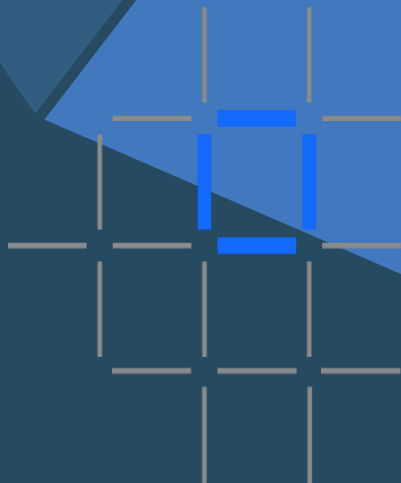




# IBM blockchain foundation developer

Video presentation slides



# Business Networks, Wealth, and Markets

## – Business Networks

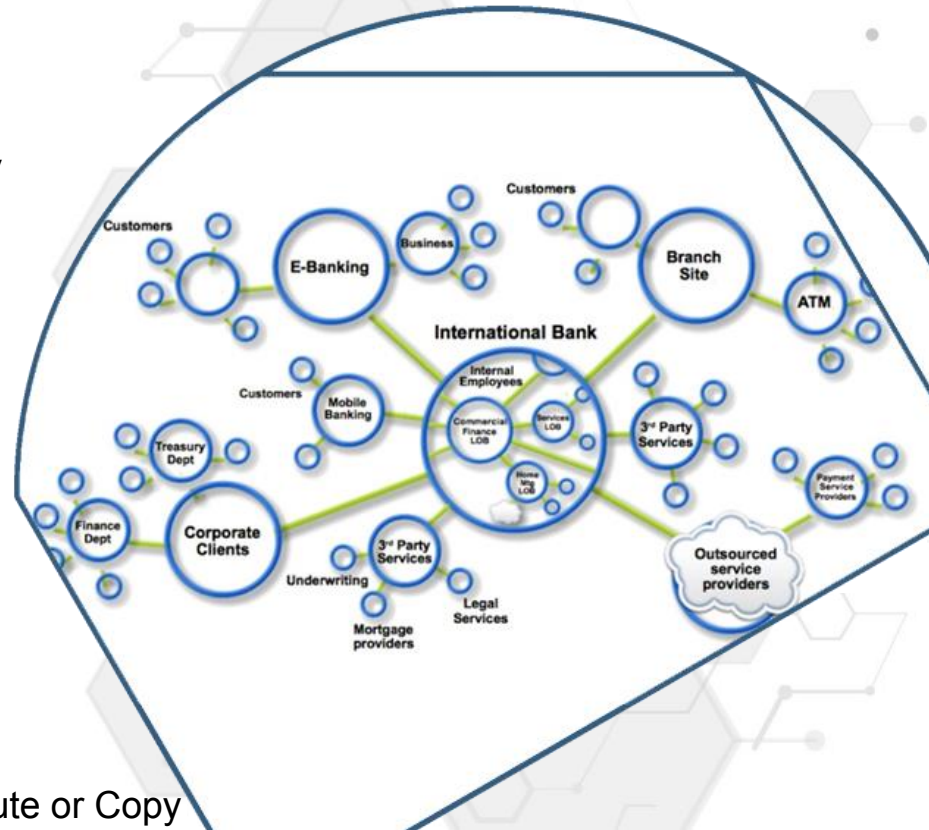
- Participants are customers, suppliers, banks, partners
- Cross geography & regulatory boundary

## – Wealth

goods & services across business network in transactions and contracts

## – Markets

- Public (fruit market, car auction), or
- Private (supply chain financing)



# 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



## Cash is also an asset

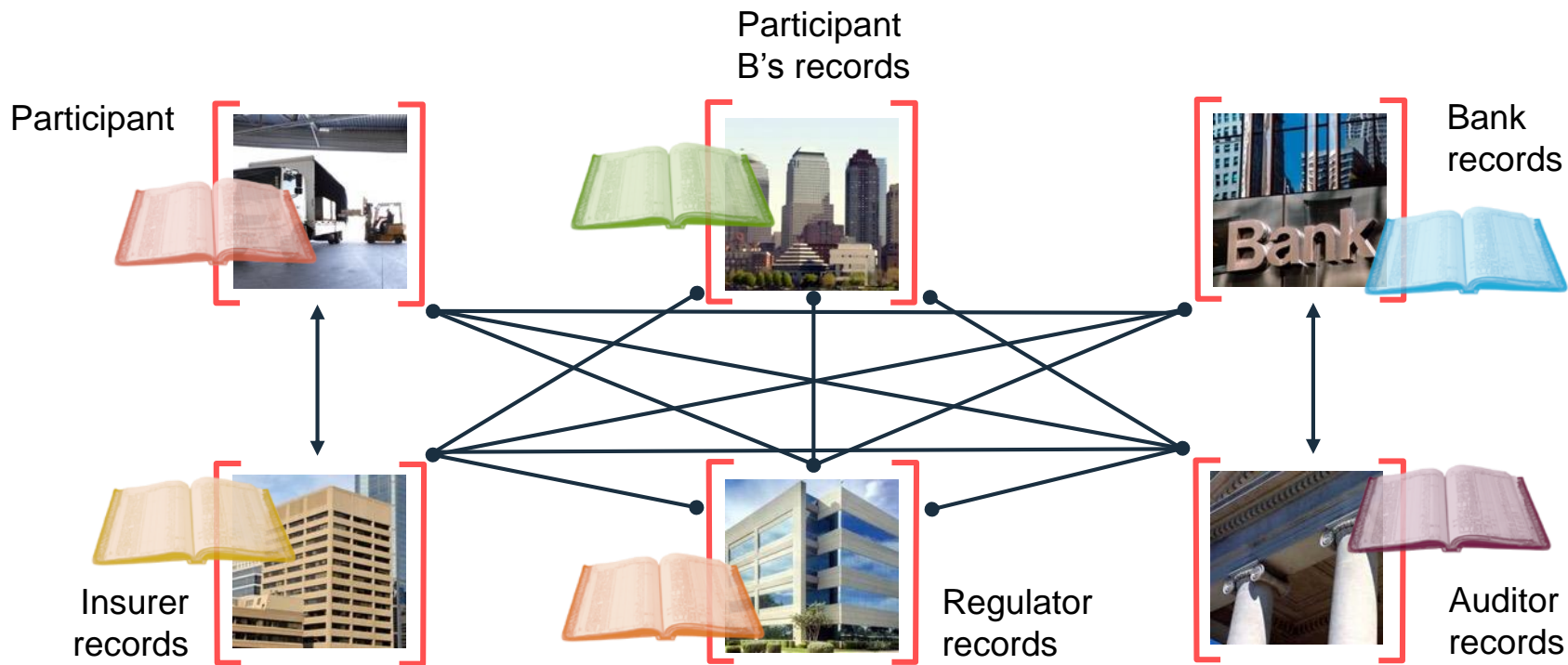
- Has property of anonymity



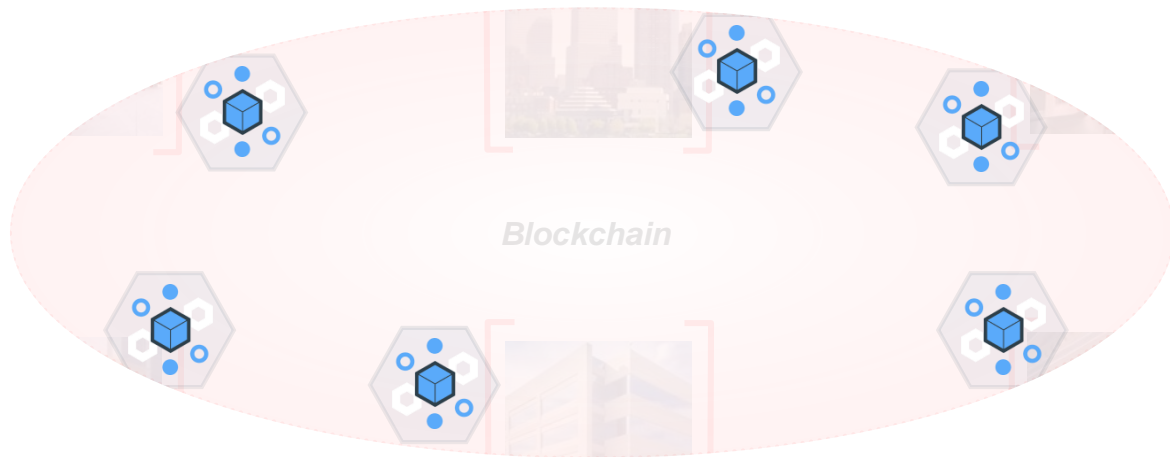
# Introducing Blockchain



# Problem ...



... inefficient, expensive, vulnerable





An unregulated shadow-currency

The first blockchain application

**Blockchain for business** differs in key areas:

*Identity* over anonymity

*Selective endorsement* over proof of work

*Assets* over cryptocurrency





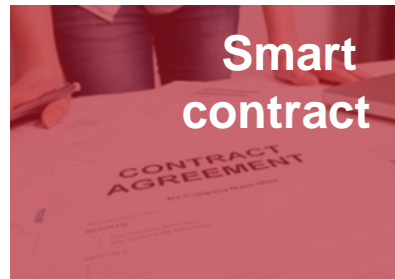
# Requirements of Blockchain for Business

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



What

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

# Blockchain Benefits



## **Saves time**

Transaction time  
from days to near  
instantaneous



## **Reduces cost**

Overheads and  
cost intermediaries



## **Reduces risk**

Tampering, fraud,  
& cyber crime



## **Increases trust**

Through shared  
processes and  
recordkeeping





# Example: Shared Reference Data

## What

- Competitors/collaborators in a business network need to share reference data, e.g. bank routing codes
- Each member maintains their own codes, and forwards changes to a central authority for collection and distribution
- An information subset can be owned by organizations

## How

- Each participant maintains their own codes within a Blockchain network
- Blockchain creates single view of entire dataset

## Benefits

1. Consolidated, consistent dataset reduces errors
2. Near real-time access to reference data
3. Naturally supports code editing and routing code transfers between participants



# Example: Supply Chain

## What

- Provenance of each component part in complex system hard to track
- Manufacturer, production date, batch and even the manufacturing machine program


## How

- Blockchain holds complete provenance details of each component part
- Accessible by each manufacturer in the production process, the aircraft owners, maintainers and government regulators

## Benefits

1. Trust increased, no authority "owns" provenance
2. Improvement in system utilization
3. Recalls "specific" rather than cross fleet





# Example: Audit and Compliance

## What

- Financial data in a large organization dispersed throughout many divisions and geographies
- Audit and Compliance needs indelible record of all key transactions over reporting period

## How

- Blockchain collects transaction records from diverse set of financial systems
- Append-only and tamperproof qualities create high confidence financial audit trail
- Privacy features to ensure authorized user access

## Benefits

1. Lowers cost of audit and regulatory compliance
2. Provides “seek and find” access to auditors and regulators
3. Changes nature of compliance from passive to active

## Benefits

- Bank handling letters of credit (LOC) wants to offer them to a wider range of clients including startups
- Currently constrained by costs & the time to execute

1. Increase speed of execution  
(less than 1 day)
2. Vastly reduced cost
3. Reduced risk,  
e.g. currency fluctuations
4. Value added services,  
e.g. incremental payment

- Blockchain provides common ledger for letters of credit
- Allows all counter-parties to have the same validated record of transaction and fulfillment

# Further Examples by Selected Industry

## Potential use cases



### Financial

Trade Finance  
Cross currency  
payments  
Mortgages



### Public Sector

Asset  
Registration  
Citizen Identity  
Medical records  
Medicine supply



### Retail

Supply chain  
Loyalty programs  
Information  
sharing (supplier  
– retailer)



### Insurance

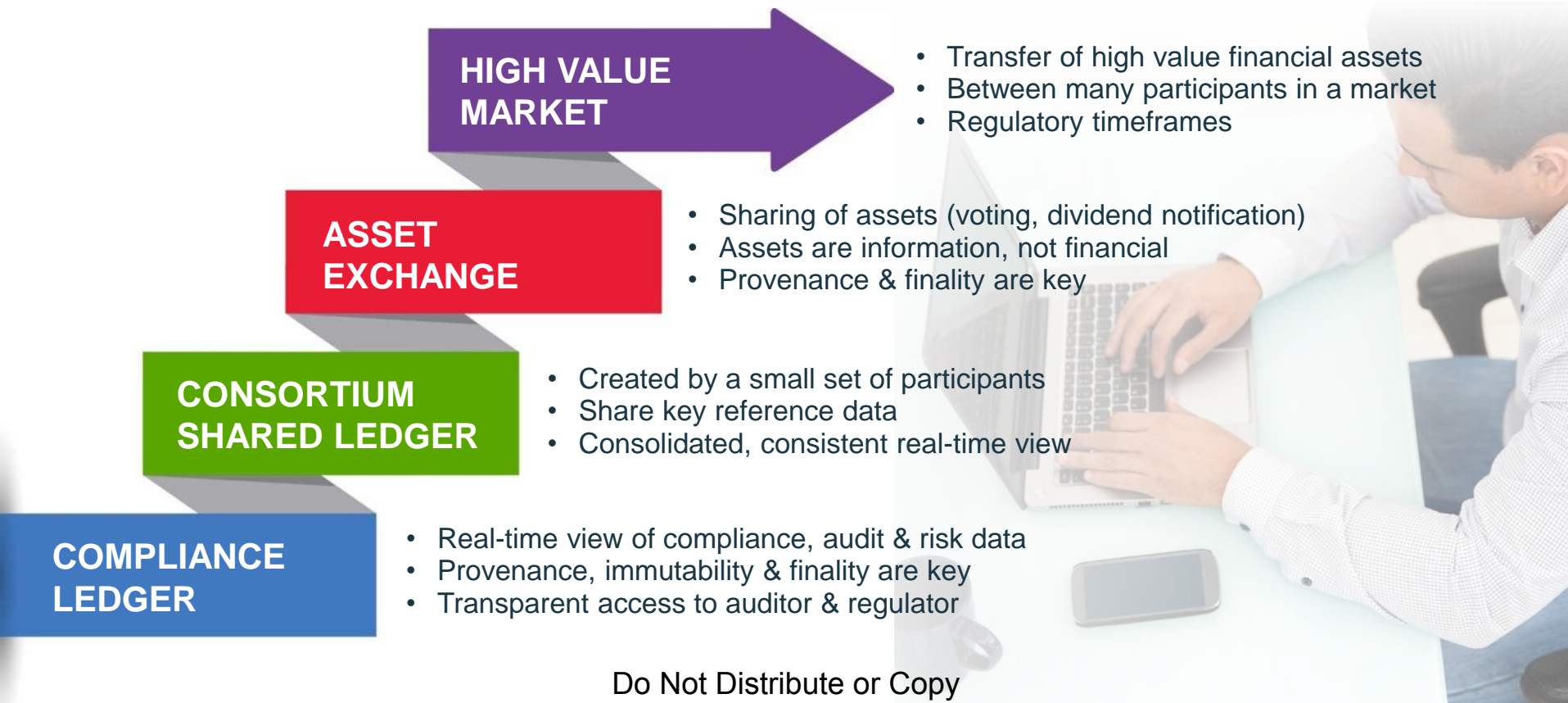
Claims  
processing  
Risk provenance  
Asset usage  
history  
Claims file



### Manufacturing

Supply chain  
Product parts  
Maintenance  
tracking

# Patterns for Customer Adoption



# 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

# How IBM Can Help



## Technology



**HYPERLEDGER**

BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Hyperledger  
Fabric

Hyperledger  
Composer



## Hosting and Support



High Security  
Business Network



IBM Cloud



docker



## Making blockchain real for clients



Garages



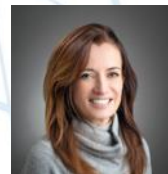
Engagement

# 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
- One active framework (“Fabric”) and seven projects in incubation
- IBM is a premier member of Hyperledger



**Brian Behlendorf**  
*Executive Director*



**Blythe Masters**  
*Board Chair*



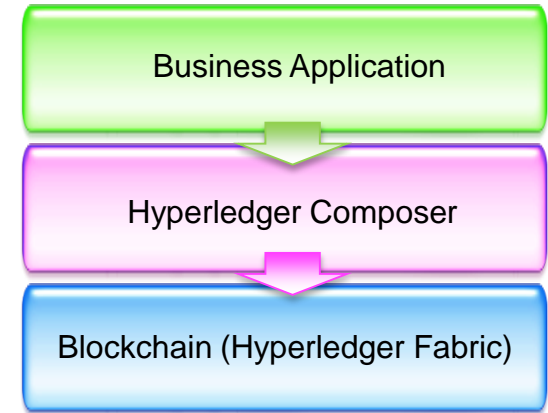
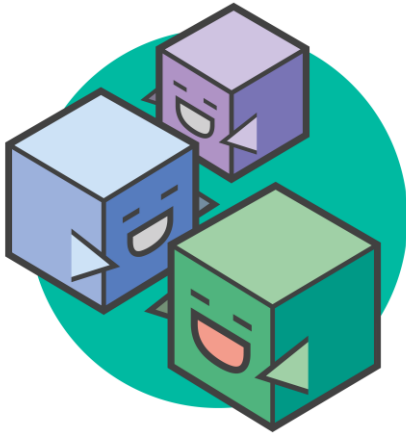
**Chris Ferris**  
*TSC Chair*

[www.hyperledger.org](http://www.hyperledger.org)



# Hyperledger Composer: Accelerating time to value

- 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



- 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 one of eight Hyperledger projects
- Try a demo now! - <http://composer-playground.mybluemix.net/>



# IBM engagement model overview



1. Discuss Blockchain technology
2. Explore customer business model
3. Show Blockchain Application demo

**Remote**



1. Understand Blockchain concepts & elements
2. Hands on with Blockchain on Bluemix
3. Standard demo customization

**Digital**



1. Design Thinking workshop to define business challenge
2. Agile iterations incrementally build project functionality
3. Enterprise integration

**Face to face**



1. Scale up pilot or Scale out to new projects
2. Business Process Re-engineering
3. Systems Integration

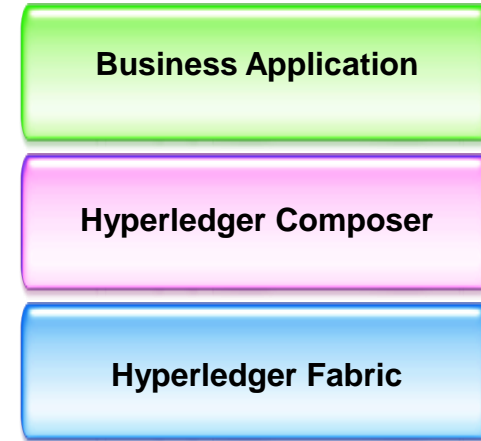
**Face to face**



# Hyperledger Composer

# What is Hyperledger Composer?

- **Blockchains provide a low-level interface for business applications**
  - Smart contract code run on a distributed processing system
  - Inputs go into an immutable ledger; outputs to a data store
  - Applications are built on top of a low level of abstraction
- **Hyperledger Composer**
  - A suite of high level application abstractions for business networks
  - Emphasis on business-centric vocabulary for quick solution creation
- **Features**
  - Model your business network, test and deploy
  - Applications use APIs to interact with a business network
  - Integrate existing systems of record using loopback/REST
- **Open Tools, APIs and libraries to support these activities**
  - Exploits Hyperledger Fabric blockchain technology
  - Fully open and part of Linux Foundation Hyperledger



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

# Benefits of Hyperledger Composer



## **Increases understanding**

Bridges simply from business concepts to blockchain



## **Saves time**

Develop blockchain applications more quickly and cheaply



## **Reduces risk**

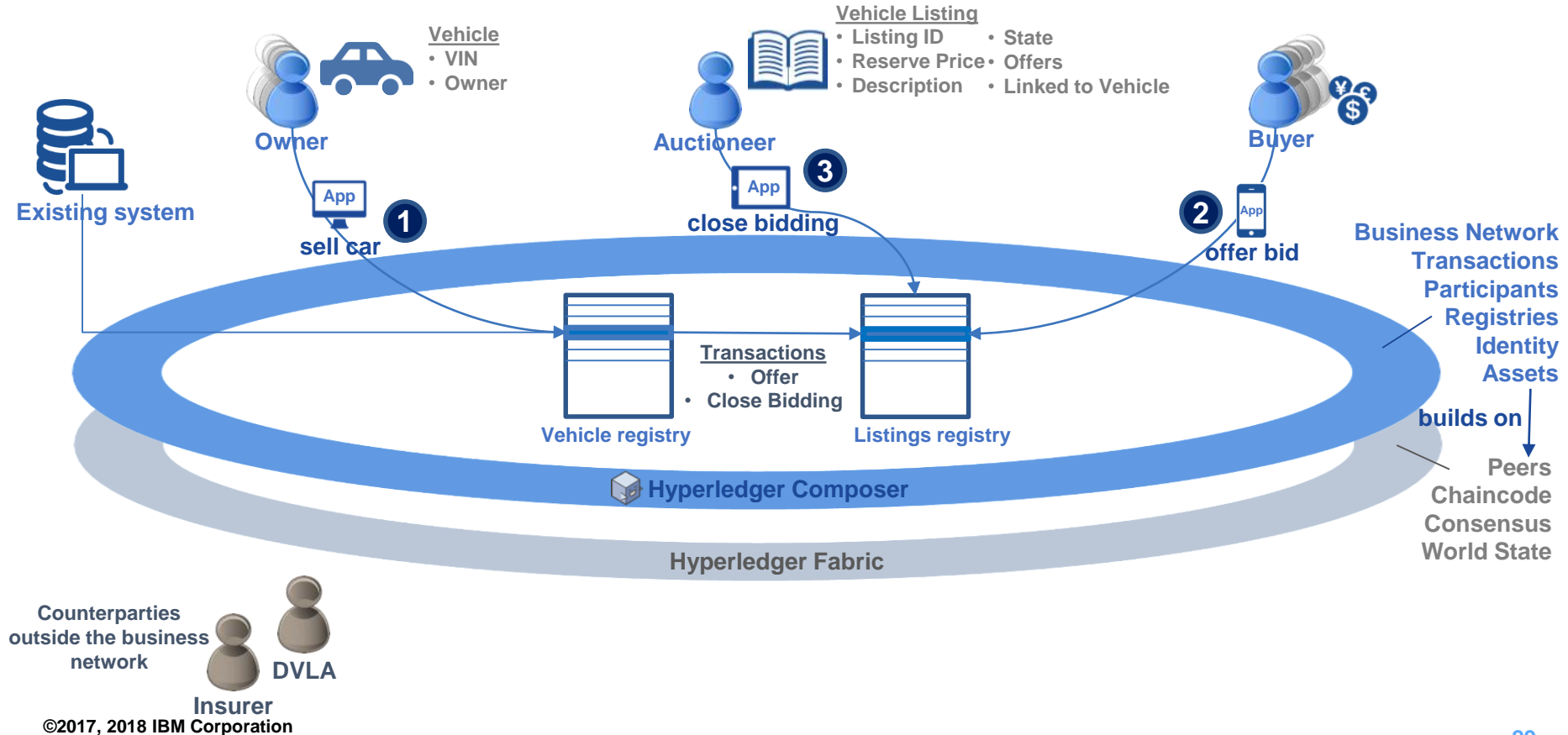
Well tested, efficient design conforms to best practice



## **Increases flexibility**


Higher level abstraction makes it easier to iterate


# An Example Business Network – Car Auction Market

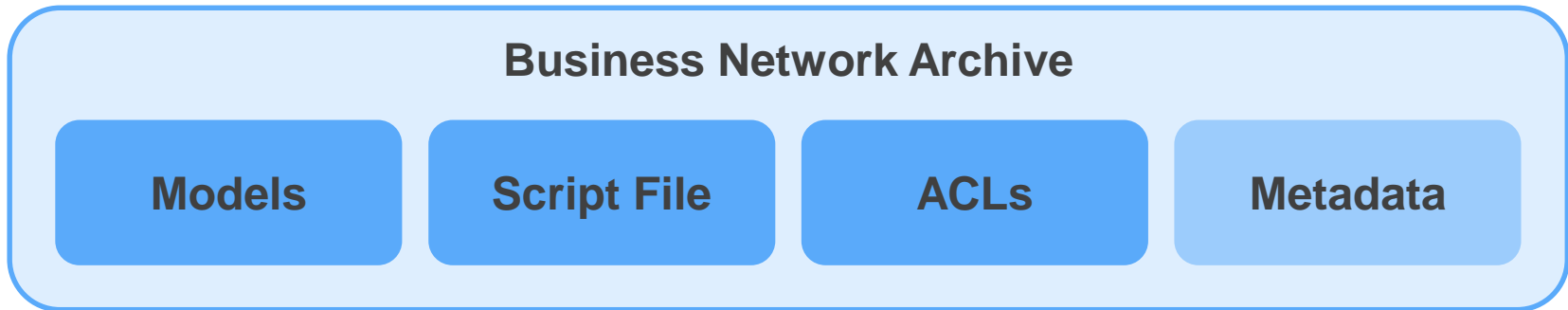


# Conceptual Components and Structure of Composer

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** provision the target environment and may manage deploy



# Extensive, Familiar, Open Development Tool set

```
asset Animal identi
  o String animal
  o AnimalType sp
  o MovementStatu
  o ProductionTyp
```

Data modelling



JavaScript  
business logic



Web playground

composer-client  
composer-admin



Client libraries



Editor support

\$ composer

CLI utilities



Code generation

Powered by



LoopBack  
Node.js Framework



Swagger

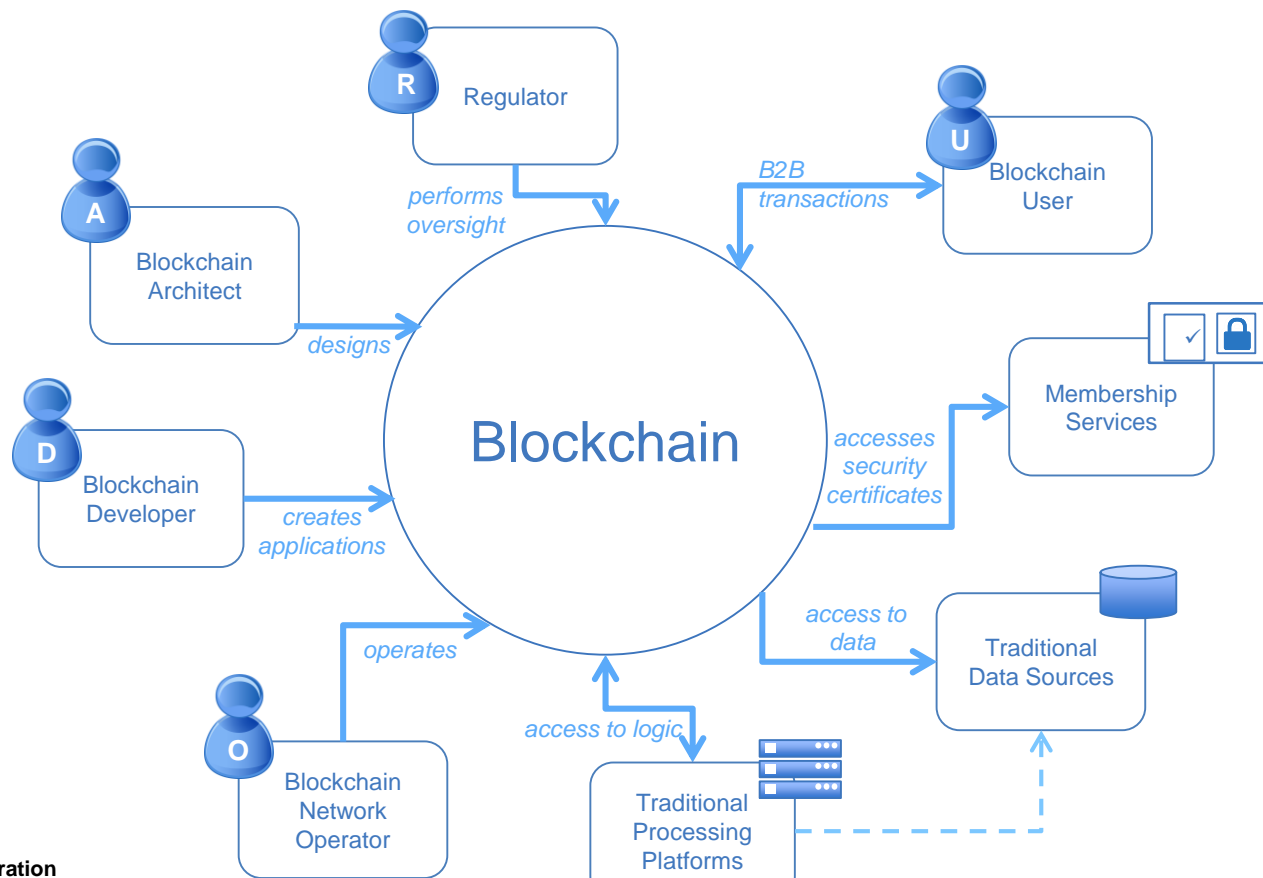
Existing systems and  
data











# Blockchain Fabric Development






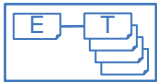




# Actors in a Blockchain Solution



# Actors in a Blockchain Solution

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		An existing data system which may provide data to influence the behavior of smart contracts.

# Components in a Blockchain Solution

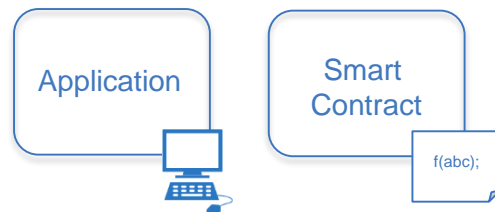
Ledger		A ledger is a channel's chain and current state data which is maintained by each peer on the channel.
Smart Contract		Software running on a ledger, to encode assets and the transaction instructions (business logic) for modifying the assets.
Peer Network		A broader term overarching the entire 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.

# The Blockchain Developer



Blockchain  
Developer

Blockchain developers' primary interests are...



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



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



Peers

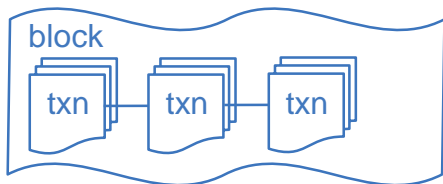
Consensus

Security

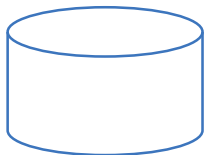
Do Not Distribute or Copy

# How the Developer Interacts with the Ledger

## A ledger often consists of two data structures



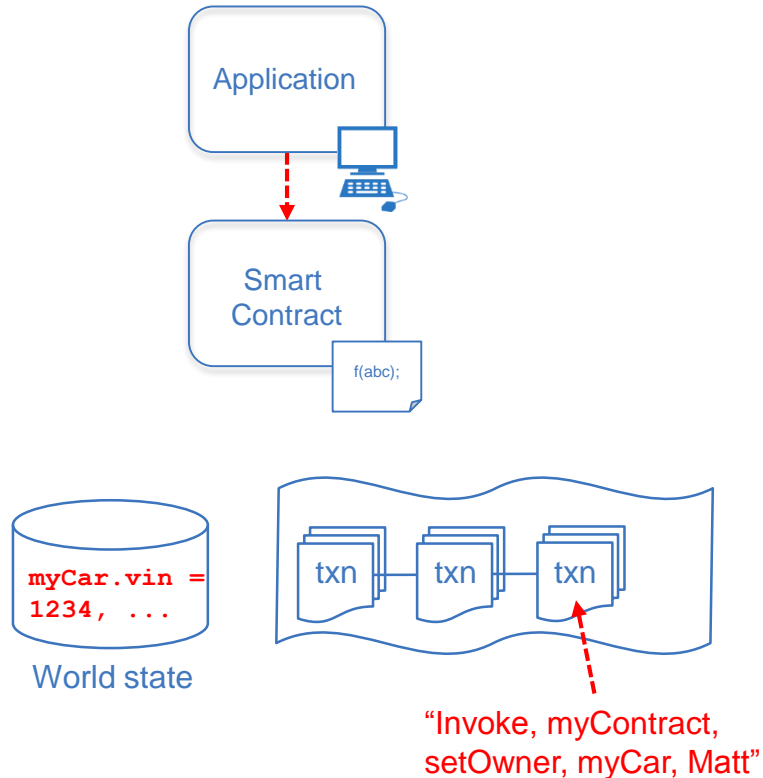
Blockchain



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

# Working with the Ledger: Example of a Change of Ownership Transaction (change car1 owner to Matt)



Transaction input - sent from application

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

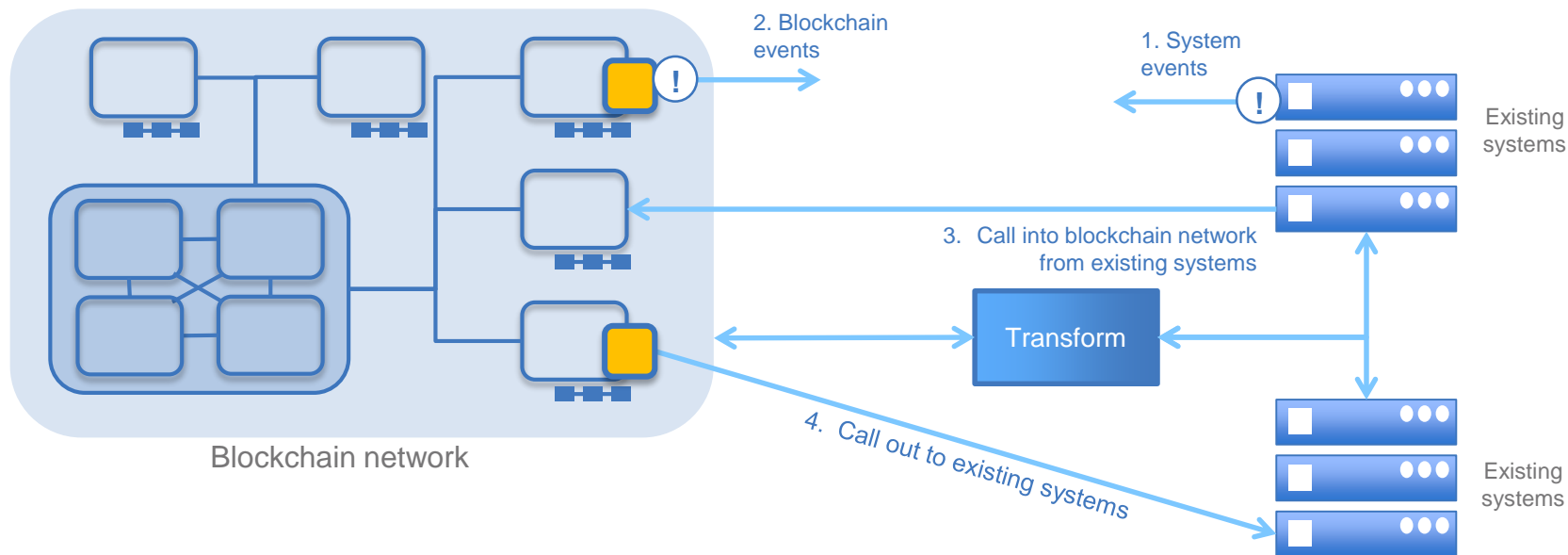
Smart contract implementation

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

World state: new contents

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

# Integrating with Existing Systems





# Blockchain Architecture

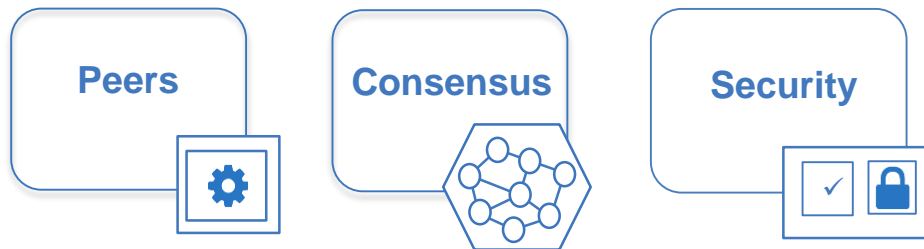


# The Blockchain Administrator (Operator)



Blockchain  
Administrator

Blockchain administrators' primary interests are in the deployment and operation of part of the blockchain:



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

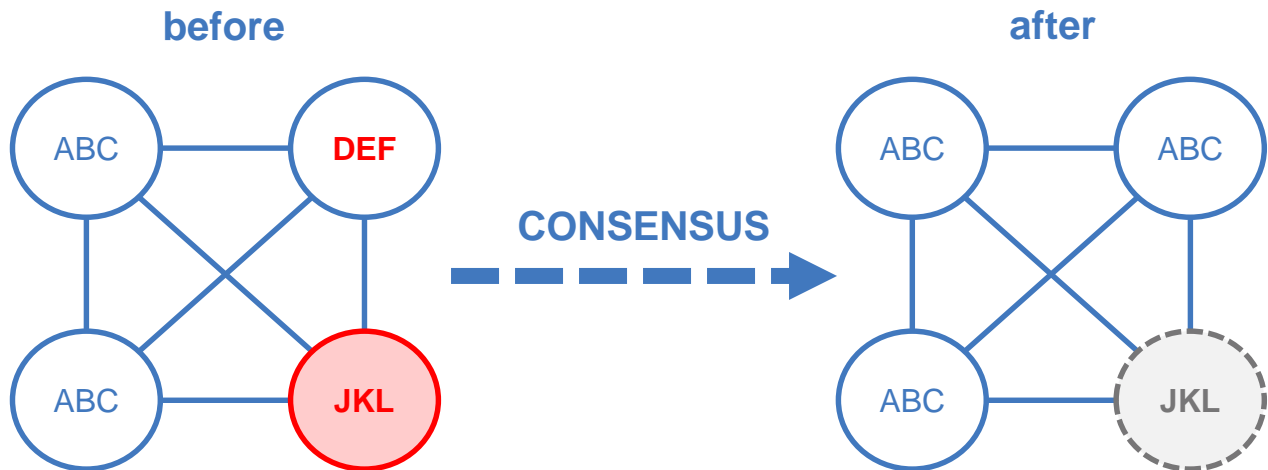


Application code

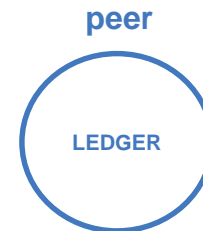
Smart contract code

Events and integration

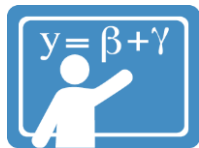
# Consensus: The Process of Maintaining a Consistent Ledger



**Keep all peers up to date.  
Fix any peers in error.  
Ignore all malicious nodes.**



# Some Examples of Consensus Algorithms



**Proof of work**



**Proof of stake**



**Solo**



**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

**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

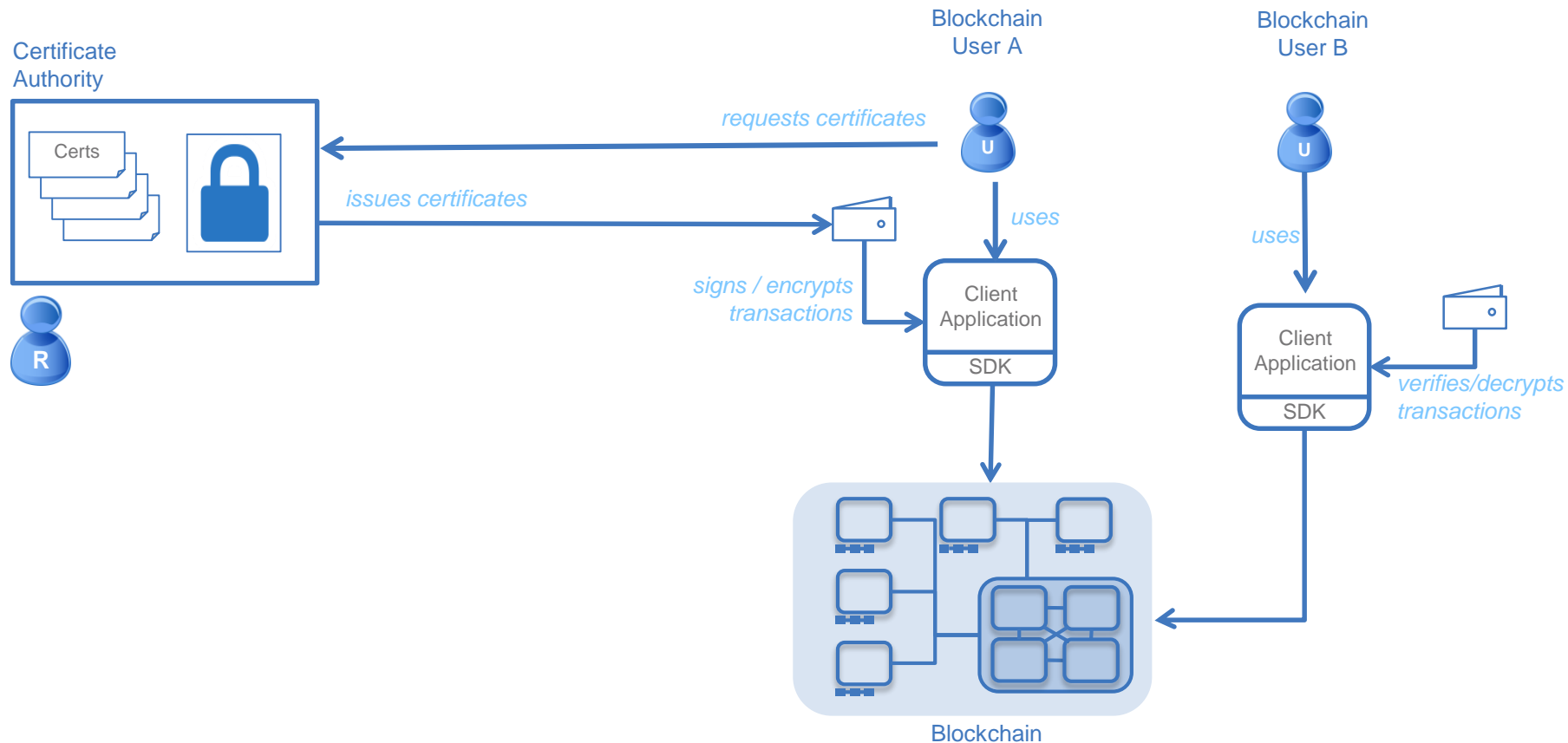
## Private blockchains



- 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

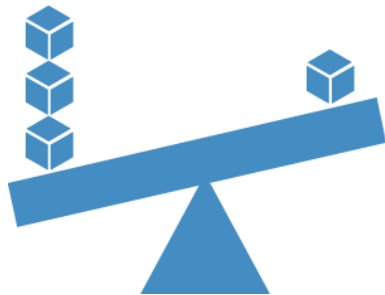
# Certificate Authorities and Blockchain



# Other Nonfunctional Requirements




- Performance
  - The amount of data being shared
  - Number and location of peers
  - Latency and throughput
  - Batching characteristics
- Security
  - Type of data being shared, and with whom
  - How is identity achieved
  - Confidentiality of transaction queries
  - Who verifies (endorses) transactions
- Resiliency
  - Resource failure
  - Malicious activity
  - Non-determinism

**Consider the trade-offs  
between performance,  
security, and resiliency!**

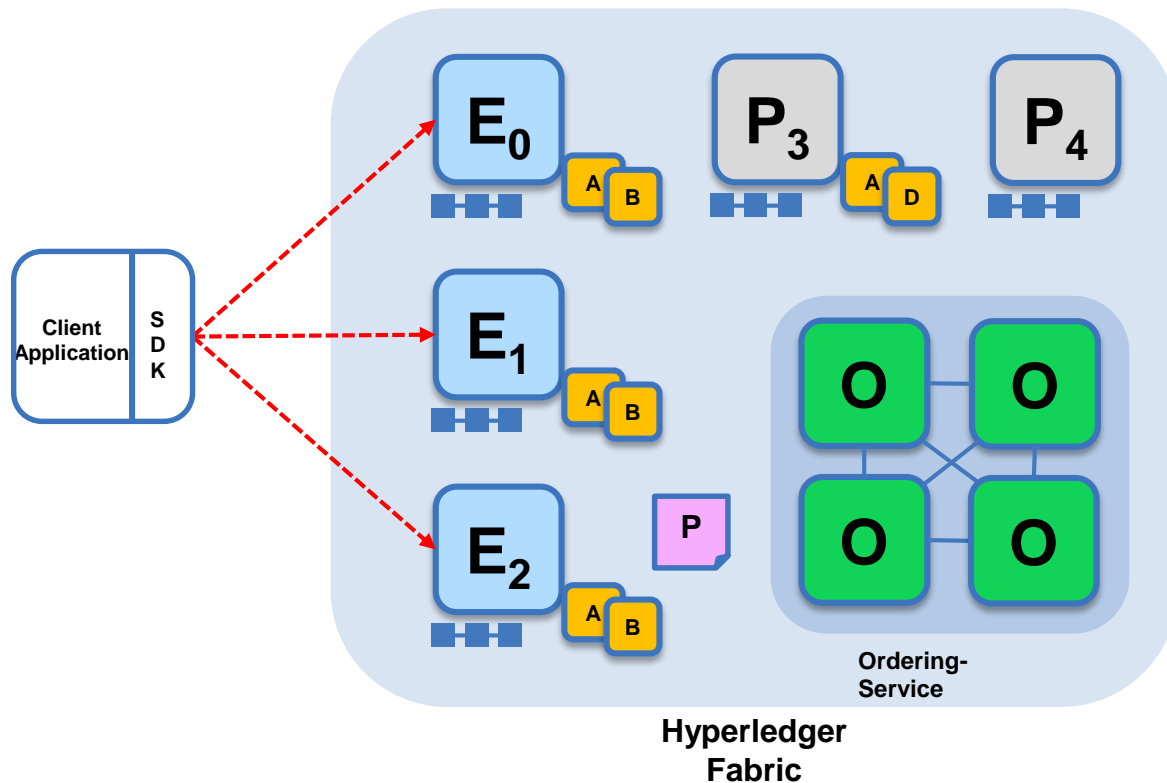




# Nodes and Roles

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



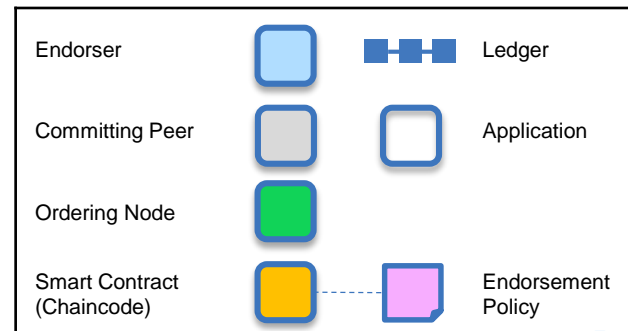
**Application proposes transaction**

**Endorsement policy:**

- “E<sub>0</sub>, E<sub>1</sub> and E<sub>2</sub> must sign”
- (P<sub>3</sub>, P<sub>4</sub> are not part of the policy)

Client application submits a transaction proposal for Smart Contract A. It must target the required peers {E<sub>0</sub>, E<sub>1</sub>, E<sub>2</sub>}.

Key:



# Sample Transaction: Step 2/7 – Execute Proposal

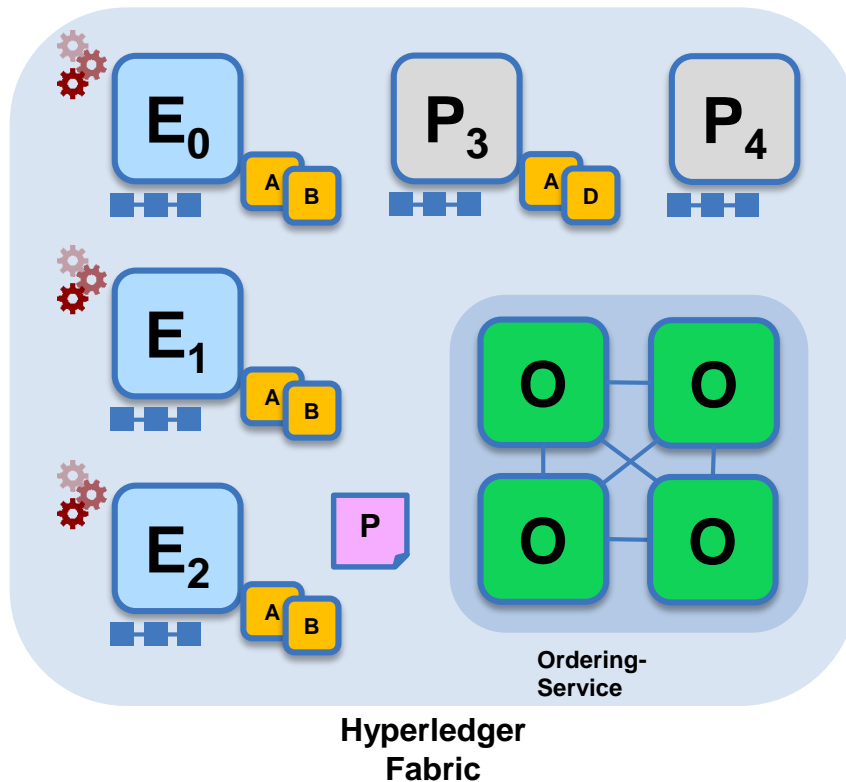
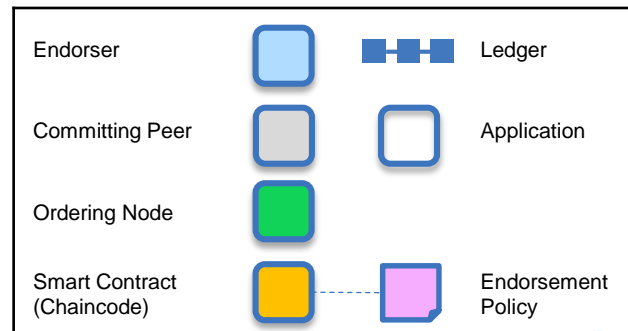
## Endorsers Execute Proposals

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

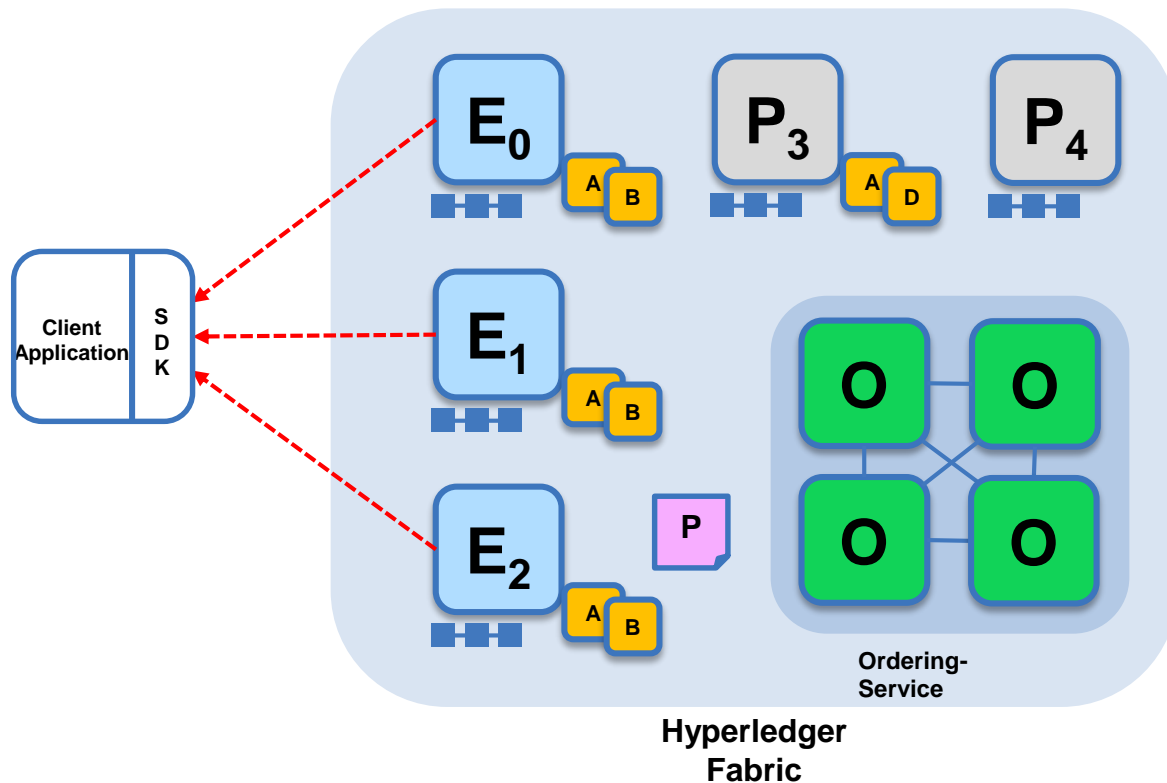
Each execution will capture the set of Read and Written data, called RW sets, which will now flow in the fabric.

Transactions can be signed and encrypted.

Key:



# Sample Transaction: Step 3/7 – Proposal Response



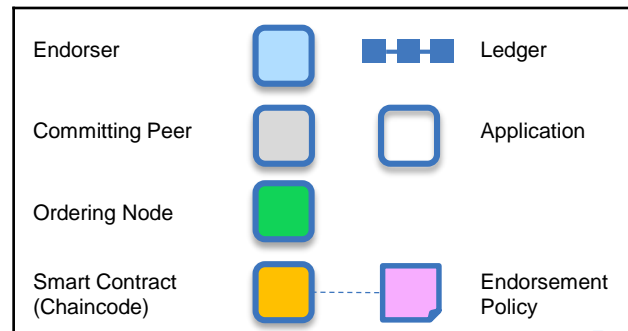
## Application receives responses

RW sets are asynchronously returned to application.

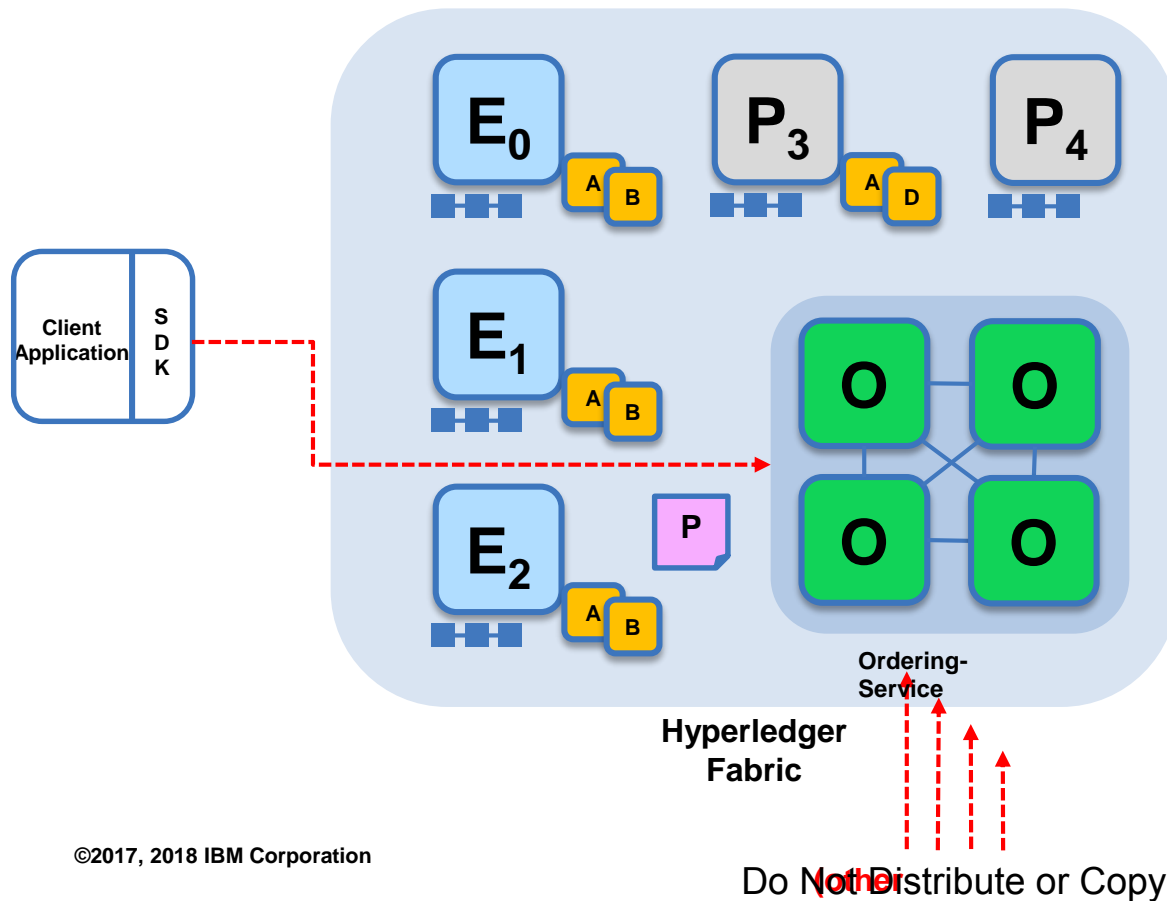
The RW sets are signed by each endorser, and also includes each record version number.

This information will be checked much later in the consensus process.

Key:



# Sample Transaction: Step 4/7 – Order Transaction

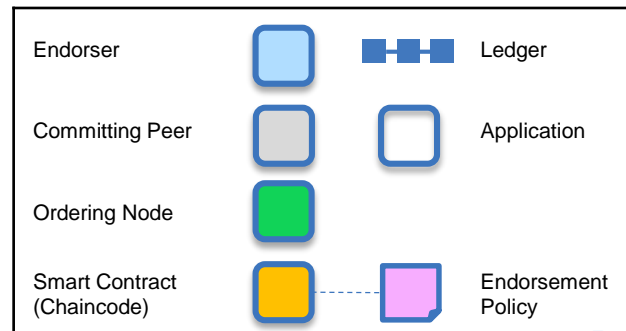


**Application submits responses for ordering**

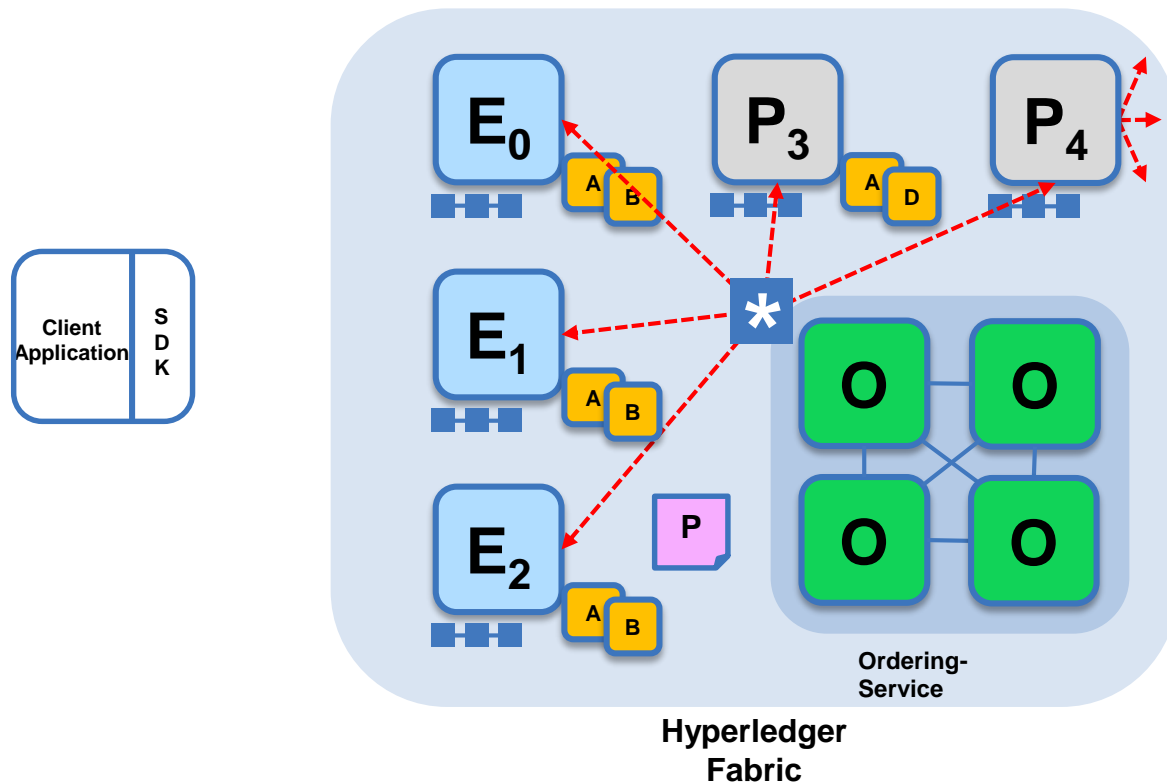
Application submits responses as a transaction to be ordered.

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

Key:



# Sample Transaction: Step 5/7 – Deliver Transaction



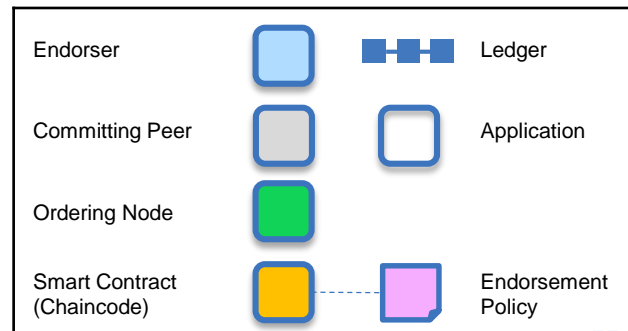
**Orderer delivers to all committing peers**

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:



# Sample Transaction: Step 6/7 – Validate Transaction

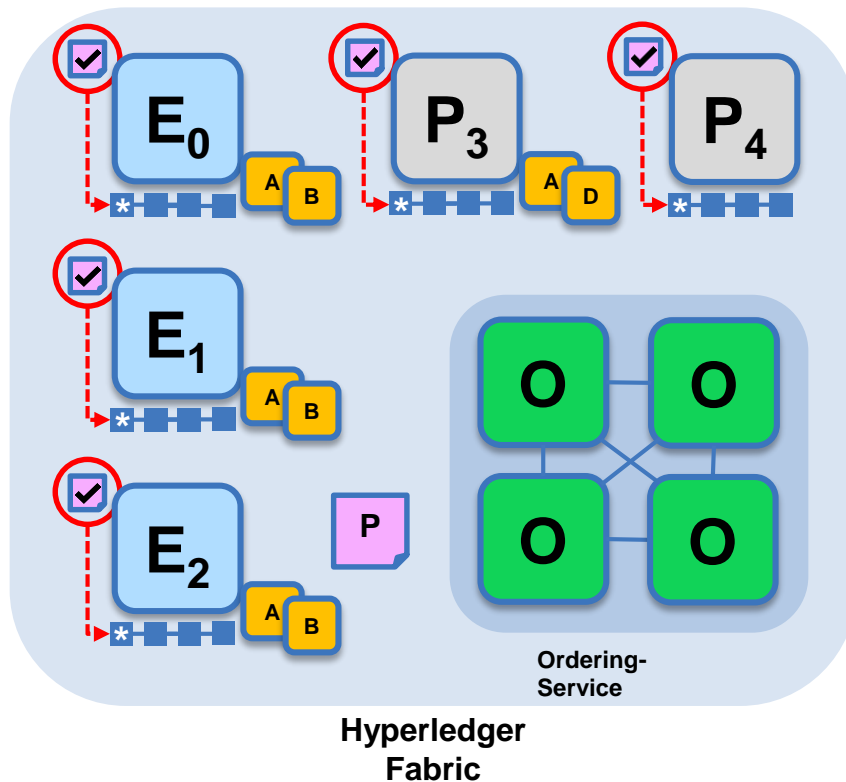
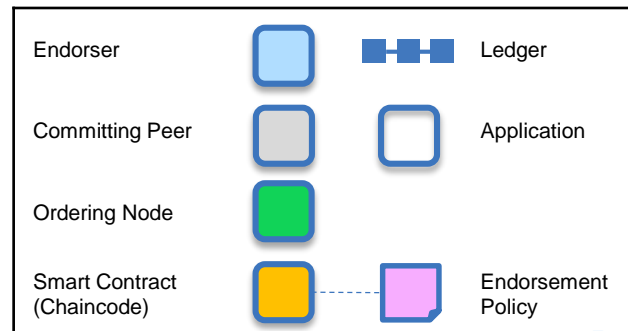
## Committing peers validate transactions

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

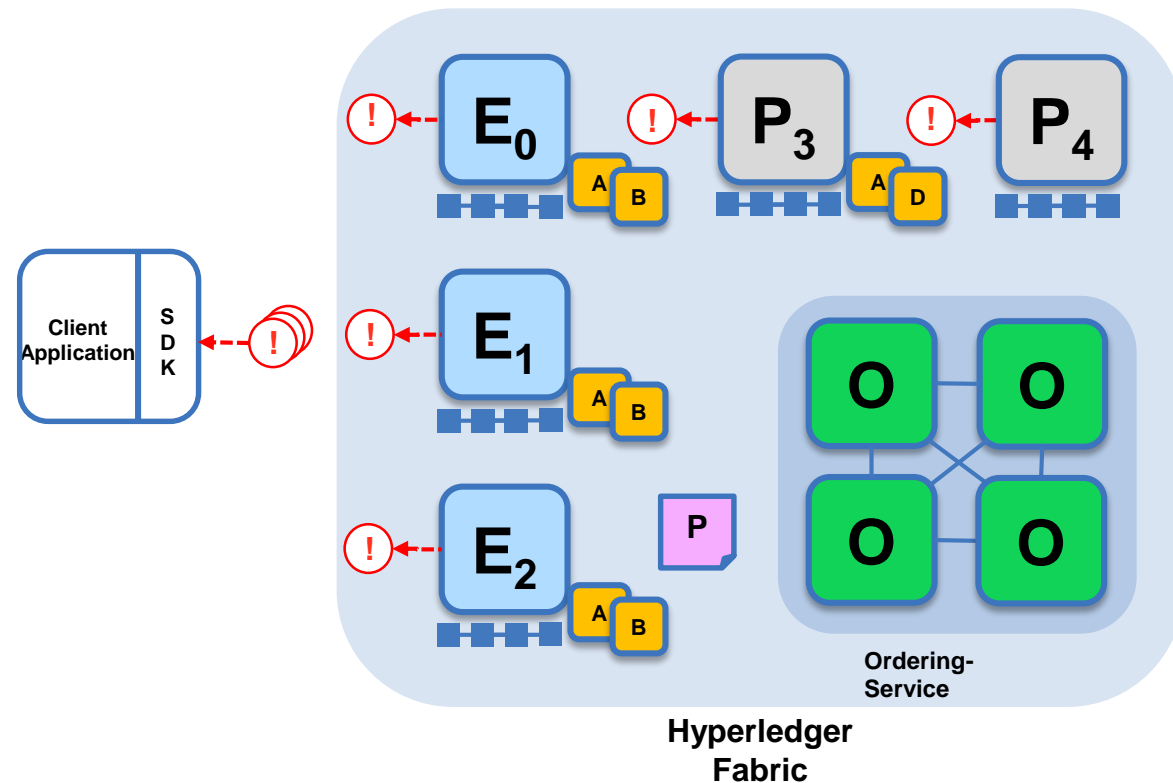
Validated transactions are applied to the world state and retained on the ledger.

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

Key:



## Sample Transaction: Step 7/7 – Notify Transaction

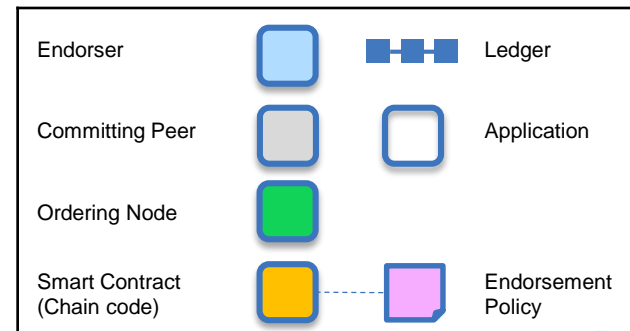


## Committing peers notify applications

**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.**

**Key:**





---

**© Copyright IBM Corporation 2017, 2018**

IBM, the IBM logo and ibm.com are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

This document is current as of the initial date of publication and may be changed by IBM at any time.

The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. This information is based on current IBM product plans and strategy, which are subject to change by IBM without notice. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.