Introduction to Hyperledger

Carlos Castro-Iragorri

Blockchain technologies

- Cryptographic security, identification.
- P2P networks.
- Shared and decentralized digital registry.
- Consensus mechanism, guarantees consistency across the nodes in the network.
- Validation rules (users and transactions).
- Virtual Machines (VM)

Blockchain technologies

Consensus

PoW, PoS, POET, RaFT, BFT, PBFT

Crypto/Security

PKI, HASH, SHA-256, zk-SNARK, HE, ECC, EXDSA, SGX

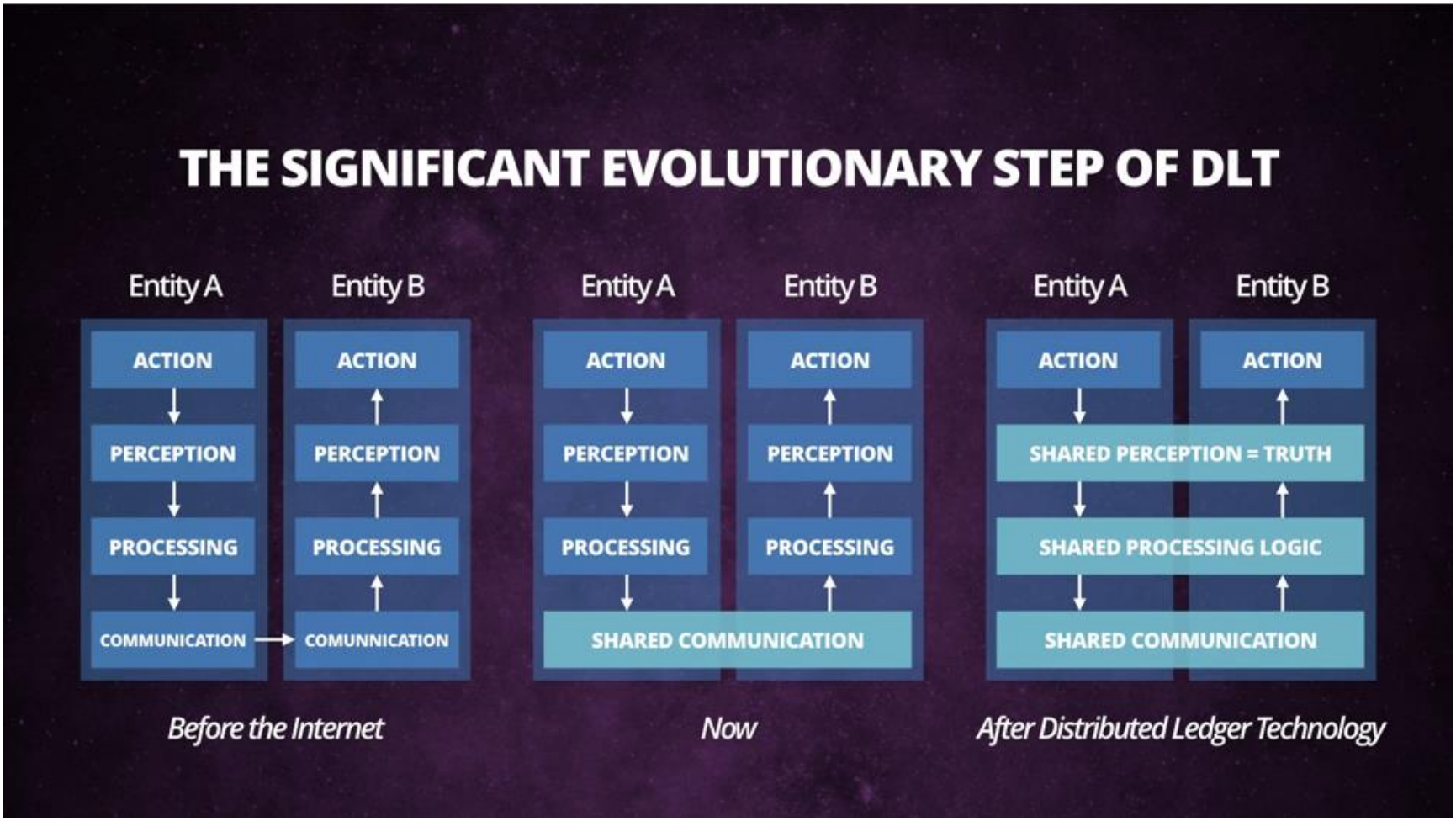
Ledger Concepts

Mining, Blocks, Forks, Parents, Uncles, Merkle Trees

Platform Concepts

Nodes, Oracles, Notaries, Wallet, Smart Contracts

DLT/Blockchain



HYPERLEDGER PROJECT

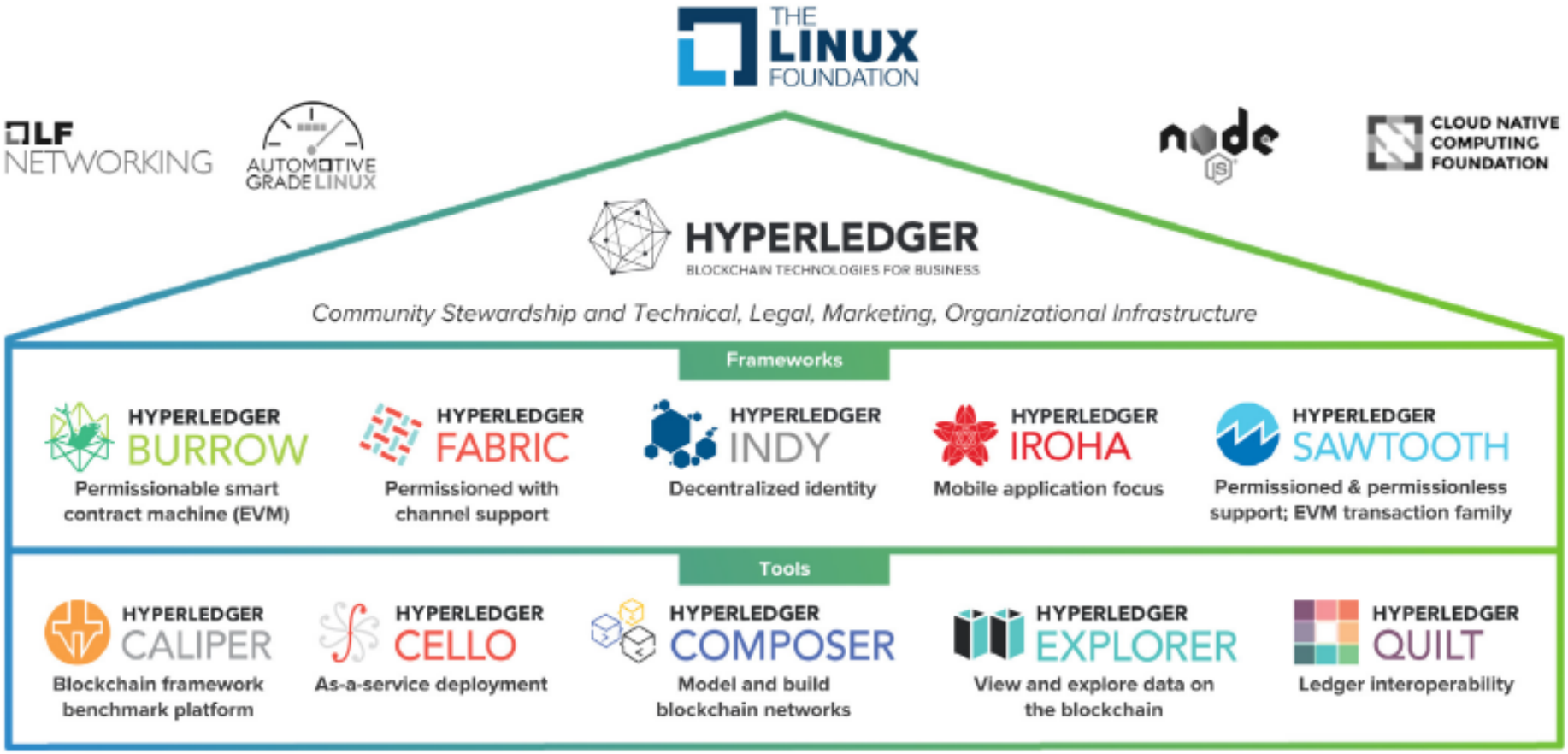
WWW.HYPERLEDGER.ORG/

Hyperledger Project

- Linux foundation
- Open Source
- Contains:
  - Infrastructure: ecosystem to accelerate development and commercial adoption.
  - Frameworks: a portfolio of solutions (business blockchain) contributed by the members.
  - Tools: facilitate development.

Hyperledger Projects: Design Philosophy

Green house structure



Frameworks

HYPERLEDGER BURROW	A modular blockchain client with a permissioned smart contract interpreter developed in part to the specifications of the Ethereum Virtual Machine (EVM).
HYPERLEDGER FABRIC	A platform for building distributed ledger solutions with a modular architecture that delivers a high degree of confidentiality, flexibility, resiliency, and scalability. This enables solutions developed with Fabric to be adapted for any industry.
HYPERLEDGER INDY	A distributed ledger that provides tools, libraries, and reusable components purpose-built for decentralized identity.
HYPERLEDGER IROHA	A blockchain framework designed to be simple and easy to incorporate into enterprise infrastructure projects.
HYPERLEDGER SAWTOOTH	A modular platform for building, deploying, and running distributed ledgers. Sawtooth features a new type of consensus, proof of elapsed time (PoET) which consumes far fewer resources than proof of work (PoW).

Tools

HYPERLEDGER CALIPER	A blockchain benchmark tool that measures the performance of any blockchain by using a set of predefined use cases.
HYPERLEDGER CELLO	A set of tools to bring the on-demand deployment model to the blockchain ecosystem with automated ways to provision and manage blockchain operations that reduce effort.
HYPERLEDGER COMPOSER	An open development toolset and framework to make developing blockchain applications easier.
HYPERLEDGER EXPLORER	A dashboard for viewing information on the network, including blocks, node logs, statistics, smart contracts, and transactions.
HYPERLEDGER QUILT	A set of tools that offer interoperability by implementing ILP, which is primarily a payments protocol designed to transfer value across distributed and non-distributed ledgers.

WHAT MAKES HYPERLEDGER DIFFERENT?

Types of blockchains: permissions

- Permission less, public.
- Permissioned, private.
- With respect to writing and reading privileges on the registry.

Permission less, public Blockchain

- Transactions are processed by all the nodes
- Transactions are completely visible (public reading)
- Large number of nodes
  - Ethereum: 13,978
  - Bitcoin: 9,563.
- Benefits: public writing and reading, distributed registry resistant to censorship, the authenticity of the registry is guaranteed by the “controlling” mining rule (>51%)

Permissioned, private Blockchain

- Transactions are processed by some of the nodes (node specialization).
- Transactions may be visible or private
- Locally distributed network across different organizations.
- Benefits: firms or institutions that want to keep control over their information and transactions, faster transaction processing, better scalability, efficient consensus.

Types of Blockchain(s)

	Public (Permissionless)	Private (Permissioned)
Access to Ledger	Open Read/Write	Permissioned Read/Write
Identity	Anonymous	Known Identities
Security and Trust	Open Network (Trust Free)	Controlled Network(Trusted)
Transaction Speed	Slower	Faster
Consensus	POW/POS	Proprietary or Modular
Open Source	Yes	Depends on Blockchain
Code Upkeep	Public	Consortium or Managed
Examples	Ethereum, Multichain	R3 Corda, Quantum, Hyperledger

Consensus

- A mechanism to reach an agreement within a group.
- In blockchain the consensus is with regards to the information kept in the registry, “World State”.
- The consensus mechanism is based on rules and incentives.
- There are various designs for these types of mechanisms.

Hyperledger modular consensus

- Plenum Byzantine Fault Tolerance (PBFT):
  - In PBFT each node maintains a copy of the registry.
  - Each node receives and message and signs it to manifest authenticity.
  - Once a sufficient number of messages have been received with the same characteristics then consensus is reached and a transaction is valid.

Hyperledger consensus

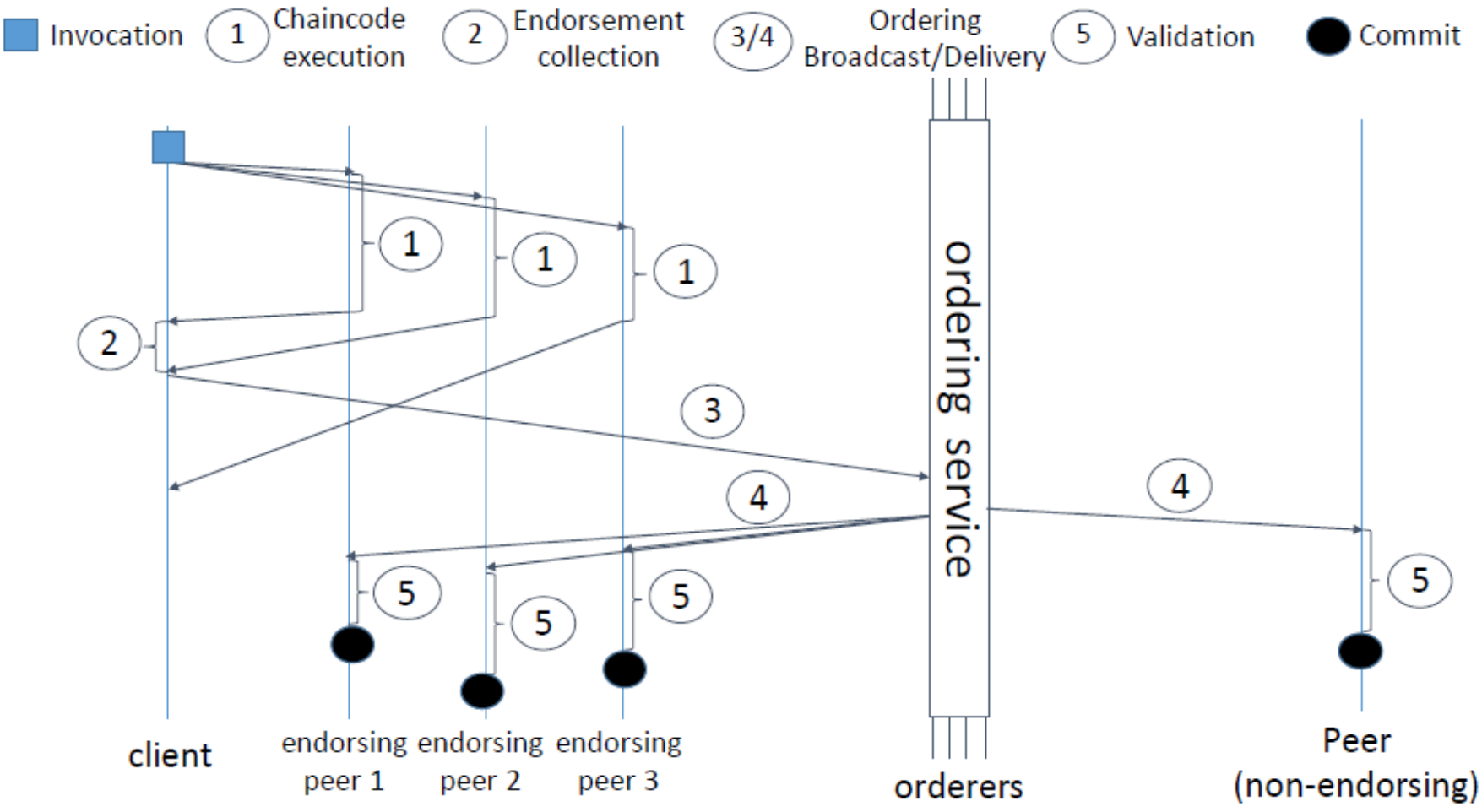
- Hyperledger uses a consensus mechanism similar to voting, where signed messages are received from participating.
- Consensus algorithms based on voting systems can process a large number of messages with low-latency.
- However, there is a *trade-off* between scalability and performance; more participating nodes (voting), requires a longer time to reach a consensus.

Hyperledger Consensus

Three steps :

- Endorsement: rule: m out of n signatures are needed to support a transaction.
- Ordering: collects all supported transactions and determines the order of the transactions in the registry.
- Validation: analyzes and validates the blocks.

Hyperledger Consensus



HYPERLEDGER FABRIC

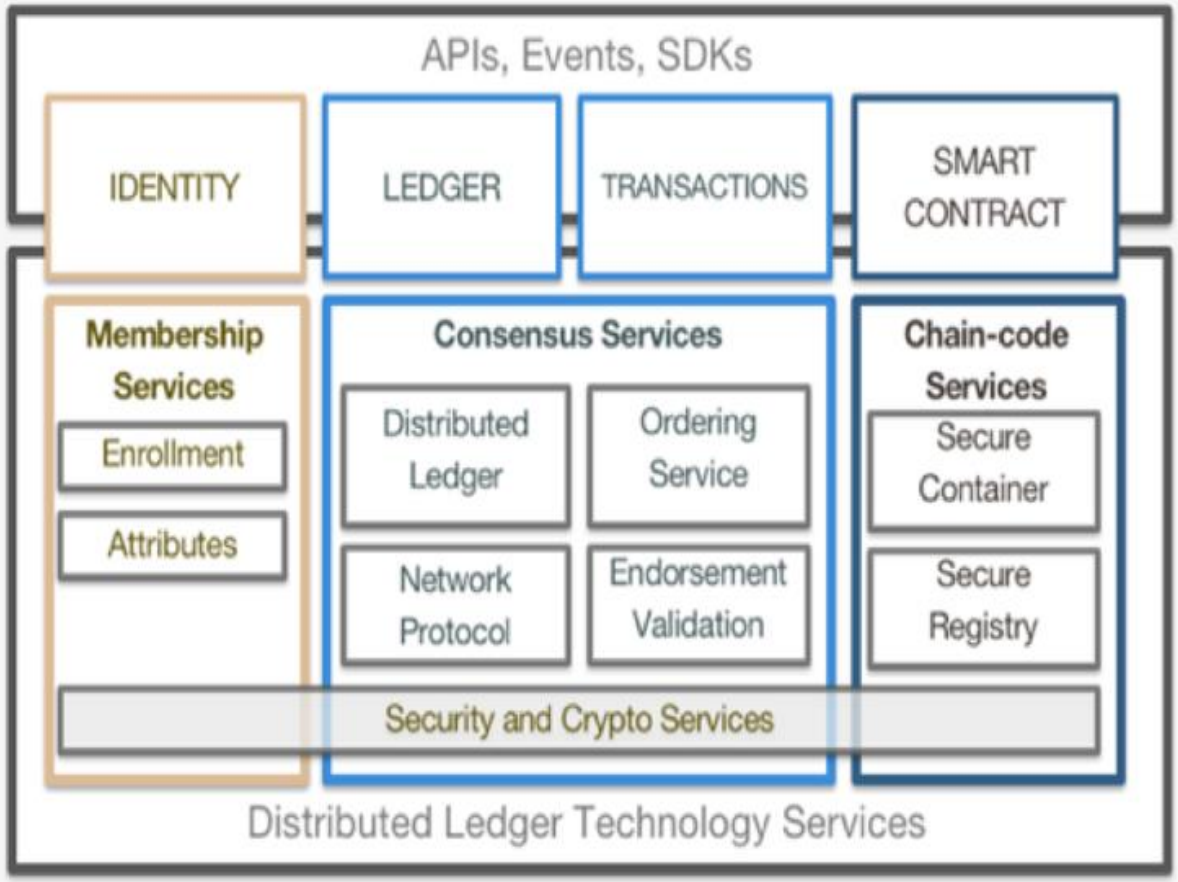
Hyperledger Fabric

- Blockchain implementation designed to support a modular deployment and an scalable arquitectures for enterprise blockchains.
- The modular architecture allows interaction between different implementations and over time.

Fabric modularity:

- Permissioned
- Various consensus mechanisms
- Smart Contracts (chaincode)
- Communication (channels)
- API’s
- Its not design to require a cryptocurrency or token.
- Identity services (membership service) and security.
- Flexible registries (different levels of complexity for queries)
- Interoperability.

Reference Arquitecture



IDENTITY
Pluggable, Membership, Privacy and Auditability of transactions.

LEDGER | TRANSACTIONS
Distributed transactional ledger whose state is updated by consensus of stakeholders

SMART CONTRACT
"Programmable Ledger", provide ability to run business logic against the blockchain (aka smart contract)

APIs, Events, SDKs
Multi-language native SDKs allow developers to write DLT apps

Network architecture: the nodes

- Nodes use a peer-to-peer communication system to maintain the registry updated.
- The network is composed of specialized nodes.
- Nodes must have a valid certificate (permissions) to interact with the network.
- The certificate of each of the nodes is used to sign the transactions that are processed.

Type of nodes: client

- Client: starts a transaction. Must connect to a peer node in order to interacts with the network.
- They can connect with any peer node.
- Initiates and invokes the transactions (chain code).

Types of nodes: Peer

Committing Peer

- Maintains ledger and state
- Commits transactions
- May hold smart contract (chaincode)

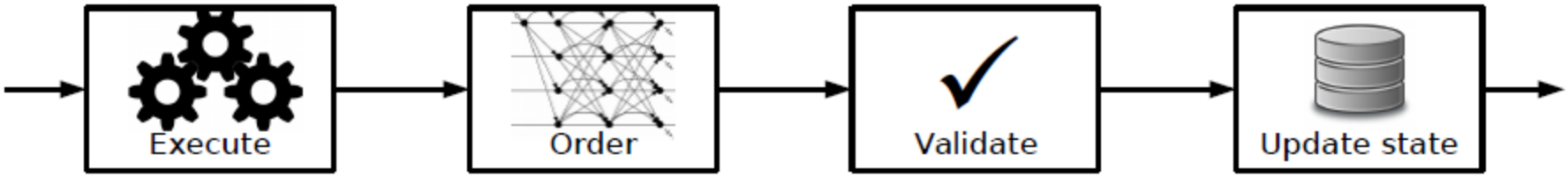
Endorsing Peer

- Receives a transaction proposal for endorsement, responds granting or denying endorsement
- Must hold smart contract
- Verifies that its content obeys a given smart contract
- Endorser “signs” the contract

Ordering Node

- Approves the inclusion of transaction blocks into the ledger and communicates with committing and endorsing peer nodes
- Controls what goes in the ledger making sure that the ledger is consistent
- Does not hold smart contract
- Does not hold ledger

Types of nodes: Peer



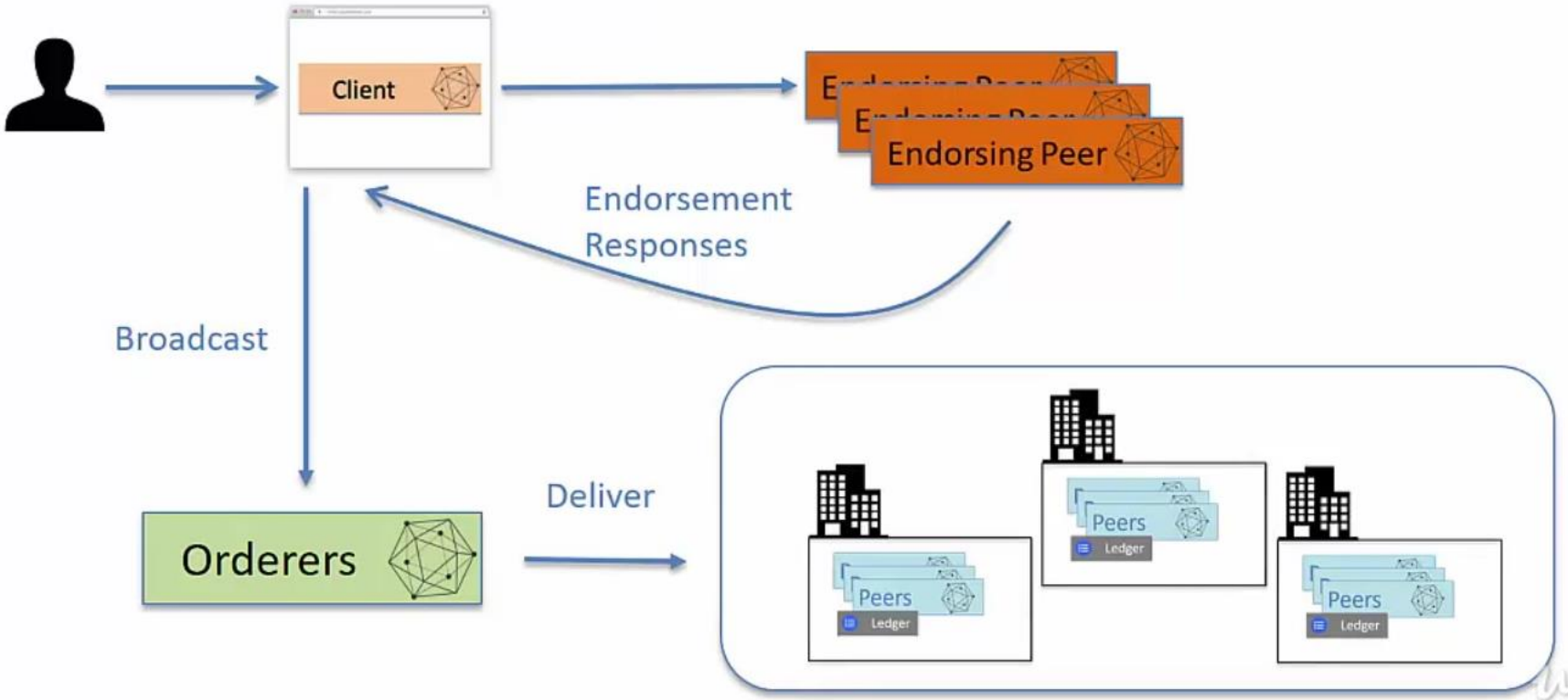
- Simulate trans. and endorse
- Create rw-set
- Collect endorse-ments

- Order rw-sets
- Atomic broadcast (consensus)
- Stateless ordering service

- Validate endorse-ments & rw-sets
- Eliminate invalid and conflicting trans.

- Persist state on all peers

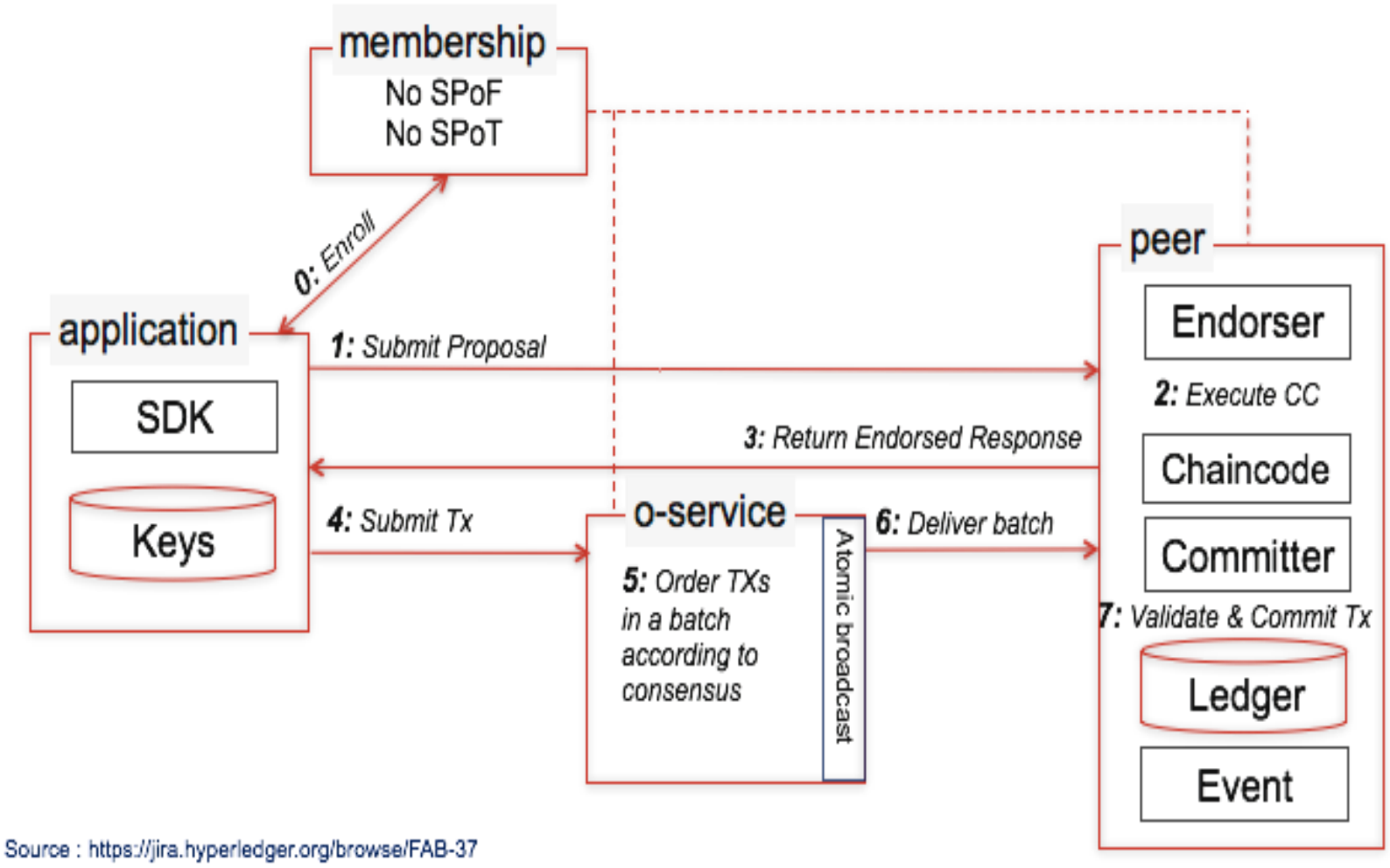
Interaction across the nodes



Transactions

- Fabric allow the definition of policies with respect to transaction execution (chain code)
- Endorsement policies define which peers must agree on the result of the transaction before it is send to the registry. For example peers A,B and C must approve transaction X.
- Endorsement policies are written in code as part of the configuration.
- 3500 tps, 100 peers, Adroulaki et al. (2018)

Transaction flow



Source : https://jira.hyperledger.org/browse/FAB-37

Transactions per second, VISA: 24,000

	Block Generation Time	Transactions Per Second (tps)[23]
Bitcoin	10 minutes	Average 3 tps (Max: 7 tps)
Corda	n/a	> 500 tps
Ethereum	10-19 seconds	Average 15-20 tps, but no theoretical limit
Fabric	variable	> 10 tps
Multichain	Configurable (≥ 2 seconds)	Configurable
Neo	15 seconds	10,000 tps
NXT	1 minute	12 tps
Quorum	50 mSec	>500 tps
Sawtooth	Configurable	>500 tps

Fabric: registry

The ledger is a sequence of assertions(resistant to manipulation) that track the changes in the state of the elements of interest (**assets**).

The changes over the state are generated by the transactions that are invoked by the **participants**.

Fabric: The Ledger

1. State data (CRUD operations): represent the current state of the asset. The state changes as a function of the transactions that affect the assets.
2. Transaction log (immutable): The registry of all the transactions that modify the State data.

Create, Read, Update and Delete

Database properties

	Transaction Logs	State Date (World)
Type	Immutable	Mutable
Operations	Create, Retrieve	ALL-CRUD
DC	levelDB	levelDB/CouchDB
Attitude	Embedded in peers	Key-Value Paired(JSON, Binary)
Query	Simple	Couch DB for Complex

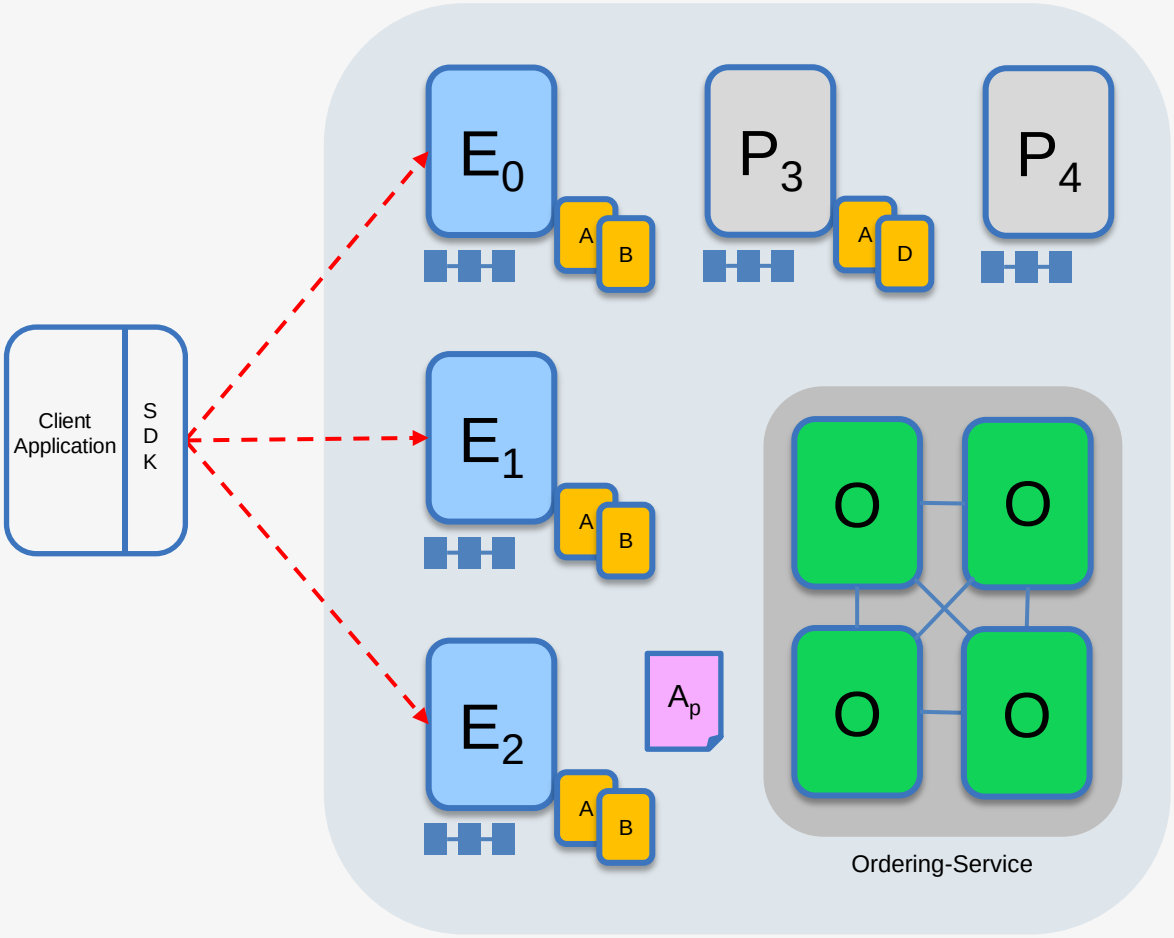
Fabric-UHLevel overview

- Install pre-requisites and download the Fabric Docker images.
- Architecture design through configuration files (*.yml).
  - Configure artifacts: nodes and their roles.
  - Cryptographic components
- Initiate the network, create the communication channels, create administration cards (at different levels: network, nodes,..), deploy the Business Network Application.

Appendix

(Hyperledger, 2017)

Sample transaction: Step 1/7 – Propose transaction



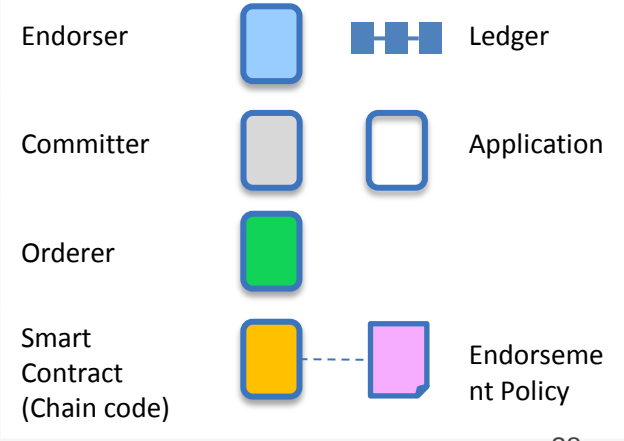
Hyperledger Fabric

Application proposes transaction

Endorsement policy:
- "**E_0, E_1** and **E_2** must sign"
- (**P_3, P_4** are not part of the policy)

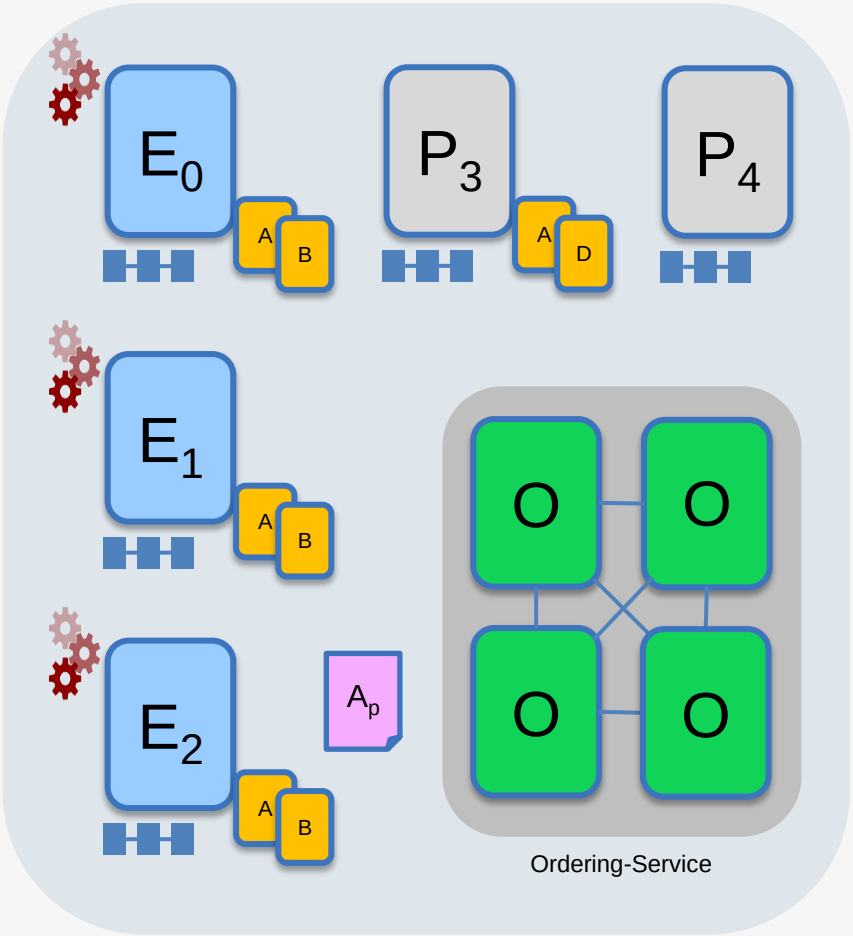
Client application submits a transaction proposal for **chaincode A.** It must target the required peers {**E_0, E_1, E_2**}

Key:



HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Sample transaction: Step 2/7 – Execute proposal



Client Application

SDK

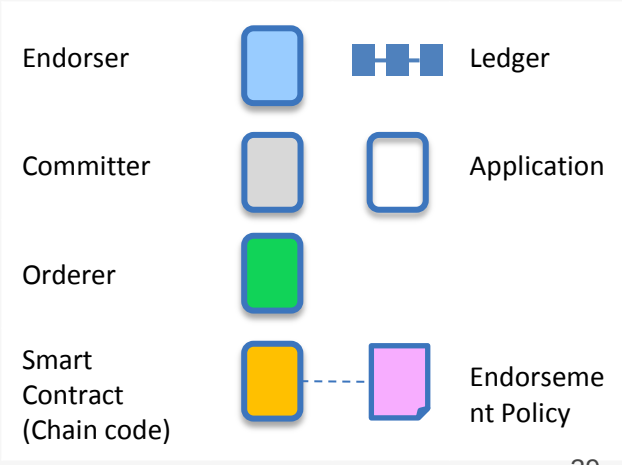
Hyperledger Fabric

Endorsers Execute Proposals

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

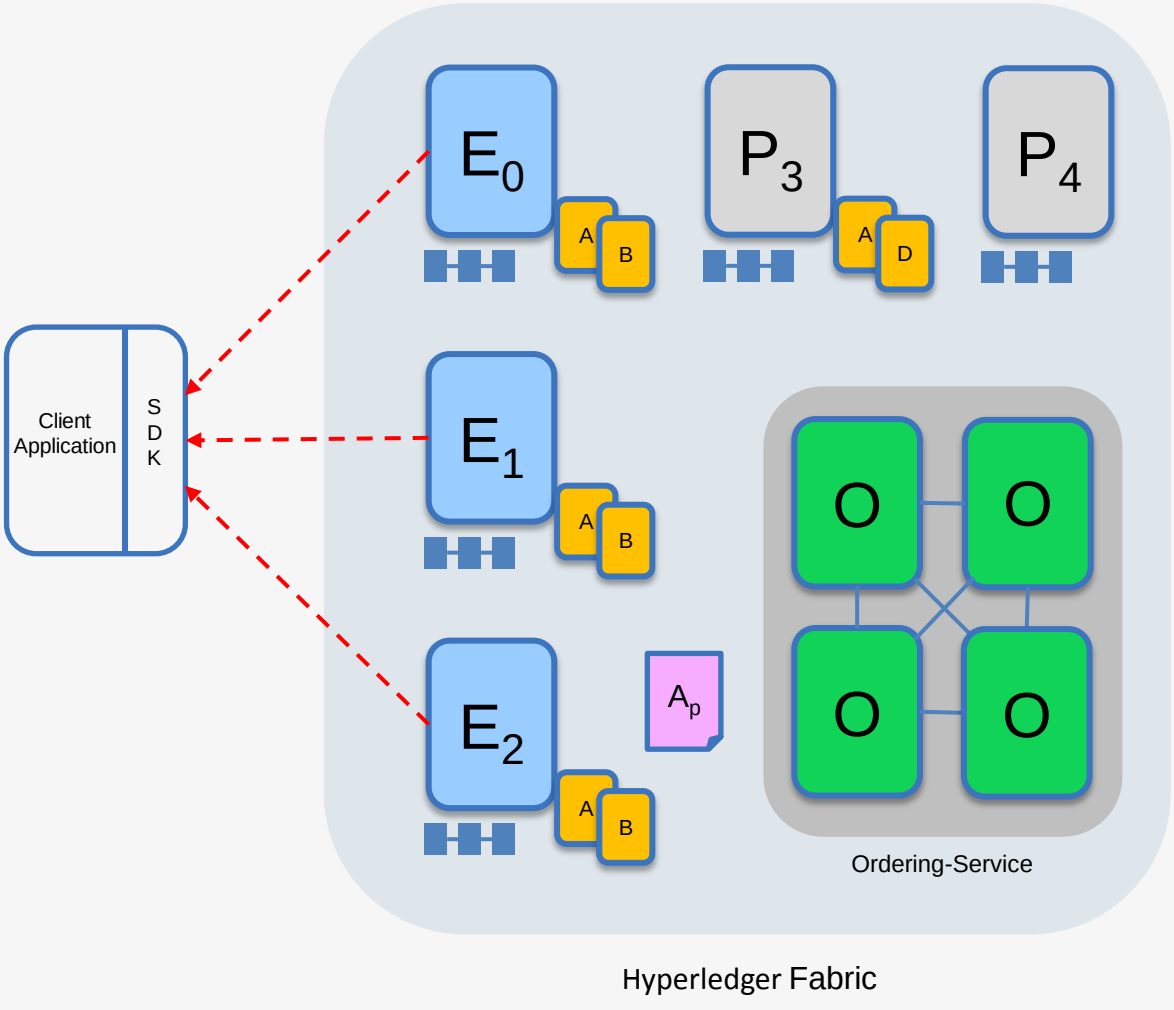
Each execution will capture the set of **R**ead and **W**ritten data, called **RW sets,** which will now flow in the fabric.

Key:



HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Sample transaction: Step 3/7 – Proposal Response

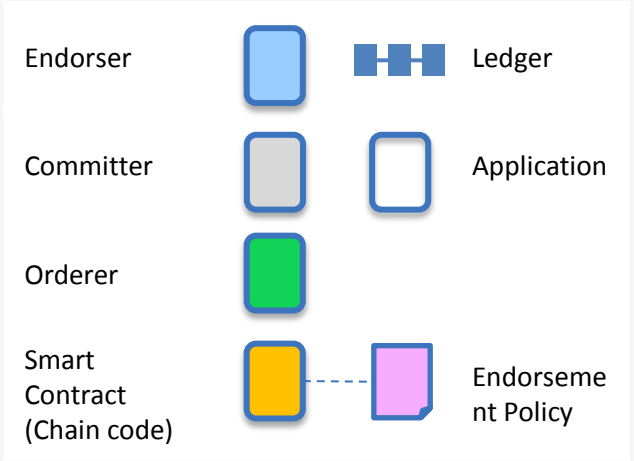


Hyperledger Fabric

Application receives responses

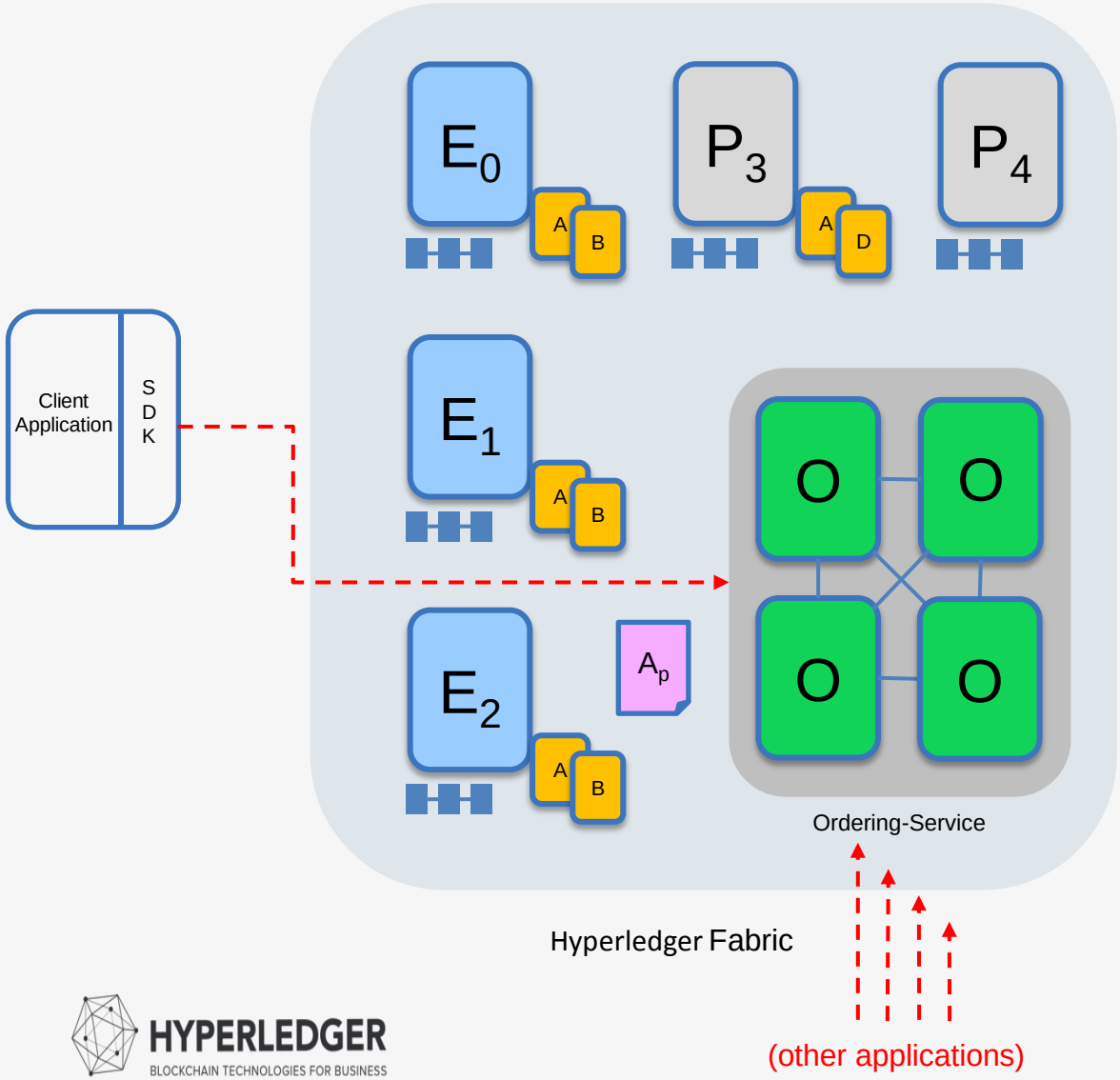
The RW sets are signed by each endorser and returned to the application

Key:



HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Sample transaction: Step 4/7 – Order Transaction

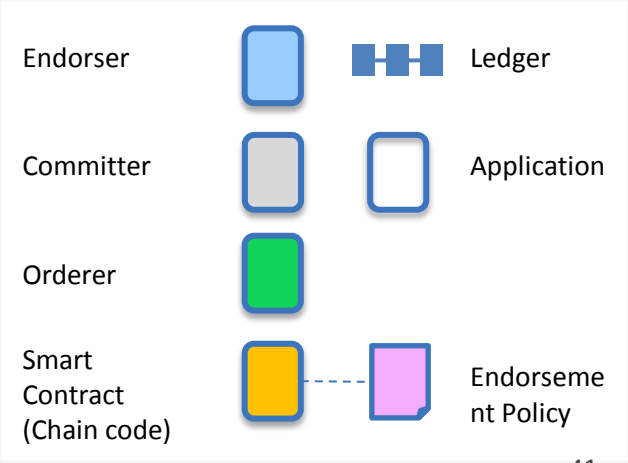


Application submits responses for ordering

Application submits responses as a **transaction** to be ordered.

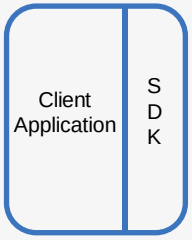
Ordering happens across the fabric in parallel with transactions submitted by other applications

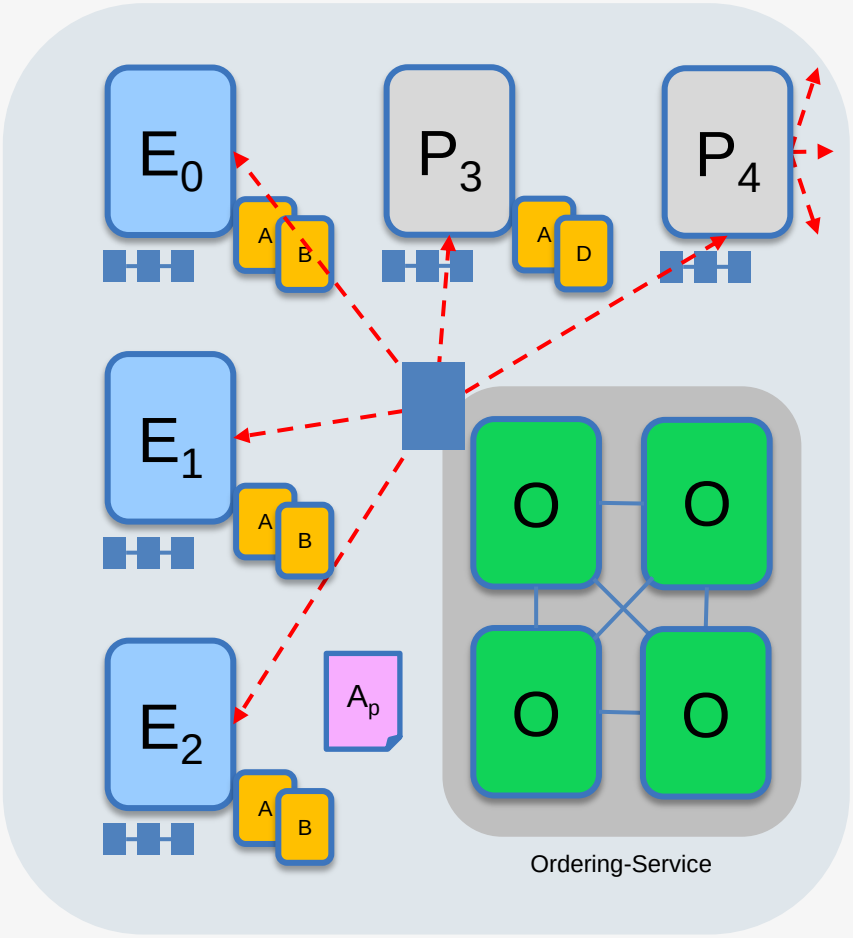
Key:



HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Sample transaction: Step 5/7 – Deliver Transaction





Hyperledger Fabric

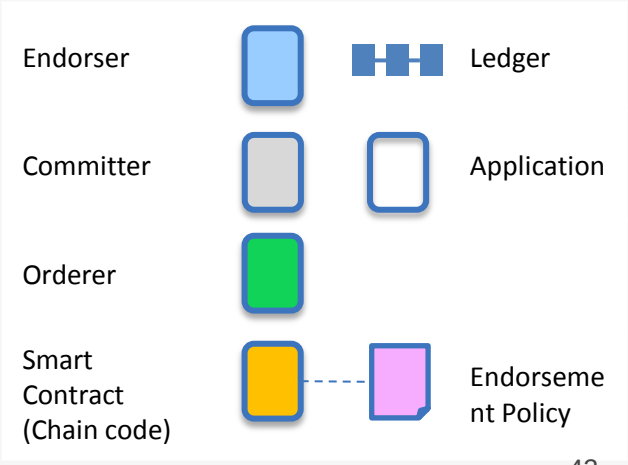
Orderer delivers to all committing peers

Ordering service collects transactions into blocks for distribution to committing peers. Peers can deliver to other peers using gossip (not shown)

Different ordering algorithms available:

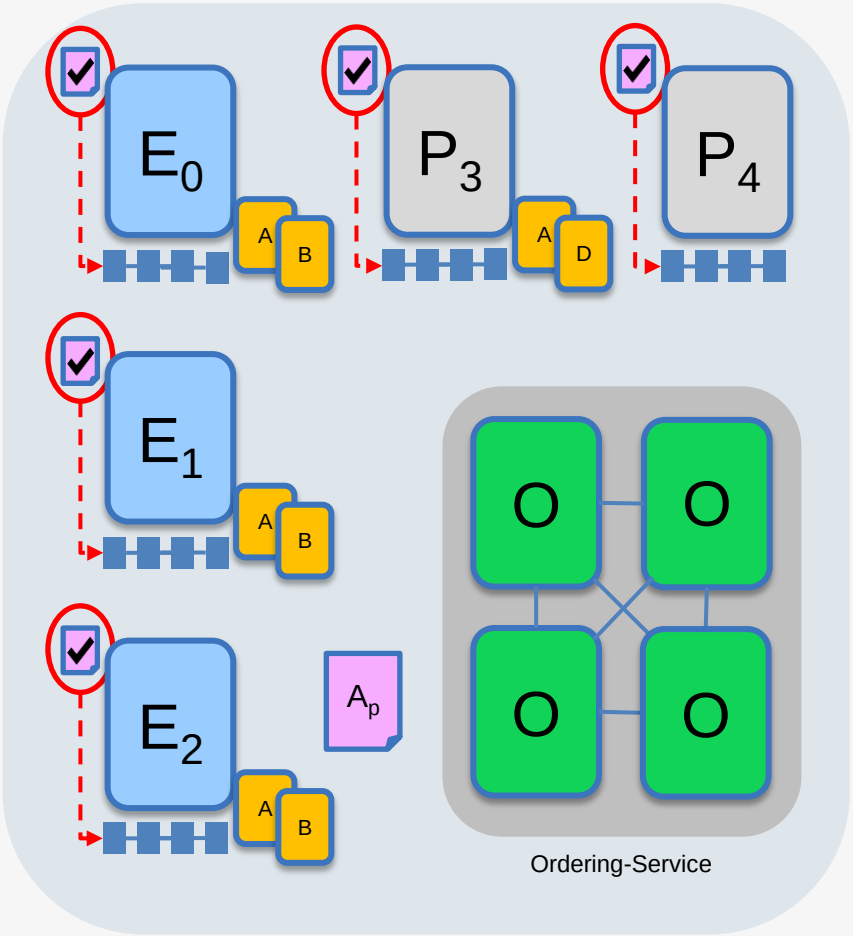
- SOLO (single node, development)
- Kafka (blocks map to topics)
- SBFT (tolerates faulty peers, future)

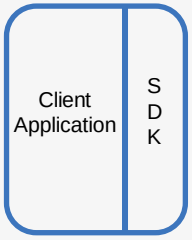
Key:



HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Sample transaction: Step 6/7 – Validate Transaction





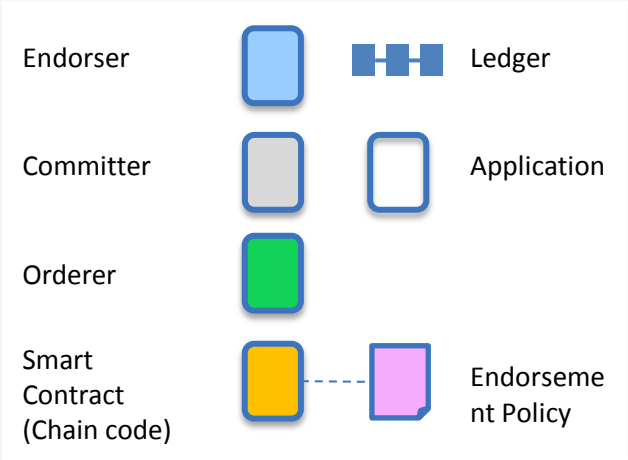
Hyperledger Fabric

Committing peers validate transactions

Every committing peer validates against the endorsement policy. Also check RW sets are still valid for the current state

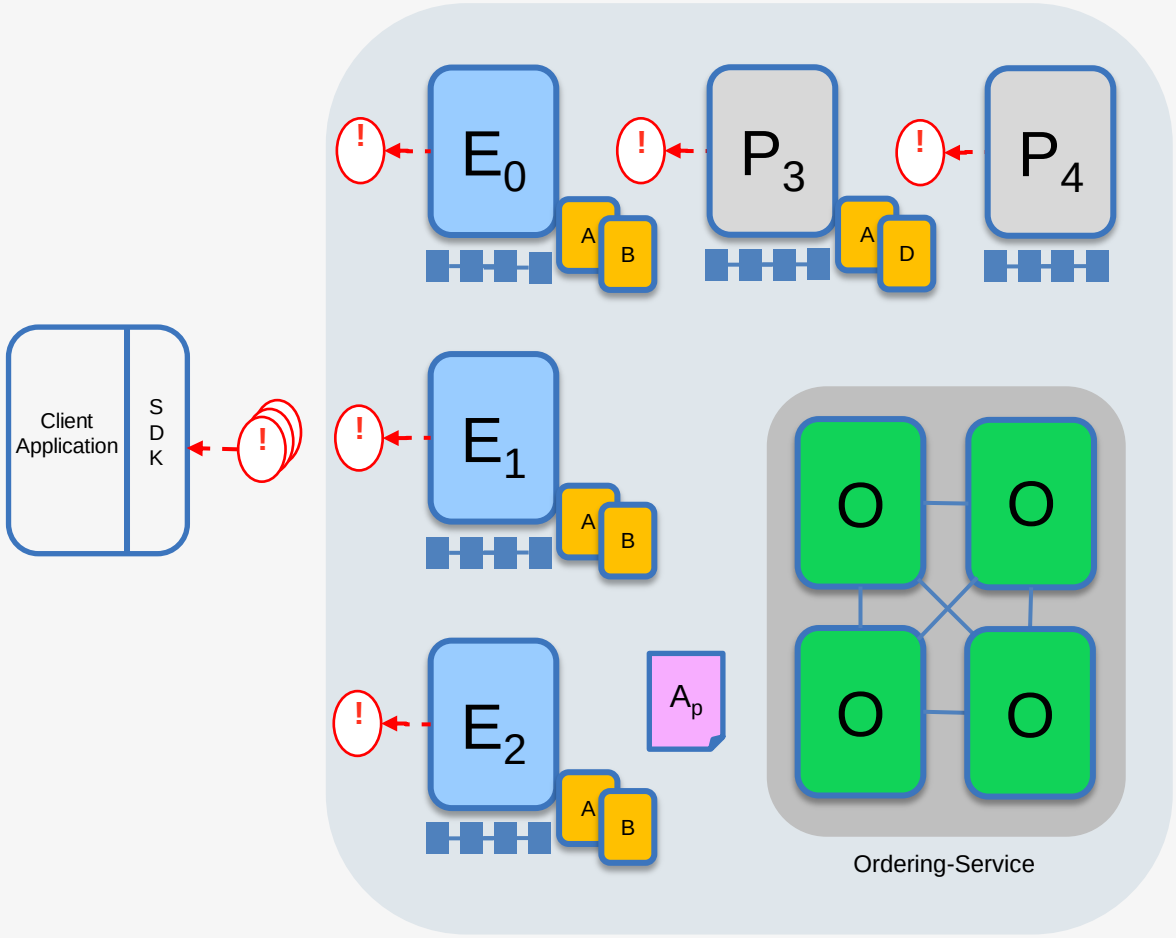
Transactions are written to the ledger and update caching DBs with validated transactions

Key:



HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Sample transaction: Step 7/7 – Notify Transaction



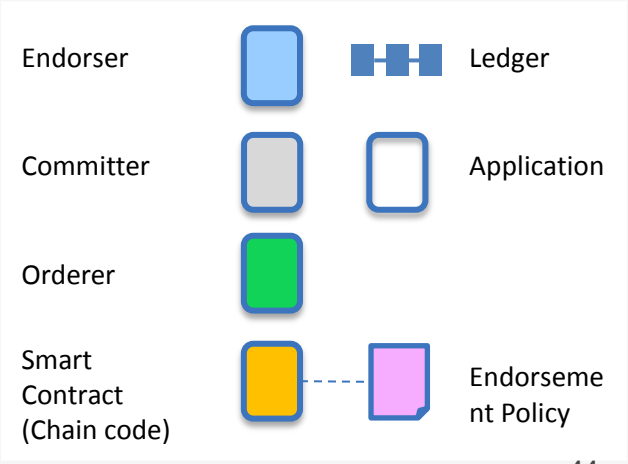
Hyperledger Fabric

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
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS