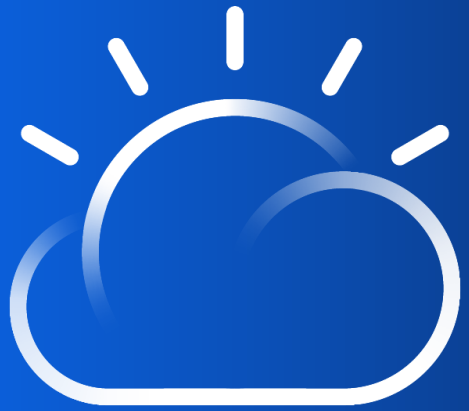
Develop your First Smart Contract with Hyperledger Composer

—

Carlos Rischioto
Blockchain Technical Leader

Tito Garrido Ogando
Blockchain Technical Consultant



IBM Cloud

Hyperledger Fabric Architecture

Hyperledger Fabric Technical Deep Dive

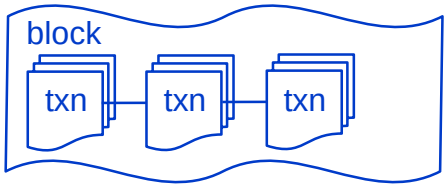
Hyperledger Fabric
Architecture

Hyperledger Fabric
Technical Deep Dive

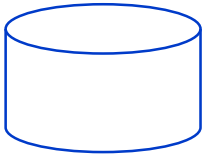
Components in a blockchain solution

Ledger	A ledger is a channel's chain and current state data which is maintained by each peer on the channel.
Smart Contract	Software running on a ledger, to encode assets and the transaction instructions (business logic) for modifying the assets.
Peer Network	A broader term overarching the entire transactional flow, which serves to generate an agreement on the order and to confirm the correctness of the set of transactions constituting a block.
Membership	Membership Services authenticates, authorizes, and manages identities on a permissioned blockchain network.
Events	Creates notifications of significant operations on the blockchain (e.g. a new block), as well as notifications related to smart contracts.
Systems Management	Provides the ability to create, change and monitor blockchain components
Wallet	Securely manages a user's security credentials
Systems Integration	Responsible for integrating Blockchain bi-directionally with external systems. Not part of blockchain, but used with it.

A ledger often consists of two data structures



Blockchain

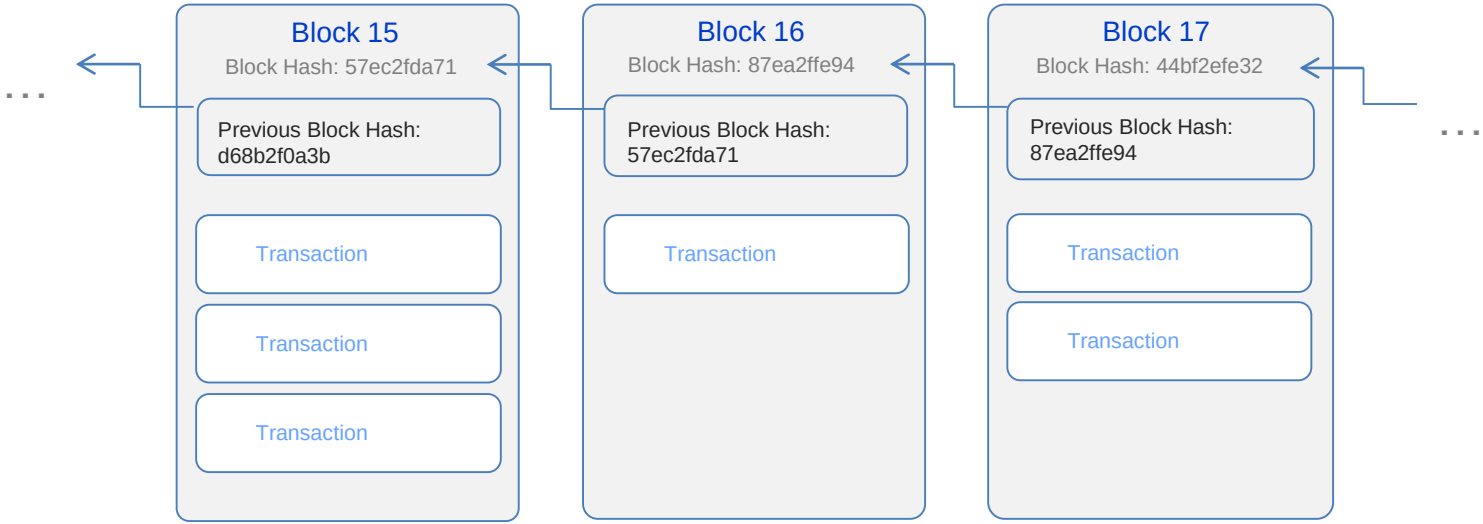


World state

- Blockchain
  - A linked list of blocks
  - Each block describes a set of transactions (e.g. the inputs to a smart contract invocation)
  - Immutable – blocks cannot be tampered

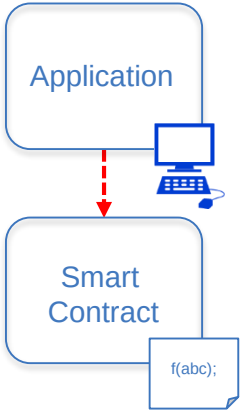
- World State
  - An ordinary database (e.g. key/value store)
  - Stores the combined outputs of all transactions
  - Not usually immutable

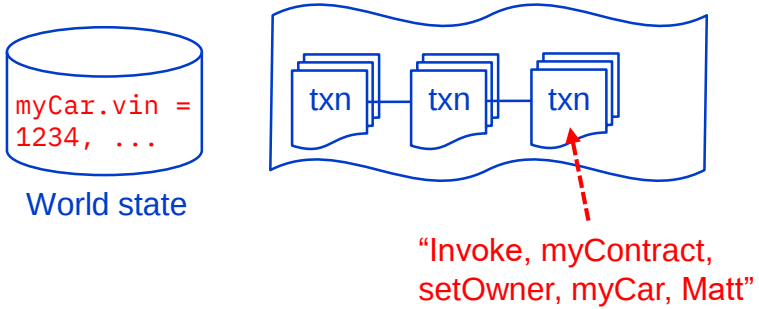
Block detail (simplified)



- A blockchain is made up of a series of blocks with new blocks always added to the end
- Each block contains zero or more transactions and some additional metadata
- Blocks achieve immutability by including the result of a hash function of the previous block
- The first block is known as the “genesis” block

Working with the ledger example: a change of ownership transaction





Transaction input - sent from application

```
invoke(myContract, setOwner,
        myCar, Matt)
…
```

Smart contract implementation

```
setOwner(Car, newOwner) {
   set Car.owner = newOwner
}
```

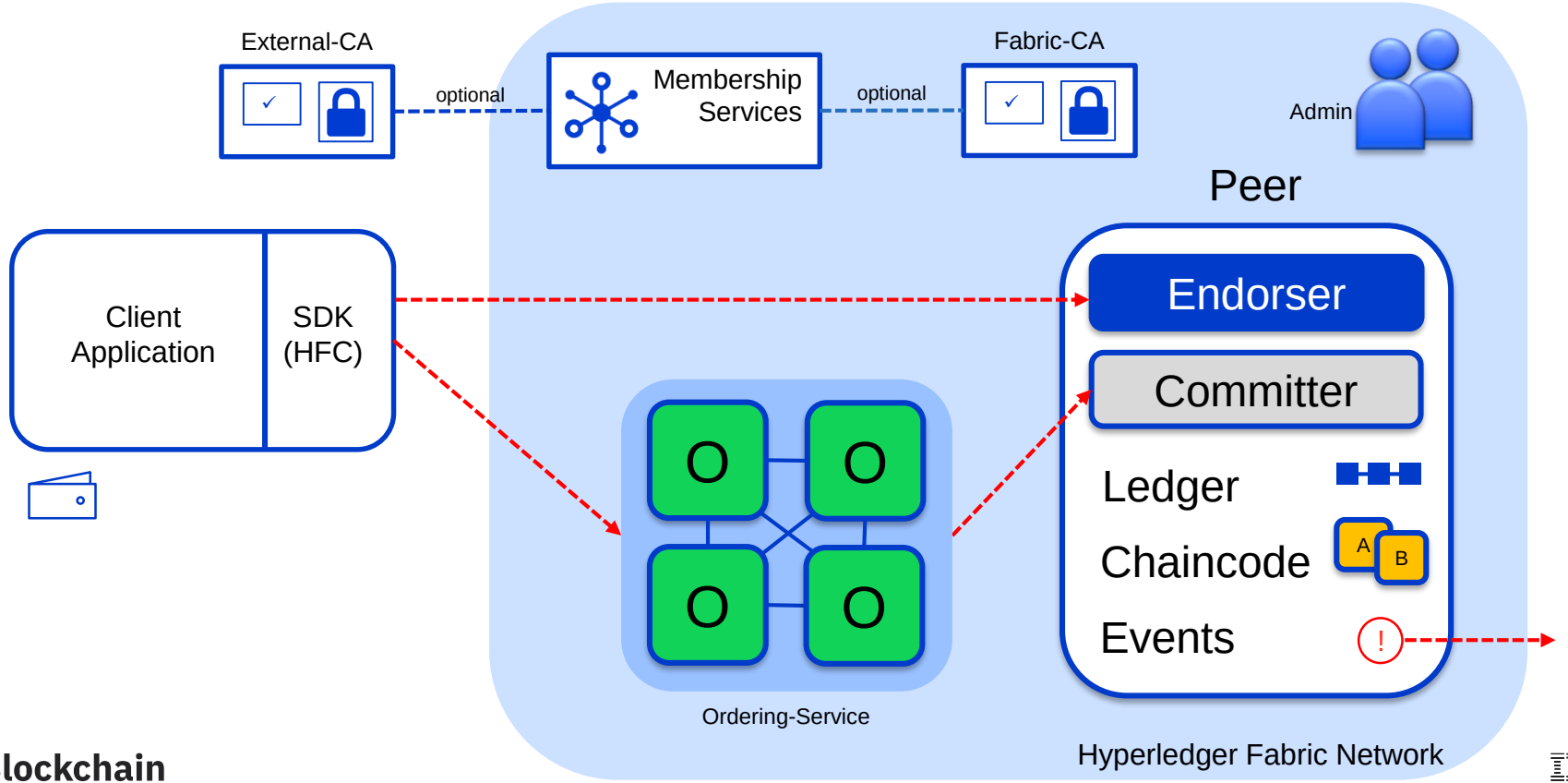
World state: new contents

```
myCar.vin = 1234
myCar.owner = Matt
myCar.make = Audi
…
```

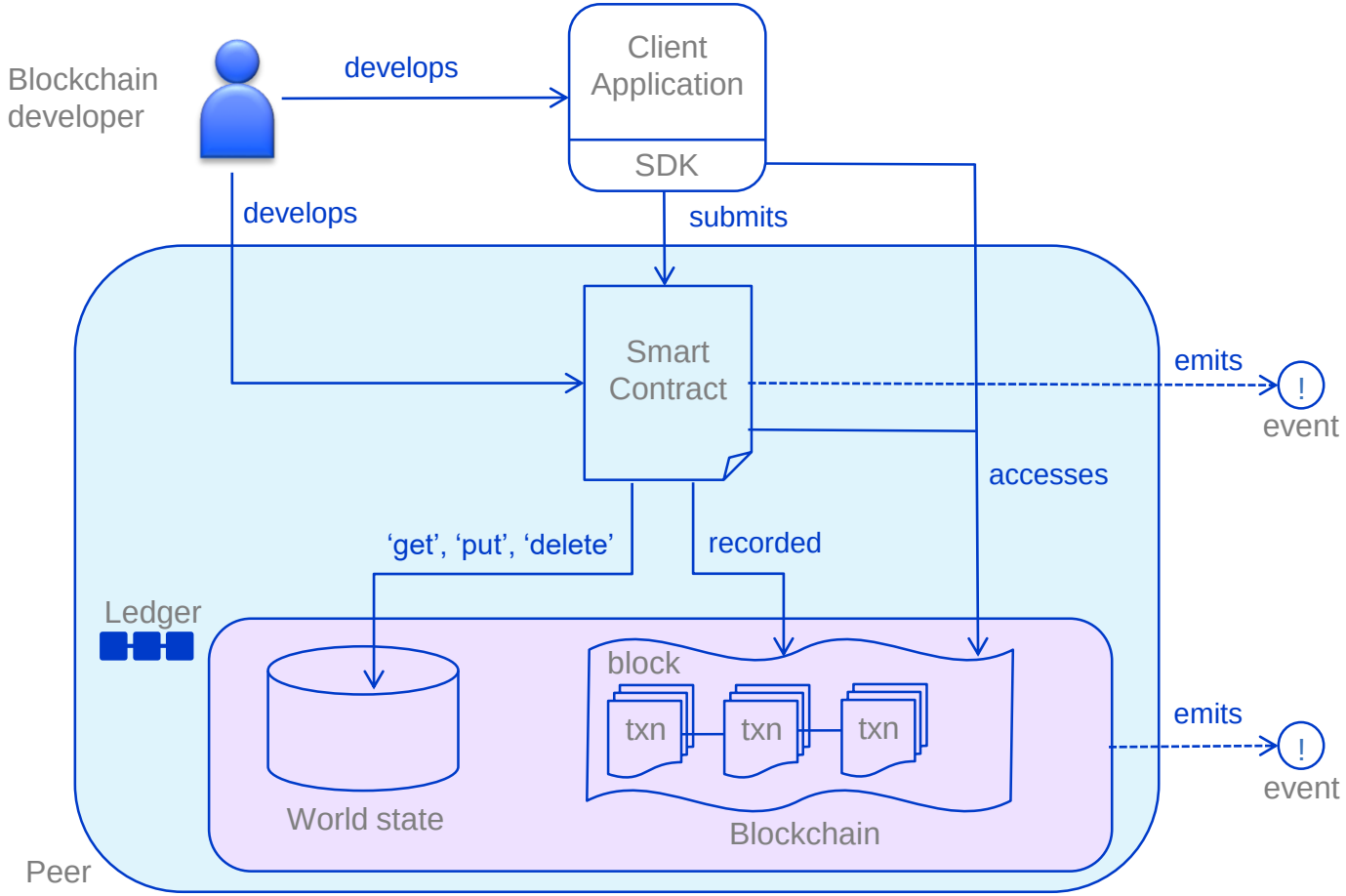
Hyperledger Fabric
Architecture

Hyperledger Fabric
Technical Deep Dive

Hyperledger Fabric V1 Architecture

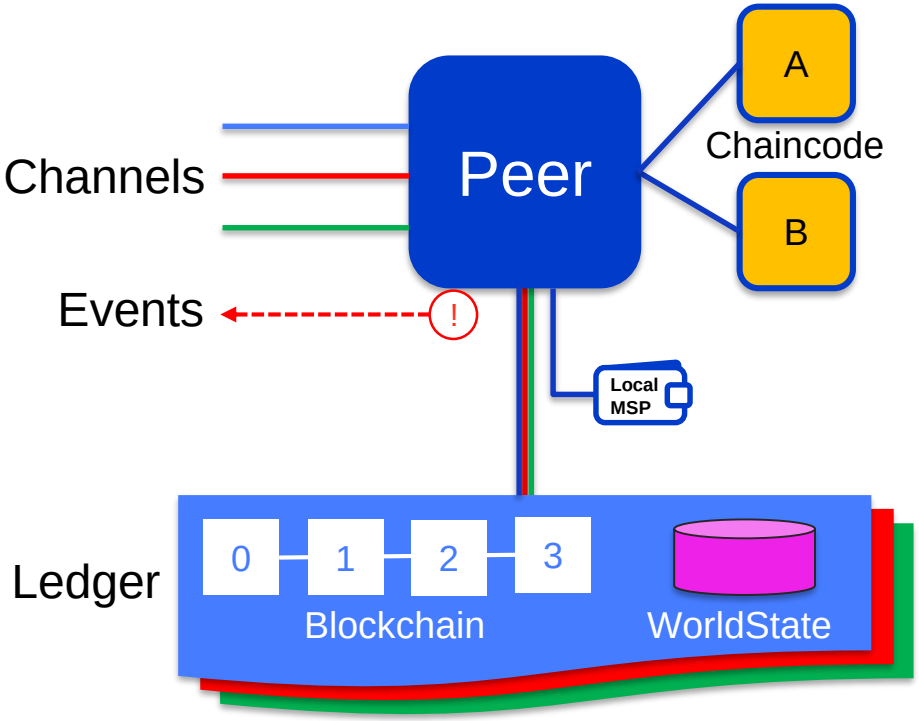


How applications interact with the ledger



Fabric Peer

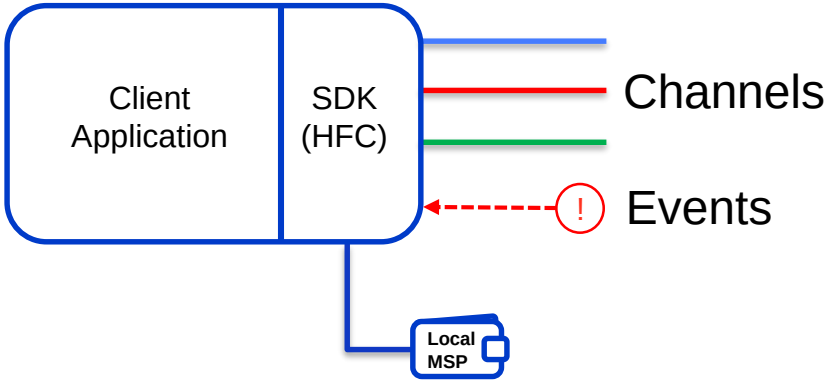
- Each peer:
  - Connects to one or more channels
  - Maintains one or more ledgers per channel
  - Maintains installed chaincode
  - Manages runtime docker containers for instantiated chaincode
    - Chaincode is instantiated on a channel
    - Runtime docker container shared by channels with same chaincode instantiated (no state stored in container)
  - Has a local MSP (Membership Services Provider) that provides crypto material
  - Emits events to the client application



Client Application

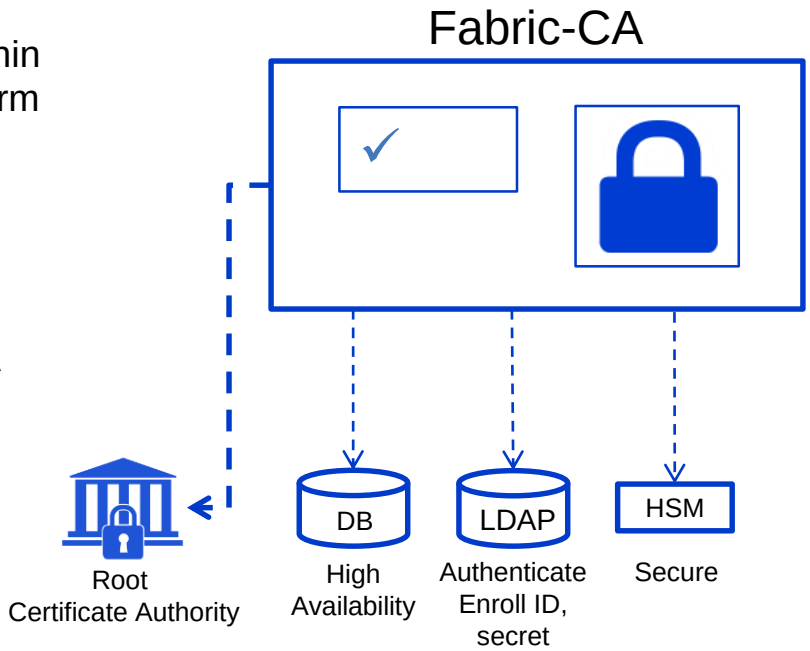
- Each client application uses Fabric SDK to:
  - Connects over channels to one or more peers
  - Connects over channels to one or more orderer nodes
  - Receives events from peers
  - Local MSP provides client crypto material

  - Client can be written in different languages (Node.js, Go, Java, Python?)

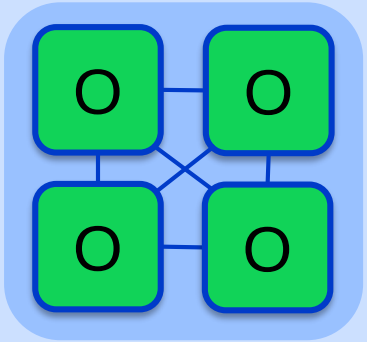


Fabric-CA

- Default (optional) Certificate Authority within Fabric network for issuing Ecerts (long-term identity)
- Supports clustering for HA characteristics
- Supports LDAP for user authentication
- Supports HSM for security
- Can be configured as an intermediate CA



Bootstrap Network (1/6) - Configure & Start Ordering Service



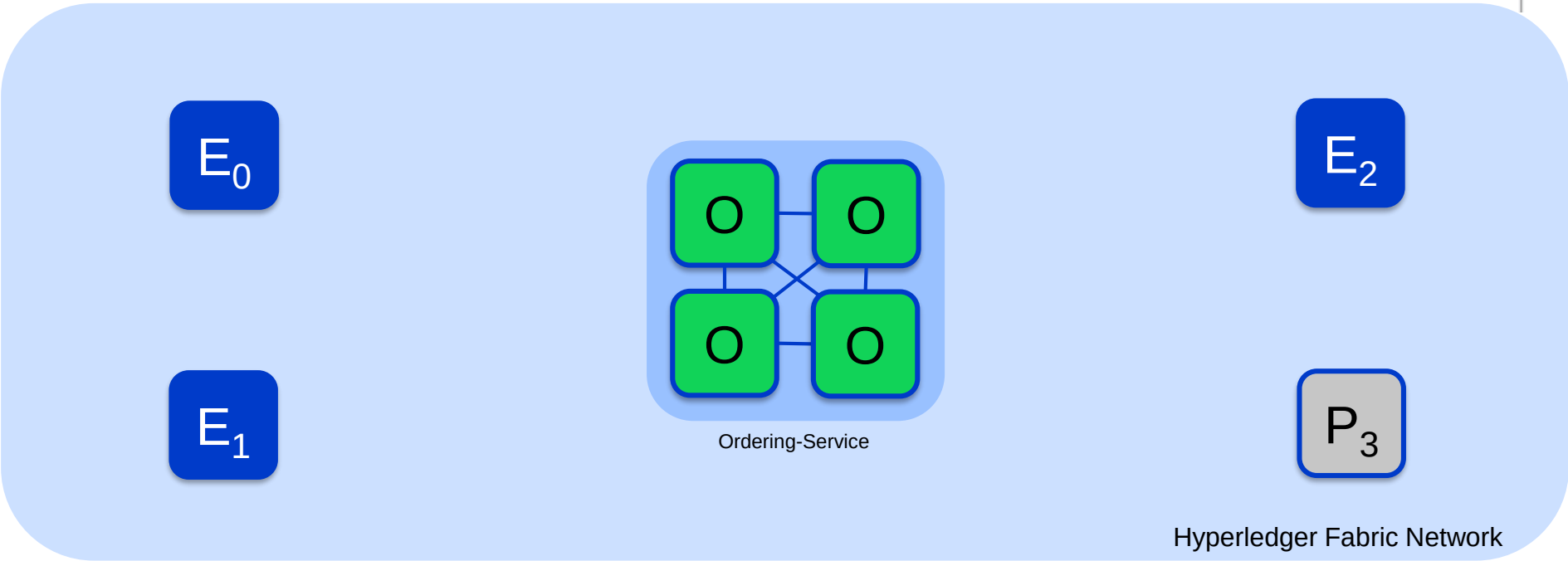
Ordering-Service

Hyperledger Fabric Network

An Ordering Service is configured and started for the network:

$ docker-compose [-f orderer.yml] ...

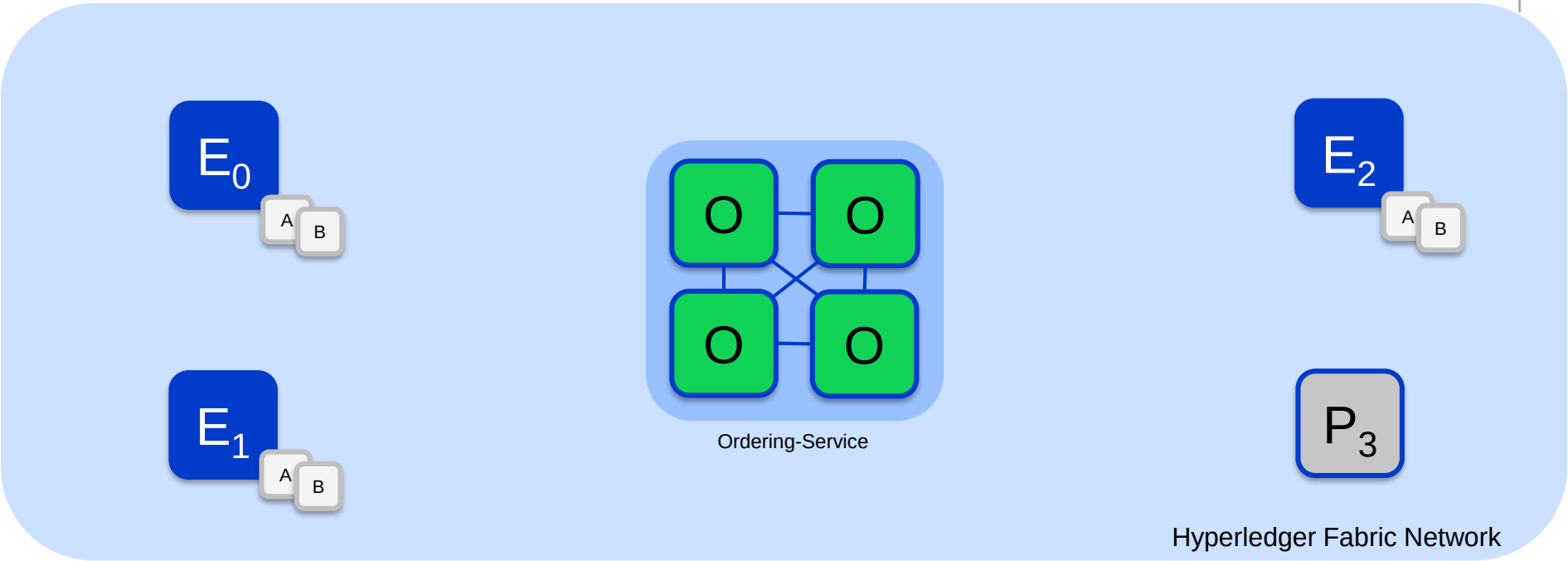
Bootstrap Network (2/6) - Configure and Start Peer Nodes



A peer is configured and started for each Endorser or Committer in the network:

$ peer node start ...

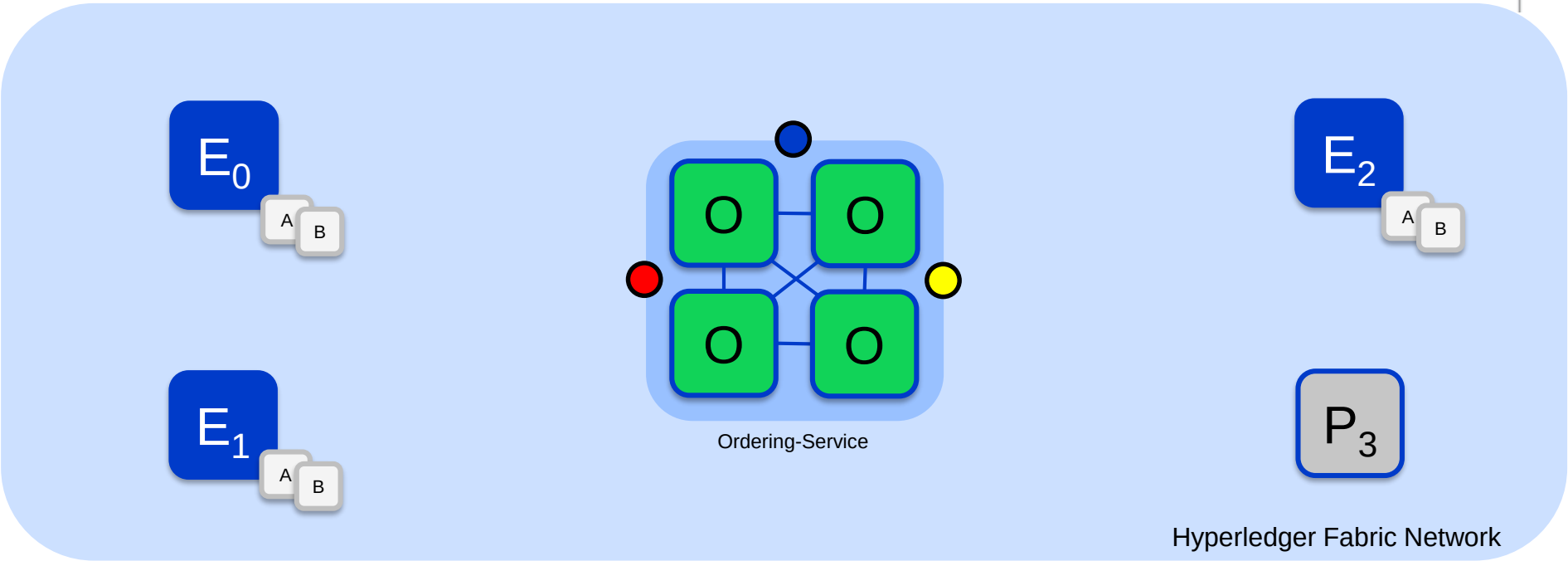
Bootstrap Network (3/6) - Install Chaincode



Chaincode is installed onto each Endorsing Peer that needs to execute it:

$ peer chaincode install ...

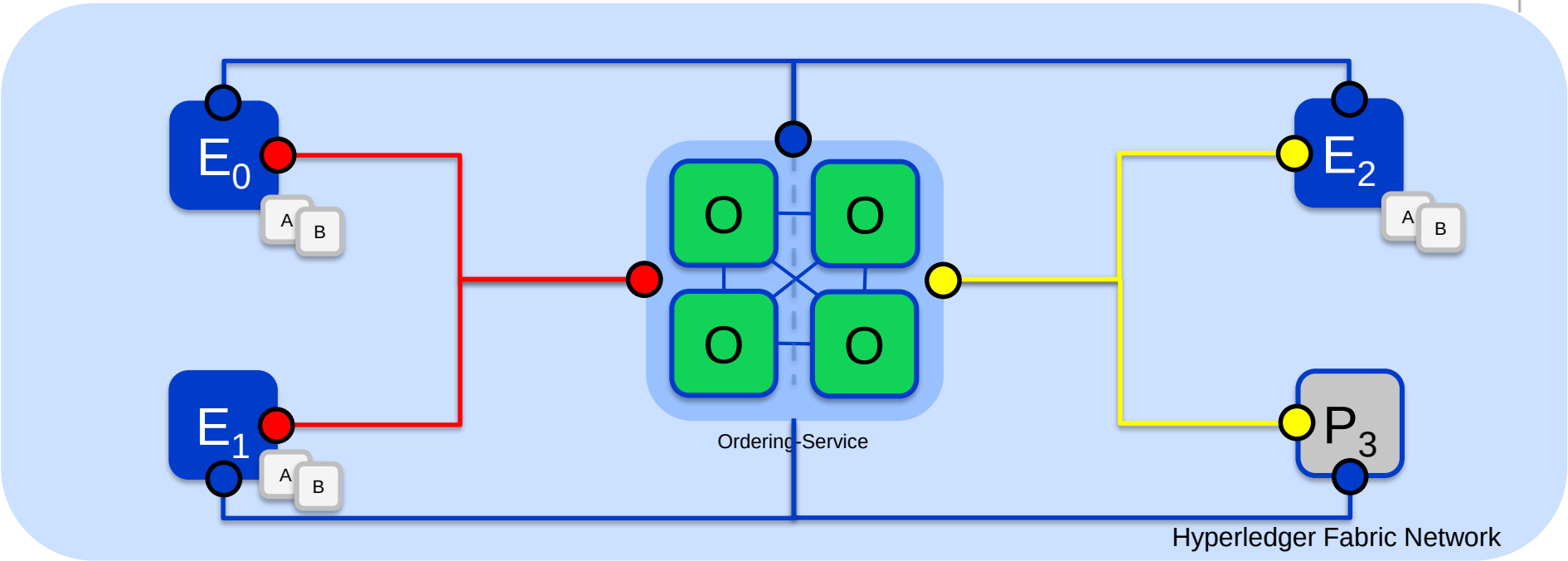
Bootstrap Network (4/6) – Create Channels



Channels are created on the ordering service:

$ peer channel create –o [orderer] ...

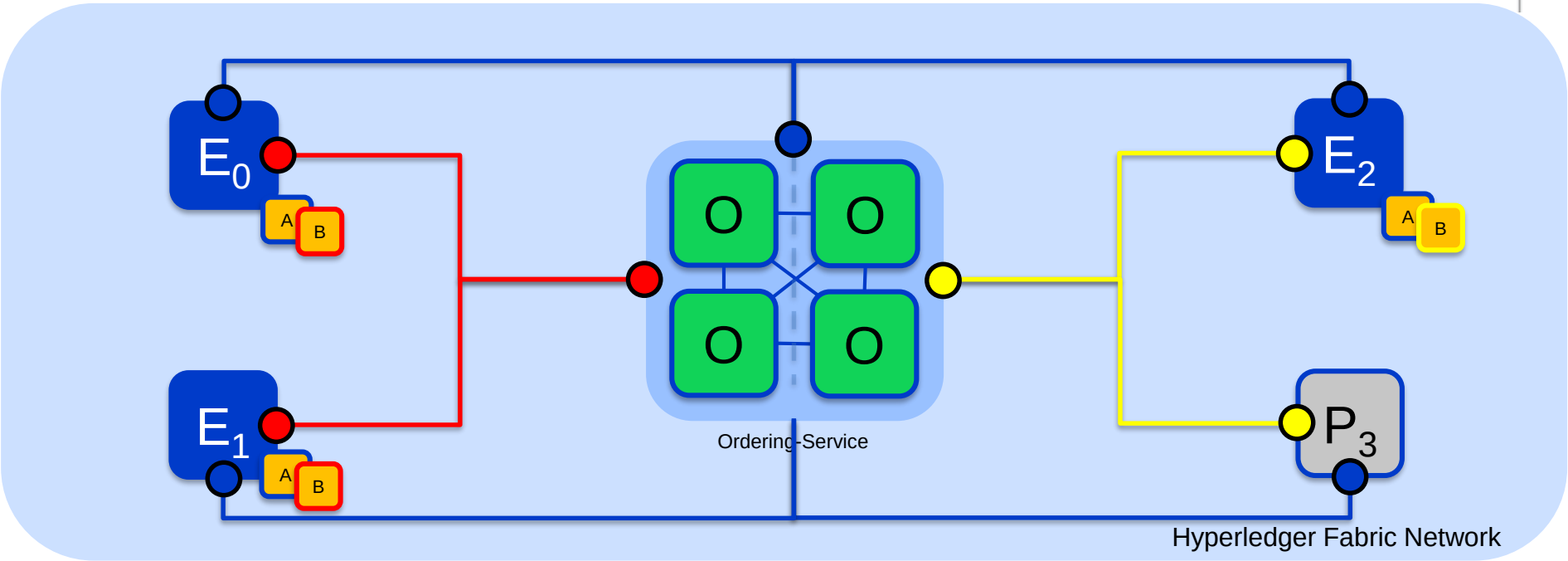
Bootstrap Network (5/6) – Join Channels



Peers that are permissioned can then join the channels they want to transact on:

$ peer channel join ...

Bootstrap Network (6/6) – Instantiate Chaincode



Peers finally instantiate the Chaincode on the channels they want to transact on:

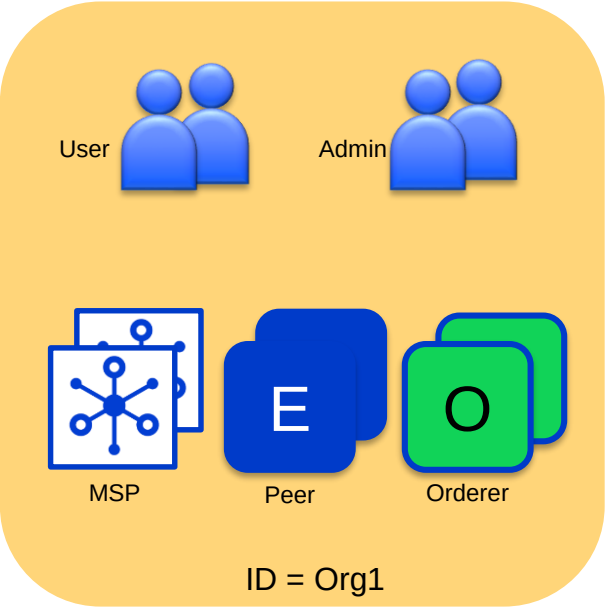
$ peer chaincode instantiate ... –P 'policy'

An Endorsement Policy is specified and once instantiated chaincode can process transactions.

Organizations

Organizations define boundaries within a Fabric Blockchain Network

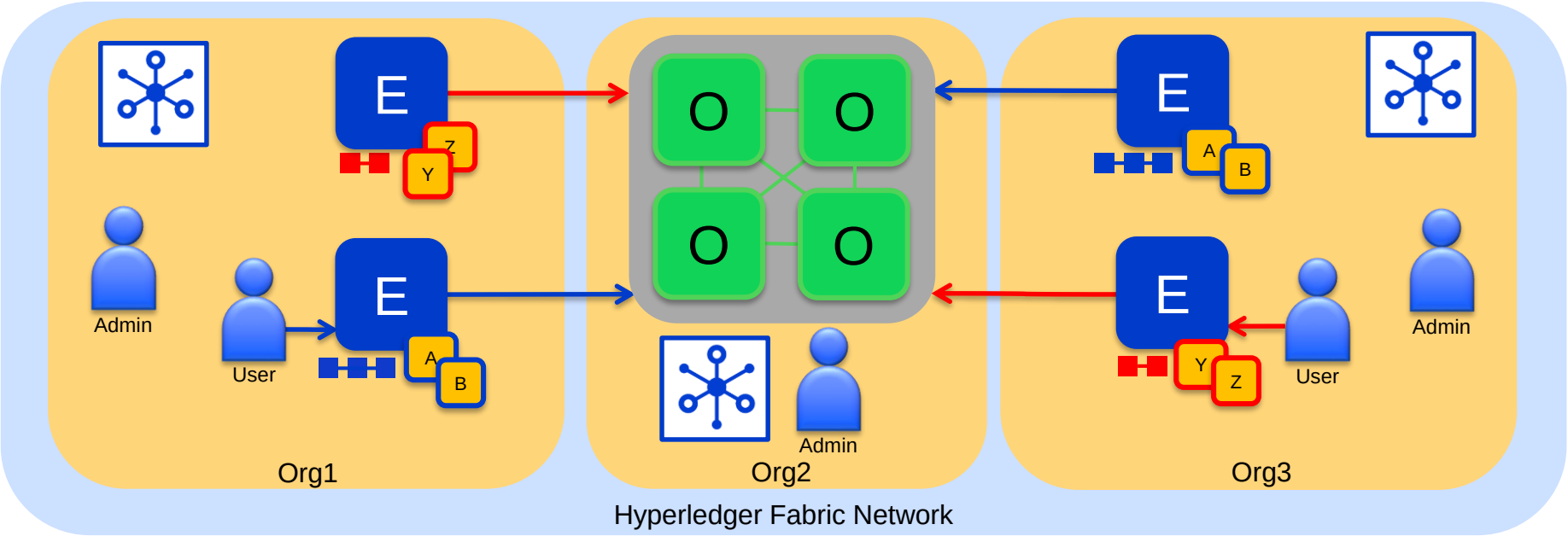
- Each organization defines:
  - Membership Services Provider (MSP) for identities
  - Administrator(s)
  - Users
  - Peers
  - Orderers (optional)
- A network can include many organisations representing a consortium
- Each organization has an ID



Consortium Network

An example consortium network of 3 organisations

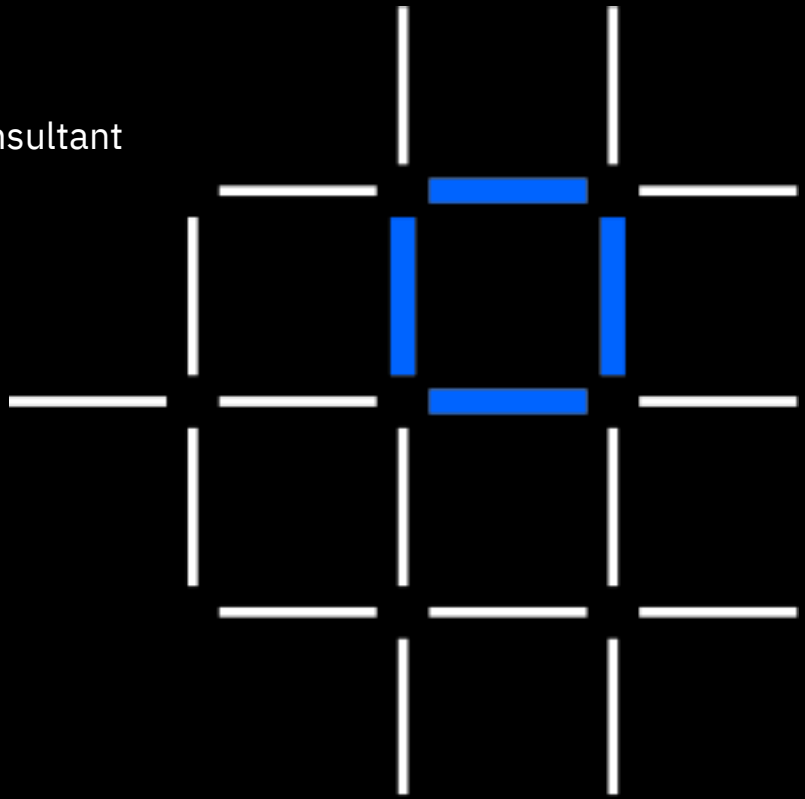
- Orgs 1 and 3 run peers
- Org 2 provides the ordering service only



Thank you

Carlos Rischioto
Blockchain Technical Leader
carlosr@br.ibm.com
rischioto

Tito Garrido Ogando
Blockchain Technical Consultant
titog@br.ibm.com
titogarrido



Questions? Tweet us or go to ibm.com/blockchain

@IBMBlockchain
IBM Blockchain
IBM Blockchain