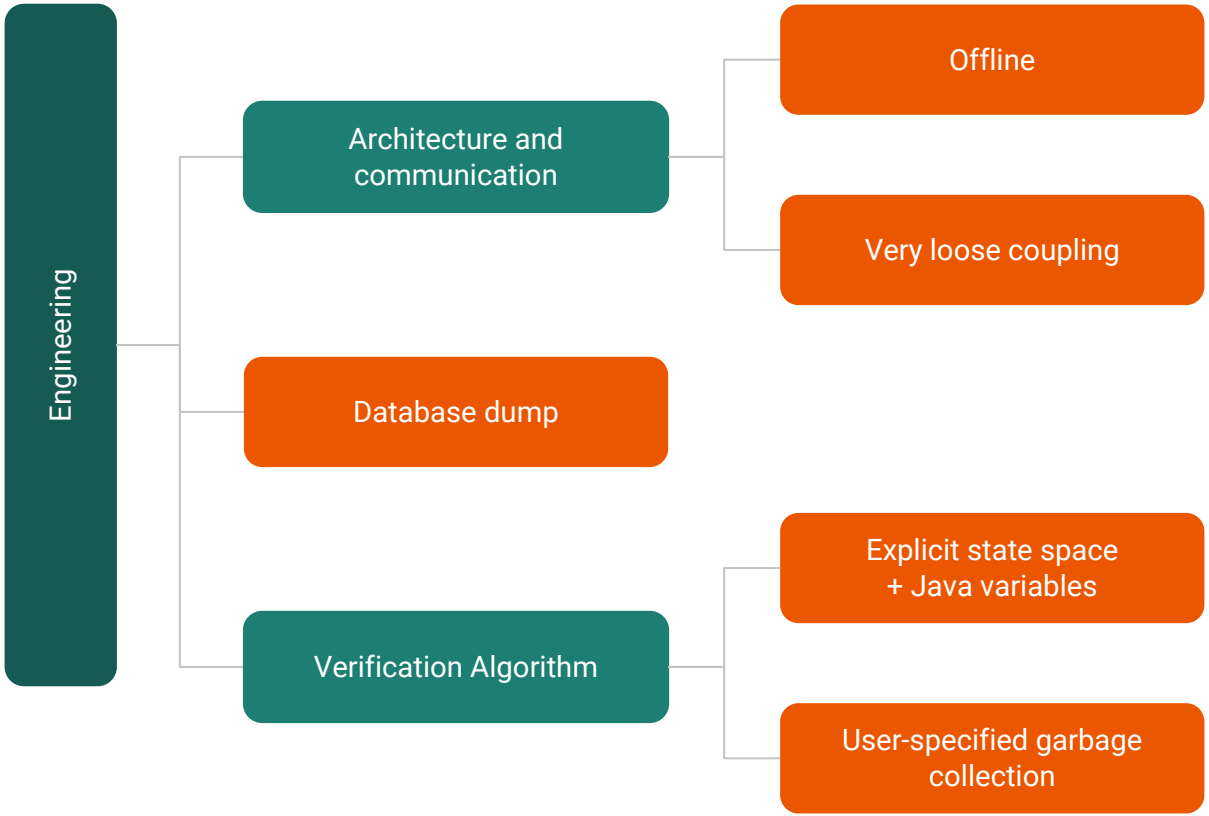
Description of situation (Case Study 2):

- Java system with mySQL database
- Efficiency concerns
- Trust issues

What would you choose in each case?



Case study 2

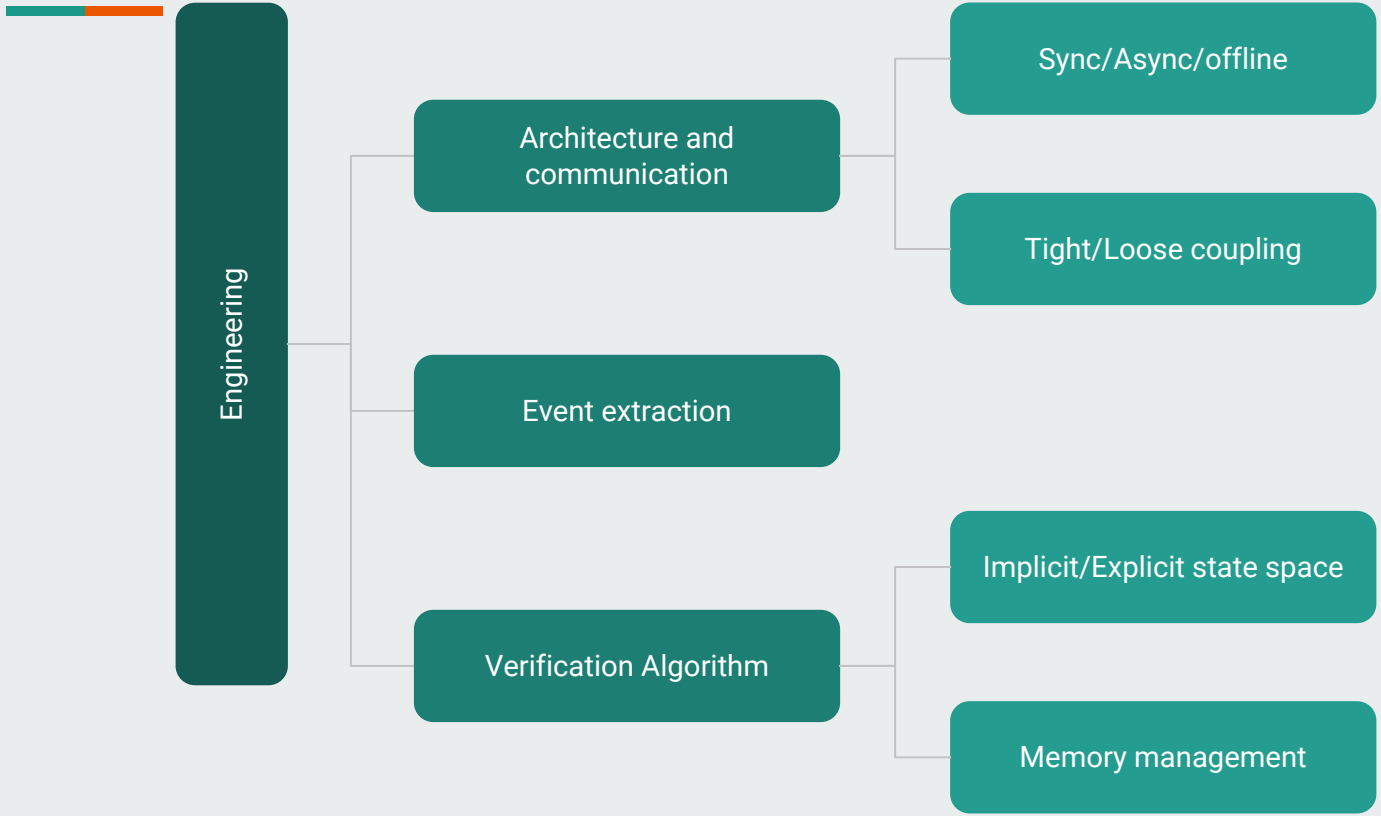

Engineering
Architecture and communication
Offline
Very loose coupling
Database dump
Verification Algorithm
Explicit state space + Java variables
User-specified garbage collection

Exercise - Engineering (CS3)

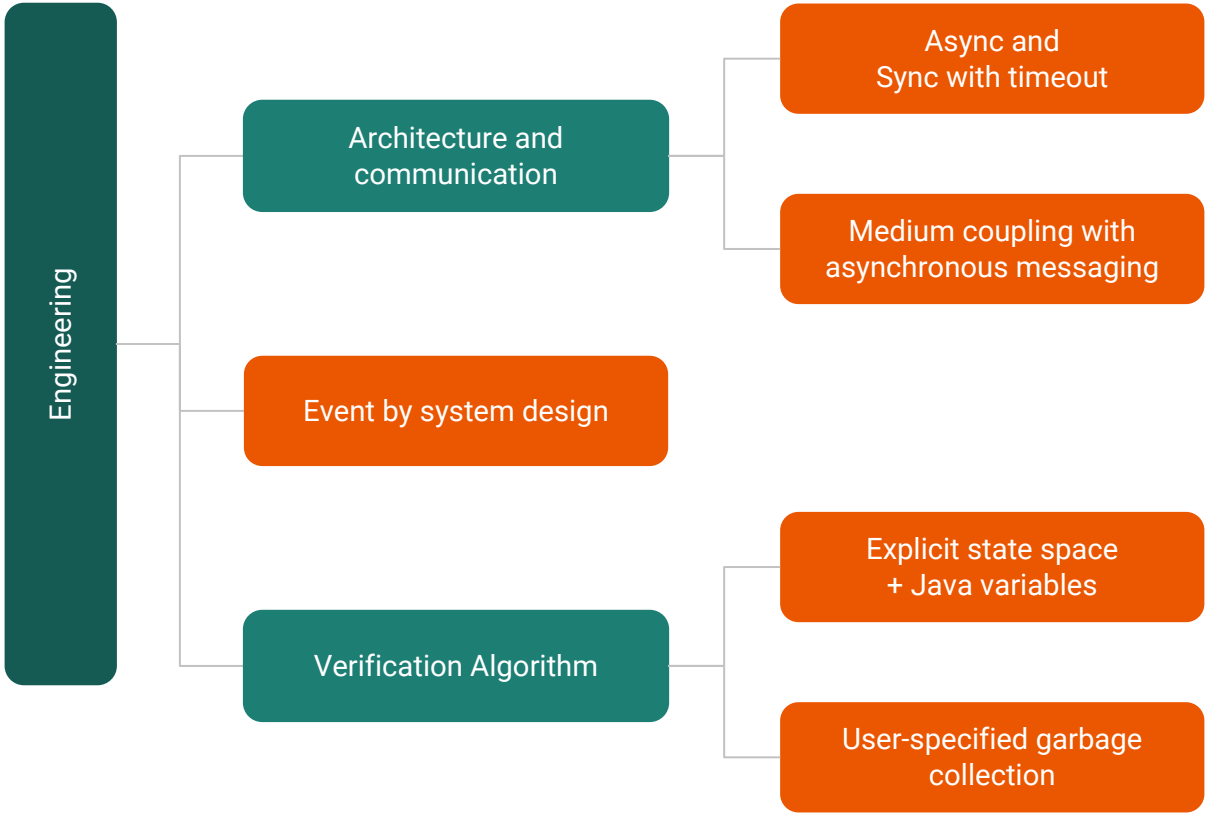
Description of situation (Case Study 3):

- Microservices built using Akka actors
- Efficiency concerns
- System design with RV in mind

What would you choose in each case?



Case study 3


Engineering
Architecture and communication
Async and Sync with timeout
Medium coupling with asynchronous messaging
Event by system design
Verification Algorithm
Explicit state space + Java variables
User-specified garbage collection

Evaluation and lessons learnt

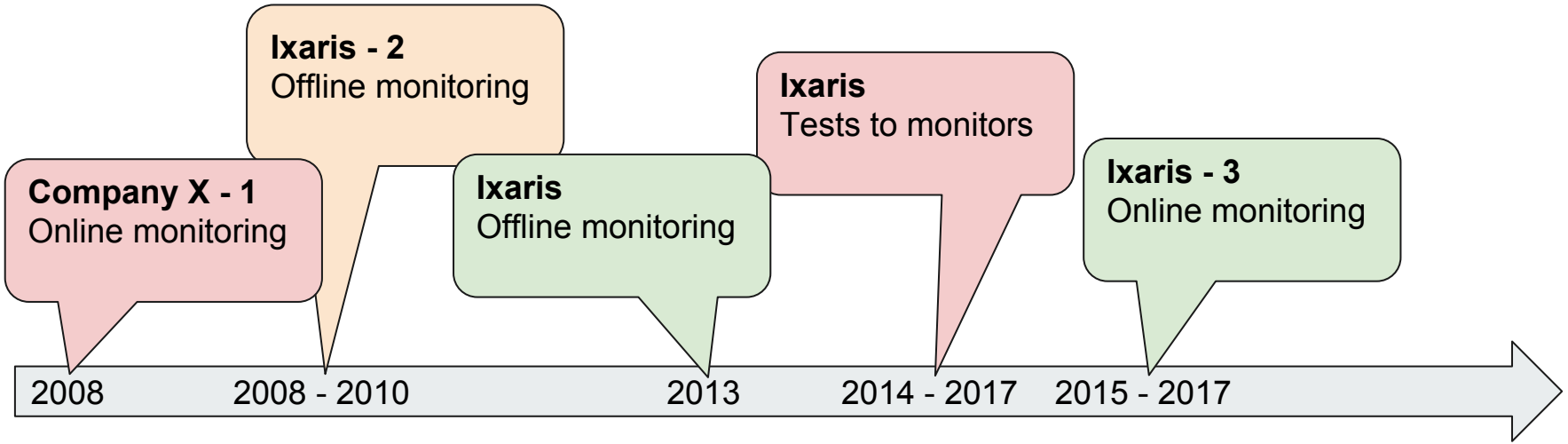
Defining success

Usefulness to industry?

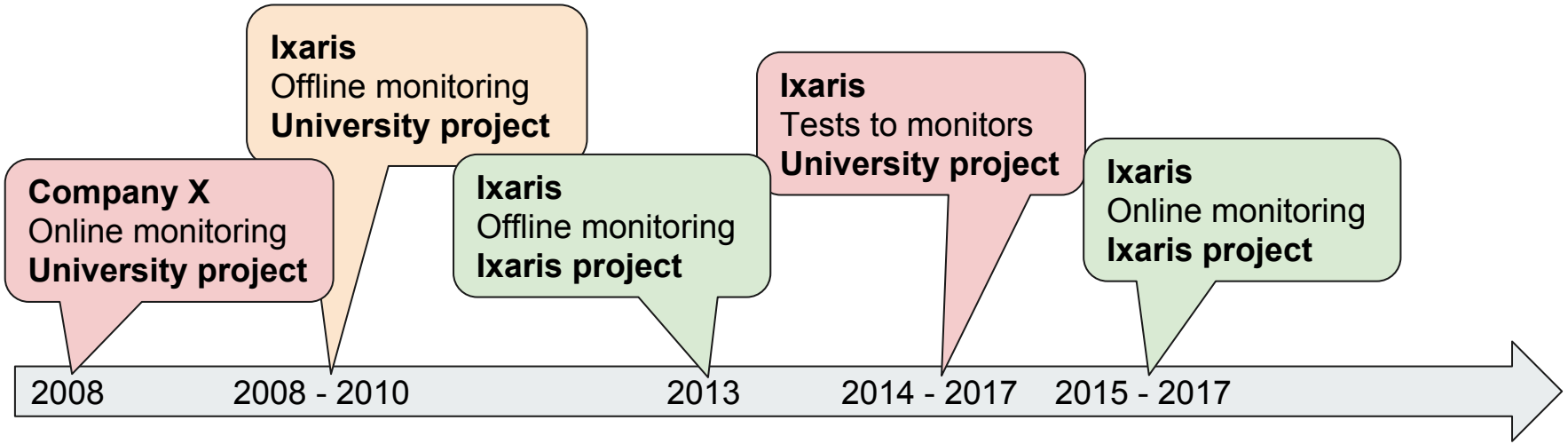
Number of publications?

Create a startup?

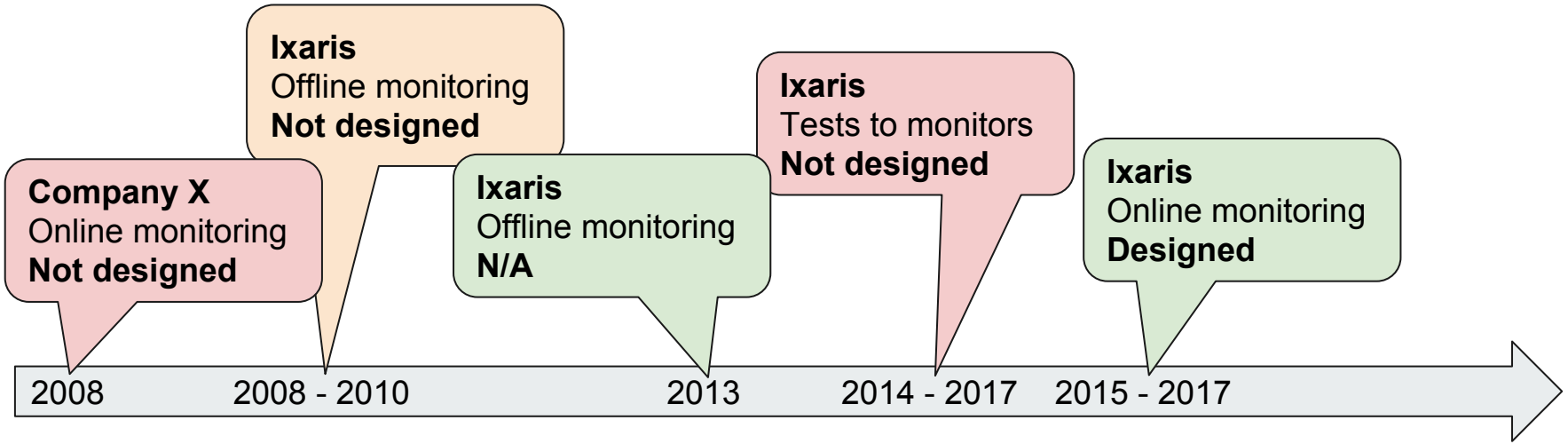
Success - actual use in industry



Lesson Learnt 1: University- vs Industry-led



Lesson Learnt 2: RV from design (or not)



Architecture

	Industry	Academia
RV in design	Sync, async	Case-by-case
No RV in design	Case-by-case	Offline

Communication

		Sync	Offline
	RV in design	Tight coupling possible	Loose coupling
	No RV in design	Case-by-case	Loose coupling

Event extraction

	Sync	Offline
RV in design	Custom events	Lots of options
No RV in design	AOP, Proxy	Database

Implicit/explicit state space

	(Logic compilable to) explicit	implicit
Non-strict memory constraints	Both ok	implicit
Strict memory constraints	explicit	No ideal solution

Garbage collection

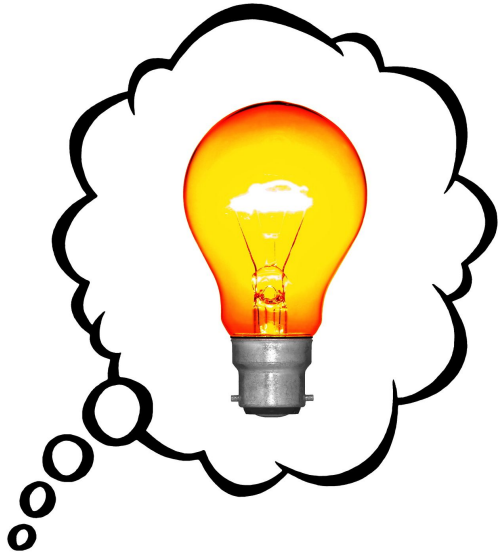
		(Property writer)	
		technical	Non-technical
	Non-strict memory constraints	Both ok	Automatic
	Strict memory constraints	User-managed	Open problem

Case study 3 - A success story

The Open Payments Ecosystem

One morning…

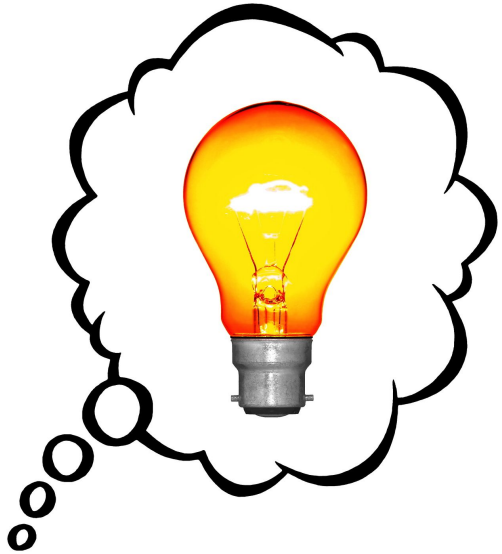
You wake up with a great idea…



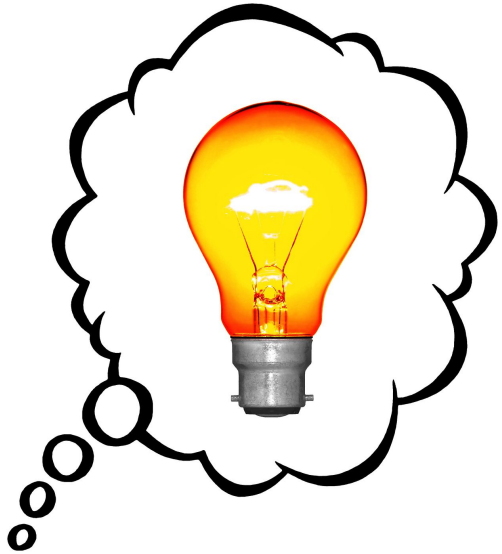
One morning…

You wake up with a great idea…

A mobile app which gives users complete visibility of their **money spending habits**



One morning…

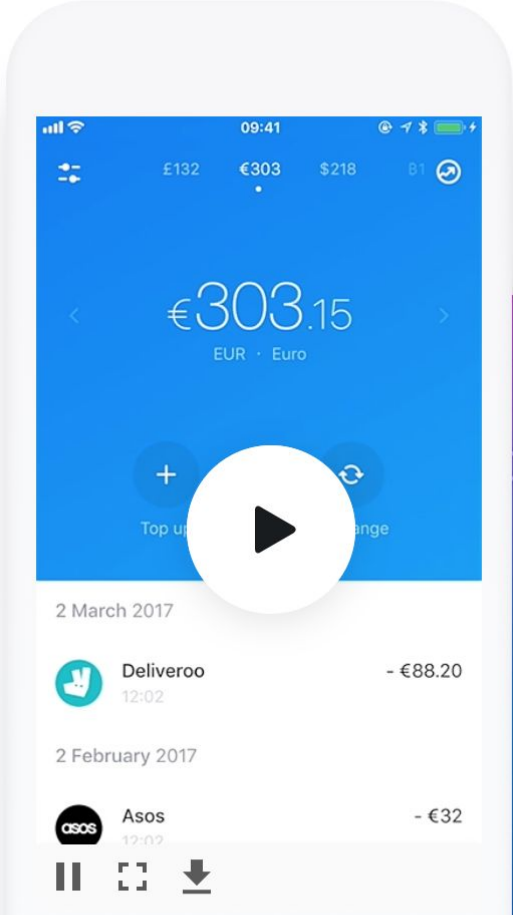


You wake up with a great idea…

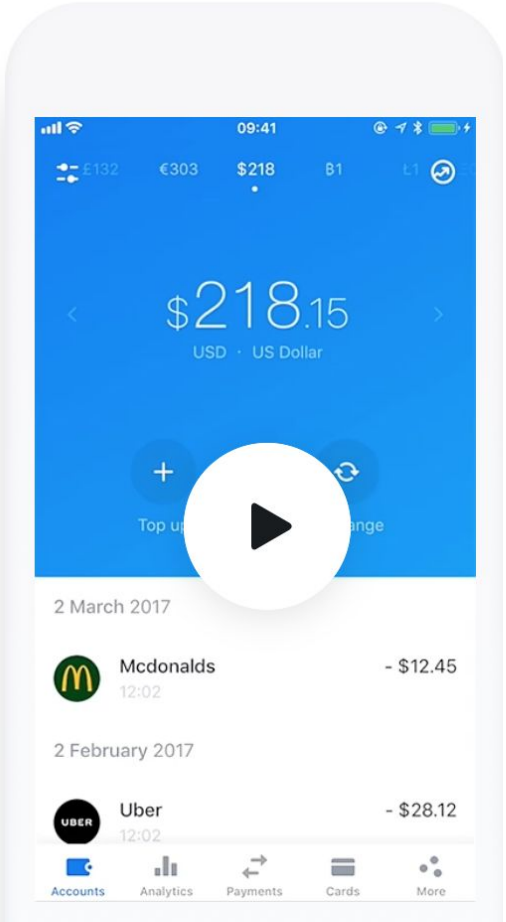
A mobile app which gives users complete visibility of their **money spending habits**

It allows them to **budget their salary**, save every month, ...

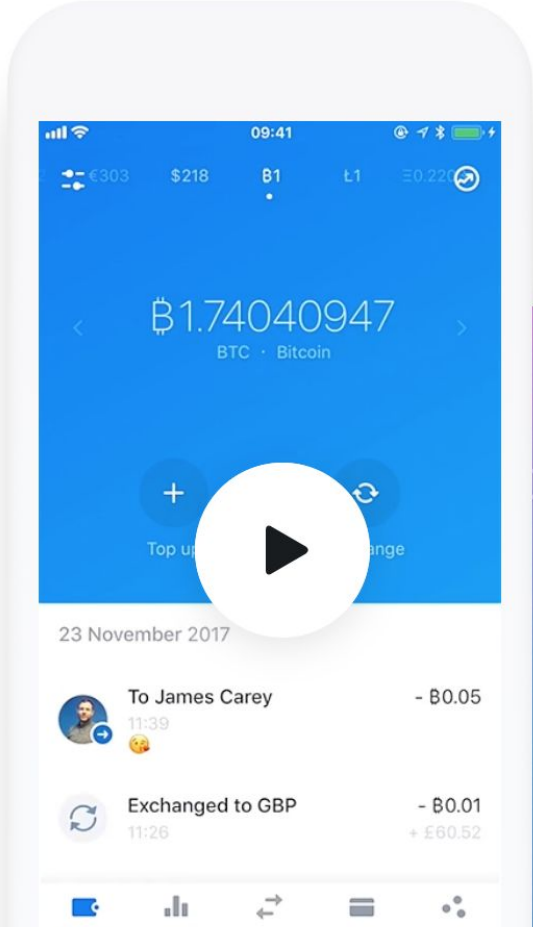
You start imagining how it would look...



You start imagining how it would look...



You start imagining how it would look...



There is just one problem...

You need a **bank license**!

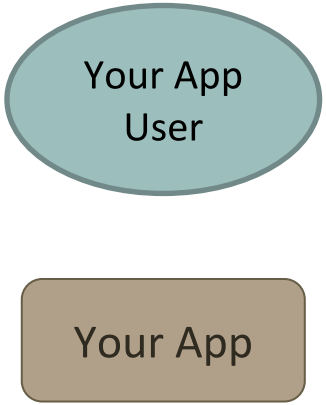
(Those were screenshots from Revolut btw)

Payment programme setup costs

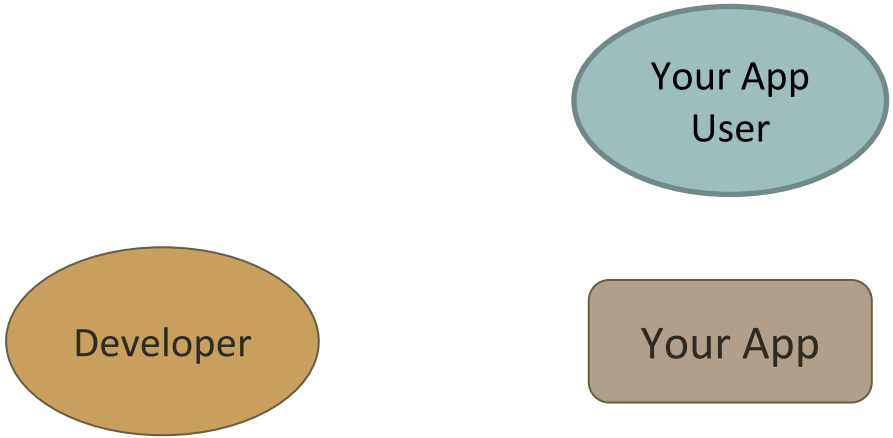
- Implementing card processes
- Agreement with bank
- Compliance to legislation
- Auditing
- Dispute resolution
- ...

Building a payment application

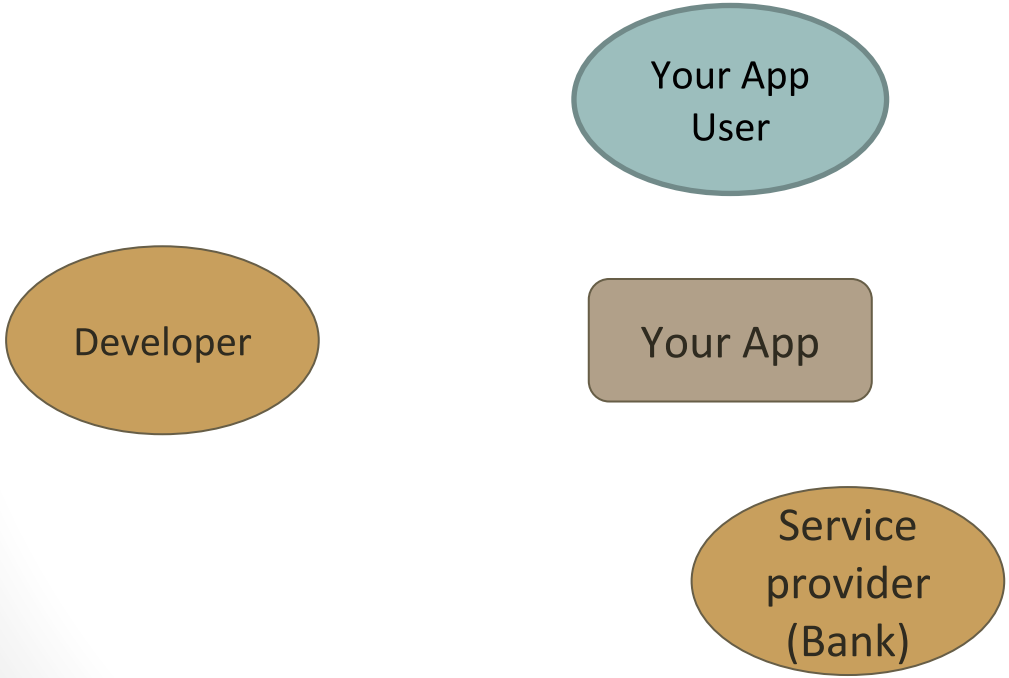
Building a payment application



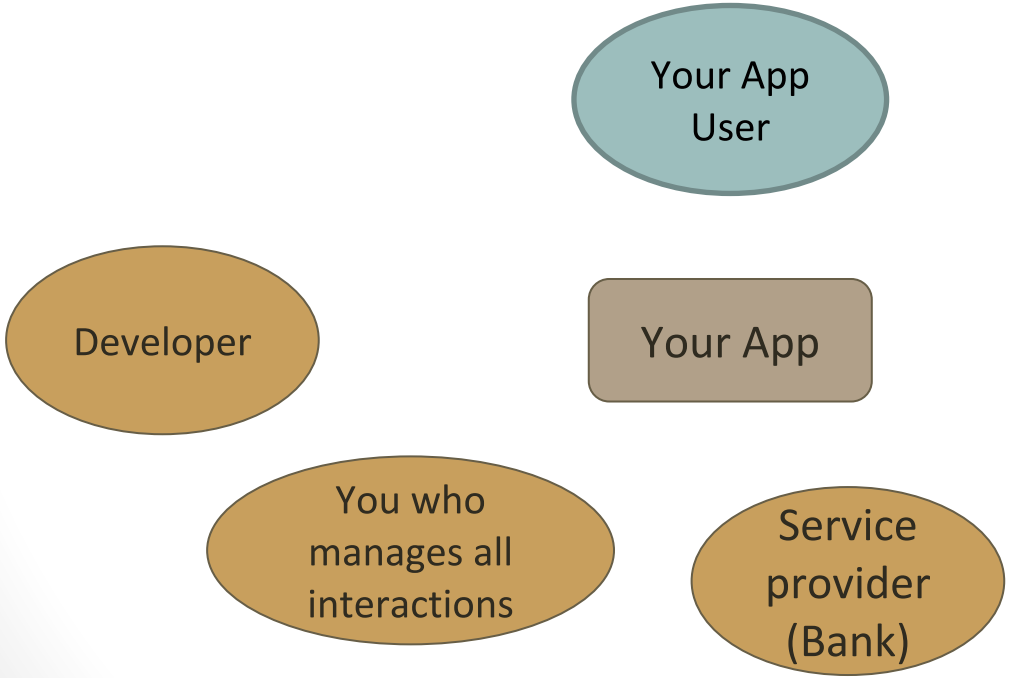
Building a payment application



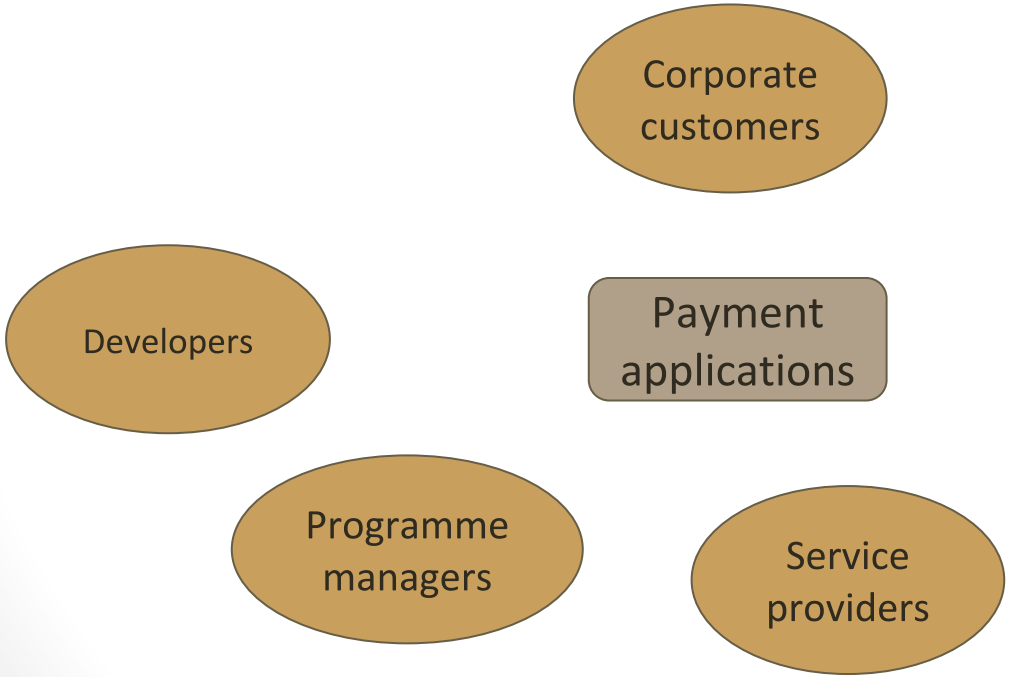
Building a payment application



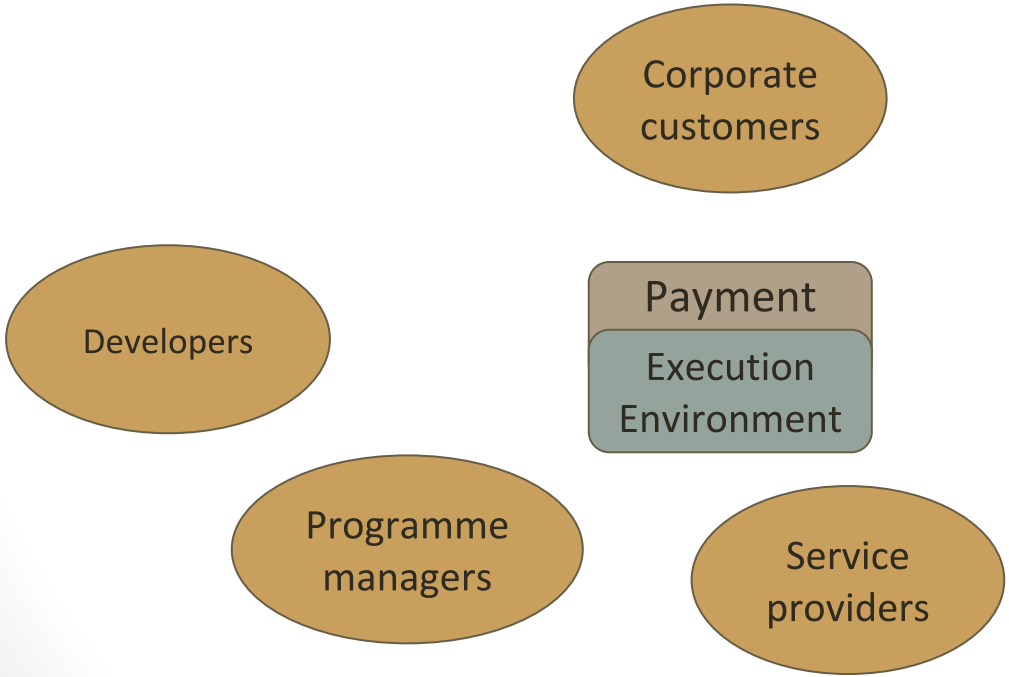
Building a payment application



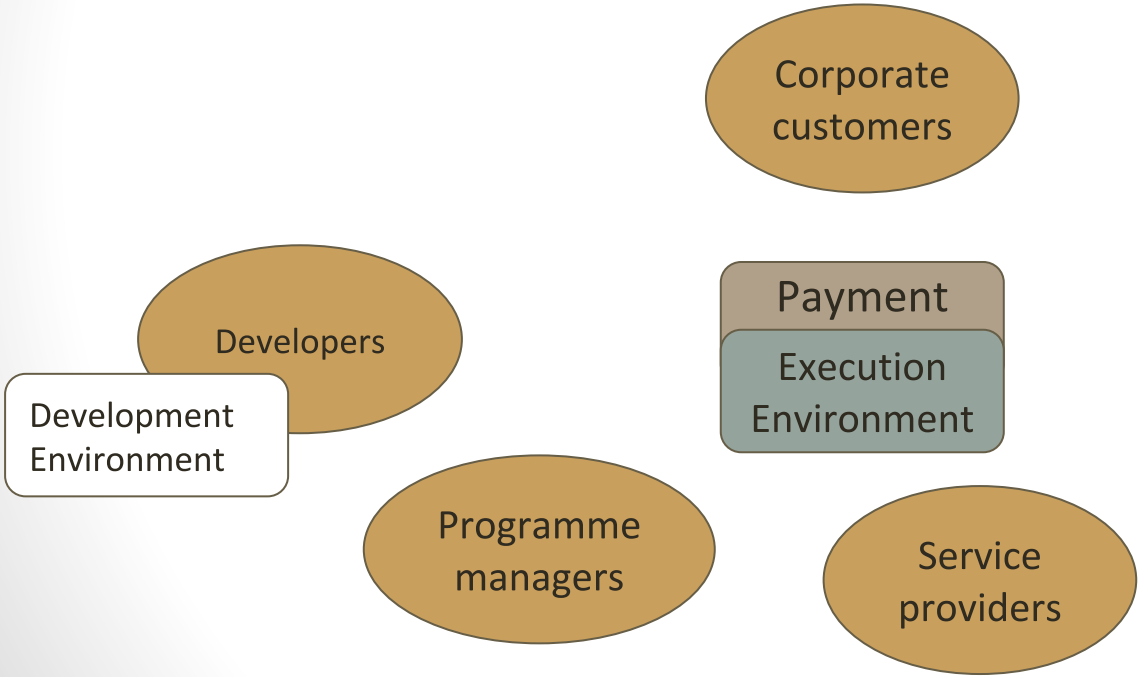
Open Payments Ecosystem



Open Payments Ecosystem



Open Payments Ecosystem



Open Payments Ecosystem

Corporate customers

Developers

Development Environment

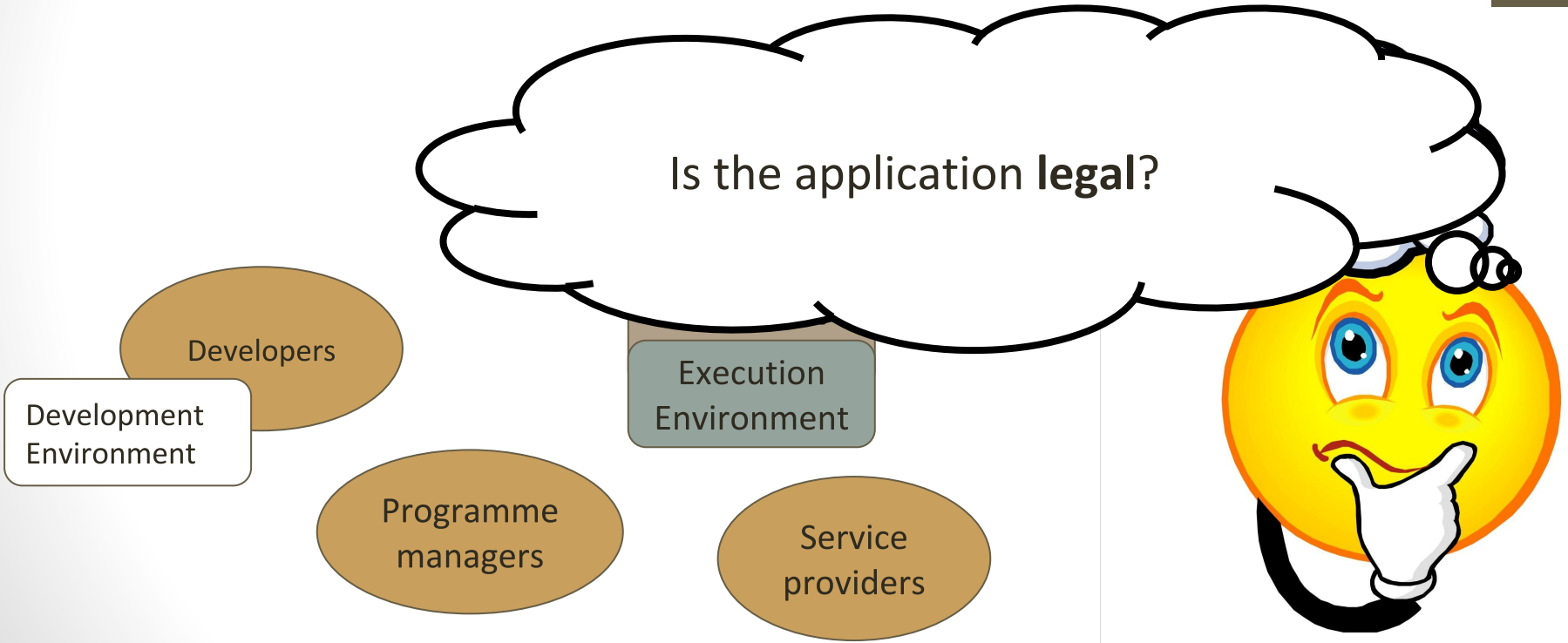
Payment

Execution Environment

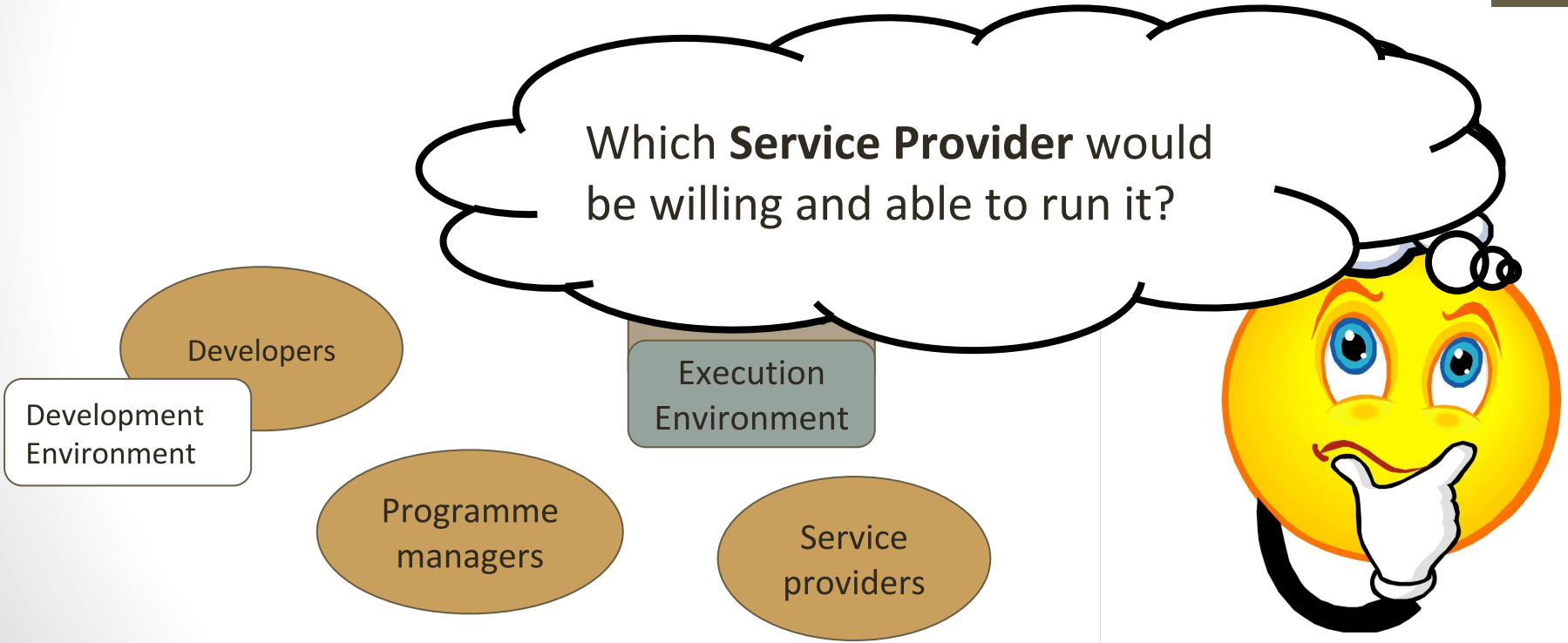
Programme managers

Service providers

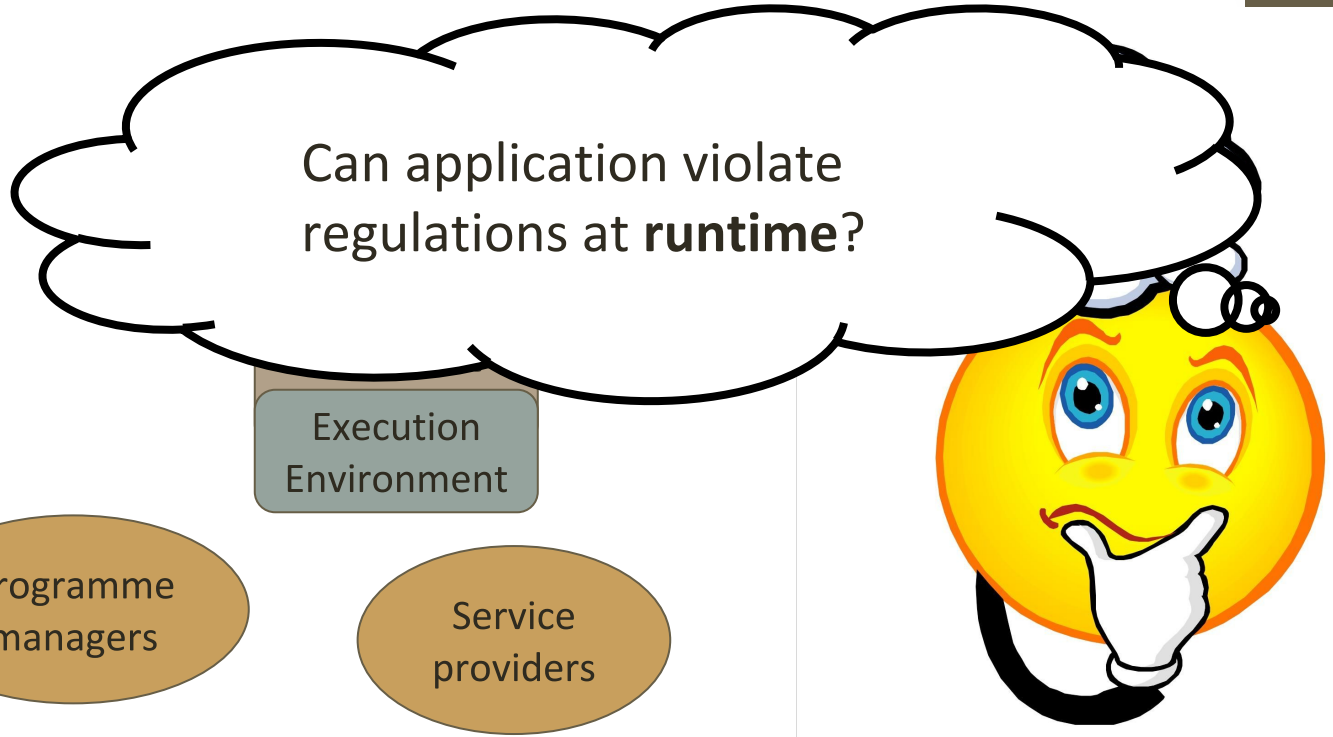
Open Payments Ecosystem



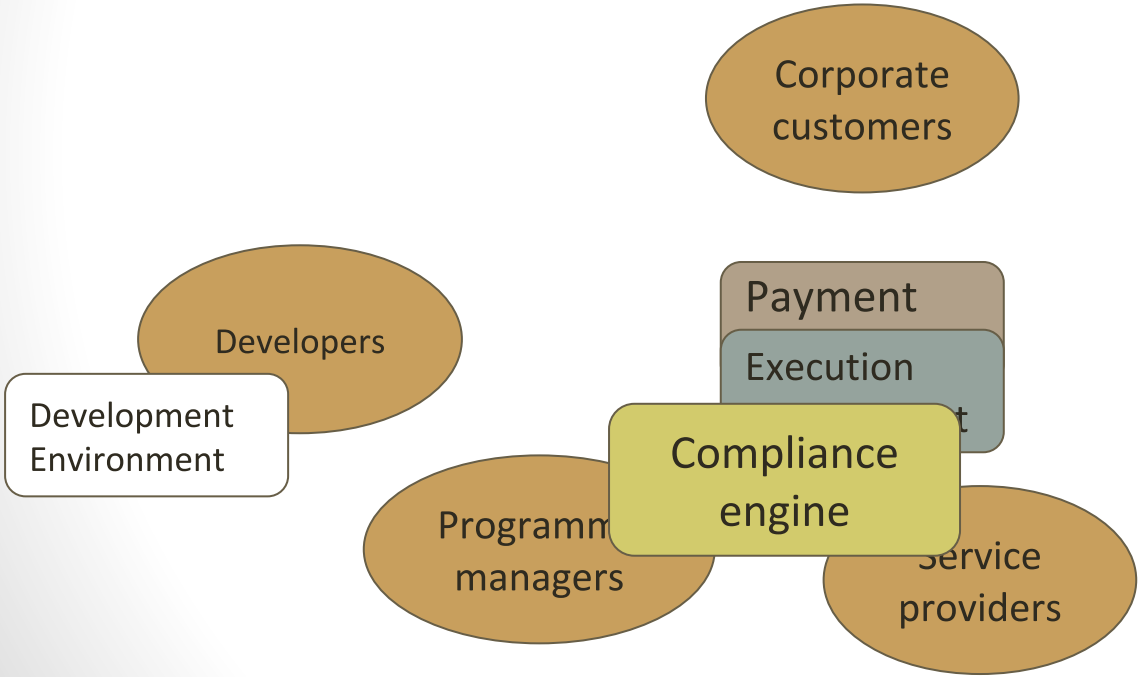
Open Payments Ecosystem



Open Payments Ecosystem



Open Payments Ecosystem + Compliance



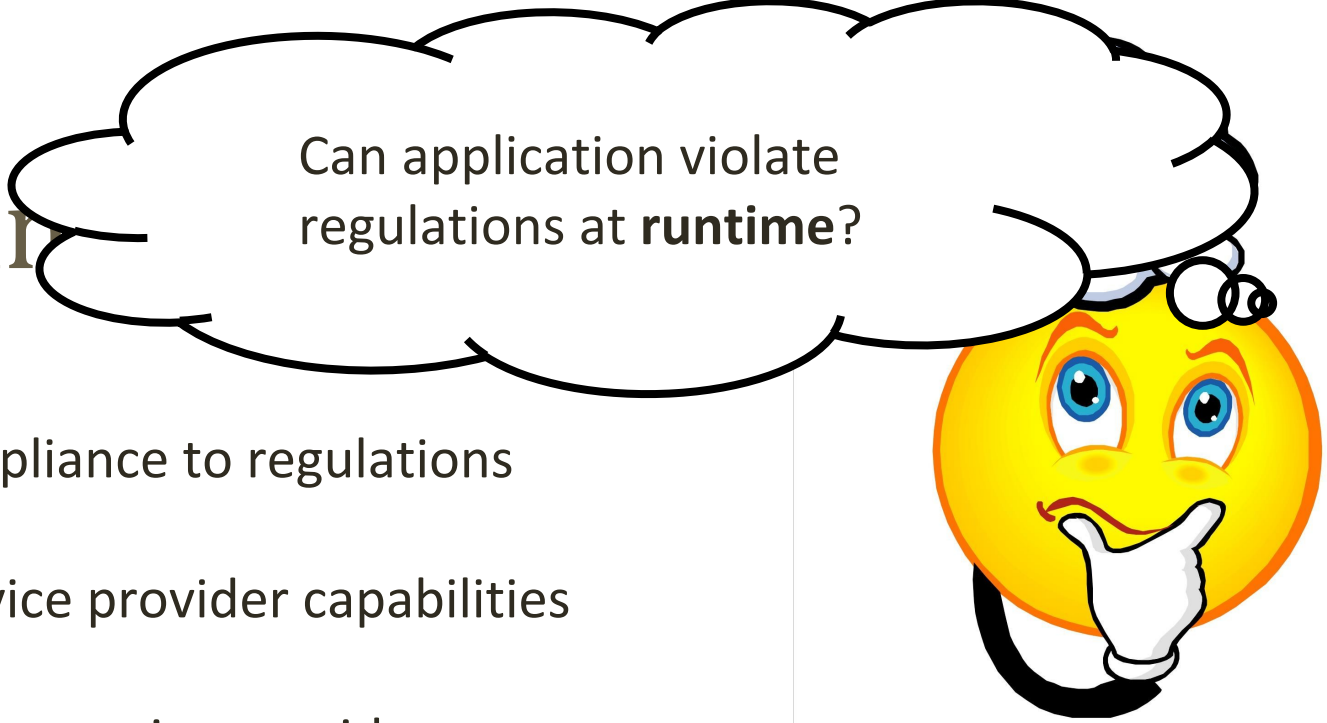
Compliance

1. Checking compliance to regulations

Compliance

1. Checking compliance to regulations
2. Matching service provider capabilities

Compliance



1. Checking compliance to regulations
2. Matching service provider capabilities
3. Limiting risk for service providers

Example

UK e-money regulations state that funds on financial instruments should be redeemable at par value.

1. Compliance to regulations
2. Capability matching
3. Risk mitigation

Example

1. Is the jurisdiction the UK?

UK e-money regulations state that funds on financial instruments should be redeemable at par value.

1. Compliance to regulations
2. Capability matching
3. Risk mitigation

Example

1. Does the application fall under the definition of e-money?

UK e-money regulations state that funds on financial instruments should be redeemable at par value.

1. Compliance to regulations
2. Capability matching
3. Risk mitigation

Example

UK e-money regulations state that funds on financial instruments should be redeemable at par value.

1. Are funds redeemable through the application?

1. Compliance to regulations
2. Capability matching
3. Risk mitigation

Example

UK e-money regulations state that funds on financial instruments should be redeemable at par value.

1. Is correct value given to the user

1. Compliance to regulations
2. Capability matching
3. Risk mitigation

Example

2. Can service provider support e-money applications?

UK e-money regulations state that funds on financial instruments should be redeemable at par value.

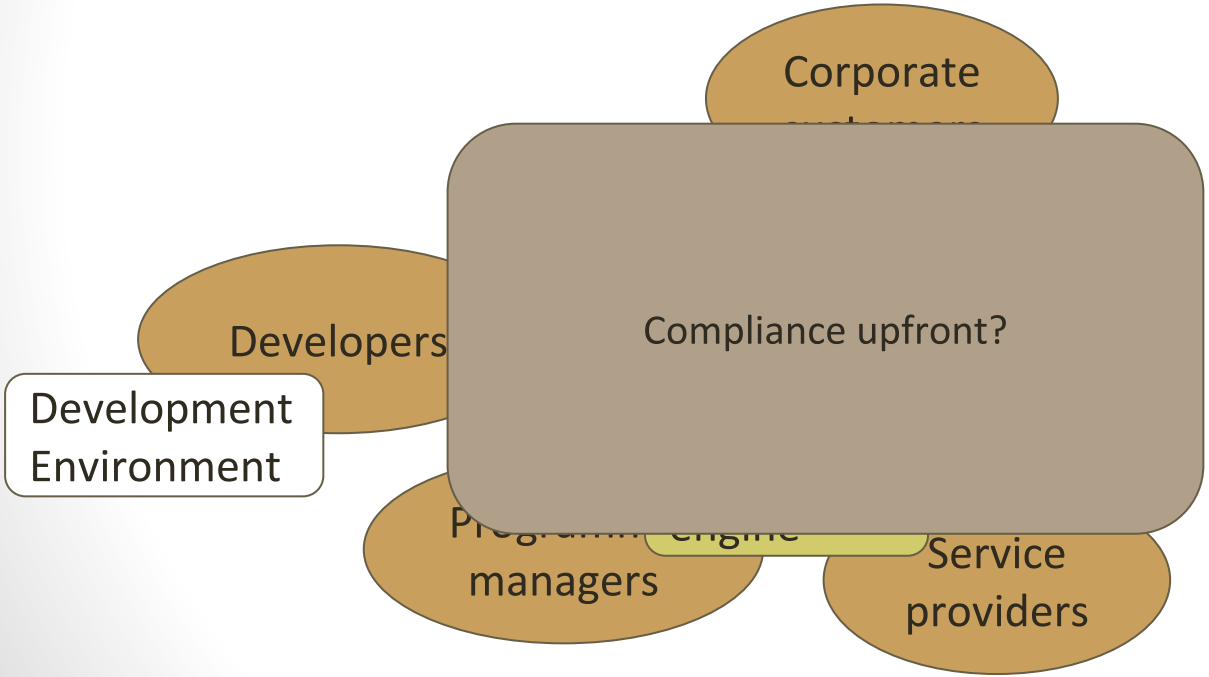
1. Compliance to regulations
2. Capability matching
3. Risk mitigation

Example

UK e-money regulations state that funds on financial instruments should be redeemable at par value.

3. How many funds are allowed on instruments?

Open Payments Ecosystem + Compliance



Compliance Challenges

- Not all properties are checkable upfront

Implication: SA not enough

Compliance Challenges

- Not all properties are checkable upfront

Implication: SA not enough

- Application is run by third party:
  - Not all information is available

Implication: SA can only be done on info provided

Compliance Challenges

- Not all properties are checkable upfront

Implication: SA not enough

- Application is run by third party:
  - Not all information is available
  - We cannot trust the application

Implication: SA can only be done on info provided

Implication: We have to verify model adherence at runtime

Compliance Challenges

- Not all properties are checkable upfront

Implication: SA not enough

- Application is run by third party:
  - Not all information is available
  - We cannot trust the application

Implication: SA can only be done on info provided

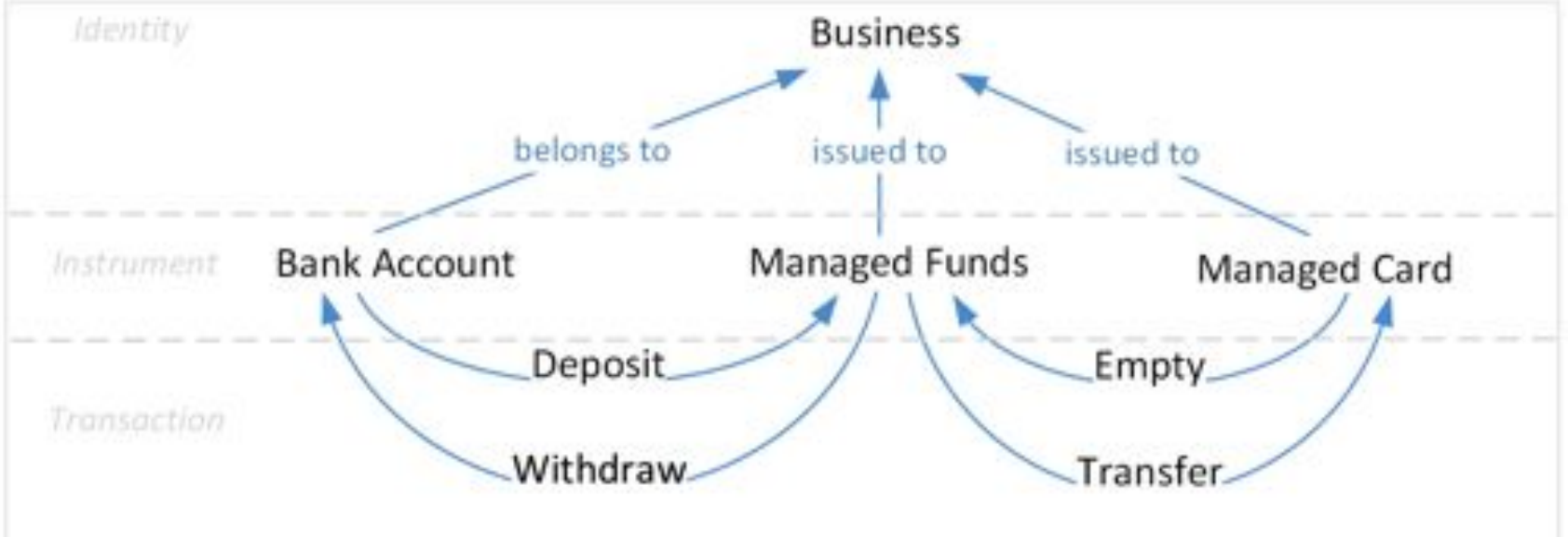
Implication: We have to verify model adherence at runtime

- Understanding the domain
  - Frequent legislation changes
  - Academics will not remain involved

Implication: We need a common language

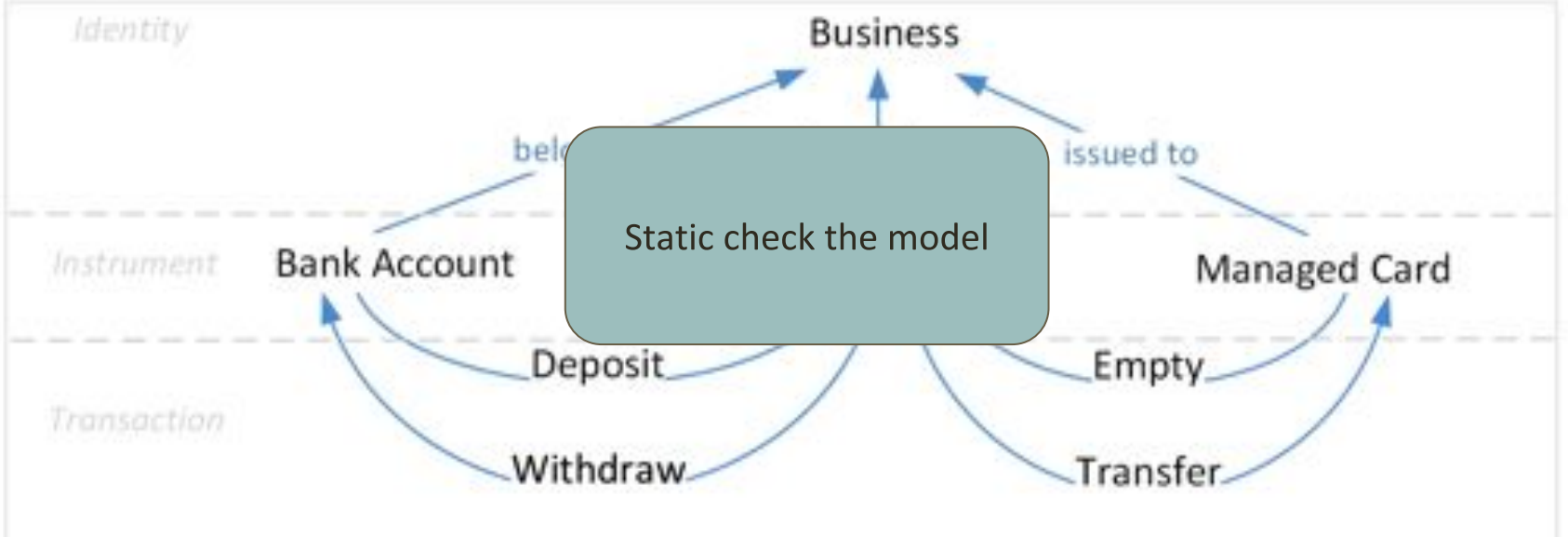
Info Provided by Developer - The Model

Developer submits model of application rather than implementation



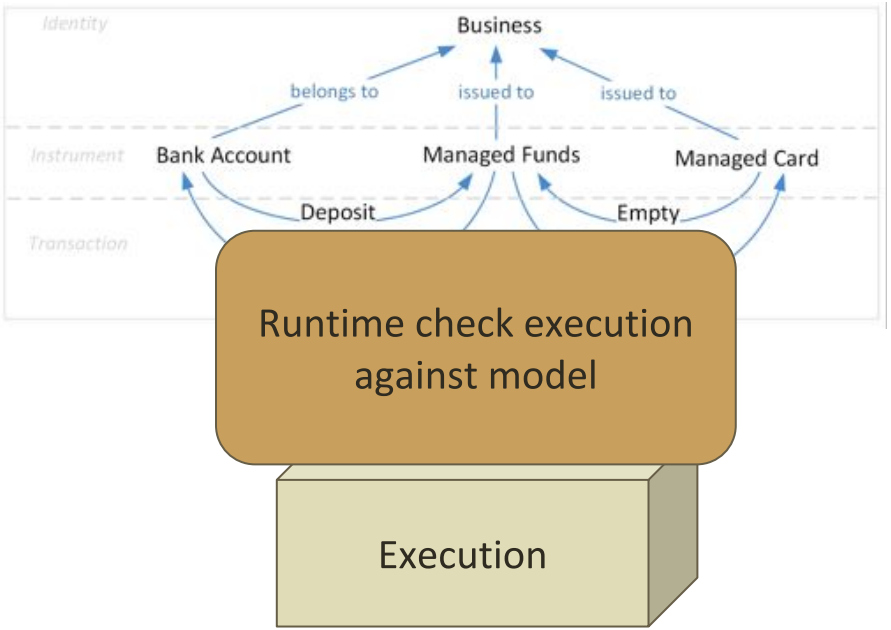
Info Provided by Developer - The Model

Developer submits model of application rather than implementation

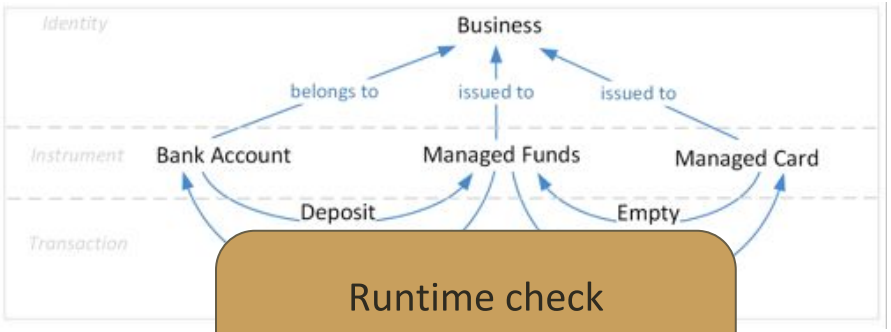


Static check the model

Application Execution



Application Execution



Runtime check implementation against model

Execution

Runtime verify remaining checks, eg: limits, at par value, delays

Example

UK e-money regula ncial instruments should be redee

1. Is the jurisdiction the UK?

2. Can service provider support e-money applications?

1. Does the application fall under the definition of e-money?

1. Are funds redeemable through the application?

Static check the model

Example

Runtime check implementation against model

UK e-money regulations state that funds on financial instruments should be redeem

1. Does the application fall under the definition of e-money?

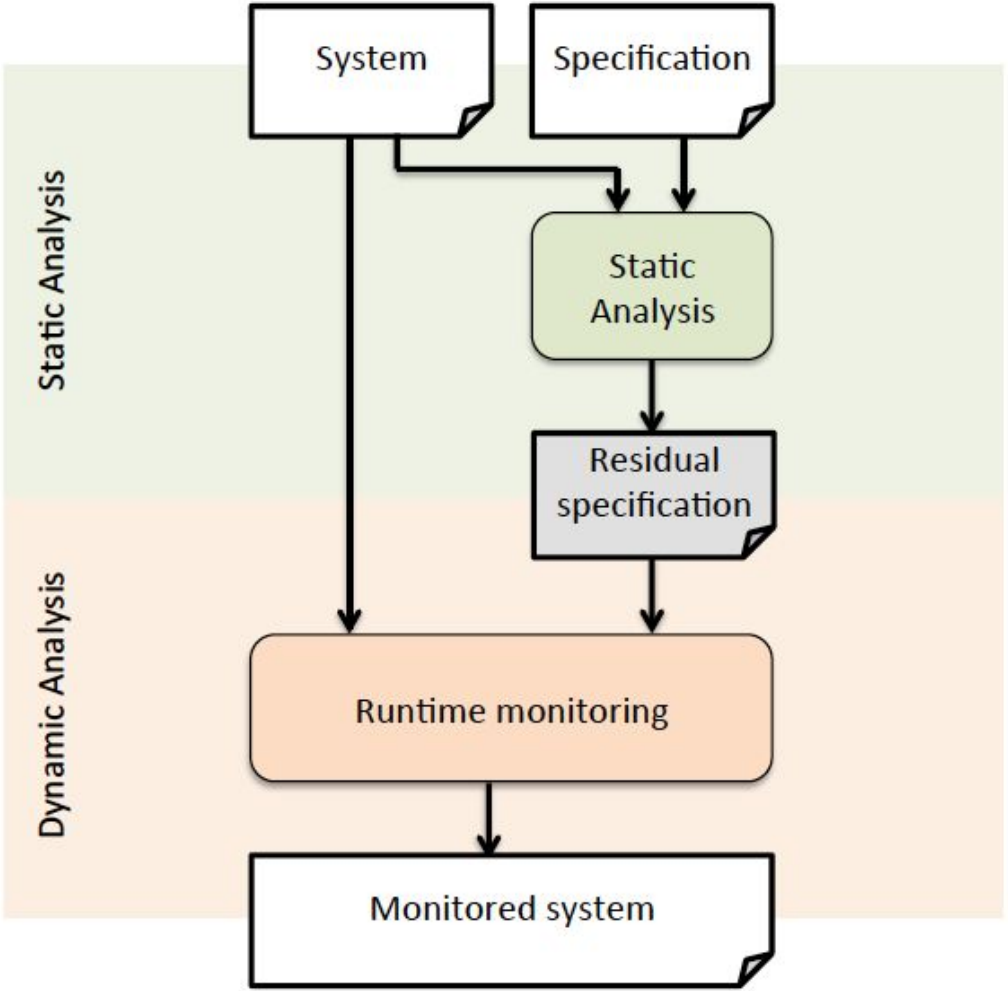
Example

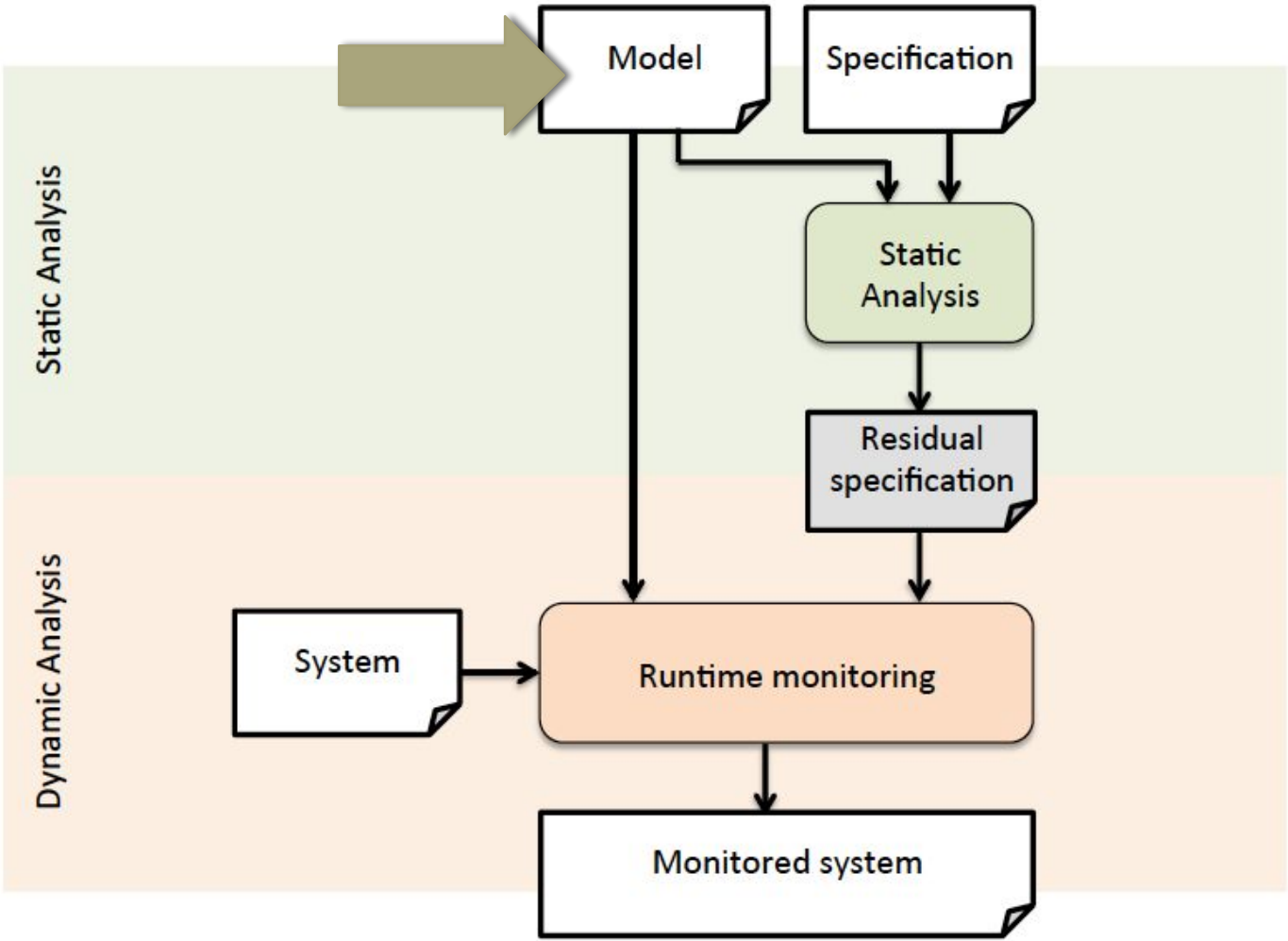
Runtime verify remaining checks,
eg: limits, at par value, delays

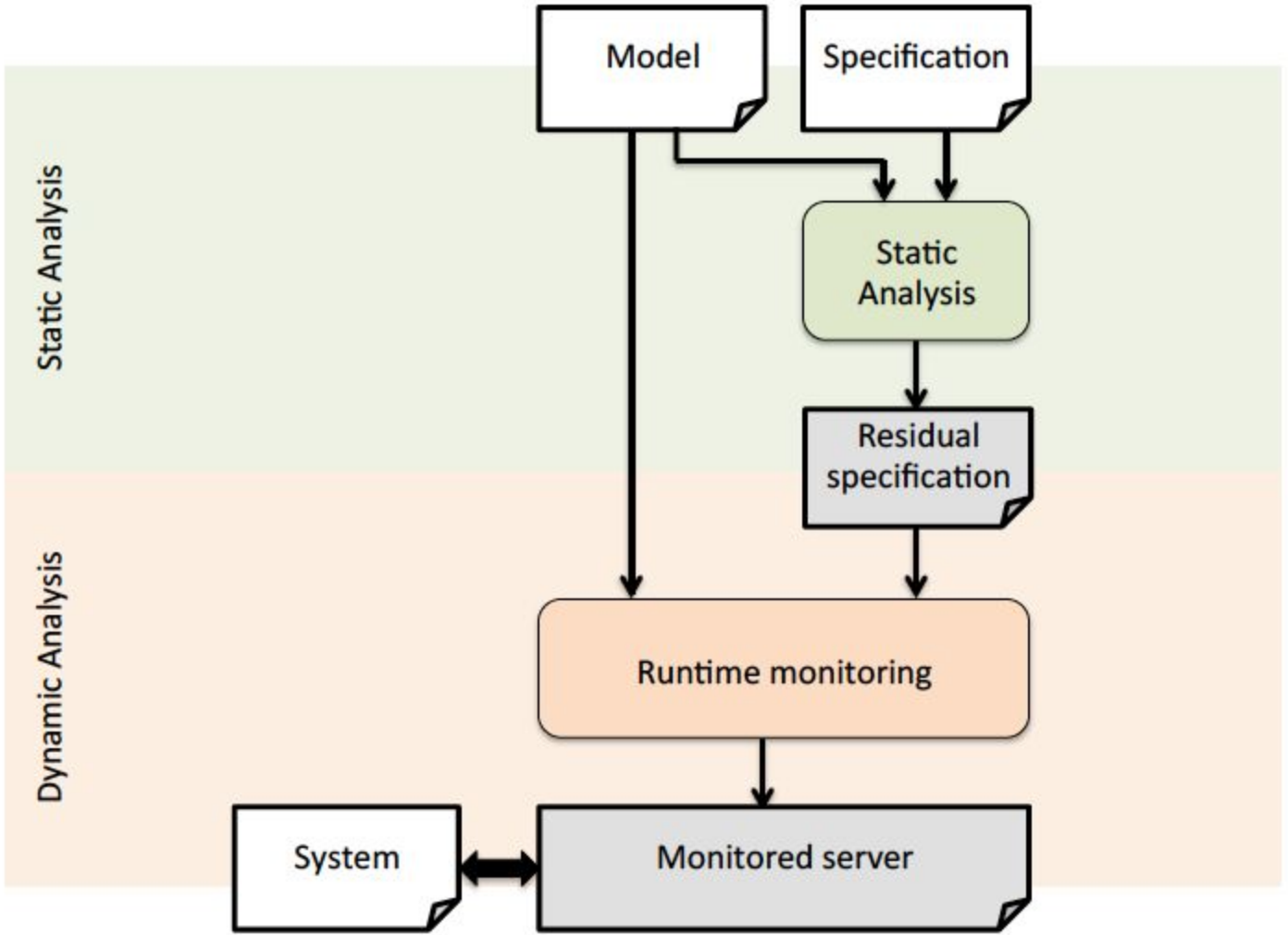
UK e-money regulations state that funds on financial instruments should be redeemable at par value.

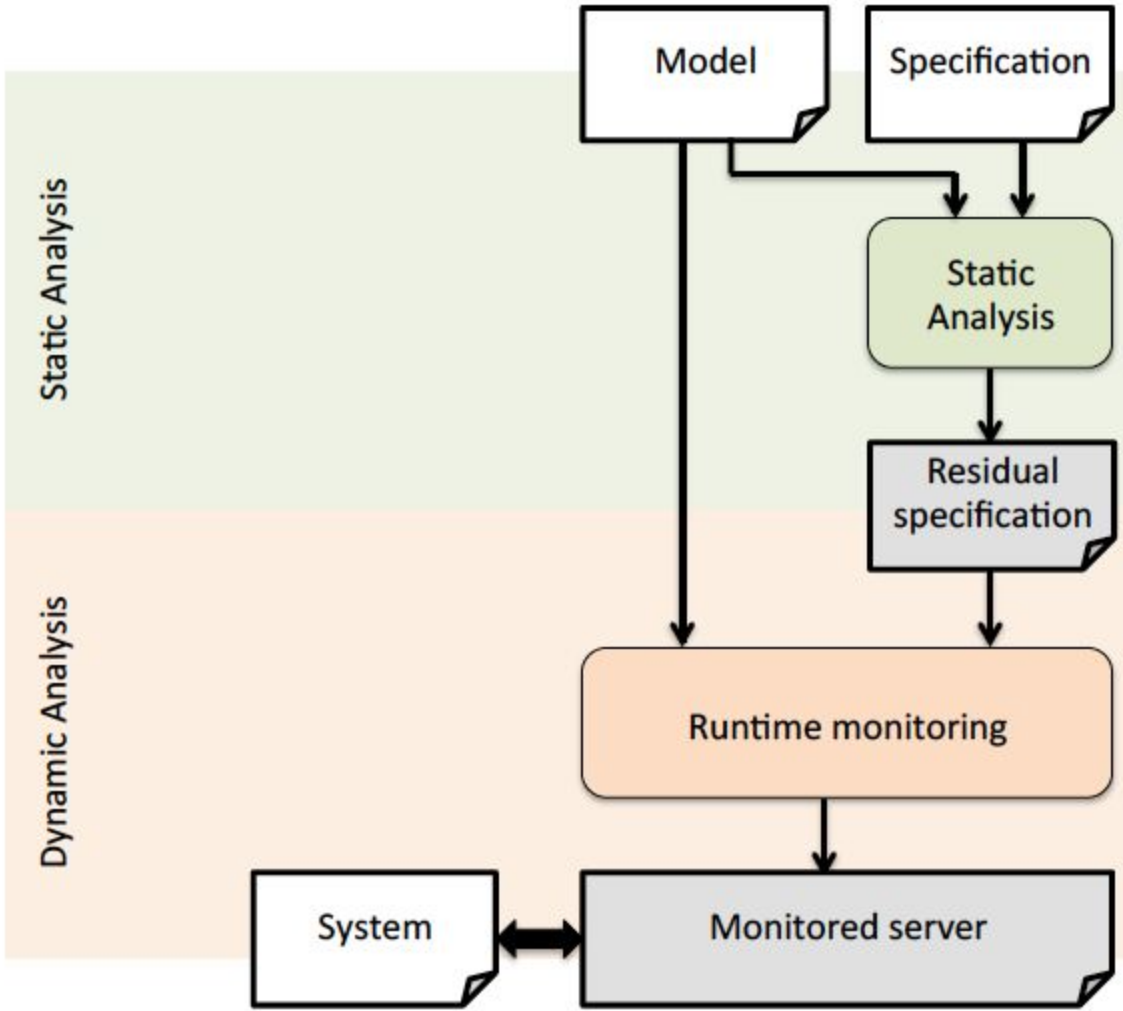
1. Is correct value given to the user

3. How many funds are allowed on instruments?


System
Specification
Static Analysis
Static Analysis
Residual specification
Dynamic Analysis
Runtime monitoring
Monitored system


Static Analysis
Dynamic Analysis
Model
Specification
Static Analysis
Residual specification
System
Runtime monitoring
Monitored system


Model
Specification
Static Analysis
Static Analysis
Residual specification
Dynamic Analysis
Runtime monitoring
System
Monitored server


Static Analysis
Dynamic Analysis
Model
Specification
Static Analysis
Residual specification
Runtime monitoring
System
Monitored server

RV design choices - Process

The process was not in our control

RV design choices - Engineering

- **Synchronised** monitoring with a **timeout** for each transaction
  - Monitor gives go ahead
- Properties which do not affect particular transaction success are monitored **asynchronously**
  - E.g., Fraud risk monitoring
  - E.g., Statistics gathering

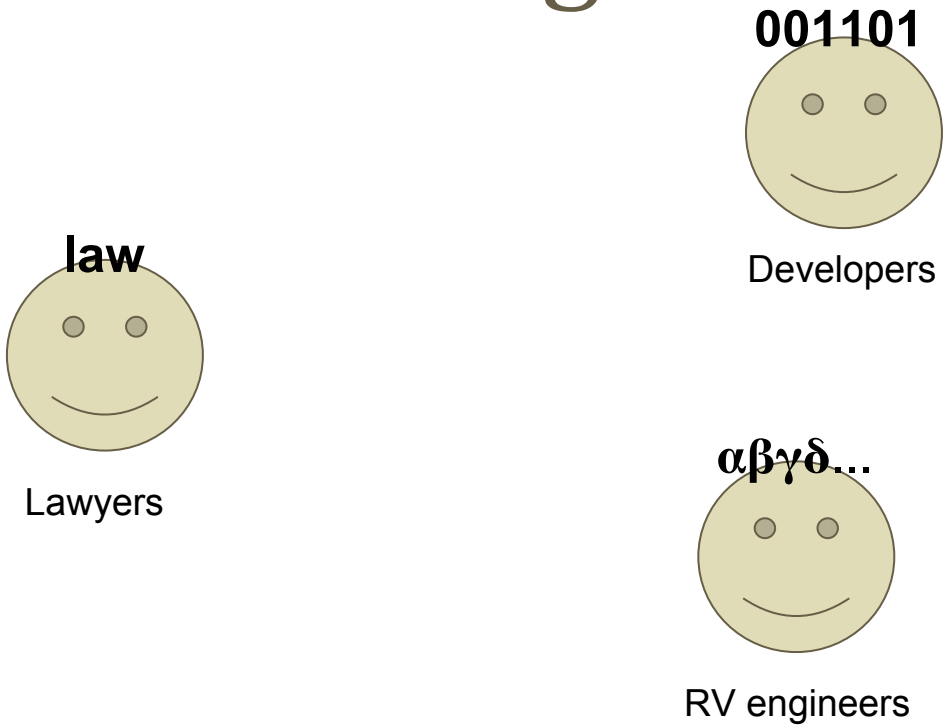
RV design choices - Engineering

Communication: asynchronous messaging in Akka microservices architecture

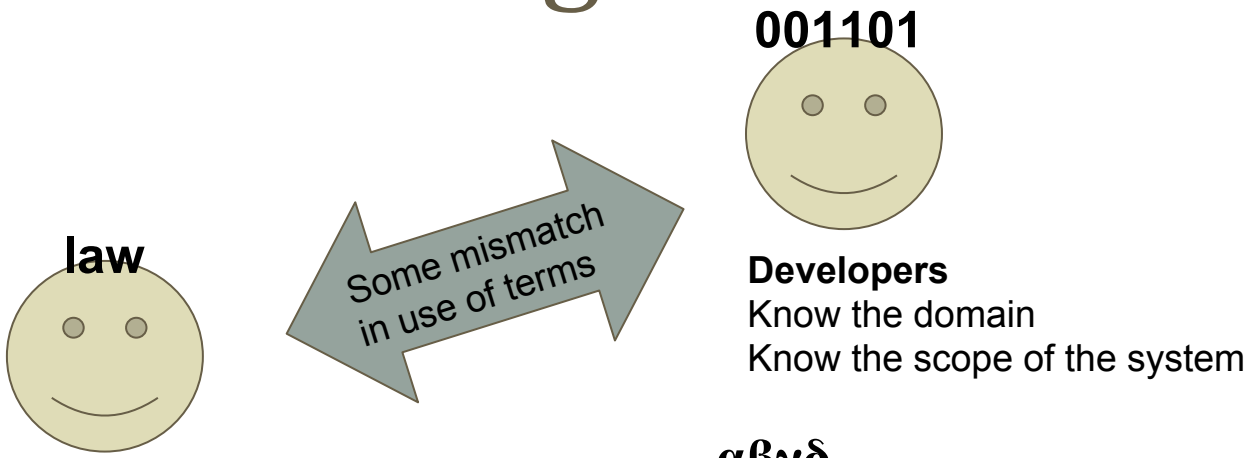
Event extraction: events-by-design

Algorithm: Java code if-then-else statements
(automatically translated from controlled natural language)

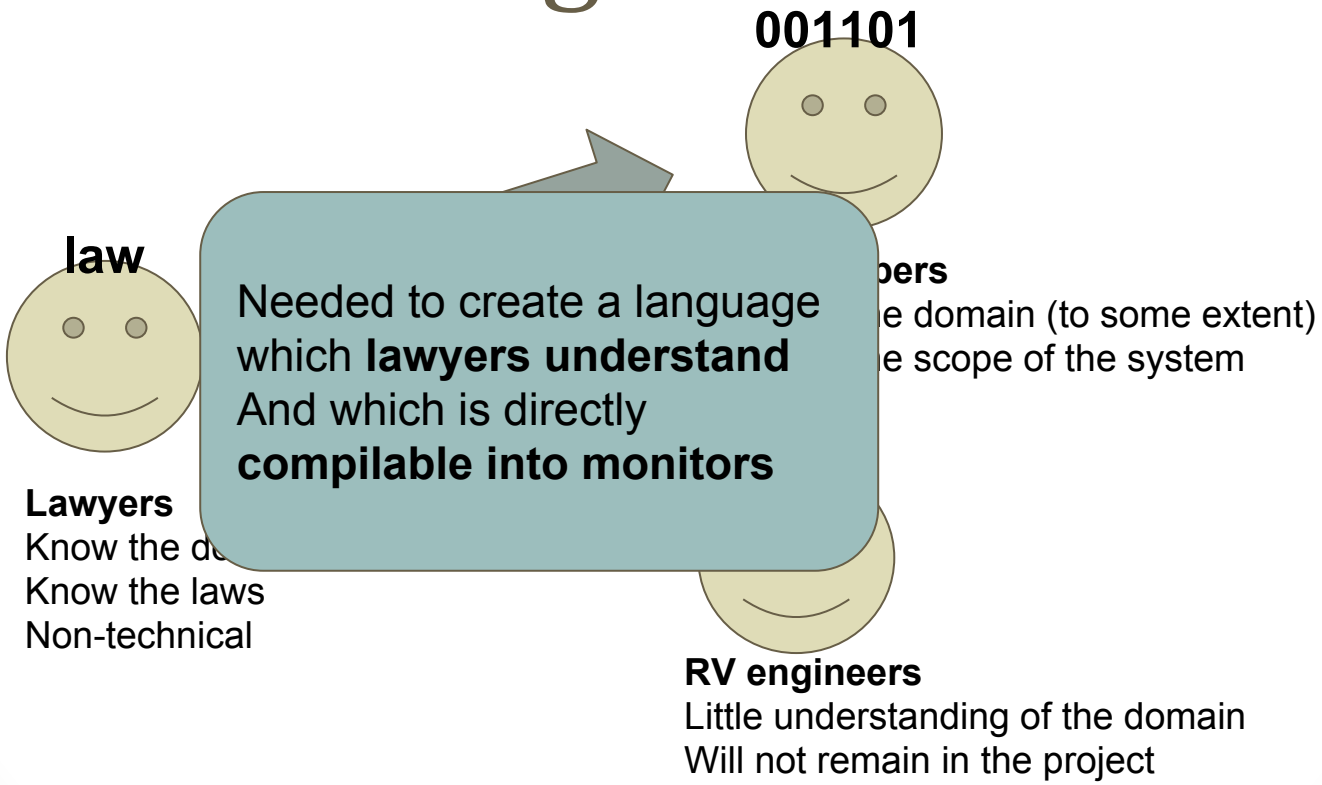
Understanding the domain

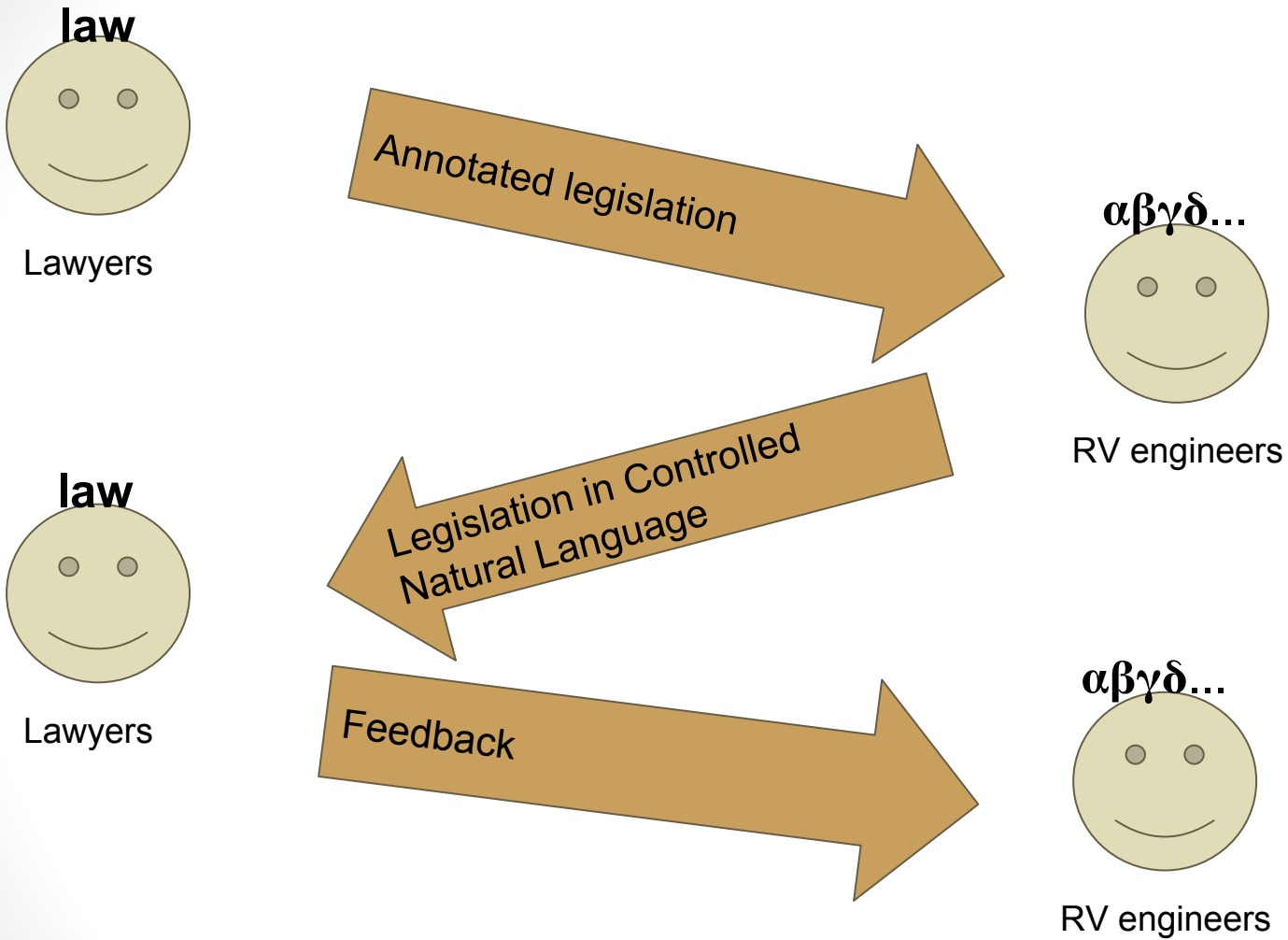


Understanding the domain



Understanding the domain

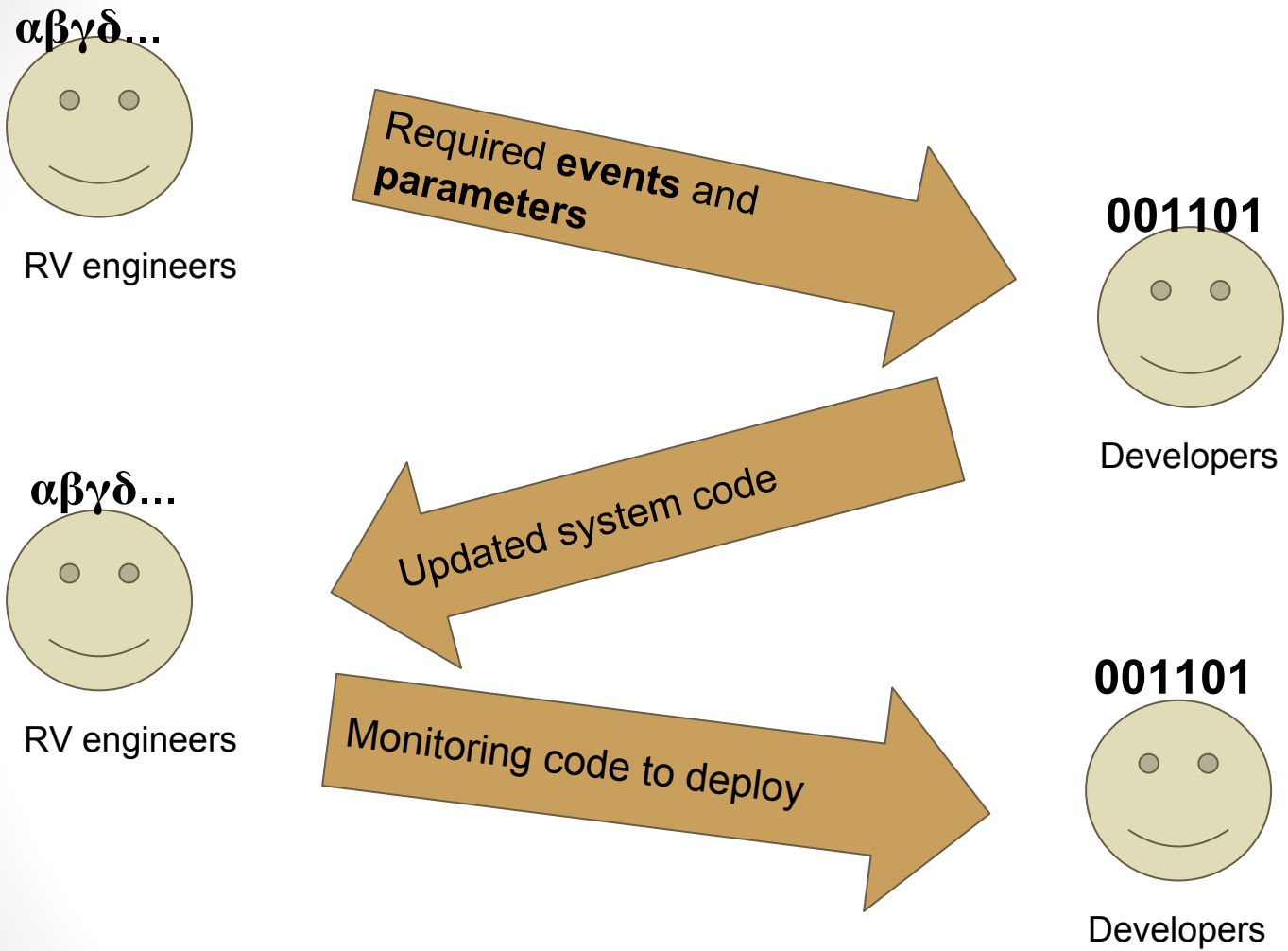



law
Lawyers
Annotated legislation
αβγδ…
RV engineers
law
Lawyers
Legislation in Controlled Natural Language
Feedback
αβγδ…
RV engineers

Example - Controlled Natural Language

LAW: An electronic money issuer must not award (a) interest in respect of the holding of electronic money; or (b) any other benefit related to the length of time during which an electronic money holder holds electronic money.

CNL: For each programme p, and instrument i, where p is regulated in the UK, and i deals with e-money, then i does not give time-based rewards.


αβγδ…
RV engineers
Required **events** and **parameters**
001101
Developers
αβγδ…
RV engineers
Updated system code
Monitoring code to deploy
001101
Developers

Thoughts on this project

- Challenge in understanding domain
- Industry-led project and RV by-design made life “easy”

- Interesting solution for combining static and dynamic checking
- Interesting combination of sync and async monitoring

Conclusion

Conclusion - Do they live happily ever after?

RV has a lot to offer to industry

Better communication is needed both ways

Balance of industrial and academic “success”

Open challenges

Overhead fine-grained control to system administrators

- Eg: switching properties on and off

How much state to keep per monitor

- Jump-starting the state when turning properties on

Robustness

- Persisting monitors, caching, etc

Up to us...


REGISTRAR
"HAPPY EVER AFTER" LAND
CHURCH
TO HELL AND BACK
DIVORCE COURTS
JUST MARRIED