



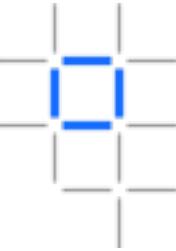
Creating Blockchain applications on Hyperledger Fabric and the IBM Cloud

Lennart Frantzell, IBM Developer Advocate

alf@us.ibm.com

August 2018

<http://ibm.biz/portblock>



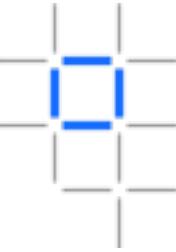
CALL FOR CODE®

Answer the Call for Code by building global solutions for disaster preparedness.

The most impactful project will be implemented with the help of IBM, The Linux Foundation, UN Human Rights, and American Red Cross.

Get started

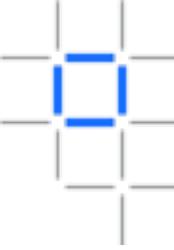
[Answer the call as an organization >](#)



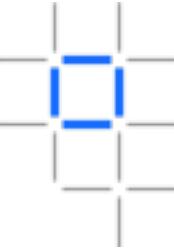
My background

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

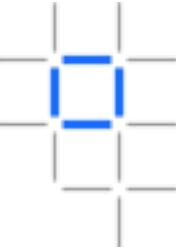
Agenda



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
6. Hosting Hyperledger Fabric in the Cloud

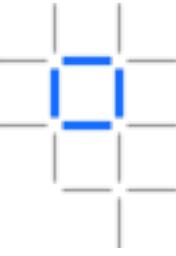


IBM and Blockchain

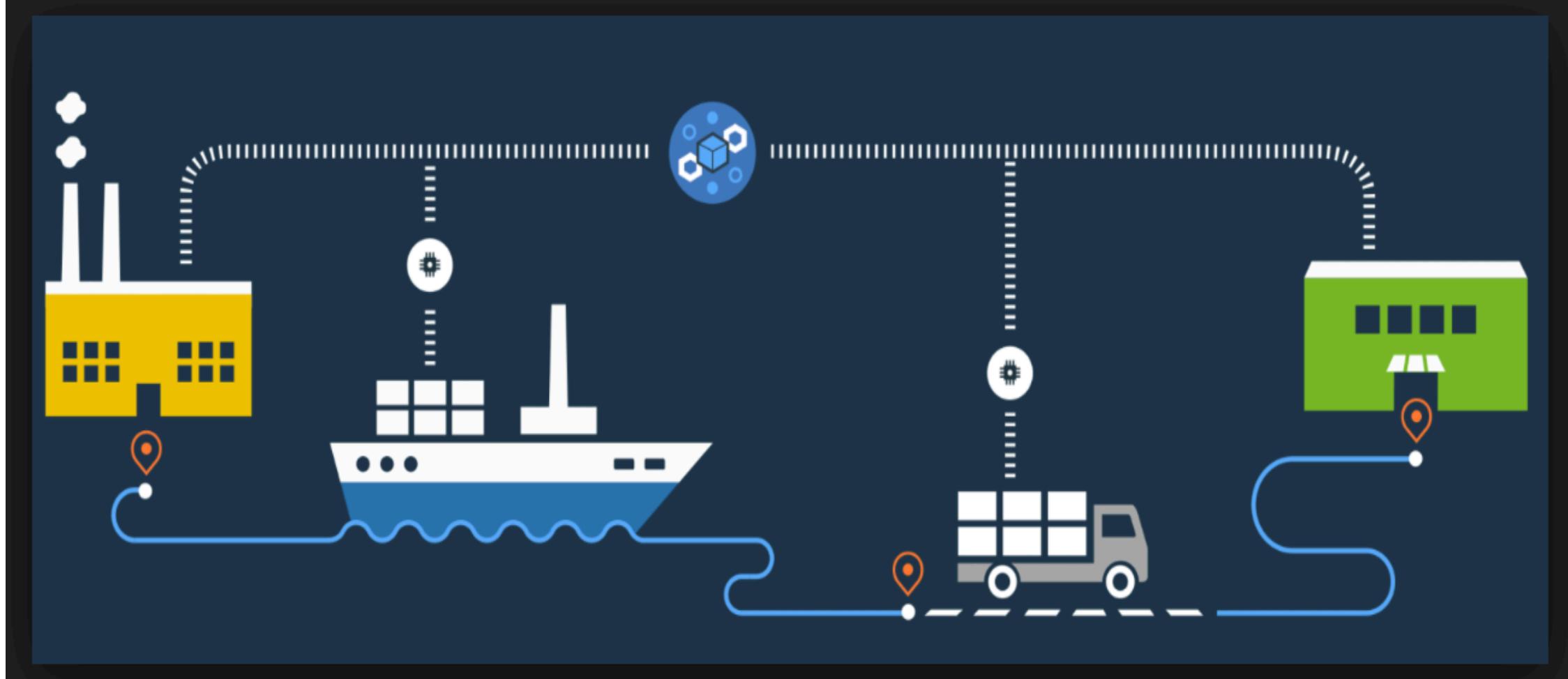


IBM's three most important technologies:

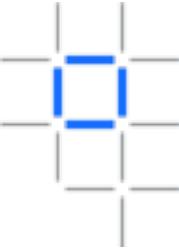
- Watson AI
- Blockchain with a team of 1,500 people
- Internet of Things



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

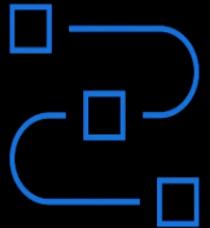


Blockchain and IoT



The blockchain process

0 1 0 1
1 0 1 0
0 1 0 1
1 0 1 0



1. IoT data

Each transaction is recorded, put into a block, and added to a secure, irreversible data chain

2. Watson IoT Platform

Select data is managed, analyzed, and customized to share among permissioned clients and partners

3. IBM Blockchain Platform

The IBM Blockchain Platform, powered by The Linux Foundation's Hyperledger, provides the necessary infrastructure for developing, operating, and governing data enterprise solutions

4. Business consumption

IBM Blockchain streamlines processes and creates new business value along your ecosystem

Spectrum of Blockchains

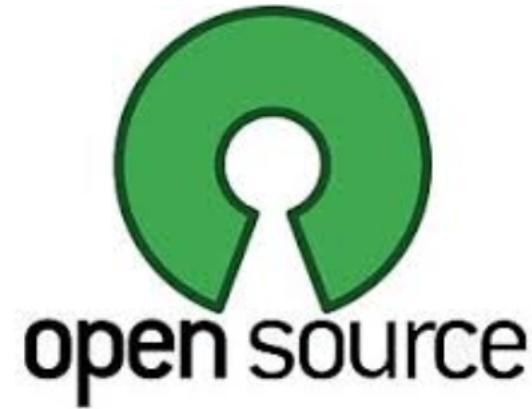
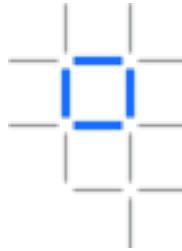


Permissionless Public	Permissionless Private	Permissioned Public	Permissioned Private
Bitcoin, Ethereum	Public Polls	Land Titles, University Degrees	Business Apps Medical Records

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 and the Blockchain Story



**HYPERLEDGER
FABRIC**

IBM and Blockchain



Search



IBM Blockchain

Learn ▾

Solutions ▾

Services

Industries ▾

Ecosystem

News and events

IBM Blockchain

Let's use blockchain to bring transparency to every shipment. Let's put smart to work.TM

→ Explore IBM Blockchain Platform → See all IBM Blockchain products



Disrupting the music industry

→ Register to hear blockchain startup True Tickets tell their story

New transparency in media buying

⇒ Register to learn how this could change advertising

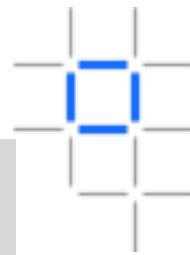
Start developing now

→ Claim \$500 in credits for your first network

Blockchain best practices

→ Register to learn from other IBM clients — The Founder's Handbook

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



TRADELENS: DIGITIZING THE GLOBAL SUPPLY CHAIN

TradeLens is an open and neutral industry platform underpinned by Blockchain technology, supported by major industry players.

•

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

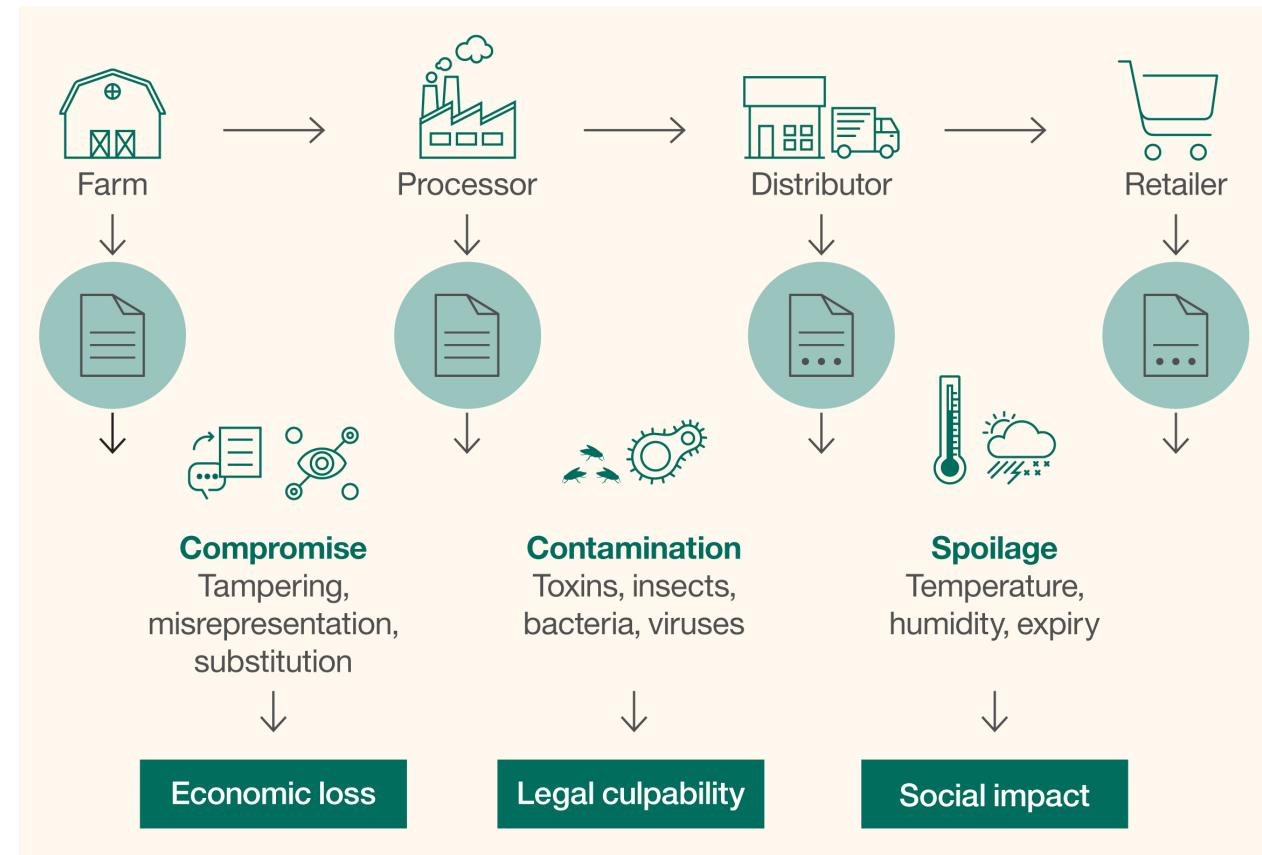
- Provide a trusted source of information and traceability to improve transparency and efficiency across the food network.

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.





Global Financing: Dispute Resolution

What?

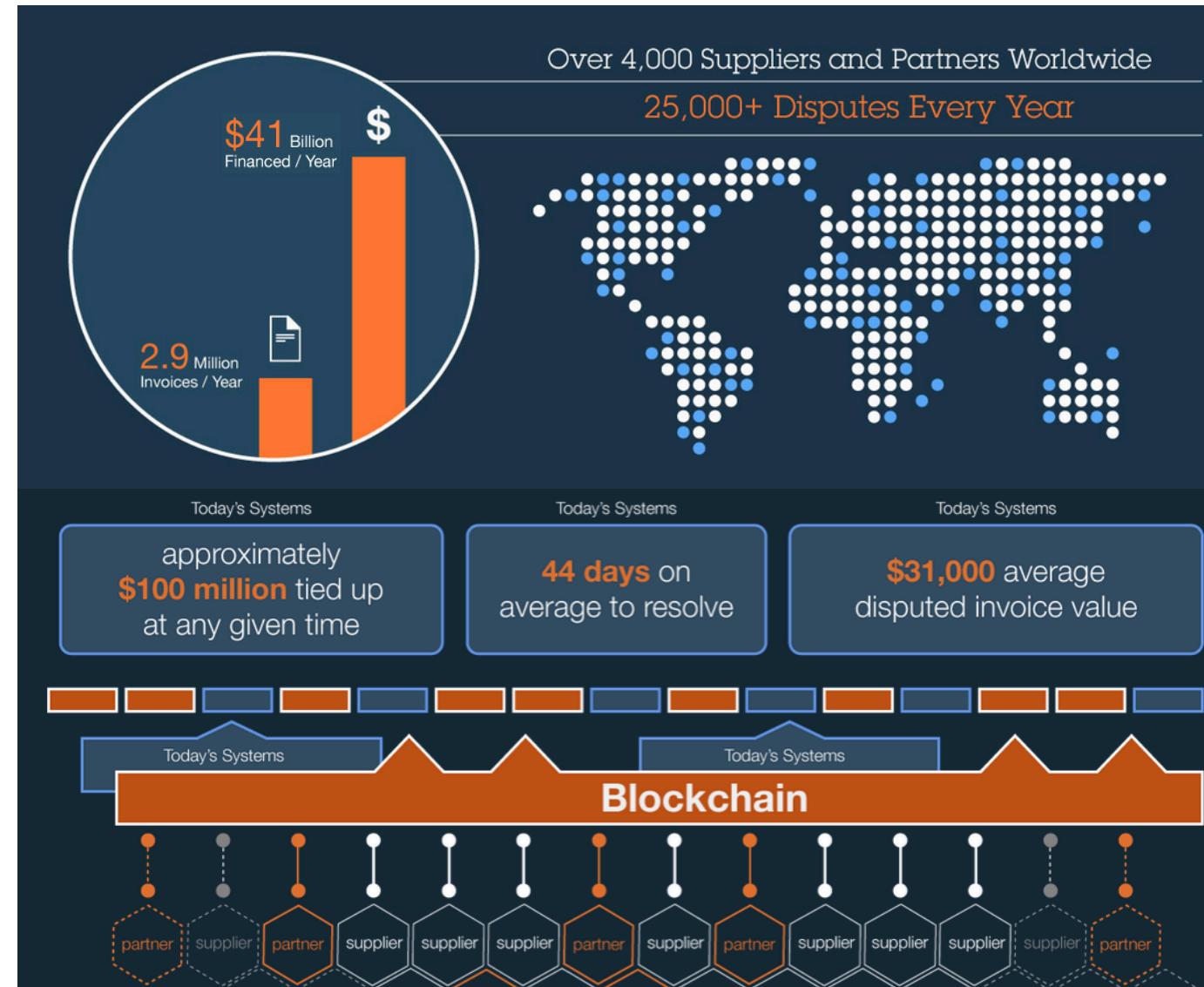
- IBM Global Finance provides a \$41bn channel financing per year. There are a number of disputes that take time to resolve and can lock up transactions costing time and money

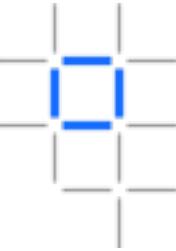
How?

- **Blockchain provides visibility and provenance end-to-end across supply chain**

Benefits

1. Reduced dispute resolution time by 75%
2. Released working capital from \$100m
3. Combine IGF and Supplier info to further expand benefits further
4. In production since Sept 2016





Some key IBM'ers in Blockchain development



Tweets 16.9K Following 1,844 Followers 4,641 Likes 44.5K Lists 12

Following

Chris Ferris



Tweets 39 Following 111 Followers 361

christian cachin @cczurich
Program is out -- Workshop on Blockchain Technology and Theory
cachin.com/cc/
Joined December 2013

Tweets **Tweets & replies** **Media**

christian cachin @cczurich · Jul 11
Program is out -- Workshop on Blockchain Technology and Theory
[@podc_conference](https://www.podc-conference.org) In two weeks in London! Looking forward to great selection
of talks by excellent speakers: 2018.blockchain-workshop.net/program.shtml

Christian Cachin



Tweets 517 Following 114 Followers 480 Likes 489

Arnaud J Le Hors @jhors
Senior Technical Staff Member, Web and Blockchain Open Technologies, IBM
<standard_disclaimer>

Tweets **Tweets & replies** **Media**

Arnaud J Le Hors Retweeted
Dr. Angel Luis Diaz @angelluisdiaz · Aug 9
Why IBM is offering \$200k to developers to create tech solutions for natural disaster relief techcrunch.com/2018/08/09/why-ibm-is-offering-200k-to-developers-to-create-tech-solutions-for-natural-disaster-relief/ via @techrepublic

Sandro Hawke @sandha...
Follow

Jerry Cuomo @JerryCuomo

IBM Blockchain Arnaud Le Hors



Tweets 553 Following 239 Followers 898 Likes 1,105 Lists 1

Simon Stone @mrsimonstone

Simon Stone @mrsimonstone · Aug 5

Tweets **Tweets & replies** **Media**

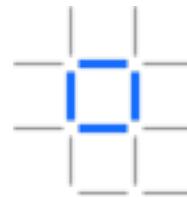
Introducing Hyperledger

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

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

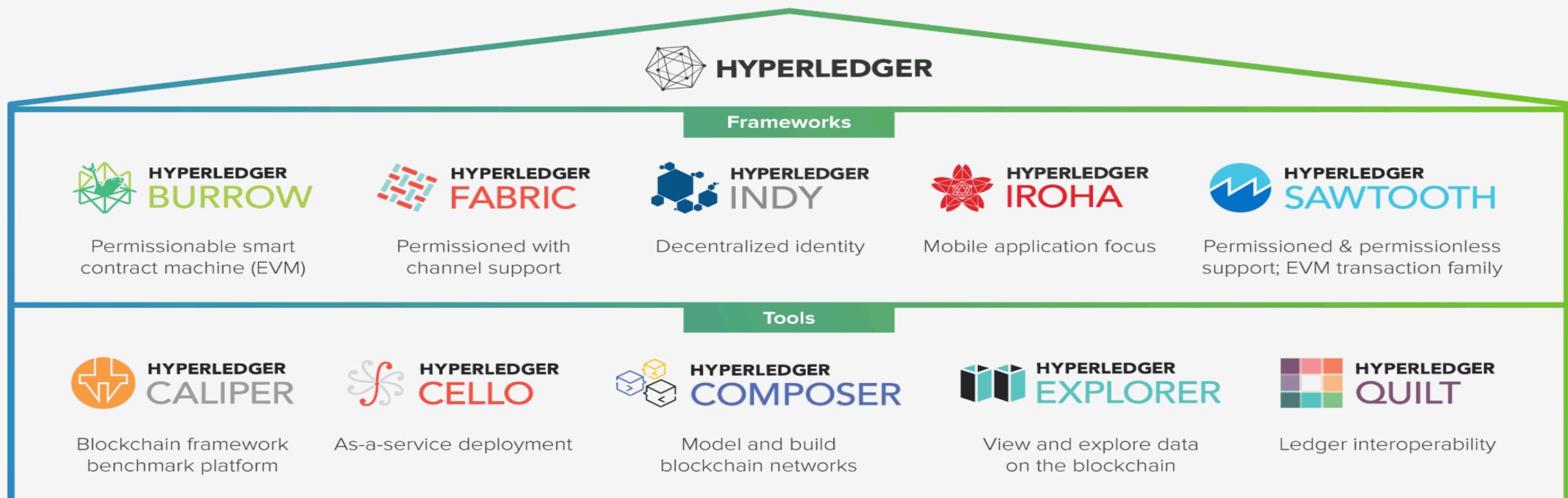
Global collaboration
spanning **finance,**
banking, IoT, supply
chain, healthcare,
manufacturing, govt,
technology and more.

Which Blockchain do we want to use?

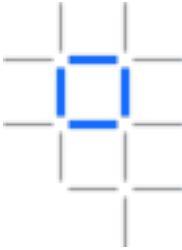


The Hyperledger Greenhouse

Business Blockchain Frameworks & Tools Hosted by Hyperledger



<https://www.hyperledger.org/>



Hyperledger Fabric

 **HYPERLEDGER**

Members Projects Community Resources News & Events Blog About 



HYPERLEDGER FABRIC

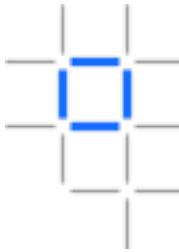
GET THE CODE **BUILD YOUR FIRST NETWORK**

Type: DLT, Smart Contract Engine
Status: Active

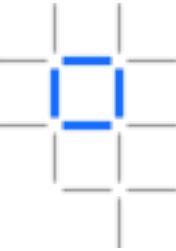
Hyperledger Fabric Explainer  

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

Languages in Hyperledger Fabric



1. Hyperledger Fabric is written in GoLang.
2. Chaincode is written in [Go](#), [node.js](#), and eventually in other programming languages such as Java, that implements a prescribed interface.



Hyperledger Composer



HYPERLEDGER

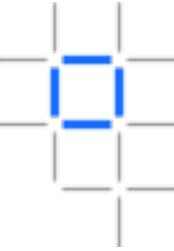
Members Projects Community Resources News & Eve



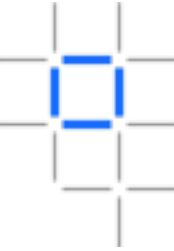
GET THE CODE

GET STARTED

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

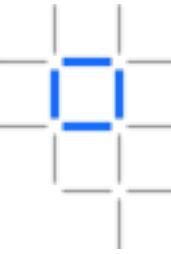


Creating an application in Hyperledger Fabric



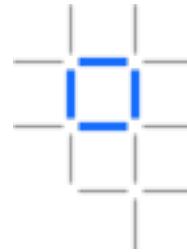
Step 1 Use Cases

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 trust
 - 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](#)

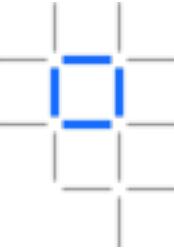


European trade finance network: we.trade

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

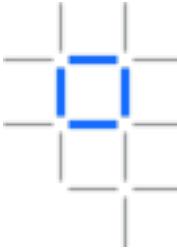
[▶ Watch the video \(02:48\)](#)

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



Step 2 Architecture

Opportunities and limitations



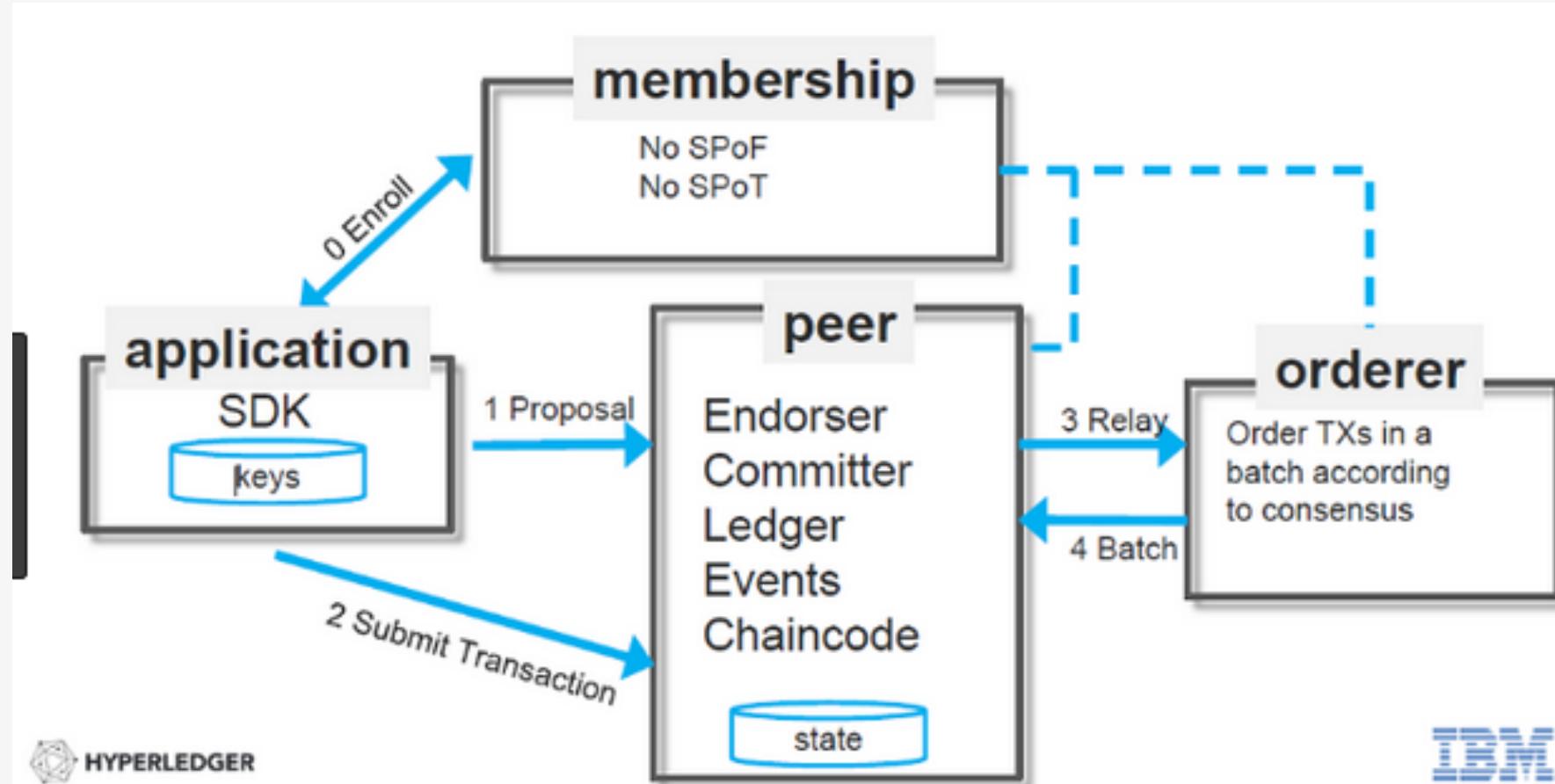
Hyperledger Fabric has three components

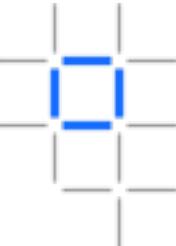
1. **Membership Services** – which manages digital certificates and access,
 2. **Ordering service** – which keeps the peers in alignment,
 3. and **peers** – which hold the ledger.
 1. A peer is also where **chaincode**, also known as smart contracts, are executed.
- Membership and ordering services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

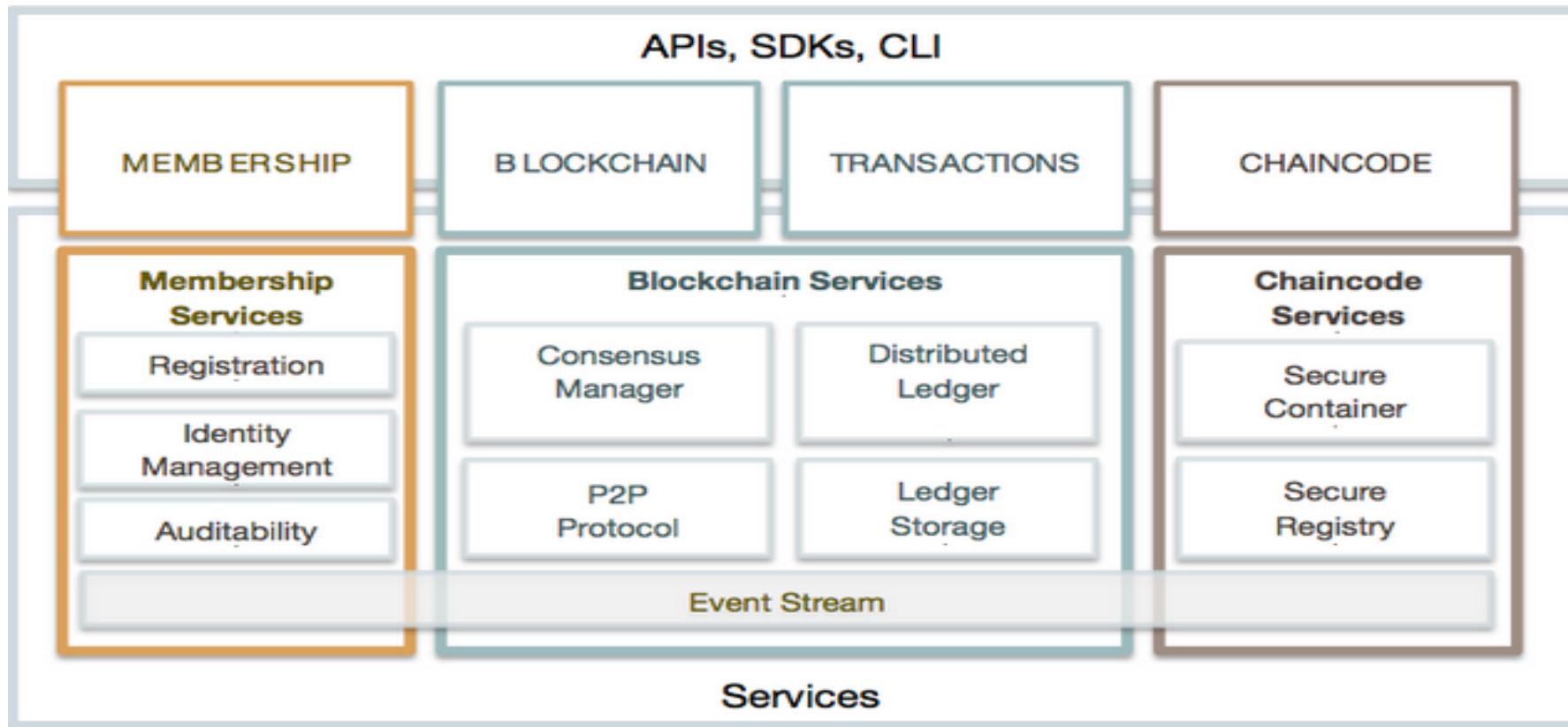
<https://www.ibm.com/blogs/systems/blockchain-how-should-you-organize-your-peers/>

Hyperledger architecture



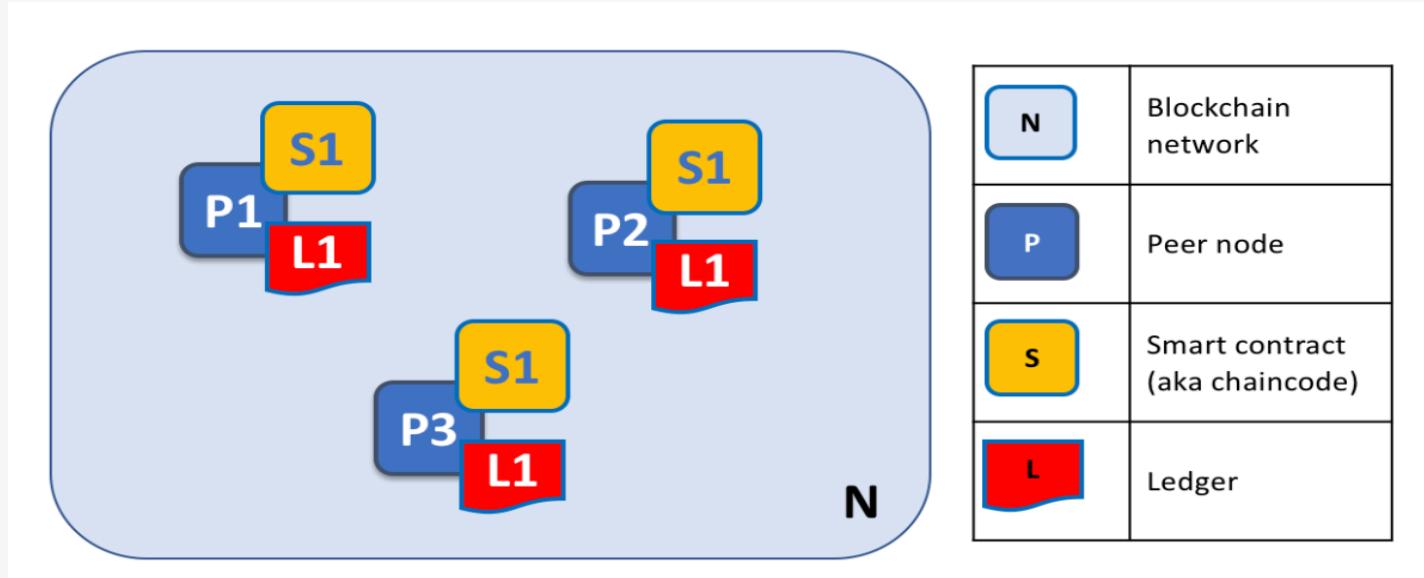


Hyperledger Fabric architecture



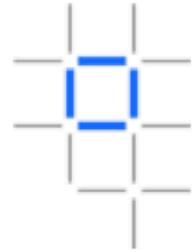
- Hyperledger Fabric allows components, such as consensus and membership services, to be **plug-and-play**.
- Hyperledger Fabric leverages **container technology** (Docker) to host **smart contracts called “chaincode”** that comprise the application logic of the system.
- Chaincode is like Stored Procedures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

Hyperledger Fabric components

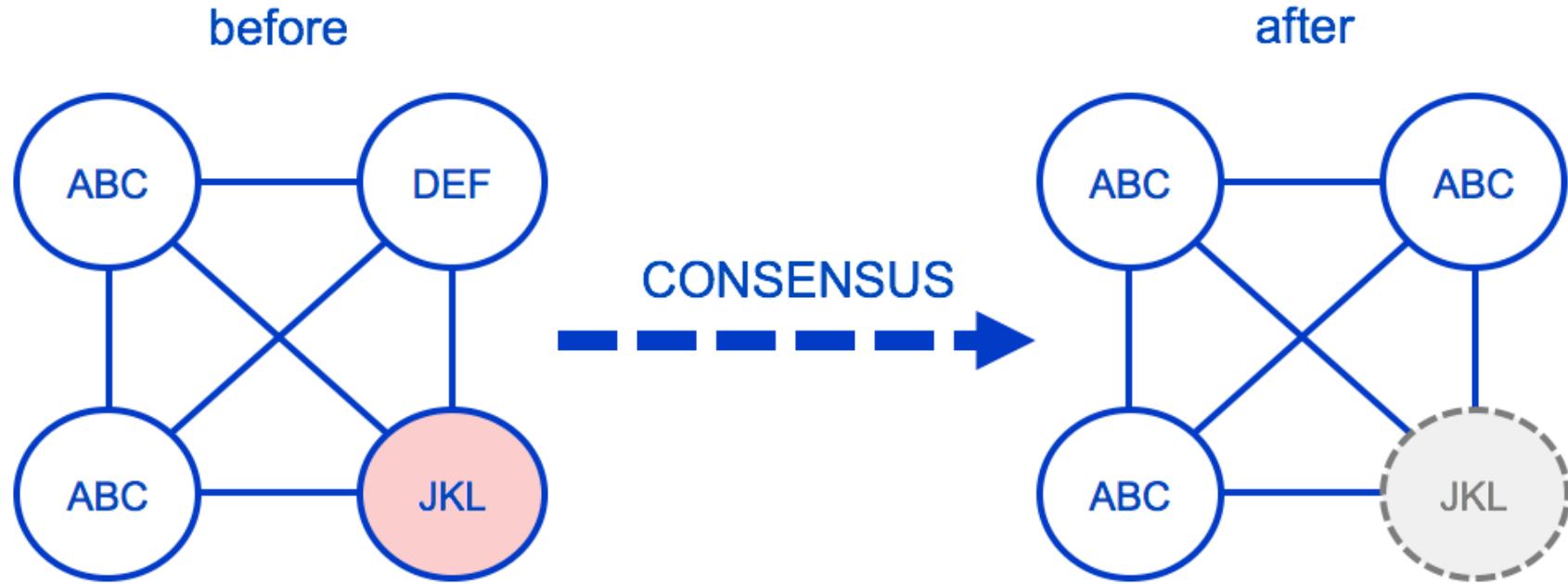


A blockchain network is comprised of peer nodes, each of which can hold copies of ledgers and copies of smart contracts. In this example, the network N consists of peers P1, P2 and P3, each of which maintain their own instance of the distributed ledger L1. P1, P2 and P3 use the same chaincode, S1, to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

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

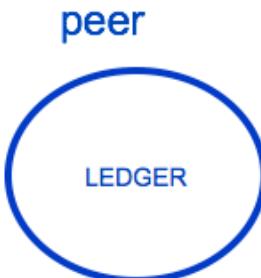


Consensus: The process of maintaining a consistent ledger



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



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

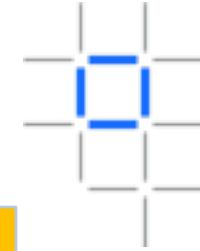
CONS: Does not guard against malicious activity

Example usage: Hyperledger Fabric V1



HYPERLEDGER

BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

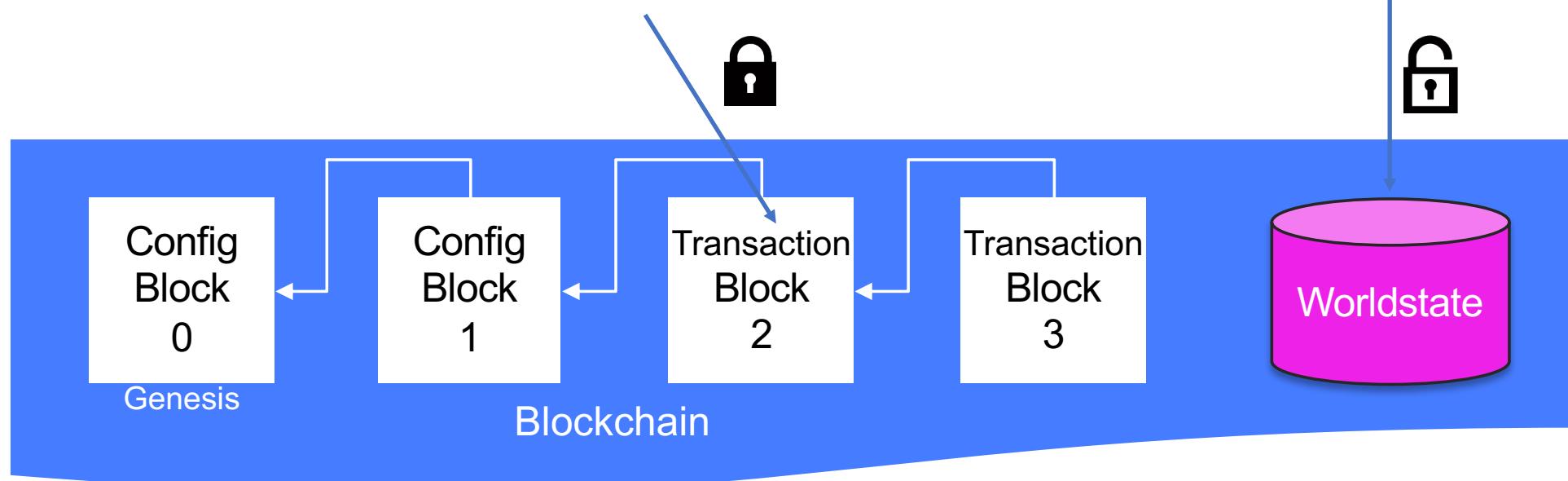


The “Right to Erasure” and blockchain

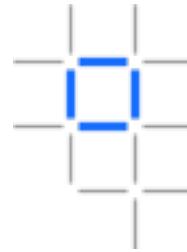
Enables an individual to request the deletion or removal of their personal data

Transactions
in
the
blockchain
are immutable

Data in the world
state is mutable



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



- **What**

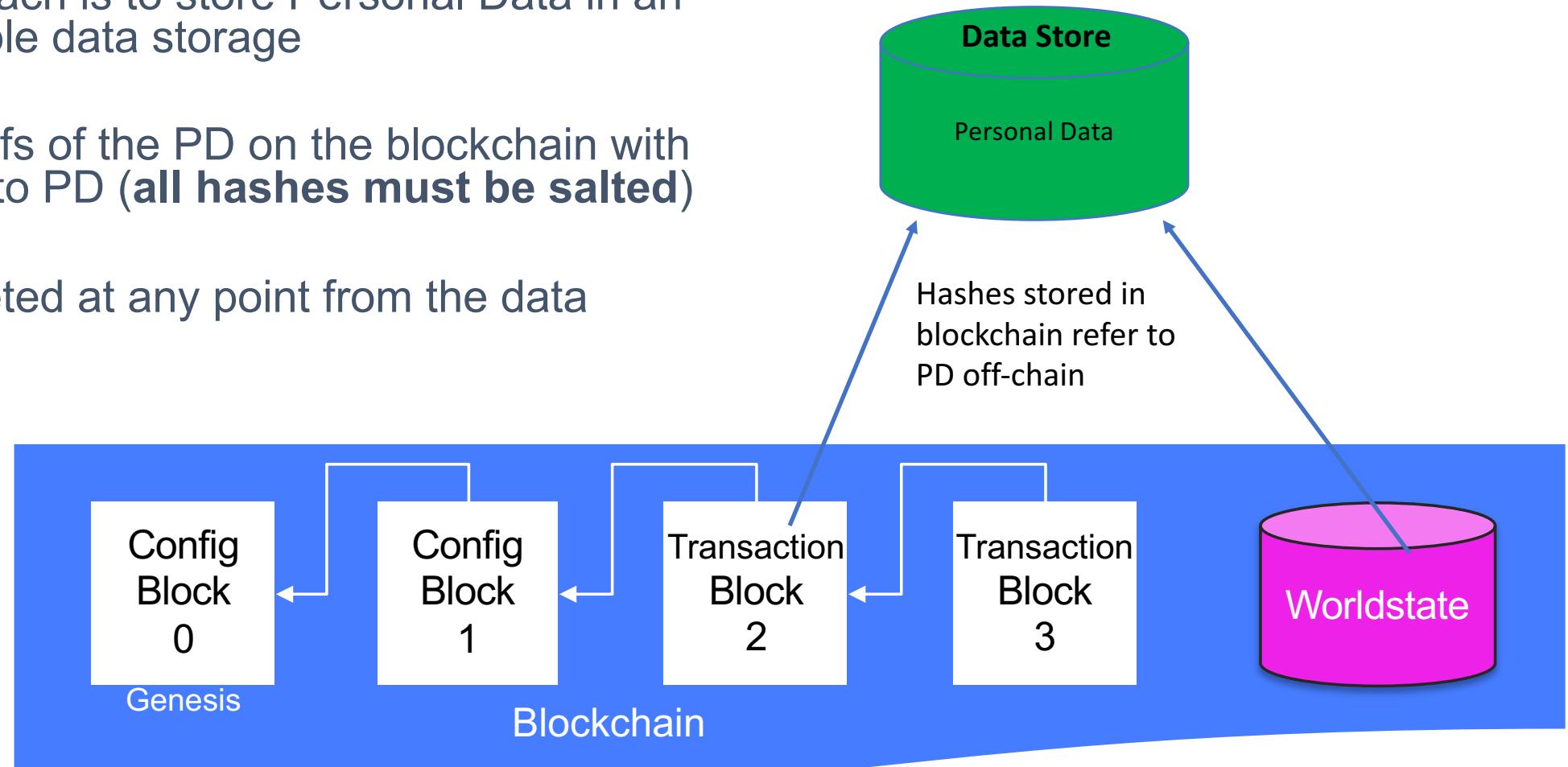
- The only approach is to store Personal Data in an off-chain mutable data storage

- **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)



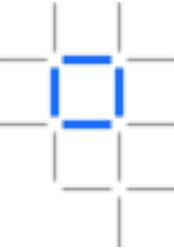
Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains

[Elli Androulaki](#), [Artem Barger](#), [Vita Bortnikov](#), [Christian Cachin](#), [Konstantinos Christidis](#), [Angelo De Caro](#), [David Enyeart](#), [Christopher Ferris](#), [Gennady Laventman](#), [Yakov Manevich](#), [Srinivasan Muralidharan](#), [Chet Murthy](#), [Binh Nguyen](#), [Manish Sethi](#), [Gari Singh](#), [Keith Smith](#), [Alessandro Sorniotti](#), [Chrysoula Stathakopoulou](#), [Marko Vukolić](#), [Sharon Weed Cocco](#), [Jason Yellick](#)

Fabric is a modular and extensible open-source system for deploying and operating permissioned blockchains and one of the Hyperledger projects hosted by the Linux Foundation (www.hyperledger.org).

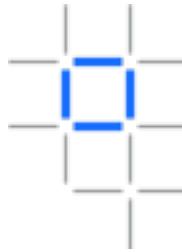
Fabric is the first truly extensible blockchain system for running distributed applications. It supports modular consensus protocols, which allows the system to be tailored to particular use cases and trust models. Fabric is also the first blockchain system that runs distributed applications written in standard, general-purpose programming languages, without systemic dependency on a native cryptocurrency. This stands in sharp contrast to existing blockchain platforms that require "smart-contracts" to be written in domain-specific

This paper describes Fabric, its architecture, the rationale behind various design decisions, its most prominent implementation aspects, as well as its distributed application programming model. We further evaluate Fabric by implementing and benchmarking a Bitcoin-inspired digital currency. We show that Fabric achieves end-to-end throughput of more than 3500 transactions per second in certain popular deployment configurations, with sub-second latency, scaling well to over 100 peers.



Step 3 Planning our app

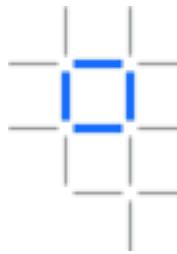
Planning the Peers



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.

Which user roles do you need in a business network?



- **The consortium owner:** This is the organization with the vision for the network, its business benefits and how it will disrupt how business is done. Today, for practical purposes, it is best if there is only one consortium owner, since that entity will be responsible for making the network work.
- **Members:** They are part of the Hyperledger network. **They have a peer, which means they can store data, execute chain code (smart contracts), and participate in the consensus process** (if the consortium owner allows it). Members bring substance and validity to the network, and it is their data and intelligence that is shared. As the network matures, members can move up in status to become consortium owners.
- **General users:** These are users of the network and they are sponsored and enabled by the consortium owner and the members. **They can access and input data, but they do not own data.** As with members, as the network matures, general users can become members if they bring value to the network.

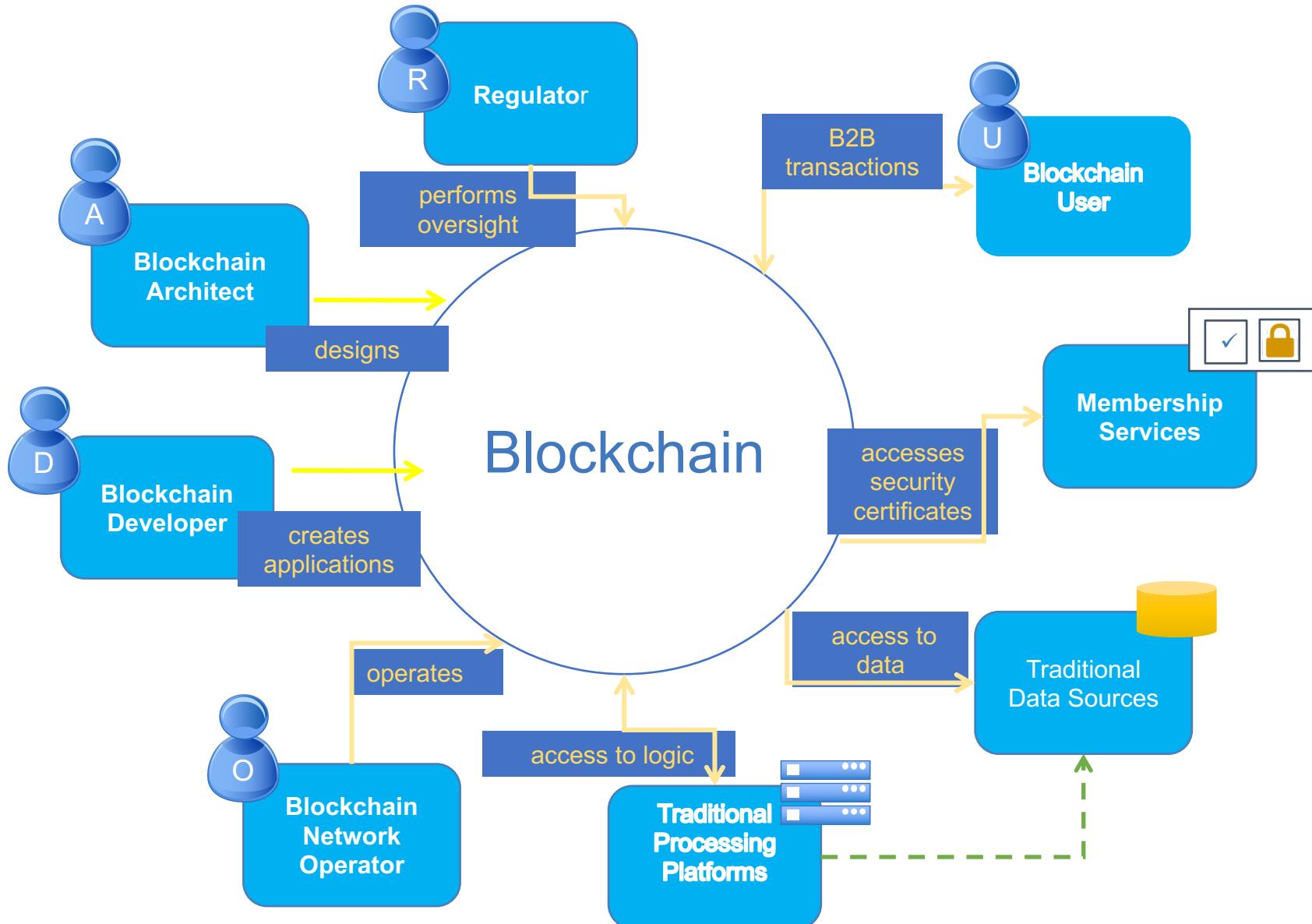
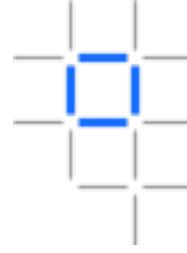
<https://www.ibm.com/blogs/systems/blockchain-who-should-be-in-your-consortium/>

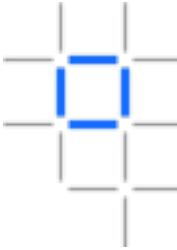
Hyperledger Fabric Performance

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/>

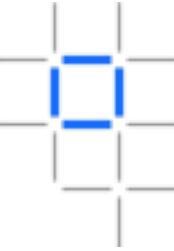
Actors in a Blockchain solution





The Blockchain architect

- Domain and Use Case experience
- **Fundamental grasp of cryptography** and Hyperledger solution architectures
- Apply Hyperledger security technologies
- Experience with Open source and Blockchain platforms, including Cryptocurrencies
- Technical expertise in Cloud-based blockchain services

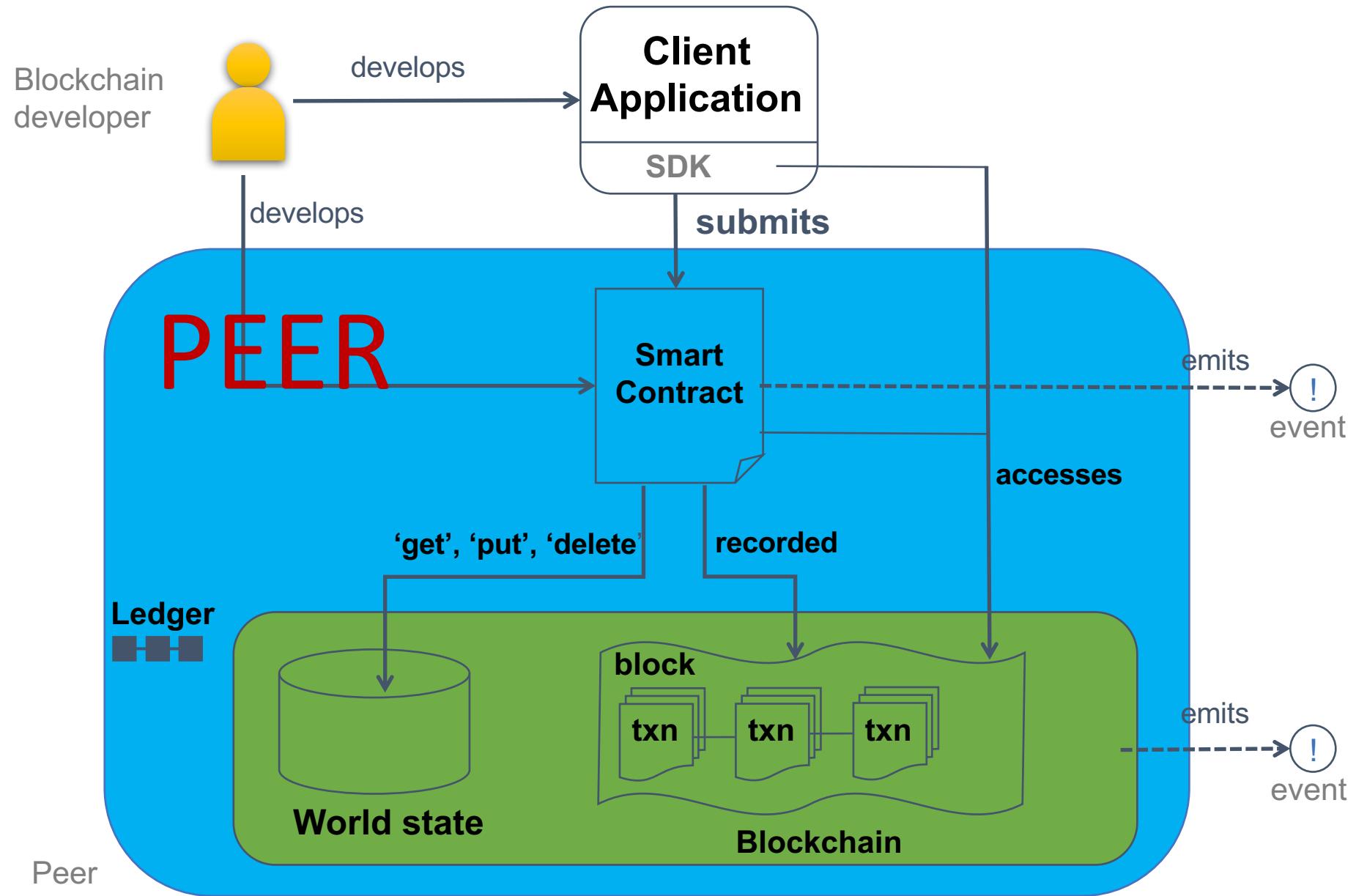
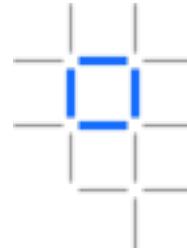


Step 4: Programming the ledger with Chaincode alias Smart Contracts

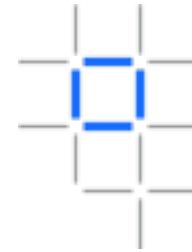
Chaincode and Smart Contracts

- 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

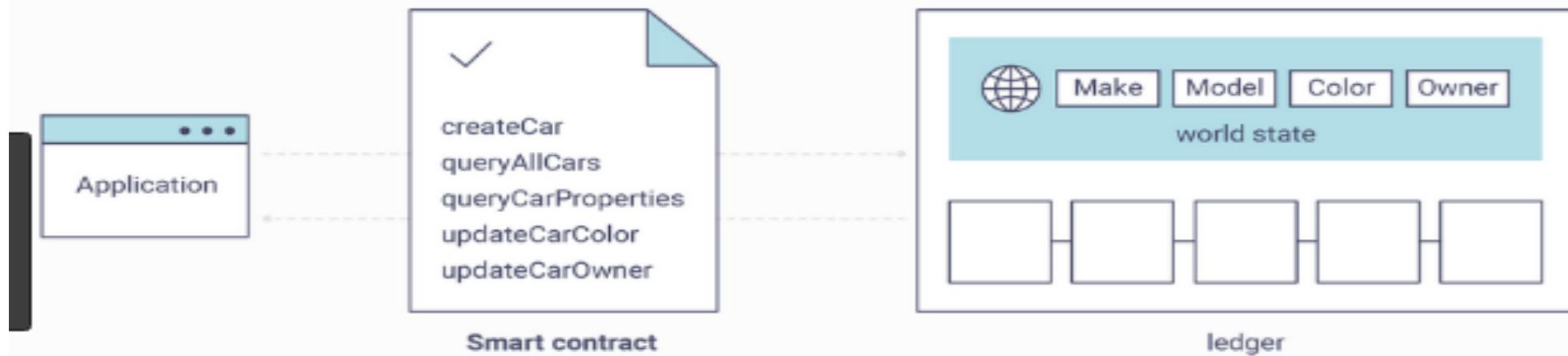
Programming the ledger



Chaincode, aka Smart Contracts

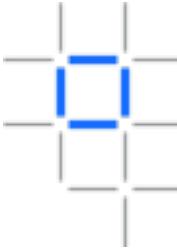


Below is a representation of how an app would call different functions in chaincode. Each function must be coded against an available API in the chaincode shim interface, which in turn allows the smart contract container to properly interface with the peer ledger.



We can see our `queryAllCars` function, as well as one called `createCar`, that will allow us to update the ledger and ultimately append a new block to the chain in a moment.

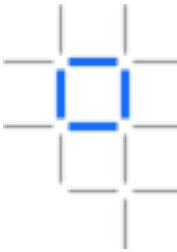
<https://hyperledger-fabric.readthedocs.io/en/release-1.2/chaincode.html>



Two Chaincode Personas

- One, from the perspective of an application developer developing a blockchain application/solution entitled **Chaincode for Developers**, and the other,
- **Chaincode for Operators** oriented to the blockchain network operator who is responsible for managing a blockchain network/
<https://hyperledger-fabric.readthedocs.io/en/release-1.2/chaincode.html>

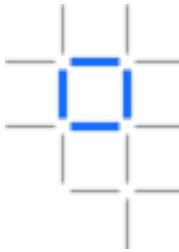
More on Chaincode



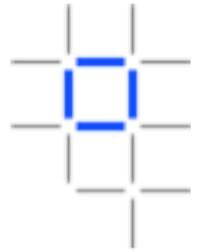
- A chaincode typically handles business logic agreed to by members of the network, so it may be considered as a “smart contract”.
- State created by a chaincode is scoped exclusively to that chaincode and can't be accessed directly by another chaincode.
- However, within the same network, given the appropriate permission a chaincode may invoke another chaincode to access its state.

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

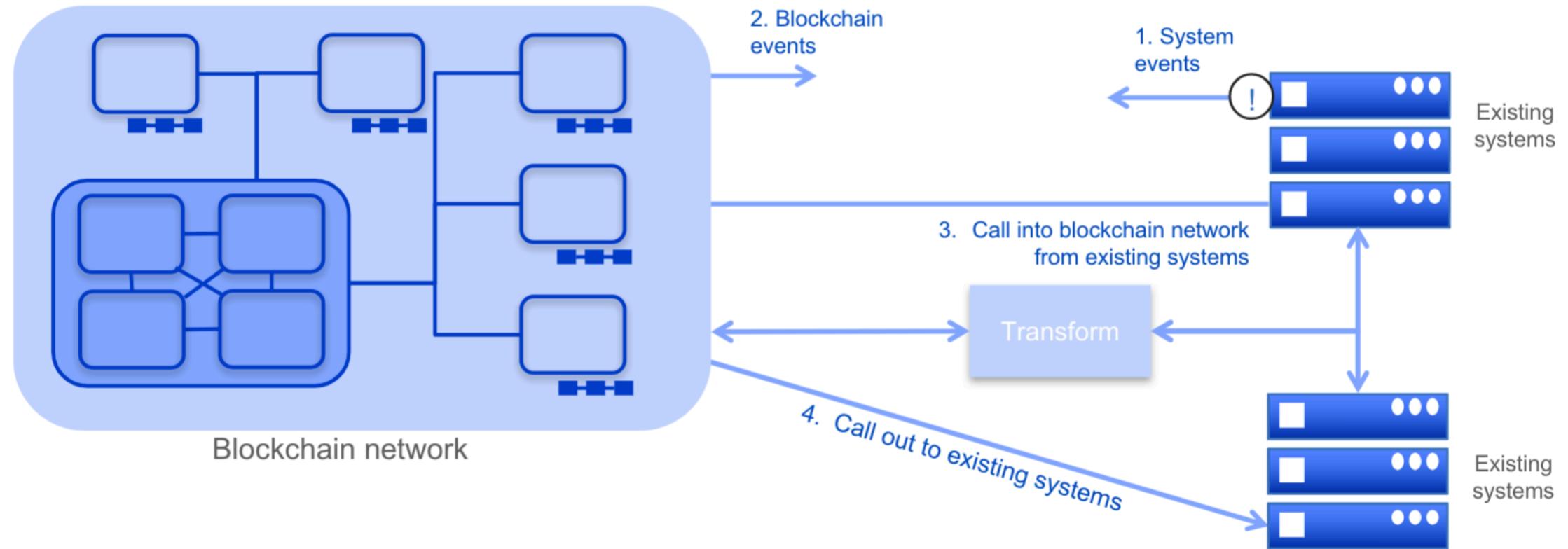
Off-chain, the problem with the Systems of Record

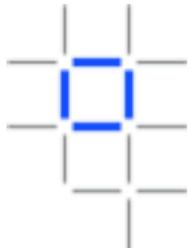


- All data can not be stored on the Blockchain
 - Example X-ray images
 - Diamonds
 - Art works
 - Automobiles
 - Aircraft parts
 - Food



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

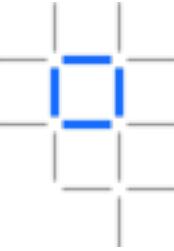
random()

getExchangeRate()

getDateTime()

getTemperature()

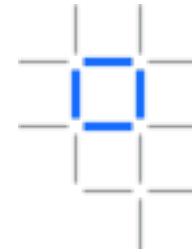
incrementValue
inExternalSystem(...)



Step 4.1: Programming with Hyperledger Composer

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

Welcome to Hyperledger Composer Playground!

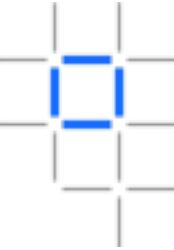


In this web sandbox, you can deploy, edit and test business network definitions. Have a play and learn what Hyperledger Composer Playground is all about.

Let's Blockchain!



Not sure where to start? View our Playground tutorial.



Hyperledger Composer Playground

My Business Networks

Connection: Web Browser

The screenshot shows the Hyperledger Composer Playground interface. It features a sidebar on the left with a globe icon and Japanese text "こんにちは". Below it, the text "Hello, Composer!" is displayed. A call-to-action button "Get Started →" is at the bottom. The main content area shows a "basic-sample-network" card with a plus sign icon and the text "Deploy a new business network".

Hello, Composer!

Get started with the basic-sample-network, or view our [Playground tutorial](#)

BUSINESS NETWORK

basic-sample-network

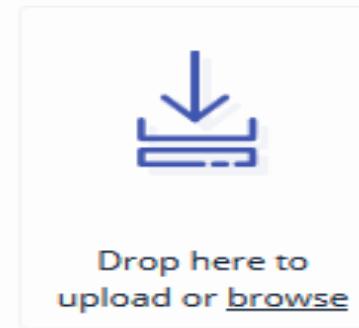
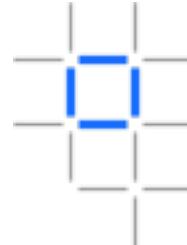
Get Started →

The screenshot shows the Hyperledger Composer Playground interface. It features a sidebar on the left with a globe icon and Japanese text "こんにちは". Below it, the text "Hello, Composer!" is displayed. A call-to-action button "Get Started →" is at the bottom. The main content area shows a "basic-sample-network" card with a plus sign icon and the text "Deploy a new business network".

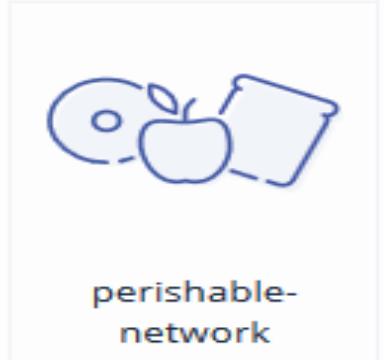
Deploy a new business network

Choose a Business Network Definition to start with:

Choose a sample to play with, start a new project, or import your previous work



Samples on npm



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`

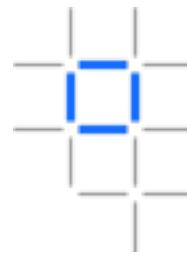
Transaction `SampleTransaction`

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:

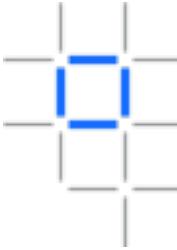
Hyperledger Composer Modeling Language



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

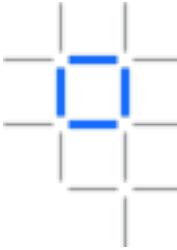
1. A single namespace. All resource declarations within the file are implicitly in this namespace.
2. A set of resource definitions, encompassing assets, transactions, participants, and events.
3. Optional import declarations that import resources from other namespaces.

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



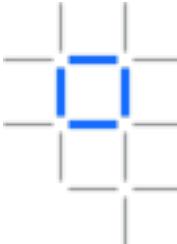
Model File models/sample.cto

```
13  /*
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
```

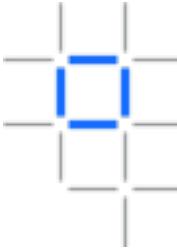


Script File

```
13  /**
14
15  /* global getAssetRegistry getFactory emit */
16
17  /**
18   * Sample transaction processor function.
19   * @param {org.example.basic.SampleTransaction} tx The sample transaction instance.
20   * @transaction
21  */
22  async function sampleTransaction(tx) { // eslint-disable-line no-unused-vars
23
24      // Save the old value of the asset.
25      const oldValue = tx.asset.value;
26
27      // Update the asset with the new value.
28      tx.asset.value = tx.newValue;
29
30      // Get the asset registry for the asset.
31      const assetRegistry = await getAssetRegistry('org.example.basic.SampleAsset');
32      // Update the asset in the asset registry.
33      await assetRegistry.update(tx.asset);
34
35      // Emit an event for the modified asset.
36      let event = getFactory().newEvent('org.example.basic', 'SampleEvent');
37      event.asset = tx.asset;
38      event.oldValue = oldValue;
39      event.newValue = tx.newValue;
40      emit(event);
41  }
42 }
```



```
* 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
}
```

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.

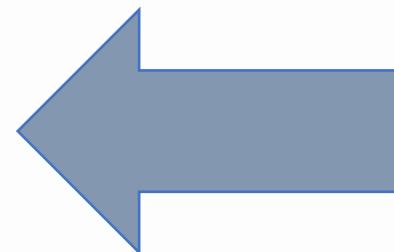
This business network defines:

Participant SampleParticipant

Asset SampleAsset

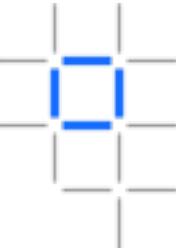
Transaction SampleTransaction

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:



Create a `SampleParticipant` participant:

```
{  
  "$class": "org.example.basic.SampleParticipant",  
  "participantId": "Toby",  
  "firstName": "Tobias",  
  "lastName": "Hunter"  
}
```

Create a `SampleAsset` asset:

```
{  
  "$class": "org.example.basic.SampleAsset",  
  "assetId": "assetId:1",  
  "owner": "resource:org.example.basic.SampleParticipant#Toby",  
  "value": "original value"  
}
```

Submit a `SampleTransaction` transaction:

```
{  
  "$class": "org.example.basic.SampleTransaction",  
  "asset": "resource:org.example.basic.SampleAsset#assetId:1",  
  "newValue": "new value"  
}
```

After submitting this transaction, you should now see the transaction in the Transaction Registry and that a `SampleEvent` has been emitted. As a result, the value of the `assetId:1` should now be `new value` in the Asset Registry.

Participant registry for org.example.basic.SampleParticipant

[+ Create New Participant](#)

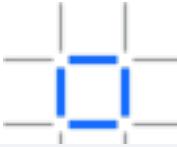
ID

Data

Toby

```
{  
    "$class": "org.example.basic.SampleParticipant",  
    "participantId": "Toby",  
    "firstName": "Tobias",  
    "lastName": "Hunter"  
}
```





Asset registry for org.example.basic.SampleAsset

+ Create New Asset

ID

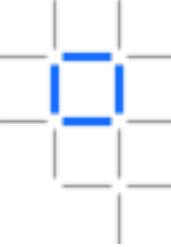
Data

assetId:1

```
{  
    "$class": "org.example.basic.SampleAsset",  
    "assetId": "assetId:1",  
    "owner": "resource:org.example.basic.SampleParticipant#Toby",  
    "value": "original value"  
}
```



Submit Transaction



Transaction Type

SampleTransaction

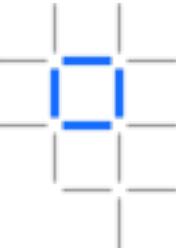


JSON Data Preview

```
1  {
2    "$class": "org.example.basic.SampleTransaction",
3    "asset": "resource:org.example.basic.SampleAsset#assetId:1",
4    "newValue": "new value"
5 }
```



Optional Properties



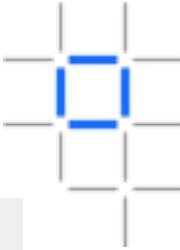
Historian Record

X

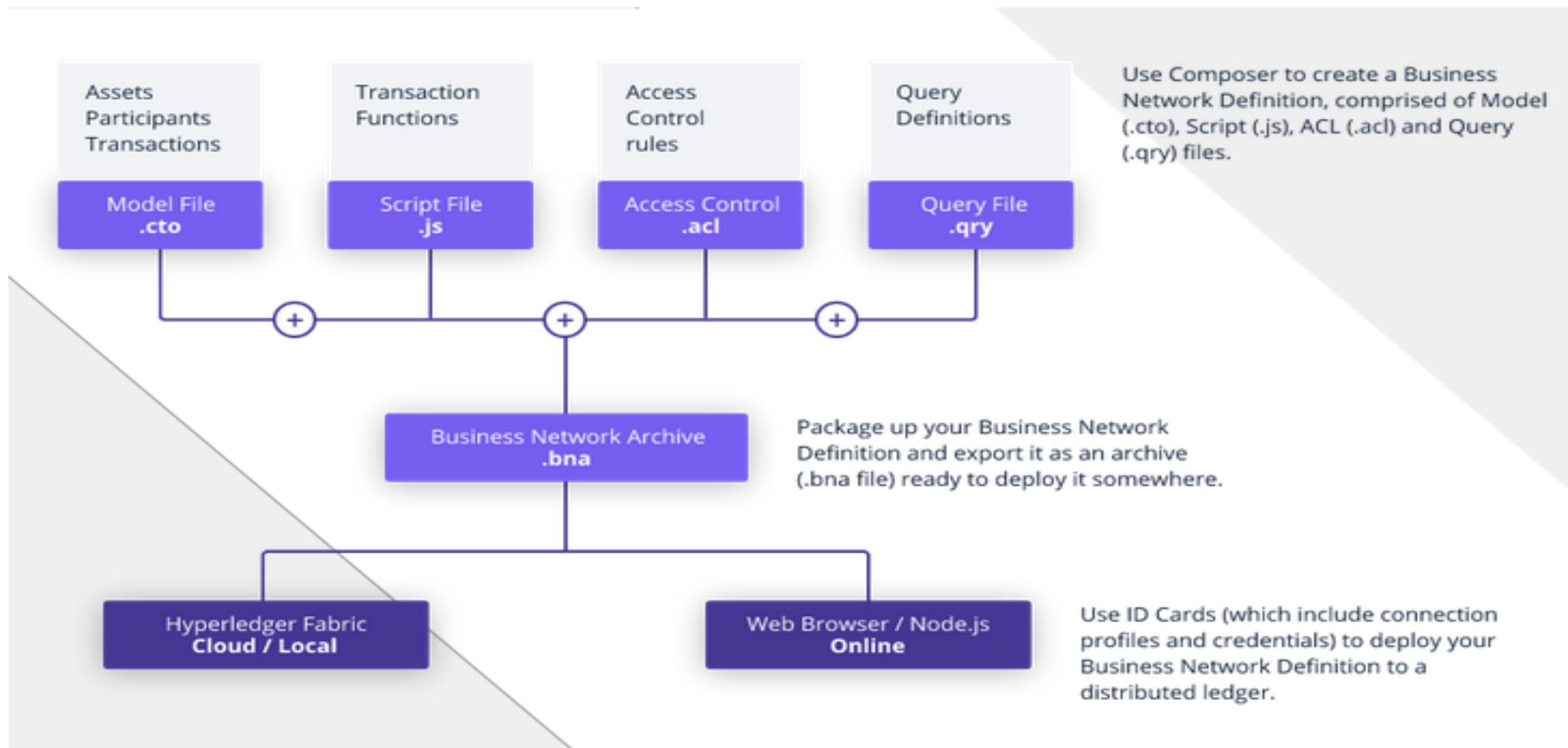
Transaction

Events (1)

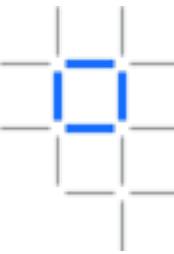
```
1  {
2    "$class": "org.example.basic.SampleTransaction",
3    "asset": "resource:org.example.basic.SampleAsset#assetId:1",
4    "newValue": "new value",
5    "transactionId": "a100b354-4ae3-47c5-a341-d255a724fdee",
6    "timestamp": "2018-06-23T19:26:21.501Z"
7 }
```



Deploying the BNA file to Hyperledger Fabric and the Cloud



Extensive, Familiar, Open Development Toolset

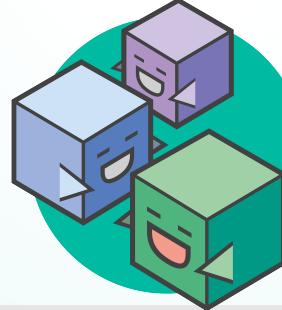


```
asset Animal identi  
  o String animal  
  o AnimalType sp  
  o MovementStatus  
  o ProductionTyp
```

Data modelling



JavaScript
business logic



Web playground

composer-client
composer-admin



Client libraries



Editor support

\$ composer

CLI utilities



Code generation



Swagger

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

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>

5.3B) 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 not.



Business Network Archive files

- CTO Model file
- JavaScript
- ACL

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

```
/**  
 * 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  
}
```

Copy

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 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 – thank you Ruby on
Rails!**

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

Hyperledger Composer REST server

Commodity : An asset named Commodity

Show/Hide | List Operations | Expand Operations

GET	/Commodity	Find all instances of the model matched by filter from the data source.
POST	/Commodity	Create a new instance of the model and persist it into the data source.
GET	/Commodity/{id}	Find a model instance by {{id}} from the data source.
HEAD	/Commodity/{id}	Check whether a model instance exists in the data source.
PUT	/Commodity/{id}	Replace attributes for a model instance and persist it into the data source.
DELETE	/Commodity/{id}	Delete a model instance by {{id}} from the data source.

System : General business network methods

Show/Hide | List Operations | Expand Operations

Trade : A transaction named Trade

Show/Hide | List Operations | Expand Operations

Trader : A participant named Trader

Show/Hide | List Operations | Expand Operations

[BASE URL: /api , API VERSION: 0.0.1]

Generate the Angular Application

Step Six: Generating an application

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

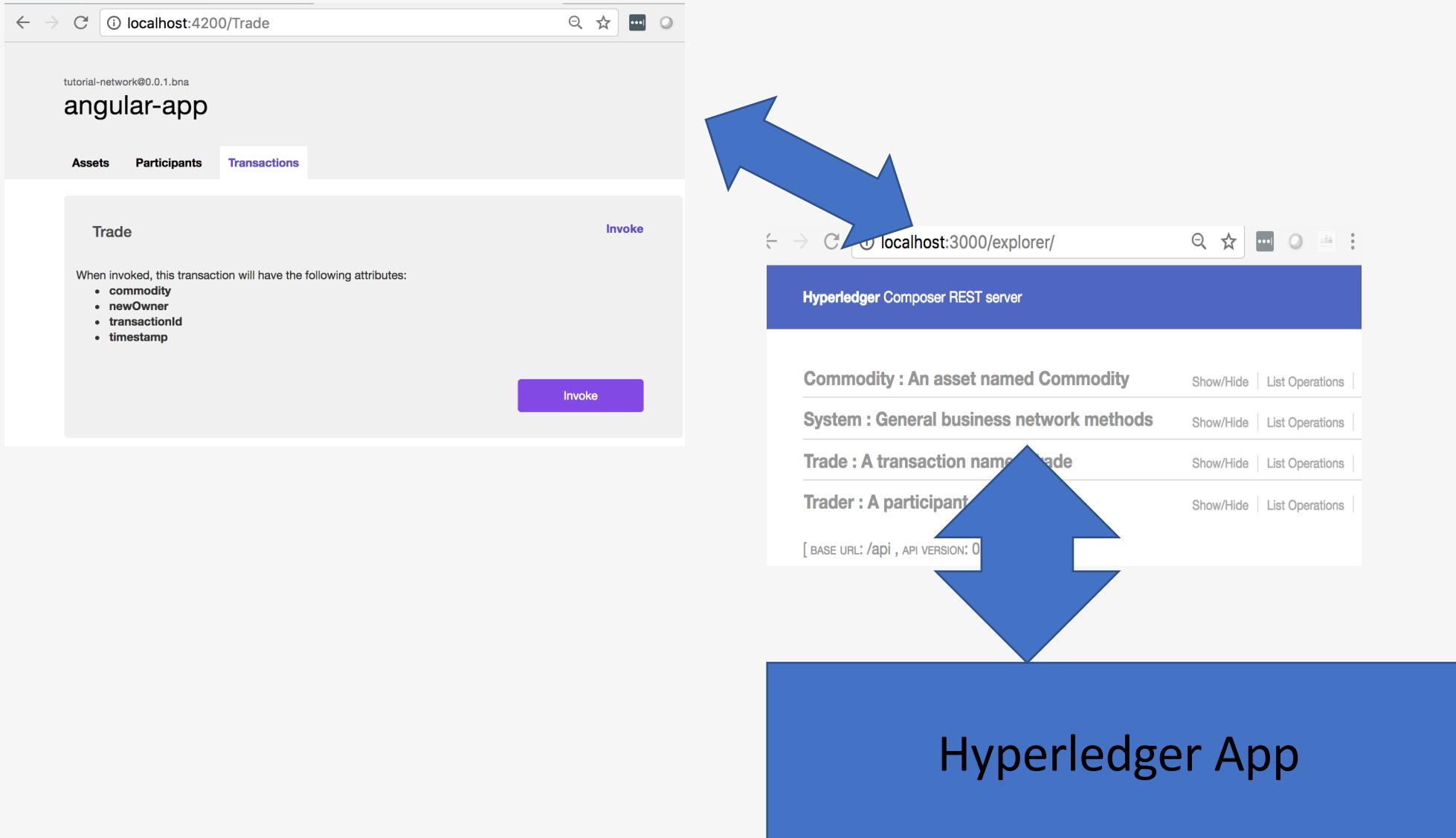
1. To create your Angular 4 application, navigate to `tutorial-network` directory and run the following command:

```
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

tutorial-network@0.0.1.bna

angular-app

Assets Participants Transactions

Commodity

+ | Create Asset

tradingSymbol	description	mainExchange	quantity	owner	Actions
ABC	Test Commodity	Euronext	72.297	resource:org.example.mynetwork.T...	 

Angular App 2

tutorial-network@0.0.1.bna

angular-app

Assets Participants Transactions

Trader

+ | Create Participant

tradId	firstName	lastName	Actions
TRADER1	Jenny	Jones	 
TRADER2	Amy	Williams	 

Angular App 3

tutorial-network@0.0.1.bna

angular-app

Assets Participants

Transactions

Trade

Invoke

When invoked, this transaction will have the following attributes:

- commodity
- newOwner
- transactionId
- timestamp

Invoke

Fixing Transaction issue in Angular for Hyperledger fabric blockchain application

Create Transaction

Enter the required values below.

product (C)

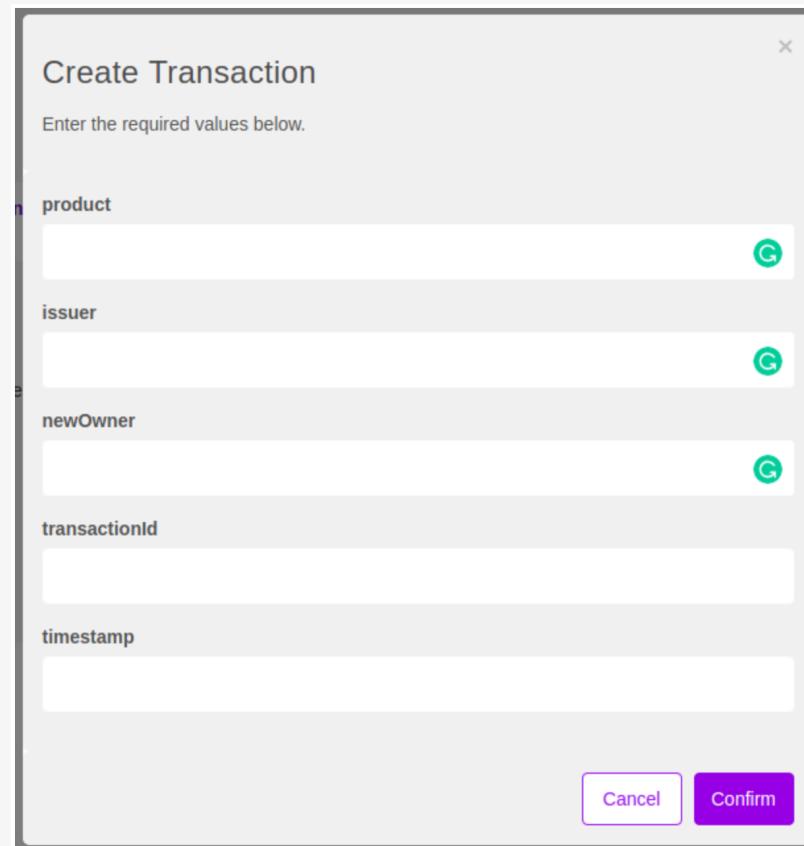
issuer (C)

newOwner (C)

transactionId

timestamp

Cancel Confirm



<https://medium.com/coinmonks/fixing-transaction-issue-in-angular-for-hyperledger-fabric-blockchain-application-fe7e28a7bb6e>

Fixing Transaction issue in Angular for Hyperledger fabric blockchain application

[Yashwanth Madaka](#)

<https://medium.com/coinmonks/fixing-transaction-issue-in-angular-for-hyperledger-fabric-blockchain-application-fe7e28a7bb6e>

Building a blockchain application using Hyperledger Fabric with Angular Frontend: Part-2

<https://medium.com/coinmonks/building-a-blockchain-application-using-hyperledger-fabric-with-angular-frontend-part-2-22ef7c77f53>

<https://medium.freecodecamp.org/ultimate-end-to-end-tutorial-to-create-an-application-on-blockchain-using-hyperledger-3a83a80cbc71>

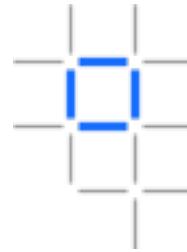
The business network and complete applications



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

Step 4: Deploy Hyperledger apps in the cloud

How do we host our Blockchain application?



IBM Cloud Catalog

 blockchain

Filter

Platform

Blockchain

The service enables the creation of blockchain business networks with ownership and control distributed across different organizations.

Blockchain



Utilize IBM's Blockchain
Technology within IBM Cloud

IBM

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](#)

MY NETWORK

Overview

Members

Channels

Notifications

APIs

MY CODE

Develop code

Install code

Try samples

[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

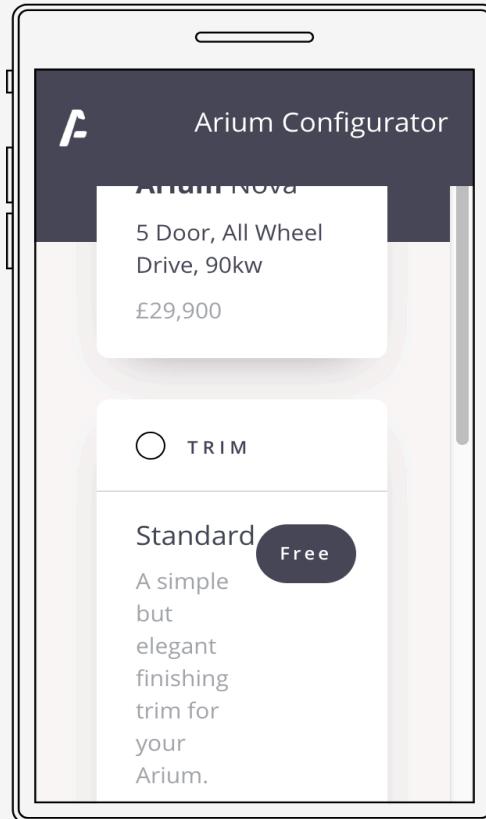
Channels



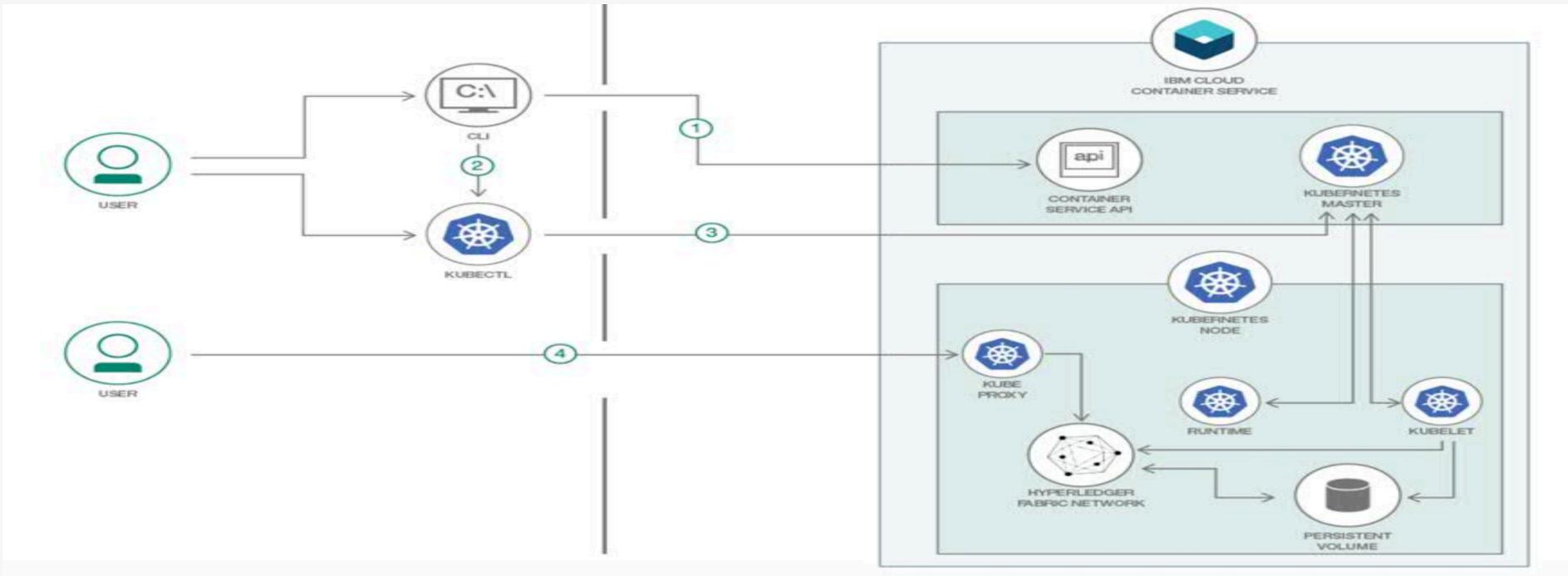
Search Channels

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

Run Blockchain apps in the IBM Cloud



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.

Thank You!

alf@us.ibm.com

<https://hyperledger.org>

<https://ibm.com/blockchain>



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