Hyperledger Fabric

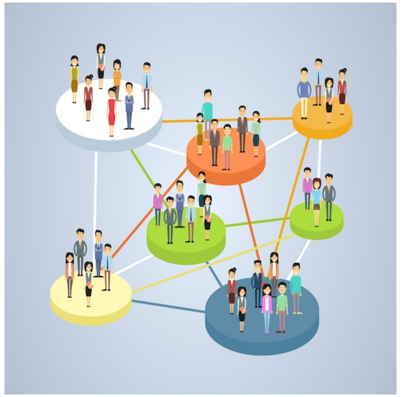
Theory and Applications of Blockchain (CS61065) - Tutorial 4

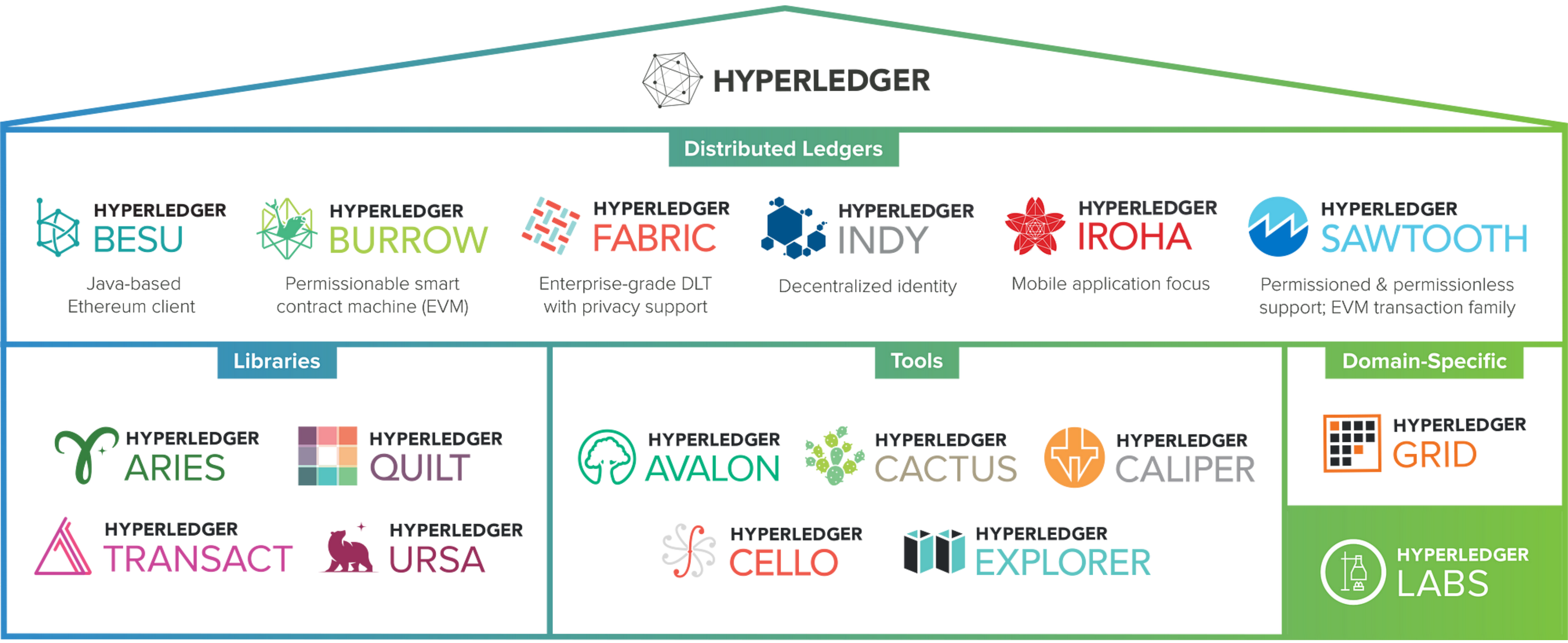
Public Blockchains

- Anyone can participate.
- Every node can see all the transactions in the network.

Use case

- The class is divided into various groups
- Each group has to collectively write some code
- If any person makes any changes, he only wants his group to see the changes
- No one from outside the class should be able to join without permission.
- Can we use a single Public Blockchain network here?




HYPERLEDGER
Distributed Ledgers
HYPERLEDGER BESU
Java-based Ethereum client
HYPERLEDGER BURROW
Permissionable smart contract machine (EVM)
HYPERLEDGER FABRIC
Enterprise-grade DLT with privacy support
HYPERLEDGER INDY
Decentralized identity
HYPERLEDGER IROHA
Mobile application focus
HYPERLEDGER SAWTOOTH
Permissioned & permissionless support; EVM transaction family
Libraries
HYPERLEDGER ARIES
HYPERLEDGER QUILT
HYPERLEDGER TRANSACT
HYPERLEDGER URSA
Tools
HYPERLEDGER AVALON
HYPERLEDGER CACTUS
HYPERLEDGER CALIPER
HYPERLEDGER CELLO
HYPERLEDGER EXPLORER
Domain-Specific
HYPERLEDGER GRID
HYPERLEDGER LABS

HYPERLEDGER

Distributed Ledgers

HYPERLEDGER BESU
Java-based Ethereum client

HYPERLEDGER BURROW
Permissionable smart contract machine (EVM)

HYPERLEDGER FABRIC
Enterprise-grade DLT with privacy support

HYPERLEDGER INDY
Decentralized identity

HYPERLEDGER IROHA
Mobile application focus

HYPERLEDGER SAWTOOTH
Permissioned & permissionless support; EVM transaction family

Libraries

HYPERLEDGER ARIES
HYPERLEDGER QUILT
HYPERLEDGER TRANSACT
HYPERLEDGER URSA

Tools

HYPERLEDGER AVALON
HYPERLEDGER CACTUS
HYPERLEDGER CALIPER
HYPERLEDGER CELLO
HYPERLEDGER EXPLORER

Domain-Specific

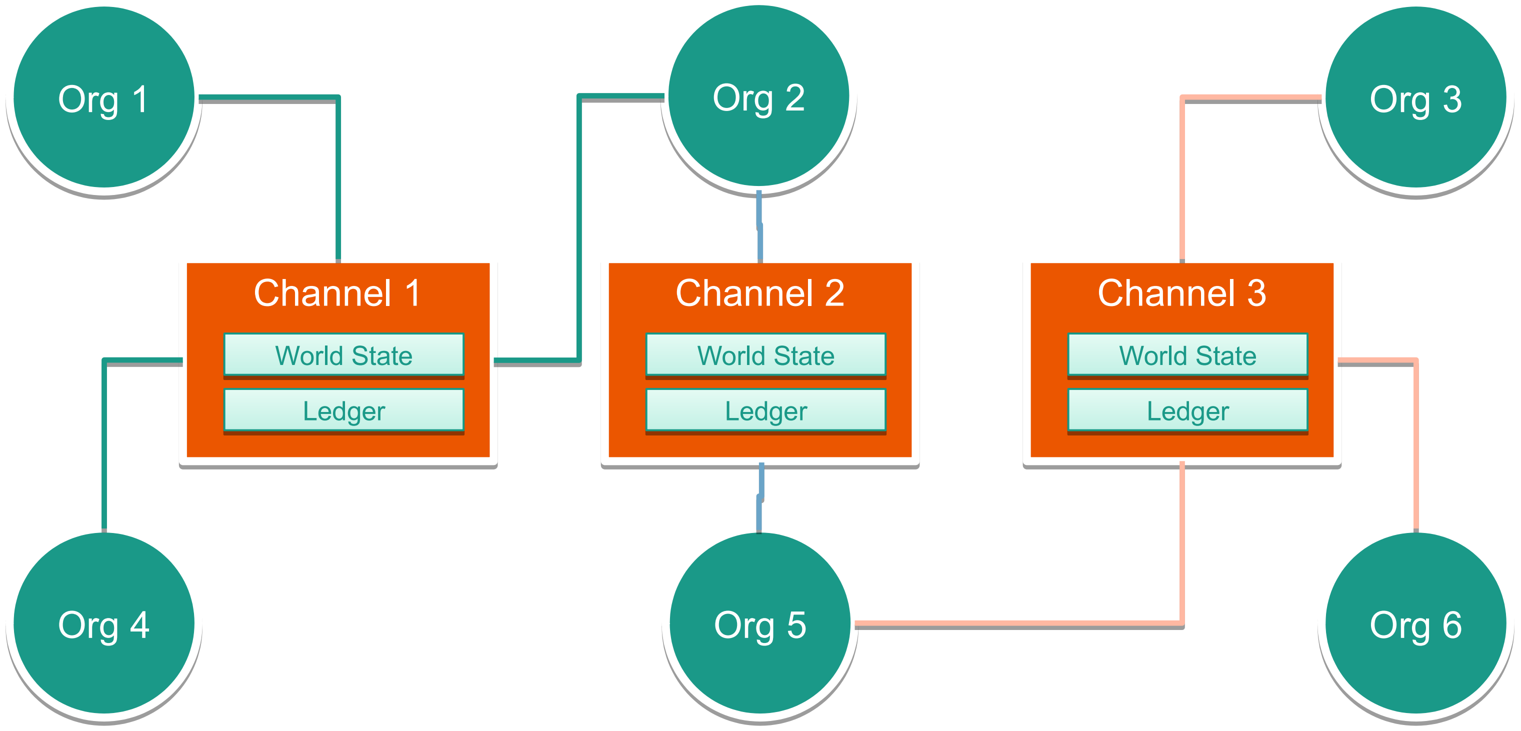
HYPERLEDGER GRID

HYPERLEDGER LABS

Hyperledger Fabric (Private Blockchain)

- Like other blockchain technologies, it has a ledger, uses smart contracts, and is a system by which participants manage their transactions.
- It is private and permissioned blockchain.
- Rather than an open permissionless system that allows unknown identities to participate in the network, the members of a Hyperledger Fabric network enroll through a trusted **Membership Service Provider (MSP)**.

Hyperledger Fabric Network



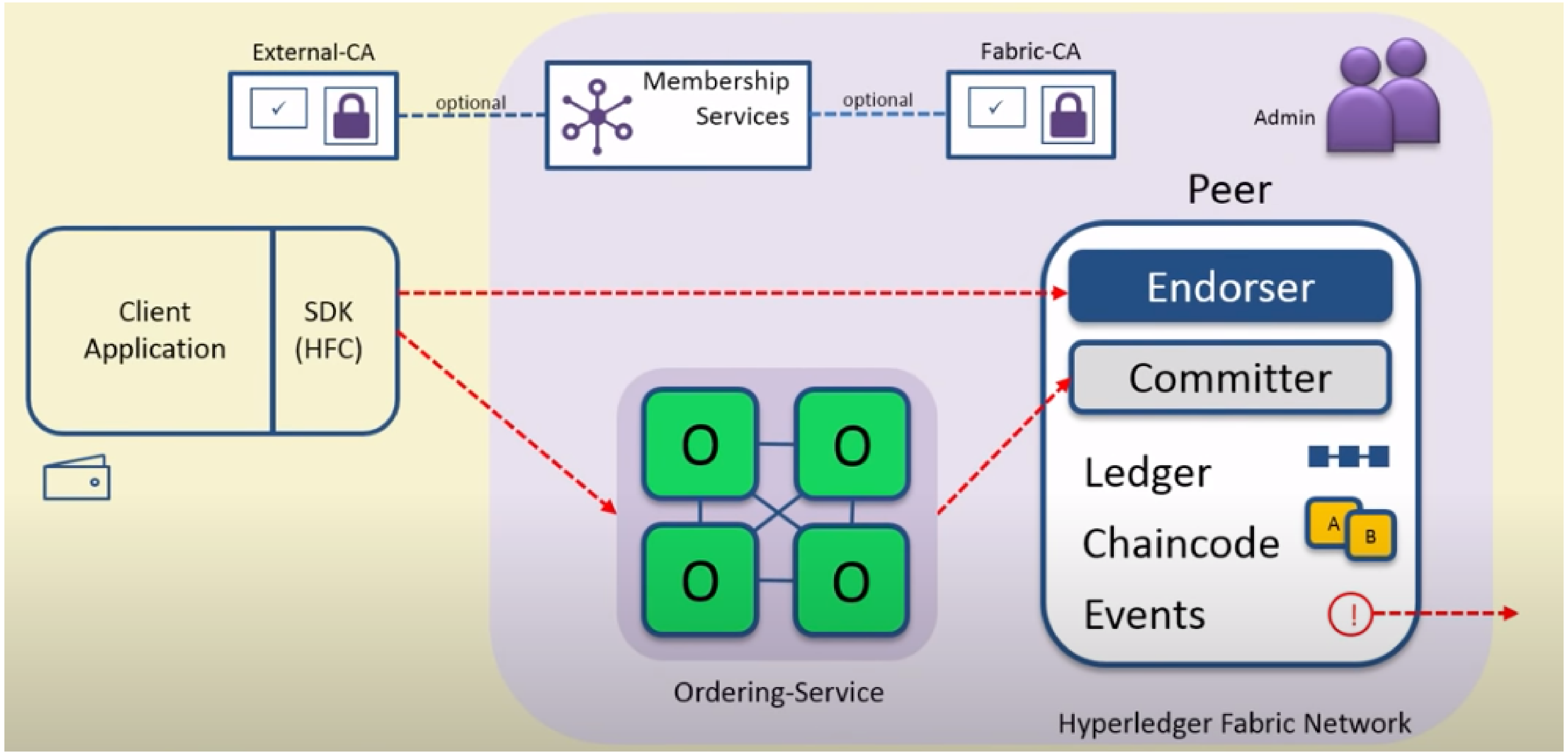
Organization

- Membership Service Providers (MSP) : Identity management of members
- Administrators : To manage the members (enroll, remove etc.)
- Users : Typically applications connecting to the blockchain network
- Peers : Hosting the chaincode and storing the ledger
- Orderers (optional) : Ordering of transactions

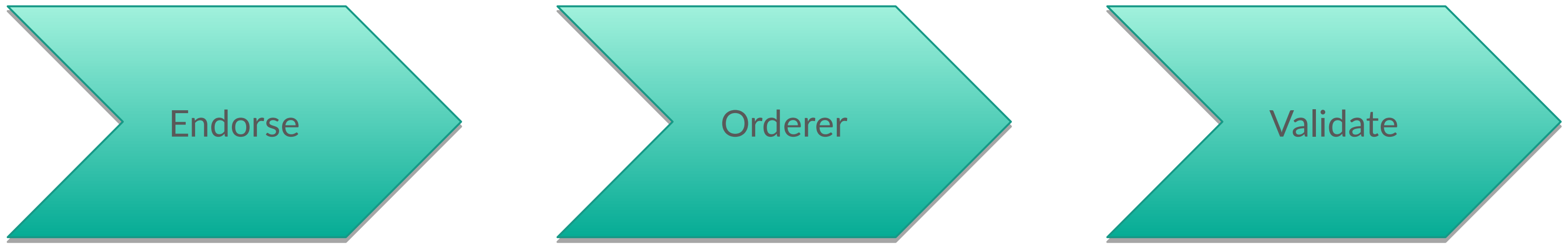
Each organization has an ID and a network can have multiple participating organizations

Image courtesy: NPTEL

Entities Involved



Flow of Transactions



Client sends the transaction proposal to endorser. The endorsers return results with their signatures. The client checks if the endorsement policy is being satisfied. If yes, it sends the transaction to the orderer.

The orderer simply orders the transactions, puts them into blocks and sends it to all the peers to be validated. It does not have to look at the transactions.

The peers validate the transactions. (Check for double spending, enforcement of the endorsement policy etc.)

Practical Use cases

- **Walmart and Food Traceability**
  - https://lf-hyperledger.atlassian.net/wiki/spaces/LMDWG/pages/18715588/Walmart

- **Circulor -Traceability of a Conflict Mineral (Tantalum)**
  - https://lf-hyperledger.atlassian.net/wiki/spaces/LMDWG/pages/18716226/Circulor+-Traceability+of+a+Conflict+Mineral+Tantalum

Setup prerequisites

Recommended to set up on a Linux based system

- Git https://git-scm.com/downloads
- CURL https://curl.se/download.html
- Docker Engine https://docs.docker.com/engine/install/
- NodeJS (recommend to use nvm (Node Version Manager))
- Docker Compose

Installing Hyperledger Fabric

Determine a location on your machine where you want to place the fabric-samples repository and enter that directory in a terminal window.

```
>> mkdir fabric

>> cd fabric

>> curl -sSLO https://raw.githubusercontent.com/hyperledger/fabric/main/scripts/install-fabric.sh &&
chmod +x install-fabric.sh

>> ./install-fabric.sh docker samples binary

>> export PATH=<path to download location>/fabric-samples/bin:$PATH
```

Starting the test network with CAs

```
>> cd fabric-samples/test-network

>> ./network.sh up -ca
```

The script will start containers for each member of the test-network namely:

- orderer
- peer0.org1.example.com
- peer0.org2.example.com
- ca_org1
- ca_org2
- ca_orderer
- cli

Creating a channel

```
>> ./network.sh createChannel -ca
```

- Creates a channel with name "mychannel"

```
>> ./network.sh createChannel -c <channel name> -ca
```

- To create a channel with a custom name

Deploying chain-code

>> ./network.sh deployCC -ccn <chain-code-name> –ccp <path-to-chaincode> -ccl <chaincode-language>

- To deploy chaincode

Example:
>> ./network.sh deployCC -ccn basic –ccp ../asset-transfer-basic/chaincode-javascript -ccl javascript

Other optional arguments:

- -ccep : To specify the endorsement policy

Running an application to invoke the chaincode

Example: Go to the asset-transfer-basic/application-javascript directory

The wallet directory stores the crypto material for a user issued by a CA. The CA is respawned every time we do ./network.sh up.

Before running app.js

- Remove the wallet directory before running app.js.
- Run **>> npm install**

To run the app.js file : >> node app.js

Things to remember

- **asset-transfer-events**

  While deploying the chaincode for this example, set the -ccep flag to “OR(‘Org1MSP.peer’, ‘Org2MSP.peer’)”

Example – asset transfer basic

- cd fabric-samples/test-network
- ./network.sh down
- ./network.sh up createChannel -ca -s couchdb
- ./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-javascript/ -ccl javascript
- cd ../asset-transfer-basic/application-gateway-javascript/
- npm install
- npm start
- cd ../../test-network
- ./network.sh down

Example – asset transfer events

- cd fabric-samples/test-network
- ./network.sh down
- ./network.sh up createChannel -ca -s couchdb
- ./network.sh deployCC -ccn events -ccp ../asset-transfer-events/chaincode-javascript/ -ccl javascript -ccep "OR('Org1MSP.peer','Org2MSP.peer')"
- cd ../asset-transfer-events/application-gateway-typescript/
- npm install
- npm start
- cd ../../test-network
- ./network.sh down

References

- Hyperledger Fabric Documentation
- Endorsement policies — hyperledger-fabricdocs main documentation
- Private data — hyperledger-fabricdocs main documentation
- Private Data — hyperledger-fabricdocs main documentation - Implicit private data
- Hyperledger Fabric SDK for Node.js Interface: Network - BlockEventListeners
- Hyperledger Fabric SDK for Node.js Interface: Contract - ContractEventListeners

NPTEL links:

- https://youtu.be/4ujj5knD3pg?si=joLsrcNJDv1l7EMk : Fabric Membership and Identity Management
- https://youtu.be/xjliVItyLRk?si=IRReh8BBsUDSnWm- : Fabric Components details
- https://youtu.be/nBXr7dLXAbE?si=x4vaXCHnM_nLyOGs : Fabric Transaction flow