HYPERLEDGER

BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

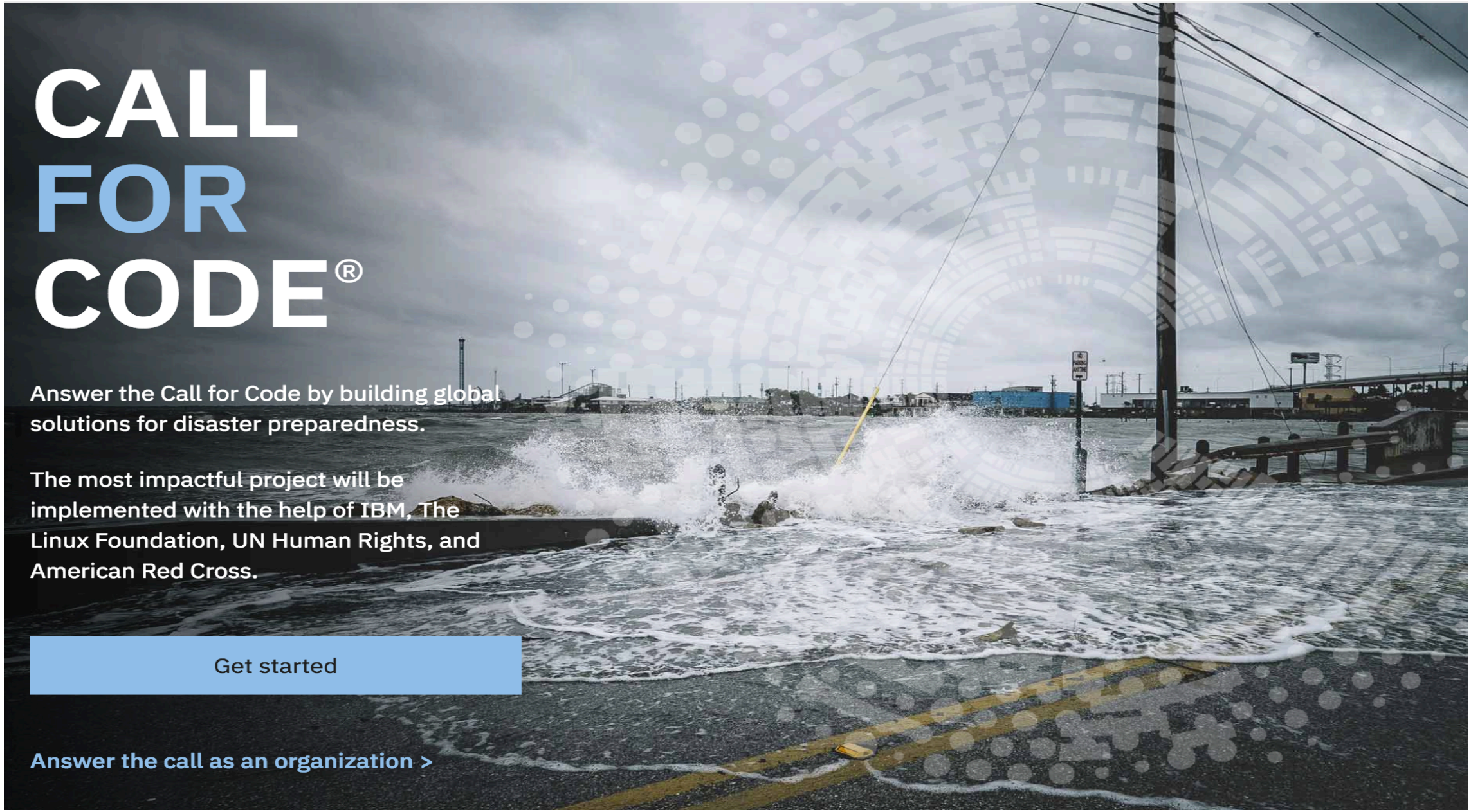
Creating Blockchain applications on Hyperledger Fabric and the IBM Cloud

Lennart Frantzell, IBM Developer Advocate

alf@us.ibm.com

August 2018

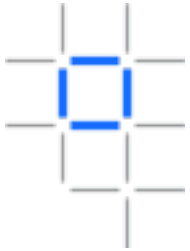
http://ibm.biz/portblock



IBM **Blockchain**

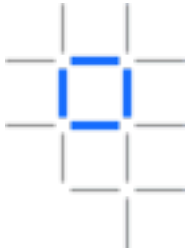
https://callforcode.org/

My background



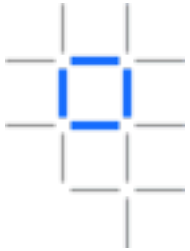
- AI and expert systems
- Object oriented programming
- Productizing IBM Research prototypes
- Cloud computing
- Working with startups
- Healthcare IT
- Blockchain

Agenda



1. IBM and Blockchain
2. What is IBM doing with Blockchain?
3. Introduction to Hyperledger
4. Creating a business network with Hyperledger Composer
5. Creating an enterprise Blockchain with Hyperledger Fabric
6. Hosting Hyperledger Fabric in the Cloud

IBM and Blockchain

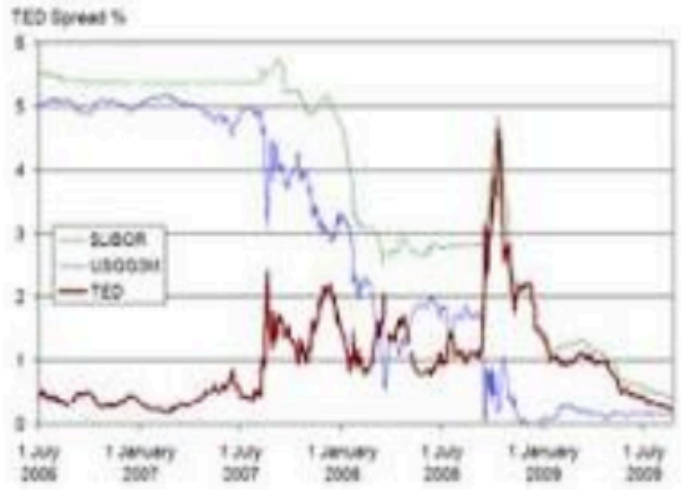


IBM's three most important technologies:

- Watson AI
- Blockchain with a team of 1,500 people
- Internet of Things

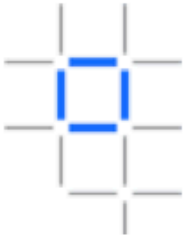
Global Financial Crisis 2007-2008

It began in **2007** with a **crisis** in the subprime mortgage market in the United States, and developed into a full-blown international banking **crisis** with the collapse of the investment bank Lehman Brothers on September 15, **2008**. ... The **crisis** was nonetheless followed by a **global economic** downturn, the Great Recession.



Financial crisis of 2007–2008 - Wikipedia
https://en.wikipedia.org/wiki/Financial_crisis_of_2007–2008

How did it all start?



October 2008. It all started with Satoshi Nakamoto and his paper Bitcoin: A Peer-to-Peer Electronic Cash System which addressed a key problem in electronic commerce:

> *A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution.*
>
> *Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending.*
>
> ***We propose a solution to the double-spending problem using a peer-to-peer network.***

https://bitcoin.org/bitcoin.pdf

Reduce trust and replace it with cryptography and decentralized trust protocols

Harvard Business Review

INTERNATIONAL BUSINESS

The Promise of Blockchain Is a World Without Middlemen

by Vinay Gupta

MARCH 06, 2017

https://hbr.org/2017/03/the-promise-of-blockchain-is-a-world-without-middlemen

12th of July 1789: Two days before the Storming of the Bastille:

Louis XVI: “Is it a revolt”?

Duc de La Rochefoucauld:

“No Sire, it is a revolution”

Why blockchain now?

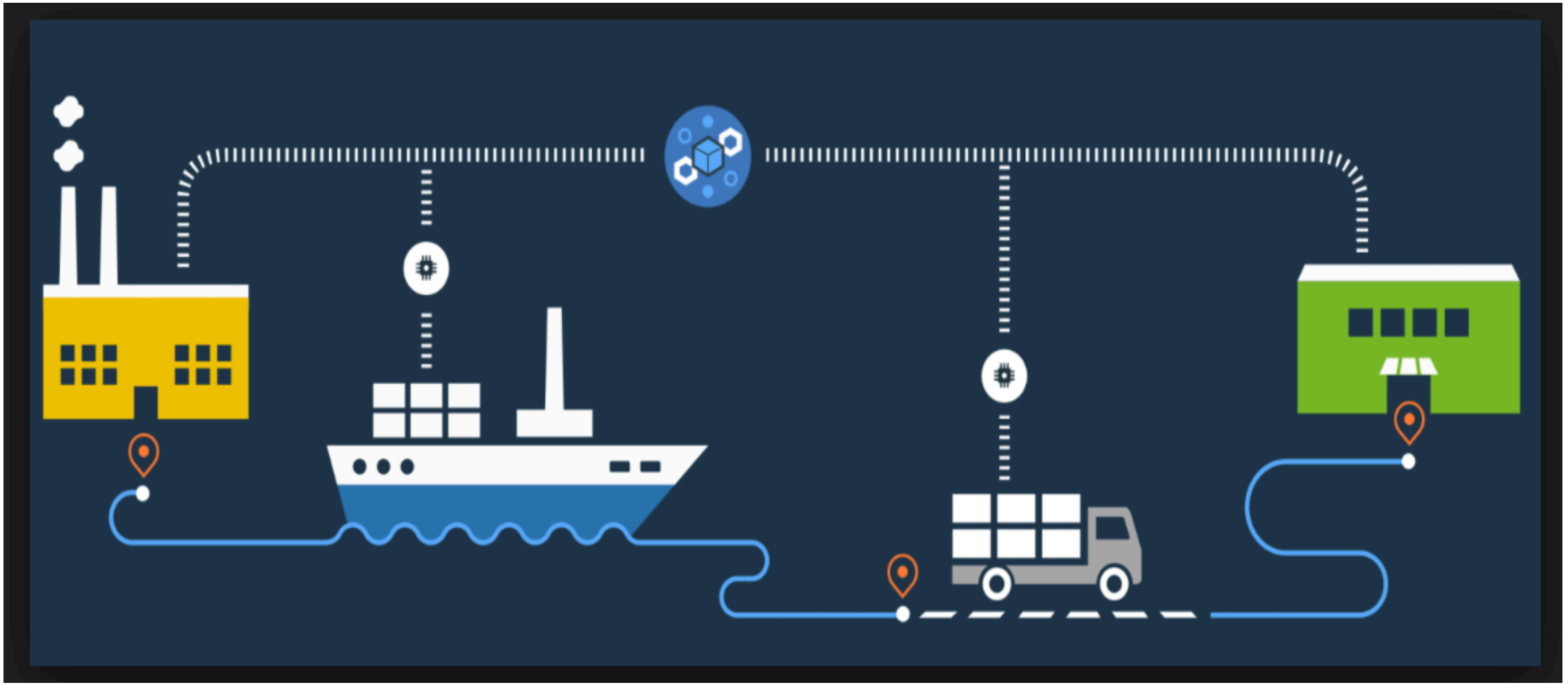
§ Cryptography has been a key technology in the financial world for decades
  – Payment networks, ATM security, smart cards, online banking ...

§ Trust model of (financial) business has not changed
  – Trusted intermediary needed for exchange among non-trusting partners
  – Today cryptography mostly secures point-to-point interactions

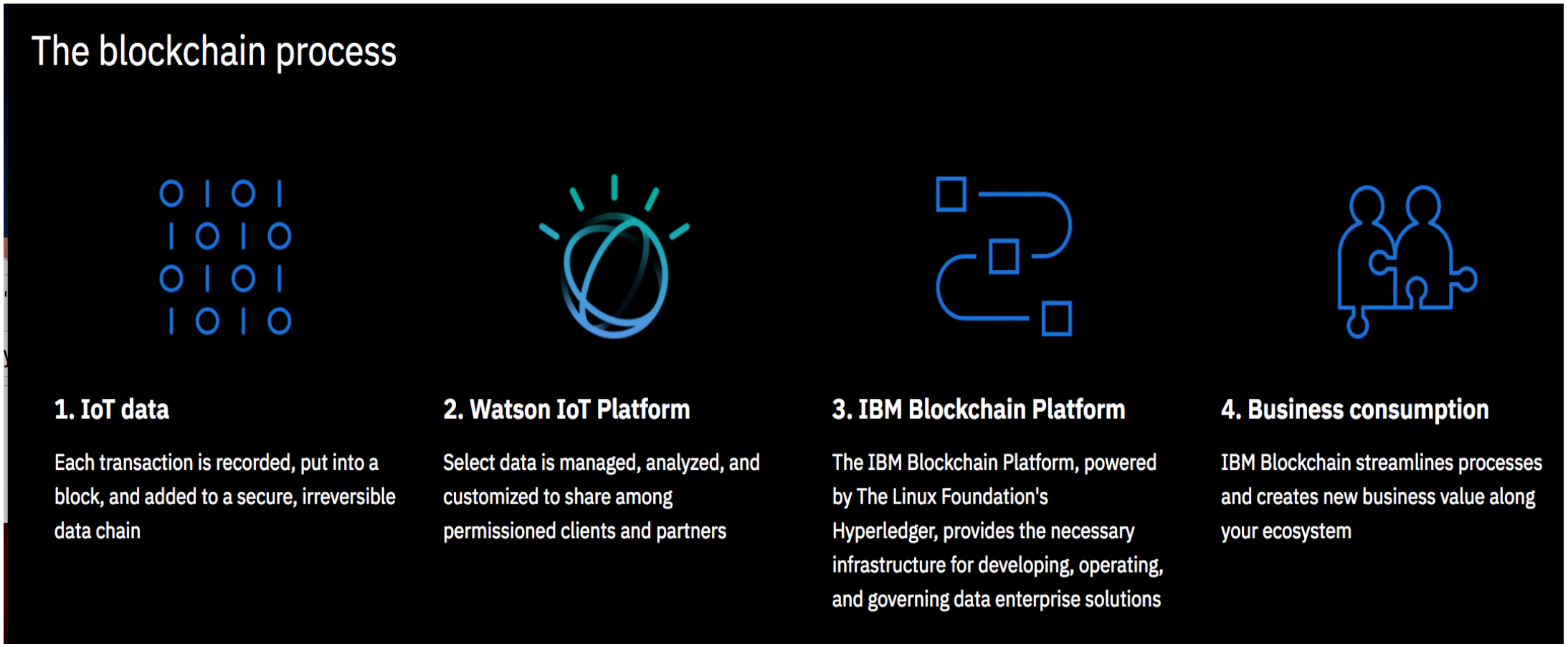
§ Bitcoin started in 2009
  – Embodies only cryptography of 1990s and earlier
  – First prominent use of cryptography for a new trust model (= trust no entity)

§ The promise of Blockchain – Reduce trust and replace it by technology
  – Exploit advanced cryptographic techniques

IoT and Supply Chain/IoT, containers, smart contracts and Blockchain Trust



Blockchain and IoT



IBM **Blockchain**

https://www.ibm.com/internet-of-things/spotlight/blockchain

Spectrum of Blockchains

Permissionless Public	Permissionless Private	Permissioned Public	Permissioned Private
Bitcoin, Ethereum	Public Polls	Land Titles, University Degrees	Business Apps Medical Records

Permissioned vs. Permissionless: Who can write to a Blockchain, i.e. accessibility
Public vs Private: who can read from a Blockchain, i.e. visibility

https://www.youtube.com/watch?v=-O3ouBdG3Xg

IBM has done more than 400 engagements around the globe to develop blockchain applications

IBM and Maersk: Tradelens.com

https://tradelens.com/

everledger

Legitimize Diamonds and Reduce Fraud

What? Provenance.

- Track diamonds across supply chain from mine to retail

How?

- **Shared ledger for storing digital certification with supporting material**

Benefits

1. Protect against the occurrence of fraud, theft, trafficking and black markets
2. Assist in the identification and reduction of synthetic stones being labelled as authentic
3. Increase speed of transparency for cross border transactions for insurance companies, banks and claimants

Food Safety

What? Provenance

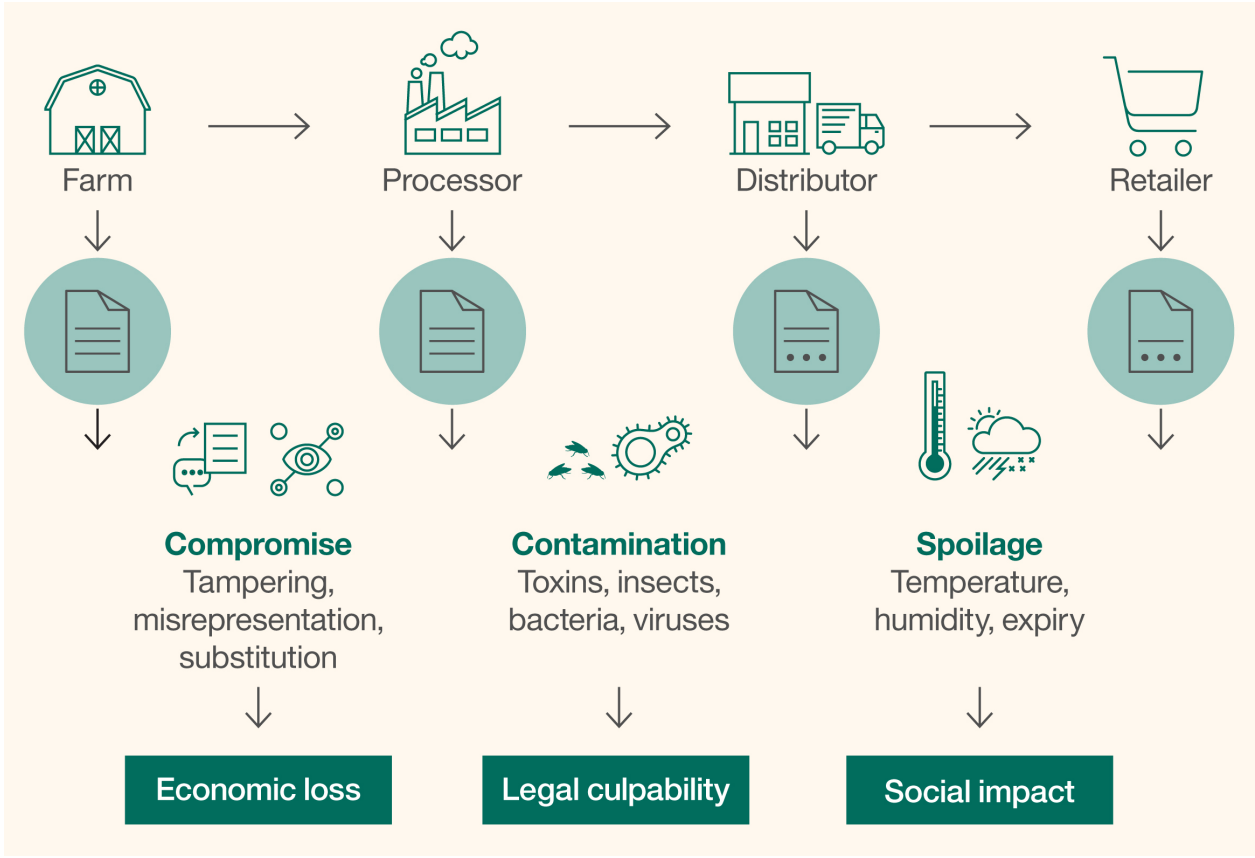
- Provide a trusted source of information and traceability to improve transparency and efficiency across the food network.

How?

- Shared ledger for storing digital compliance documentation, test results and audit certificates

Benefits

- Reduce impact of food recalls through instant access to end-to-end traceability data to verify history in the food network and supply chain.
- Help to address the 1 in 10 people sickened and 400,000 fatalities WW which occur every year from food-born illnesses.



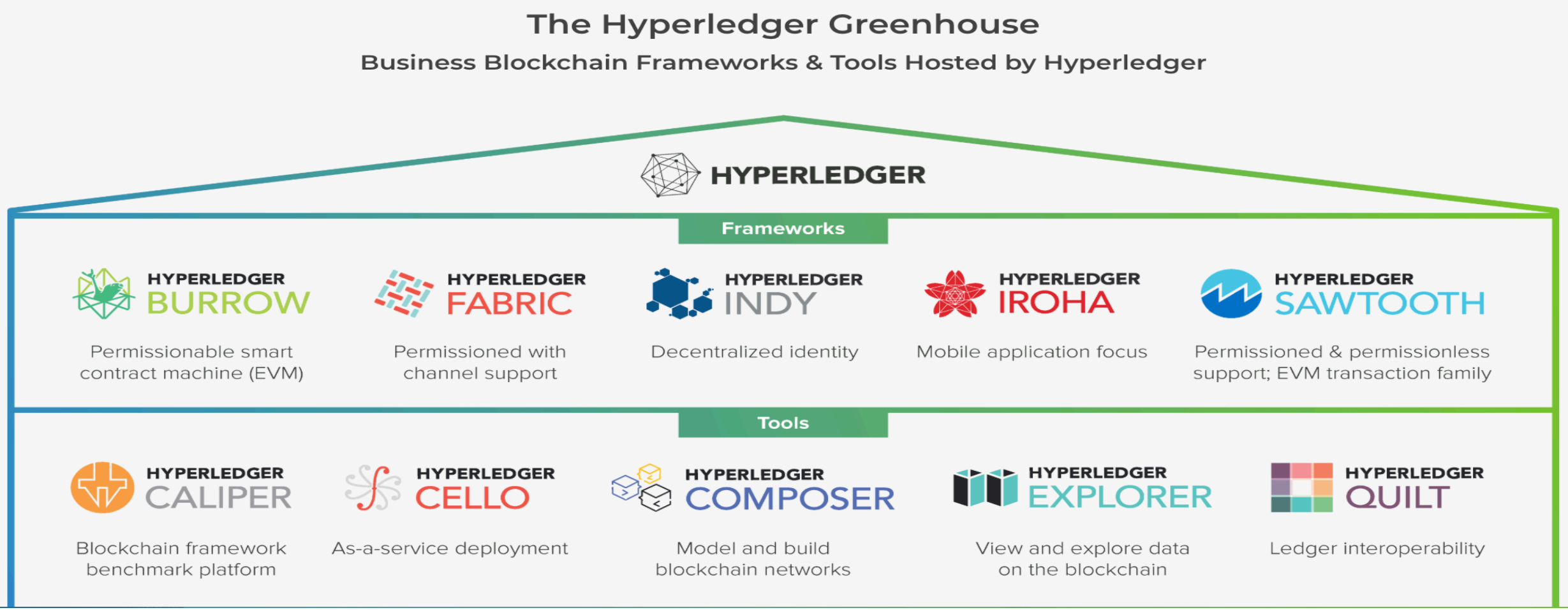
Introducing Hyperledger

Open source collaborative effort to advance **cross-industry blockchain technologies**

Hosted by **The Linux Foundation**, fastest-growing project in LF history

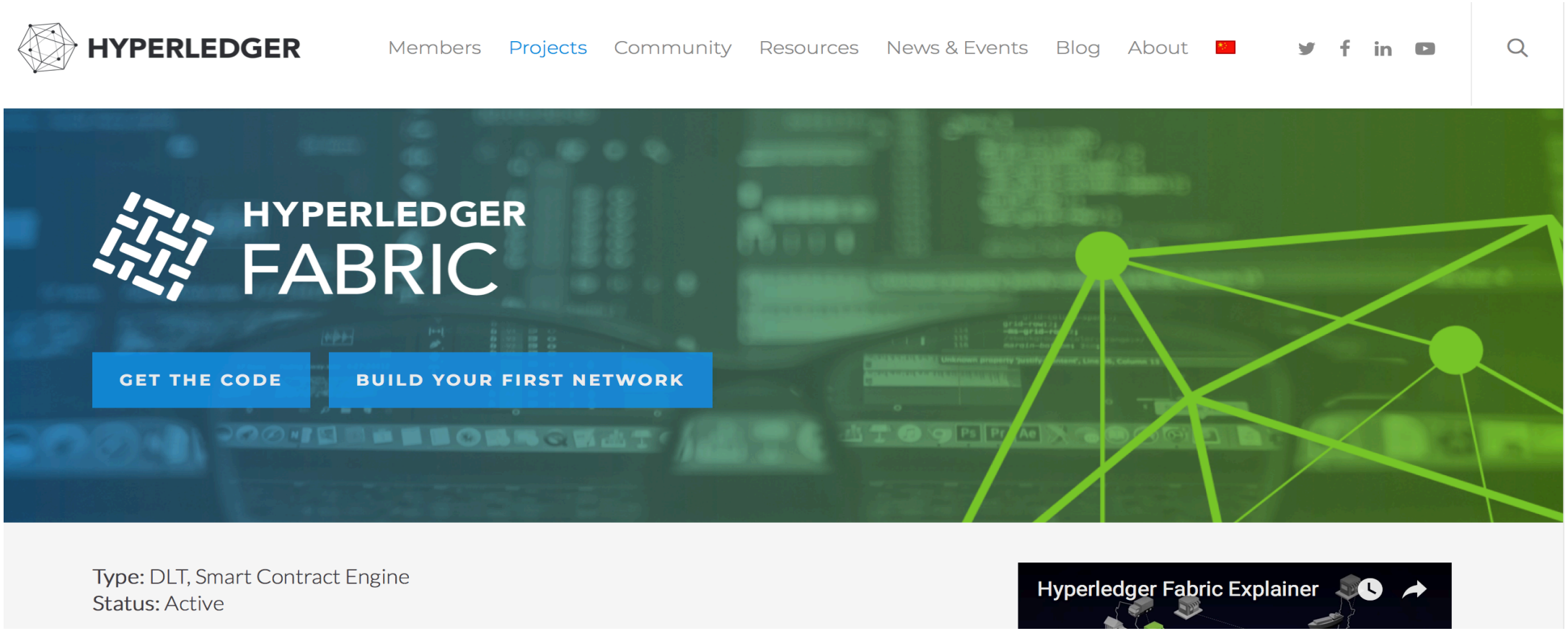
Global collaboration spanning **finance, banking, IoT, supply chain, healthcare, manufacturing, govt, technology** and more.

Which Blockchain do we want to use?



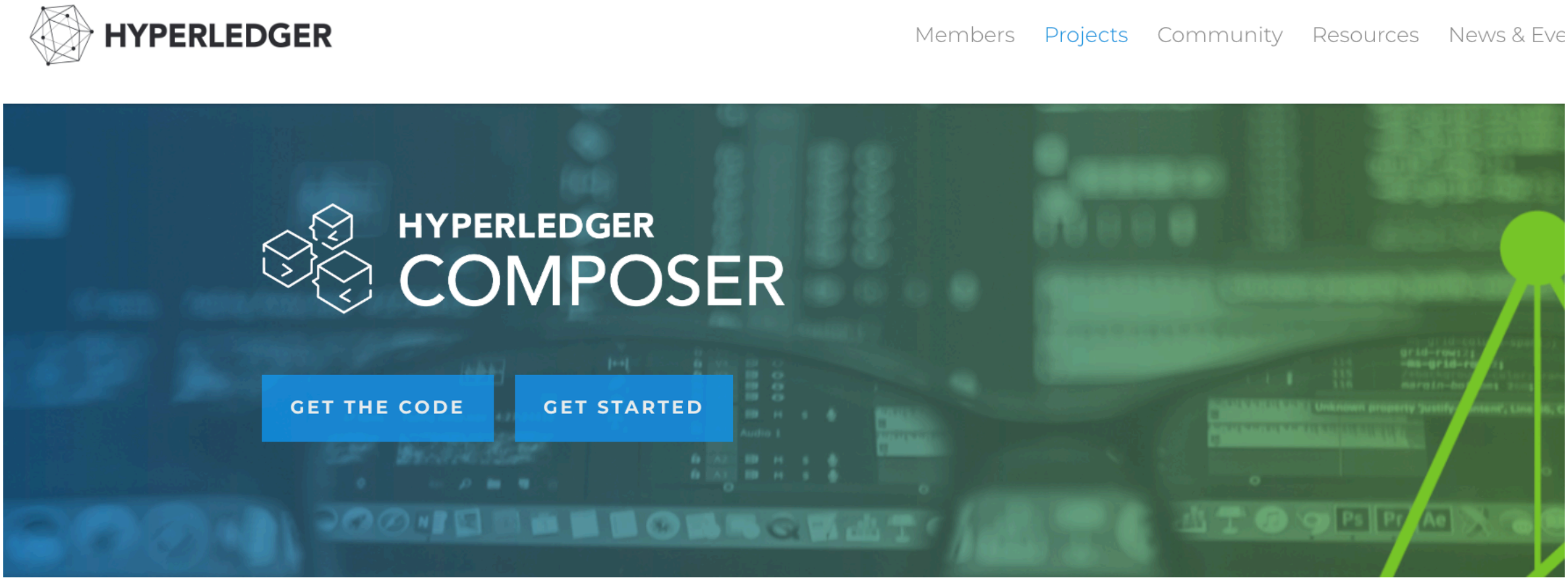
https://www.hyperledger.org/

Hyperledger Fabric

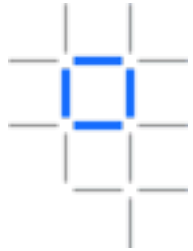


https://www.hyperledger.org/projects/fabric

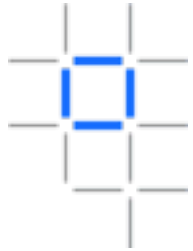
Hyperledger Composer



https://www.hyperledger.org/projects/composer

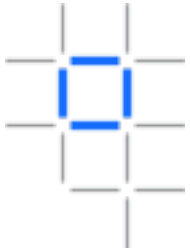


Creating an application in Hyperledger Fabric



Step 1 Use Cases

What makes a good Blockchain Use Case?



- A Business problem that cannot easily be solved with existing techniques
- An identifiable business network
  - With a shared ledger
  - With Participants, Assets and Transactions
- A need for decentralized trust protocols
  - Consensus, Immutability, Finality and Provenance

Blockchain Use Cases

Supply chain, asset registration, identity services, fraud prevention and compliance

IBM Blockchain government point of view

PDF See the infographic (50.6KB)

Payment and digital currency

IBM's universal blockchain payments solution

PDF See the infographic (1.4MB)

Payment and digital currency

IBM announces major blockchain solution to speed up global payments

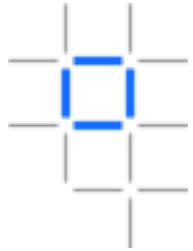
→ Read the press release

European trade finance network: we.trade

Spreading opportunity to small and medium sized businesses in traditionally underserved markets.

Watch the video (02:48)

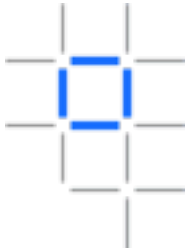
https://www.ibm.com/blockchain/use-cases/



Step 2 Architecture

Opportunities and limitations

Hyperledger Fabric has three components



1. **Membership Services** – which manages digital certificates and access,
2. **Ordering service** – which keeps the peers in alignment,
3. and **peers** – which hold **the ledger**.
    1. A peer is also where **chaincode**, also known as smart contracts, are executed.

    Membership and ordering services align well to being centralized.

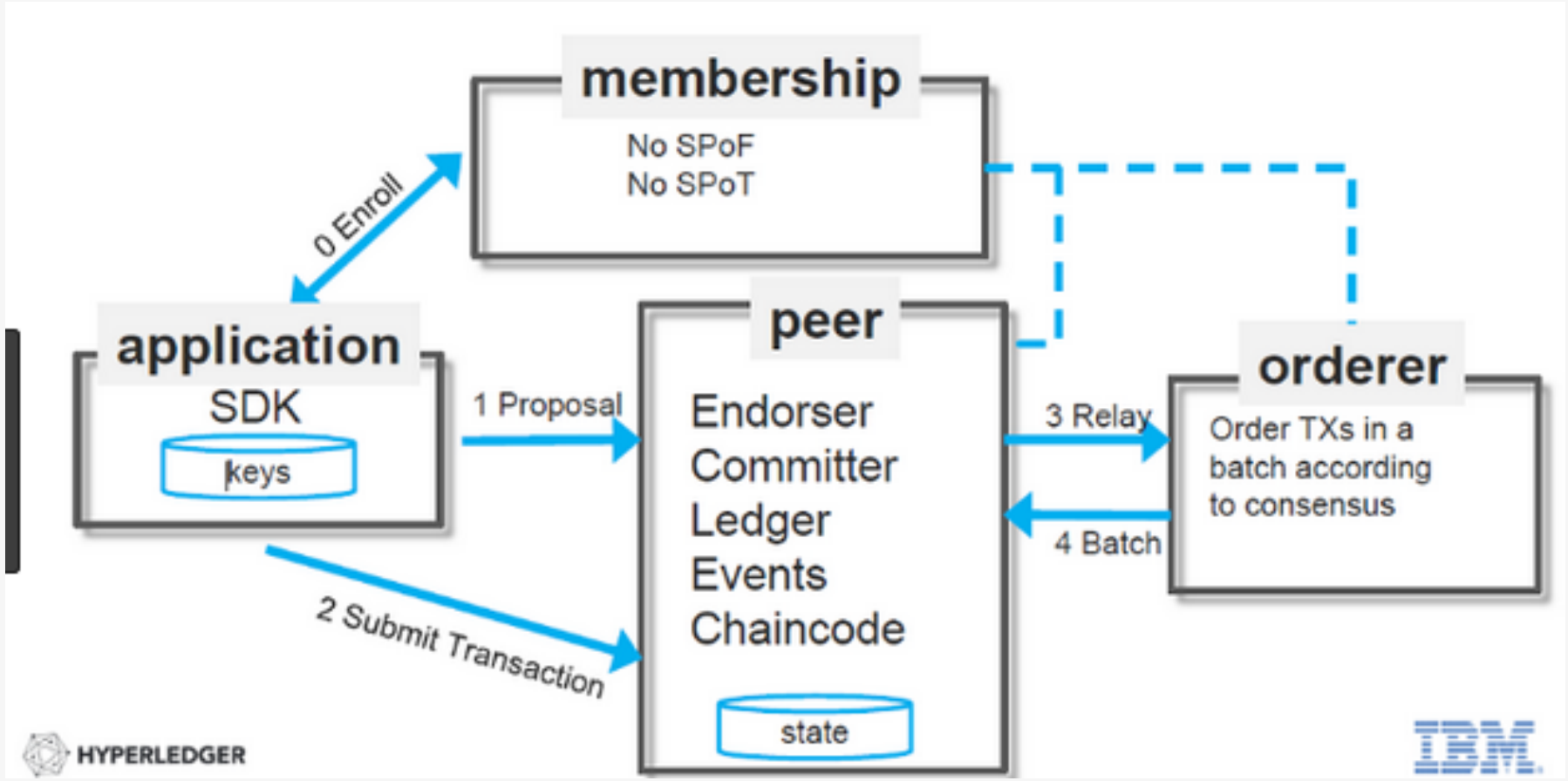
Peers, on the other hand, can be distributed and placed anywhere you desire in the world.

Each member of the Hyperledger network should have their own peer.

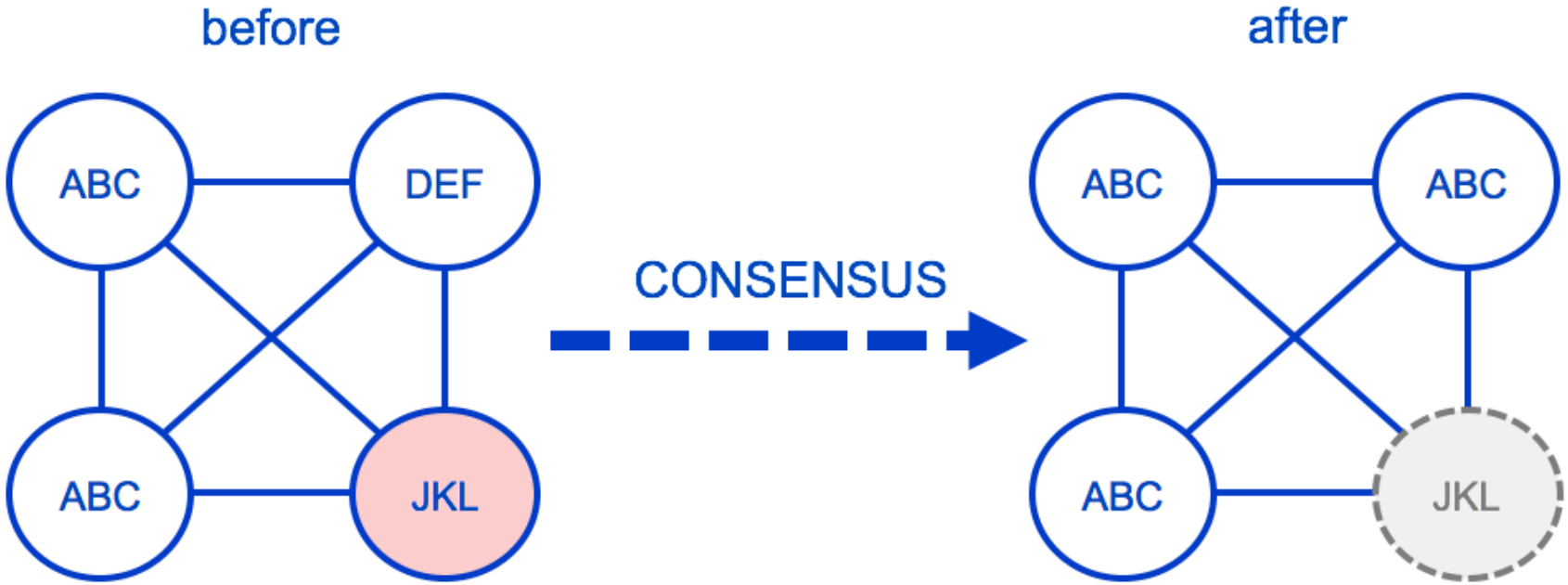
The **channels** (databases) that a peer subscribes to, and transacts against, determines the member's role and capabilities in the blockchain network.

https://www.ibm.com/blogs/systems/blockchain-how-should-you-organize-your-peers/

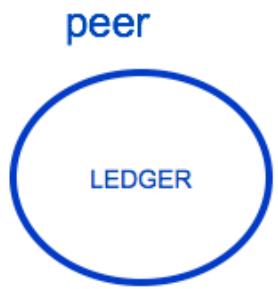
Hyperledger architecture



Consensus: The process of maintaining a consistent ledger

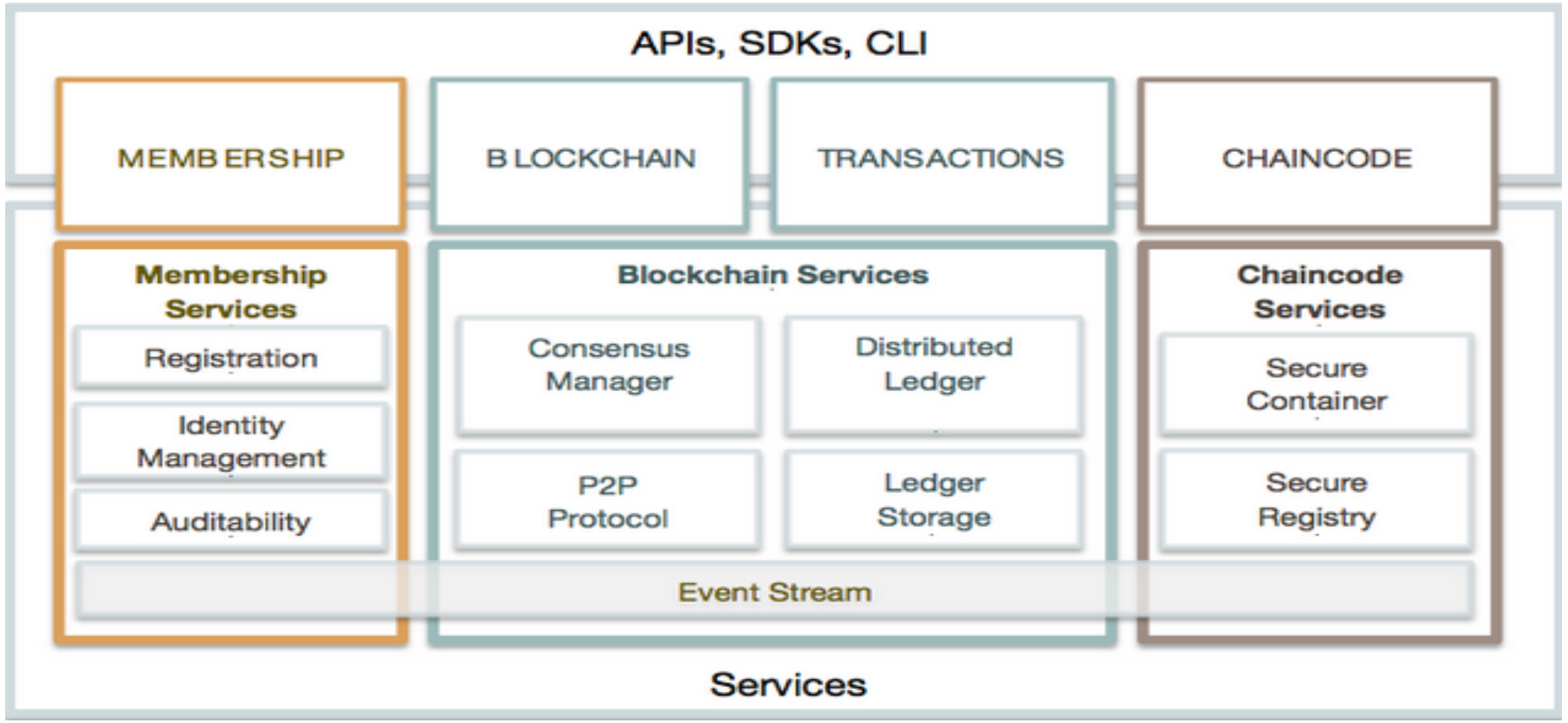


It's important to make all chaincode (transactions in Composer) deterministic, so they can be simulated



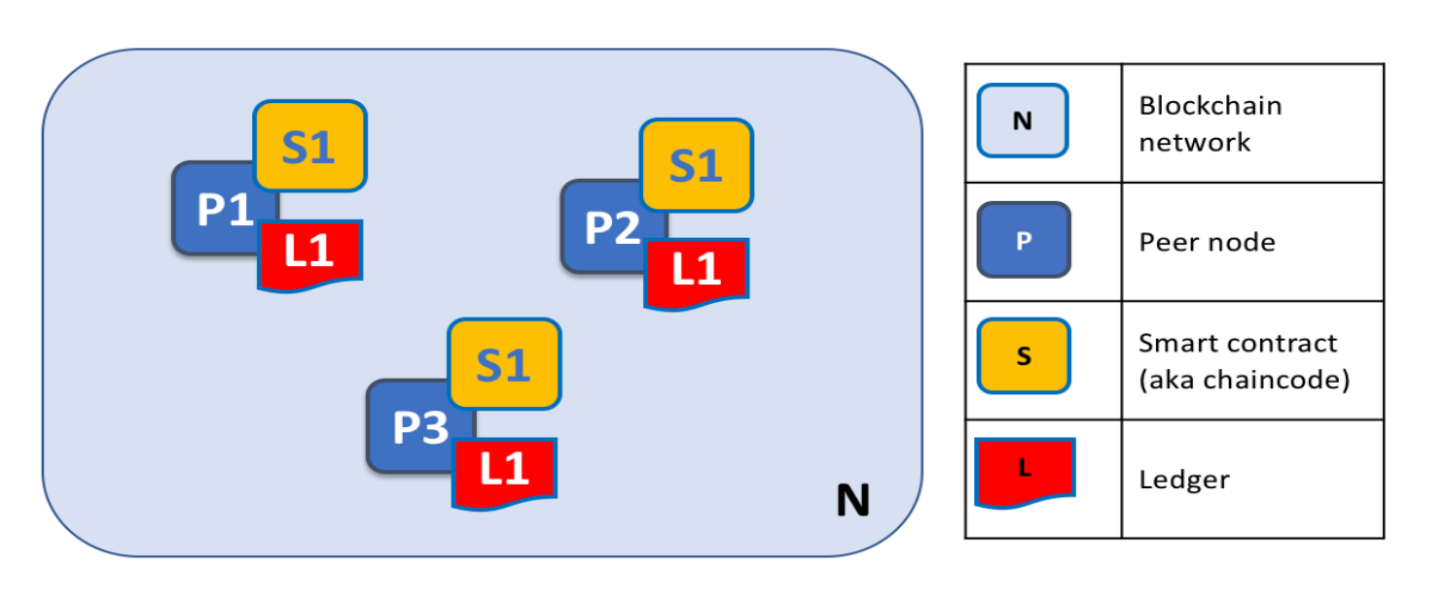
Keep all peers up-to-date
Fix any peers in error
Ignoring all malicious nodes

Hyperledger Fabric architecture



- Hyperledger Fabric allows components, such as consensus and membership services, to be **plug-and-play**.
- Hyperledger Fabric leverages **container technology** (Docker) to host **smart contracts called "chaincode"** that comprise the application logic of the system.
- Chaincode is like Stored Procedures in the traditional database world

Hyperledger Fabric components

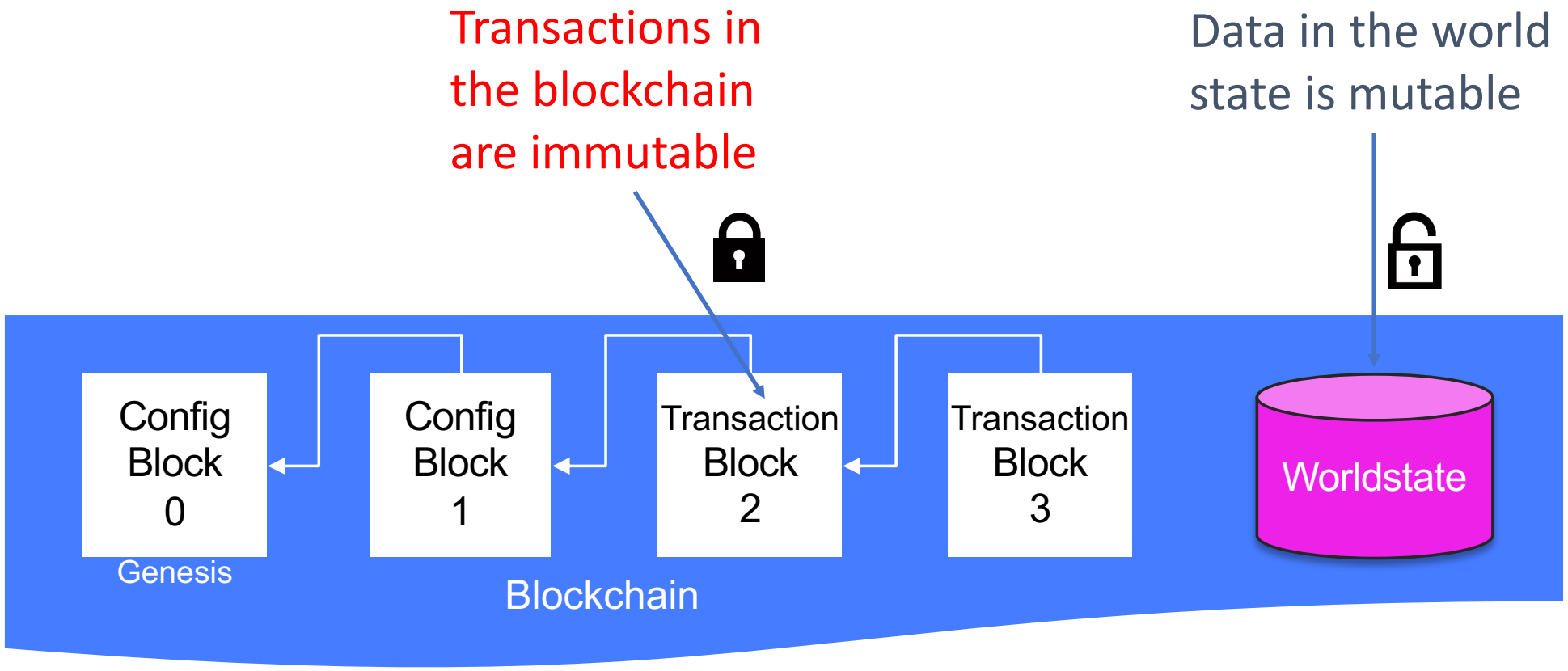


A blockchain network is comprised of peer nodes, each of which can hold copies of ledgers and copies of smart contracts. In this example, the network N consists of peers P1, P2 and P3, each of which maintain their own instance of the distributed ledger L1. P1, P2 and P3 use the same chaincode, S1, to access their copy of that distributed ledger.

https://hyperledger-fabric.readthedocs.io/en/release-1.2/peers/peers.html

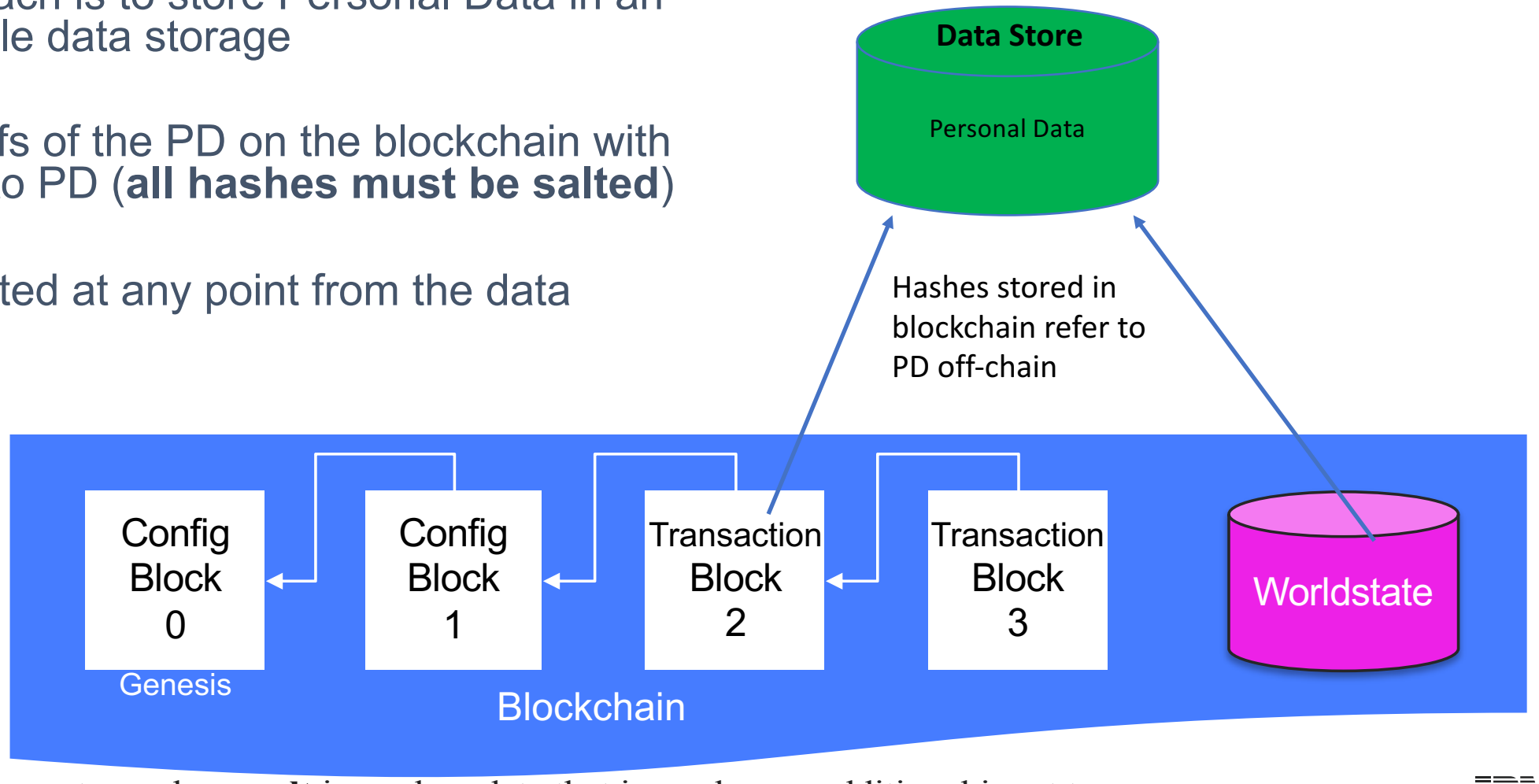
The “Right to Erasure” and blockchain

Enables an individual to request the deletion or removal of their personal data

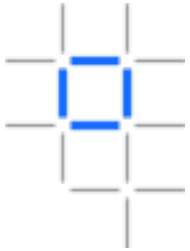


General Data Protection Regulation (GDPR) Solution – Store data off-chain

- **What**
  - The only approach is to store Personal Data in an off-chain mutable data storage
- **How**
  - Store only proofs of the PD on the blockchain with hashes linking to PD (**all hashes must be salted**)
- **Why**
  - PD can be deleted at any point from the data store(s)

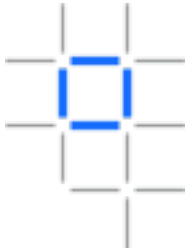


In cryptography, **a salt** is random data that is used as an additional input to a one-way function that "**hashes**" data, a password or passphrase



Step 3 Planning our app

Planning the Peers



Architecturally, the fabric is extremely flexible, but in practice consider two recommendations.

- **First, you only need one or two peers**. Per a given channel, peers are identical. They execute the same transactions and receive the same code updates in the exact same order as all other peers on that channel. **There is no reason to have lots of peers**.
  - **Get two peers if you need the availability** – which I would recommend.
- **Secondly, the front-end systems that hold the Hyperledger SDKs (those systems that provide RESTful interfaces) should be aligned to a member's peers.**
  - Which means you want your front-end systems talking to your peers and you want other member's front-end systems talking to their peers.
  - **The idea of an SDK talking to all of the peers for consensus purposes makes little to no sense when they will all return the same results.**
  - **One SDK only needs to talk to one peer or two, for high availability,** to get the optimal performance and functional capabilities of the fabric.

https://www.ibm.com/blogs/systems/blockchain-how-should-you-organize-your-peers/

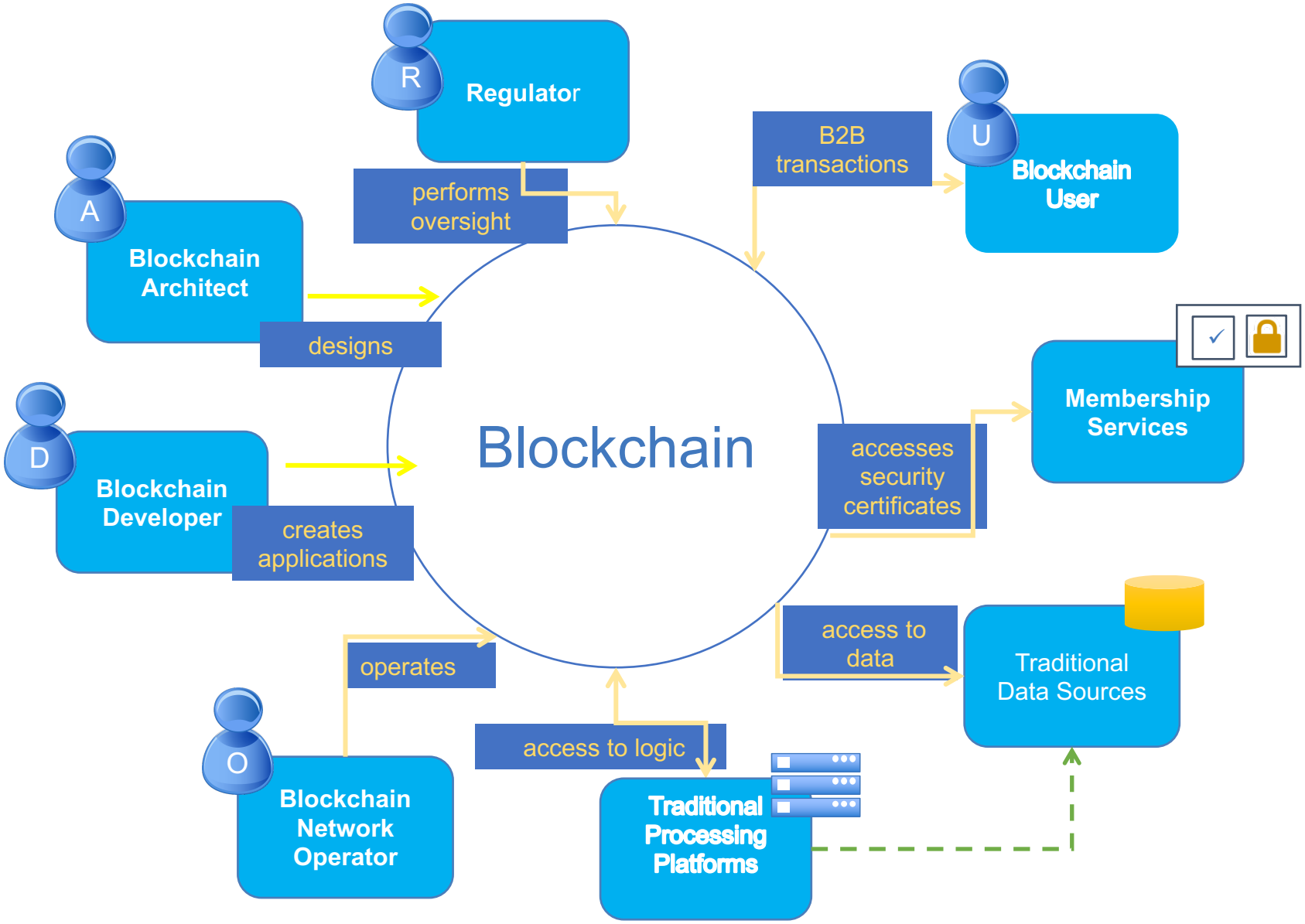
Hyperledger Fabric Performance

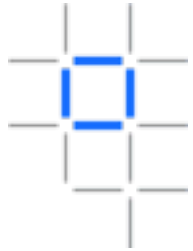
The performance numbers obtained show that Hyperledger Fabric deployed in a single cloud data center achieves an end-to-end throughput

3,500 transactions per second with latency less than one second.

https://www.ibm.com/blogs/research/2018/02/architecture-hyperledger-fabric/

Actors in a Blockchain solution



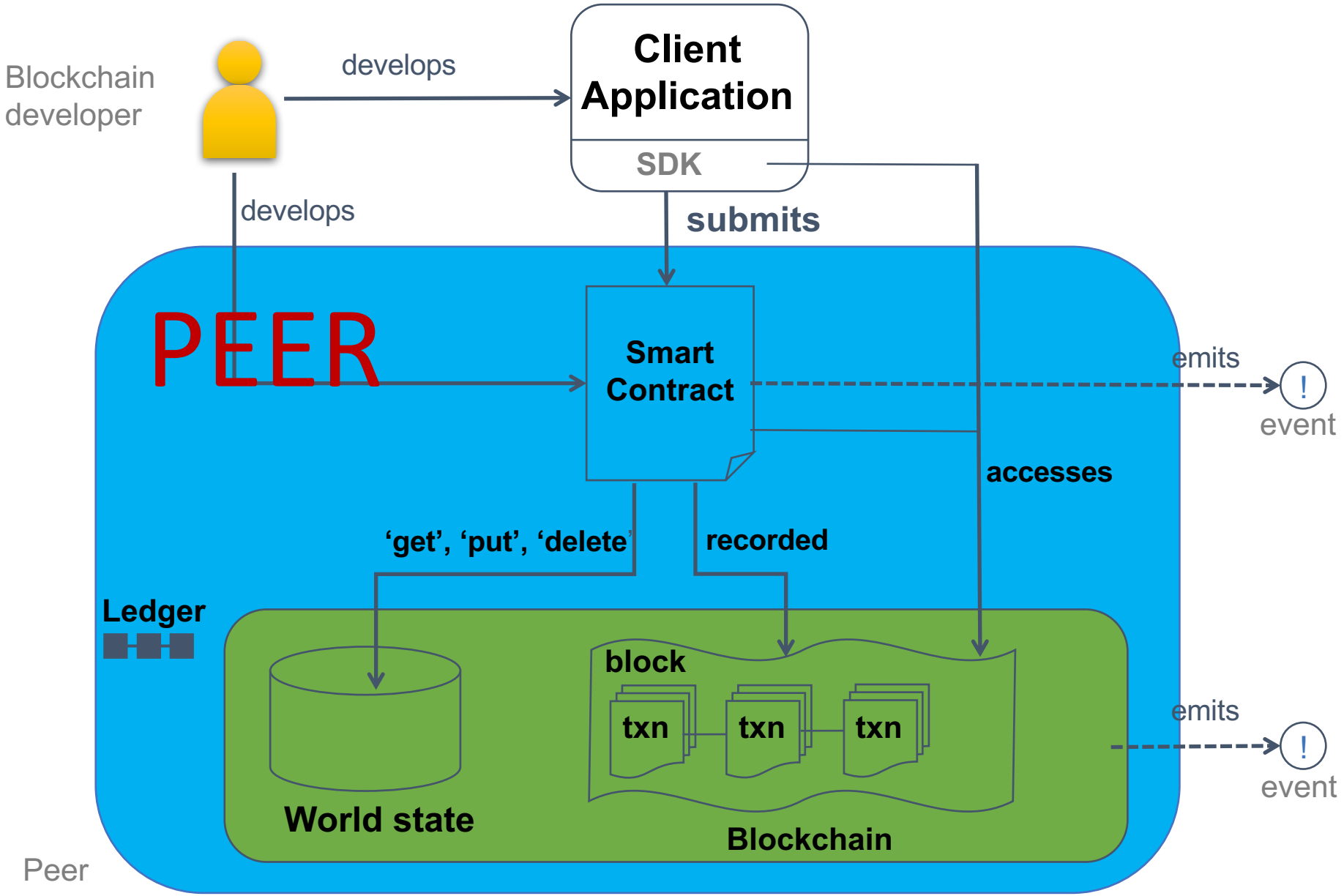


Step 4: Programming the ledger with Chaincode alias Smart Contracts

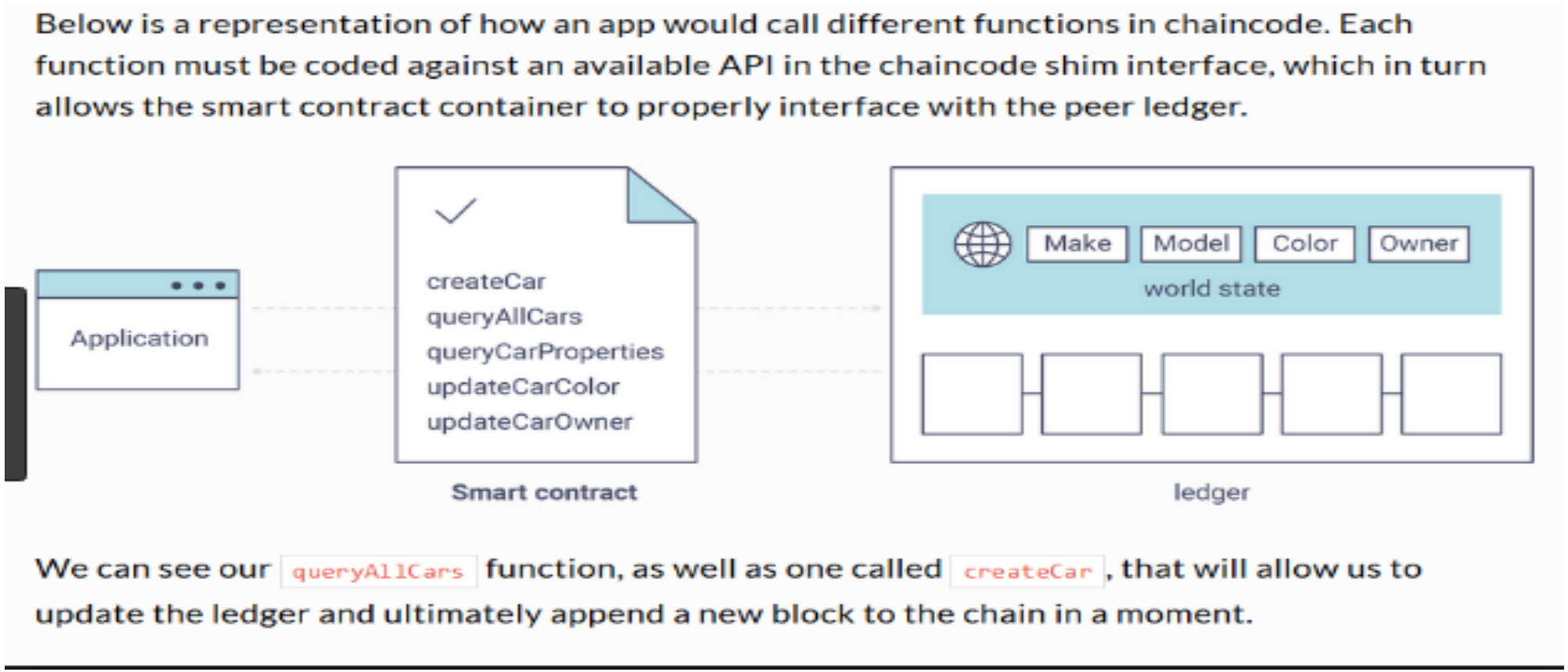
Chaincode and Smart Contracts

- **Chaincode is a program, written in Go, node.js, and eventually in other programming languages such as Java, that implements a prescribed interface. Chaincode runs in a secured Docker container isolated from the endorsing peer process.**
- **Similar to smart Contracts and Stored Procedures**

Programming the ledger



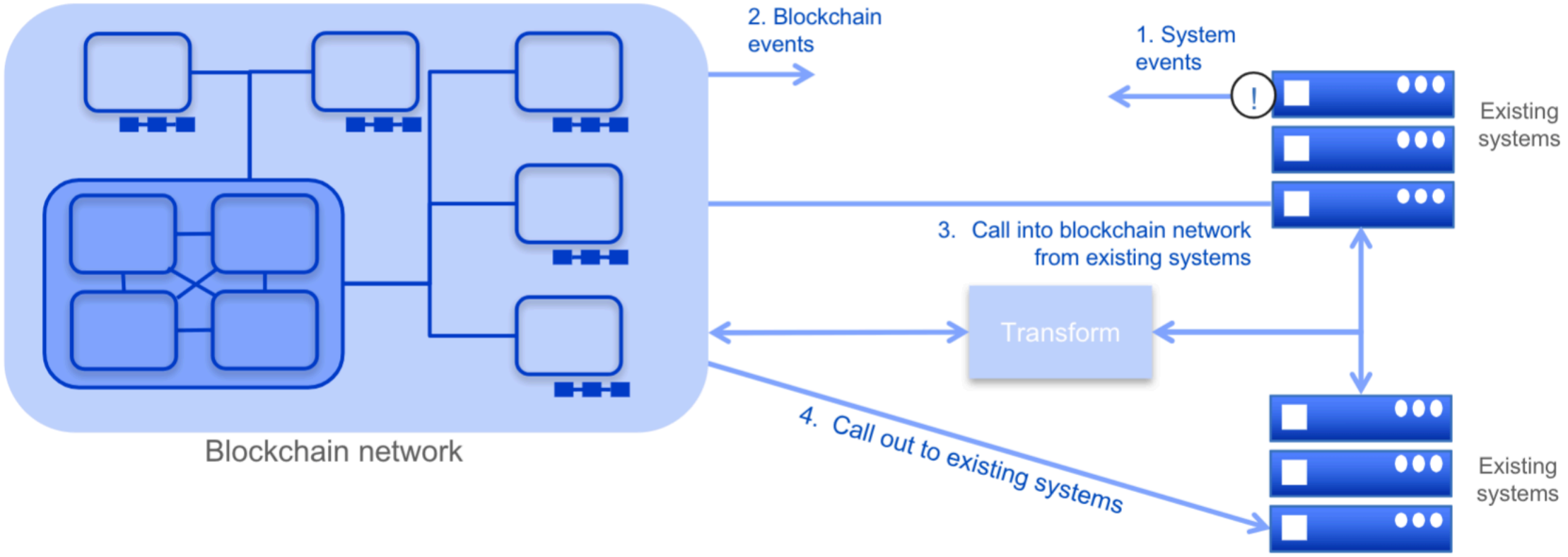
Chaincode, aka Smart Contracts



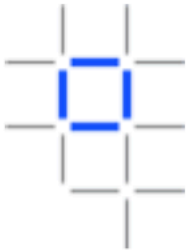
https://hyperledger-fabric.readthedocs.io/en/release-1.2/chaincode.html

Like Stored Procedures in databases

Off-chain, Integrating with existing systems - possibilities



Non-determinism in blockchain



- Blockchain is a distributed processing system
  - Smart contracts are run multiple times and in multiple places
  - As we will see, smart contracts need to run deterministically in order for consensus to work
    - Particularly when updating the world state
- It's particularly difficult to achieve determinism with off-chain processing
  - Implement services that are guaranteed to be consistent for a given transaction, or
  - Detect duplicates for a transaction in the blockchain, middleware or external system

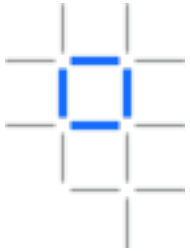
random()

getExchangeRate()

getDateTime()

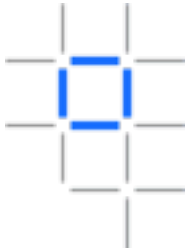
getTemperature()

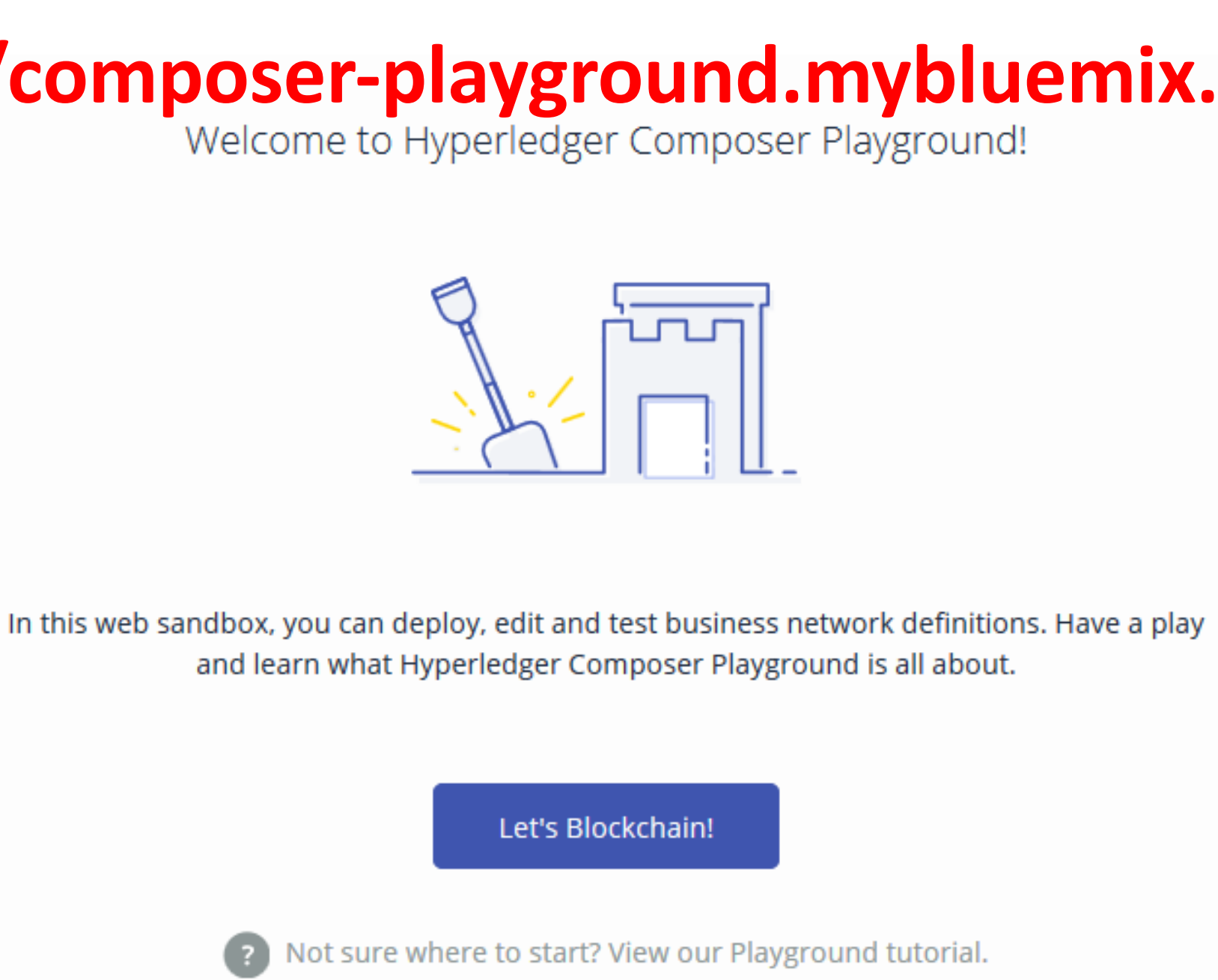
incrementValue inExternalSystem(…)



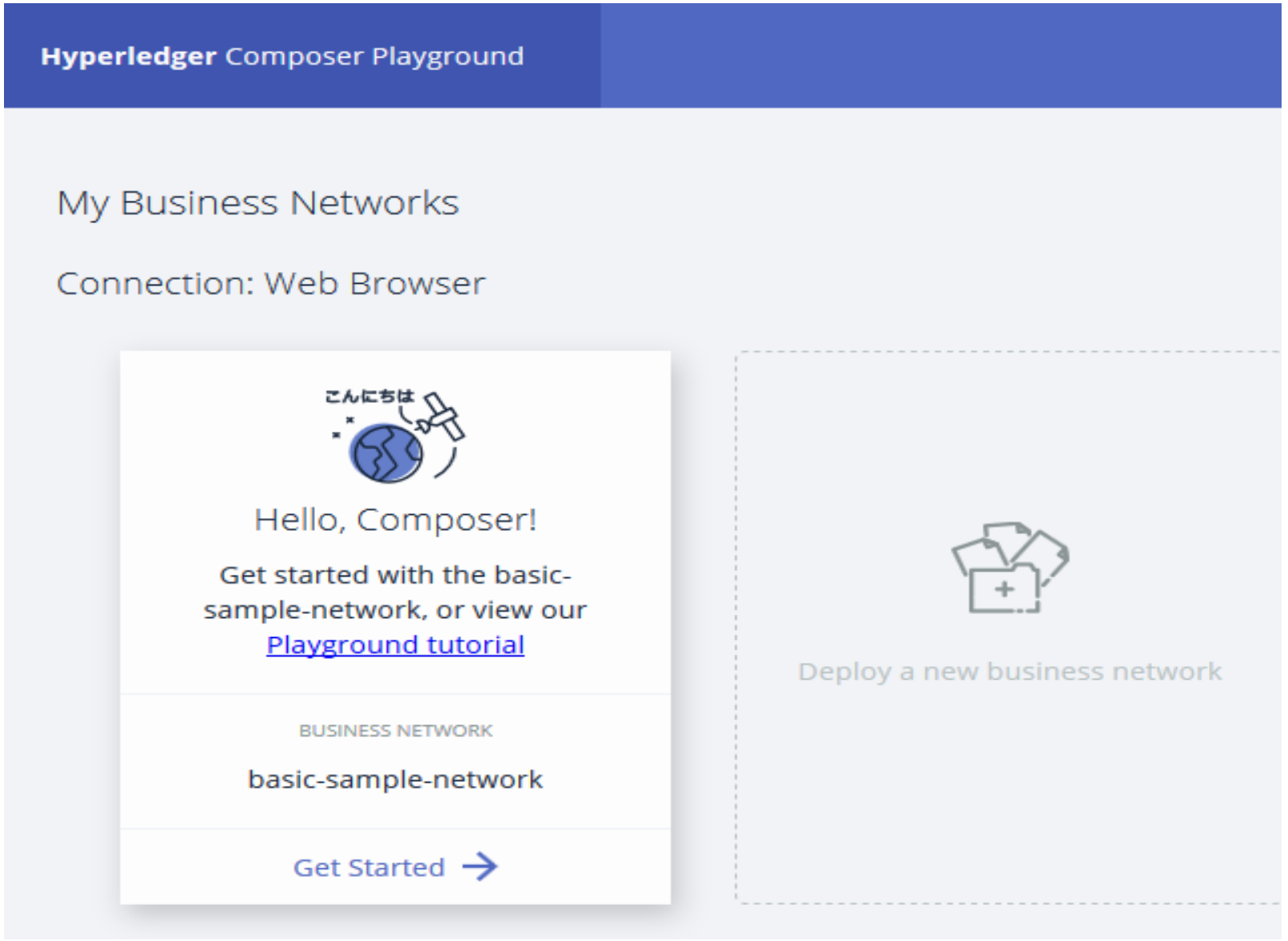
Step 4.1: Programming with Hyperledger Composer

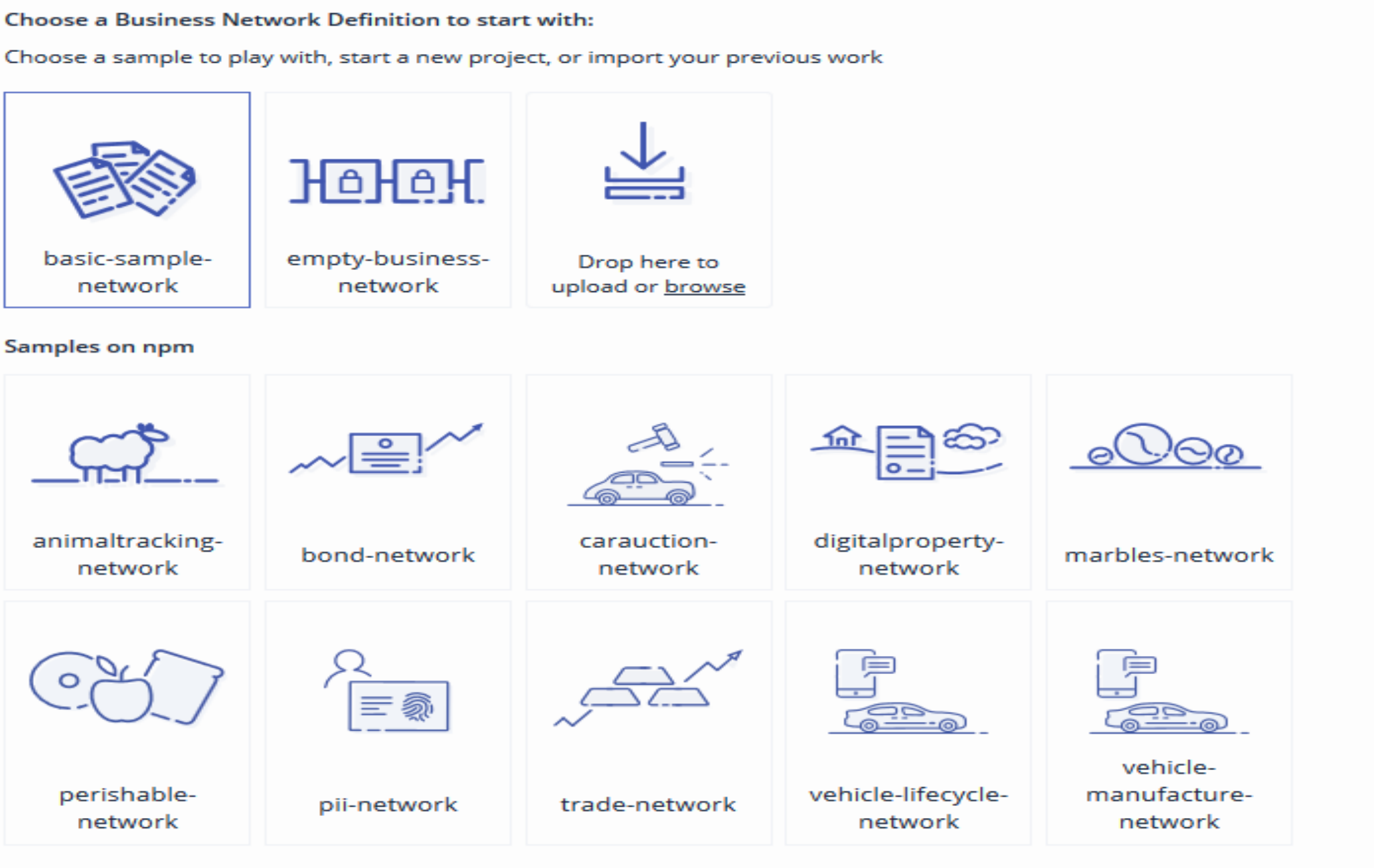
https://composer-playground.mybluemix.net/

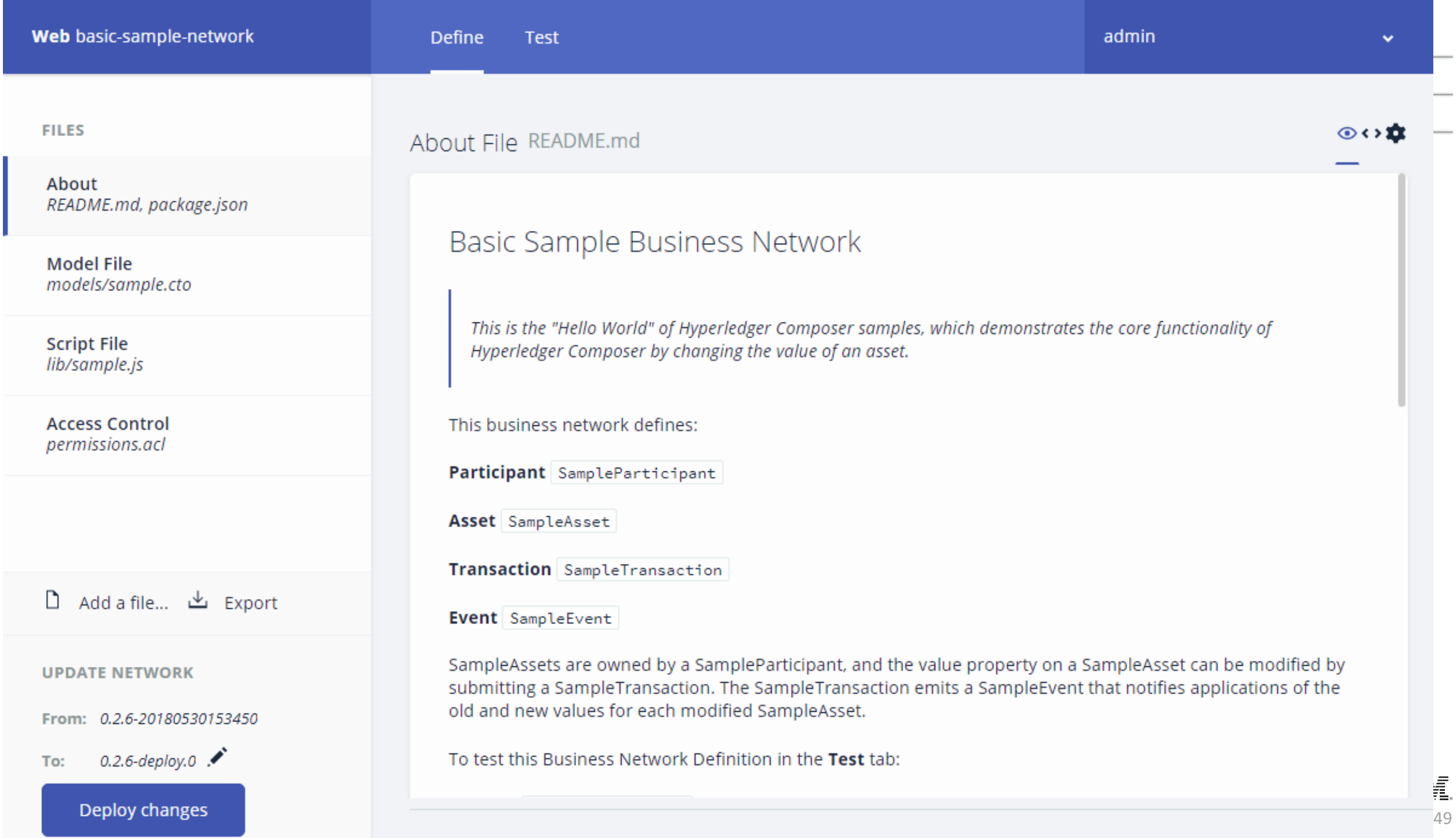




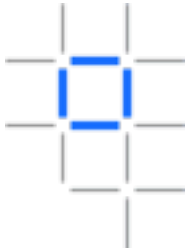
https://hyperledger.github.io/composer/latest/tutorials/playground-tutorial


Hyperledger Composer Playground
My Business Networks
Connection: Web Browser
こんにちは
Hello, Composer!
Get started with the basic-sample-network, or view our Playground tutorial
BUSINESS NETWORK
basic-sample-network
Get Started
Deploy a new business network


Choose a Business Network Definition to start with:
Choose a sample to play with, start a new project, or import your previous work
basic-sample-network
empty-business-network
Drop here to upload or browse
Samples on npm
animaltracking-network
bond-network
carauction-network
digitalproperty-network
marbles-network
perishable-network
pii-network
trade-network
vehicle-lifecycle-network
vehicle-manufacture-network


Web basic-sample-network
Define
Test
admin
FILES
About
README.md, package.json
Model File
models/sample.cto
Script File
lib/sample.js
Access Control
permissions.acl
Add a file...
Export
UPDATE NETWORK
From: 0.2.6-20180530153450
To: 0.2.6-deploy.0
Deploy changes
About File README.md
Basic Sample Business Network
This is the "Hello World" of Hyperledger Composer samples, which demonstrates the core functionality of Hyperledger Composer by changing the value of an asset.
This business network defines:
Participant SampleParticipant
Asset SampleAsset
Transaction SampleTransaction
Event SampleEvent
SampleAssets are owned by a SampleParticipant, and the value property on a SampleAsset can be modified by submitting a SampleTransaction. The SampleTransaction emits a SampleEvent that notifies applications of the old and new values for each modified SampleAsset.
To test this Business Network Definition in the Test tab:

Hyperledger Composer Modeling Language



Hyperledger Composer includes an object-oriented modeling language that is used to define the domain model for a business network definition

1. A single namespace. All resource declarations within the file are implicitly in this namespace.

2. A set of resource definitions, encompassing assets, transactions, participants, and events.

3. Optional import declarations that import resources from other namespaces.

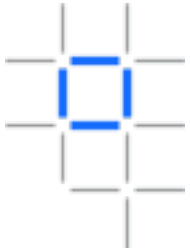
https://hyperledger.github.io/composer/latest/reference/cto_language

Model File models/sample.cto

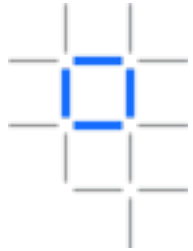
```
14
15  /**
16   * Sample business network definition.
17   */
18  namespace org.example.basic
19
20  asset SampleAsset identified by assetId {
21    o String assetId
22    --> SampleParticipant owner
23    o String value
24  }
25
26  participant SampleParticipant identified by participantId {
27    o String participantId
28    o String firstName
29    o String lastName
30  }
31
32  transaction SampleTransaction {
33    --> SampleAsset asset
34    o String newValue
35  }
36
37  event SampleEvent {
38    --> SampleAsset asset
39    o String oldValue
40    o String newValue
41  }
42
```

Script File

```
 */

/* global getAssetRegistry getFactory emit */

/**
 * Sample transaction processor function.
 * @param {org.example.basic.SampleTransaction} tx The sample transaction instance.
 * @transaction
 */
async function sampleTransaction(tx) {  // eslint-disable-line no-unused-vars

    // Save the old value of the asset.
    const oldValue = tx.asset.value;

    // Update the asset with the new value.
    tx.asset.value = tx.newValue;

    // Get the asset registry for the asset.
    const assetRegistry = await getAssetRegistry('org.example.basic.SampleAsset');
    // Update the asset in the asset registry.
    await assetRegistry.update(tx.asset);

    // Emit an event for the modified asset.
    let event = getFactory().newEvent('org.example.basic', 'SampleEvent');
    event.asset = tx.asset;
    event.oldValue = oldValue;
    event.newValue = tx.newValue;
    emit(event);
}
```

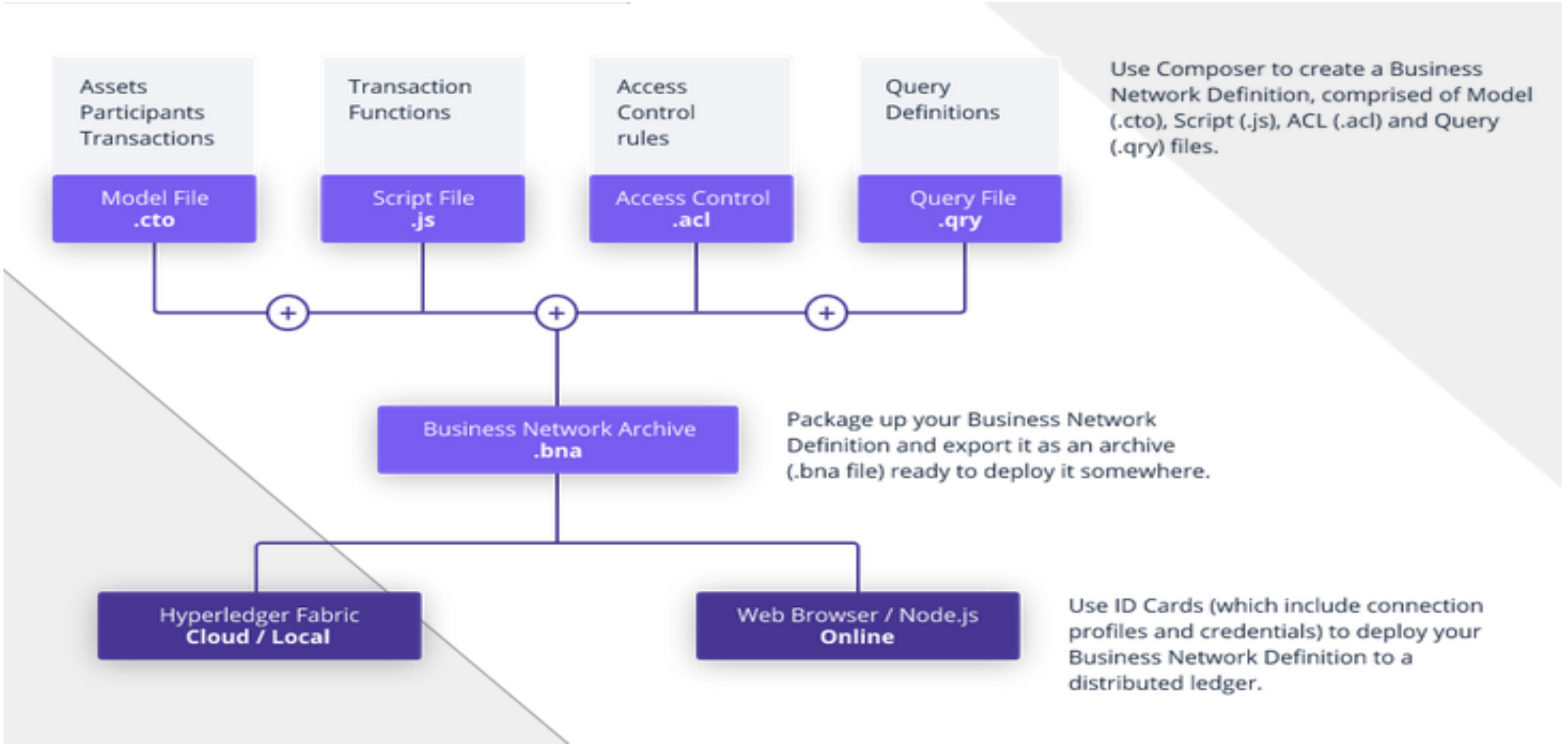


```
 * Sample access control list.
 */
rule EverybodyCanReadEverything {
    description: "Allow all participants read access to all resources"
    participant: "org.example.basic.SampleParticipant"
    operation: READ
    resource: "org.example.basic.*"
    action: ALLOW
}

rule EverybodyCanSubmitTransactions {
    description: "Allow all participants to submit transactions"
    participant: "org.example.basic.SampleParticipant"
    operation: CREATE
    resource: "org.example.basic.SampleTransaction"
    action: ALLOW
}

rule OwnerHasFullAccessToTheirAssets {
    description: "Allow all participants full access to their assets"
    participant(p): "org.example.basic.SampleParticipant"
    operation: ALL
    resource(r): "org.example.basic.SampleAsset"
    condition: (r.owner.getIdentifier() === p.getIdentifier())
    action: ALLOW
}
```

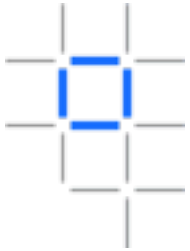


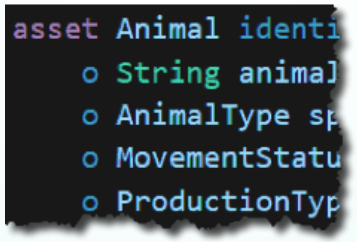
```
rule SystemACL {
    description: "System ACL to permit all access"
    participant: "org.hyperledger.composer.system.Participant"
    operation: ALL
    resource: "org.hyperledger.composer.system.**"
    action: ALLOW
}

rule NetworkAdminUser {
    description: "Grant business network administrators full access to user resources"
    participant: "org.hyperledger.composer.system.NetworkAdmin"
    operation: ALL
    resource: "**"
    action: ALLOW
}

rule NetworkAdminSystem {
    description: "Grant business network administrators full access to system resources"
    participant: "org.hyperledger.composer.system.NetworkAdmin"
    operation: ALL
    resource: "org.hyperledger.composer.system.**"
    action: ALLOW
}
```

Deploying the BNA file to Hyperledger Fabric and the Cloud



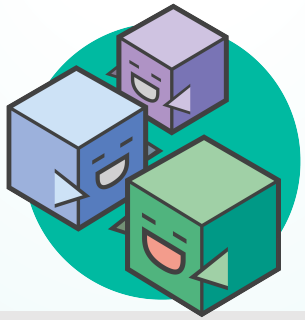
Extensive, Familiar, Open Development Toolset





Data modelling

JavaScript business logic



Web playground

Client libraries

Editor support

CLI utilities

Code generation

Existing systems and data

5 Beyond the Business Network

Pre-reqs for Ubuntu or the Mac

The following are prerequisites for installing the required development tools:

- At least 4Gb of memory
- Operating Systems: Ubuntu Linux 14.04 / 16.04 LTS (both 64-bit), or Mac OS 10.12
- Docker Engine: Version 17.03 or higher
- Docker-Compose: Version 1.8 or higher
- Node: 8.9 or higher (note version 9 is not supported)
- npm: v5.x
- git: 2.9.x or higher
- Python: 2.7.x
- A code editor of your choice, we recommend VSCode.

Note: Nvm use node

Now using node v8.11.2 (npm v5.6.0)

https://hyperledger.github.io/composer/latest/installing/installing-prereqs.html

Section 5.1 : install the Composer CLI tools and the Fabric

5. 1) Install Hyperledger Composer CLI tools

1. Essential CLI tools:

```
npm install -g composer-cli
```
Copy

2. Utility for running a REST Server on your machine to expose your business networks as RESTful APIs:

```
npm install -g composer-rest-server
```
Copy

3. Useful utility for generating application assets:

```
npm install -g generator-hyperledger-composer
```
Copy

4. Yeoman is a tool for generating applications, which utilises `generator-hyperledger-composer` :

Plus Yeoman

```
npm install -g yo
```
Copy

https://hyperledger.github.io/composer/latest/installing/development-tools.html

5.2) Install Hyperledger Fabric

```
mkdir ~/fabric-dev-servers && cd ~/fabric-dev-servers

curl -O
https://raw.githubusercontent.com/hyperledger/composer-
tools/master/packages/fabric-dev-servers/fabric-dev-
servers.tar.gz
tar -xvf fabric-dev-servers.tar.gz
```

Copy

A `zip` is also available if you prefer: just replace the `.tar.gz` file with `fabric-dev-servers.zip` and the `tar -xvf` command with a `unzip` command in the preceding snippet.

Use the scripts you just downloaded and extracted to download a local Hyperledger Fabric runtime:

```
cd ~/fabric-dev-servers
./downloadFabric.sh
```

Copy

https://hyperledger.github.io/composer/latest/installing/development-tools.html

5.3ß) Startup a new runtime

The first time you start up a new runtime, you'll need to run the start script, then generate a PeerAdmin card:

```
cd ~/fabric-dev-servers
./startFabric.sh
./createPeerAdminCard.sh
```

Copy

You can start and stop your runtime using `~/fabric-dev-servers/stopFabric.sh`, and start it again with `~/fabric-dev-servers/startFabric.sh`.

https://hyperledger.github.io/composer/latest/installing/development-tools.html

Section 2 Let’s write our application

Create a skeleton Business Network Archive with Yeoman

Create a skeleton business network using Yeoman. This command will require a business network name, description, author name, author email address, license selection and namespace.

yo hyperledger-composer:businessnetwork

1.Enter tutorial-network for the network name, and desired information for description, author name, and author email.
2.Select Apache-2.0 as the license.
3.Select org.example.mynetwork as the namespace.
4.Select No when asked whether to generate an empty network or

Business Network Archive files

- CTO Model file
- JavaScript
- ACL

Open the org.example.mynetwork.cto model file. Replace the contents with the following:

Copy

```
/**
 * My commodity trading network
 */
namespace org.example.mynetwork
asset Commodity identified by tradingSymbol {
    o String tradingSymbol
    o String description
    o String mainExchange
    o Double quantity
    --> Trader owner
}
participant Trader identified by tradeId {
    o String tradeId
    o String firstName
    o String lastName
}
transaction Trade {
    --> Commodity commodity
    --> Trader newOwner
}
```

https://hyperledger.github.io/composer/latest/tutorials/developer-tutorial.html

Replace the javascript file

```
/**
 * Track the trade of a commodity from one trader to another
 * @param {org.example.mynetwork.Trade} trade - the trade to be processed
 * @transaction
 */
async function tradeCommodity(trade) {
    trade.commodity.owner = trade.newOwner;
    let assetRegistry = await
getAssetRegistry('org.example.mynetwork.Commodity');
    await assetRegistry.update(trade.commodity);
}
```

Replace the acl file

```
/**
 * Access control rules for tutorial-network
 */
rule Default {
    description: "Allow all participants access to all resources"
    participant: "ANY"
    operation: ALL
    resource: "org.example.mynetwork.*"
    action: ALLOW
}

rule SystemACL {
  description:  "System ACL to permit all access"
  participant: "ANY"
  operation: ALL
  resource: "org.hyperledger.composer.system.**"
  action: ALLOW
}
```

Generate your Business Network Archive

From the tutorial-network directory, run the following command:

composer archive create -t dir -n .

After the command has run, a business network archive file called **tutorial-network@0.0.1.bna** has been created in the tutorial-network directory.

Deploy the business network

```
composer network install --card PeerAdmin@hlfv1 --archiveFile tutorial-network@0.0.1.bna
```

Copy

The `composer network install` command requires a PeerAdmin business network card (in this case one has been created and imported in advance), and the the file path of the `.bna` which defines the business network.

2. To start the business network, run the following command:

```
composer network start --networkName tutorial-network --networkVersion 0.0.1 --networkAdmin admin --networkAdminEnrollSecret adminpw --card PeerAdmin@hlfv1 --file networkadmin.card
```

Copy

The `composer network start` command requires a business network card, as well as the name of the admin identity for the business network, the name and version of the business network and the name of the file to be created ready to import as a business network card.

3. To import the network administrator identity as a usable business network card, run the following command:

```
composer card import --file networkadmin.card
```

Copy

The `composer card import` command requires the filename specified in `composer network start` to create a card.

4. To check that the business network has been deployed successfully, run the following command to ping the network:

```
composer network ping --card admin@tutorial-network
```

Copy

Business Network Cards

- A Business Network Card provides all of the information needed to connect to a blockchain business network.
- It is only possible to access a blockchain Business Network through a valid Business Network Card.
- A Business Network Card contains an Identity for a single Participant within a deployed business network.
- Business Network Cards are used in the Hyperledger Composer Playground to connect to deployed Business Networks.

https://hyperledger.github.io/composer/v0.16/playground/id-cards-playground

Section 3 Install Rest Server and Angular frontend and run our application.

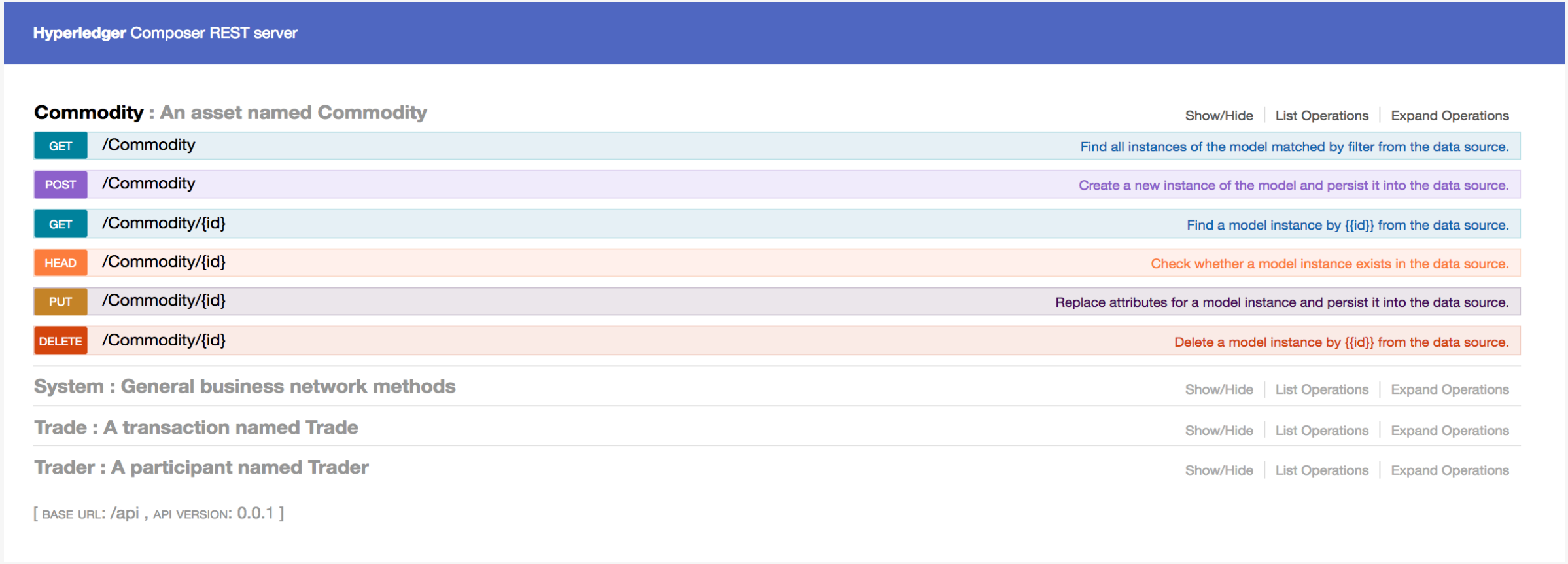
Install the Composer Rest Server on port 3000

1. To create the REST API, navigate to the `tutorial-network` directory and run the following command:

```
composer-rest-server
```

Copy

Port 3000



Generate the Angular Application

Step Six: Generating an application

Hyperledger Composer can also generate an Angular 4 application running against the REST API.

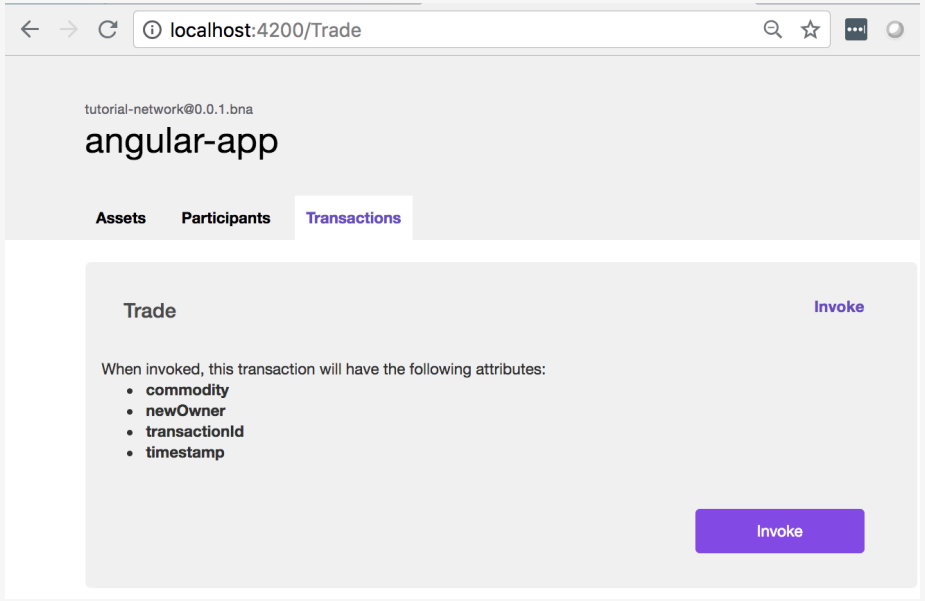
1. To create your Angular 4 application, navigate to `tutorial-network` directory and run the following command:

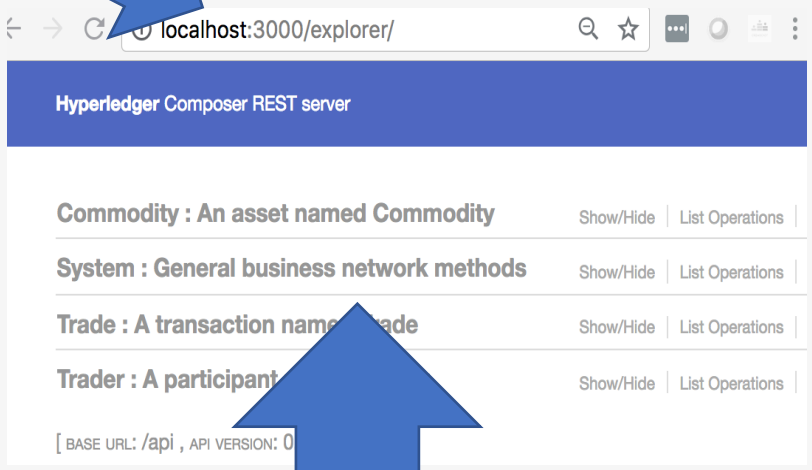
```
yo hyperledger-composer:angular
```

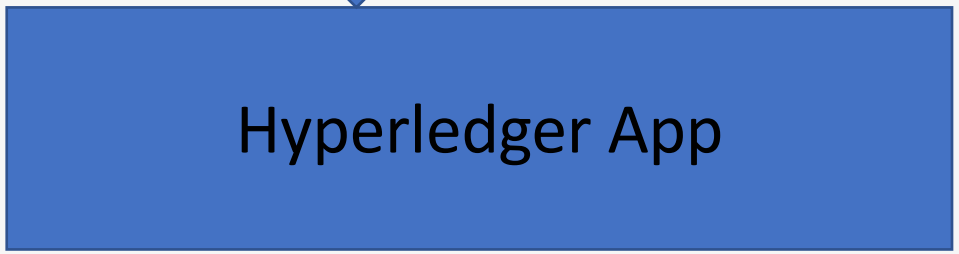
Copy

The Angular generator will then create the scaffolding for the project and install all dependencies. To run the application, navigate to your angular project directory and run **npm start .** This will fire up an Angular 4 application running against your REST API at http://localhost:4200 .

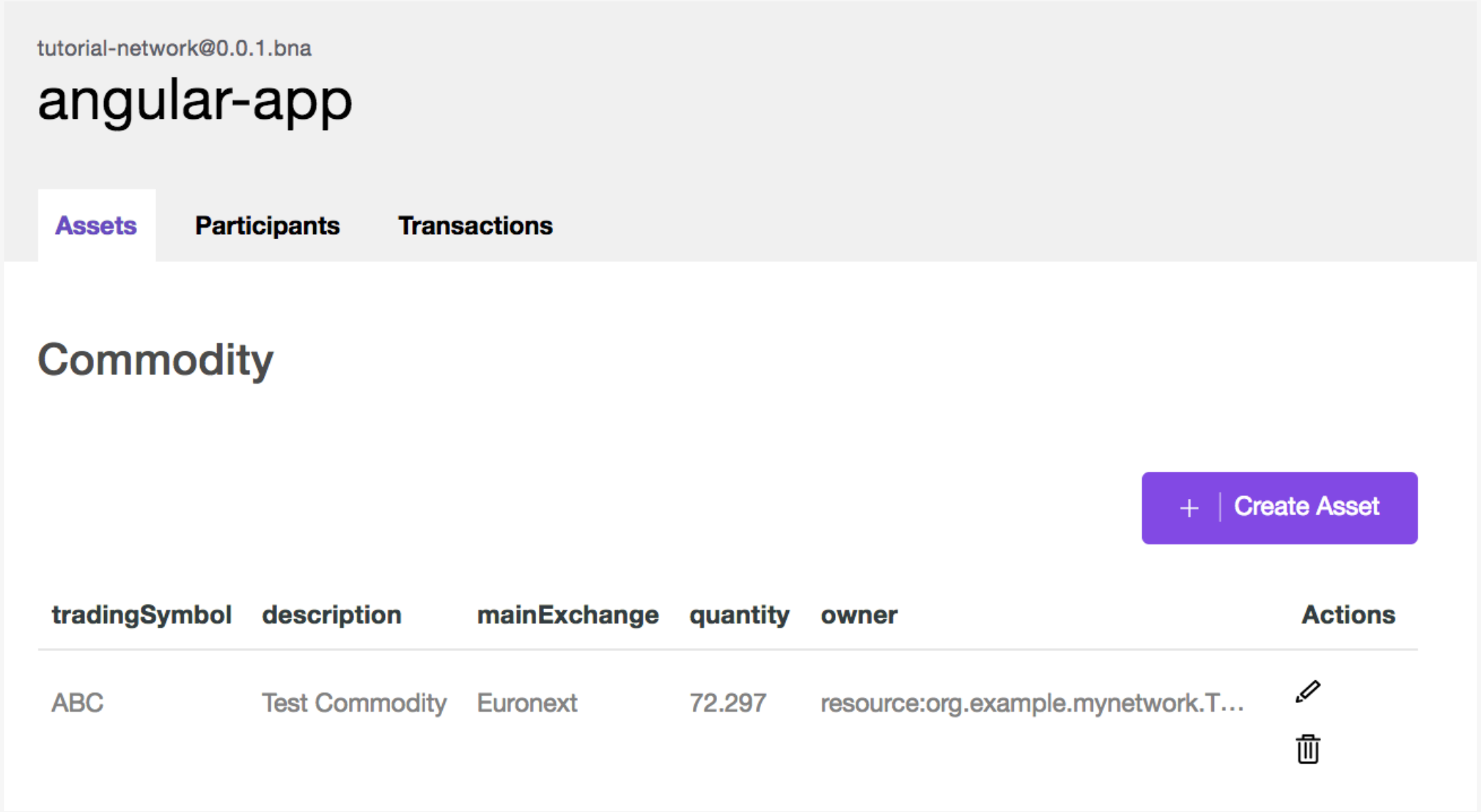
The Hyperledger App, REST server and Angular app







Angular app 1



The business network and complete applications

https://developer.ibm.com/code/patterns/decentralized-energy-hyperledger-composer/

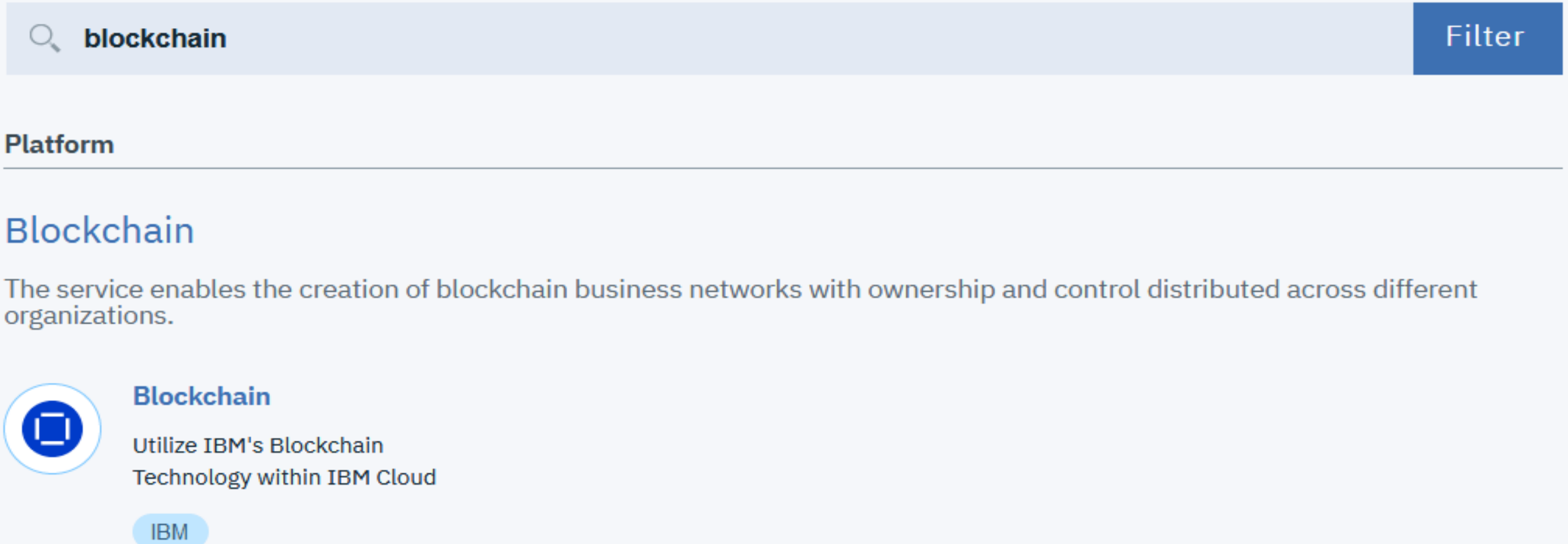
IBM Blockchain Code Patterns

https://developer.ibm.com/code/patterns/category/blockchain/

Step 4: Deploy Hyperledeger apps in the cloud

How do we host our Blockchain application?

IBM Cloud Catalog



Matterhorn Network - IPU

MY NETWORK

- Overview
- Members
- Channels
- Notifications
- APIs

MY CODE

- Develop code
- Install code
- Try samples

Overview

View and manage network resources for your organization. You can stop or start your resources and view the log file of the resource by selecting View Logs under "Actions". Learn more

Connection Profile | Add Peers

TYPE	NAME	STATUS	ACTIONS
Orderer	orderer	Running	
CA	org1-ca	Running	
Peer	org1-peer1	Running	

Members

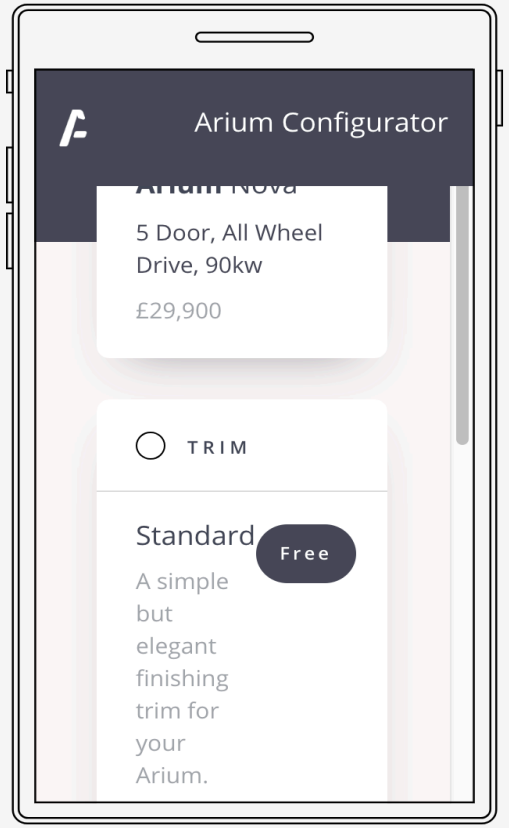
MEMBERS (2/15)	MSP ID	REQUESTER	STATUS
Company A alf@us.ibm.com	org1	--	Joined
Company B alf@us.ibm.com	org2	--	Joined

Channels

Search Channels

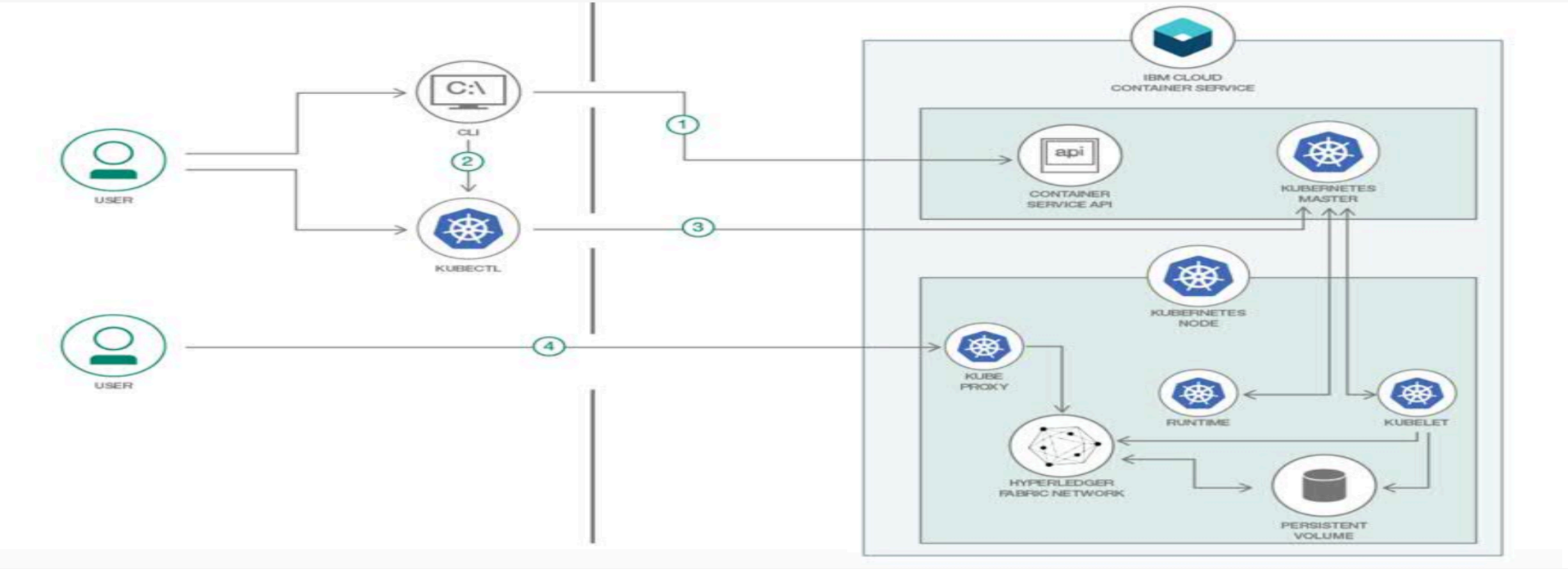
ID	TIME CREATED	BLOCK HEIGHT	PEERS	A
defaultchannel	07/18/18 06:08 PDT	1	org1-peer1	

Run Blockchain apps in the IBM Cloud



HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Deploy the Blockchain network using Kubernetes APIs on IBM Cloud



IBM Cloud Container Service allows you to create a free cluster that comes with 2 CPUs, 4 GB memory, and 1 worker node. It allows you to get familiar with and test Kubernetes capabilities. However they lack capabilities like persistent NFS file-based storage with volumes.

HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

https://github.com/IBM/blockchain-network-on-kubernetes

Thank You!

alf@us.ibm.com

https://hyperledger.org

https://ibm.com/blockchain