Hyperledger Iroha Workshop

Hyperledger Bootcamp

Hong Kong, March 7, 2019

Vadim Reutskiy

Agenda

1. Few words about Soramitsu
2. Overall description of Iroha
3. Possible use cases
4. Workshop case description

Contacts

Vadim Reutskiy

- Project lead in Japanese office
- Email: reutskiy@soramitsu.co.jp
- Telegram: @vreutskiy

Scan me

About Soramitsu

ソラミツ

- Fintech software development company
- Founded in 2016 in Tokyo, Japan
- 55+ employees

- Worldwide locations:
  - Tokyo (Japan) **JP**
  - Innopolis (Russia) **RU**
  - Phnom Penh (Cambodia) **KH**
  - Astana (Kazakhstan) **KZ**
  - Zug (Switzerland) **CH***

***In progress**

Who are we?

SORAMITSU ソラミツ

HYPERLEDGER IROHA

Creator of Hyperledger Iroha and an active member of the Linux Foundation's Hyperledger Project

NATIONAL BANK OF CAMBODIA

We are creating a payment system based on Hyperledger Iroha for the central bank and regulator of the Kingdom of Cambodia

JBA
Japan Blockchain Association

We are a proud member of the Japan Blockchain Association

Key features of Iroha

- Command-driven architecture
  - Asset management
  - Identity management
- Support of linux, macOS, Windows environment
- Byzantine fault-tolerant ordering service and consensus
- Role-based access control
- Client libraries, including example apps for iOS, JS (Vue.JS), Android (Java 8)
- Universal peer role and easy scripted deployment with Docker and Ansible
- Multi-signature transactions

Why Iroha?

- Distributed ledger platform for simple use-cases of payments and identity storage.
- Uses fixed set of commands 16 in total (e.g. asset creation, transfers, account creation) and 11 queries (e.g. get account detail, get account balance) in its client API layer.
- Has modular design of its core components: storage, consensus, etc. which allows fundamental changes and opens up possibilities for contribution.

Why Iroha?

- Unique consensus and ordering service algorithms compared to other platforms with BFT class reliability.
- Has modern C++ design which lowers maintenance efforts and increases simplicity of use for API consumers (designers of blockchain-powered applications).
- Successful proof of concept projects include: cross-chain exchange, bank settlements, asset tokenization in the blockchain, digital identity, and supply chain scenarios.

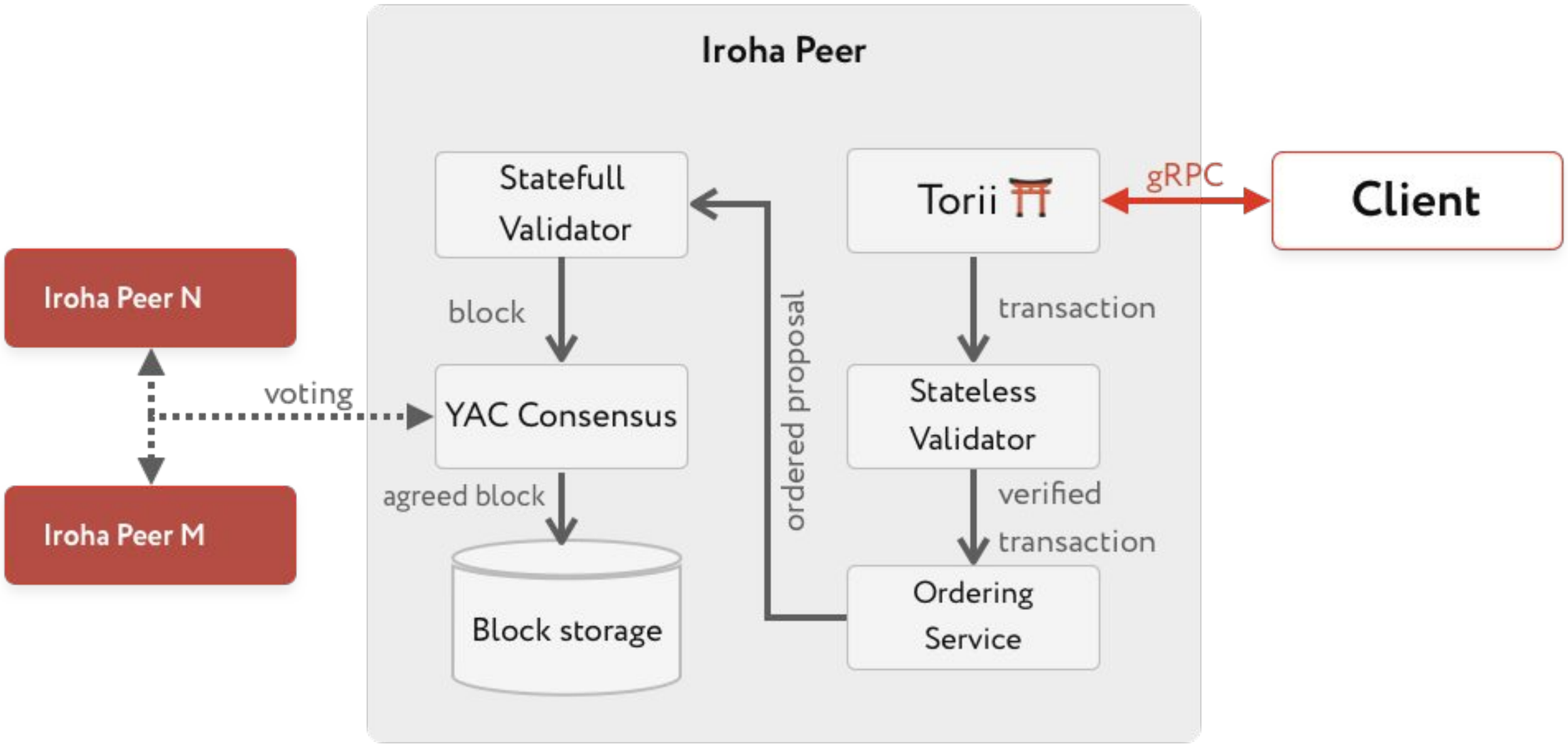
Comparison of DLT

Factor per platform	Hyperledger Fabric (and IBM blockchain)	Hyperledger Iroha	Hyperledger Sawtooth
Regional awareness	China! and the rest of the world	Asia, especially Japan	USA
Differentiators	Extendable deployment architecture, «channels»	Universal peer role, SQL state, linearly scalable consensus	Transaction processors, pluggable components
Is this a blockchain?	Yes (although it stores invalid transactions)	Yes	Yes
API	gRPC & REST	gRPC	gRPC
Business logic layer	Smart contracts in Go, Java & Solidity	Commands and queries	Transaction families and processors
Contributing companies	IBM	Soramitsu	Intel
BFT	—	+	+?
In production?	+	+-	+?

Comparison of DLT

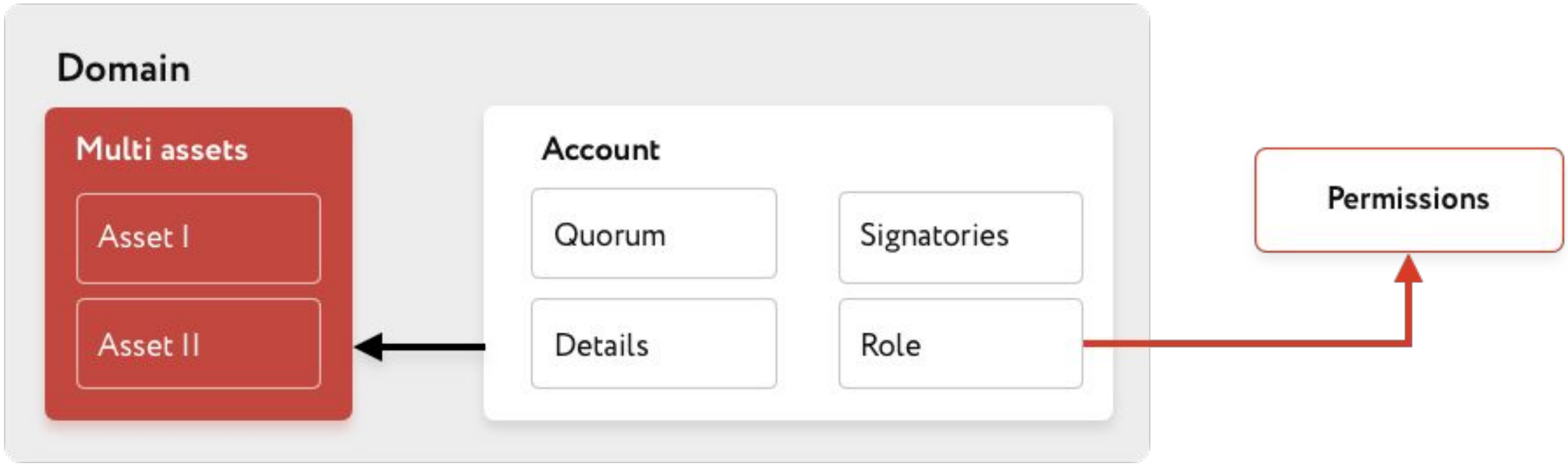
Factor per platform	Corda	Hyperledger Iroha	Ethereum
Regional awareness	UK, India, USA	Asia, especially Japan	The world
Differentiators	Scalability	Universal peer role, SQL state, linearly scalable consensus	Turing-complete smart contacts, same codebase for public and private
Is this a blockchain?	No	Yes	Yes
API	JSON-RPC?	gRPC	JSON-RPC
Business logic layer	Transactions processors in Kotlin?	Commands and queries	Solidity smart contracts
Contributing companies	R3	Soramitsu	Ethereum foundation
BFT	—	+	—
In production?	+	+-	+?

Iroha Architecture



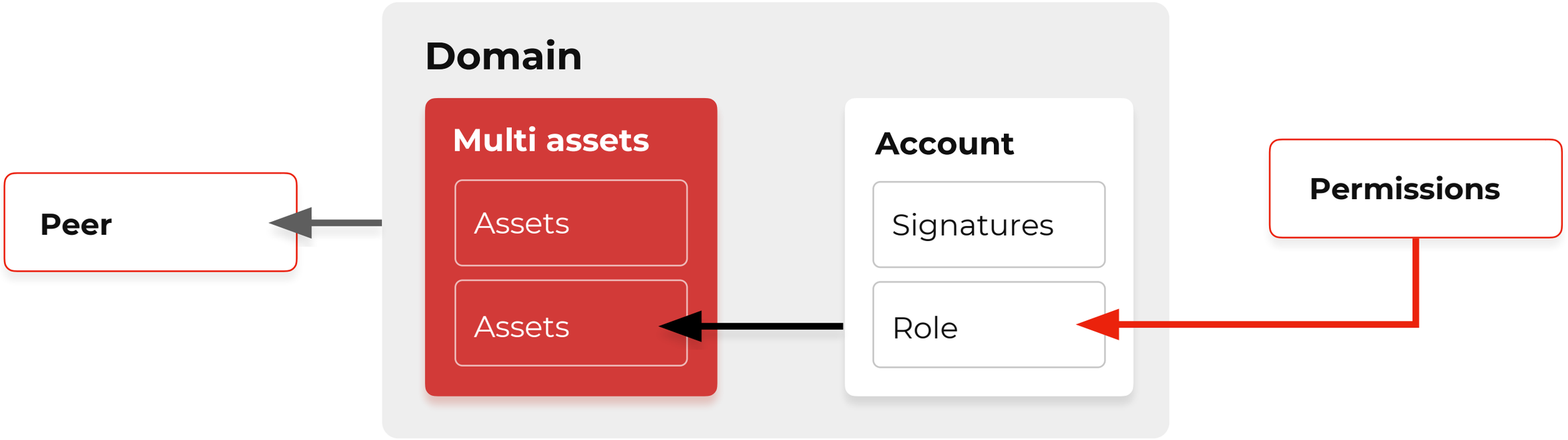
Data Structures in the Iroha

- Every user (or entity) has it’s own account: **{account_name}@{domain}**
- Every account can have several types of assets on board: **{asset_name}#{domain}**
- Every account has list of Roles with corresponding Permissions
- Every account can have more than one Signatories to perform multi-signature transactions



Commands & Queries in Hyperledger Iroha

Without writing code, asset, identity & supply chain management can be done using prepared commands in the data model .
This eases development and increases reliability.



Commands in Hyperledger Iroha

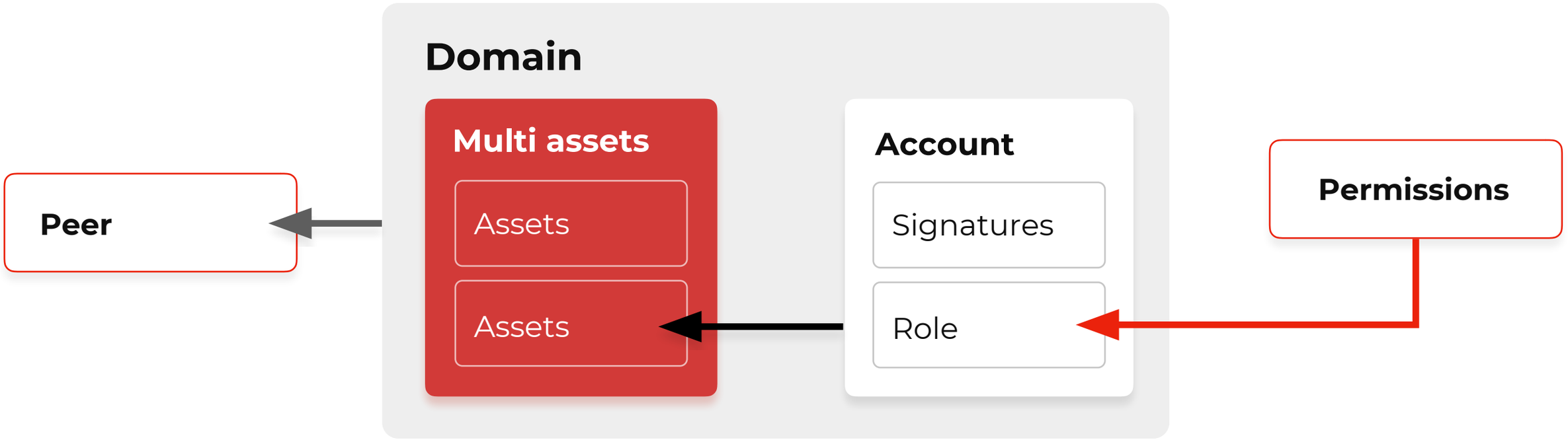
Peer
AddPeer

Domains
CreateDomain

Assets
CreateAsset
AddAssetQuantity
SubtractAssetQuantity
TransferAsset

Account
CreateAccount
AddSignatory
RemoveSignatory
SetAccountQuorum
SetAccountDetail

Permissions
CreateRole
AppendRole
DetachRole
GrantPermission
RevokePermission



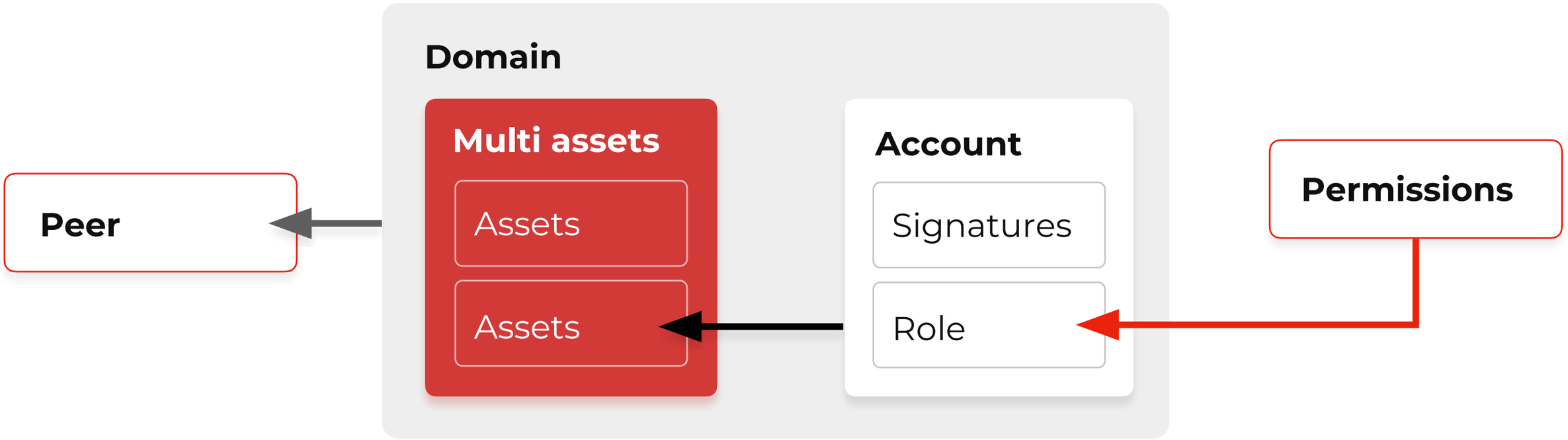
Queries in Hyperledger Iroha

Account
GetAccount
GetAccountAssets
GetAccountDetail
GetSignatories

Transactions
GetTransactions
GetAccountTransactions
GetAccountAssetTransactions
GetPendingTransactions

Assets
GetAssetInfo

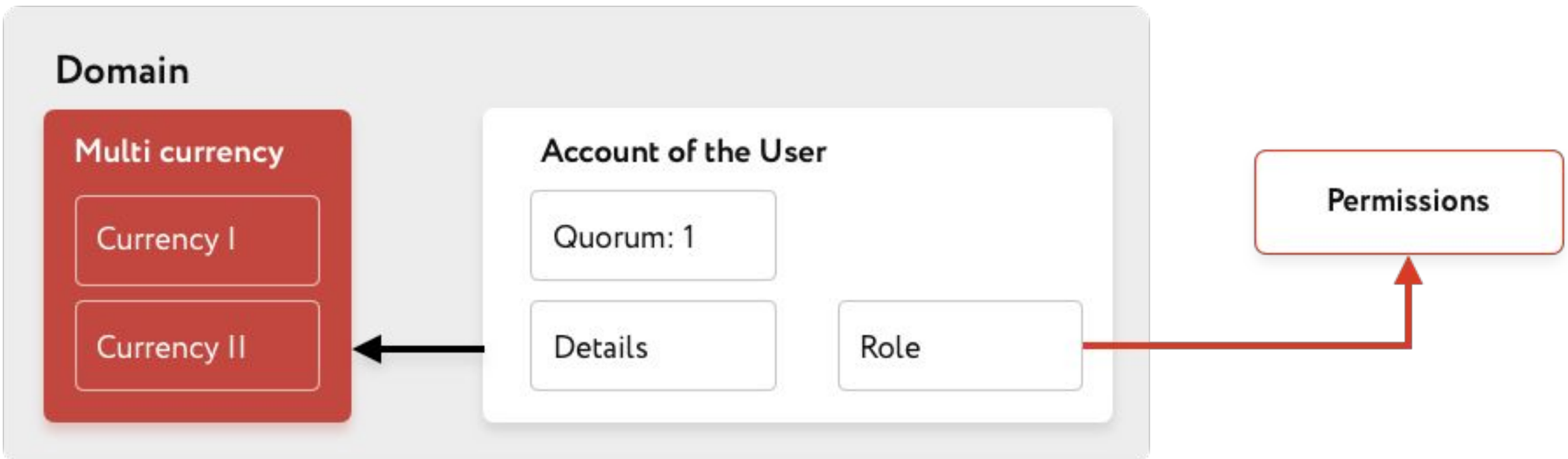
Permissions
GetRoles
GetRolePermissions



Use Case I: Digital Currency

The main use case for the blockchain based system.

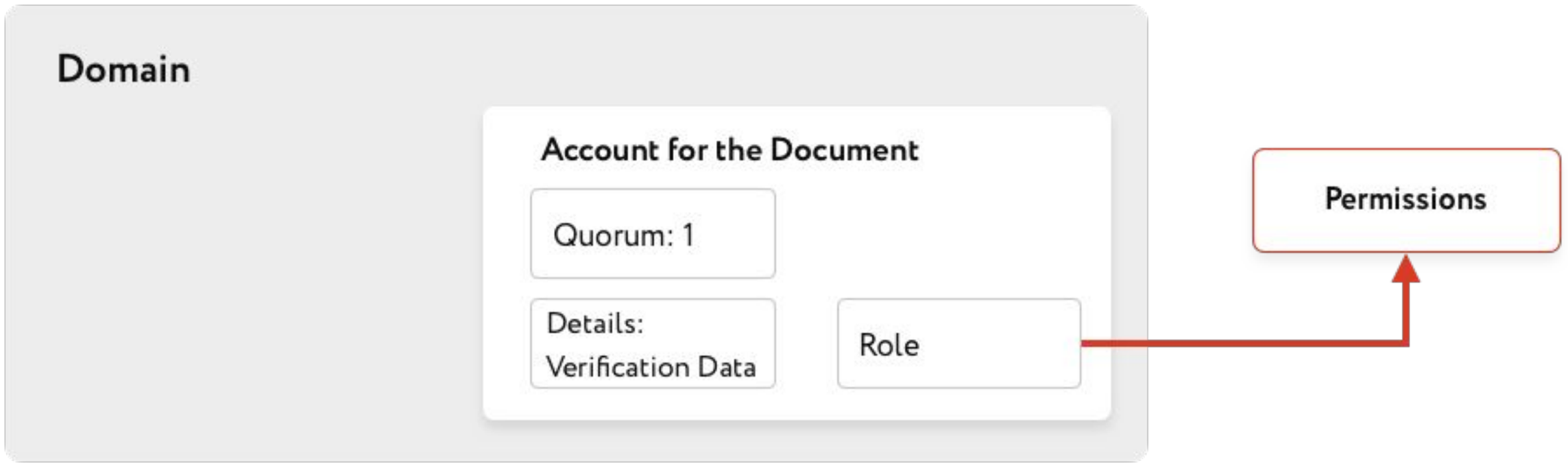
- **Account** corresponds to the **physical user** of the system
- **Asset** corresponds to single **digital currency**
- **Domain** corresponds to particular **Bank or Institution**
- **Quorum** for every account have value **1**



Use Case II: Verifiable Claims

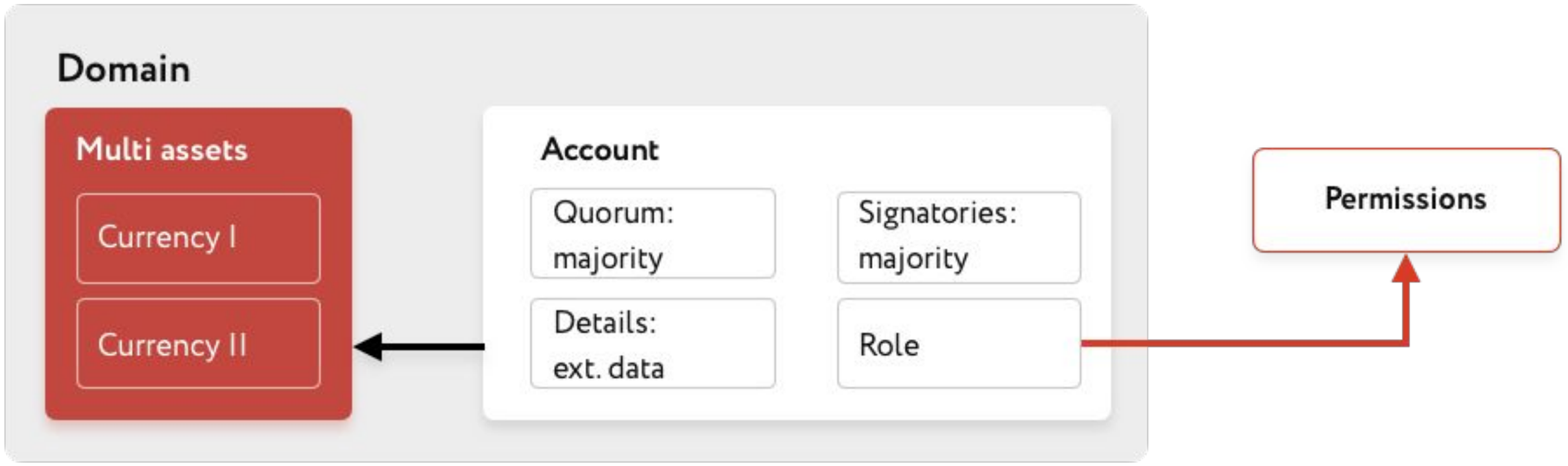
Provides trusted distributed source of truth for document verification

- **Account** corresponds to the particular **document**
- **Account details** contains **verification information**
- **Domain** corresponds to the particular **institution**

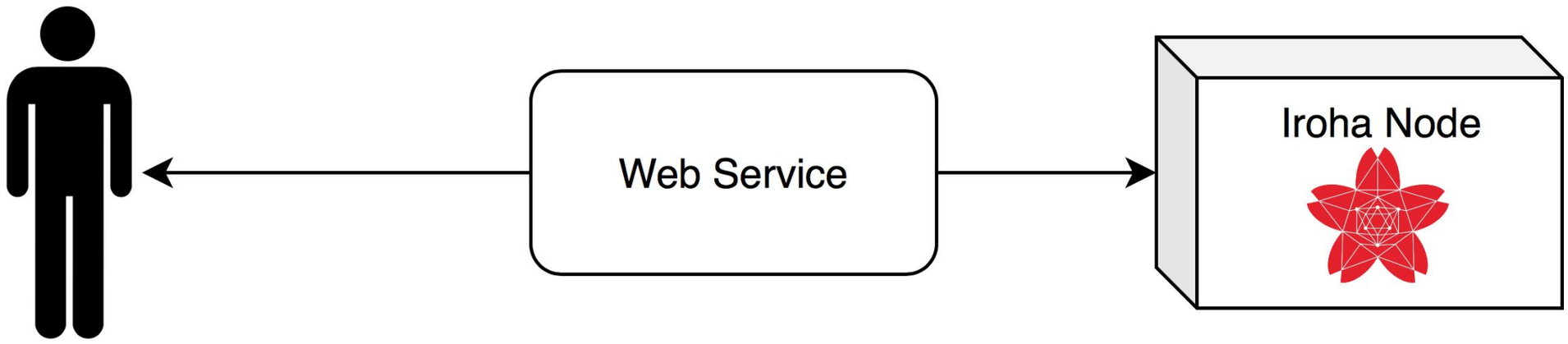


Use Case III: Decentralized Depository

- **Distributed network of trusted digital currency exchange**
- **Account** corresponds to the registered user
- **Account details** keeps important information about linked external currencies and logic of synchronization
- **Quorum** is variable and corresponds to amount of nodes inside the system



System architecture for Workshop Example



During the workshop we will create a simple web service, which can be used as middleware between the client application and Ioha Network, as we always do in the real projects.

You can use any client library and any approach, which you prefer, but for simplicity and synchronization I recommend to use prepared wireframe for the Web Service on Kotlin, which uses Iroha Java Client Library.

Workshop steps recommended order

1. Start the Iroha node locally and perform several operations over it using CLI (by following "Getting Started" document).
2. Get familiar with the Client Java Library
3. Obtain the wireframe example web server
4. Implement needed functions by example from the Client Java Library
5. Check that everything works correctly using Postman tool

SORAMITSU
ソラミツ

Thank you for attention!
Now I am open for questions

What we propose

Learn Latest Trends in Fintech Industry

- What is permissoned Blockchain and how to use it
- Learn on real cases but in a safe environment

Open source contribution

- A contribution in known project
- Create web/mobile product based on Blockchain

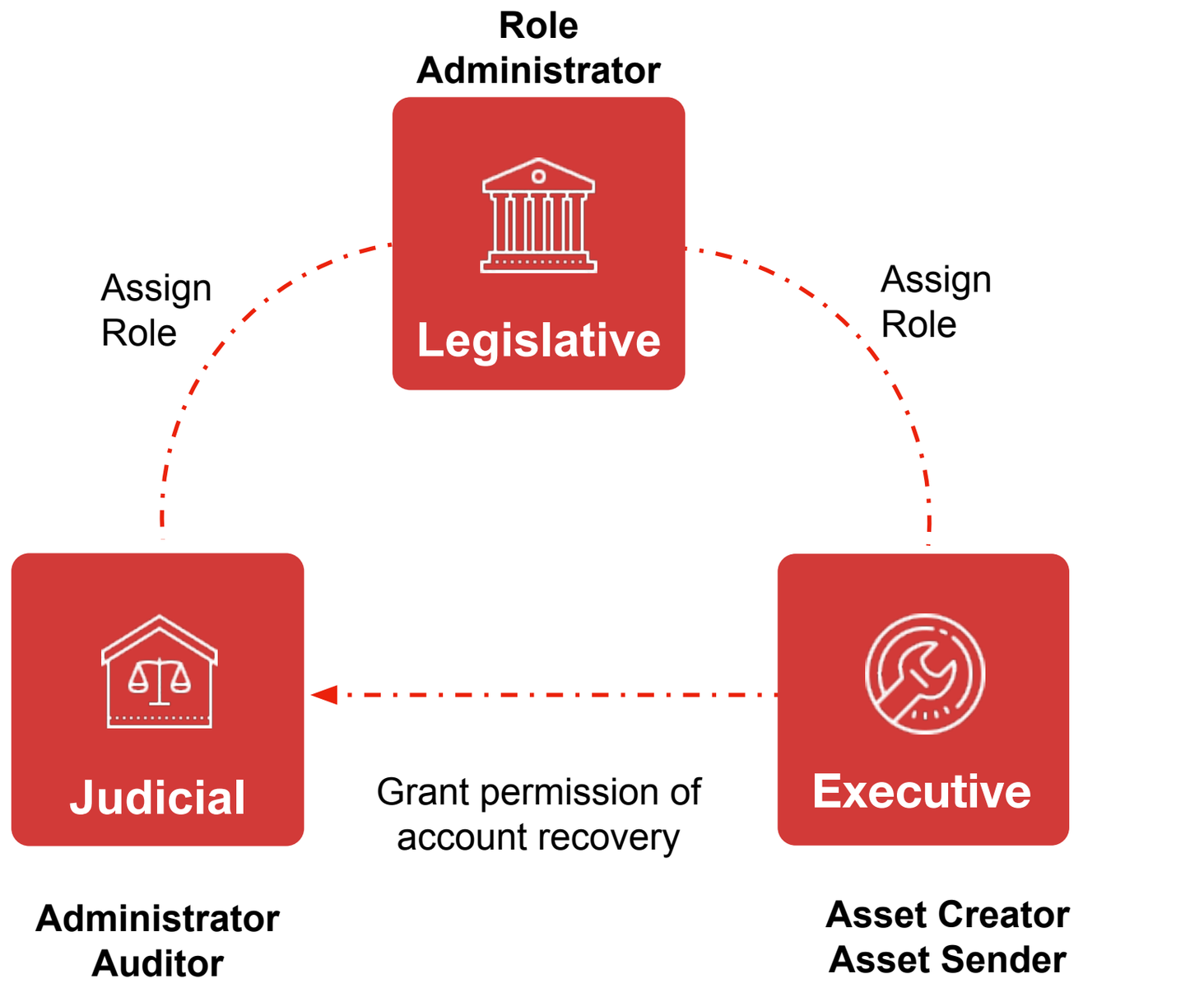
Project constraints

- Back: Any language thats supports Protobuf
- Front: VueJS, React
- Mobile: Native language
- Github / open source

Decentralized permission model

Example of Role Decentralization

- Decentralized RBAC* permission model without single point failure
- Separation of three powers can be created to avoid concentration of authority
- Roles and permissions are set determined in the genesis block



*RBAC=Role Base Access Control

Comparison of DLT Platforms

Hyperledger Fabric

Turing complete smart contracts

Supports privacy via “channels”

Supported by IBM

Corda

Supports payments and notarization of messages (but through centralized service)

Supports privacy via UTXO

Supported by R3

Hyperledger Iroha

Pre-defined smart contracts called “commands”

Supports privacy via permissions

SDKs for mobile apps

Supported by Soramitsu

Ethereum

Turing complete smart contracts

Private version has been tested by many banks around the world

Supported by the Ethereum open source community

Proposed technical solution

Shared distributed ledger

- Make a shared distributed ledger with some client information
- Each company only shares part of data to ensure that product is unique