



Bringing custom sidechain

Distributed and decentralized systems in theory and practice

ZUEV EGOR



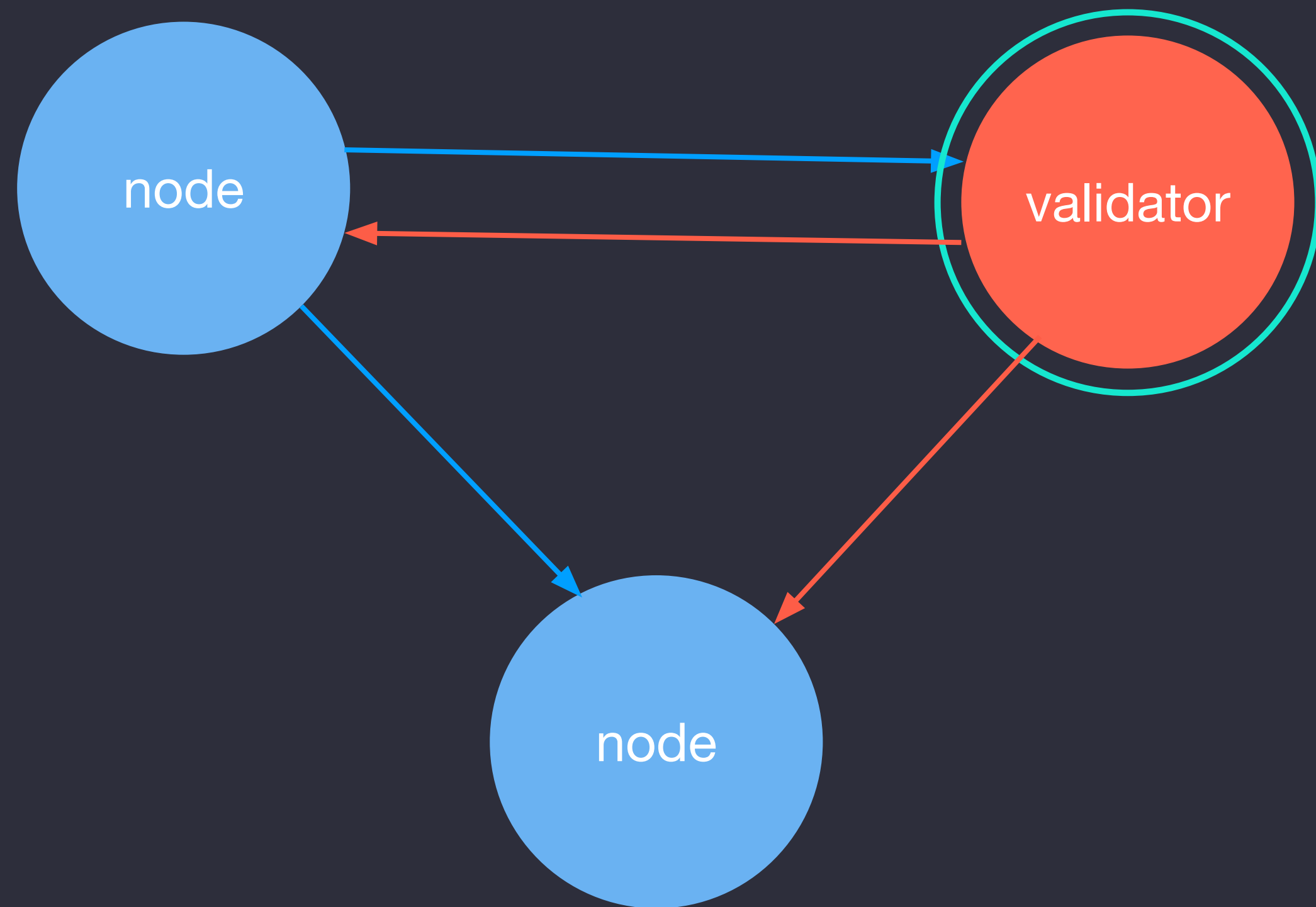
Blockchain

Blockchain – is immutable storage / ledger with decentralized peers and own consensus rules. All changes are made through the transactions.





Append new transaction



- 1 Client broadcast the tx**
Client broadcast the unconfirmed tx to all known peers. No guarantees, that leader will receive it first.
- 2 Validator validate the tx**
Validator validate the tx alongside with other txs in pool. Then form block by timer or by size.
- 3 Validator mint block**
Validator mint block and replicate it over all known peers.



Time issues

Which factors can impact on new transaction to be included in block?

1 Latency

Delays between tx has been broadcasted and validator received it

2 Fees

Higher fee – higher chances to be included in next block

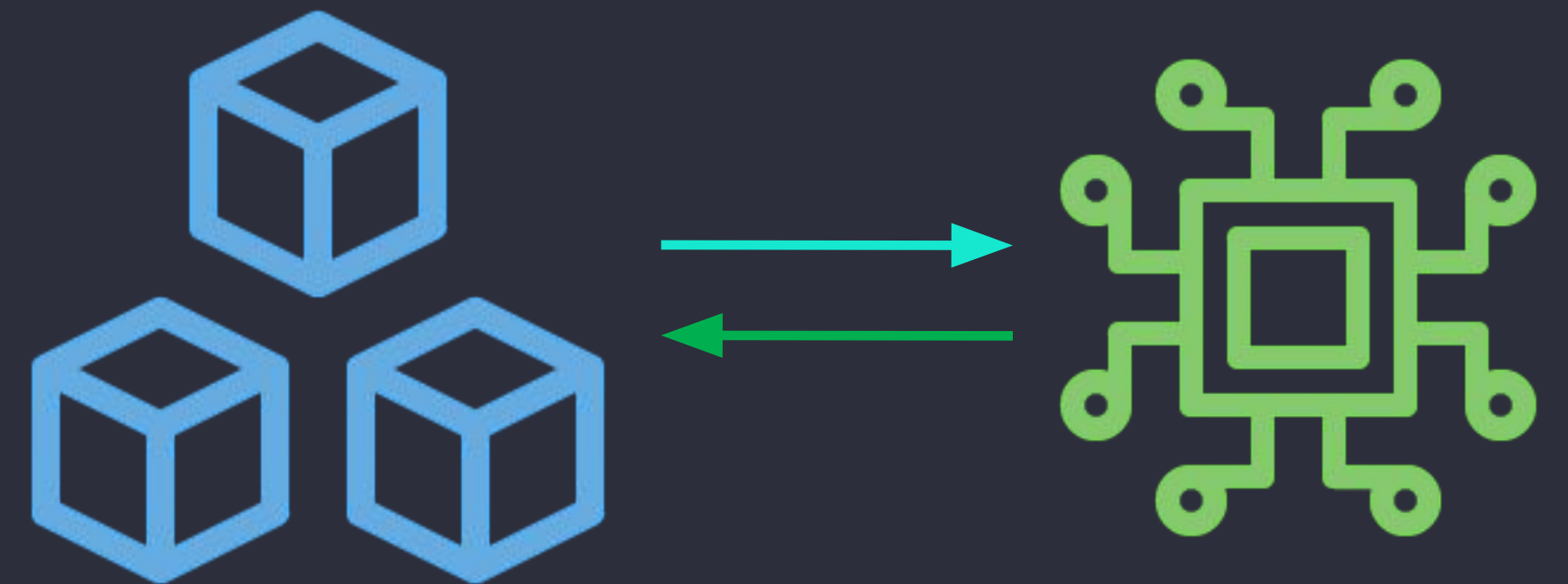
3 Block mint delay

We have to wait, until blockchain will validate enough txs, or timer will be triggered



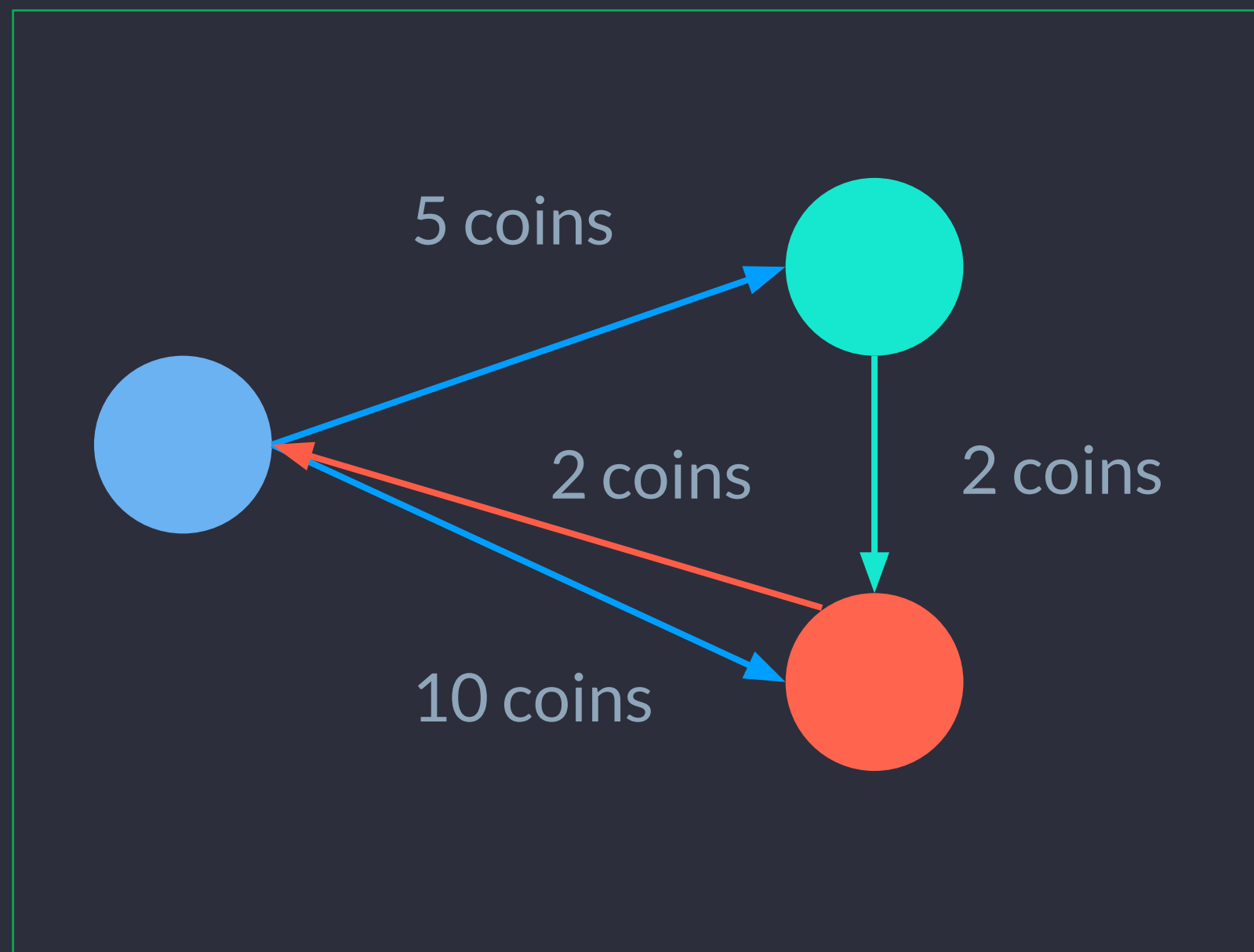
Off-chain

Off-chain is a pattern, where some operations, related to blockchain, are performed out of blockchain on custom platform





Off-chain concept



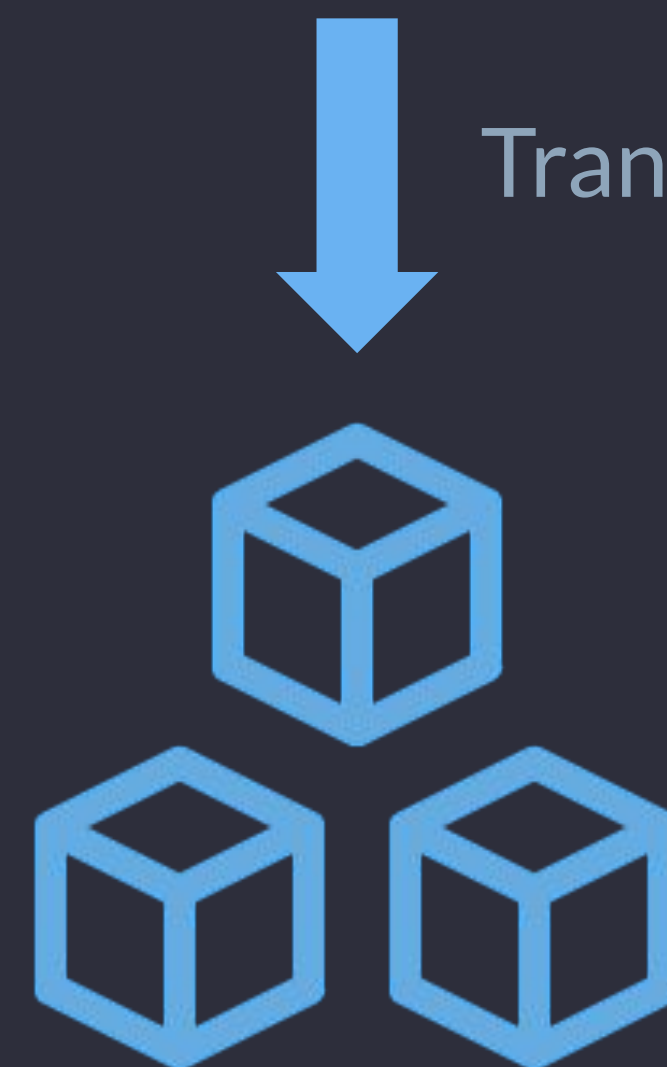
DELTA

Alice: -13 coins

Bob: +3 coins

Mike +10 coins

Transaction



1 Alice

TX1: Alice->Bob (5 coins)

TX2 Alice-> Mike (5 coins)

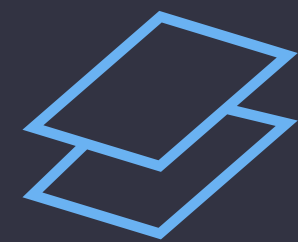
2 Bob

TX1: Bob->Mike (2 coins)

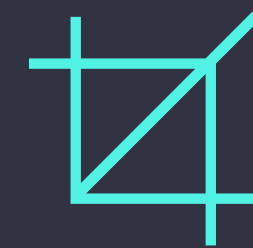
3 Mike

TX1: Mike->Alice (2 coins)

Off-chain implementations



General backend

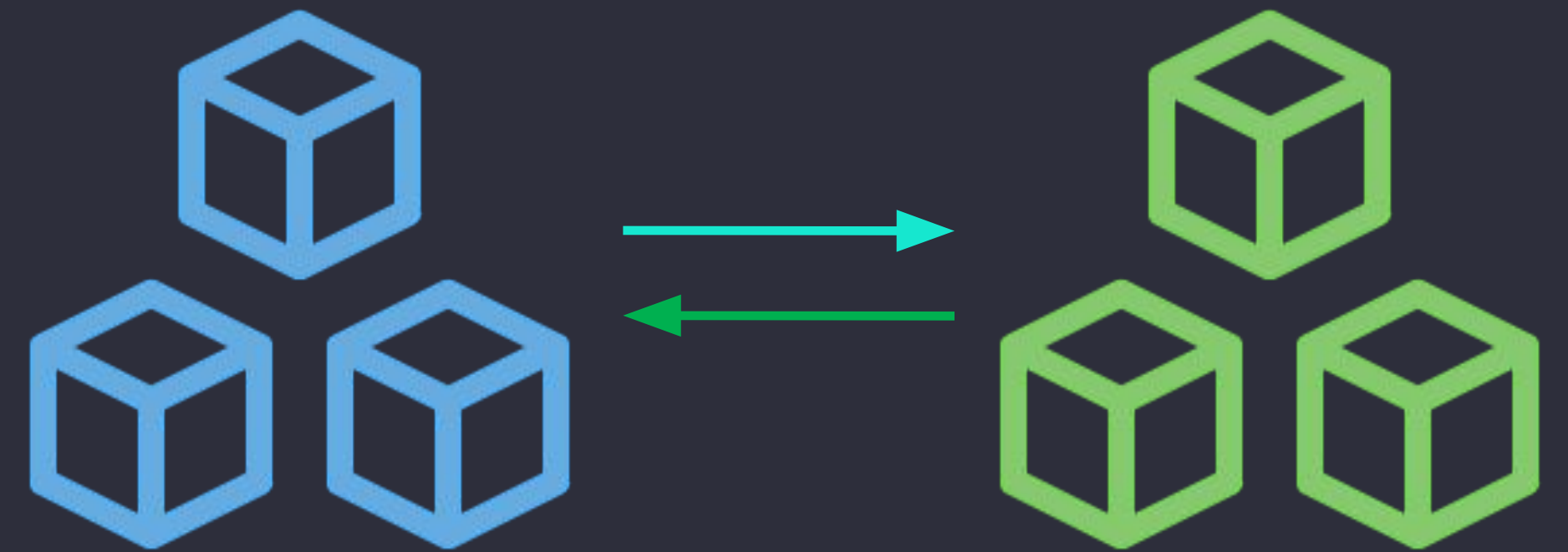


sidechain



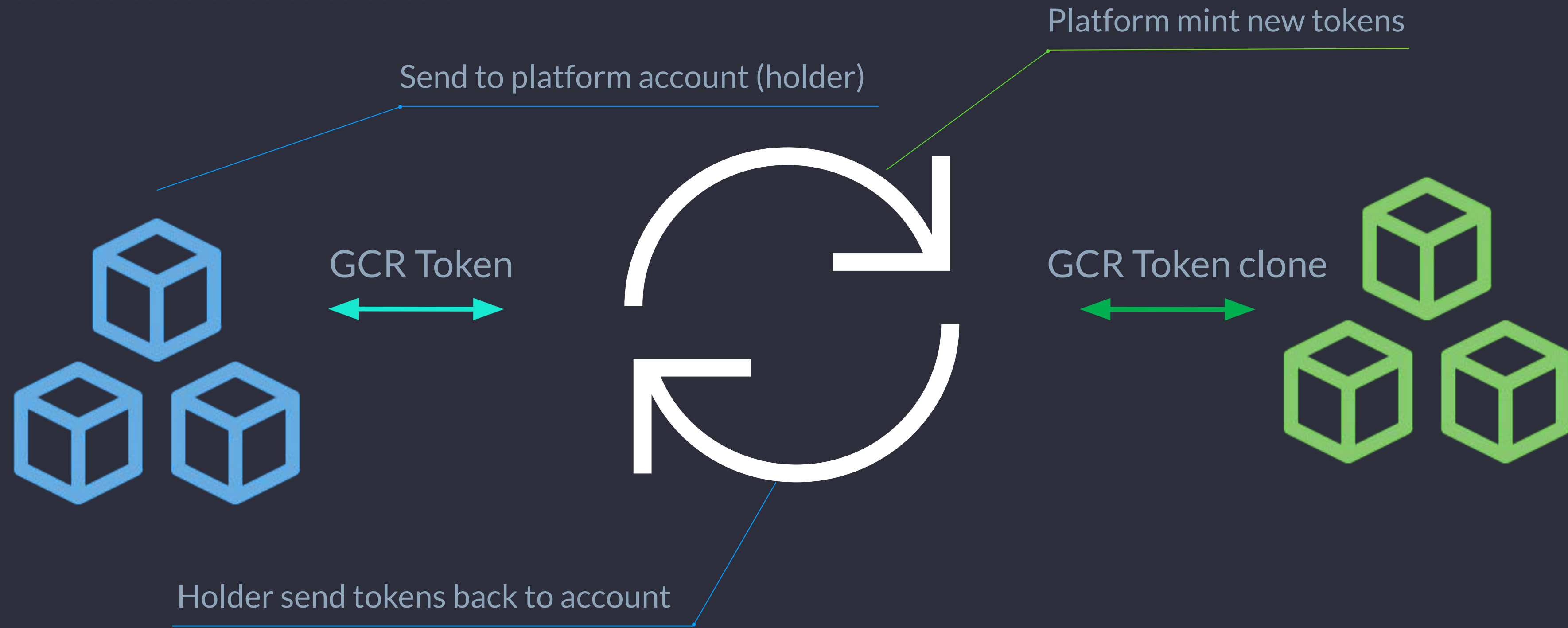
Sidechain

Sidechain is a pattern, which allows to use certain blockchain tokens on another blockchain / platform with an ability to move these tokens to original chain





Sidechain



Cluster components



State machine

Properties

Run all rules against tx

Form global state of system



Consensus Engine

Properties

Replicate all data across the cluster

Take care of collisions



State machine

State machine with
Ganache



TRUFFLE

1 Full EVM support

Full EVM machine, as on Parity and Geth clients

2 Integrated client

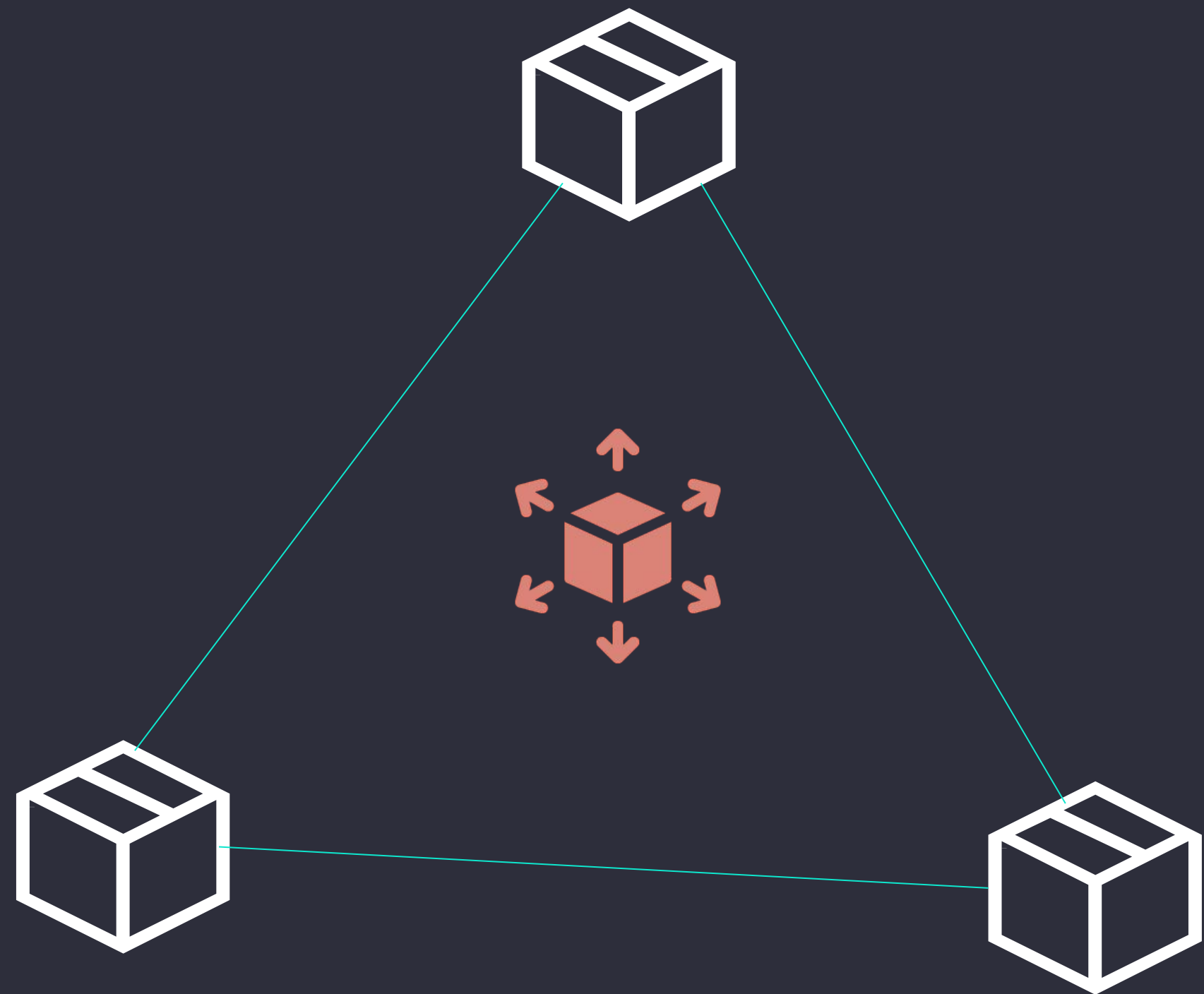
Can connect via web3

3 Quick block minting

Mint block per each performed action



Consensus Mokka



1 CP algorithm

MOKKA use single leadership model for making changes to RSM, however each follower may hold unconfirmed changes and replicate them (via gossip protocol).

2 Sync system

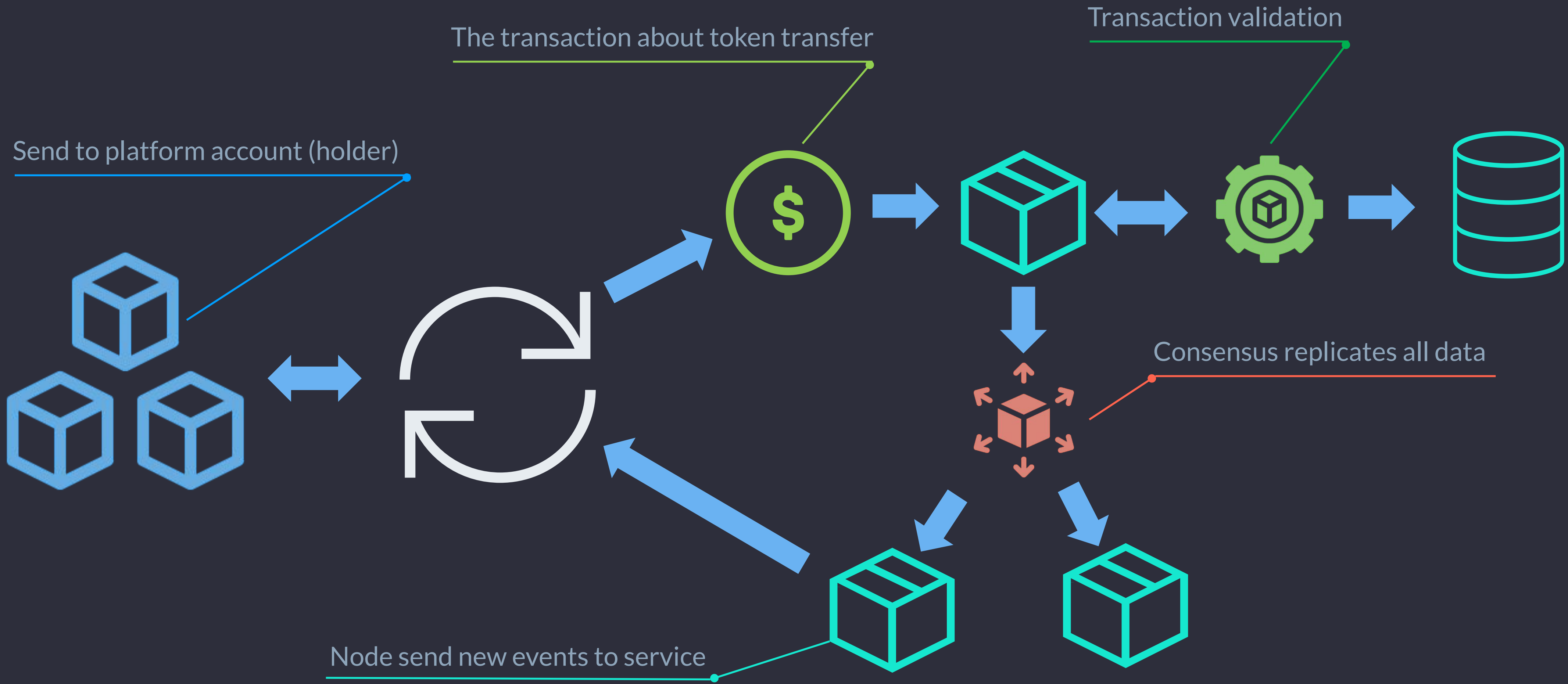
MOKKA use internal timers for checking latency and timeouts (strategy nested from RAFT).

3 Performance

Due to single leadership model, MOKKA appends changes quite slow, as all of these changes have to be processed by single leader first.

Workflow

W



Real example

Distributed ganache



<https://github.com/ega-forever/mokka/tree/master/examples/node/decentralized-ganache>

Опрос

<https://otus.ru/polls/6416/>



Thanks for listening!

Distributed and decentralized systems in theory and practice