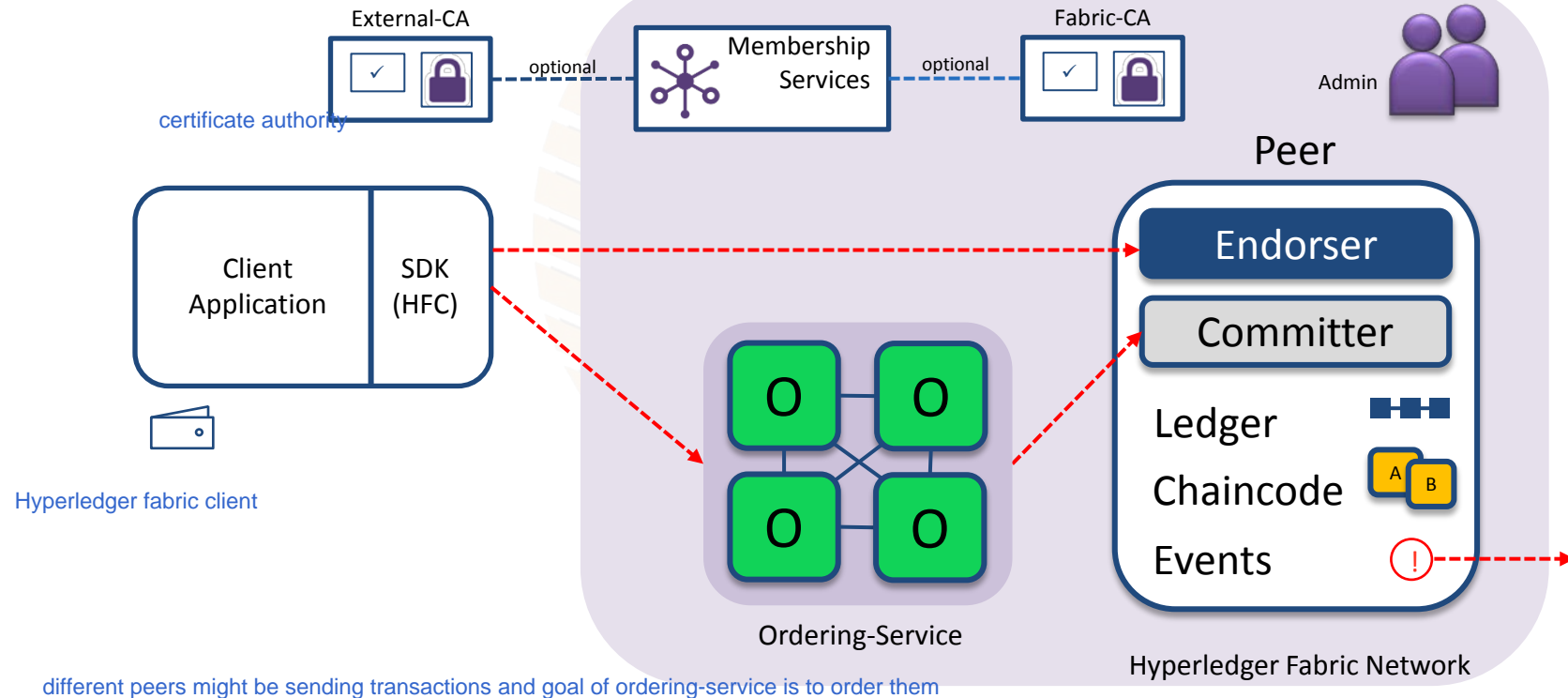





Hyperledger Fabric V1 Architecture

User can login with his signature and can also register to get signature and we have both public key and private key



Nodes and Roles

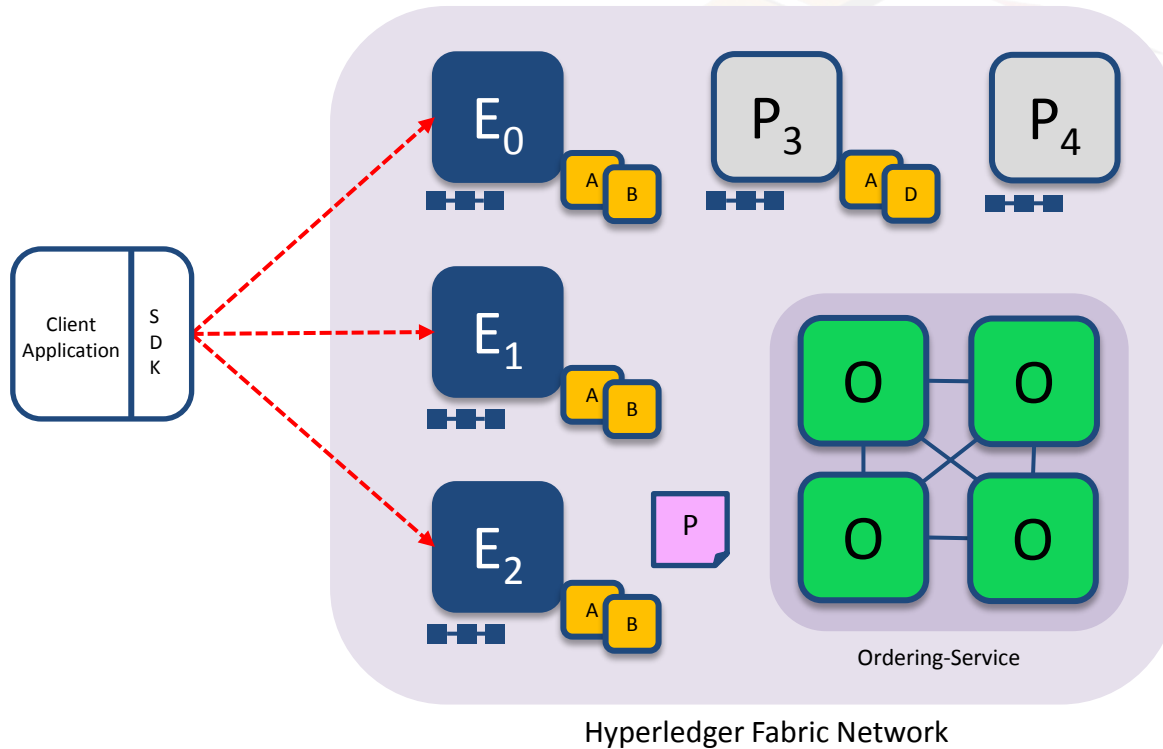
	<p>Committing Peer: Maintains ledger and state. Commits transactions. May hold smart contract (chaincode).</p>
	<p>it executes the transactions</p> <p>Endorsing Peer: Specialized committing peer that receives a transaction proposal for endorsement, responds granting or denying endorsement. Must hold 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>

Transaction Flow

Consensus is achieved using the following transaction flow:



Step 1/7: Propose Transaction



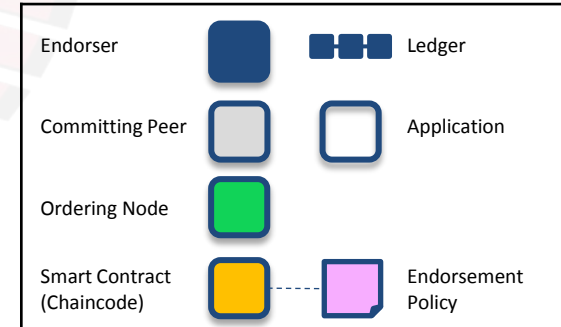
Application proposes transaction
endorsement policy is predefined by smart contract

Endorsement policy:

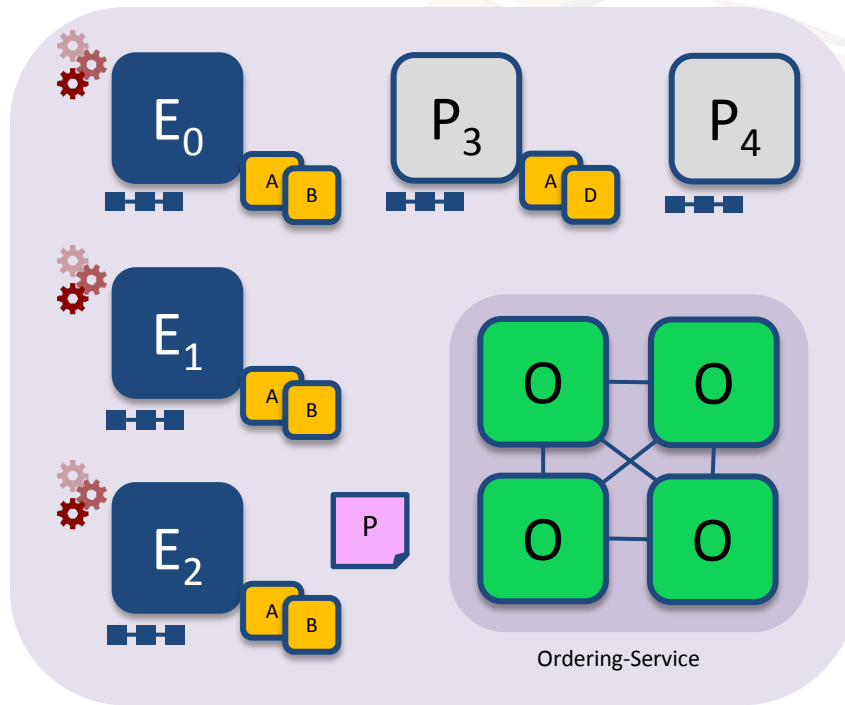
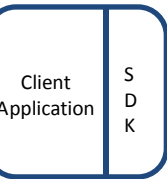
- “E₀, E₁ and E₂ must sign”
- (P₃, P₄ are not part of the policy)

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



Hyperledger Fabric Network

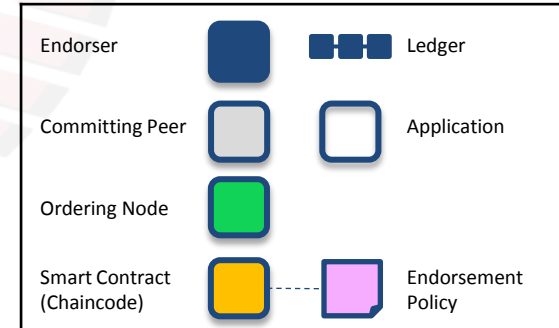
Endorsers Execute Proposals

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 Written data, called RW sets, which will now flow in the fabric.

Transactions can be signed & encrypted

Key:

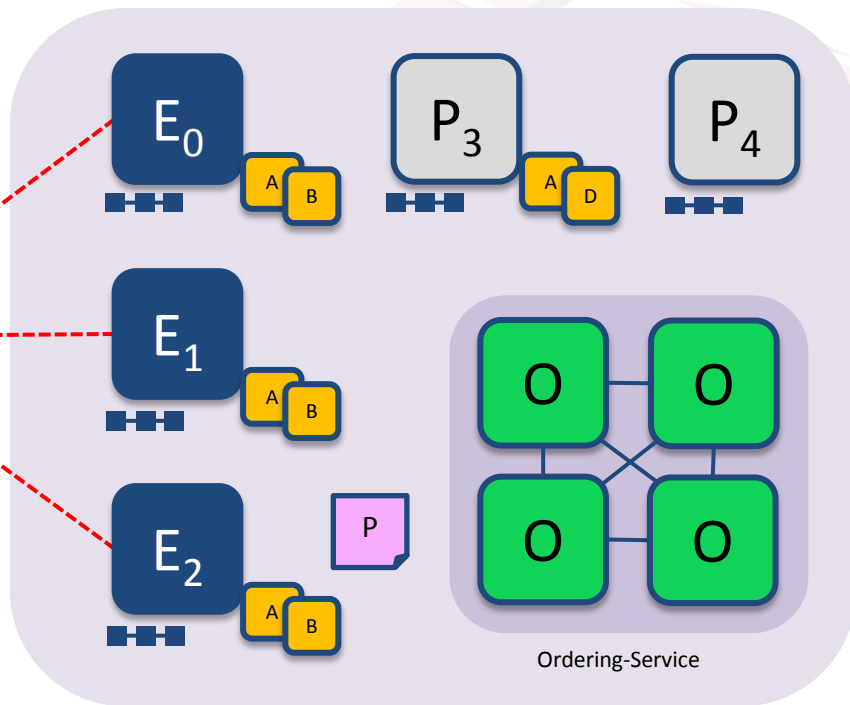
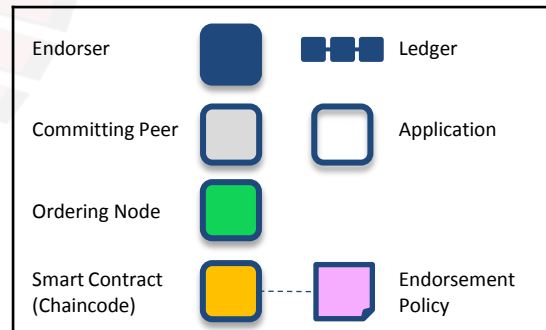


Step 3/7: Proposal Response

Application receives responses

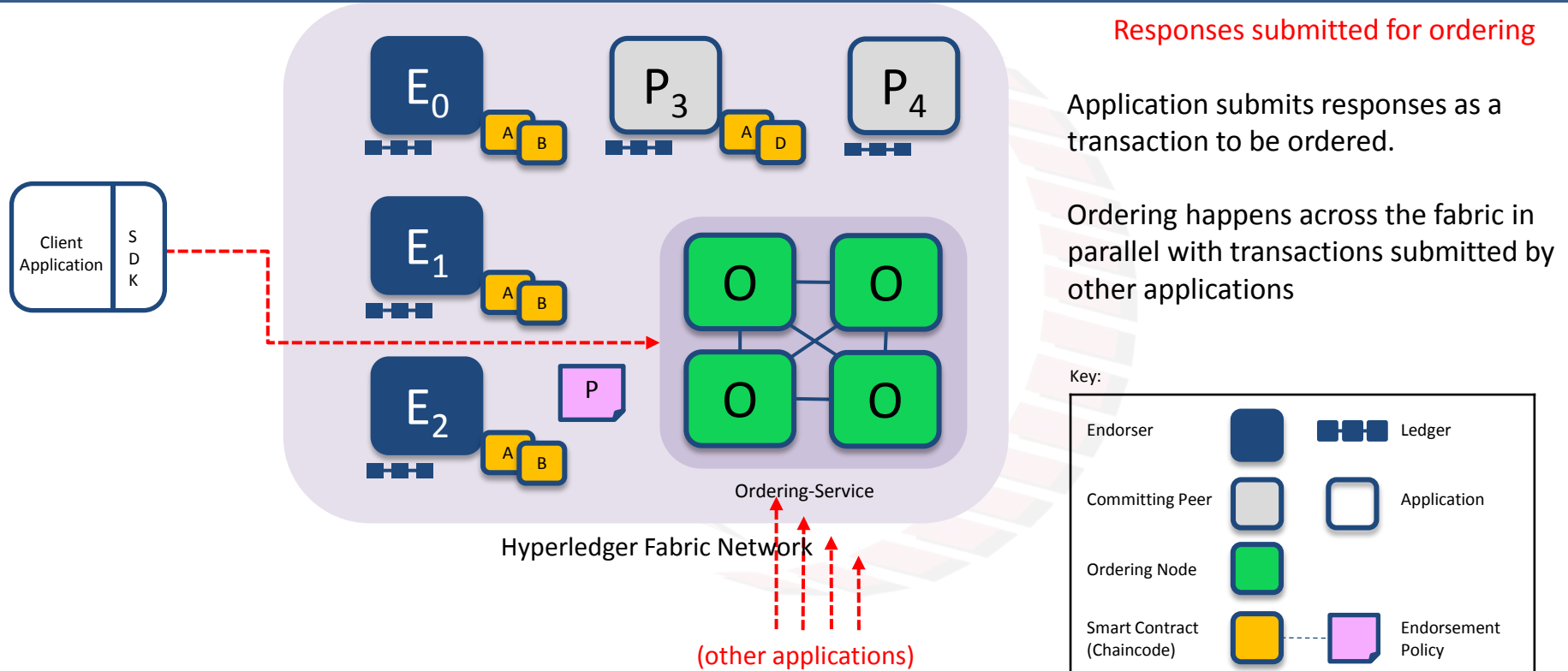
Read-Write sets are asynchronously returned to application
The RW sets are signed by each endorser, and also includes each record version number
(This information will be checked much later in the consensus process)

Key:



Hyperledger Fabric Network

Step 4/7: Order Transaction



Step 5/7: Deliver Transaction

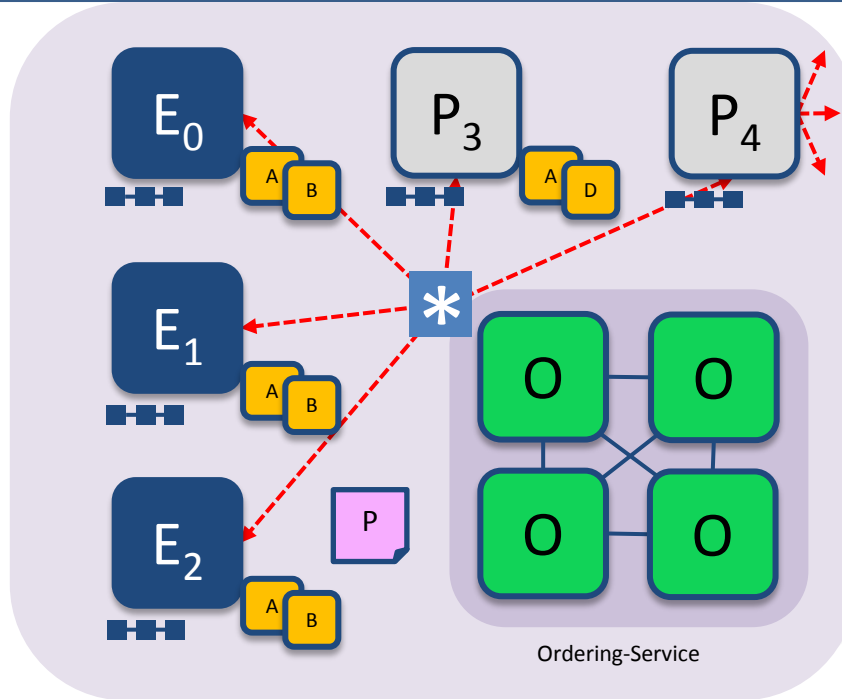
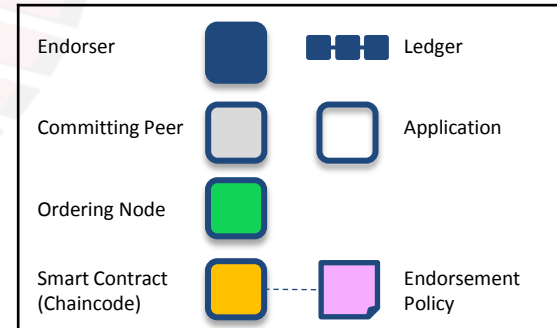
Orderer delivers to committing peers

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)

Kafka requires minimum 3 nodes

Key:



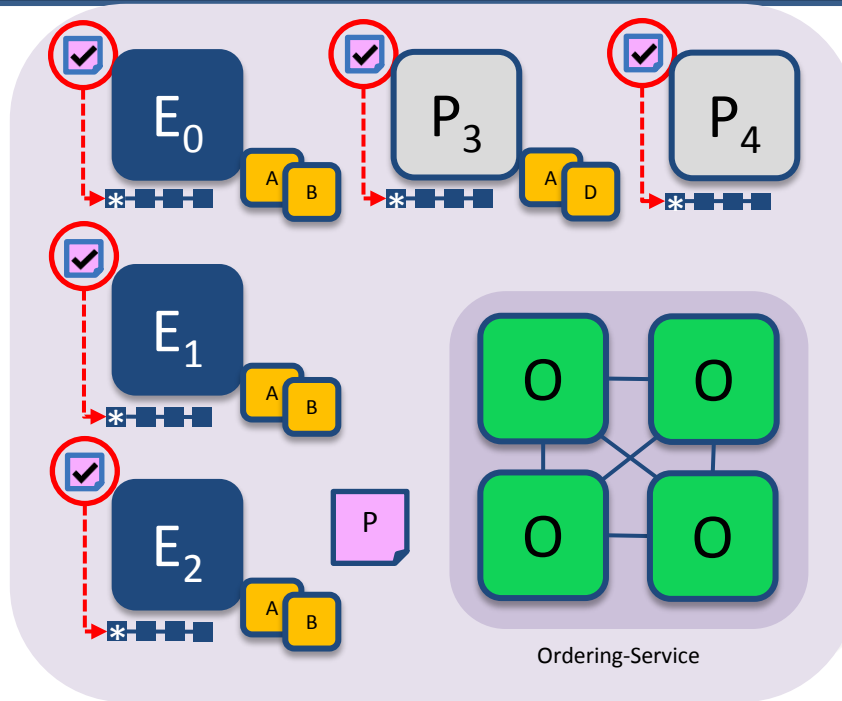
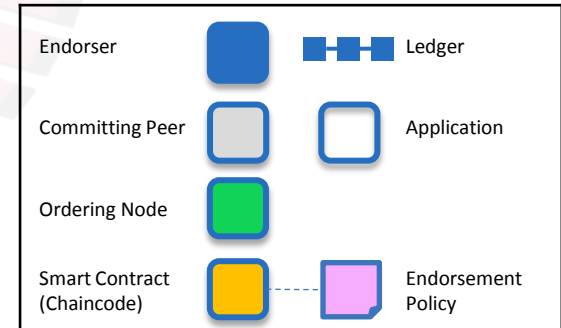
Hyperledger Fabric Network

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
Invalid transactions are also retained on the ledger but do not update world state

Key:



Hyperledger Fabric Network

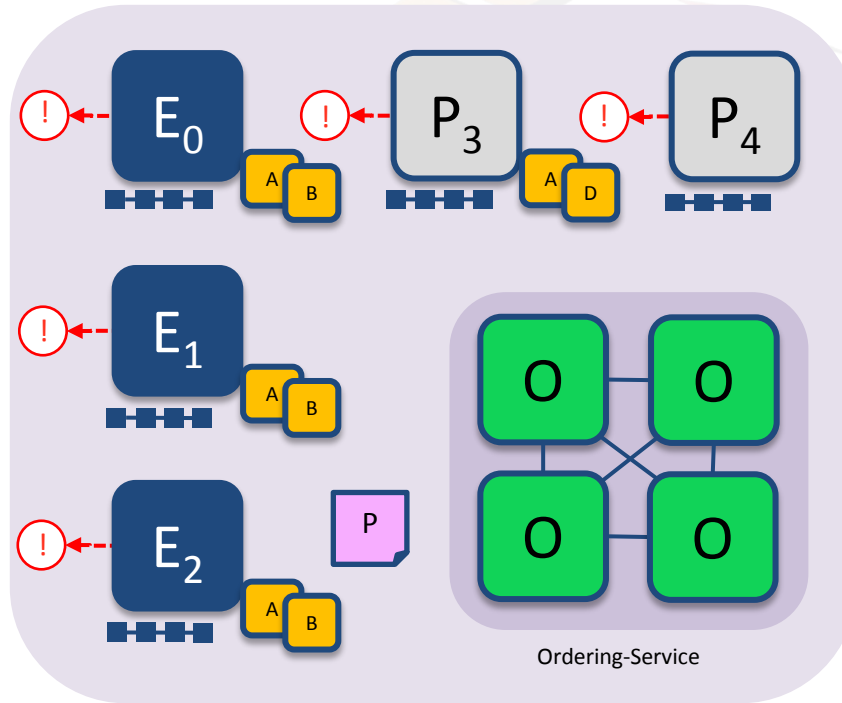
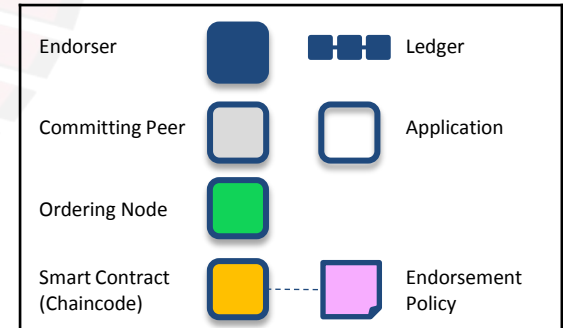
Step 7/7: Notify Transaction

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

Key Benefits of the Transaction Flow

- Better reflect business processes by specifying who endorses transactions
- Eliminate non deterministic transactions
- Scale the number of participants and transaction throughput