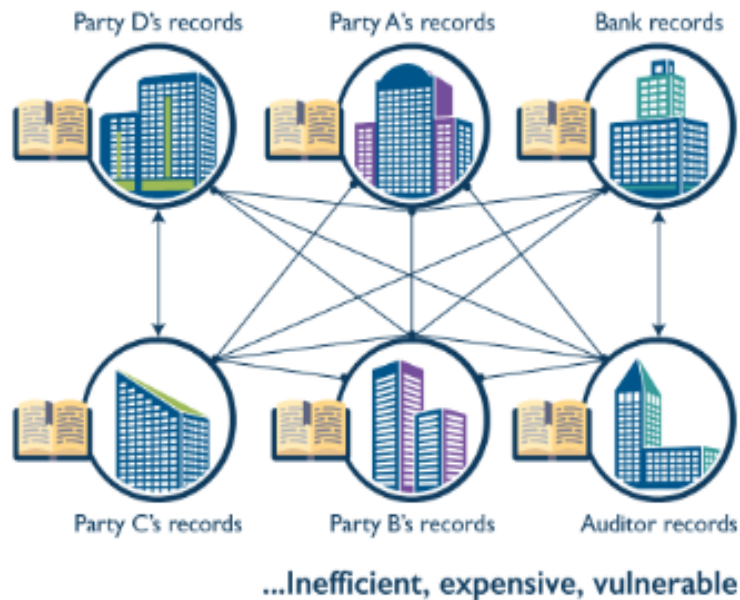


UNIT IV

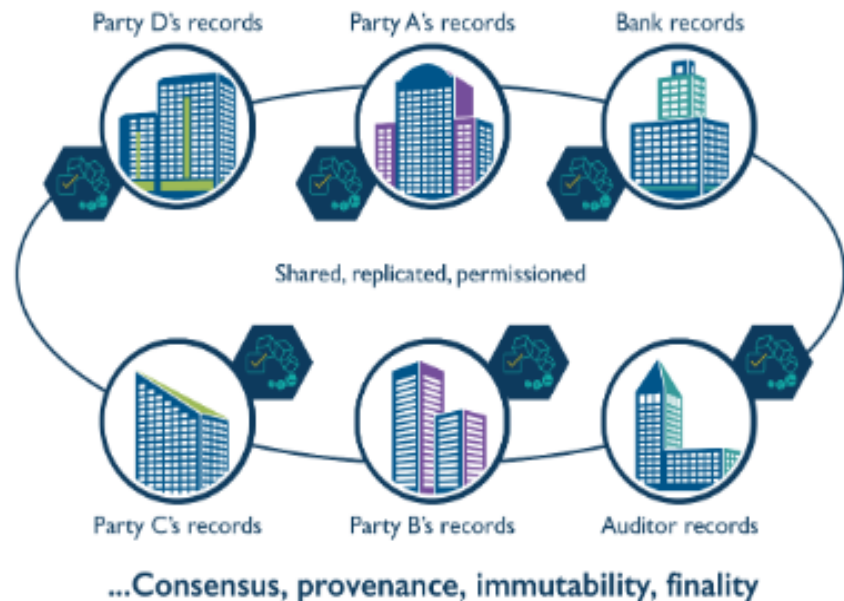
Blockchain For Enterprises Hyperledger

Traditional v/s Blockchain Business Network

Problem...



Solution...



What Is Blockchain For Business?

- It's the combination of a shared, unalterable ledger and smart contracts that streamline business processes and open new opportunities for innovation
- Consensus among Blockchain network members eliminates costly risks and inefficiencies as assets change hands throughout a business network
- This enterprise-ready Blockchain platform makes it easy to activate and manage a secure business network across multiple organizations

Shared ledger



Append-only distributed system of record shared across business network

Smart contract



Business terms embedded in transaction database & executed with transactions

Privacy



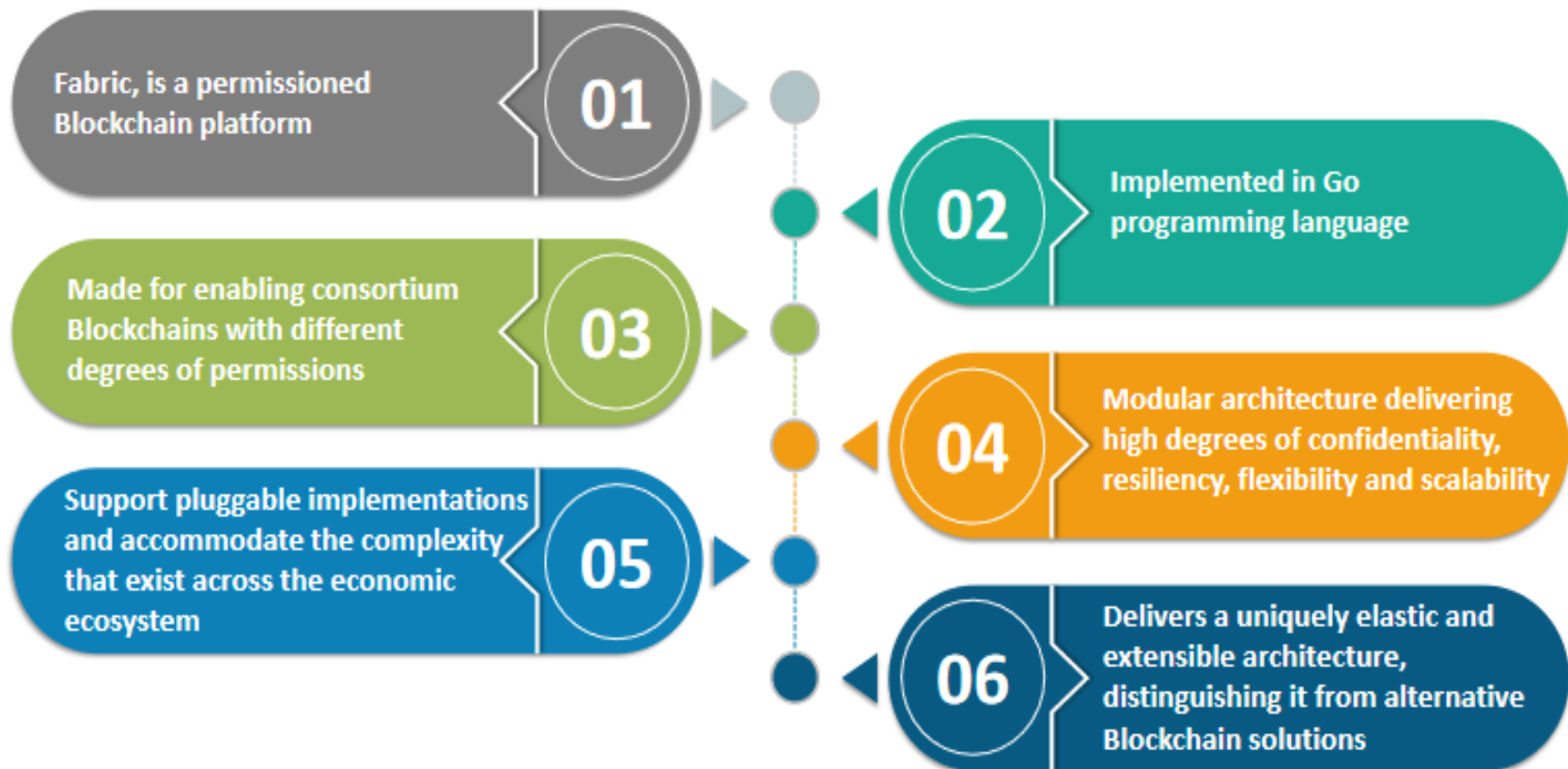
Ensuring appropriate visibility: transactions are secure, authenticated & verifiable

Consensus



All parties agree to network verified transaction

Hyperledger Fabric



Hyperledger Fabric

Hyperledger Fabric is designed for use in enterprise-level applications, and it is characterized by its modular architecture, permissioned network, and smart contract functionality, known as “chaincode”.

The platform provides a high degree of **security, privacy, and scalability**, and it supports the development of custom blockchain solutions for various use cases across industries such as **finance, supply chain, and healthcare**.

Hyperledger Fabric operates as a network of nodes, where each node performs a specific function, such as **validating transactions, maintaining the ledger, and executing chaincode**.

How does Hyperledger Fabric Work?

Components:

Hyperledger fabric is an enterprise-level permission blockchain network. For example, these organizations can be a [bank, financial institution, or a supply chain network](#). Each organization is identified and they have a fabric certificate authority. These organizations are called members.

Each member of the fabric can set up one or more authorized peers to participate in the network using the fabric certificate authority. All of these peers must be authorized properly.

There is a client-side application connected to the network written with the software development kit (SDK) of any particular programming language.

Hyperledger Fabric Consensus Algorithm

Hyperledger Fabric uses a consensus algorithm to achieve agreement among the participants in a network on the contents of the shared ledger.

The most commonly used consensus algorithms in Hyperledger Fabric are:

Practical Byzantine Fault Tolerance (PBFT): PBFT is a consensus algorithm that provides fault tolerance and reliability in a network.

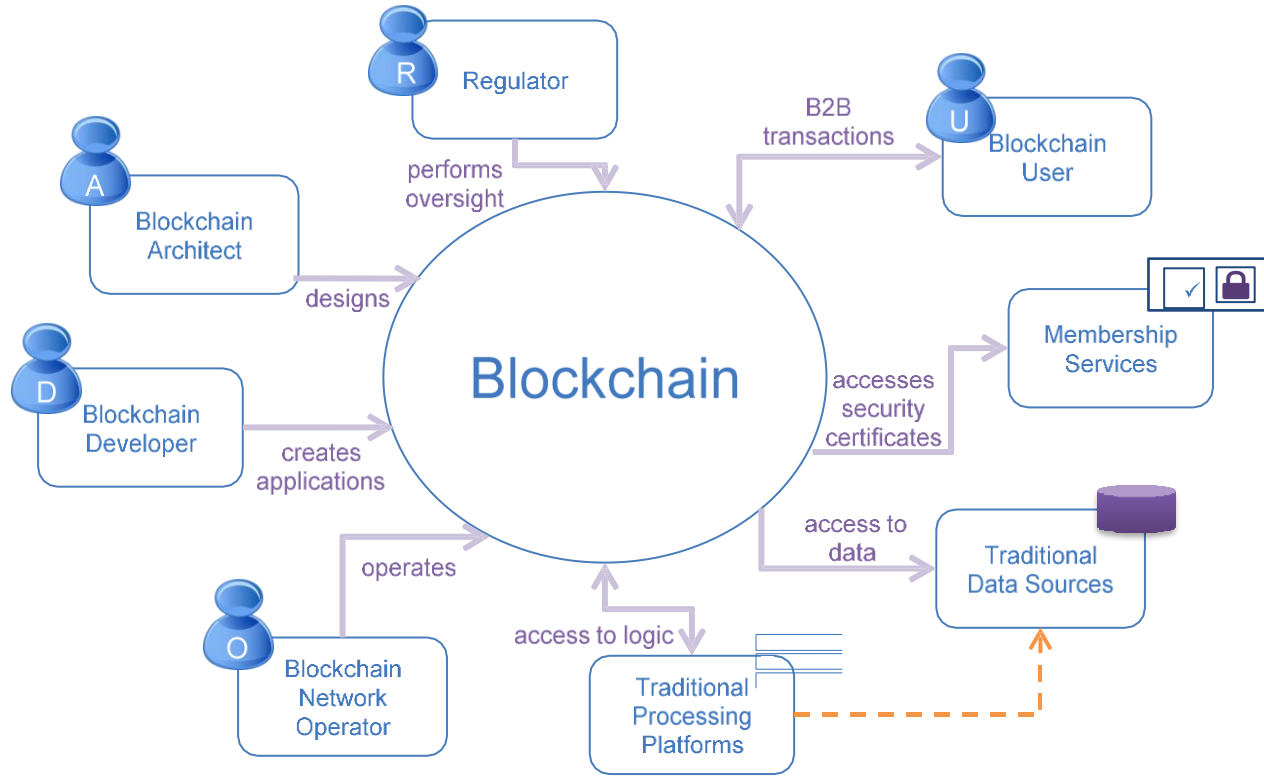
RAFT: RAFT is a consensus algorithm that is used to maintain a consistent state across multiple nodes.









Solo: Solo is a consensus algorithm that is used for testing purposes in a single-node network.

Benefits Of Hyperledger Fabric




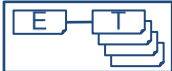




1. **Open Source:** Hyperledger fabric is an open-source blockchain framework hosted by the Linux foundation in 2015.
2. **Private and Confidential:** it identities of all participating members are authenticated.
3. **Access Control:** It employs its own mechanism for transaction ordering and provides an additional layer of access control.
4. **Chaincode Functionality:** It includes a container technology to host smart contracts called *chain code* that defines the business rules of the system.
5. **Performance:** There is no need to validate the transactions on this network so the transaction speed is faster, resulting in a better performance.

Actors

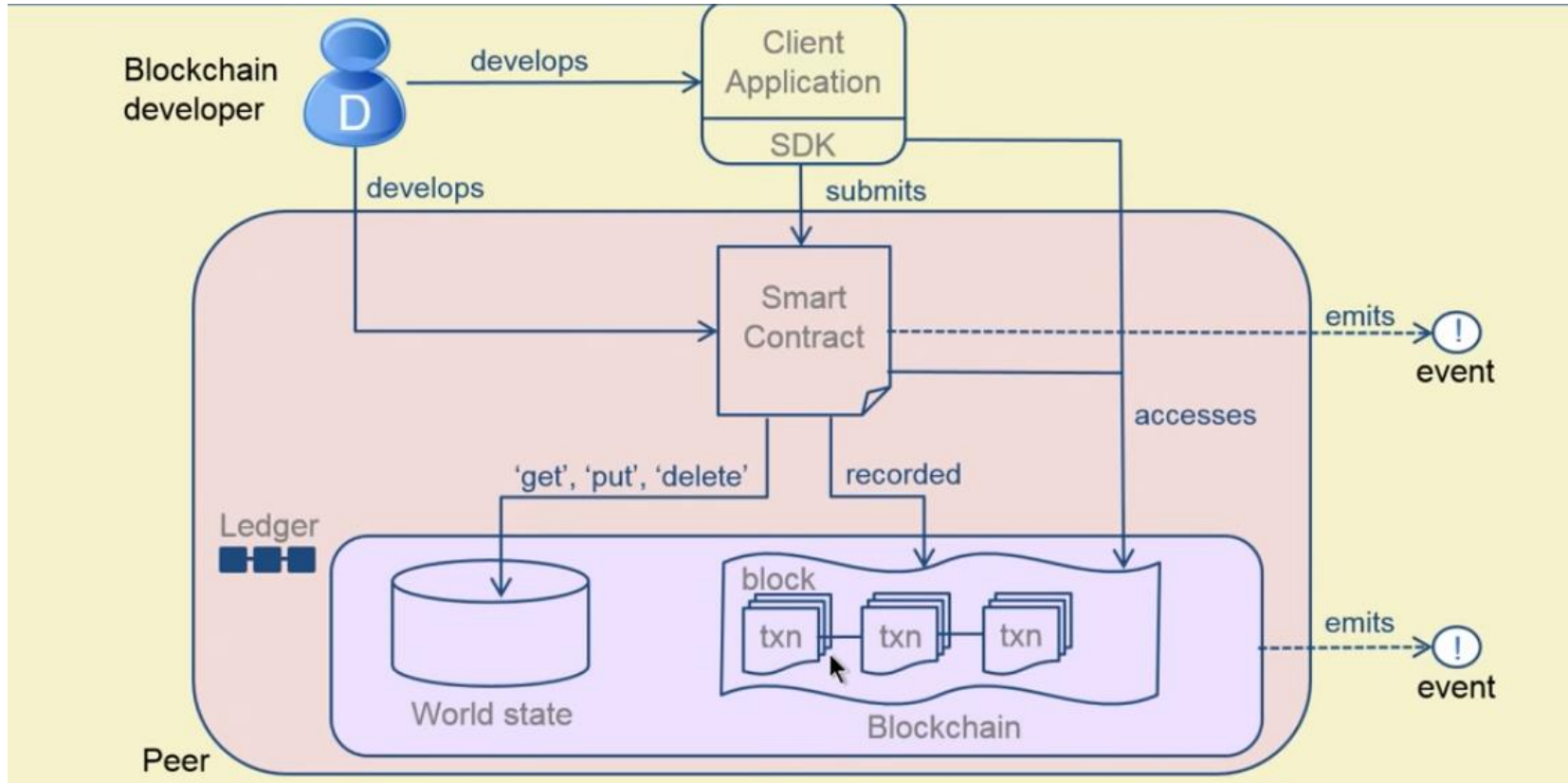


Blockchain Architect		Responsible for the architecture and design of the blockchain solution
Blockchain User		The business user, operating in a business network. This role interacts with the Blockchain using an application. They are not aware of the Blockchain.
Blockchain Regulator		The overall authority in a business network. Specifically, regulators may require broad access to the ledger's contents.
Blockchain Developer		The developer of applications and smart contracts that interact with the Blockchain and are used by Blockchain users.
Blockchain Operator		Manages and monitors the Blockchain network. Each business in the network has a Blockchain Network operator.
Membership Services		Manages the different types of certificates required to run a permissioned Blockchain.
Traditional Processing Platform		An existing computer system which may be used by the Blockchain to augment processing. This system may also need to initiate requests into the Blockchain.
Traditional Data Sources		An existing data system which may provide data to influence the behavior of smart contracts.

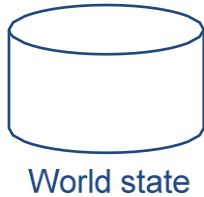
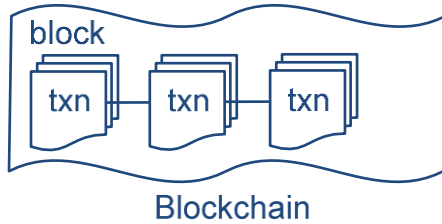
Components

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.

System Architecture:






Ledger consists of:

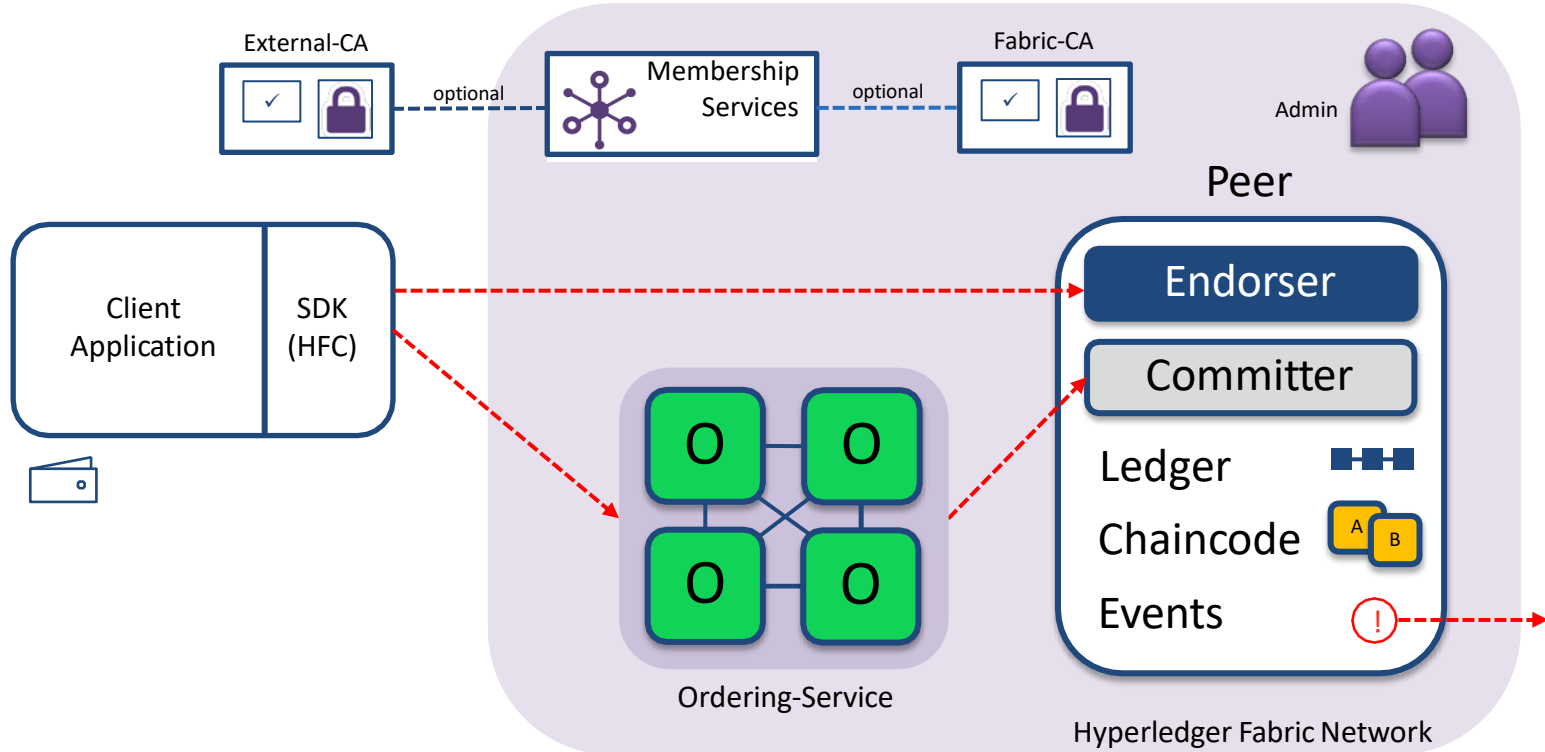


- Blockchain
 - A linked list of blocks (a hashchain)
 - Each block describes a set of transactions (e.g. the inputs to a smart contract invocation, output, identities/certs)
 - Immutable – blocks cannot be tampered
- World State
 - Stores the most recent state of smart contracts / output of transactions
 - Stored in a traditional database (e.g. key-value store)
 - Data elements can be added, modified, deleted, all recorded as transactions on blockchain

Nodes and their Roles

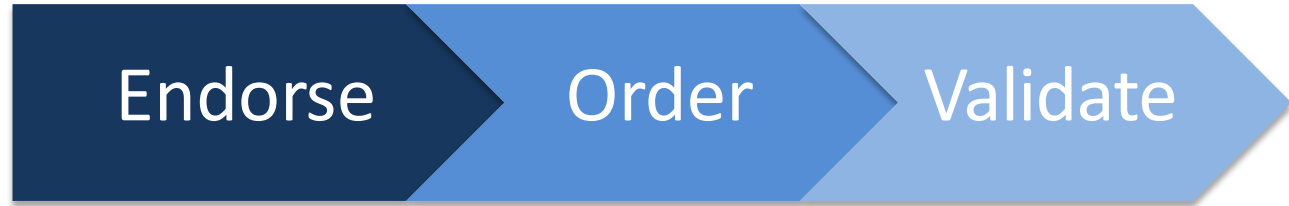
	<p>Committing Peer: Maintains ledger and state. Commits transactions. May hold smart contract (chaincode).</p>
	<p>Endorsing Peer: Specialized committing peer that receives a transaction proposal for endorsement, responds granting or denying endorsement. Must hold smart contract</p>
	<p>Ordering Node: Approves the inclusion of transaction blocks into the ledger and communicates with committing and endorsing peer nodes. Does not hold smart contract. Does not hold ledger.</p>

Architecture

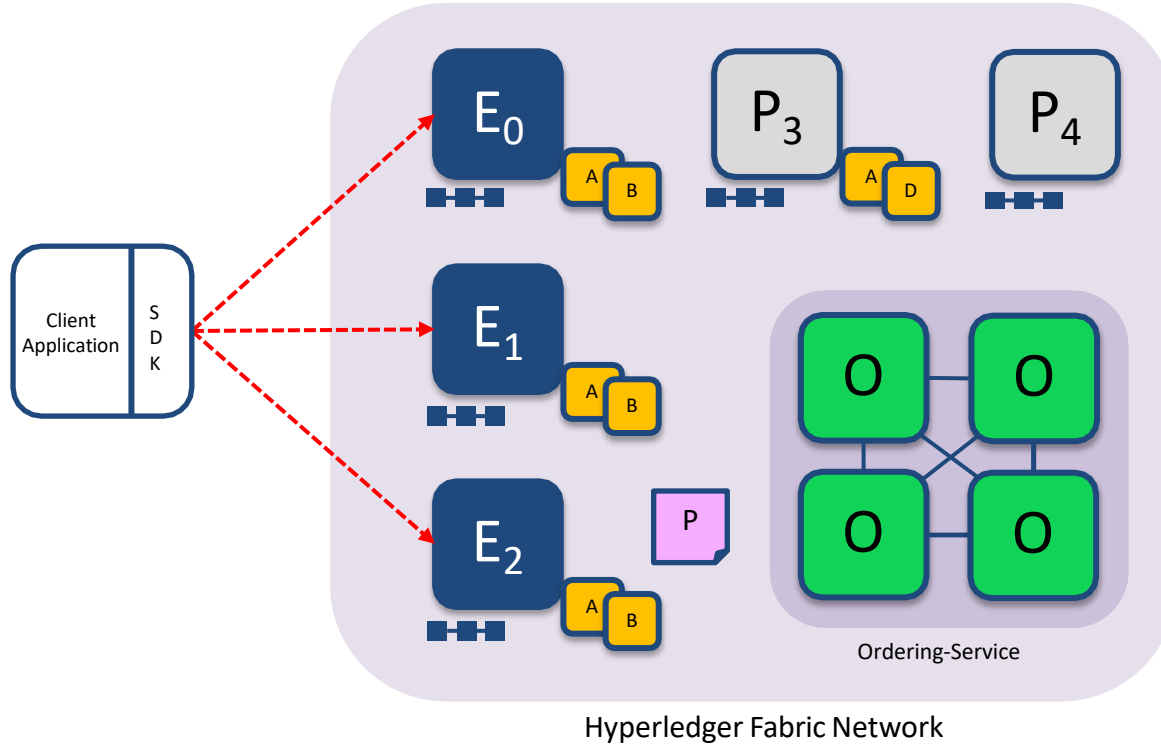


Transaction Flow

Consensus is achieved using the following transaction flow:



Step 1/7: Propose Transaction



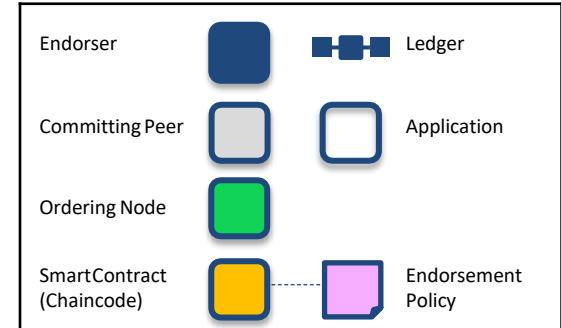
Application proposes transaction

Endorsement policy:

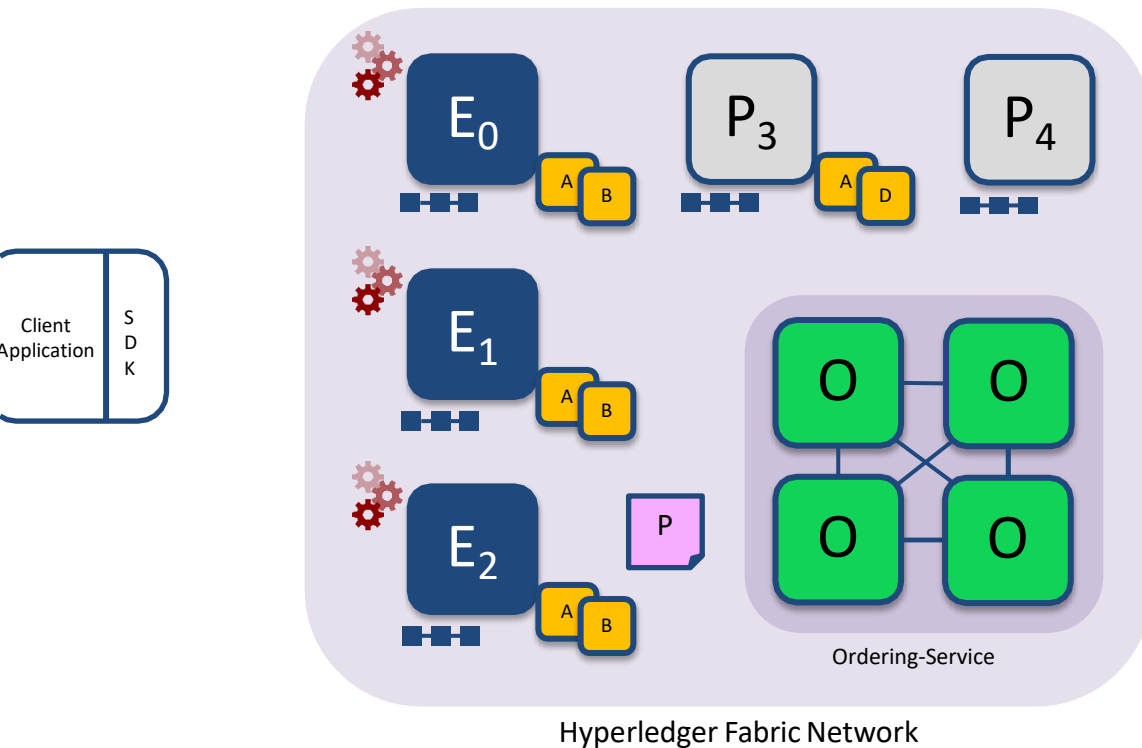
- “E₀, E₁ and E₂ must sign”
- (P₃, P₄ are not part of the policy)

Client application submits a transaction proposal for Smart Contract A. It must target the required peers {E₀, E₁, E₂}

Key:



Step 2/7: Execute Proposed Transaction



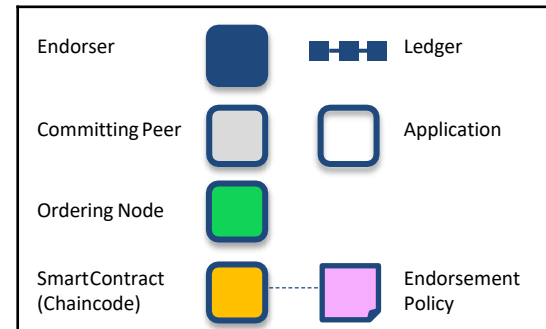
Endorsers Execute Proposals

E_0 , E_1 & E_2 will each execute the proposed transaction. None of these executions will update the ledger

Each execution will capture the set of Read and Write data, called RW sets, which will now flow in the fabric.

Transactions can be signed & encrypted

Key:



Step 3/7: Proposal Response

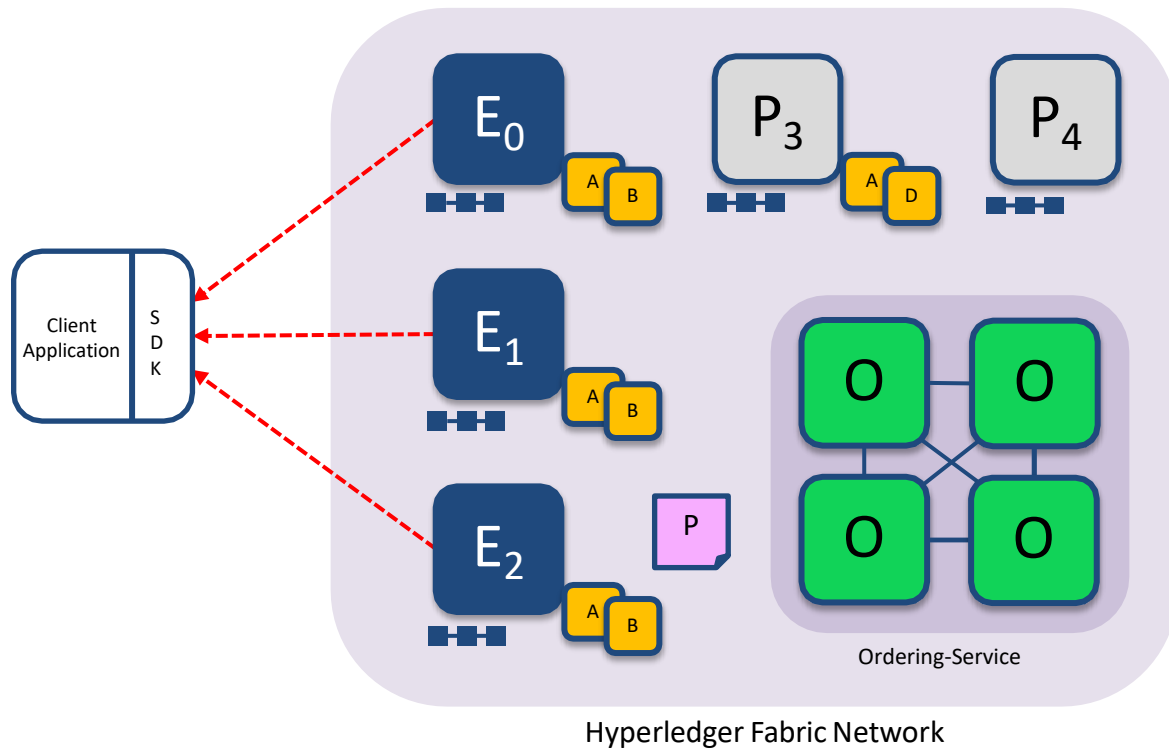
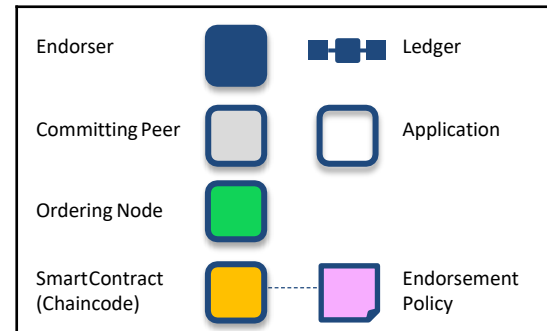
Application receives responses

Read-Write sets are asynchronously returned to application

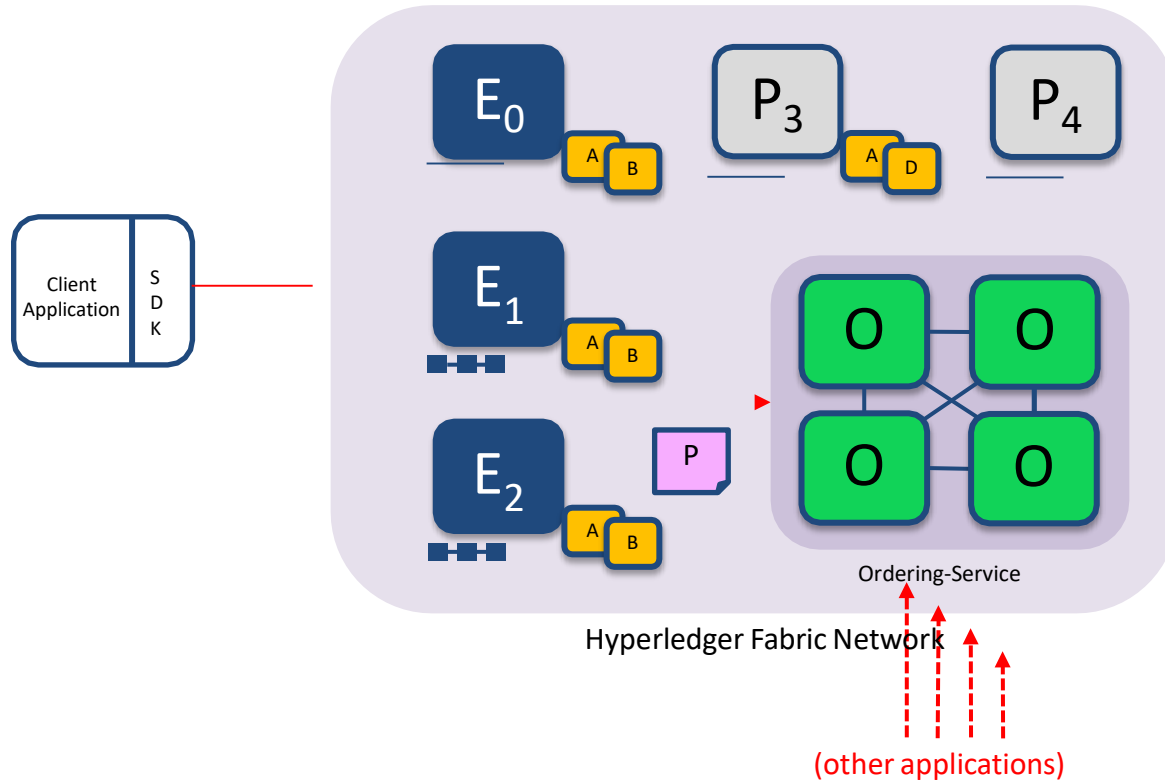
The RW sets are signed by each endorser, and also includes each record version number

(This information will be checked much later in the consensus process)

Key:



Step 4/7: Order Transaction

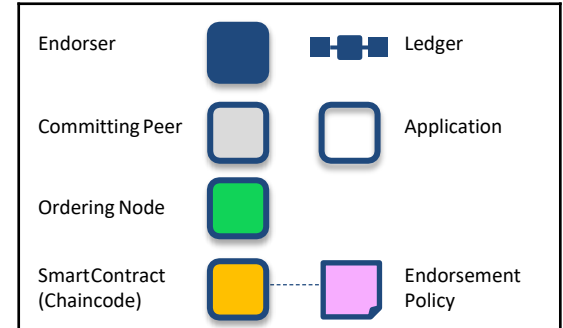


Responses submitted for ordering

Application submits responses as a transaction to be ordered.

Ordering happens across the fabric in parallel with transactions submitted by other applications

Key:



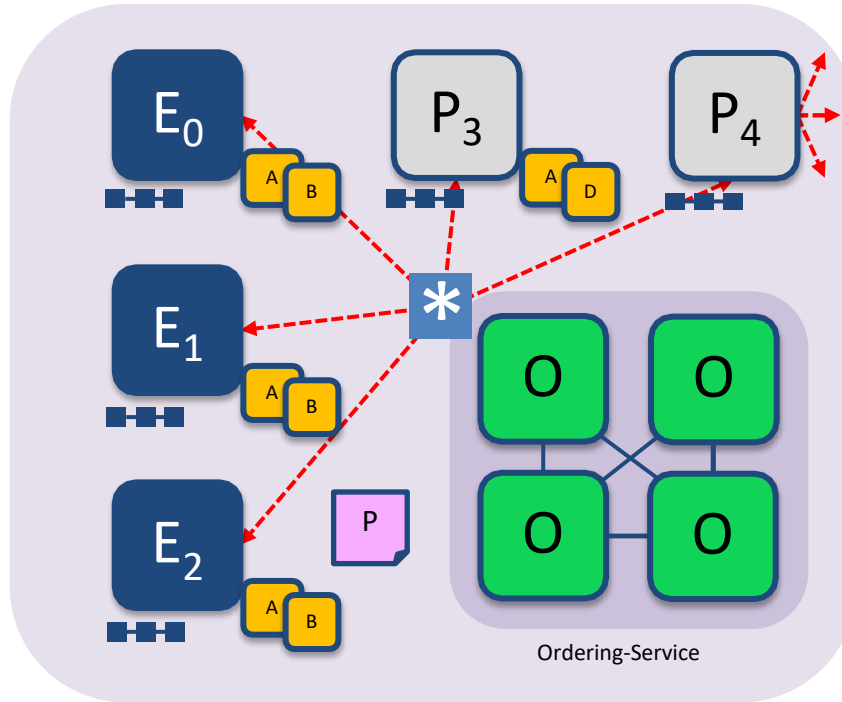
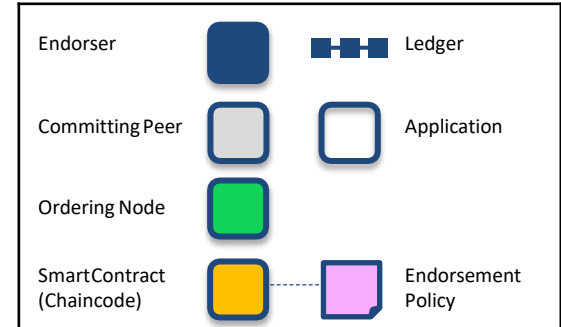
Step 5/7: Deliver Transaction

Orderer delivers to committing peers

Ordering service collects transactions into proposed blocks for distribution to committing peers. Peers can deliver to other peers in a hierarchy (not shown). Different ordering algorithms available:

- SOLO (Single node, development)
- Kafka (Crash fault tolerance)

Key:

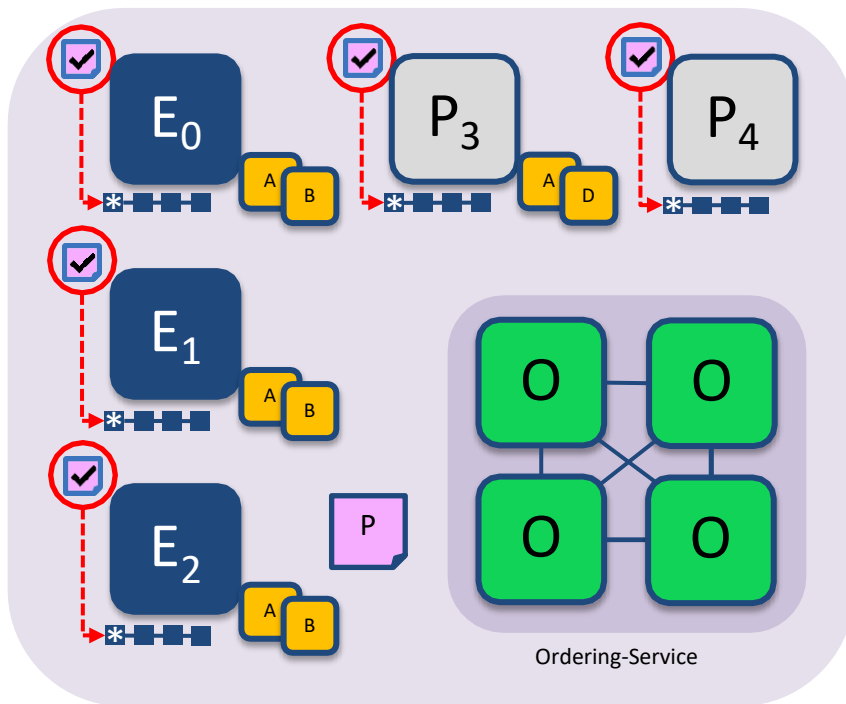
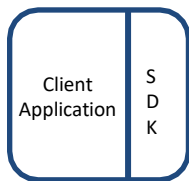


Hyperledger Fabric Network

Step 6/7: Validate Transaction

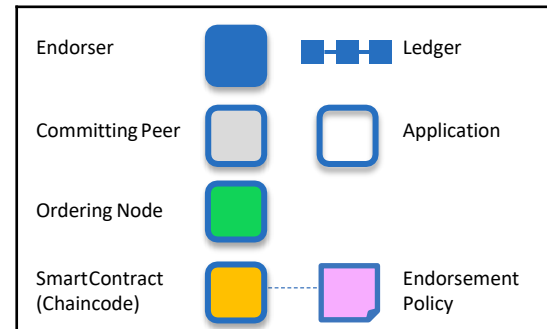
Committing peers validate transactions

Every committing peer validates against the endorsement policy. Also check RW sets are still valid for current world state
Validated transactions are applied to the world state and retained on the ledger



Hyperledger Fabric Network

Key:



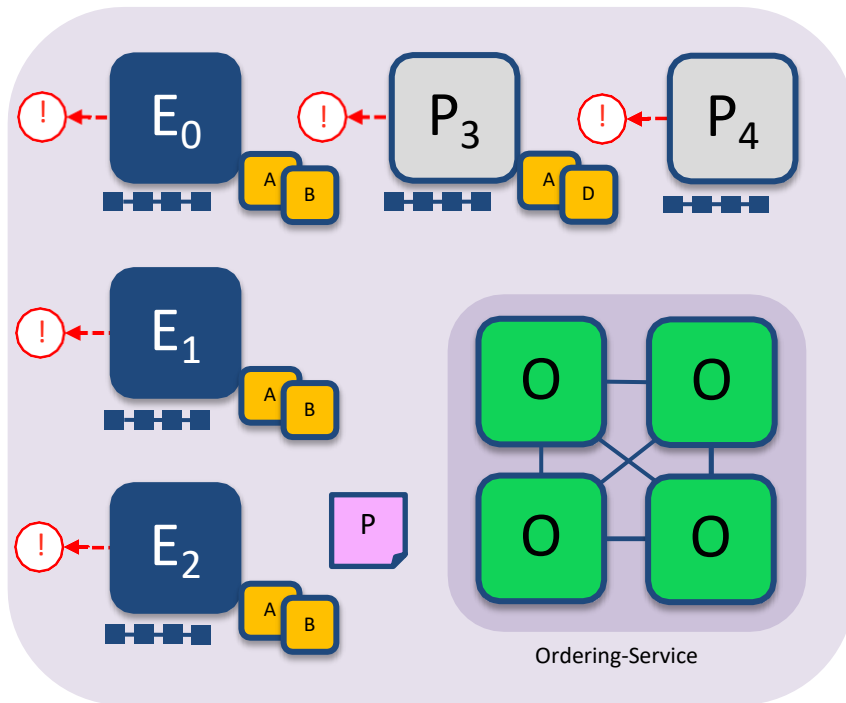
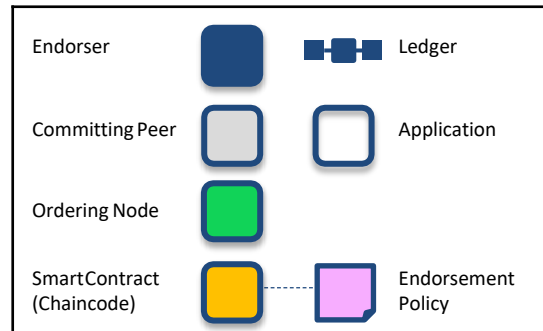
Step 7/7: Notify Transaction

Committing peers notify applications

Applications can register to be notified when transactions succeed or fail, and when blocks are added to the ledger

Applications will be notified by each peer to which they are connected

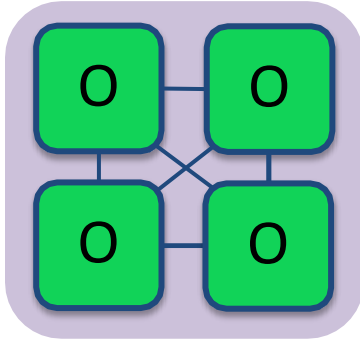
Key:



Hyperledger Fabric Network

Ordering Service

The ordering service packages transactions into blocks to be delivered to peers. Communication with the service is via channels.



Ordering-Service

Different configuration options for the ordering service include:

– SOLO

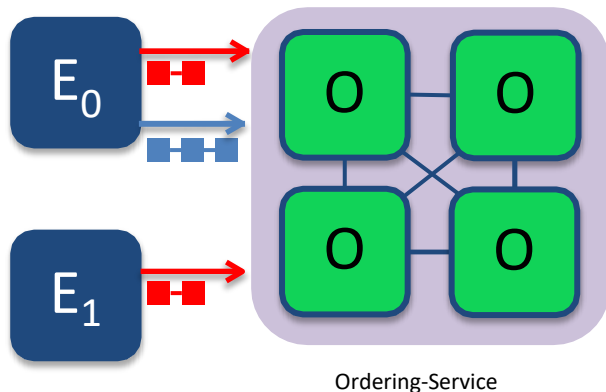
- Single node for development

– Kafka : Crash fault tolerant consensus

- 3 nodes minimum
- Odd number of nodes recommended

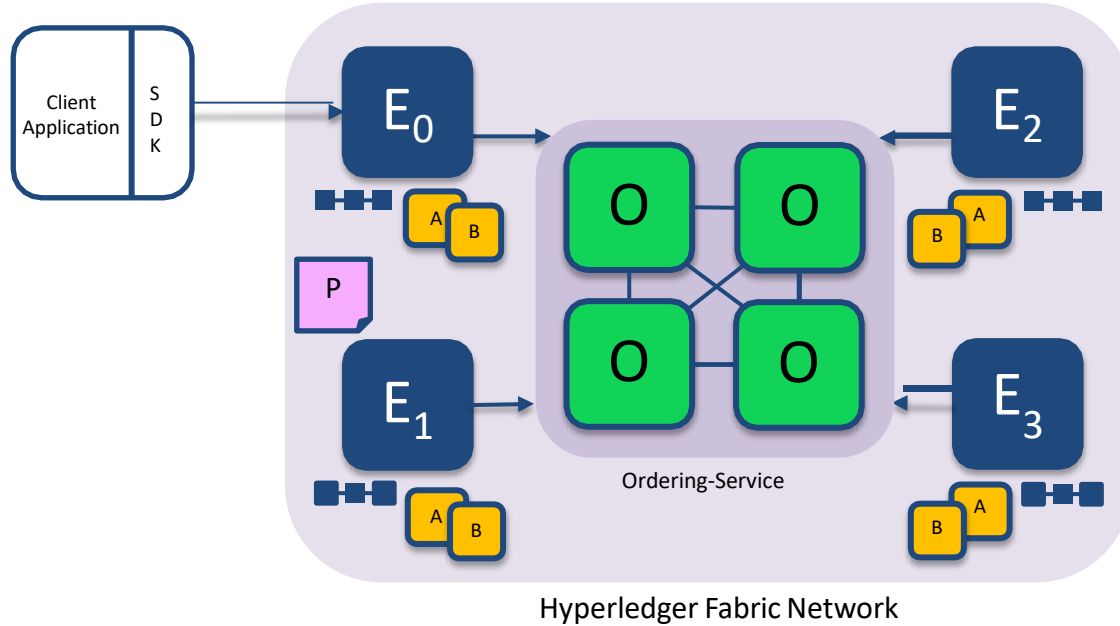
Channels

Channels provide privacy between different ledgers



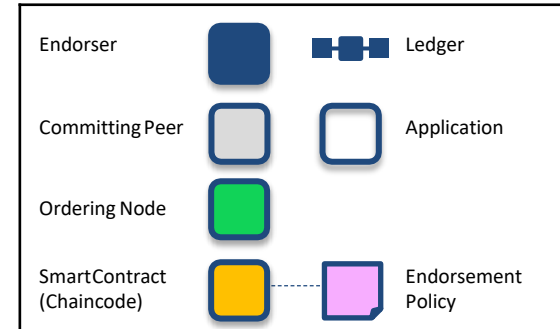
- Ledgers exist in the scope of a channel
 - Channels can be shared across an entire network of peers
 - Channels can be permissioned for a specific set of participants
- Chaincode is **installed** on peers to access the worldstate
- Peers can participate in multiple channels
- Concurrent execution for performance and scalability

Single Channel Network

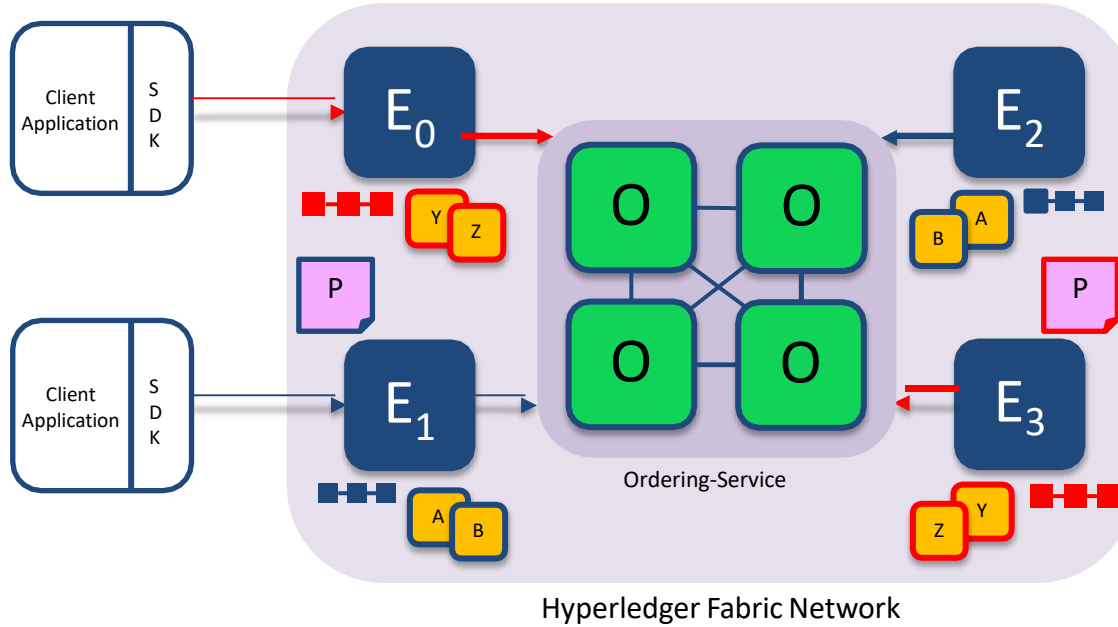


- All peers connect to the same system channel (blue).
- All peers have the same chaincode and maintain the same ledger
- Endorsement by peers E_0, E_1, E_2 and E_3

Key:

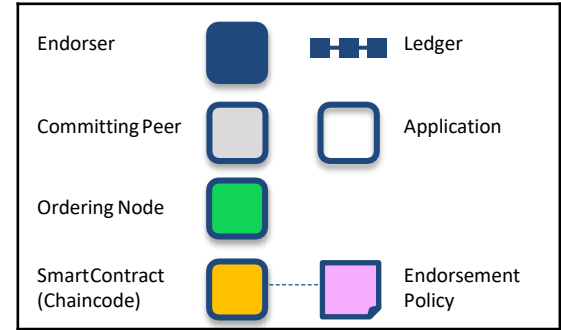


Multi-Channel Network



- Peers E₀ and E₃ connect to the **red** channel for chaincodes **Y** and **Z**
- Peers E₁ and E₂ connect to the **blue** channel for chaincodes **A** and **B**

Key:



Fabric Peer

- Each peer:
 - Connects to one or more **channels**
 - Maintains one or more **ledgers** for each channel
 - **Chaincodes are shared** across channels (no state is stored in chaincode container)
 - Local MSP (Membership Services Provider) provides **crypto material**
 - **Emits events** to the client application

