



HYPERLEDGER **FABRIC**

22nd January 2020

Oleksii Kulikov • Lukas Schöbel

Technical University of Munich

Chair for Application and Middleware Systems

AGENDA

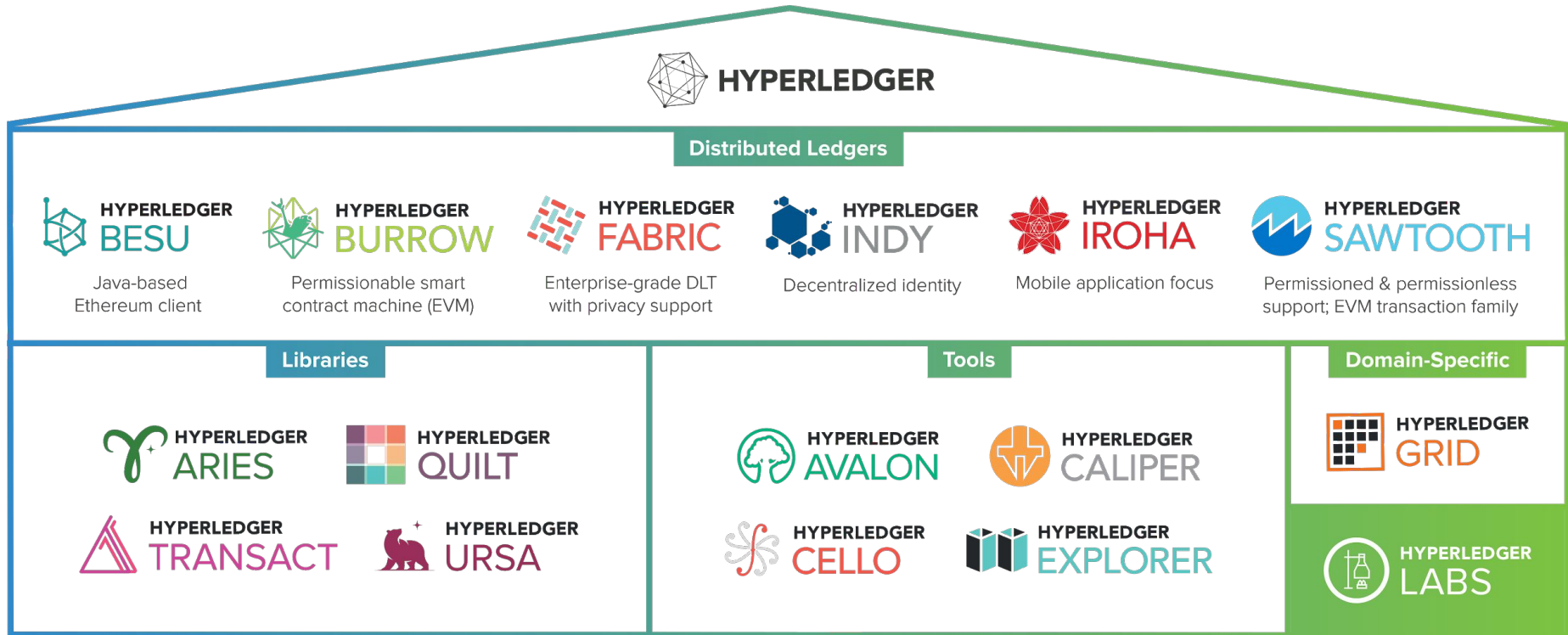
- **Hyperledger Fabric**
 - Hyperledger - History & Ecosystem
 - Classification of Hyperledger Fabric
 - Operating Principle
 - Pro & Con
- **Prototype**
 - Previous Ideas
 - Motivation for our project
 - DEMO
 - Outlook

HYPERLEDGER PROJECT



- Goal: Development of industrial large-scale blockchain applications
- Founded in 2016 (IBM, Cisco, J.P. Morgan, Deutsche Börse Group)
- 250+ companies involved today
- Distributed ledger technology ensures transparent and decentralized open standard

HYPERLEDGER ECOSYSTEM



Source: [A]

HYPERLEDGER FABRIC



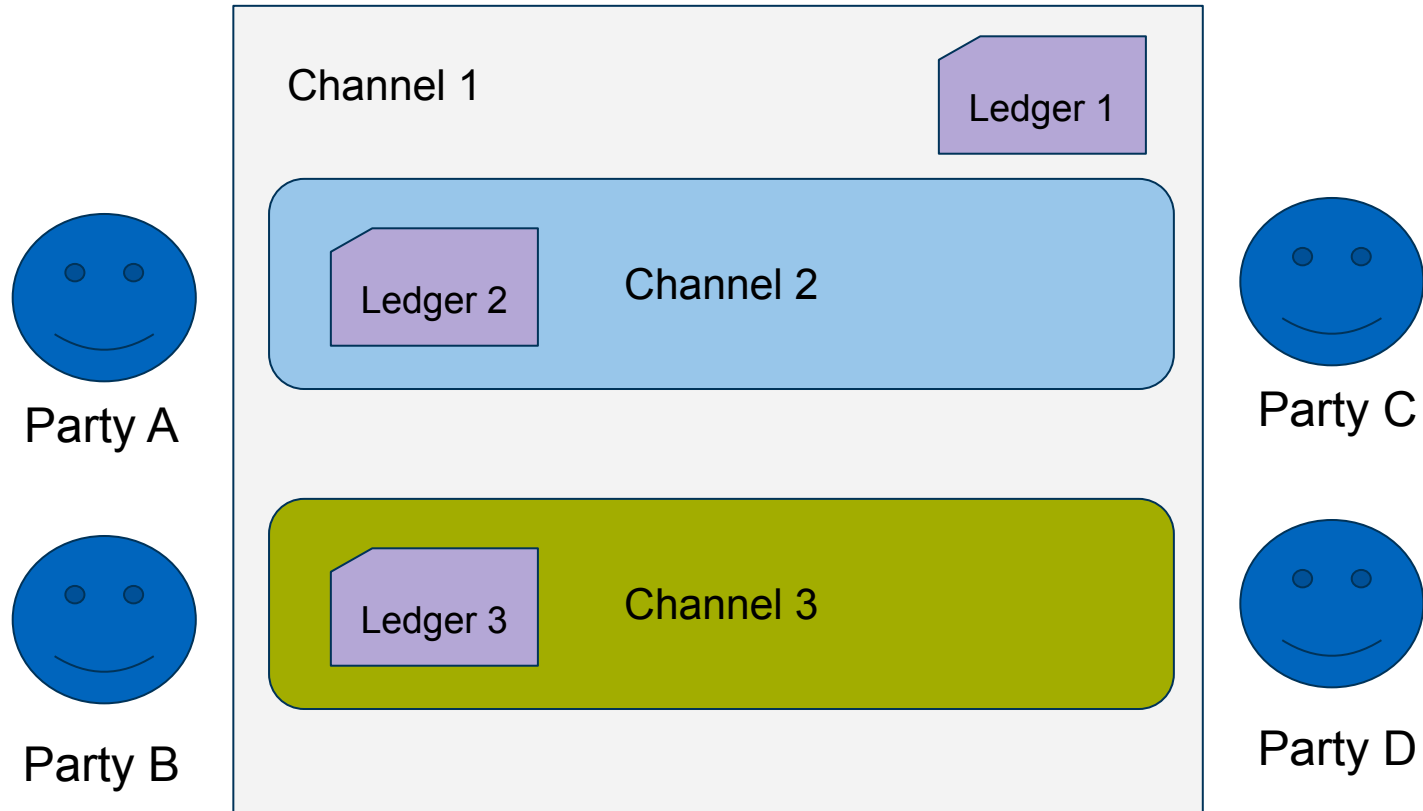
- Permissioned blockchain with modular architecture
- Throughput of 20,000 tps possible [1]
- SDK in various common languages (Java, Javascript, Go)

COMPARISON

	Bitcoin	Ethereum	Hyperledger Fabric	R3 Corda
Business Area	cryptocurrency	cryptocurrency, B2C	B2B	B2B
Type	public, permissionless	public, permissionless	private, permissioned	private, permissioned
Contracts	no smart contracts	smart contracts e.g. with Solidity	smart contracts e.g. with Java, Javascript	smart contracts e.g. with Kotlin, Java
Currency	Bitcoin	Ether	none	none

OPERATING PRINCIPLE

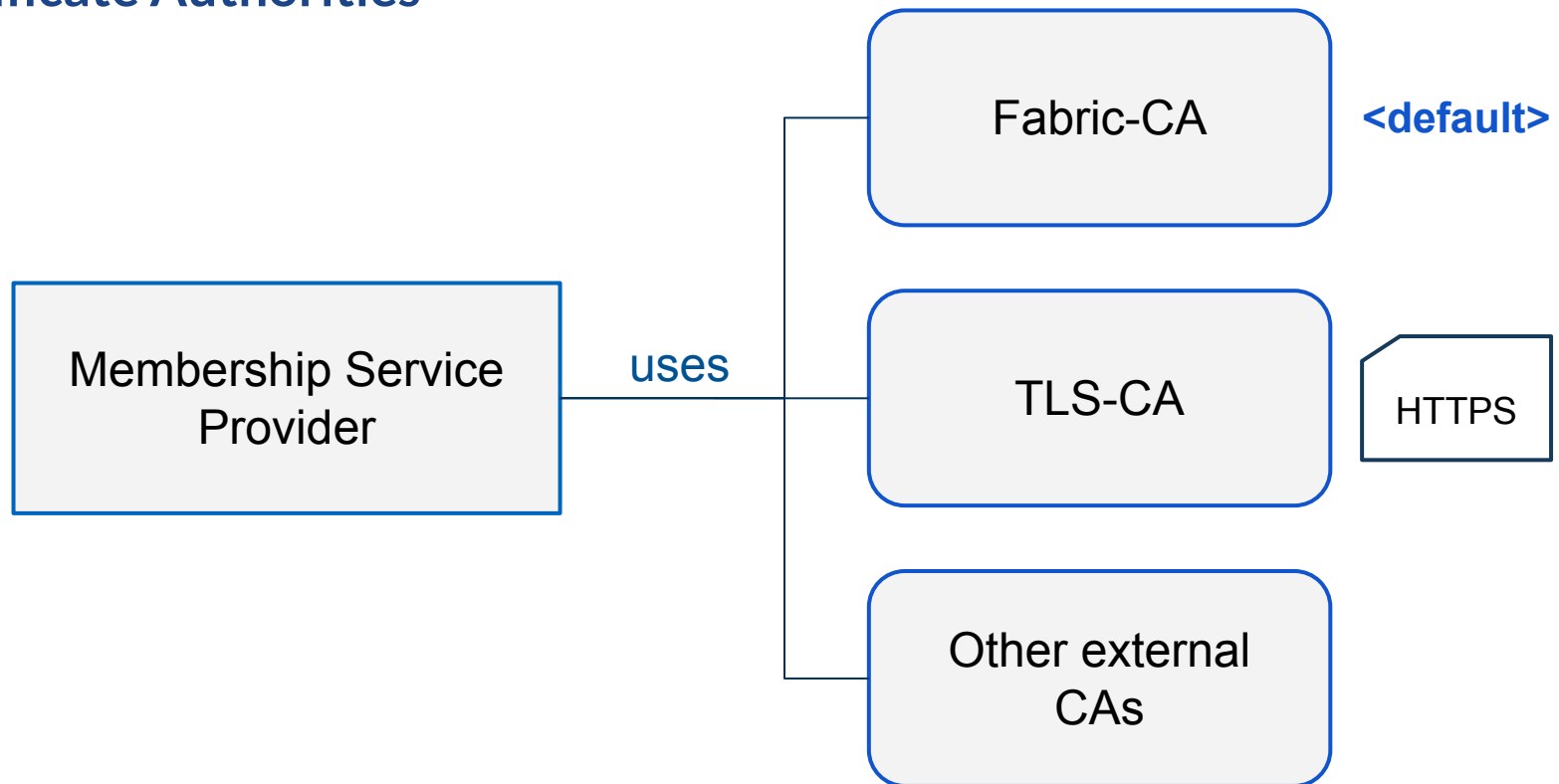
Channels



Source: [B]

OPERATING PRINCIPLE

Membership & Certificate Authorities



OPERATING PRINCIPLE

Nodes

- Endorsing peers
 - test-execute transactions
 - create chaincode

OPERATING PRINCIPLE

Nodes

- Endorsing peers
 - test-execute transactions
 - create chaincode
- Committing peers
 - validate & commit transactions

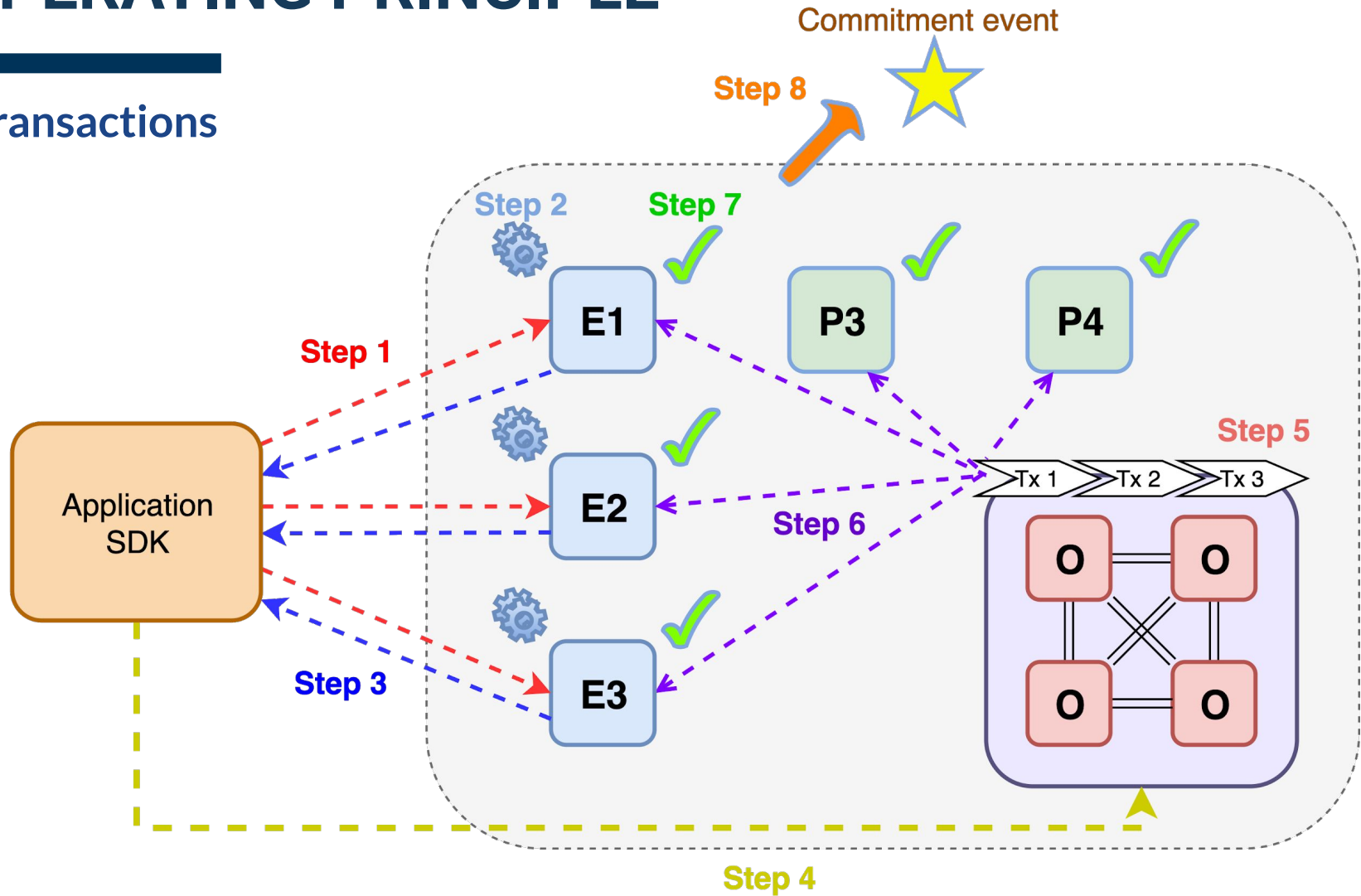
OPERATING PRINCIPLE

Nodes

- Endorsing peers
 - test-execute transactions
 - create chaincode
- Committing peers
 - validate & commit transactions
- Ordering service peers
 - order transactions

OPERATING PRINCIPLE

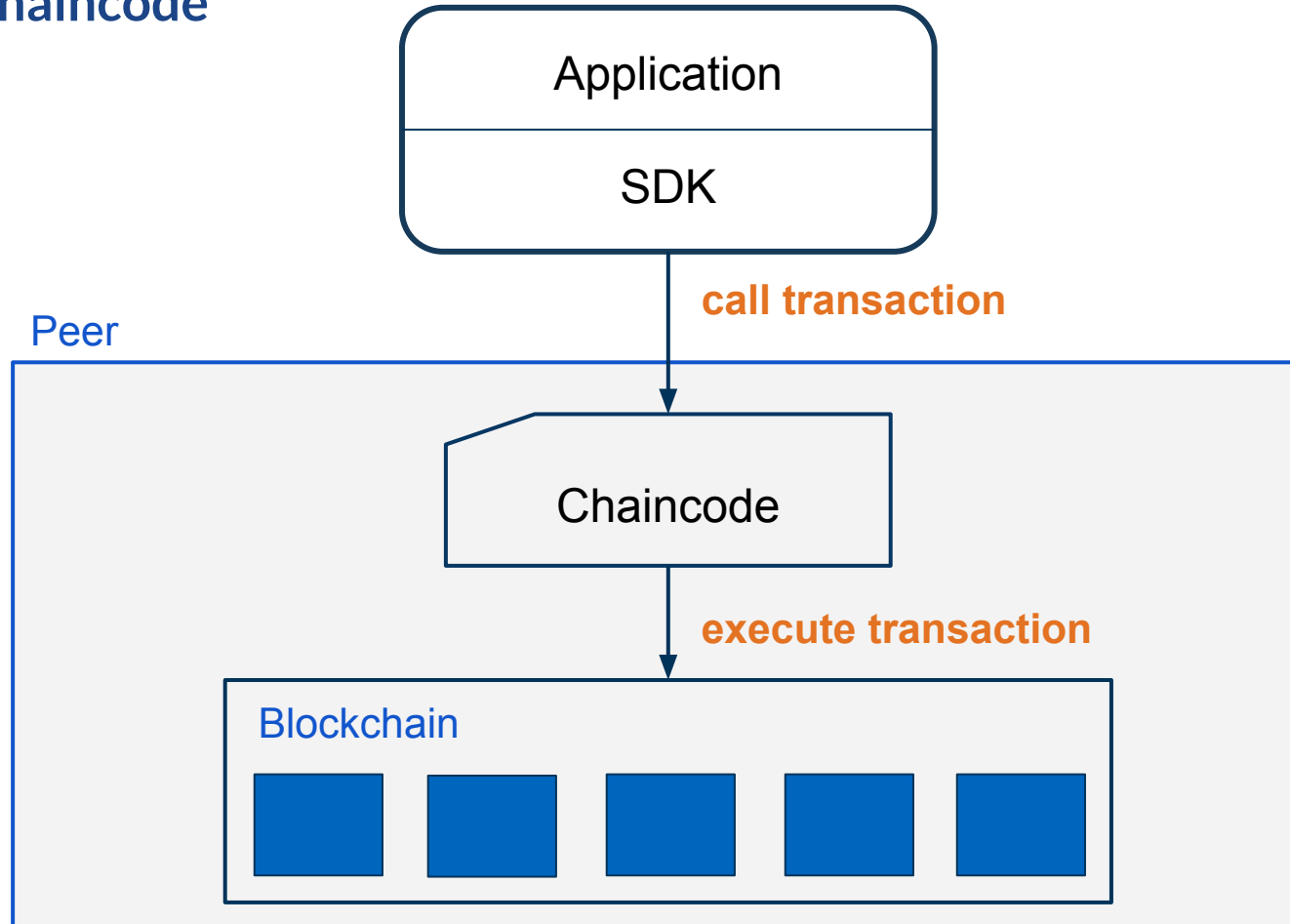
Transactions



Source: [B]

OPERATING PRINCIPLE

Apps & Chaincode



Source: [C]

PRO & CON

PRO	CON
open source development	lack of resources & documentation
modular toolbox	(too) fast development
privacy through channels	

PROTOTYPE - IDEAS



PROTOTYPE - USE CASE



PROTOTYPE - USE CASE



**Money
Laundering**



Corruption

PROTOTYPE

Authority



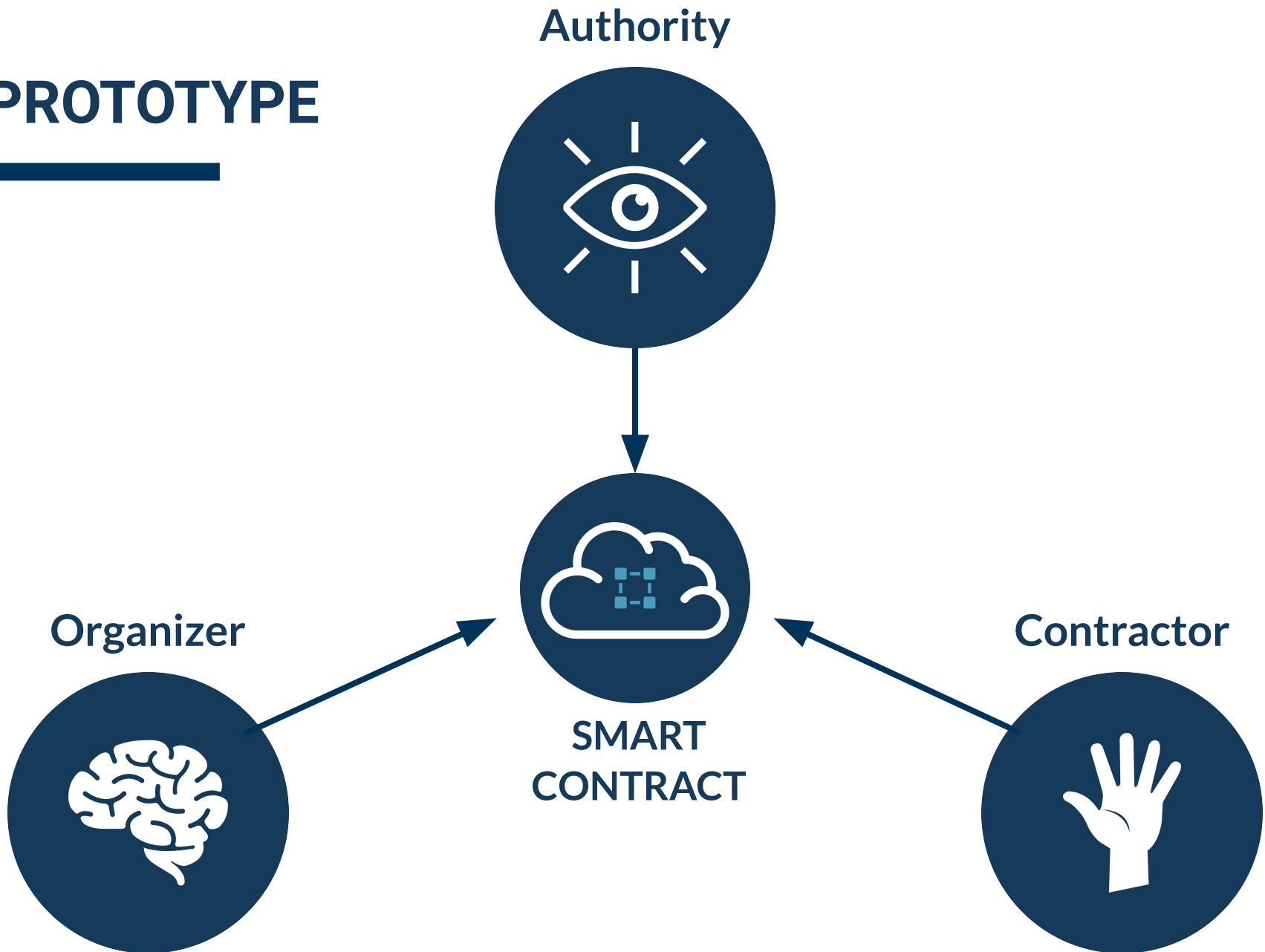
Organizer



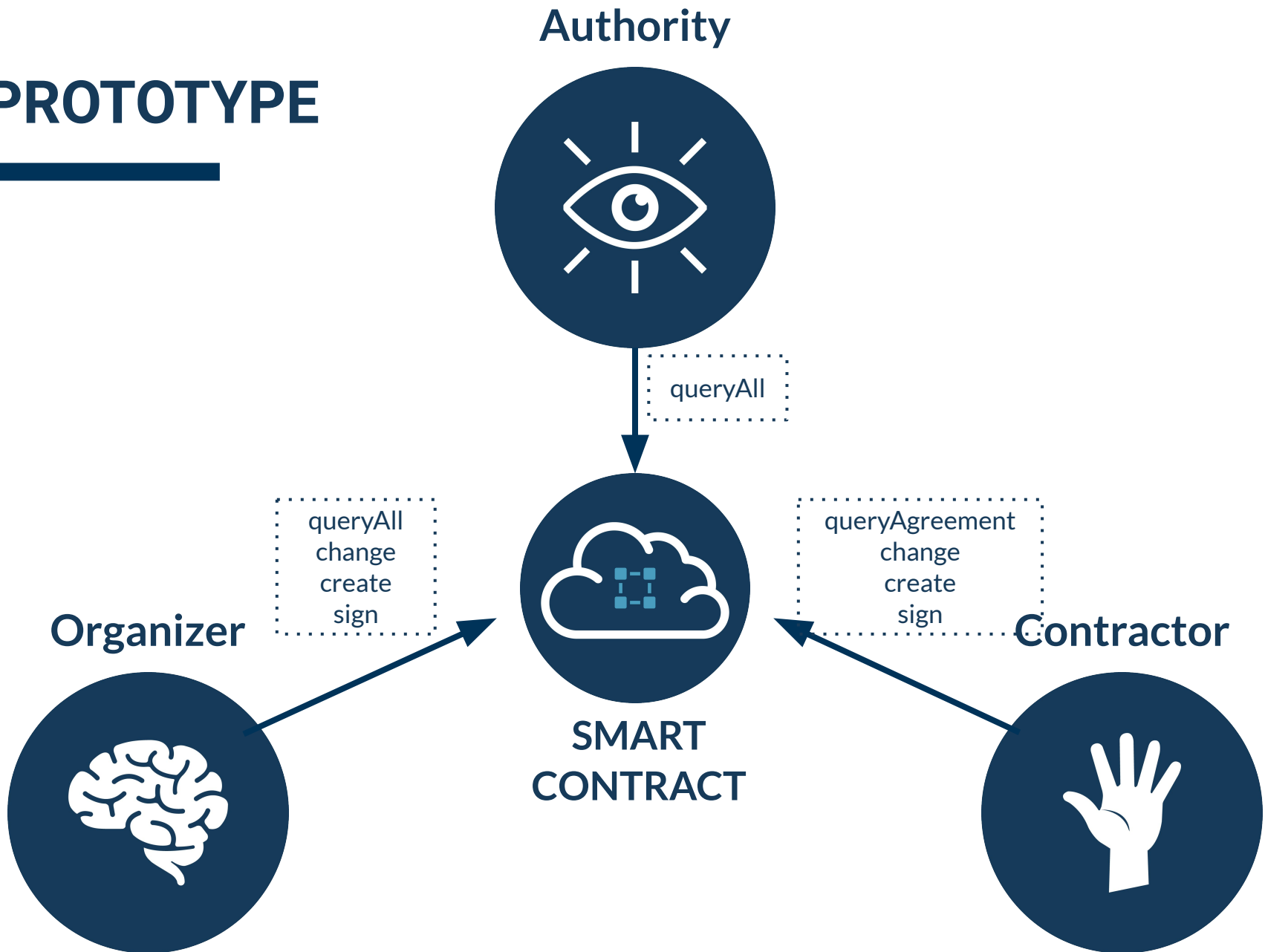
Contractor



PROTOTYPE



PROTOTYPE



PROTOTYPE - BENEFITS



Efficient



Analyzable



Safe



Transparent

USE CASE - AIRPORT BER



Source: [D]

USE CASE - AIRPORT BER

START

2006

END

2012

COST

2 B€

Source: [D]

USE CASE - AIRPORT BER

START

2006

END

2020?

COST

7.3 B€

Source: [D]

DEMO



DEMO - STARTUP

```
./startFabric.sh javascript
github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp -e CORE_PEER_TLS_ROOTCERT_FI
LE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt cl
i peer chaincode install -n cashflow -v 1.0 -p /opt/gopath/src/github.com/chaincode/cashflow/javascript -l node
2020-01-19 14:14:12.422 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc
2020-01-19 14:14:12.422 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
2020-01-19 14:14:15.971 UTC [chaincodeCmd] install -> INFO 003 Installed remotely response:<status:200 payload:"OK" >
+ echo 'Instantiating smart contract on mychannel'
Instantiating smart contract on mychannel
+ docker exec -e CORE_PEER_LOCALMSPID=Org1MSP -e CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrg
anizations/org1.example.com/users/Admin@org1.example.com/msp cli peer chaincode instantiate -o orderer.example.com:7050 -C mychannel -n ca
shflow -l node -v 1.0 -c '{"Args":[]}' -P 'AND('Org1MSP.member','Org2MSP.member')' --tls --cafile /opt/gopath/src/github.com/h
yperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem --pe
erAddresses peer0.org1.example.com:7051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org
1.example.com/peers/peer0.org1.example.com/tls/ca.crt
2020-01-19 14:14:16.696 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc
2020-01-19 14:14:16.697 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
+ echo 'Waiting for instantiation request to be committed ...'
Waiting for instantiation request to be committed ...
+ sleep 10
+ echo 'Submitting initLedger transaction to smart contract on mychannel'
Submitting initLedger transaction to smart contract on mychannel
+ echo 'The transaction is sent to the two peers with the chaincode installed (peer0.org1.example.com and peer0.org2.example.com) so that
chaincode is built before receiving the following requests'
The transaction is sent to the two peers with the chaincode installed (peer0.org1.example.com and peer0.org2.example.com) so that chaincod
e is built before receiving the following requests
+ docker exec -e CORE_PEER_LOCALMSPID=Org1MSP -e CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrg
anizations/org1.example.com/users/Admin@org1.example.com/msp cli peer chaincode invoke -o orderer.example.com:7050 -C mychannel -n cashflo
w -c '{"function":"initLedger","Args":[]}' --waitForEvent --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/orderer
Organizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem --peerAddresses peer0.org1.example.com:70
51 --peerAddresses peer0.org2.example.com:9051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizati
ons/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt
```

USE CASE

Benefits

- Make cash flow in large-scale projects more transparent
- Detect cash leaks and suspicious manipulations
- Deliver starting point for in-depth auditioning

USE CASE

Limitations

- Preventing criminal activity all by itself
- Working in an entirely corrupted environment
 - i.e. where none of the parties are genuine

OUTLOOK

- Store digital copies of legal agreements in adjacent database
- Legal agreements between contractors
- Graphical User Interface (GUI)
- Issue and react to events

Thank you!

22nd January 2020

Oleksii Kulikov • Lukas Schöbel

Technical University of Munich

Chair for Application and Middleware Systems



SOURCES

Bibliography

- [1] Christian Gorenflo, Stephen Lee, Lukasz Golab, and Srinivasan Keshav. 2019. FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second. (2019), 455–463.

Images

- [A] <https://www.hyperledger.org/>
- [B] *Hyperledger Fabric*, F. Matthes, Lecture Nr. 12 on Blockchain-based Systems Engineering (IN2359), 2019, p. 11-20.
- [C] *Hyperledger - Introduction*, F. Matthes, Lecture Nr. 11 on Blockchain-based Systems Engineering (IN2359), 2019, p. 32.
- [D] https://www.berlin-airport.de/_images/presse/mediathek/_fotos/real-estate/06-airport-city-simulation.jpg