Blockchain For Enterprise

Mani Madhukar
@manimadhukar

V4.05, 13 July 2017

Blockchain Explained

What is Blockchain?

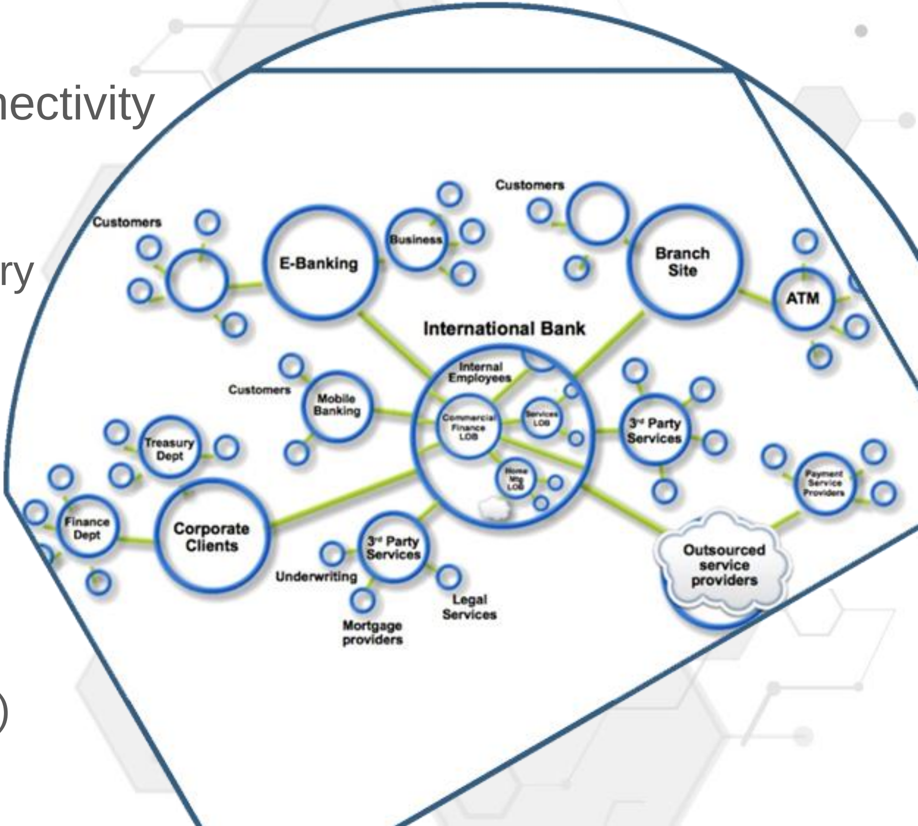
Why is it relevant for our business?

How can IBM help us apply Blockchain?

What

Business networks, wealth & markets

- **Business Networks** benefit from connectivity
  - Participants are customers, suppliers, banks, partners
  - Cross geography & regulatory boundary
- **Wealth** is generated by the flow of goods & services across business network in transactions and contracts
- **Markets** are central to this process:
  - Public (fruit market, car auction), or
  - Private (supply chain financing, bonds)



Transferring assets, building value

Anything that is capable of being owned or controlled to produce value, is an asset

Two fundamental types of asset

- Tangible, e.g. a house
- Intangible, e.g. a mortgage

Intangible assets subdivide

- Financial, e.g. bond
- Intellectual, e.g. patents
- Digital, e.g. music

Cash is also an asset

- Has property of anonymity

What

Ledgers are key …

Ledger is THE system of record for a business.

Business will have multiple ledgers for multiple business networks in which they participate.

– **Transaction** – an asset transfer onto or off the ledger

- John gives a car to Anthony (simple)

– **Contract** – conditions for transaction to occur

- If Anthony pays John money, then car passes from John to Anthony (simple)
- If car won't start, funds do not pass to John (as decided by third party arbitrator) (more complex)

Introducing Blockchain

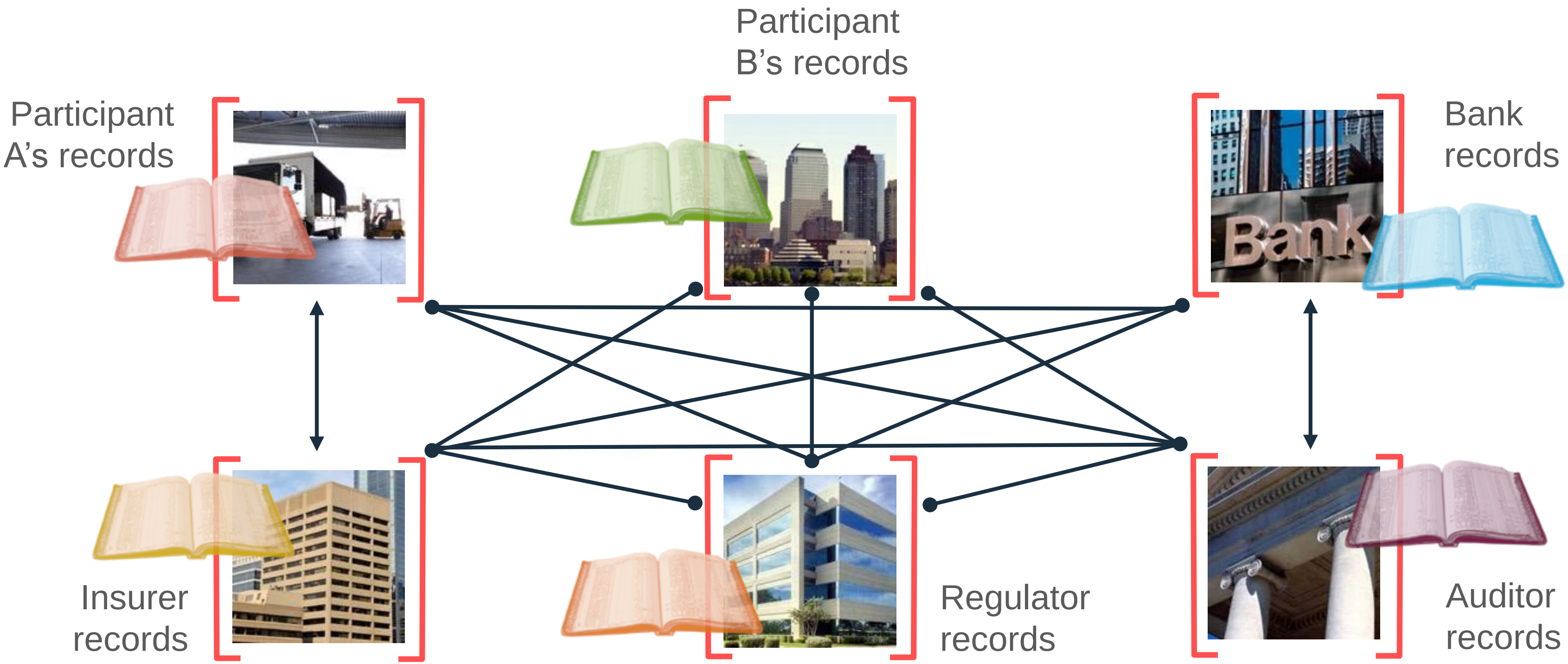
What

A trusted, distributed ledger

with shared business processes

What

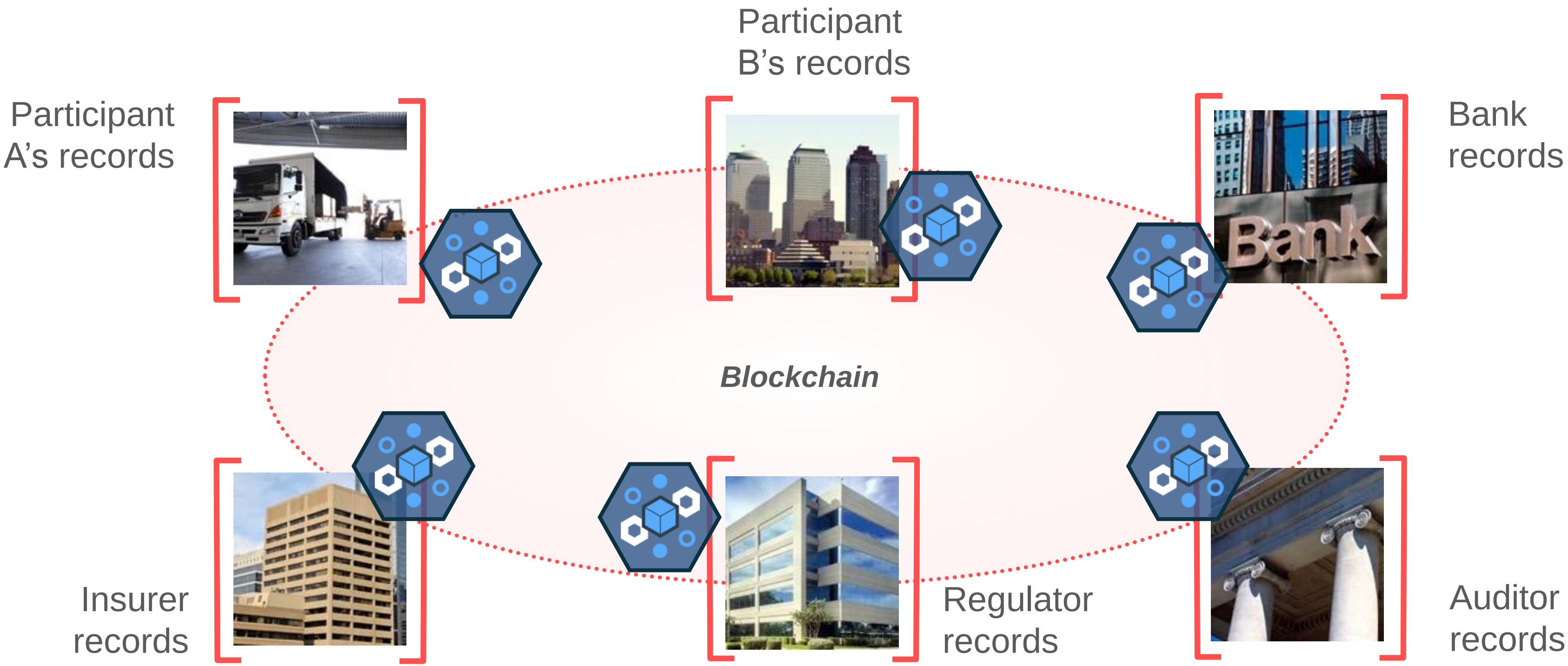
Problem …



… inefficient, expensive, vulnerable

A shared replicated, permissioned ledger…

What



Participant A’s records

Participant B’s records

Bank records

Blockchain

Insurer records

Regulator records

Auditor records

… with consensus, provenance, immutability and finality

Requirements of blockchain for business

Append-only distributed system of record shared across business network

Shared ledger

Smart contract

Business terms embedded in transaction database & executed with transactions

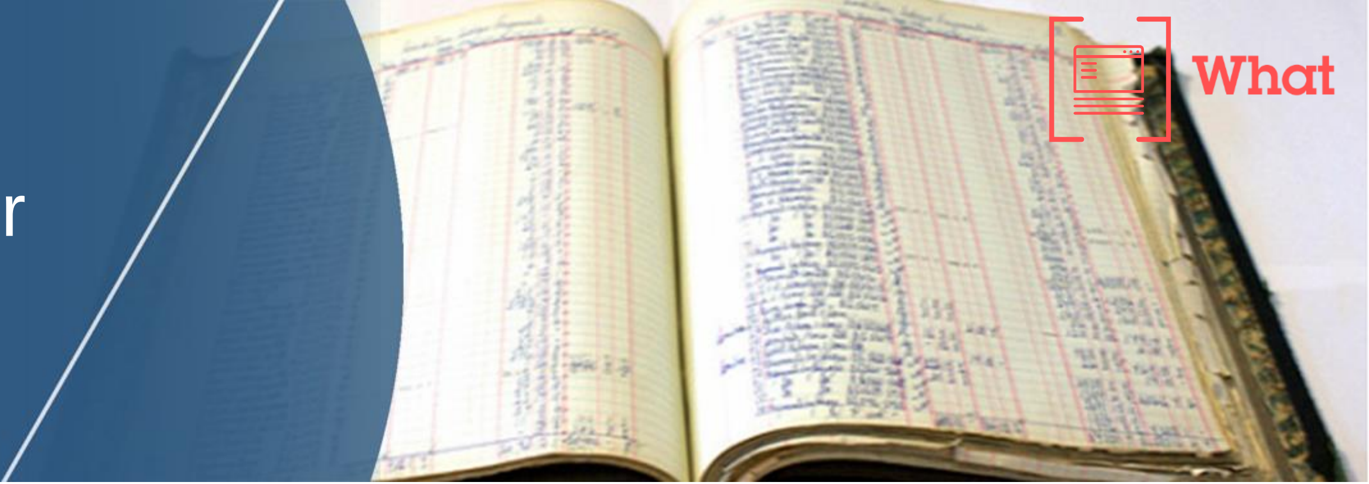
Ensuring appropriate visibility; transactions are secure, authenticated & verifiable

Privacy

Trust

Transactions are endorsed by relevant participants

Shared ledger



Records all transactions across business network

- Shared between participants
- Participants have own copy through replication
- Permissioned, so participants see only appropriate transactions
- THE shared system of record

Smart contract

Business rules implied by the contract … embedded in the Blockchain and executed with the transaction

- Verifiable, signed
- Encoded in programming language
- Example:
  - Defines contractual conditions under which corporate Bond transfer occurs

Privacy

What

The ledger is shared, but participants require privacy

- Participants need:
  - Appropriate confidentiality between subsets of participants
  - Identity not linked to a transaction
- Transactions need to be authenticated
- Cryptography central to these processes

Trust

The ledger is a trusted source of information

- Participants **endorse** transactions
  - Business network decides who will endorse transactions
  - Endorsed transactions are added to the ledger with appropriate confidentiality
- Assets have a verifiable audit trail
  - Transactions cannot be modified, inserted or deleted
- Achieved through consensus, provenance, immutability and finality

Why

Blockchain benefits

Saves time

Transaction time from days to near instantaneous

Removes cost

Overheads and cost intermediaries

Reduces risk

Tampering, fraud & cyber crime

Increases trust

Through shared processes and recordkeeping

Few examples by (selected) industry

Financial	Public Sector	Retail	Insurance	Manufacturing
Trade Finance	Asset Registration	Supply chain	Claims processing	Supply chain
Cross currency payments	Citizen Identity	Loyalty programs	Risk provenance	Product parts
Mortgages	Medical records	Information sharing (supplier – retailer)	Asset usage history	Maintenance tracking
Audit & Compliance	Medicine supply chain		Claims file	Pharma Industry
Letter of Credit				

IndiaHack Fintech Themes

THEMES

Lending

Financial Inclusion

Financial Advisory

Personal Finance

Blockchain

Financial Security

Blockchain Solutions

Hyperledger – Linux foundation – Open source platform

Hyperledger Composer- simplifying Blockchain

IBM Blockchain Platform

How IBM can help

Technology

HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Hyperledger Fabric

Hyperledger Composer

Hosting and Support

High Security Business Network

IBM Bluemix

docker

Making blockchain real for clients

DESIGN THINKING
EXTREME PROGRAMMING XP
LEARN
LEAN START-UP MVP
BLUEMIX SERVICES
BLUEMIX DEVOPS
BLUEMIX PAAS

Garages

Engagement

Hyperledger: A Linux Foundation Project

- A collaborative effort created to advance cross-industry blockchain technologies for business
- Announced December 2015, now over 140 members
- Open source, open standards, open governance
- Five frameworks and three tools projects
- IBM is a premier member of Hyperledger

Brian Behlendorf
Executive Director

Blythe Masters
Board Chair

Chris Ferris
TSC Chair

www.hyperledger.org

Hyperledger Members

Premier

- accenture High performance. Delivered.
- AIRBUS
- AMERICAN EXPRESS
- CHANGE HEALTHCARE
- CME Group
- DEUTSCHE BÖRSE GROUP
- DAIMLER
- Digital Asset
- DTCC Securing Today. Shaping Tomorrow.
- FUJITSU
- HITACHI Inspire the Next
- IBM
- intel
- J.P.Morgan
- NEC
- r3.
- SAP
- 万达·飞凡科技 WANDA FFAN TECHNOLOGY

General

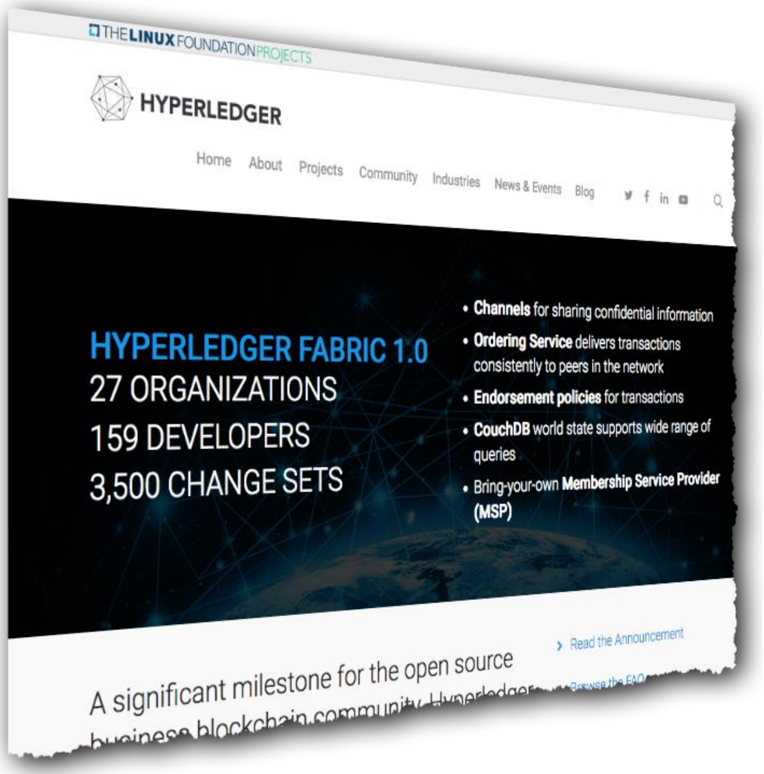
- 1World Blockchain
- ABN·AMRO
- AESTHETIC INTEGRATION
- AlphaPoint
- ALTOROS
- ANZ
- BBVA
- 博图纵横 BOTUZONGHENG
- belink
- BITMARK
- bitSE
- BLOCKCHAIN
- blocko
- Blockstream
- bloq
- BNP PARIBAS
- BNY MELLON
- Broadridge
- bubi.cn 布比
- ca technologies
- Calastone
- 链链 Chain Connected
- JBI JIANGSU HUAXIN BLOCKCHAIN RESEARCH INSTITUTE
- 招商银行 CHINA MERCHANTS BANK
- cisco
- 中信 CITIC
- clause
- cloudsoft
- CLS Fundamental to FX
- coinplug
- collector
- colu.
- ConsensusBase
- CONSENSYS
- Cuscal The complete payments partner
- Deloitte.
- ENERGY BLOCKCHAIN LABS
- EY
- Eurostep Digital
- FACTOM
- 房掌柜 FZG360.COM
- Gem
- 众享比特 PeerSafe
- guardtime
- 33.CN 复杂美
- HASHED HEALTH
- HUAWEI
- HUNDSUN
- 趣链科技 Hyperchain
- intellect
- intuit
- INVeSHARE
- IROOTECH
- KAISER PERMANENTE.
- KRX KOREA EXCHANGE
- KSD Korea Securities Depository
- koscom Financial IT Partner
- .kubique
- LedgerDomain
- Libra
- loyyal
- Lykke
- MadHive
- Milligan Partners
- MIRACL
- MONAX INDUSTRIES
- MonetaGo
- ML
- MOSCOW EXCHANGE
- MURPHY & MCGONIGLE A Professional Corporation
- NSE
- NEC Orchestrating a brighter world
- NETKI
- NEX
- NOKIA
- norbloc
- NTT DaTa Global IT Innovator
- onchain
- 橙色魔方 Orange Magic Cube
- PAXOS
- PDX
- RadarWin 雷盈
- redhat
- SAMSUNG SAMSUNG SDS
- SANY
- SBERBANK
- Schroders
- SECURE KEY
- GINGKOO 金丘股份
- onechain
- 四方精创 FORMS SYNTRON
- NEXGO
- 点融网 Dianrong.com
- SORAMITSU
- STATE STREET
- SWIFT
- swisscom
- symbiont
- 太一 TAI
- tequa creek
- THOMSON REUTERS
- TMX
- UMF 联动优势
- vmware
- WELLS FARGO
- 云象 YUNPHANT
- 梧桐树 wutongtree.com
- 保全网 BaoQuan.com

Associate

- BANK OF ENGLAND
- CHAMBER OF DIGITAL COMMERCE
- CSA cloud security alliance
- ColoredCoins
- FEDERAL RESERVE BANK OF BOSTON
- IC3 Initiative for Cryptocurrencies & Contracts
- Investrata Foundation
- NXT FOUNDATION
- OSCRE
- sovrin
- BLOCKCHAIN RESEARCH INSTITUTE
- TNO innovation for life
- The Illinois BLOCKCHAIN Initiative
- Fondazione INUIT
- vsp Global

Source: https://www.hyperledger.org/about/members
Updated June 2017

Hyperledger Fabric: Distributed Ledger Platform



HYPERLEDGER FABRIC 1.0
27 ORGANIZATIONS
159 DEVELOPERS
3,500 CHANGE SETS

- **Channels** for sharing confidential information
- **Ordering Service** delivers transactions consistently to peers in the network
- **Endorsement policies** for transactions
- **CouchDB** world state supports wide range of queries
- Bring-your-own **Membership Service Provider (MSP)**

› Read the Announcement

A significant milestone for the open source business blockchain community. Hyperledger

Browse the FAQ

- An implementation of blockchain technology that is a foundation for developing blockchain applications
- Emphasis on ledger, smart contracts, consensus, confidentiality, resiliency and scalability.
- V1.0 released July 2017
  - 159 developers from 27 organizations
  - IBM is one contributor of code, IP and development effort to Hyperledger Fabric

http://hyperledger-fabric.readthedocs.io/

Why Hyperledger Fabric?

Open Governance
Anyone can join or contribute

Built from the ground up for enterprise
With a maturity model to help companies move to production

Performance
Supports up to 1000 tps*

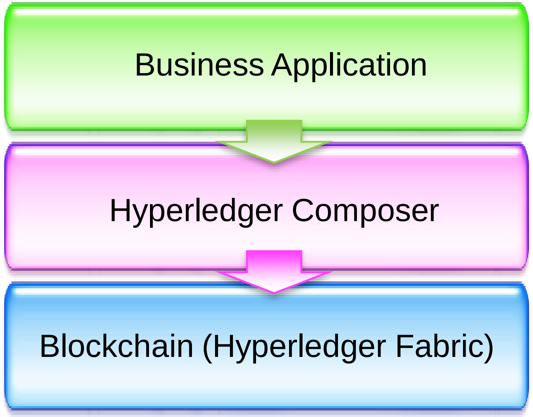
Confidentiality and privacy
Built-in channels for isolation and membership services for signing and encryption. Supports IBM High Security Business Network.

Modularity and flexibility
Choice of consensus algorithms and programming languages

Hyperledger Composer: Accelerating time to value

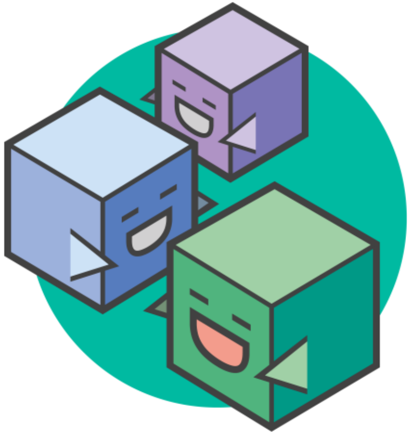
https://hyperledger.github.io/composer/

- A suite of high level application abstractions for business networks
- Emphasis on business-centric vocabulary for quick solution creation
- Reduce risk, and increase understanding and flexibility


Business Application

Hyperledger Composer

Blockchain (Hyperledger Fabric)



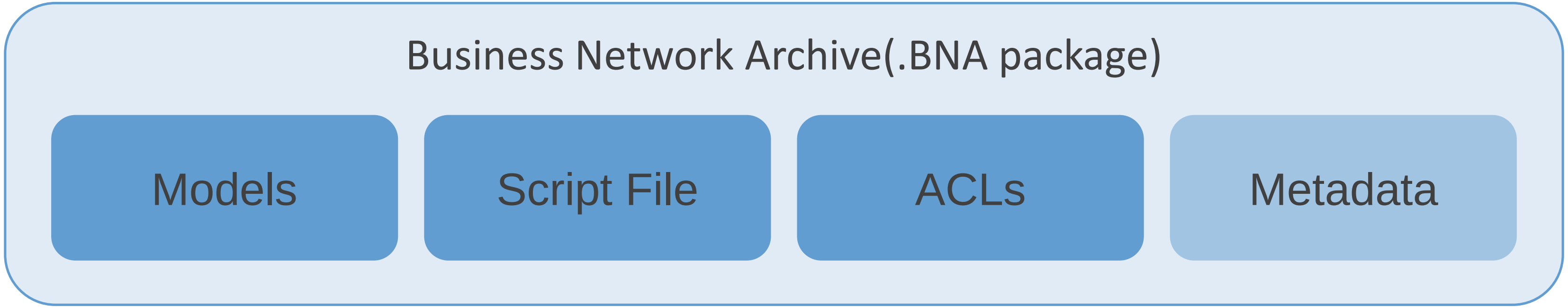
- Features
  - Model your business networks, test and expose via APIs
  - Applications invoke APIs transactions to interact with business network
  - Integrate existing systems of record using loopback/REST
- **Fully open** and part of Linux Foundation Hyperledger
- Try it in your web browser now: http://composer-playground.mybluemix.net/

Conceptual Components and Structure

Business Network is defined by **Models, Script Files, ACLs** and **Metadata** and packaged in a **Business Network Archive**

Solution Developer models the business network, implements the script files that define transaction behaviour and packages into a business network archive

Solution Administrator provisions the target environment and manages deploy



Business Network Archive(.BNA package)

Models

Script File

ACLs

Metadata

Benefits of Hyperledger Composer

Increases understanding

Bridges simply from business concepts to blockchain

Saves time

Develop blockchain applications more quickly and cheaply

Reduces risk

Well tested, efficient design conforms to best practice

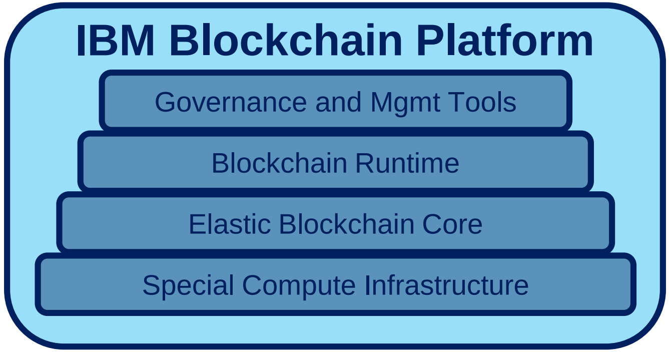
Increases flexibility

Higher level abstraction makes it easier to iterate

Introducing the IBM Blockchain Platform:

A full stack Blockchain Service

A **full stack** Blockchain Platform with Hyperledger Fabric tightly integrated and optimized



IBM Blockchain Platform

Governance and Mgmt Tools

Blockchain Runtime

Elastic Blockchain Core

Special Compute Infrastructure

Most complete platform for Enterprise Blockchains

Self Service Production Ready Networks in minutes

Unmatched Security across the entire stack

Simplified Multi-org Operation with native Governance Tools

Managed Blockchain with dashboards and controls

Why IBM?

Industry Expert

- Hundreds of experienced consultants, researchers and developers
- Deep systems integration and middleware experience

Secure by Design

- IBM Blockchain **High Security Business Network**
- Dedicated compute, cryptography hardware, tamper-resistant container.

Open By Design

- Linux Foundation Hyperledger founding member
- Ongoing donation of code, developers and intellectual property to Hyperledger

Fast Start

- 400+ clients in engagement pipeline in 2016
- IBM **Blockchain Garage** engagement model to implement MVP rapidly

Hyper Scale

- **Choice of deployment** including on-prem, off-prem, self-managed or *aaS
- Supports rapid expansion of initial solution.

Selected References

FX Netting

CLS
Fundamental to FX

Settlements through digital currency

MIZUHO

Identity management

Crédit Mutuel
ARKEA

Food Safety

Walmart

Private Equity

NORTHERN
TRUST

Channel Financing

IBM
IBM Global Financing

Low liquidity securities trading and settlement

JPX
TOKYO STOCK EXCHANGE

Cross Border Supply Chain

IBM | MAERSK

Contract Management

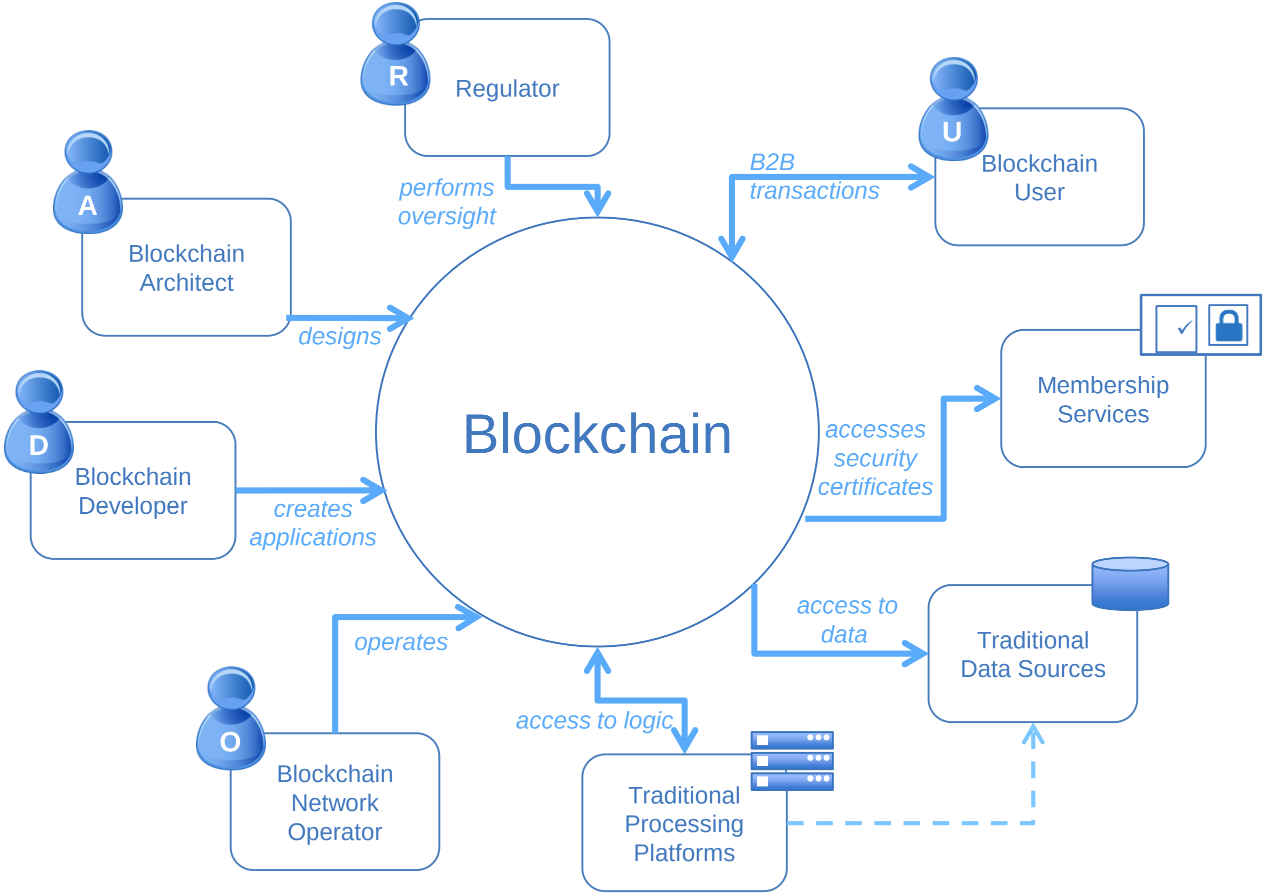
MUFG

Blockchain Architected

The technical concepts and components of a blockchain solution

Considerations for the blockchain developer, operator and architect

Actors in a blockchain solution



Actors in a blockchain solution

Blockchain Architect	A	Responsible for the architecture and design of the blockchain solution
Blockchain User	U	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	R	The overall authority in a business network. Specifically, regulators may require broad access to the ledger’s contents.
Blockchain Developer	D	The developer of applications and smart contracts that interact with the Blockchain and are used by Blockchain users.
Blockchain Operator	O	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 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	f(abc);	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	E T	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.

Key players for Blockchain adoption

Regulator

- An organization who enforces the rules of play
- Regulators are keen to support Blockchain based innovations
- Concern is systemic risk – new technology, distributed data, security

Industry Group

- Often funded by members of a business network
- Provide technical advice on industry trends
- Encourages best practice by making recommendations to members

Market Maker

- In financial markets, takes buy-side and sell-side to provide liquidity
- More generally, the organization who innovates
  - Creates a new good or service, and business process (likely)
  - Creates a new business process for an existing good or service

Considerations for the blockchain developer

The blockchain developer

D

Blockchain Developer

Blockchain developers' primary interests are…

- Application
- Smart Contract

f(abc);

…and how they interact with the ledger and other systems of record:

- Ledger
- Traditional Processing Platforms
- Traditional Data Sources
- Events
- Systems Integration

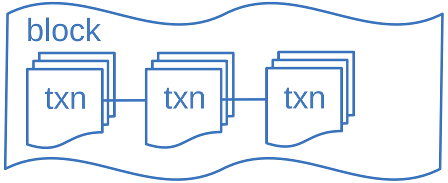
!

They should NOT have to care about operational concerns, such as:

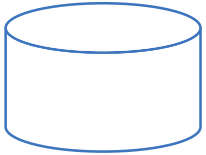
X

- Peers
- Consensus
- Security

A ledger often consists of two data structures



Blockchain

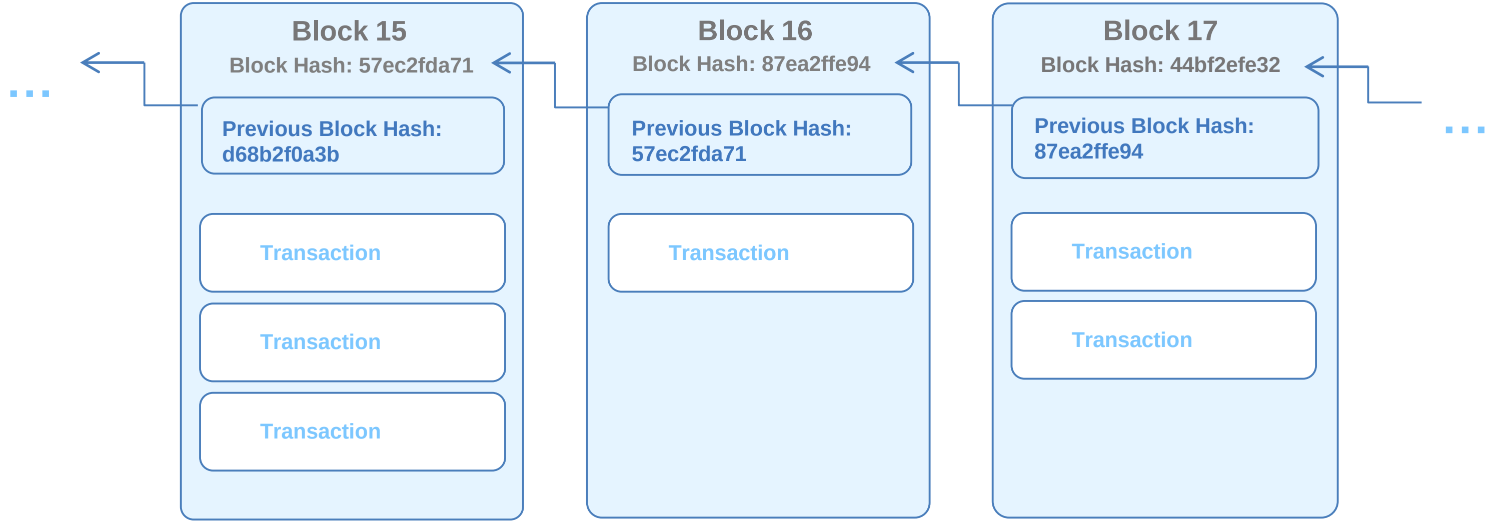


World state

- 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

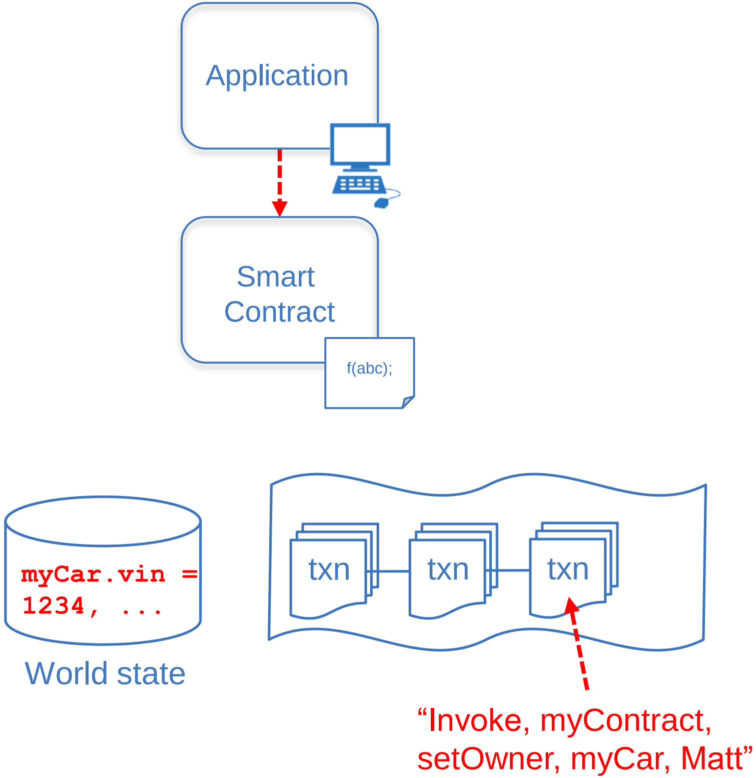
- World State
  - An ordinary database (e.g. key/value store)
  - Stores the combined outputs of all transactions
  - Not usually immutable

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 of a change of ownership transaction



Transaction input - sent from application

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

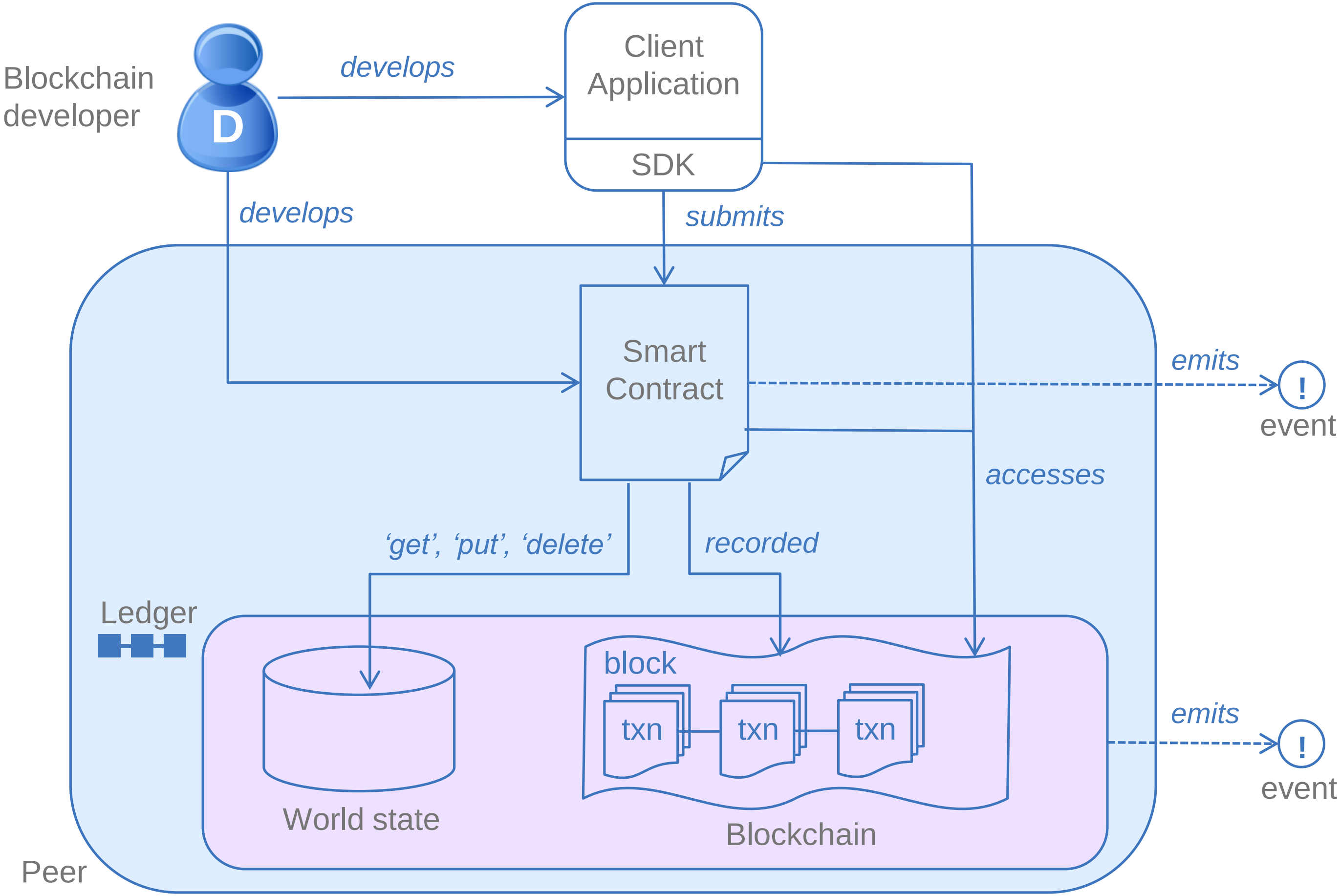
Smart contract implementation

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

World state: new contents

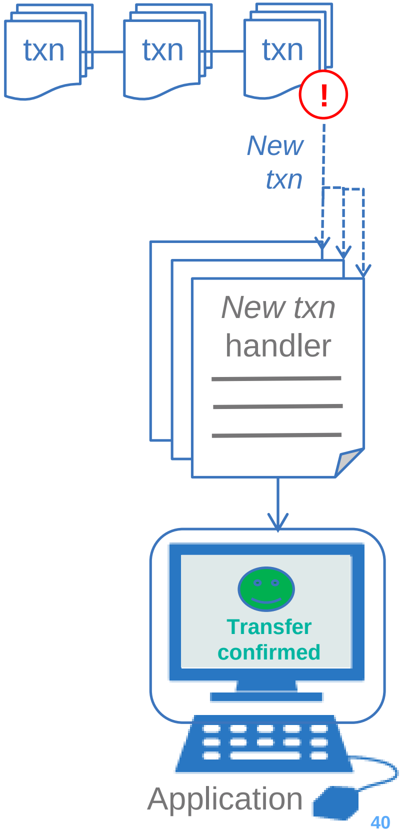
```
myCar.vin = 1234
myCar.owner = Matt
myCar.make = Audi
…
```

How applications interact with the ledger

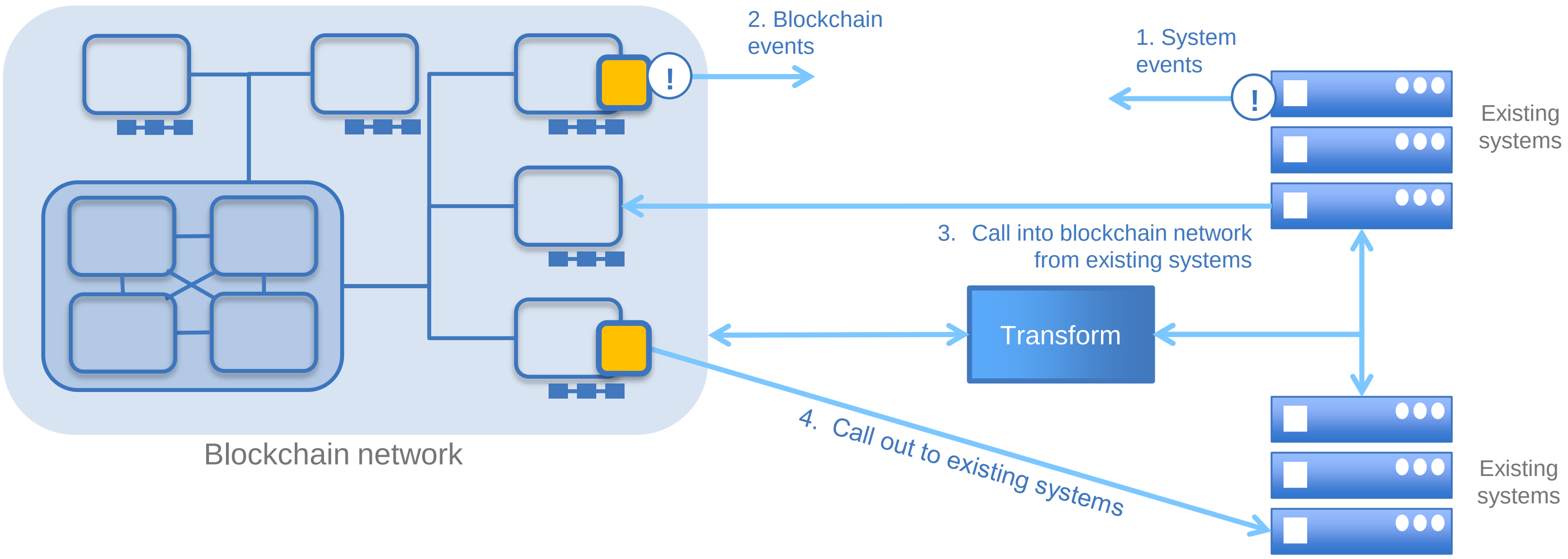


How events are used in blockchain

- In computing, an event is an occurrence that can trigger handlers
  - e.g. disk full, fail transfer completed, mouse clicked, message received, temperature too hot…
- Events are important in asynchronous processing systems like blockchain
- The blockchain can emit events that are useful to application programmers
  - e.g. Transaction has been validated or rejected, block has been added…
- Events from external systems might also trigger blockchain activity
  - e.g. exchange rate has gone below a threshold, the temperature has gone up, a time period has elapsed…



Integrating with existing systems – possibilities



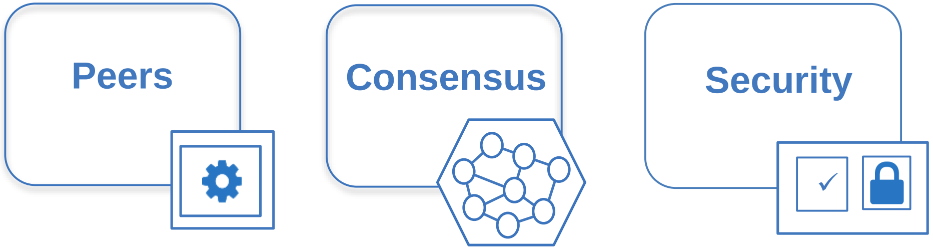
Considerations for the blockchain operator

The blockchain operator

O

Blockchain Operator

Blockchain operators’ primarily interests are in the deployment and operation of part of the blockchain:



Peers

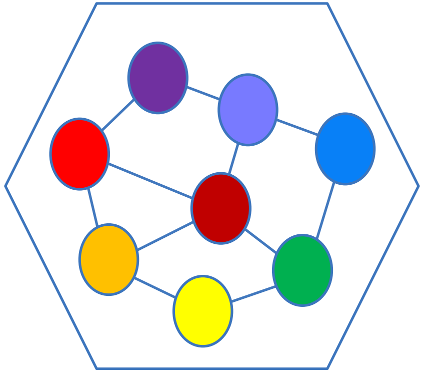
Consensus

Security

They should NOT have to care about development concerns, such as:

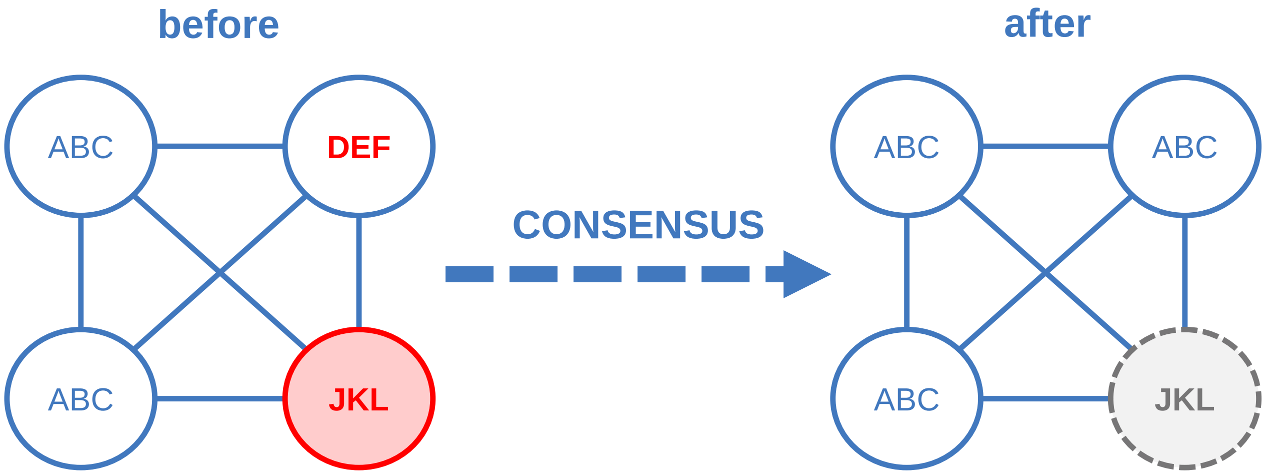
X Application code | Smart contract code

Peers



- Peers are the technical services that a blockchain requires in order to work
  - Peers hold and maintain the ledger
  - They receive transactions from applications (and other peers)
  - Peers can validate transactions
  - They notify applications about the outcome of submitted transactions
- Peers are implemented as an operating system process
  - …to which applications and other peers can connect
  - Very similar to web servers!
- Peers connect to other peers to form nodes on a peer-to-peer blockchain network
  - Peers can be run wherever makes sense; allows for heterogeneous technology choices
  - Some blockchains are worldwide, others are private to a business network
  - However, peers from one blockchain implementation cannot talk with others (yet!)
    - For example, an Ethereum blockchain cannot transfer assets to a Hyperledger Fabric blockchain
  - It might make sense to have one peer per business network participant, but this is not necessarily so

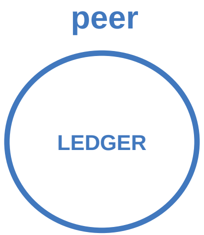
Consensus: The process of maintaining a consistent ledger



Keep all peers up-to-date

Fix any peers in error

Ignoring all malicious nodes



Some examples of consensus algorithms

Proof of work

Proof of stake

Solo / No-ops

Kafka / Zookeeper

Proof of Elapsed Time

PBFT based

Consensus algorithms have different strengths and weaknesses

Proof of work

Require validators to solve difficult cryptographic puzzles

PROs: Works in untrusted networks

CONS: Relies on energy use; slow to confirm transactions

Example usage: Bitcoin, Ethereum

Proof of stake

Require validators to hold currency in escrow

PROs: Works in untrusted networks

CONS: Requires intrinsic (crypto)currency, ”Nothing at stake” problem

Example usage: Nxt

Proof of Elapsed Time

Wait time in a trusted execution environment randomizes block generation

PROs: Efficient

CONS: Currently tailored towards one vendor

Example usage: Sawtooth-Lake

Consensus algorithms have different strengths and weaknesses

Solo /
No-ops

Validators apply received transactions without consensus

PROs: Very quick; suited to development

CONS: No consensus; can lead to divergent chains

Example usage: Hyperledger Fabric V1

PBFT-based

Practical Byzantine Fault Tolerance implementations

PROs: Reasonably efficient and tolerant against malicious peers

CONS: Validators are known and totally connected

Example usage: Hyperledger Fabric V0.6

Kafka /
Zookeeper

Ordering service distributes blocks to peers

PROs: Efficient and fault tolerant

CONS: Does not guard against malicious activity

Example usage: Hyperledger Fabric V1

Security: Public vs. private blockchains

Public blockchains

- For example, Bitcoin
- Transactions are viewable by anyone
- Participant identity is more difficult to control

Private blockchains

- For example, Hyperledger Fabric
- Network members are known but transactions are secret

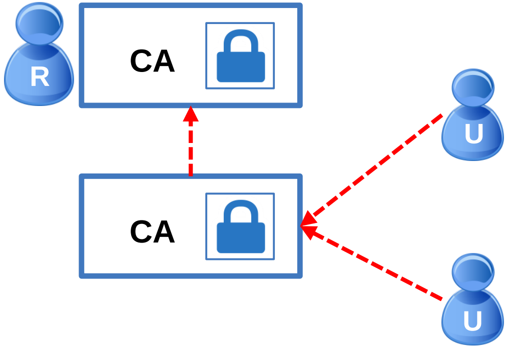
- Some use-cases require anonymity, others require privacy
  - Some may require a mixture of the two, depending on the characteristics of each participant

- **Most business use-cases require private, permissioned blockchains**
  - Network members know who they’re dealing with (required for KYC, AML etc.)
  - Transactions are (usually) confidential between the participants concerned
  - Membership is controlled

Security: Real-world vs. digital identity

- Consider real-world identity documents…
  - The issuers of the identity documents are trusted third parties (e.g. passport office)
  - There is usually a chain of trust (e.g. to get a bank card you need a drivers license or passport)
  - Identity documents are often stored in **wallets**

- In the digital world, identities consist of public/private key pairs known as certificates
  - Identity documents are issued by trusted third parties known as **Certificate Authorities** (CAs)



- Private blockchain networks also require CAs
  - So network members know who they're dealing with
  - May sit with a regulatory body or a trusted subset of participants

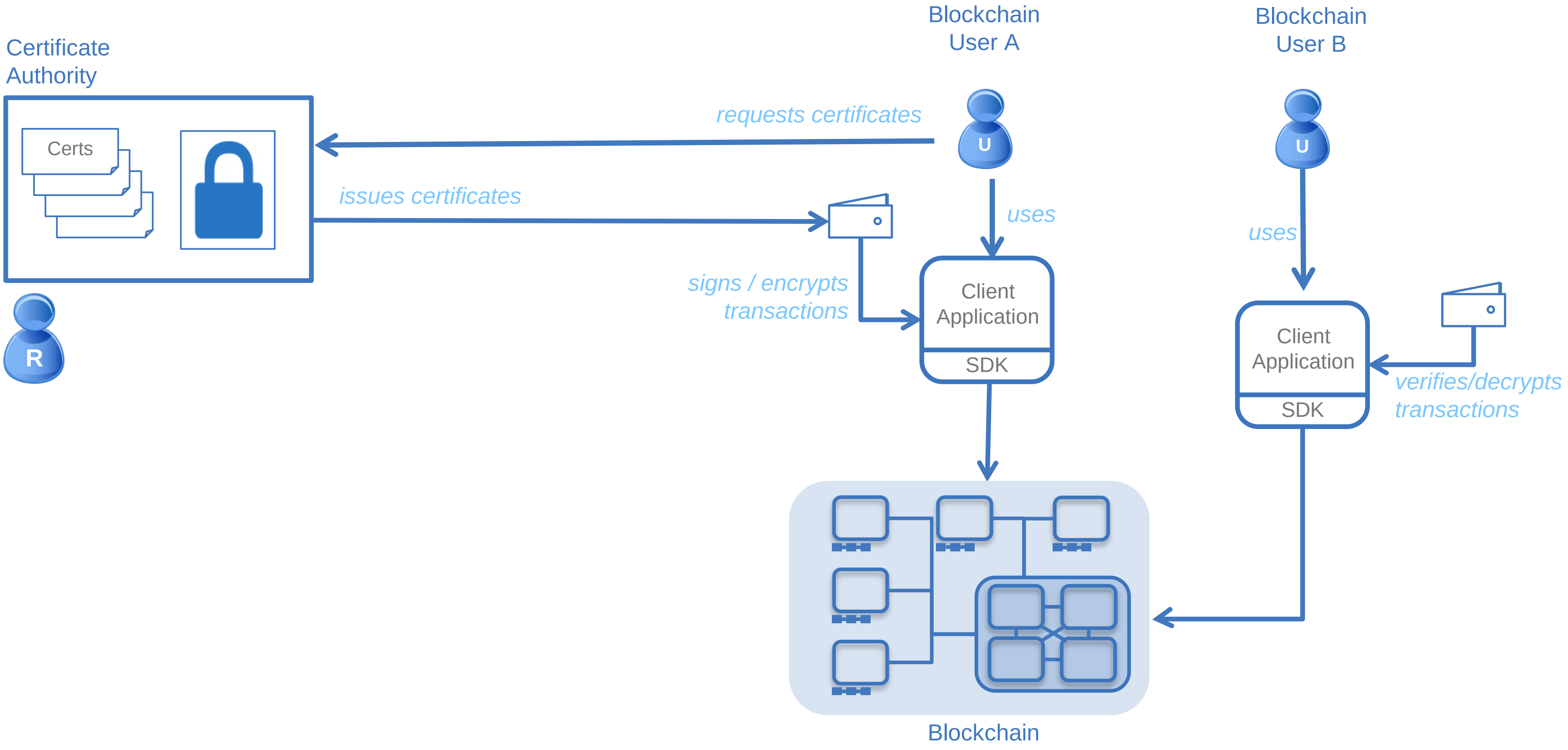
Security: Encryption and Signing

- Cryptography basics
  - Every member of the network has (at least) one public key and one private key
  - Assume that every member of the network knows all public keys and only their own private keys
  - **Encryption** is the process applying a transformation function to data such that it can only be decrypted by the other key in the public/private key pair
  - Users can **sign** data with a private key; others can verify that it was signed by that user

- For example
  - Alice can sign a transaction with her private key such that anyone can verify it came from her
  - Anyone can encrypt a transaction with Bob’s public key; only Bob’s private key can decrypt it

- In private, permissioned blockchains
  - Transactions and smart contracts can be signed to verify where they originated
  - Transactions and their payloads can be encrypted such that only authorized participants can decrypt

Certificate Authorities and Blockchain



Additional considerations for the blockchain architect

The blockchain architect

Blockchain Architect

For a successful solution, blockchain architects need a good understanding of many development and operational concerns discussed in this session:

- Applications
- Smart contracts
- Events and Integration
- Peers
- Consensus
- Security

However there are additional considerations for architects to bear in mind from the outset. For example:

- Business concerns
- Design Tradeoffs

Blockchain Explored

HYPERLEDGER FABRIC

- Project status and roadmap
- Technical Deep Dive

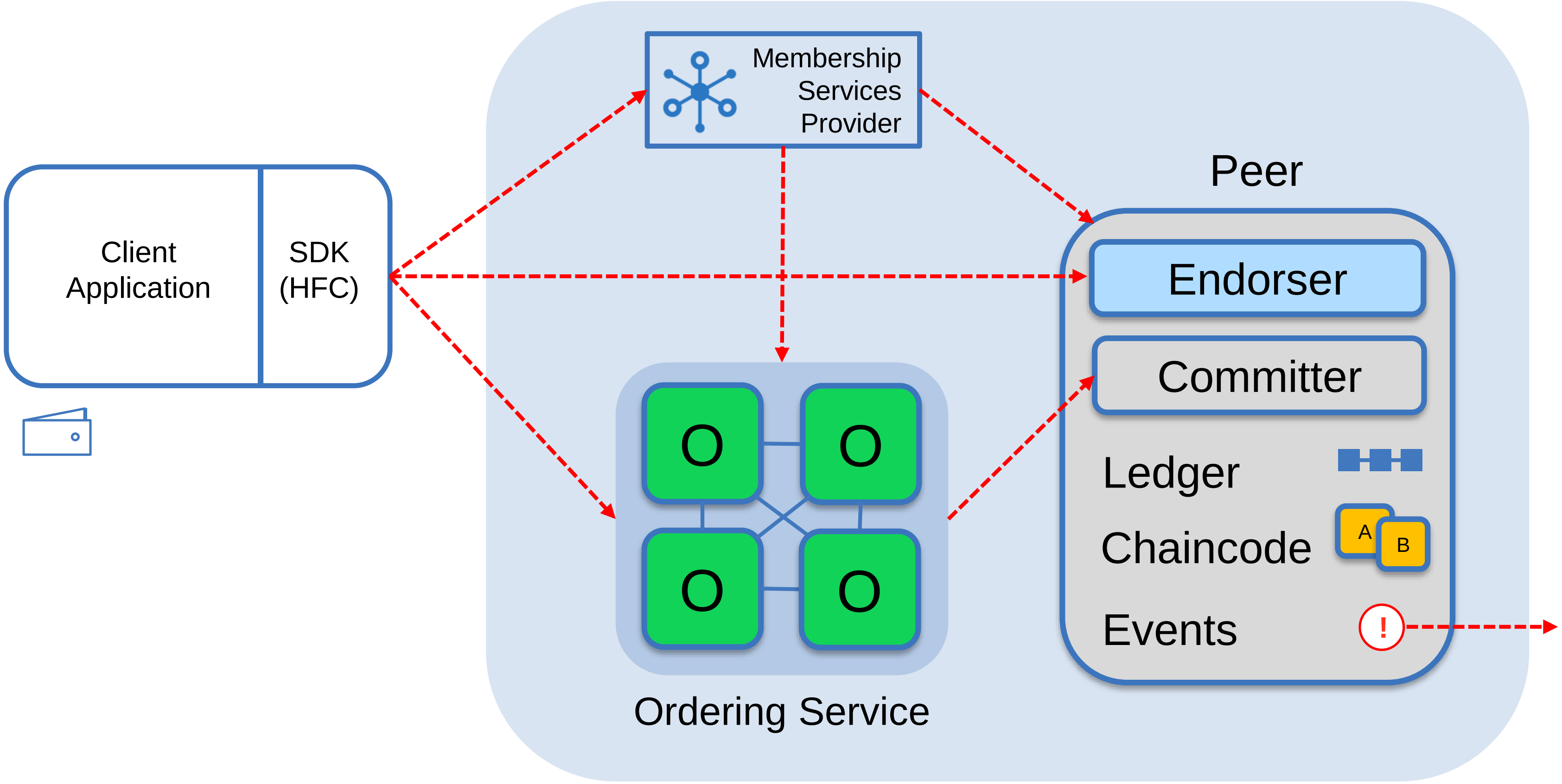
What is Hyperledger Fabric

- Linux Foundation Hyperledger
  - A collaborative effort created to advance cross-industry blockchain technologies for business

- Hyperledger Fabric
  - An implementation of blockchain technology that is intended as a foundation for developing blockchain applications
  - Key technical features:
    - A shared ledger and smart contracts implemented as “chaincode”
    - Privacy and permissioning through membership services
    - Modular architecture and flexible hosting options

- V1.0 released July 2017: contributions by 159 engineers from 27 organizations
  - IBM is one contributor to Hyperledger Fabric

Hyperledger Fabric V1 Architecture



Contents

HYPERLEDGER FABRIC

Project status and roadmap

Technical Deep Dive

Hyperledger Fabric V1 - Deep Dive Topics

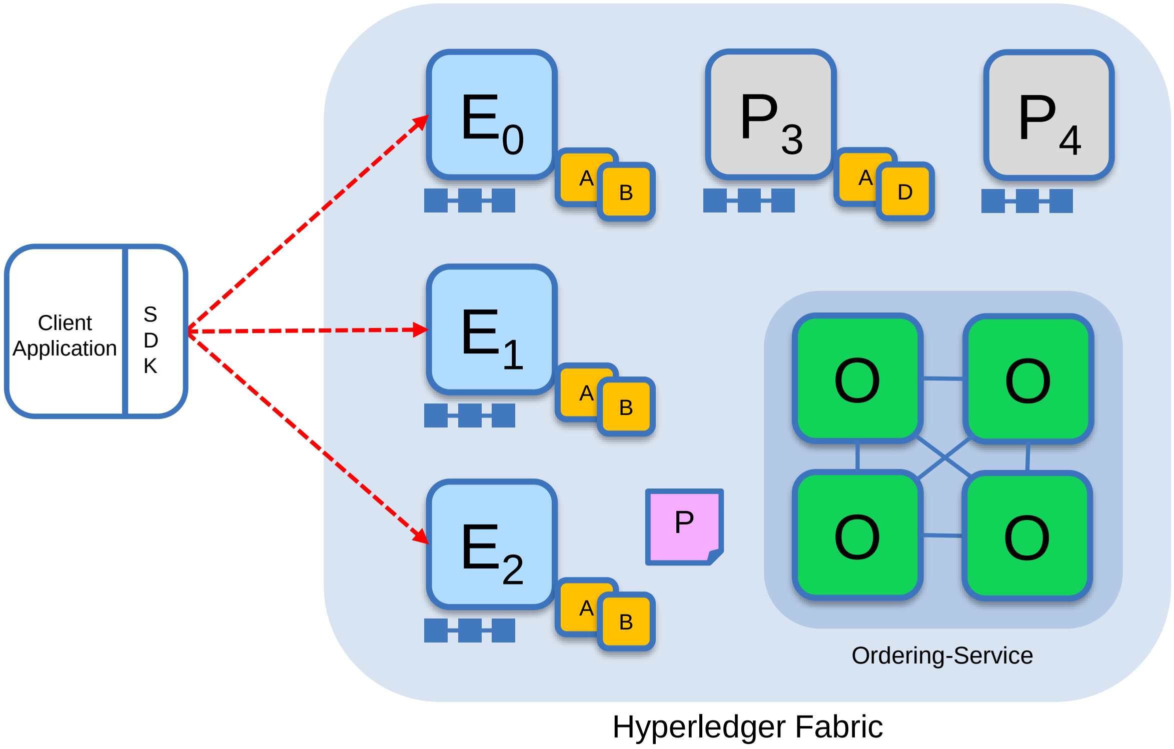
- Network Consensus
- Channels and Ordering Service
- Network setup
- Endorsement Policies
- Permissioned ledger access
- Pluggable world-state

Network Consensus

Nodes and roles

⬜	**Committing Peer**: Maintains ledger and state. Commits transactions. May hold smart contract (chaincode).
🟦	**Endorsing Peer**: Specialized committing peer that receives a transaction proposal for endorsement, responds granting or denying endorsement. Must hold smart contract
🟩	**Ordering Nodes (service)**: 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.

Sample transaction: Step 1/7 – Propose transaction



Hyperledger Fabric

Application proposes transaction

Endorsement policy:

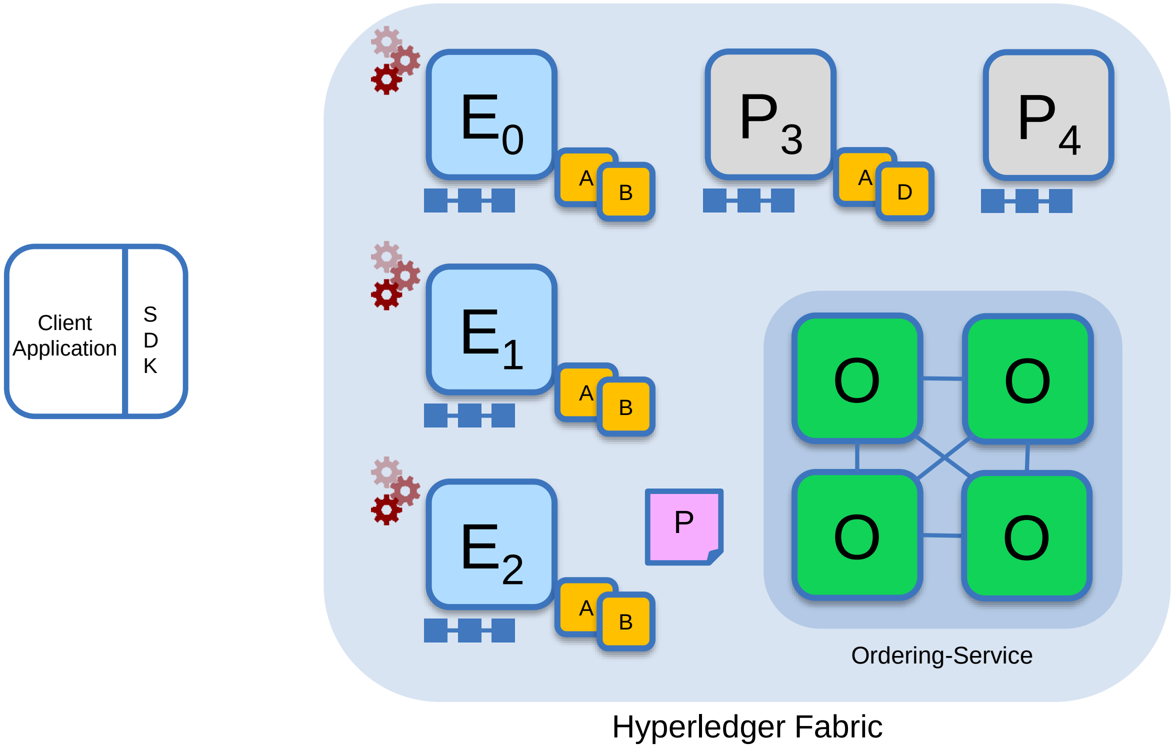
- "$\mathbf{E_0}$, $\mathbf{E_1}$ and $\mathbf{E_2}$ must sign"
- ($\mathbf{P_3}$, $\mathbf{P_4}$ are not part of the policy)

Client application submits a transaction proposal for **Smart Contract A.** It must target the required peers {$\mathbf{E_0}$, $\mathbf{E_1}$, $\mathbf{E_2}$}

Key:

Endorser	🟦	■-■-■	Ledger
Committing Peer	⬜	□	Application
Ordering Node	🟩		
Smart Contract (Chaincode)	🟨	🟪	Endorsement Policy

Sample transaction: Step 2/7 – Execute proposal



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.

Transactions can be signed & encrypted

Key:

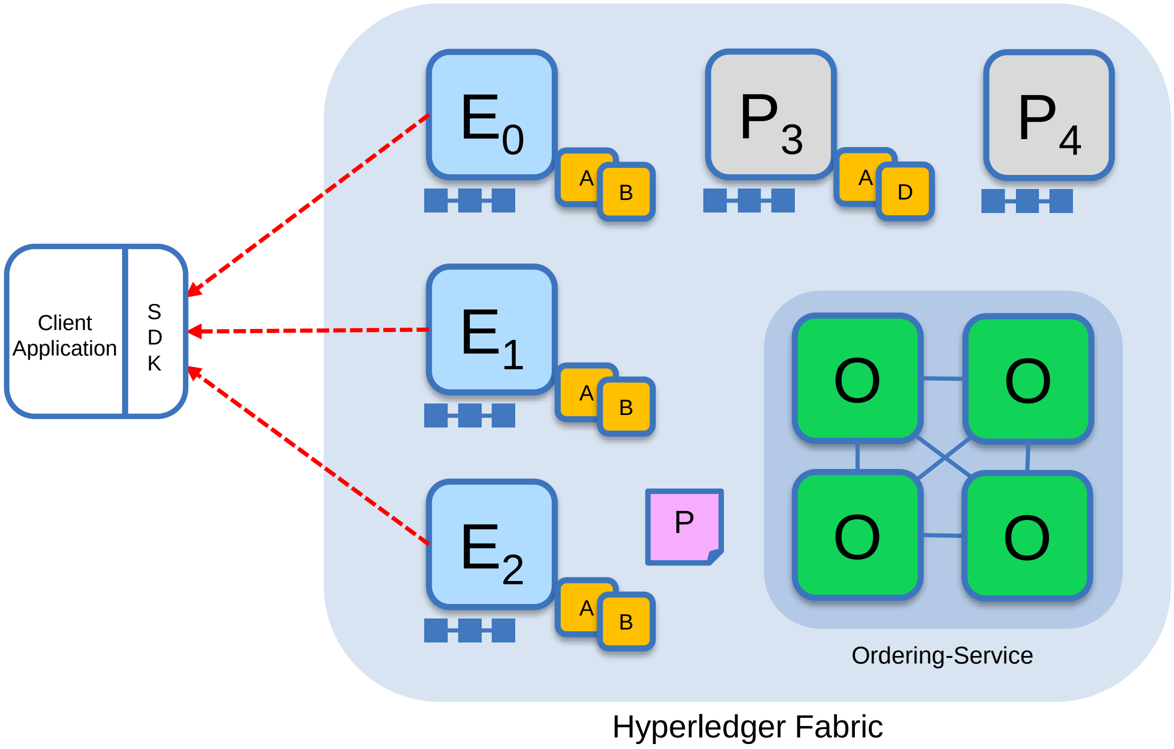
Endorser 🟦 ▬ Ledger

Committing Peer ⬜ □ Application

Ordering Node 🟩

Smart Contract (Chaincode) 🟨 - - - ▨ Endorsement Policy

Sample transaction: Step 3/7 – Proposal Response



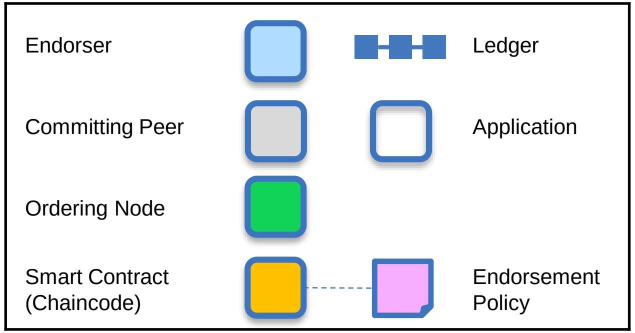
Application receives responses

RW sets are asynchronously returned to application

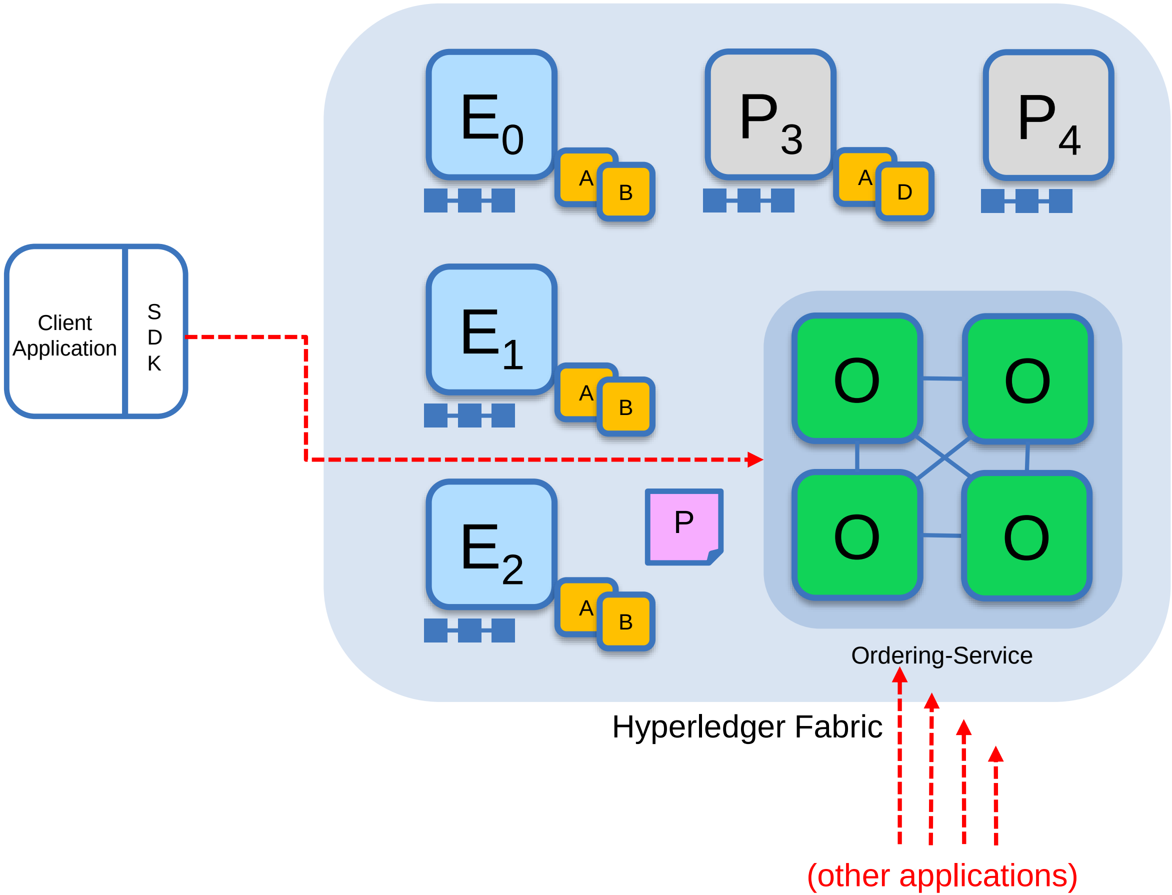
The RW sets are signed by each endorser, and also includes each record version number

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

Key:



Sample transaction: Step 4/7 – Order Transaction

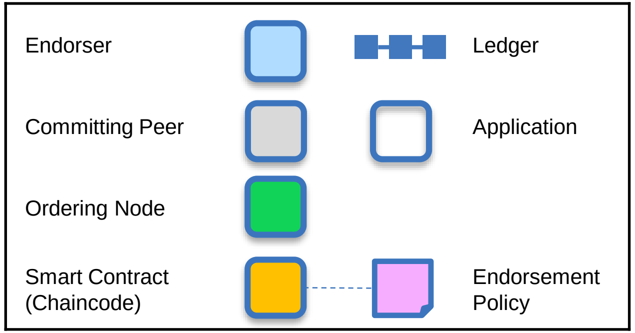


Application submits responses for ordering

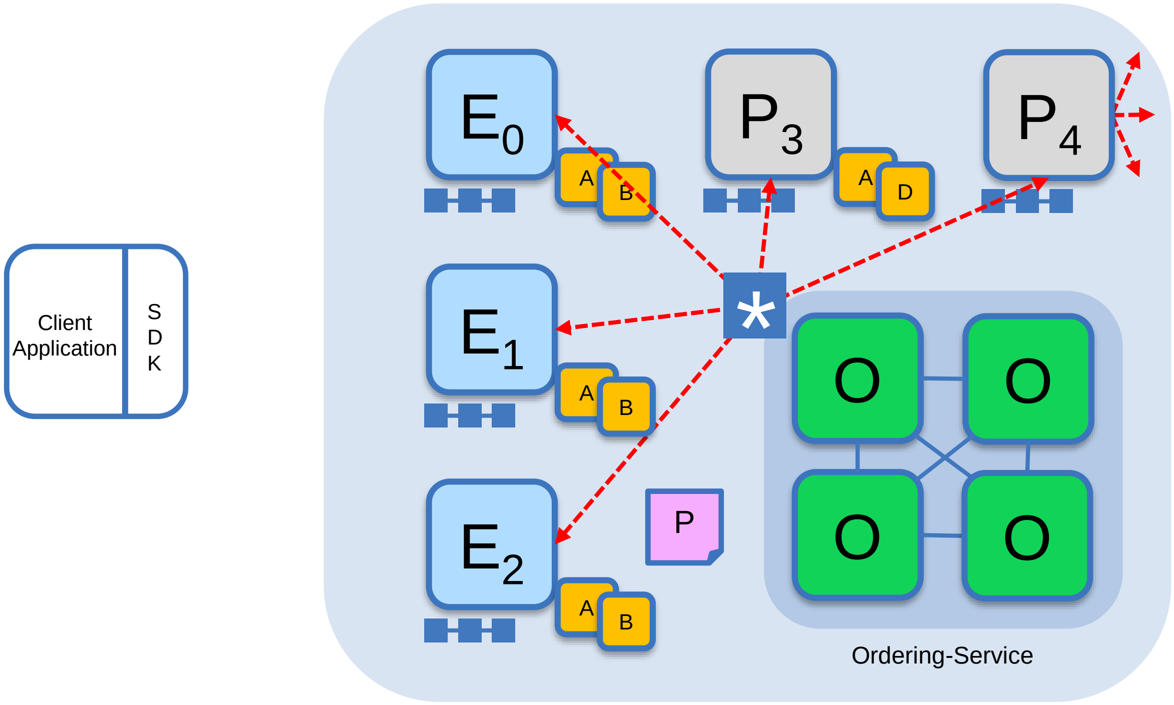
Application submits responses as a **transaction** to be ordered.

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

Key:



Sample transaction: Step 5/7 – Deliver Transaction



Hyperledger Fabric

Orderer delivers to all 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)

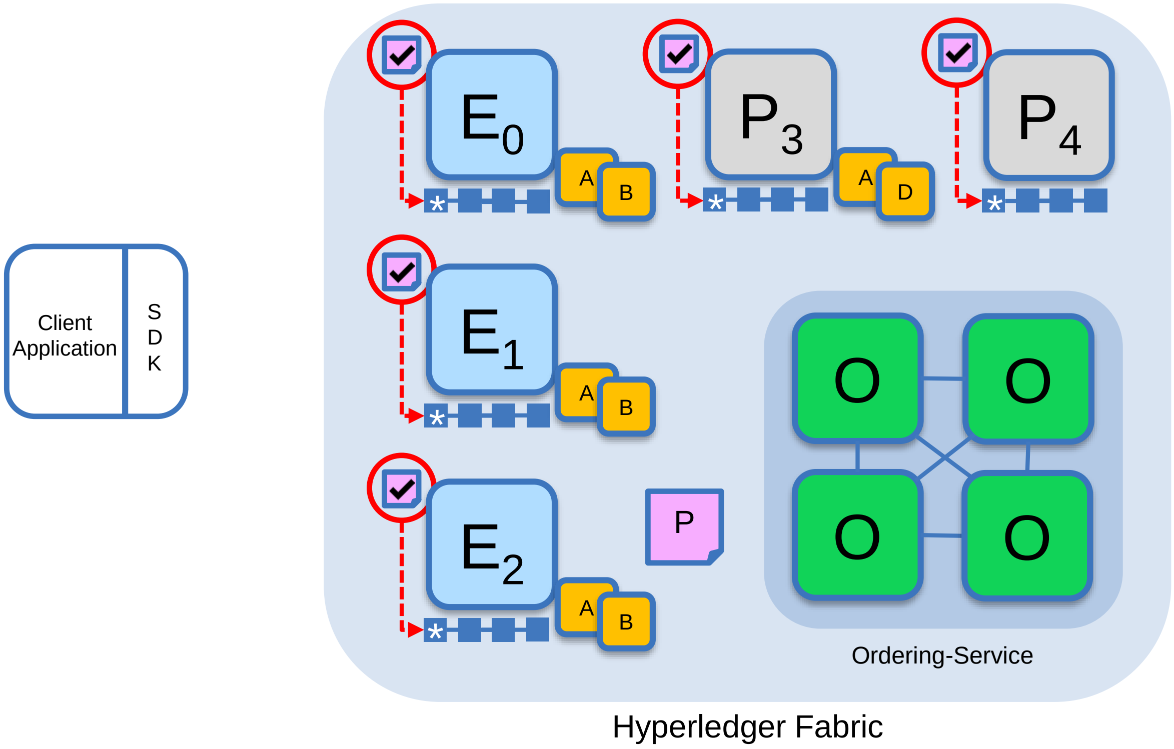
Different ordering algorithms available:

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

Key:

Endorser	🟦	■-■-■	Ledger
Committing Peer	⬜	□	Application
Ordering Node	🟩		
Smart Contract (Chaincode)	🟨	🟪	Endorsement Policy

Sample transaction: Step 6/7 – Validate Transaction



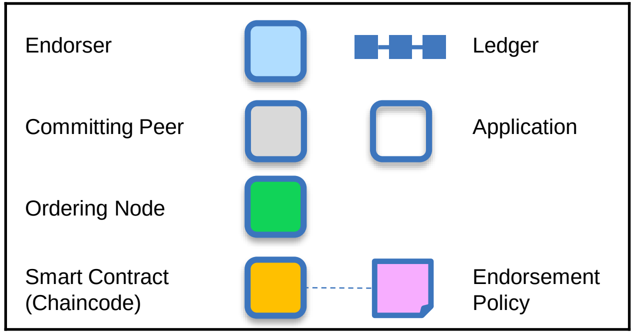
Committing peers validate transactions

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

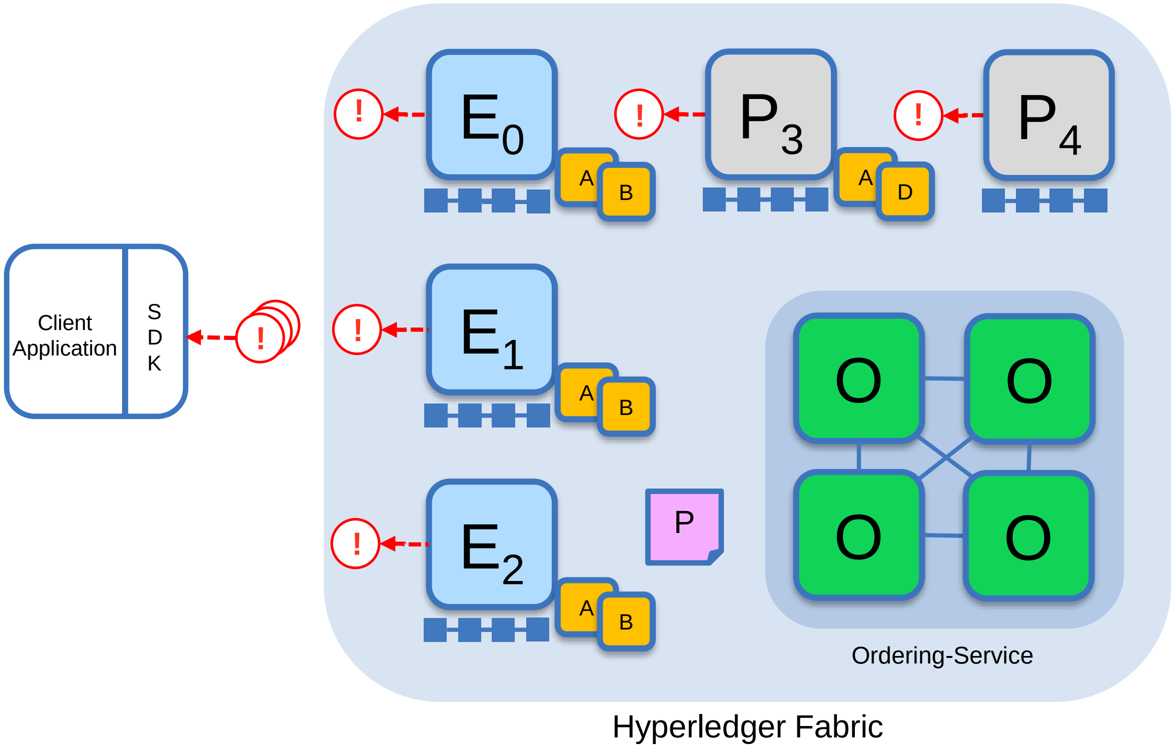
Validated transactions are applied to the world state and retained on the ledger

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

Key:



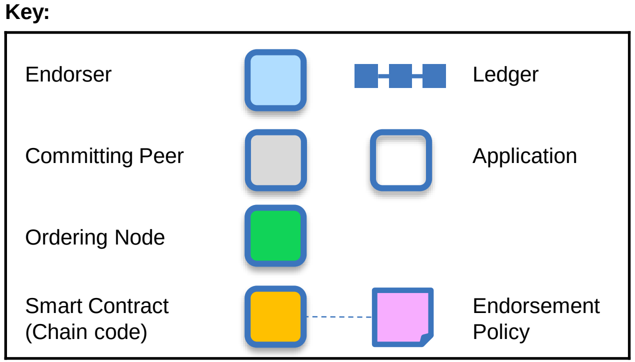
Sample transaction: Step 7/7 – Notify Transaction



Committing peers notify applications

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

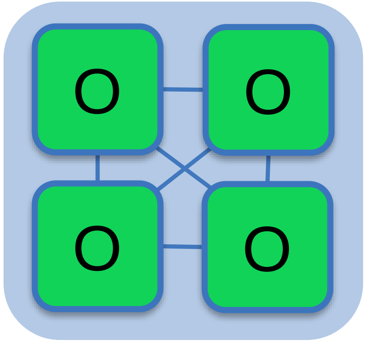
Applications will be notified by each peer to which they are connected



Channels and Ordering Service

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**

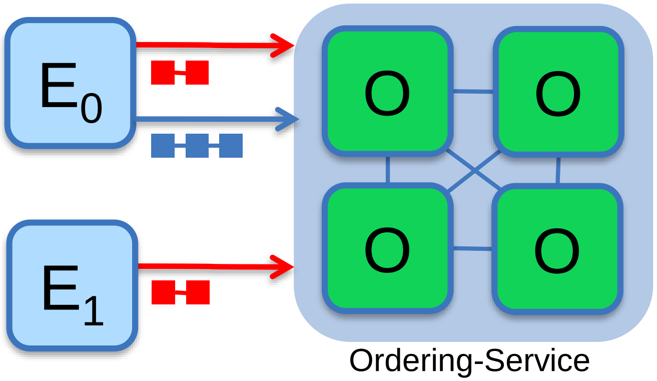
- Single node for development

– **Kafka :** Crash fault tolerant consensus

- 3 nodes minimum
- Odd number of nodes recommended

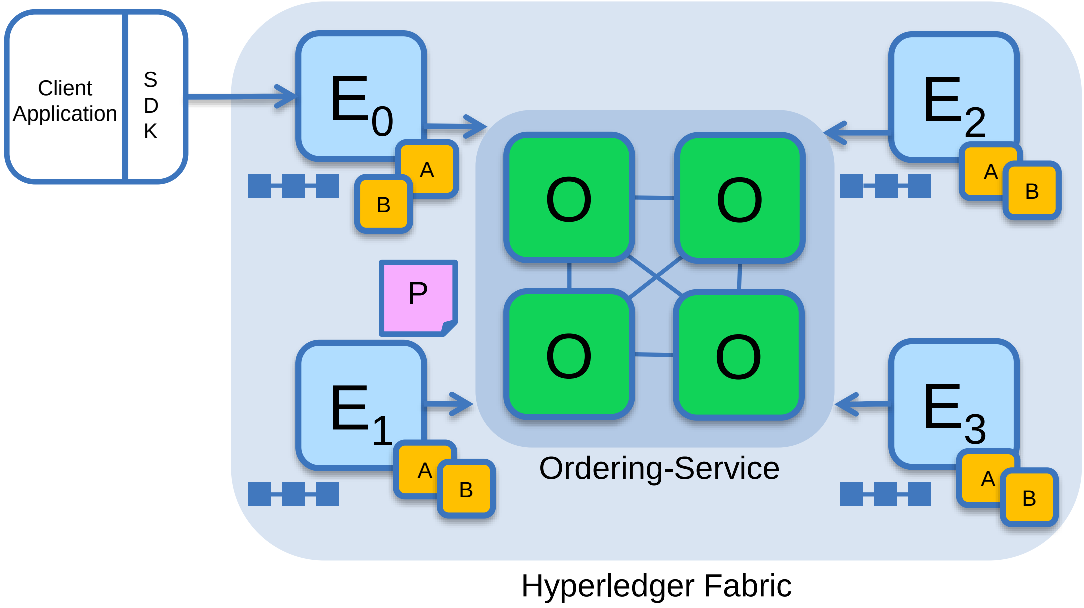
Channels

Separate channels isolate transactions on different ledgers



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

Single Channel Network

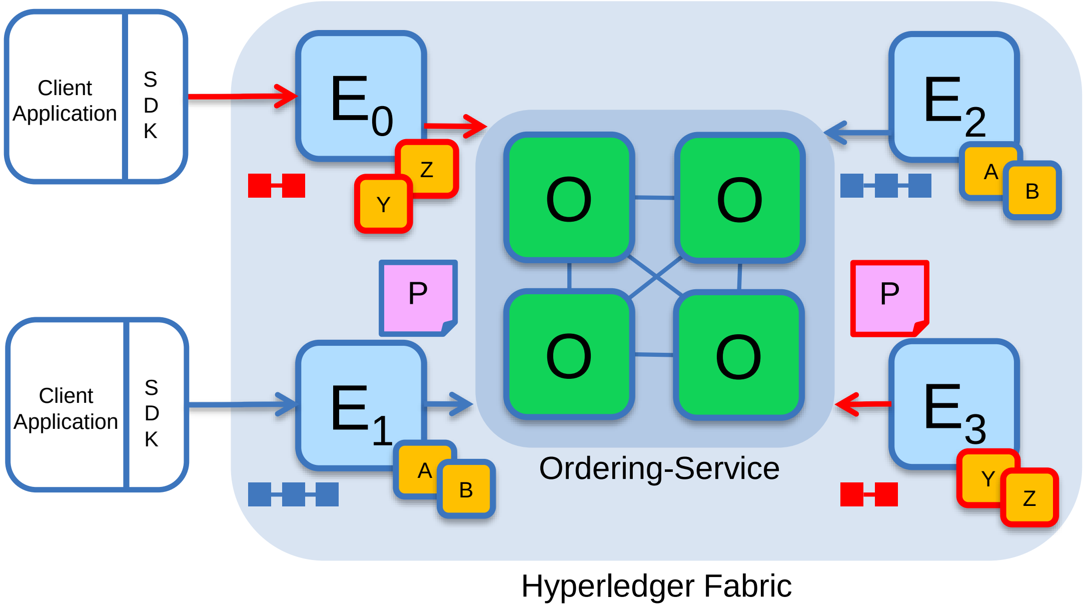


- Similar to v0.6 PBFT model
- All peers connect to the same system channel (blue).
- All peers have the same chaincode and maintain the same ledger
- Endorsement by peers $E_{0,}$ $E_{1,}$ E_2 and E_3

Key:

Endorser	🟦	■-■-■	Ledger
Committing Peer	⬜	□	Application
Ordering Node	🟩		
Smart Contract (Chaincode)	🟨	🟪	Endorsement Policy

Multi Channel Network



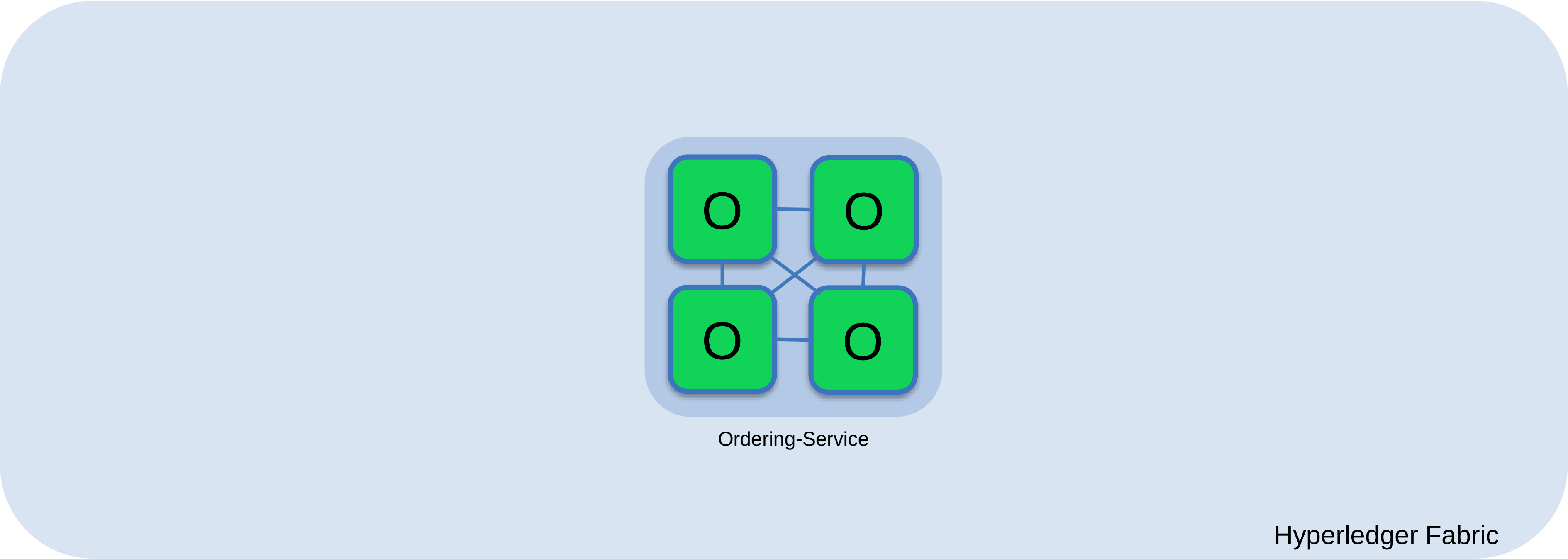
- Peers E_0 and E_3 connect to the red channel for chaincodes **Y** and **Z**
- Peers E_1 and E_2 connect to the blue channel for chaincodes **A** and **B**

Key:

Endorser	🟦	■-■-■	Ledger
Committing Peer	⬜	□	Application
Ordering Node	🟩		
Smart Contract (Chaincode)	🟨	🟪	Endorsement Policy

Network Setup

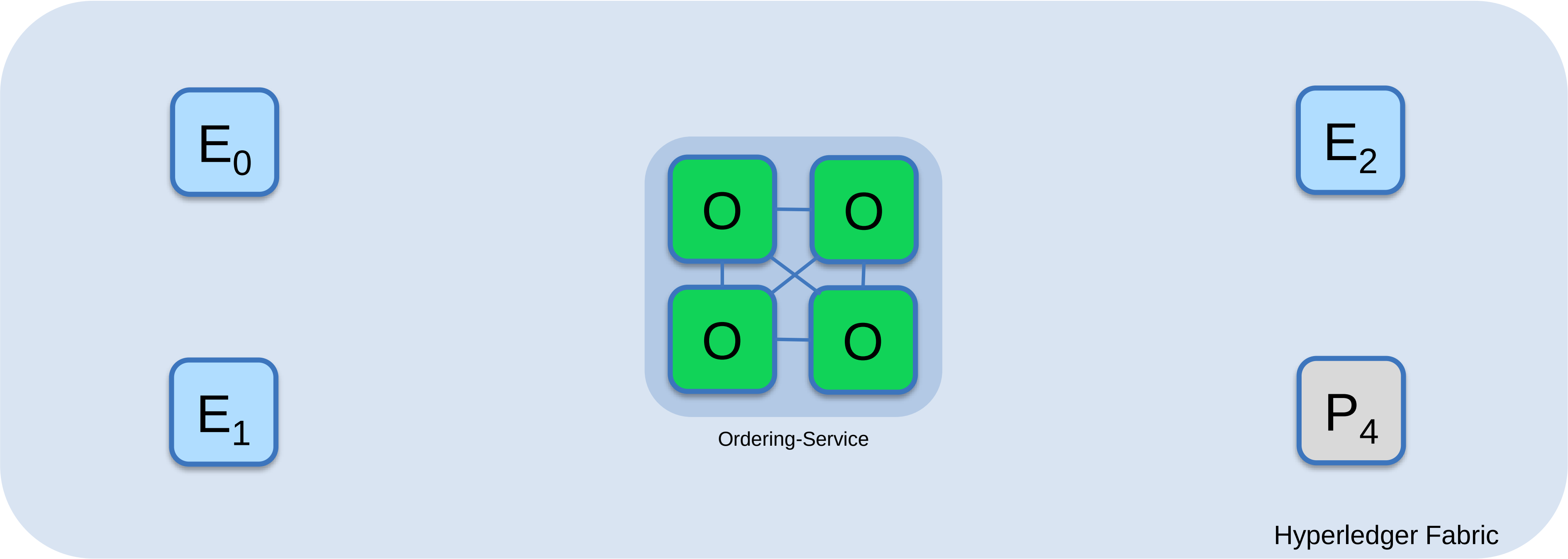
Bootstrapping the Network (1/6) – Configure & start Ordering Service



- An Ordering Service is **configured** and started for other network peers to use

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

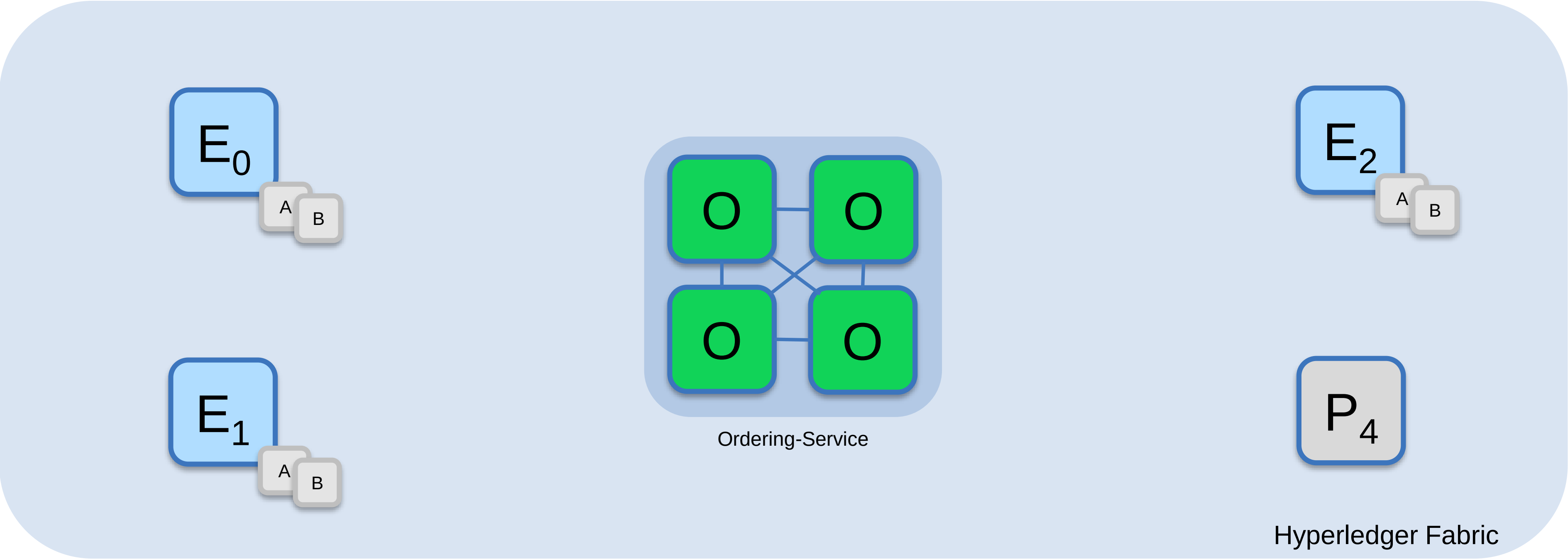
Bootstrapping the Network (2/6) – Configure and Start Peer Nodes



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

```
$ peer node start ...
```

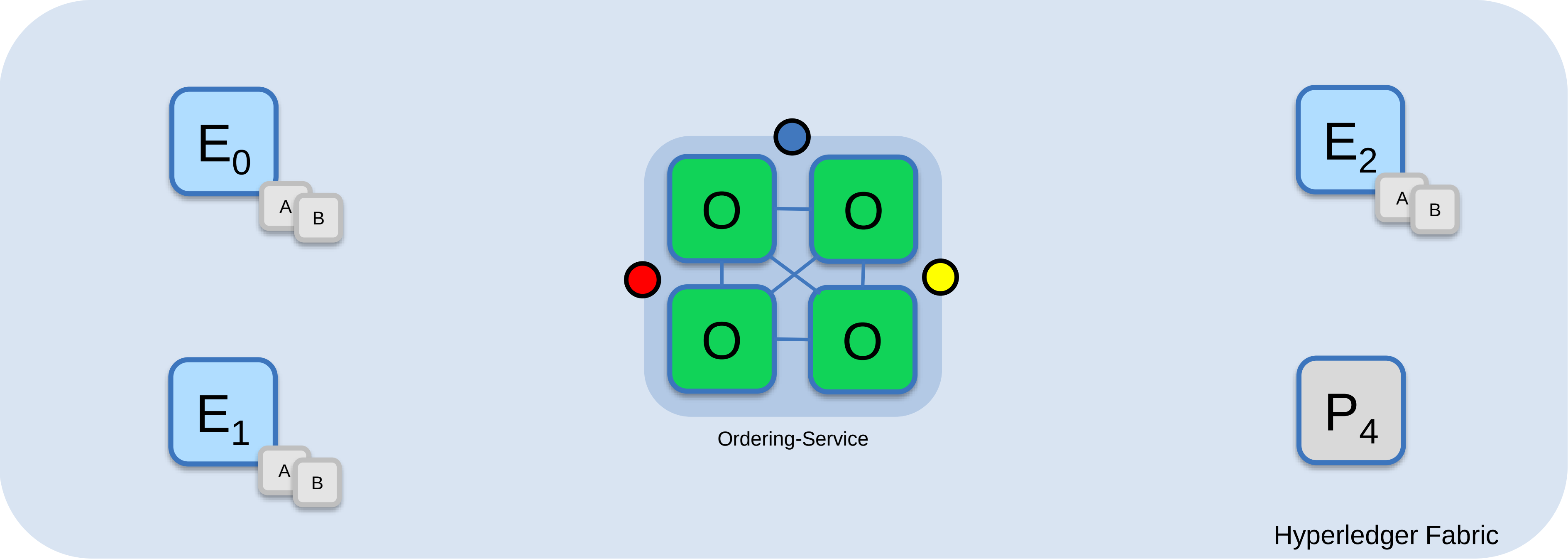
Bootstrapping the Network (3/6) – Install Chaincode



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

```
$ peer chaincode install ...
```

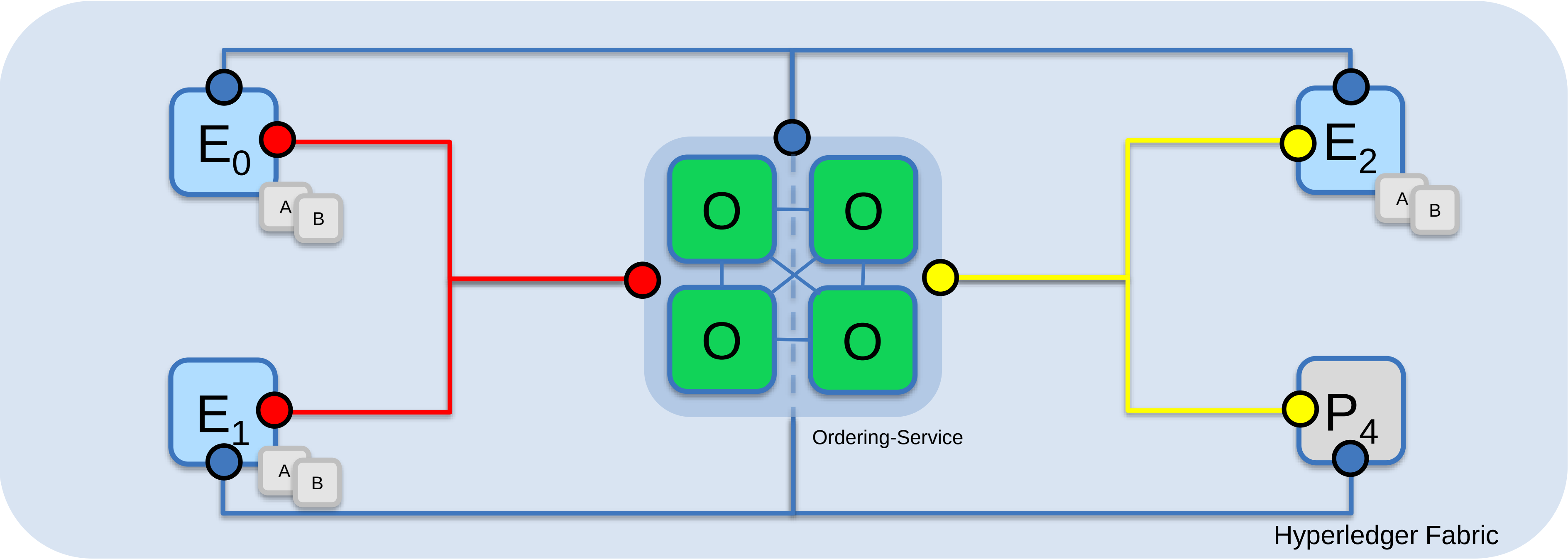
Bootstrapping the Network (4/6) – Create Channels



- Channels are **created** on the ordering service

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

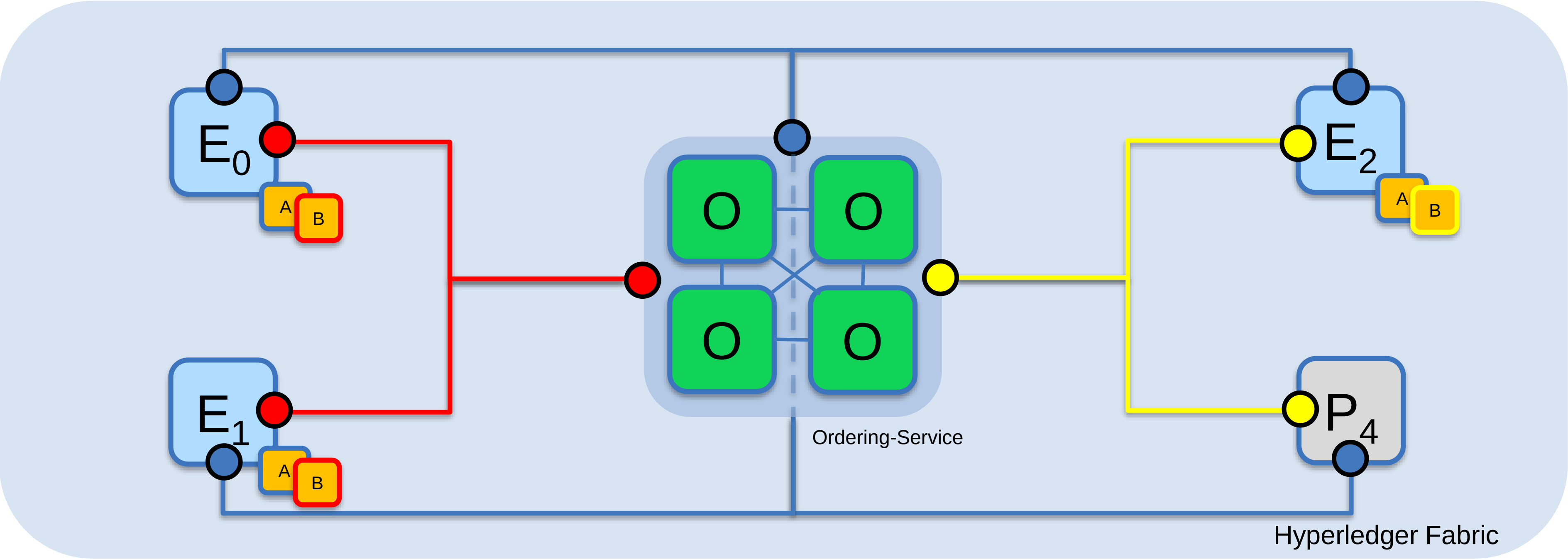
Bootstrapping the Network (5/6) – Join Channels



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

```
$ peer channel join ...
```

Bootstrapping the Network (6/6) – Instantiate Chaincode



- Peers finally **instantiate** the Chaincode on the channels they want to transact on

```
$ peer channel instantiate ... –P ‘policy’
```

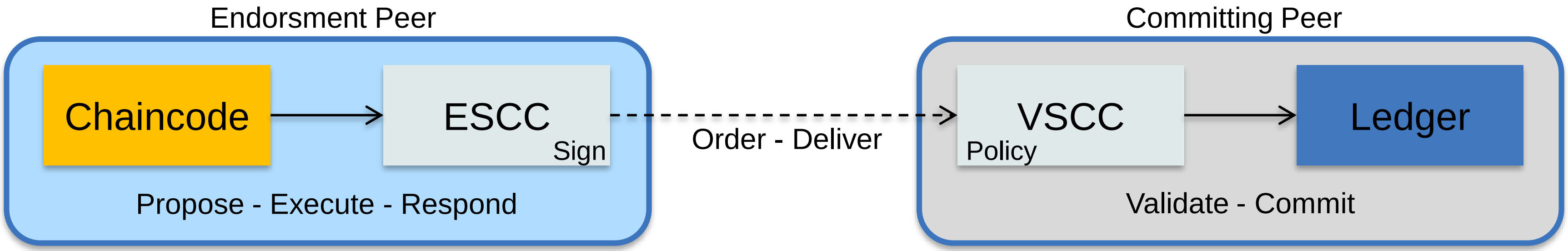
- Once instantiated a Chaincode is live and can process transaction requests
- Endorsement Policy is specified at instantiation time

Endorsement Policies

Endorsement Policies

An endorsement policy describes the conditions by which a transaction can be endorsed. A transaction can only be considered valid if it has been endorsed according to its policy.

- Each chaincode is associated with an Endorsement Policy
- Default implementation: Simple declarative language for the policy
- ESCC (Endorsement System ChainCode) signs the proposal response on the endorsing peer
- VSCC (Validation System ChainCode) validates the endorsements



Endorsement Policy Syntax

```
$ peer chaincode instantiate
-C mychannel
-n mycc
-v 1.0
-p chaincode_example02
-c '{"Args":["init","a", "100", "b","200"]}'
-P "AND('Org1MSP.member')“
```

This command instantiates the chaincode *mycc* on channel *mychannel* with the policy AND('Org1MSP.member')

Policy Syntax: **EXPR(E[, E...])**

Where **EXPR** is either **AND** or **OR** and **E** is either a principal or nested EXPR.

Principal Syntax: **MSP.ROLE**

Supported roles are: **member** and **admin**.

Where **MSP** is the MSP ID required, and **ROLE** is either “member” or “admin”.

Endorsement Policy Examples

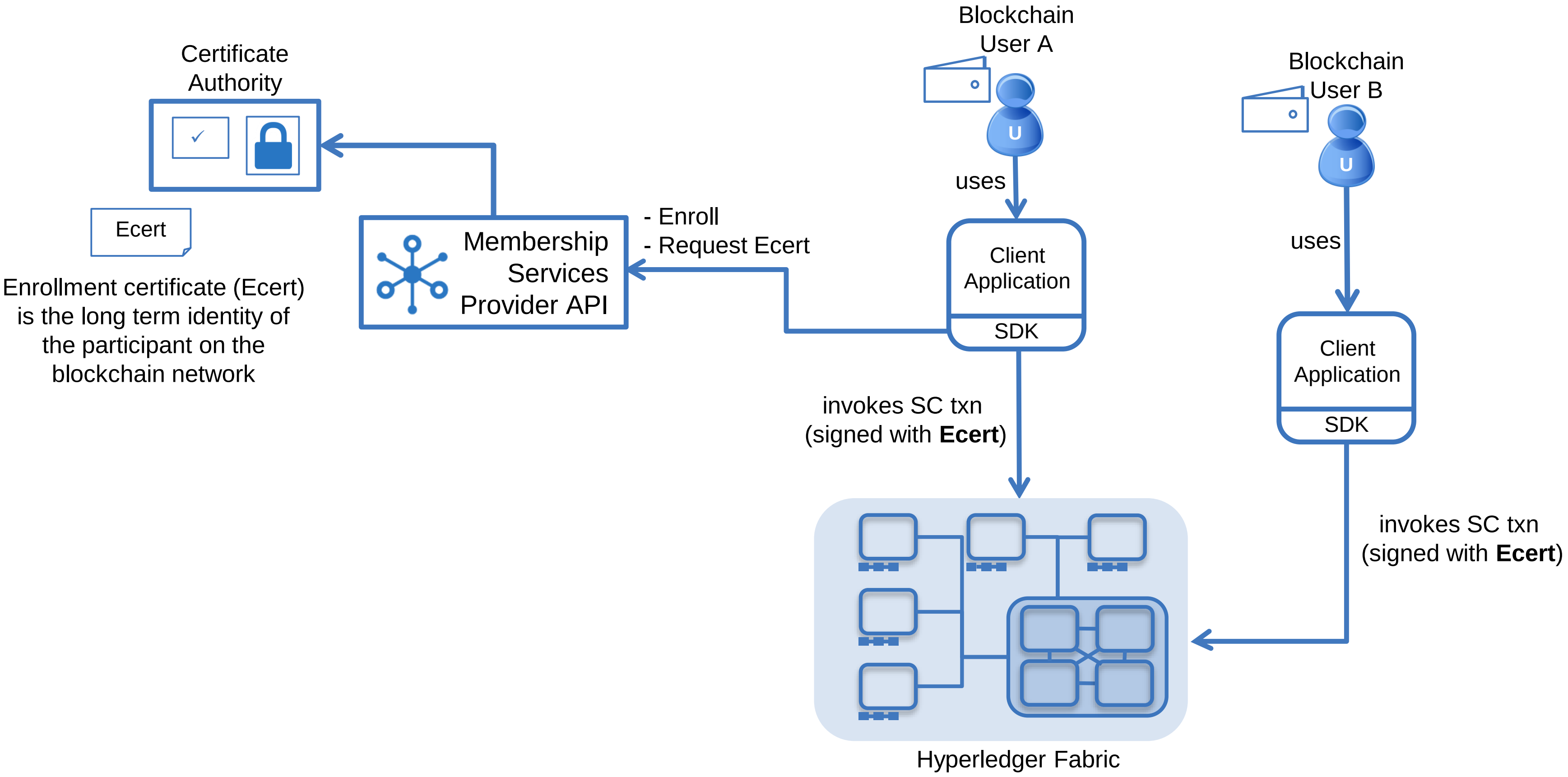
Examples of policies:

- Request 1 signature from all three principals
  - AND('Org1.member', 'Org2.member', 'Org3.member')
- Request 1 signature from either one of the two principals
  - OR('Org1.member', 'Org2.member')
- Request either one signature from a member of the Org1 MSP or (1 signature from a member of the Org2 MSP and 1 signature from a member of the Org3 MSP)
  - OR('Org1.member', AND('Org2.member', 'Org3.member'))

Permissioned Ledger Access

Transaction and identity privacy

Membership Services Overview



Certificate Authority

Ecert

Enrollment certificate (Ecert) is the long term identity of the participant on the blockchain network

Membership Services Provider API

- Enroll
- Request Ecert

Blockchain User A

uses

Client Application

SDK

invokes SC txn (signed with **Ecert**)

Blockchain User B

uses

Client Application

SDK

invokes SC txn (signed with **Ecert**)

Hyperledger Fabric

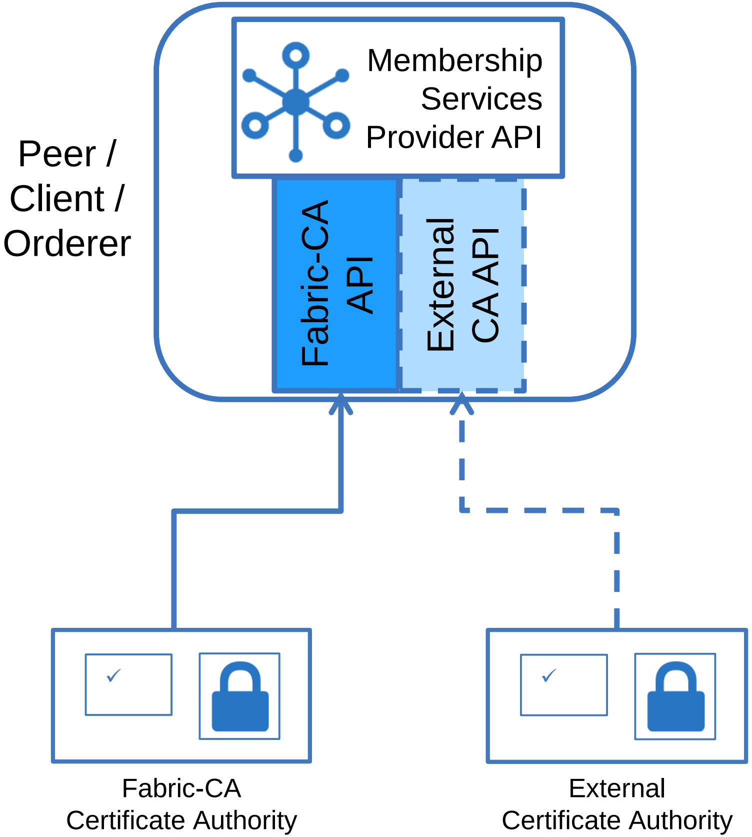
Transaction and Identity Privacy

- Enrollment Certificates, Ecerts
  - Long term identity
  - Can be obtained offline, bring-your-own-identity

- Permissioned Interactions
  - Users sign with their Ecert

- Membership Services
  - Abstract layer to credential providers

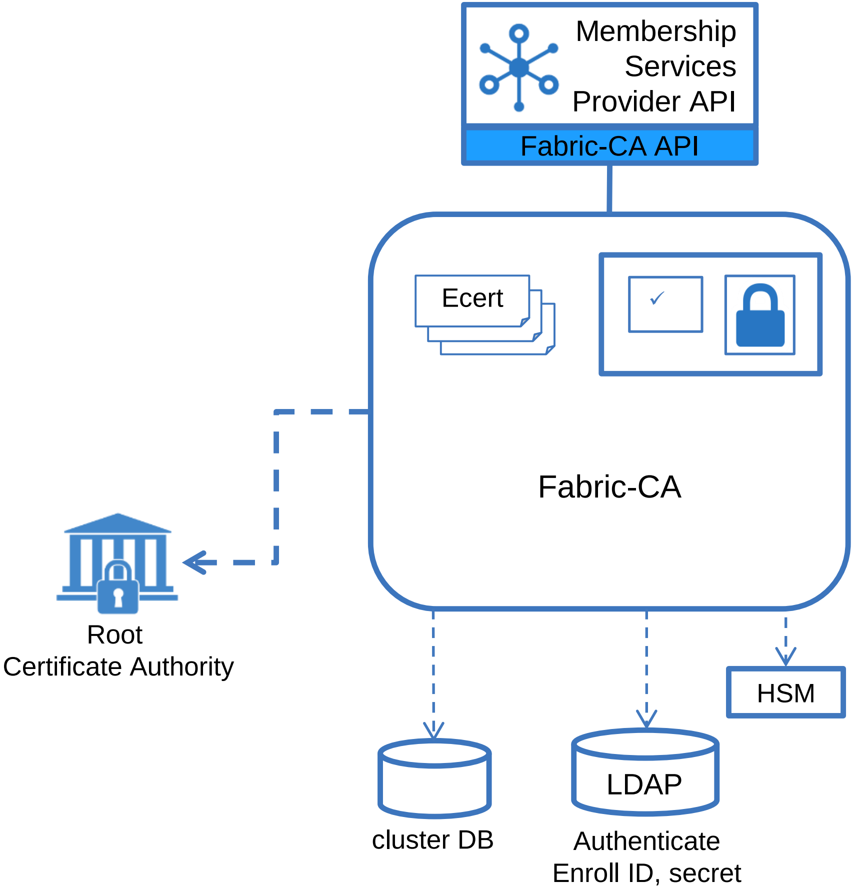
Membership Services Provider API



Membership Services Provider API

- Pluggable interface supporting a range of credential architectures
- Default implementation calls Fabric-CA.
- Governs identity for Peers and Users.
- Provides:
  - User authentication
  - User credential validation
  - Signature generation and verification
  - Optional credential issuance
- Additional offline enrollment options possible (eg File System).

Fabric-CA Details



Fabric-CA

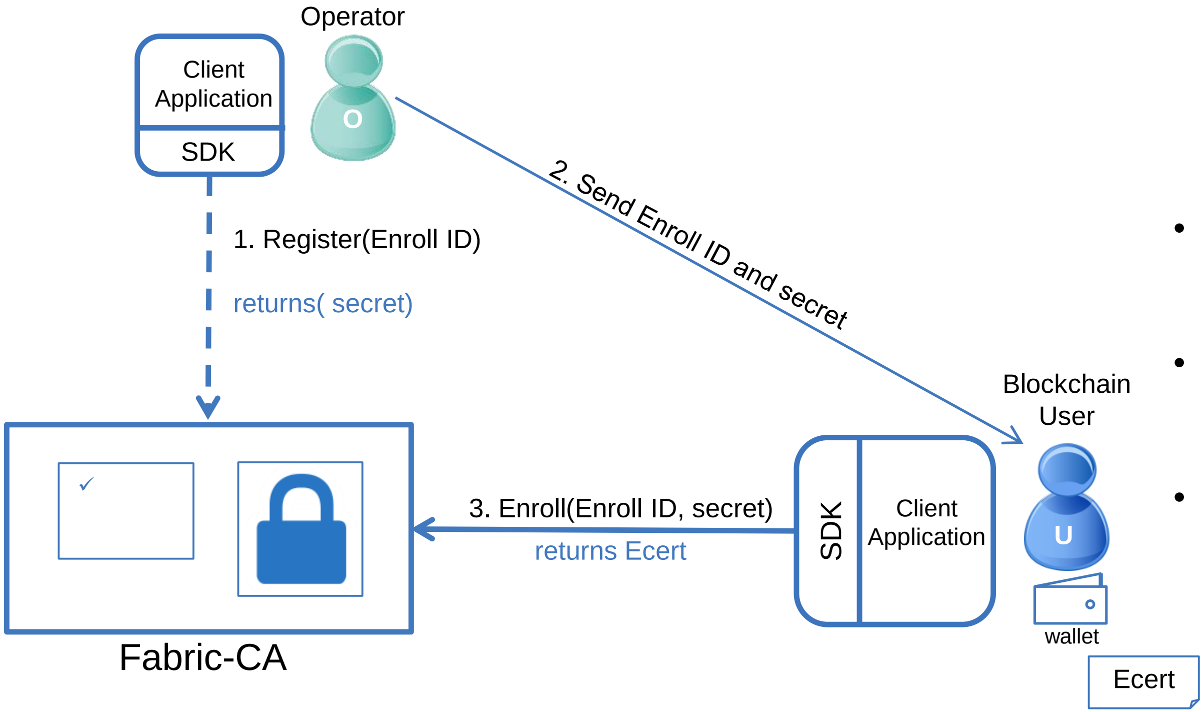
- Default implementation of the Membership Services Provider Interface.
- Issues Ecerts (long-term identity
- Supports clustering for HA characteristics
- Supports LDAP for user authentication
- Supports HSM

Fabric-CA

Certificate Authority

- Issues Ecerts and manages renewal and revocation
- Supports:
  - Clustering for HA characteristics
  - LDAP server for registration and enrollment
  - Hardware Security Modules

New User Registration and Enrollment



Registration and Enrollment

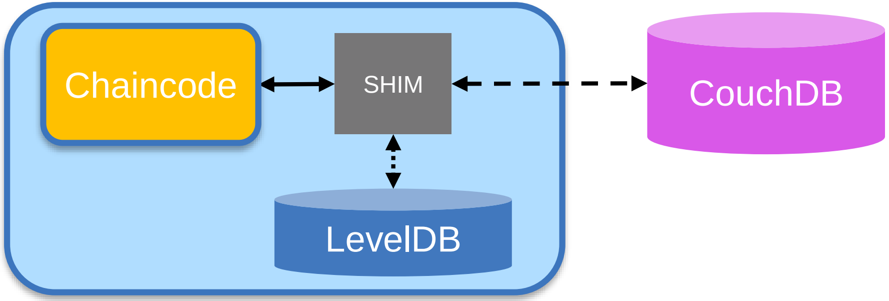
- Admin registers new user with Enroll ID
- User enrolls and receives credentials
- Additional offline registration and enrollment options available

Pluggable world state

How is data managed on the ledger?

WorldState Database

- Pluggable worldstate database
- Default embedded key/value implementation using LevelDB
  - Support for keyed queries, but cannot query on value
- Support for Apache CouchDB
  - Full query support on key and value (JSON documents)
  - Meets a large range of chaincode, auditing, and reporting requirements
  - Will support reporting and analytics via data replication to an analytics engine such as Spark (future)
  - Id/document data model compatible with existing chaincode key/value programming model



Continuing your education journey…

https://ibm.co/2fl7iji

- Links to IBM Think Academy
- Blockchain use cases
- Learn about Blockchain ecosystem

https://ibm.co/2sxniPZ

- Blockchain essentials course with Badge program

https://ibm.co/2hAp82m

- Developer journey with complete code and architecture diagram

https://ibm.co/2wlpw8c

- Understand Blockchain and BPM

https://ibm.co/2cg8hue

- Blockchain development center
- Deploy your first Blockchain code

Some more links -

Want to learn more? How much time do you have?

5 minutes? Read a primer on distributed ledger technology

10 minutes? Learn to distinguish Bitcoin vs. blockchain for business

20 minutes? Check out this intro to distributed ledgers

45 minutes? Download and read the *Blockchain for Dummies* e-book

2 hours? Take the Blockchain essentials course (and earn a badge!)

Car lease Demo- https://github.com/IBM-Blockchain/car-lease-demo.git

Thank you!

www.ibm.com/blockchain

developer.ibm.com/blockchain

www.hyperledger.org

Further Information – Use case Links

Northern Trust: http://www-03.ibm.com/press/us/en/pressrelease/51655.wss

Maersk: http://www-03.ibm.com/press/us/en/pressrelease/51712.wss

HSBC, Bank of America, IDA: http://www.coindesk.com/hsbc-bank-america-blockchain-supply-chain/

ABN AMRO: https://www.abnamro.com/en/newsroom/blogs/arjan-van-os/2016/walking-the-walk-exploring-the-power-of-blockchain.html

Crédit Mutuel Arkéa: http://www.coindesk.com/ibm-completes-blockchain-trial-french-bank-credit-mutuel/

JPX: http://www.ibm.com/press/us/en/pressrelease/49088.wss

Kouvola Innovation: http://www.ibm.com/press/us/en/pressrelease/49029.wss

London Stock Exchange: http://www.ibtimes.co.uk/linux-foundation-blockchain-consortium-digital-asset-ibm-credits-london-stock-exchange-board-1533798

Mizuho: http://www.coindesk.com/mizuho-digital-currency-powered-blockchain-settlement/

IBM Global Finance: http://www.coindesk.com/ibm-building-blockchain-dispute-resolution-system/

Everledger: https://www-03.ibm.com/press/us/en/pressrelease/50169.wss

Bank of Tokyo Mitsubishi: https://www-03.ibm.com/press/us/en/pressrelease/50544.wss

China UnionPay: http://www.coindesk.com/ibm-china-unionpay-blockchain-loyalty-exchange/

CLS: http://www.coindesk.com/cls-to-develop-blockchain-payment-service-on-ibm-fabric/

UBS: http://www.coindesk.com/ubs-blockchain-prototype-trade/