CSE446: Blockchain & Cryptocurrencies

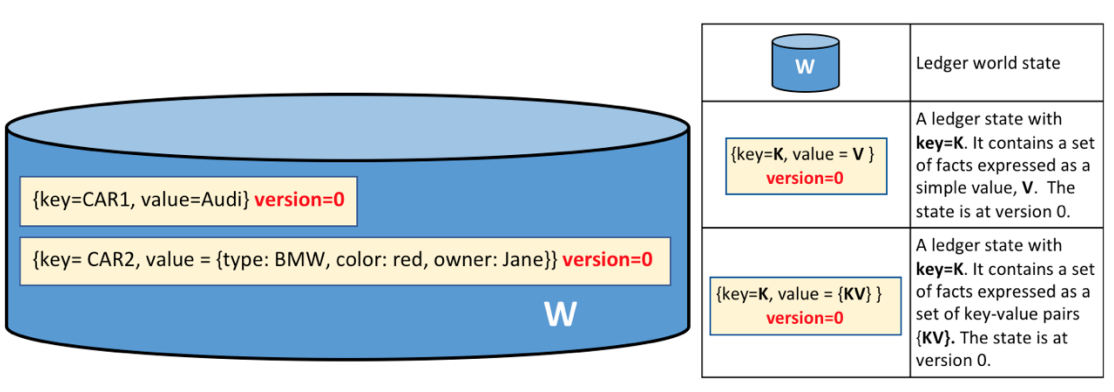
Lecture – 17: Hyperledger Fabric - 2

Agenda

- Hyperledger Fabric

Fabric network: ledger

- A ledger world state containing two types of states
- The first state is: key=CAR1 and value=Audi
- Here, the value is simple
- The second state has a more complex value:
- key=CAR2 and value={model:BMW, color=red, owner=Jane}
- Both states are at version 0
- The version number is incremented as the states are updated

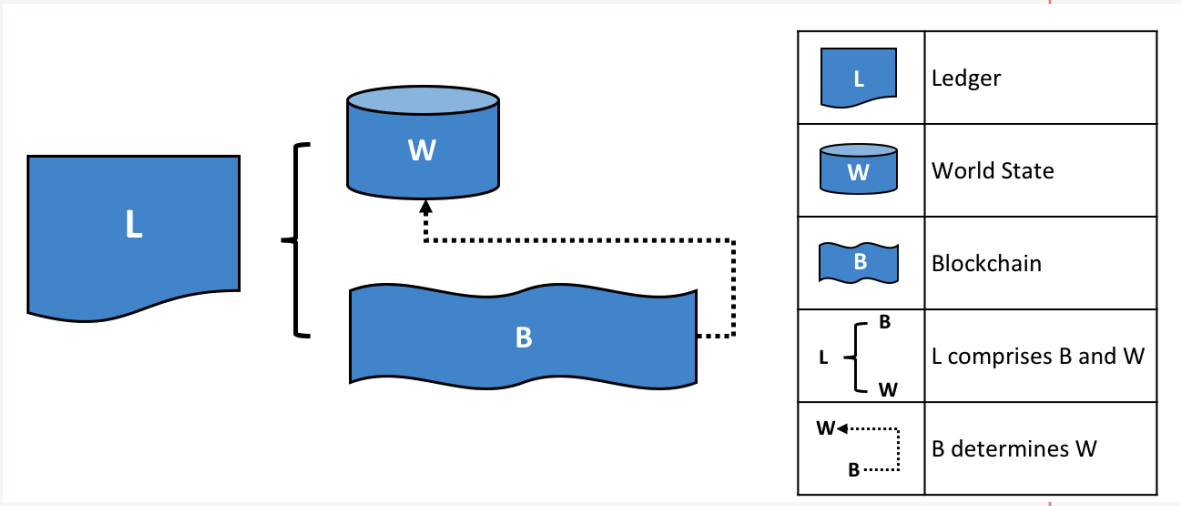


Fabric network: ledger

- At the initial stage, when a ledger is first created, the world state is empty
- When a transaction is created and it is then recorded in the blockchain
  - The world state is updated and recorded in the database

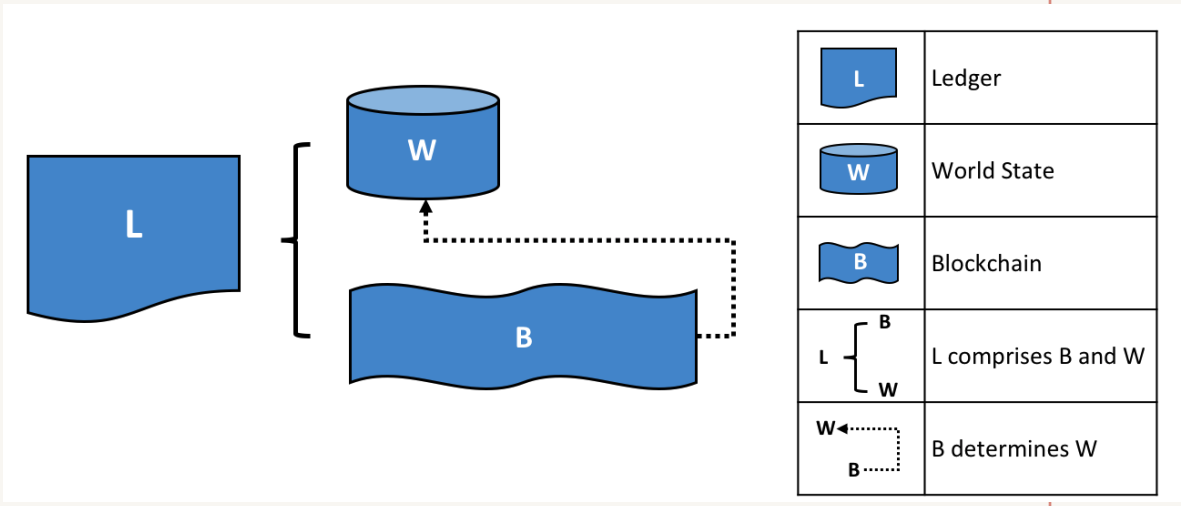
Fabric network: ledger

- The second part is the **blockchain**
- A blockchain is a transaction log that records all the changes that have resulted in the current the world state
- Transactions are collected inside blocks that are appended to the blockchain
- This enables you to understand the history of changes that have resulted in the current world state



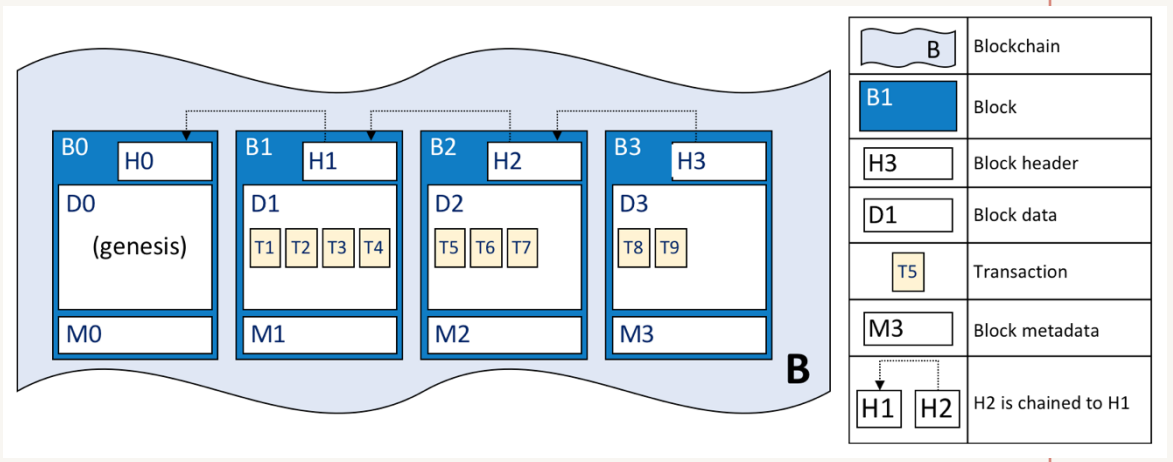
Fabric network: ledger

- The blockchain data structure is very different to the world state because once written, it cannot be modified
  - it is **immutable**
- We can also say that world state W is derived from blockchain B

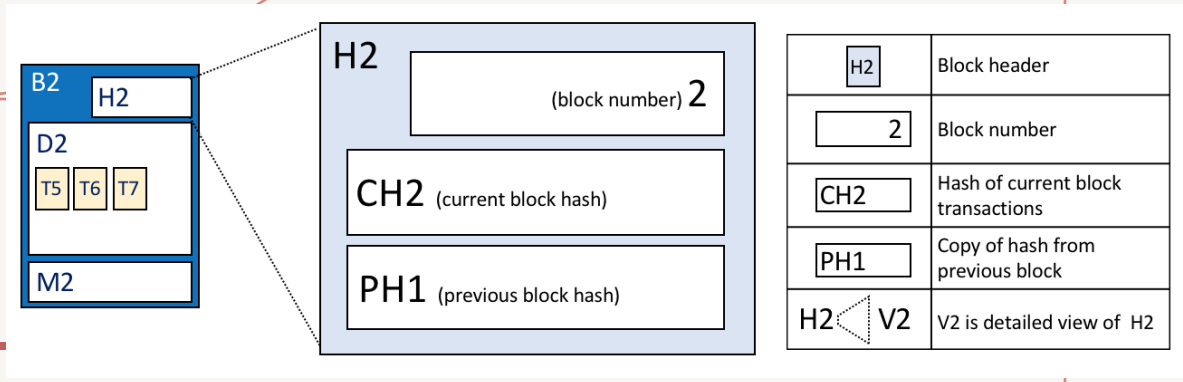


Fabric network: blocks

- A blockchain B containing blocks B0 – B3
- B0 is the first block in the blockchain, the genesis block
- We can see that block B2 has a block data D2 which contains all its transactions: T5, T6, T7
- B2 has a block header H2
- This creates the chain among the blocks

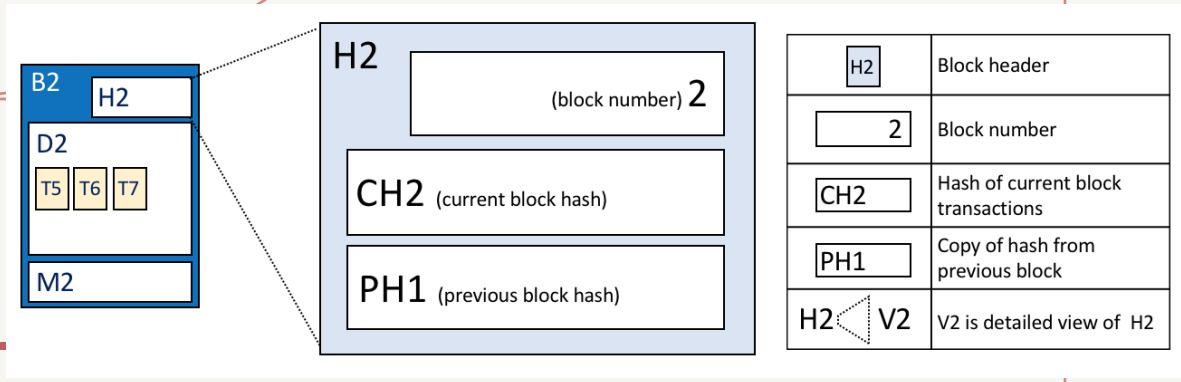


Fabric network: blocks



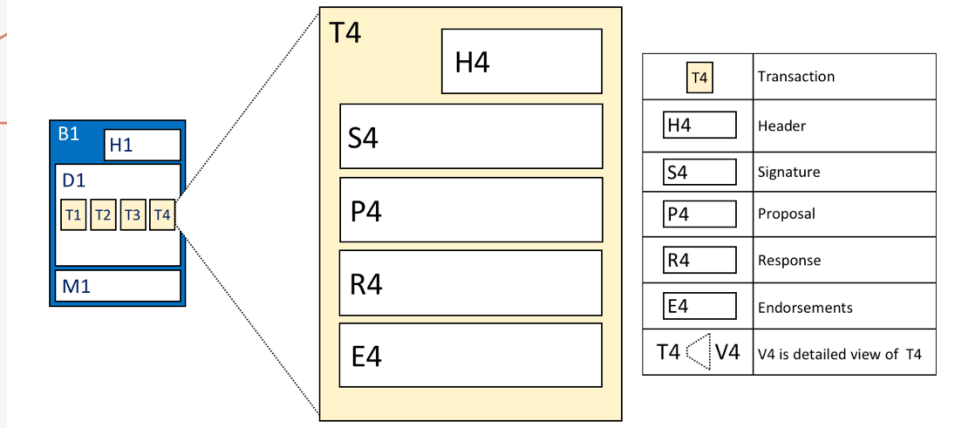
- Block Header: This section comprises three fields, written when a block is created
  - Block number: An integer starting at 0 (the genesis block), and increased by 1 for every new block appended to the blockchain
  - Current Block Hash: The hash of all the transactions contained in the current block
  - Previous Block Hash: A hash of the previous block header in the blockchain

Fabric network: blocks



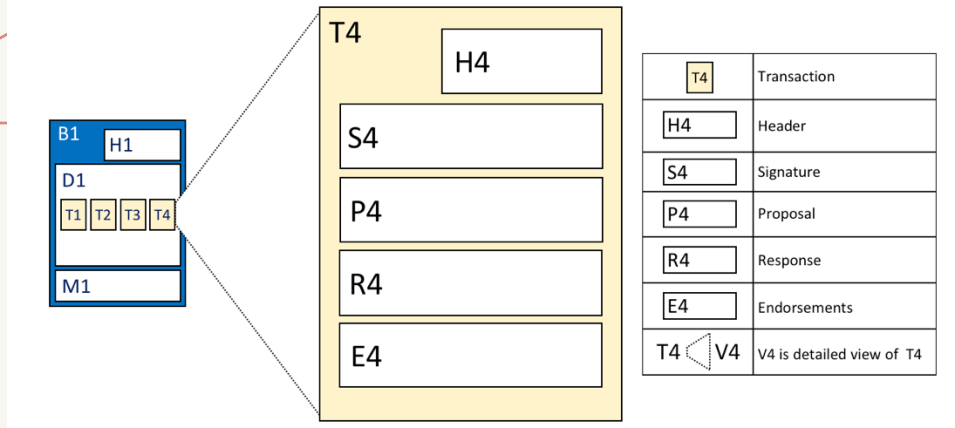
- Block Data
  - This section contains a list of transactions arranged in order
  - It is written when the block is created by the ordering service (block creation service)
- Block Metadata
  - This section contains the time when the block was written, as well as the certificate, public key and signature of the block writer
- Such metadata also contains a valid/invalid indicator for every transaction

Fabric network: transactions



- Header: This section, illustrated by H4, captures some essential metadata about the transaction
  - for example, the name of the relevant chaincode, and its version
- Signature: This section, illustrated by S4, contains a cryptographic signature, created by the client application
  - This field is used to check that the transaction details have not been tampered with, as it requires the application's private key to generate it
- Proposal: This field, illustrated by P4, encodes the input parameters supplied by an application to the smart contract which creates the proposed ledger update

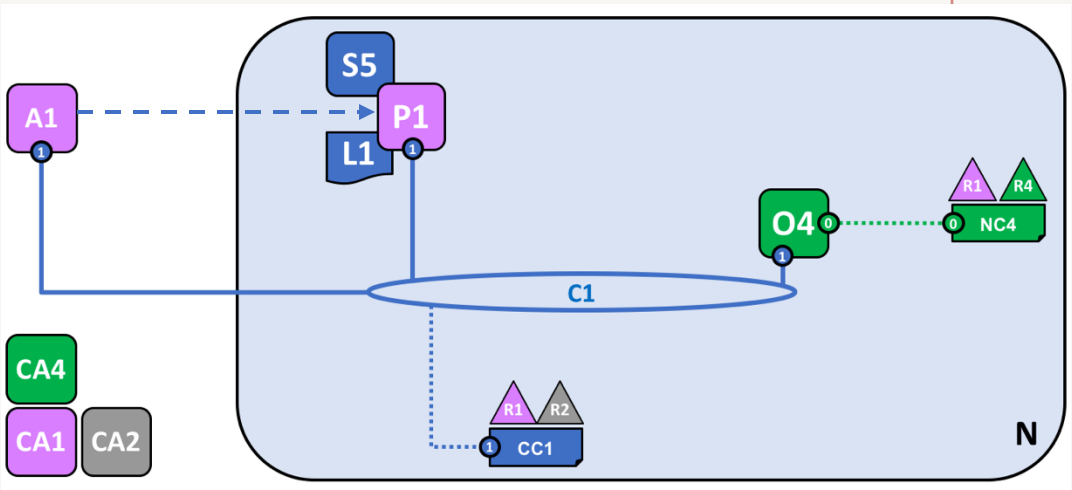
Fabric network: ledger



- Response: This section, illustrated by R4, captures the before and after values of the world state, as a Read Write set (RW-set)
  - It is the output of a smart contract
  - If the transaction is successfully validated, it will be applied to the ledger to update the world state
- Endorsements: As shown in E4, this is a list of signed transaction responses from each required organisation sufficient to satisfy the endorsement policy (explained later)

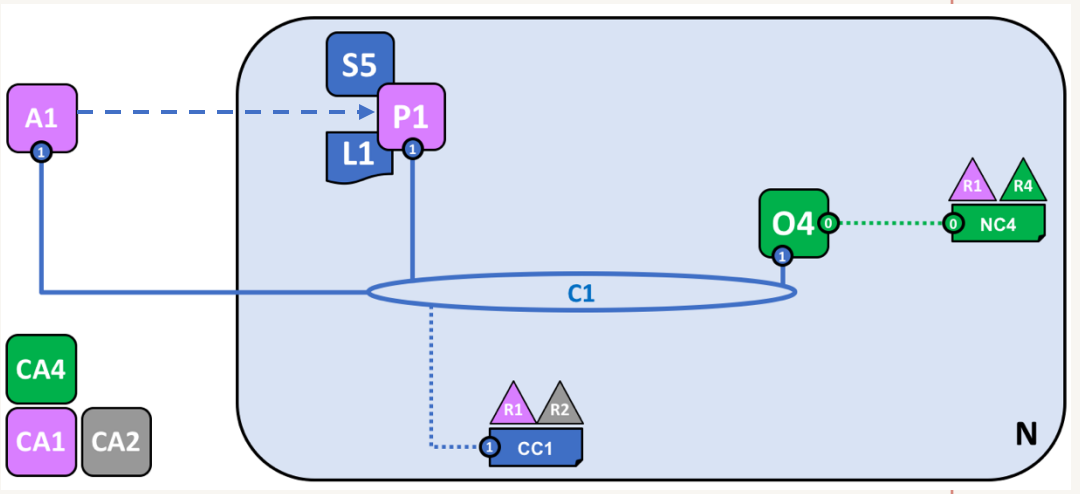
Fabric network

- Next, we add a client application to consume some of the services provided by the peer
- For this, a smart contract S5 has been installed onto P1
- Client application A1 in organisation R1 can use S5 to access the ledger L1 via peer node P1
- A1, P1 and O4 are all joined to channel C1, i.e. they can all make use of the communication facilities provided by that channel



Fabric network: installing a smart-contract

- After a smart contract S5 has been developed, an administrator in organisation R1 must install it onto peer node P1
- Specifically, P1 can see the code of S5
- Then an administrator in R1 must instantiate S5 on channel C1 using P1



Fabric network: endorsement policy

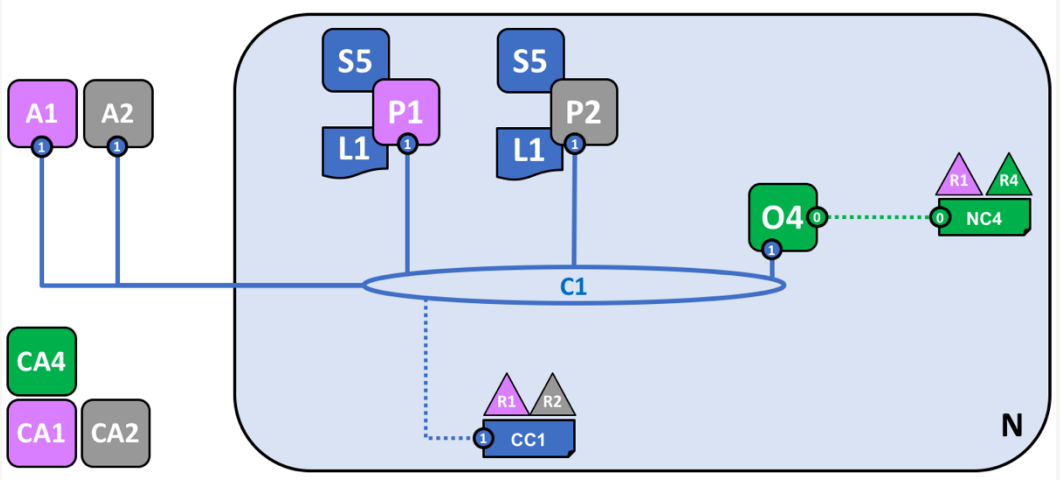
- The most important piece of additional information supplied at instantiation is an endorsement policy
- It describes which organisations must approve transactions before they will be accepted by other organisations onto their copy of the ledger
- In our sample network, let us assume that transactions can be only be accepted onto ledger L1 if R1 or R2 endorse them

Fabric network: endorsement policy

- Once a smart contract has been installed on a peer node and instantiated on a channel it can be invoked by a client application
- Client applications do this by sending transaction proposals to peers owned by the organisations specified by the smart contract endorsement policy
- The transaction proposal serves as input to the smart contract, which uses it to generate an endorsed transaction response
- The response is returned by the peer node to the client application

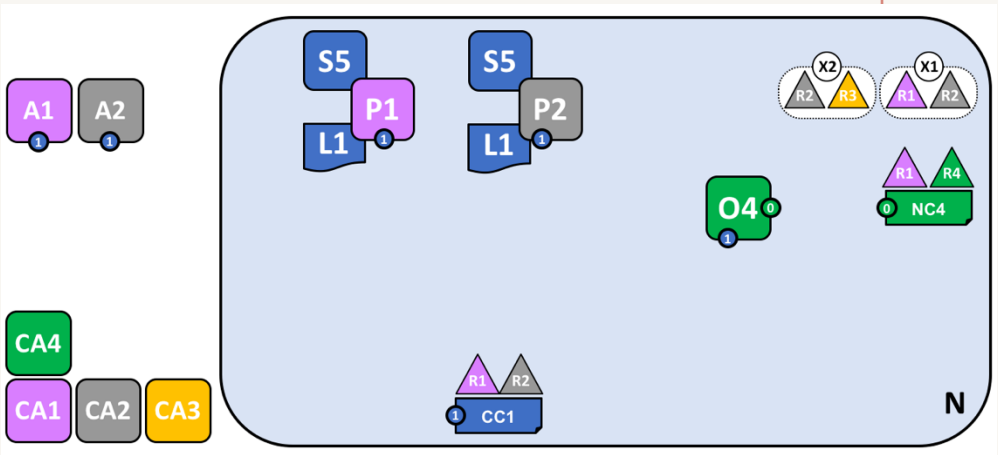
Fabric network

- The network has grown through the addition of infrastructure from organisation R2
- Specifically, R2 has added peer node P2, which hosts a copy of ledger L1, and chaincode S5
- P2 has also joined channel C1, has application A2
- A2 and P2 are identified using certificates from CA2
- All of this means that both applications A1 and A2 can invoke S5 on C1 either using peer node P1 or P2



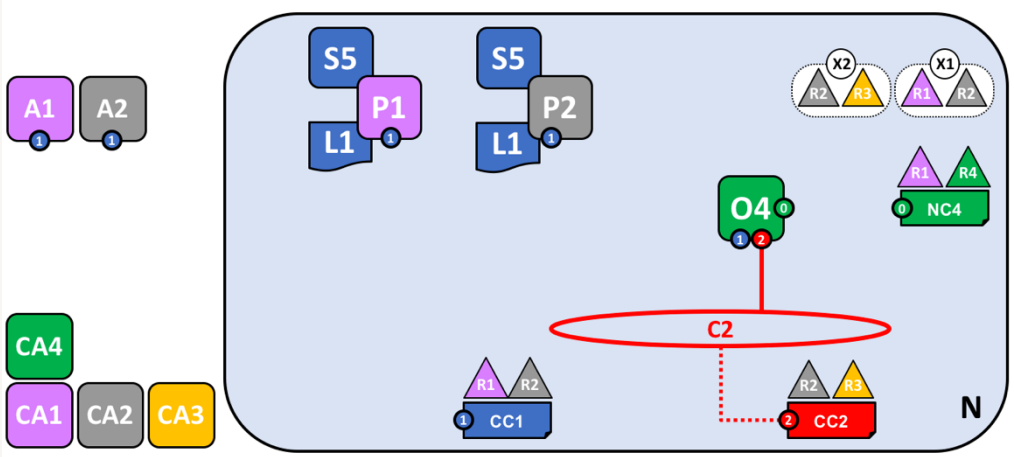
Fabric network

- In this next phase of network development, we introduce organisation R3
- A network administrator from organisation R1 or R4 has added a new consortium definition, X2, which includes organisations R2 and R3
- We're going to give organisations R2 and R3 a separate application channel for the new consortium which allows them to transact with each other
- This application channel will be completely separate to that previously defined, so that R2 and R3 transactions can be kept private to them



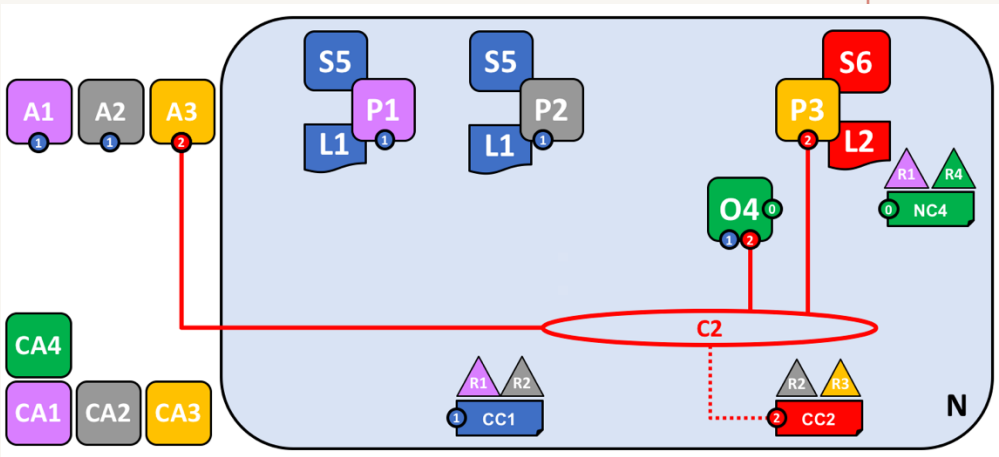
Fabric network

- A new channel C2 has been created for R2 and R3 using consortium definition X2
- The channel has a channel configuration CC2, completely separate to the network configuration NC4, and the channel configuration CC1
- Channel C2 is managed by R2 and R3 who have equal rights over C2 as defined by a policy in CC2
- R1 and R4 have no rights defined in CC2 whatsoever



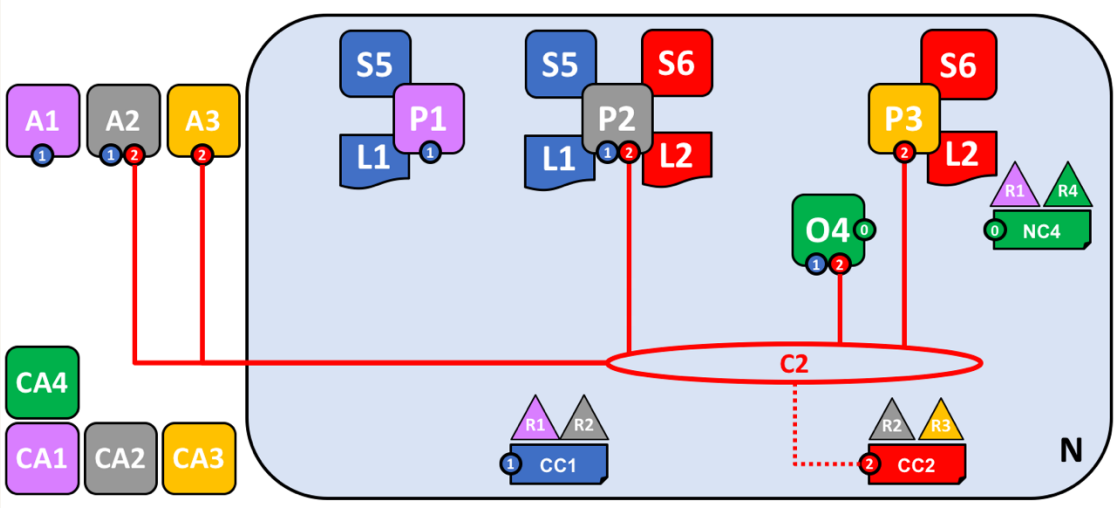
Fabric network

- Peer P3 and its corresponding application A3 have been added to the network
- Client applications A1 and A2 can use channel C1 for communication with peers P1 and P2, and ordering service O4
- Client applications A3 can use channel C2 for communication with peer P3 and ordering service O4
- Ordering service O4 can make use of the communication services of channels C1 and C2
- Channel configuration CC1 applies to channel C1, CC2 applies to channel C2



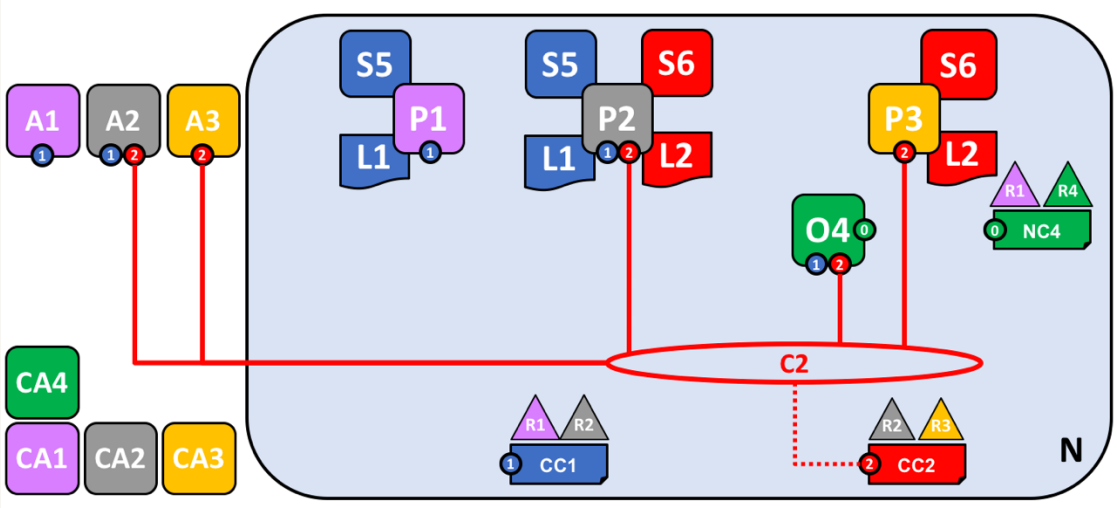
Fabric network

- P2 and A2 have also been added to C2
- Client applications A1 can use channel C1 for communication with peers P1 and P2, and ordering service O4
- Client application A2 can use channel C1 for communication with peers P1 and P2 and channel C2 for communication with peers P2 and P3 and ordering service O4



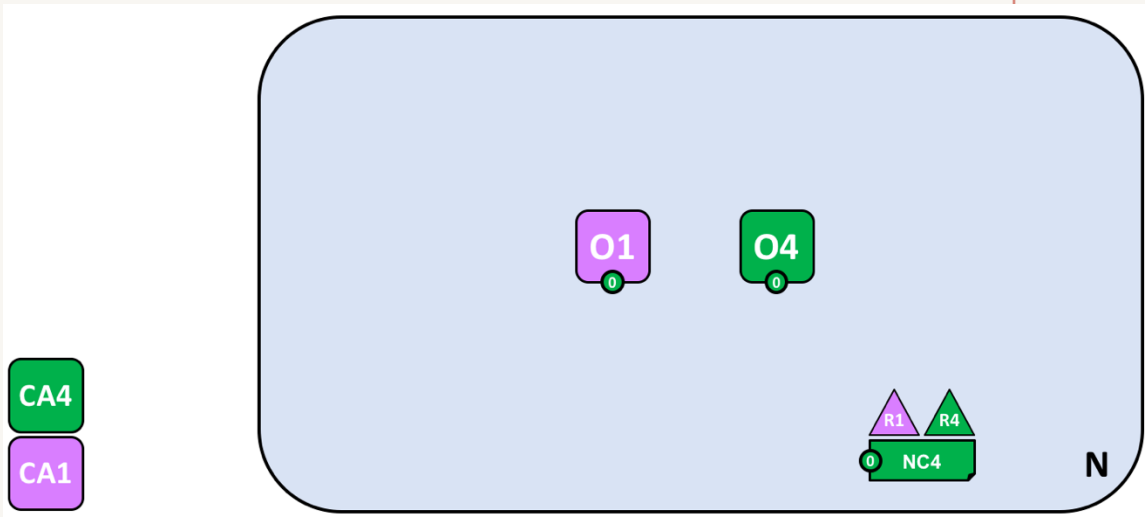
Fabric network

- Client application A3 can use channel C2 for communication with peer P3 and P2 and ordering service O4
- Ordering service O4 can make use of the communication services of channels C1 and C2
- Channel configuration CC1 applies to channel C1, CC2 applies to channel C2



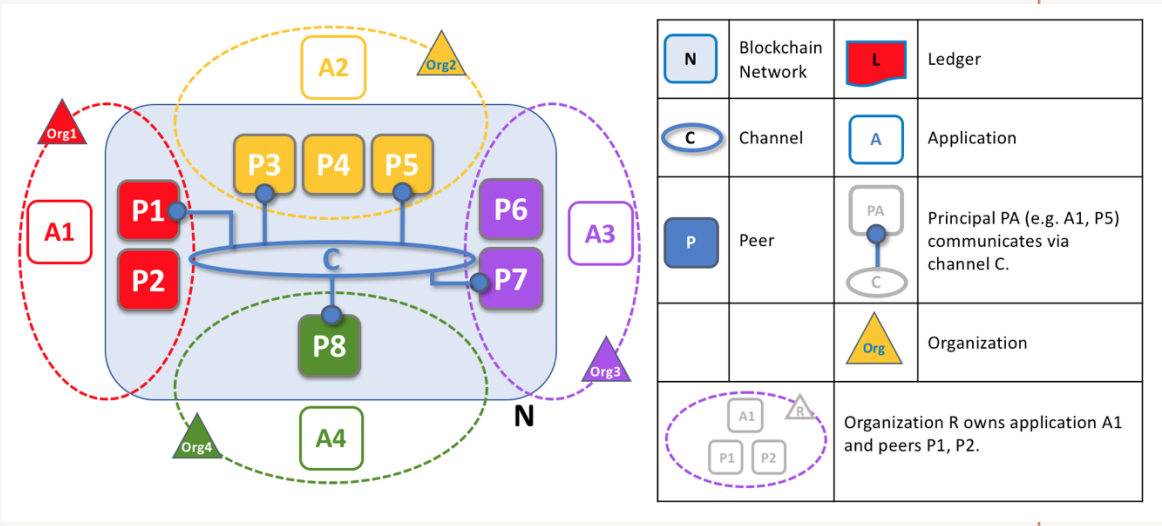
Fabric network

- A multi-organisation ordering service
- The ordering service comprises ordering service nodes O1 and O4
- O1 is provided by organisation R1 and node O4 is provided by organisation R4
- The network configuration NC4 defines network resource permissions for actors from both organisations R1 and R4



Fabric network

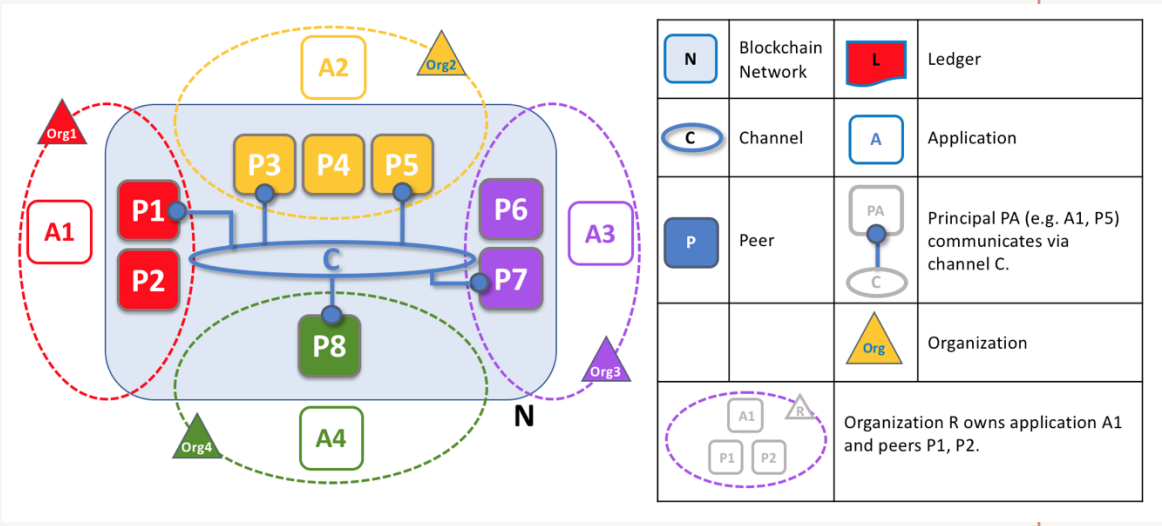
- In this example, we see four organisations contributing eight peers to form a network
- The channel C connects five of these peers in the network N – P1, P3, P5, P7 and P8
- The other peers owned by these organisations have not joined to this channel, but are typically connected to at least one other channel



Peers in a blockchain network with multiple organisations

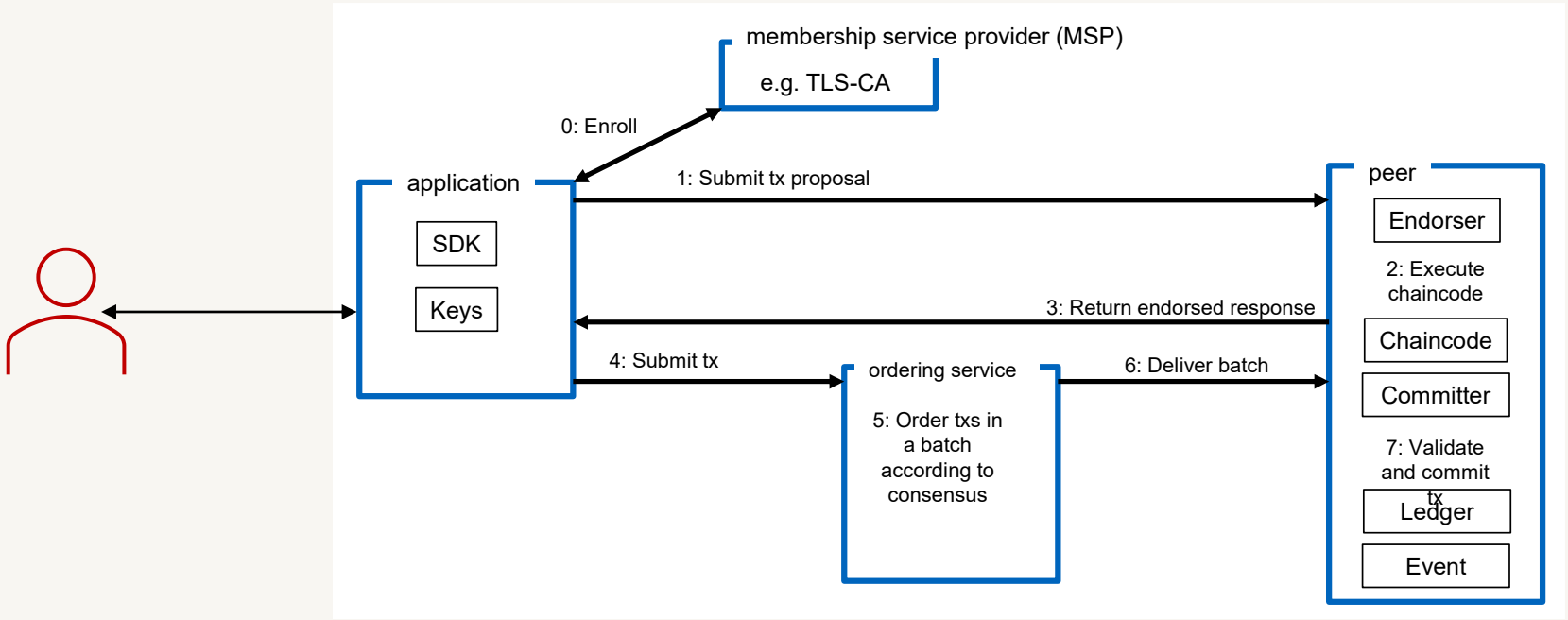
Fabric network

- Applications that have been developed by a particular organisation will connect to their own organisation's peers as well as those of different organisations
- For simplicity, an orderer node is not shown in this diagram



Peers in a blockchain network with multiple organisations

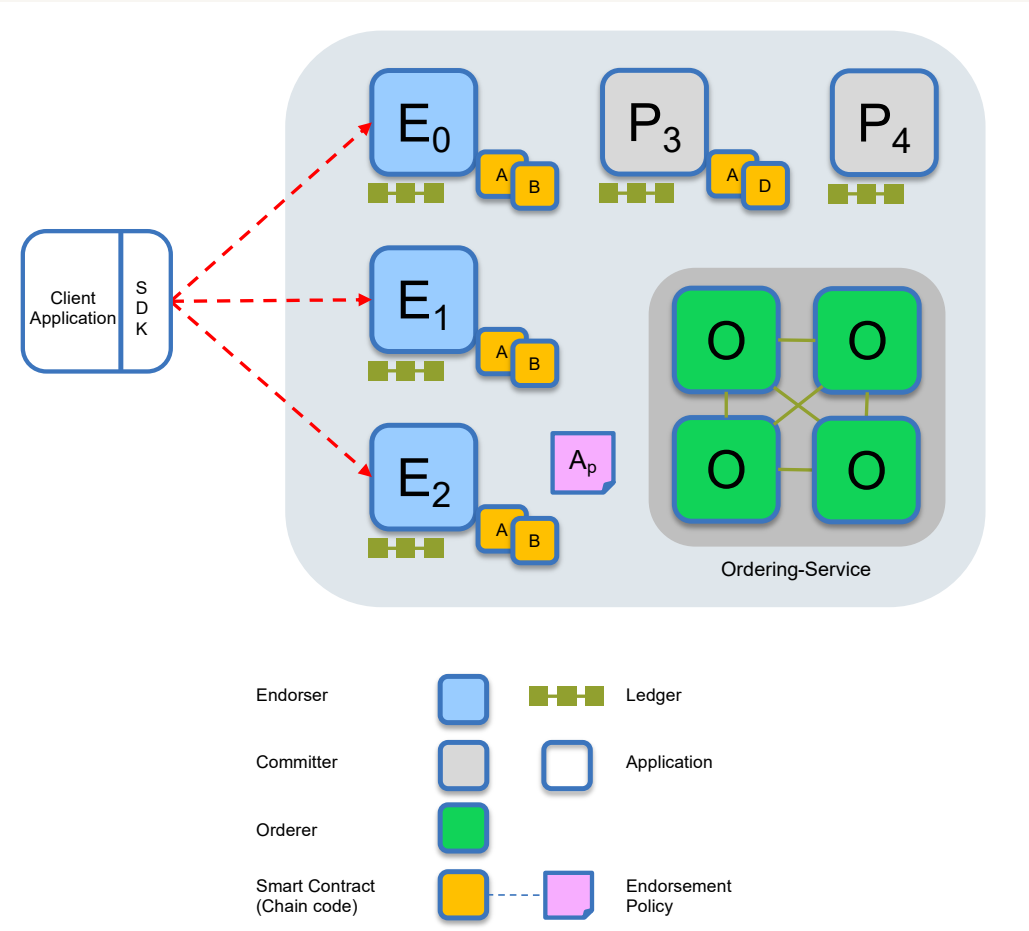
Transaction flow



In Fabric, a transaction initiates a seven step process from **simulation** of the executed chaincode (endorsement) to the **generation** of a read/write set which is then broadcasted to the ordering service and finally included in a new block (**committing**)

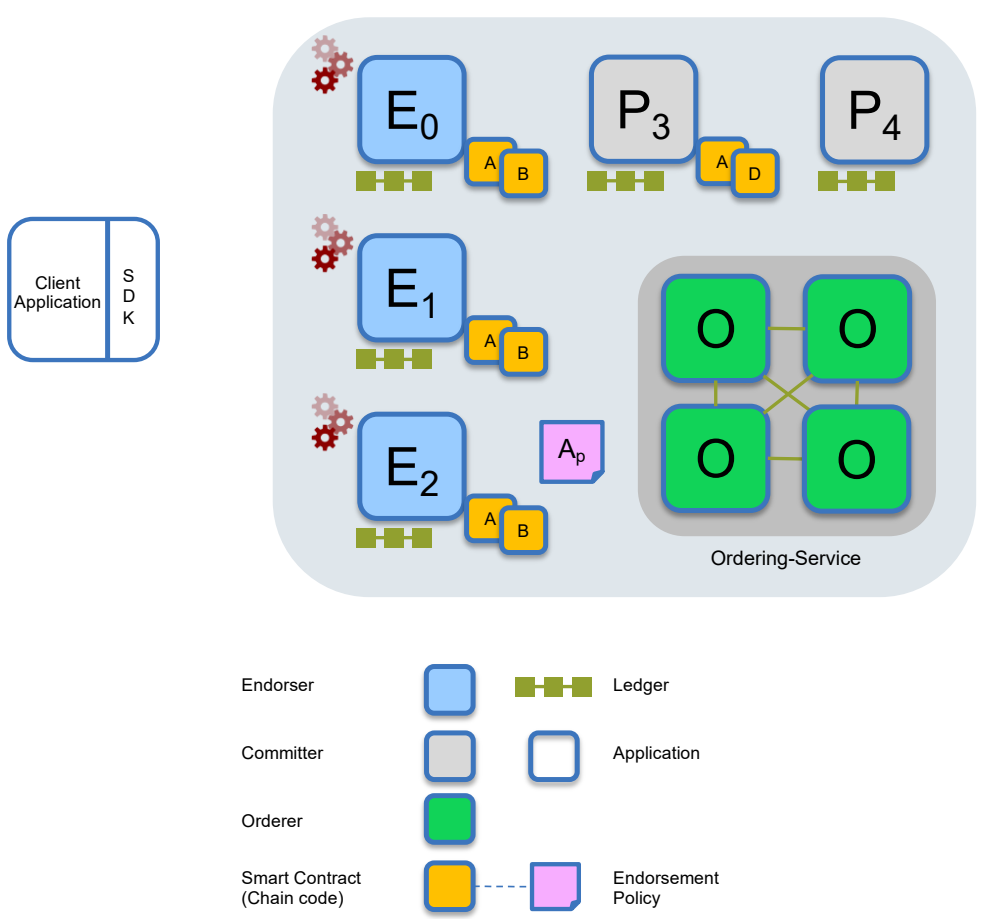
Transaction flow: steps 1/7

- Application creates a transaction proposal for **chaincode A** and sends it to all peers that are part of the endorsement policy **Ap**
- Endorsement Policy Ap
  - E_0, E_1 and E_2 must sign the transaction
  - P_3 and P_4 are not part of the policy
- Since only the peers **E_0**, **E_1** and **E_2** are part of the endorsement policy **Ap,** it is not required to send the transaction proposal to **P_3** and **P_4**



Transaction flow: steps 2/7

- E0, E1 and E2 will each simulate the execution of the *proposed* transaction from the application
- None of these executions will update the ledger
- The simulation will be used to capture the read and write operations on the ledger
- After the transaction is executed, each peer will have a generated read/write set (RW set)

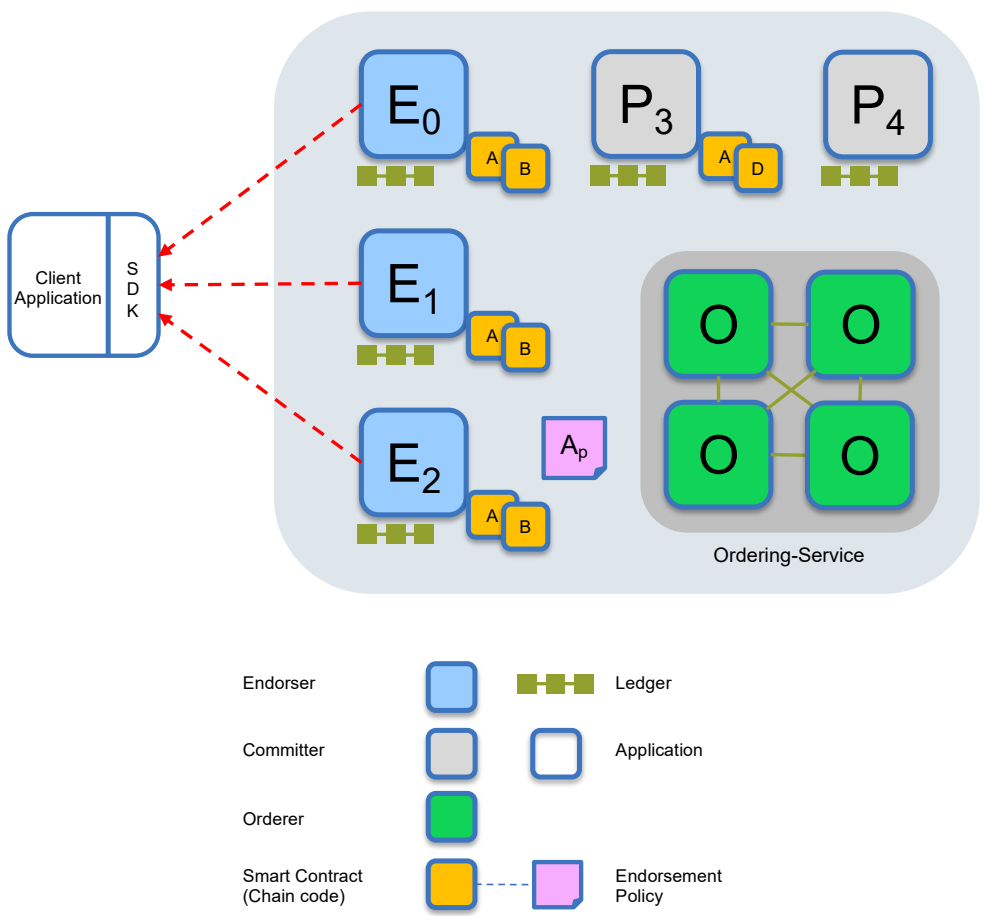



```
<TxReadWriteSet>
  <NsReadWriteSet name="chaincode1">
    <ReadSet>
      <read key="K1", version="1">
      <read key="K2", version="1">
    </ReadSet>
    <WriteSet>
      <write key="K1", value="V1">
      <write key="K3", value="V2">
      <write key="K4", isDelete="true">
    </WriteSet>
  </NsReadWriteSet>
<TxReadWriteSet>
```

Source: https://www

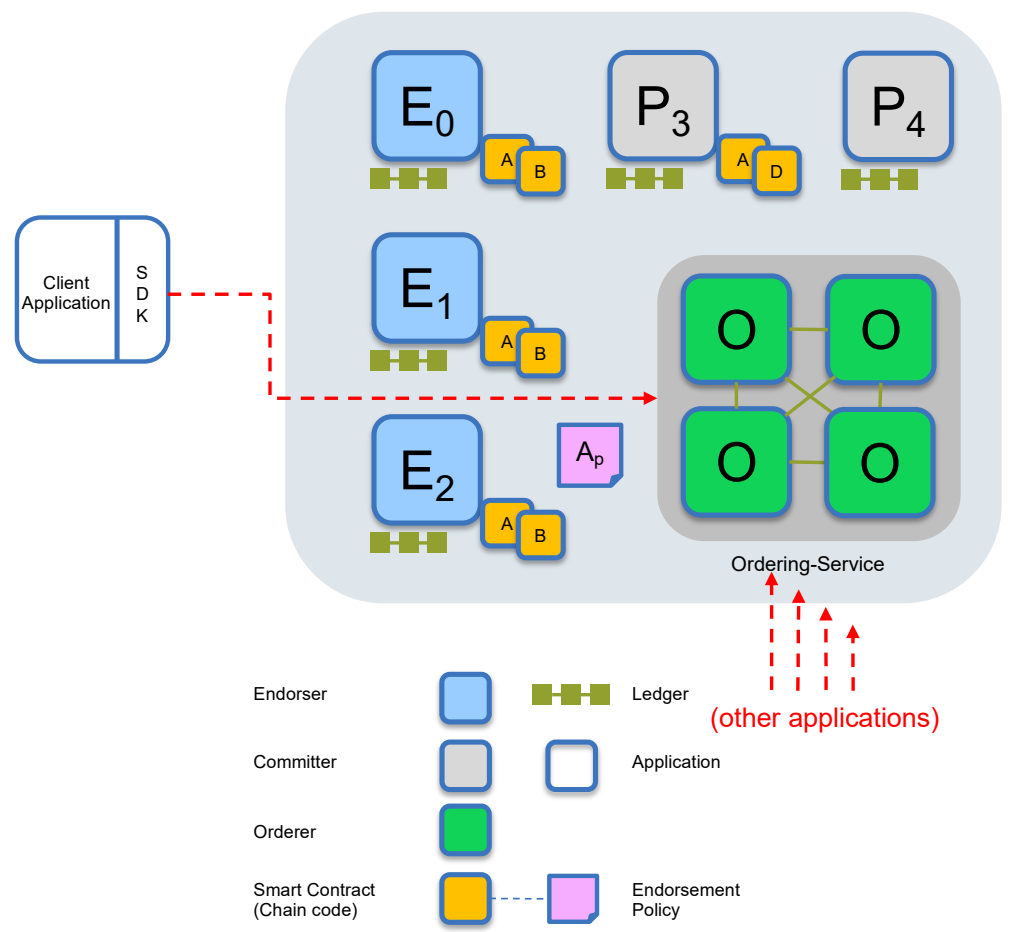
Transaction flow: steps 3/7

- **E_0**, **E_1** and **E_2** will each sign their generated read/write set and return it to the application that invoked the transaction



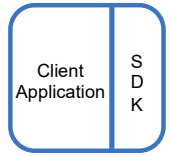
Transaction flow: steps 4/7

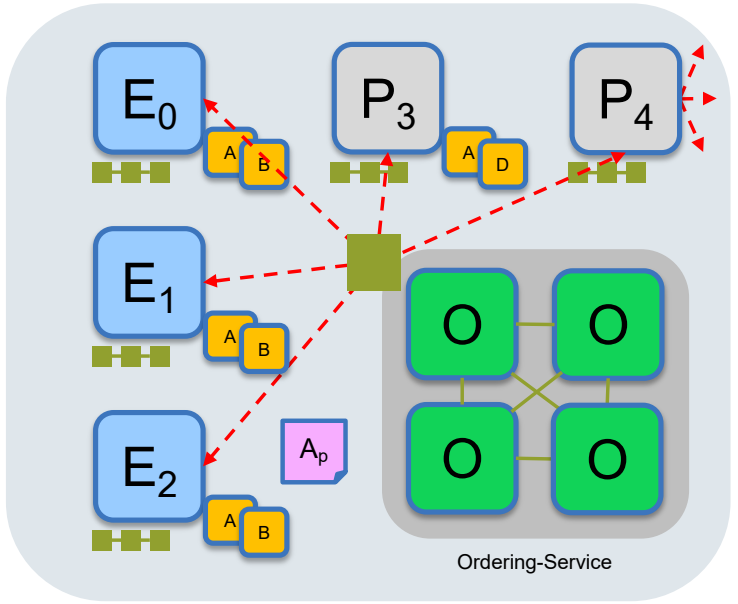
- The application submits the signed responses from **E0**, **E1** and **E2** to the ordering service
- The ordering service is responsible to order all transactions from all applications in the network
- The service tries to **serialize** the incoming transactions

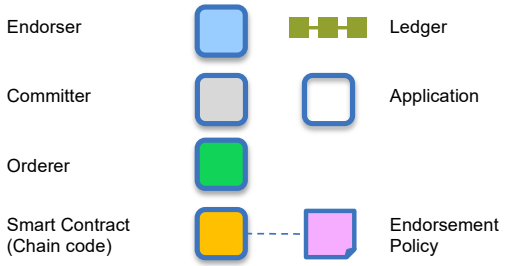


Transaction flow: steps 5/7

- The ordering service creates a new block based on the incoming transactions
- The block will then be broadcasted to all committing peers in the channel
- Currently, the ordering service supports three different ordering algorithms:
  - SOLO (single node, development)
  - Kafka (blocks map to topics)
  - Raft (crash fault tolerant (CFT), follows a “leader and follower” model)

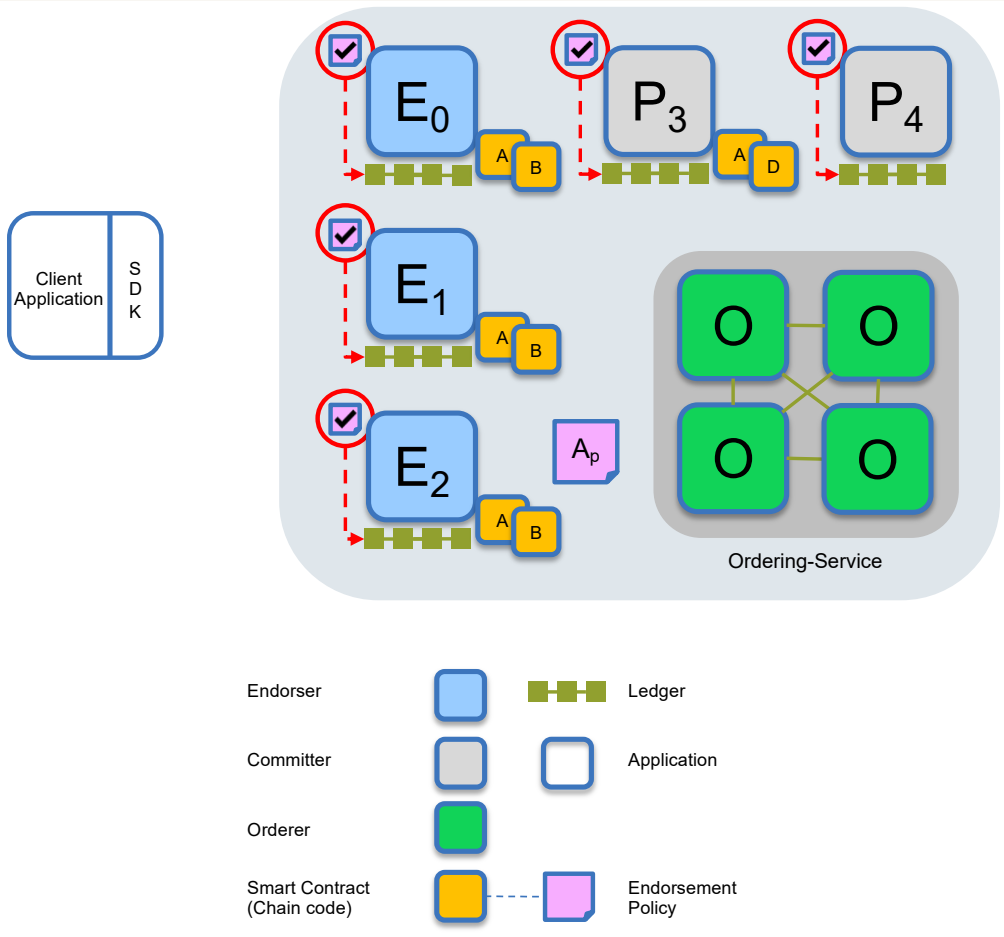






Transaction flow: steps 6/7

- All committing peers in the channel validate the transaction (read/write set) according to the endorsement policy of the chaincode A
- If the transaction is valid, the read and write set is written to the ledger and added as a new block to the blockchain
- The databases used for caching are updated with the new state information accordingly

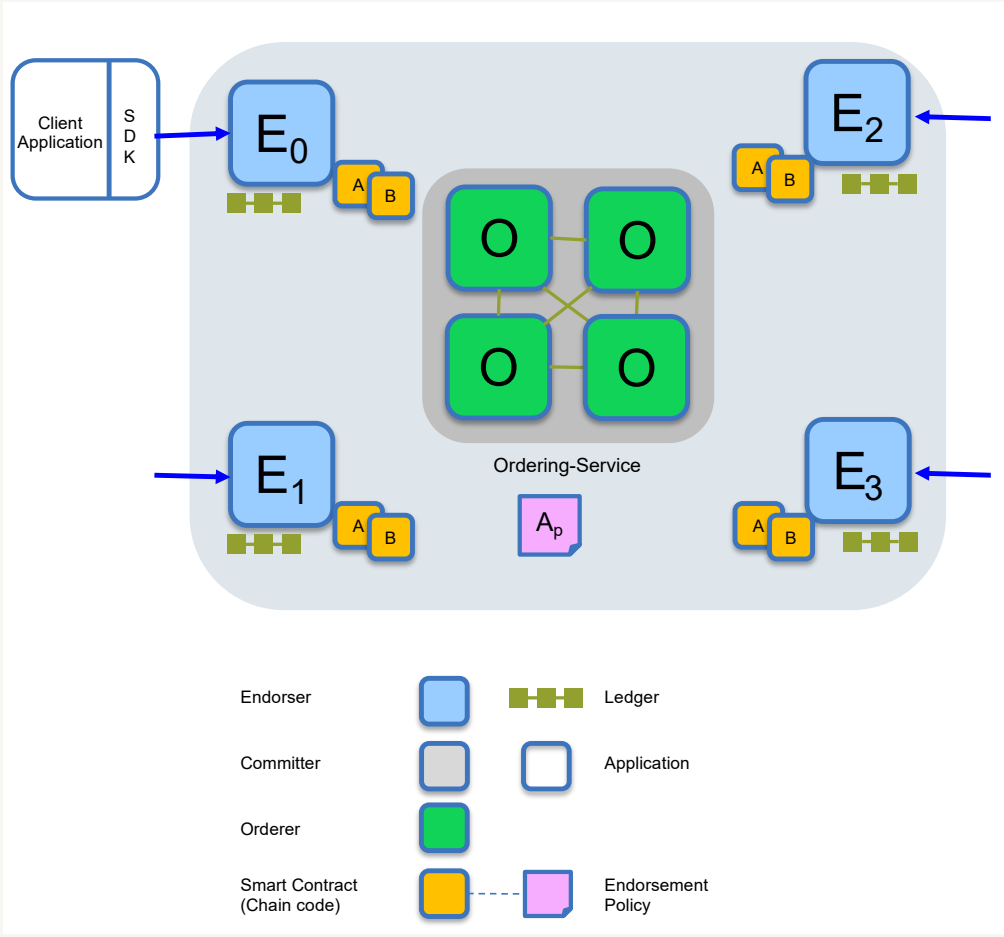


Channel

- A channel is a separate blockchain
- This blockchain is only managed by a subset of all available nodes as defined by the membership service provider
- Separate channels isolate transactions on different ledgers
- Other members on the network are not allowed to access the channel and will not see transactions on the channel
- A chaincode may be deployed on multiple channels with each instance isolated within its channel
- Peers can participate on multiple channels

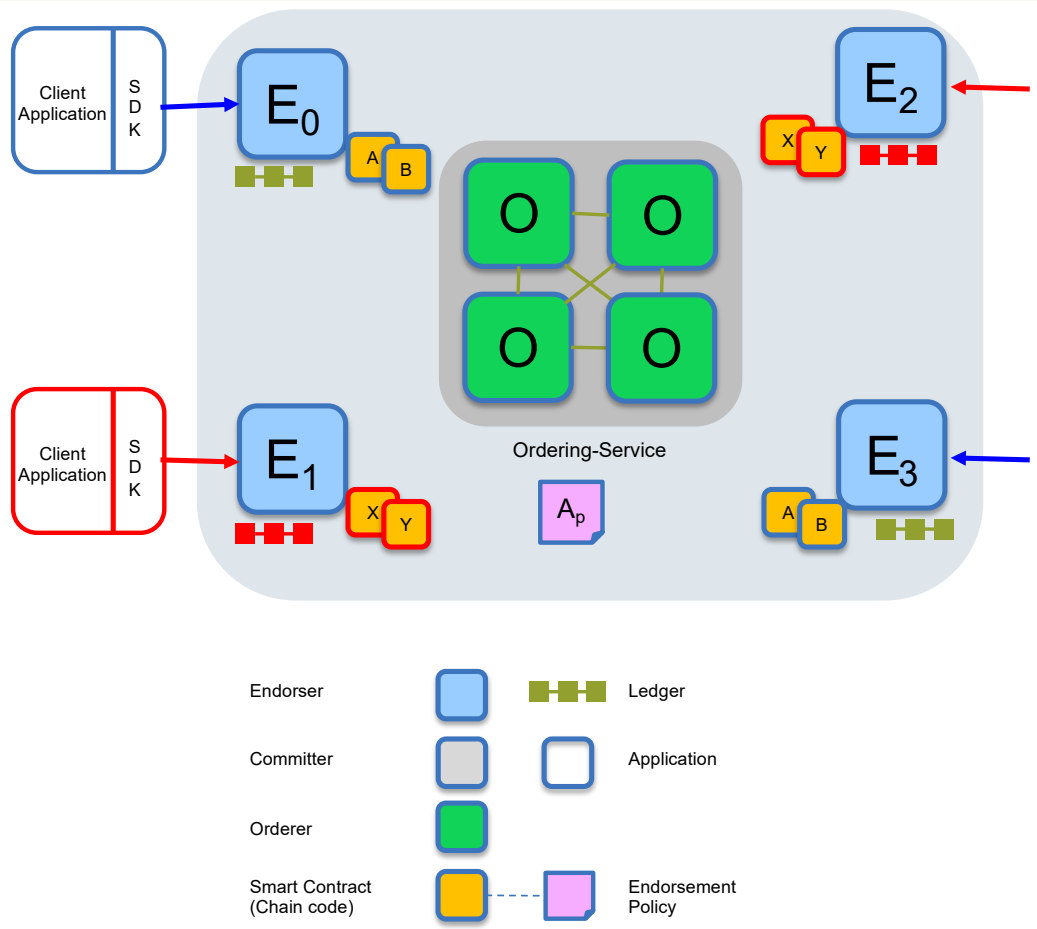
Single channel network

- All peers connected to the same channel (blue)
- All peers have the same chaincode and maintain the same ledger endorsed by peers E_0, E_1, E_2, and E_3
- A single channel network is similar to traditional, public blockchain network where the whole world state is shared with all participating nodes



Multi channel network

- Fabric support multi channel networks
- Each peer only shares the ledger with nodes that are in the same channel
- Smart contracts also operate on a channel basis and are not globally available
- Peers E_1 and E_2 connect to the red channel for X and Y chaincodes
- Peers E_0 and E_3 connect to the blue channel for A and B chaincodes



Question?