HYPERLEDGER
FABRIC

Workshop Hyperledger Fabric

Sergio.torres@blocknitive.com

Sergio.torres@blocknitive.com

SOMOS TECHS. SOMOS BUSINESS.

Acercamos la tecnología Blockchain al negocio

Blockchain

1

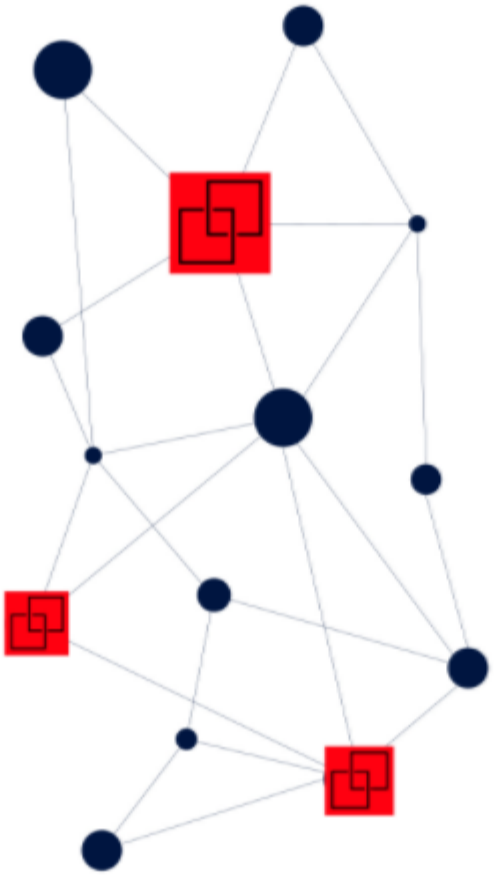
DESCENTRALIZACIÓN

Es un sistema descentralizado. Cada parte puede validar los registros de sus socios de transacción sin un intermediario.

2

SEGURIDAD

Los registros no se pueden modificar, ya que cada uno de ellos están vinculado al anterior. Se utilizan diversos algoritmos y enfoques computacionales para asegurar la permanencia, disponibilidad y protección.



3

PRIVACIDAD

Cada transacción y su valor son visibles para cualquier persona con acceso al sistema. Los usuarios pueden elegir permanecer en el anonimato o proporcionar una prueba de su identidad a otros. Las transacciones ocurren entre las direcciones de bloques.

4

LÓGICA DE NEGOCIO

La lógica de los negocios es fácilmente trasladable a la tecnología blockchain, permitiendo una reducción de costes y la ya mencionada descentralización,seguridad y privacidad

El propósito de Blocknitive es conseguir incrementar los resultados de negocio a través de su integración con la tecnología Blockchain.

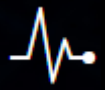
IDENTIFICACIÓN & IDEACIÓN

Apoyamos equipos técnicos en la adquisición de conocimiento de tecnologías Blockchain con el fin de identificar e idear nuevos casos de uso aplicados al negocio a través de la unión de perspectivas de negocio y tecnología.

CONSULTORÍA

Ayudamos a crear y definir arquitecturas junto con otras empresas.

Apoyamos equipos técnicos desde una perspectiva objetiva para el acompañamiento en el desarrollo de proyectos.



DESARROLLO PROYECTOS

Aportamos experiencia en el desarrollo tecnológico de casos de uso a través de un equipo cualificado compuesto de diferentes roles. Nuestra experiencia en el sector tecnológico nos avala.

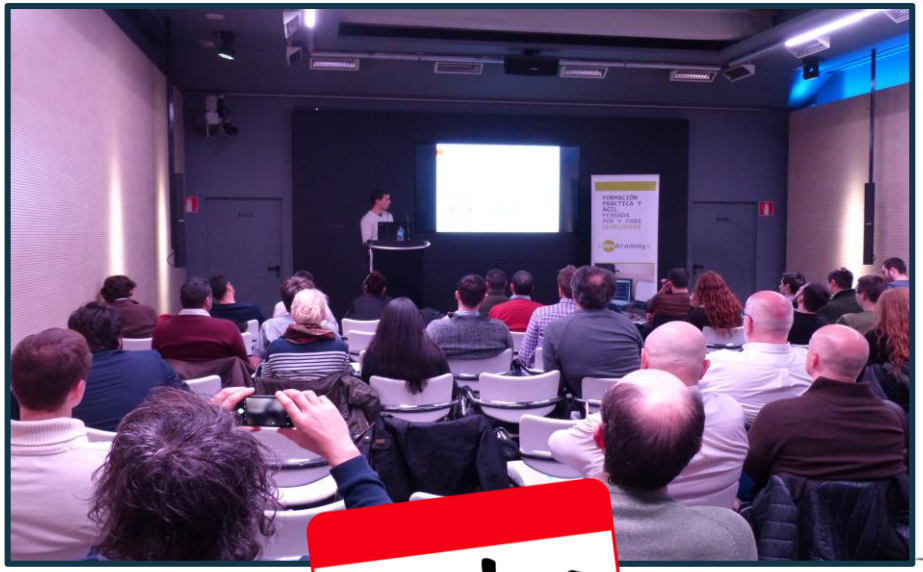
FORMACIÓN

Preparamos equipos técnicos con el fin de tranmitir conocimientos de desarrollo de proyectos Blockchain y adquirir skills en el desarrollo y utilización de tecnologías del ecosistema, despliegue de nodos, creación de redes, desarrollo de SmartContracts, etc.

SOMOS TECHS. SOMOS BUSINESS.

Comunidad Blocknitive

Comunidad blockchain

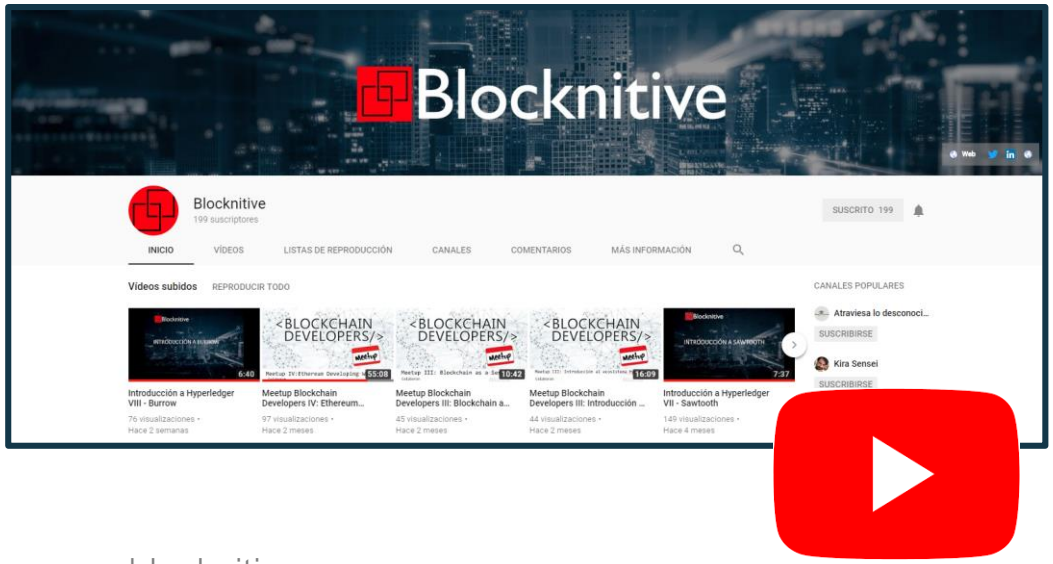


Meetup +670 miembros
Youtube 200 subscriptores
Github proyectos públicos
Blog 2000 usuarios/mes
Telegram +600 usuarios

Con la colaboración de

IBM

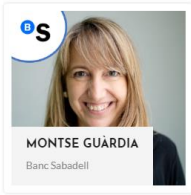
Telefónica

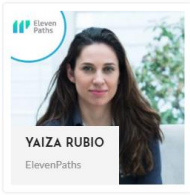


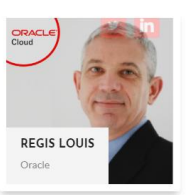
GitHub

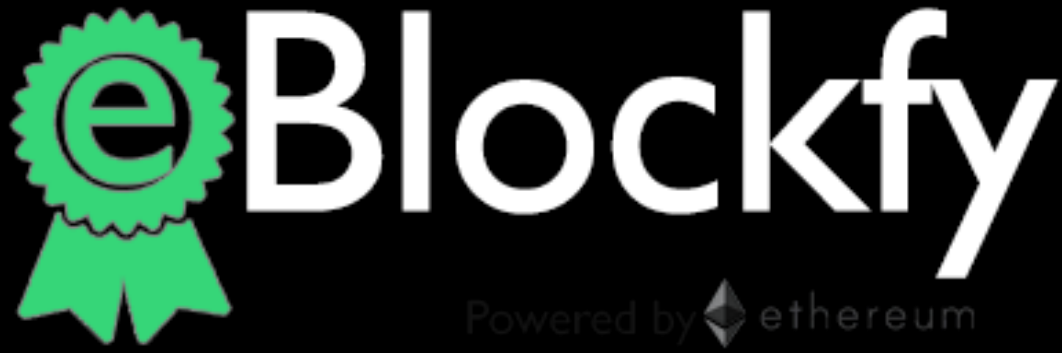
Evento BlockchainCon

+500 Inscripciones - Workshops técnicos - Empresas colaboradoras a nivel internacional





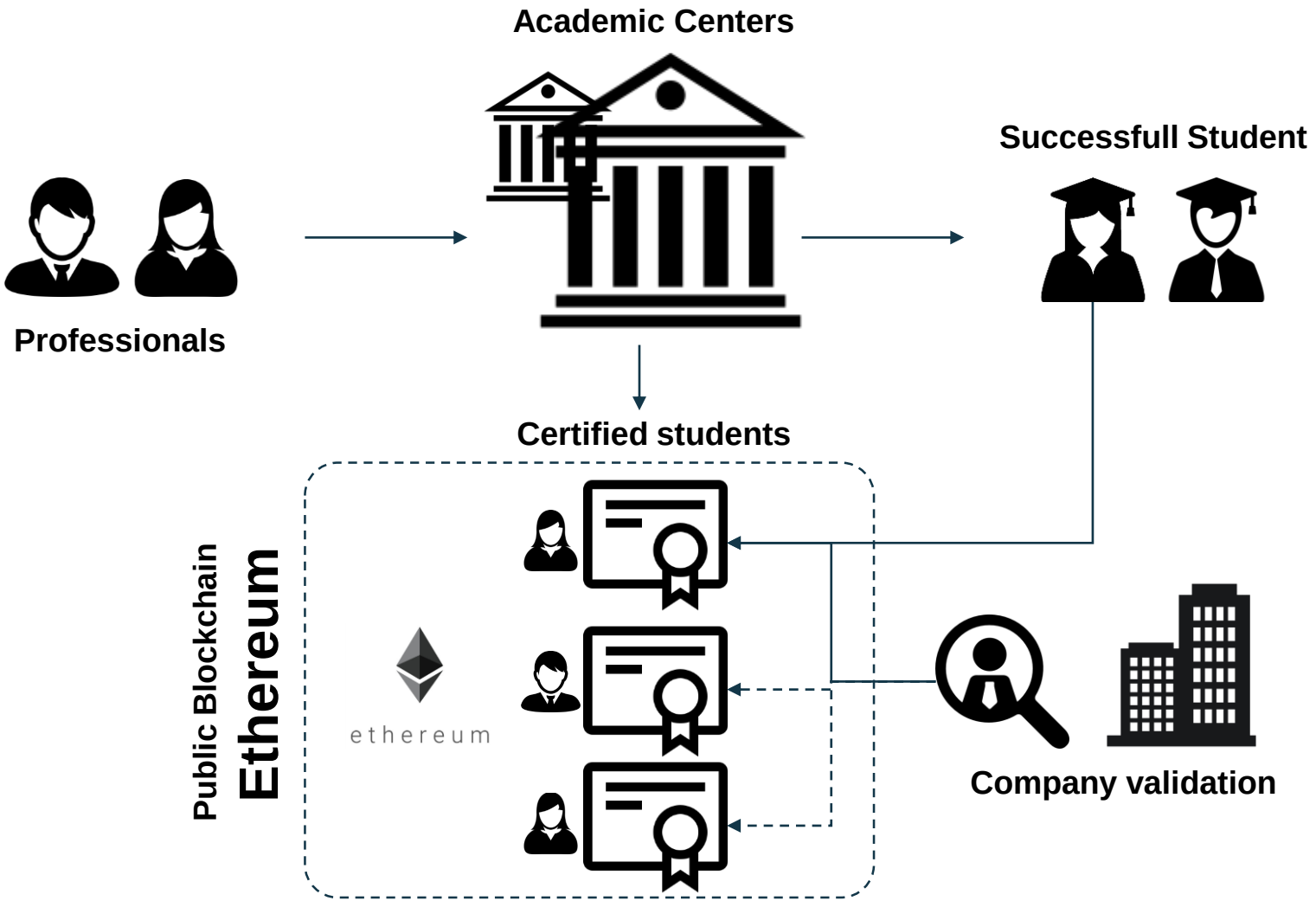



eBlockfy
Powered by ethereum

Aplicación

Aplicación de Blockchain al caso de uso

eBlockfy
Powered by ethereum

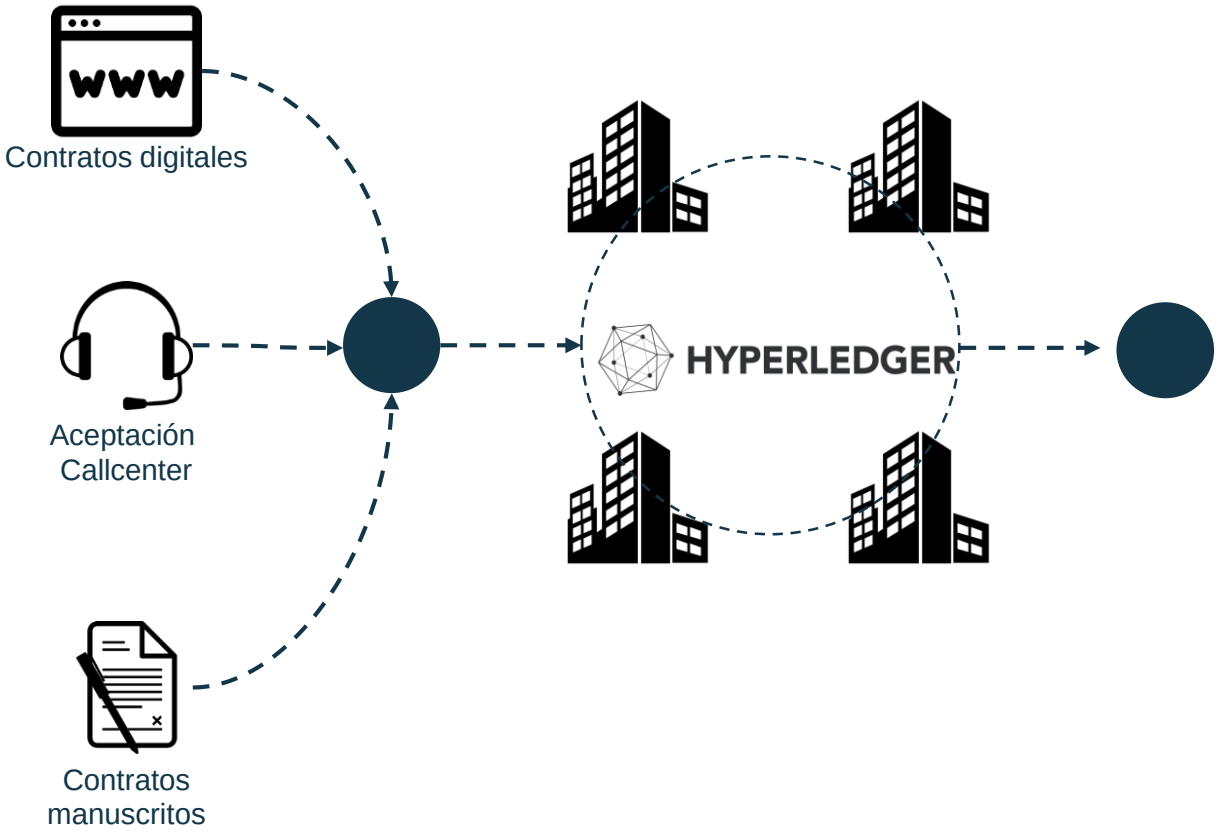



Asentify
Powered by
HYPERLEDGER

Fuentes de integración

Tipos de integraciones

Asentify
Powered by HYPERLEDGER

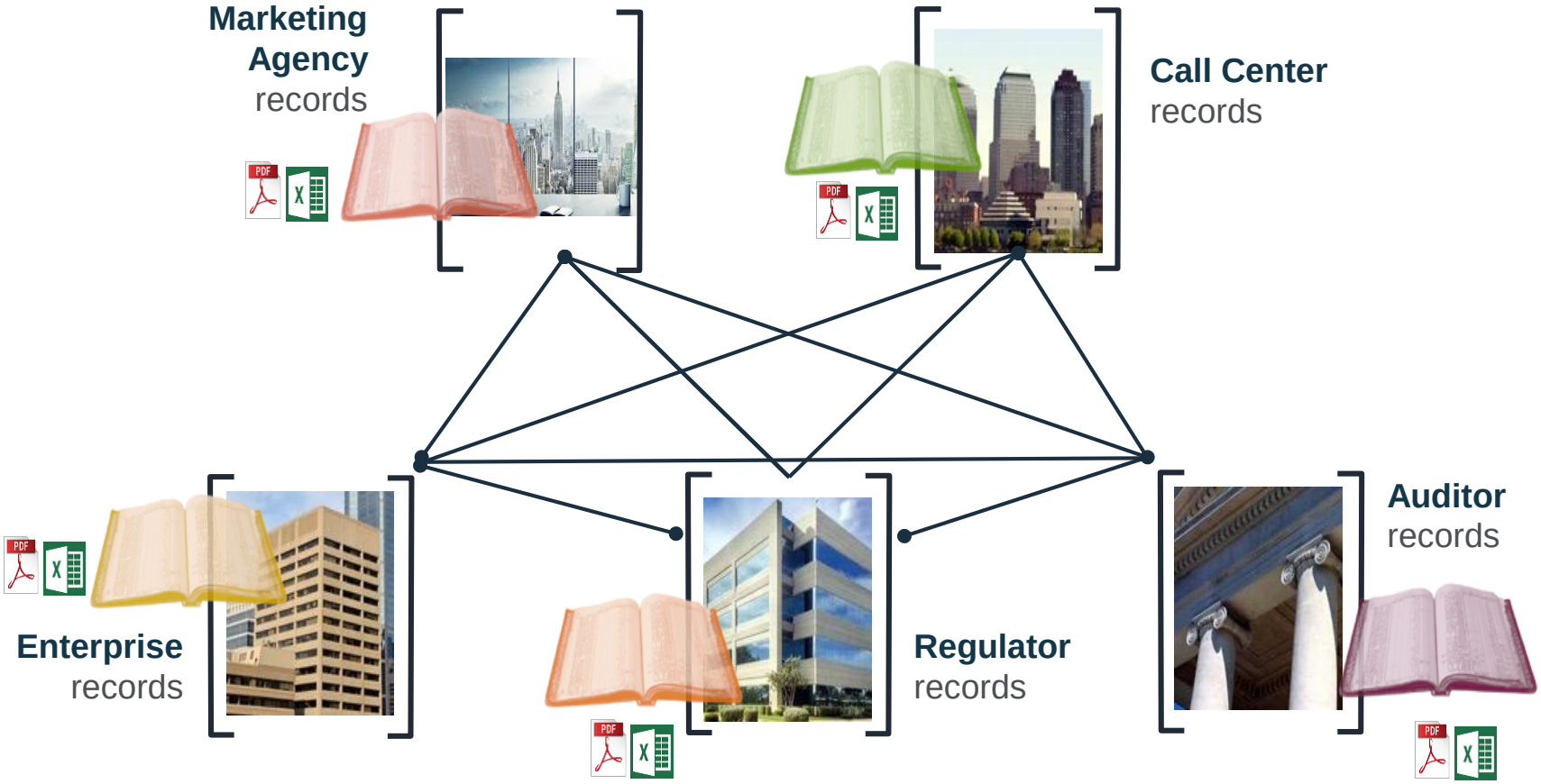


- ✓Analytics y supervisión
- ✓Impacto regulación
- ✓Trazabilidad de dato

< / Comencemos >

Compartir información entre todos

¿Por qué Blockchain?



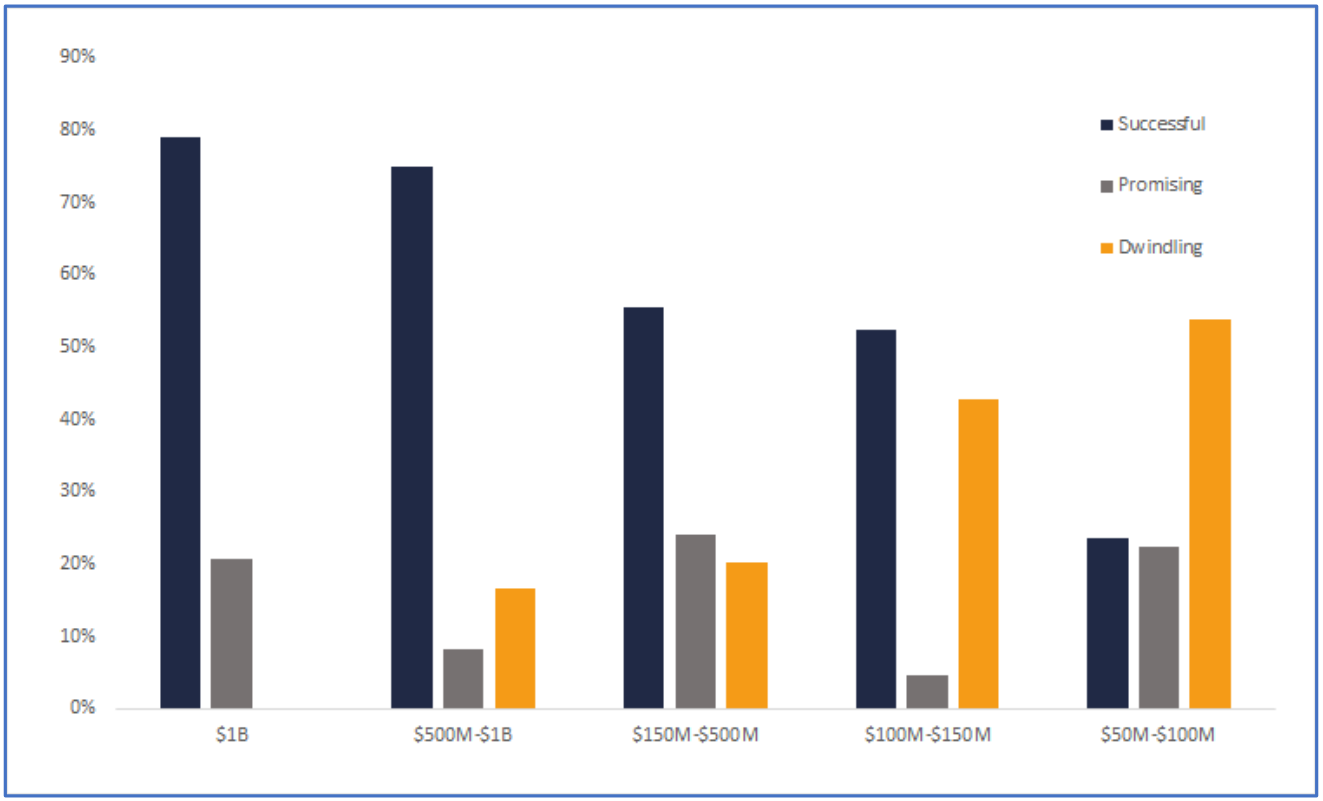
¿Por qué Blockchain?

Blockchain, una idea que nació para proteger la transacción de dinero entre particulares dentro de la red.

Es una base distribuida en una red descentralizada,
con un historial de transaciones producido e inmodificable

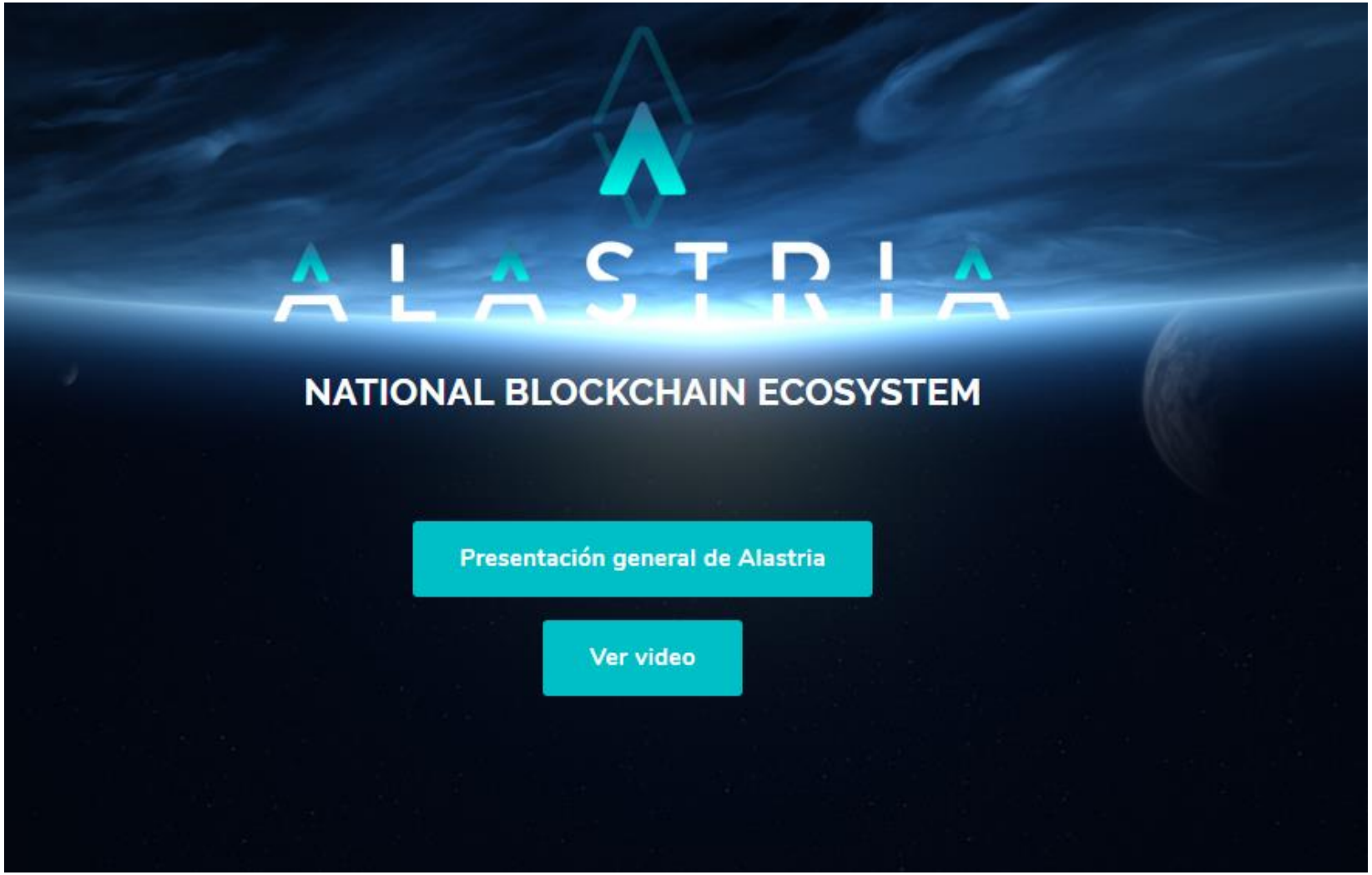
Actualidad Blockchain

New Study: 80% of ICOs are Scams, Only 8% Reach an Exchange



Actualidad Blockchain

Actualidad Blockchain



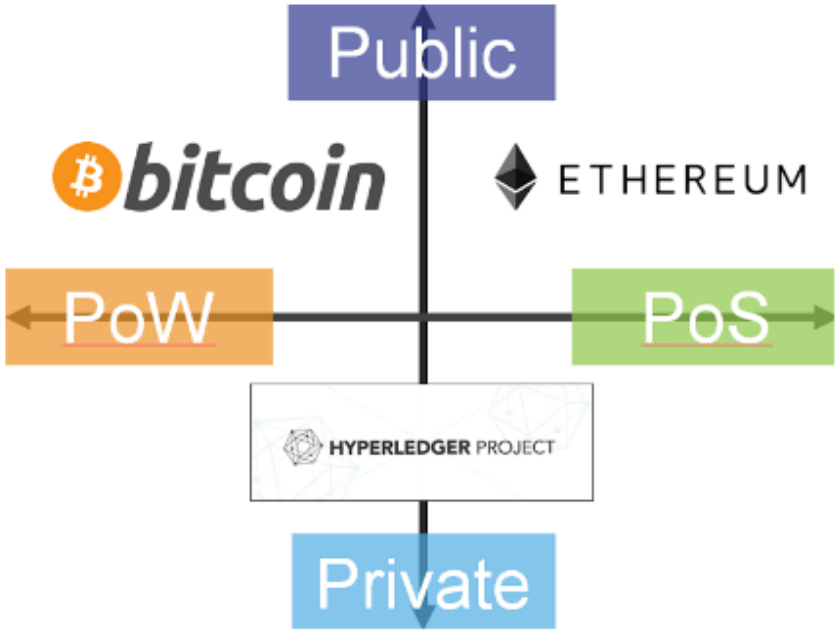
Tipos de Blockchain

PÚBLICA

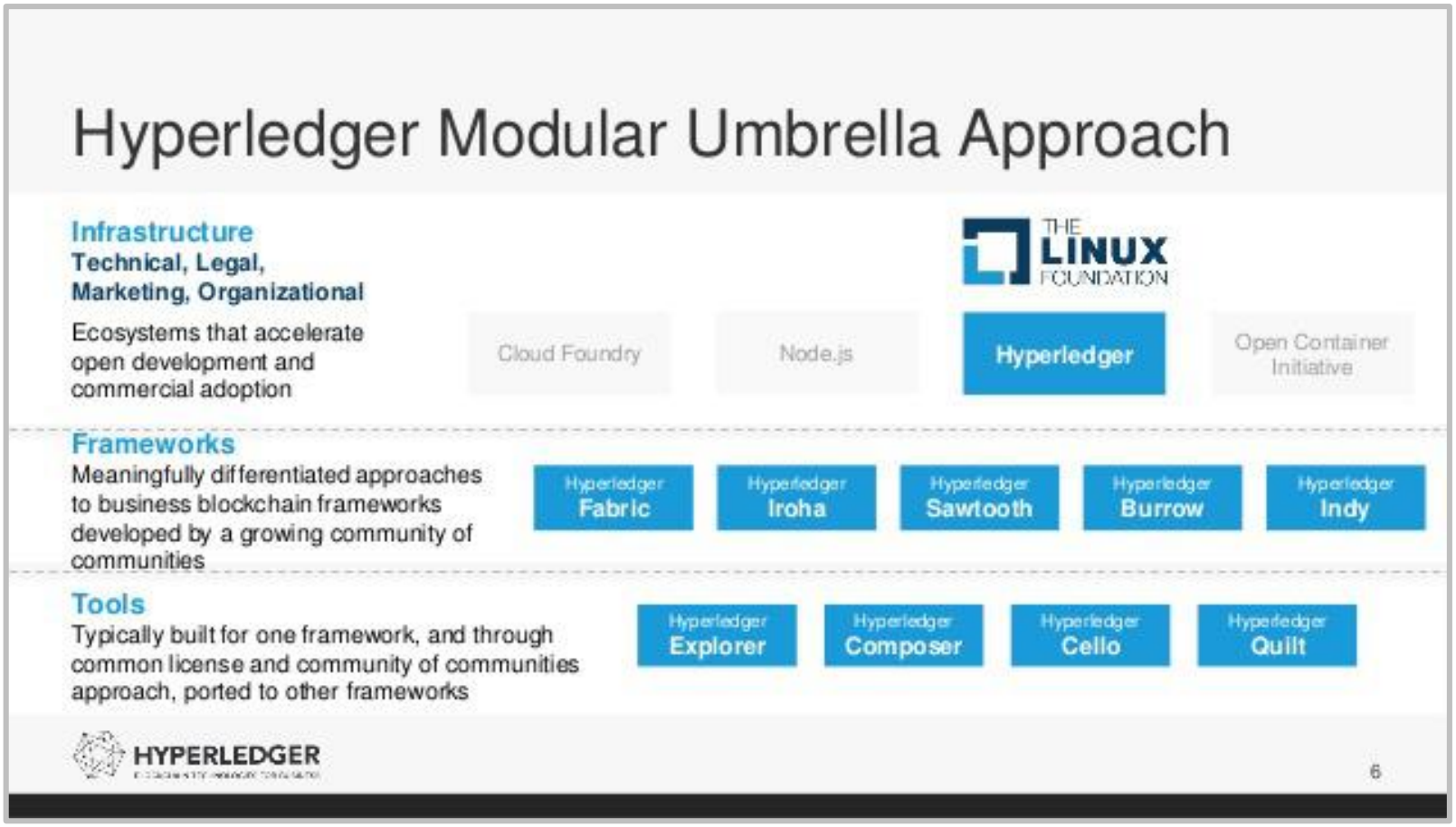
- SIN RESTRICCIONES
- ANONIMATO

PRIVADA

- SE LIMITAN LAS OPERACIONES
- PARA USUARIOS CON ACCESO



Universo Hyperledger



www.hyperledger.org/projects

Hyperledger Frameworks

Hyperledger Frameworks

Hyperledger Sawtooth is a modular platform for building, deploying, and running distributed ledgers. Hyperledger Sawtooth includes a novel consensus algorithm, Proof of Elapsed Time (PoET), which targets large distributed validator populations with minimal resource consumption.

» LEARN MORE

Hyperledger Iroha is a business blockchain framework designed to be simple and easy to incorporate into infrastructural projects requiring distributed ledger technology.

» LEARN MORE

Intended as a foundation for developing applications or solutions with a modular architecture, Hyperledger Fabric allows components, such as consensus and membership services, to be plug-and-play.

» LEARN MORE

Hyperledger Burrow is a permissionable smart contract machine. The first of its kind when released in December, 2014, Burrow provides a modular blockchain client with a permissioned smart contract interpreter built in part to the specification of the Ethereum Virtual Machine (EVM).

» LEARN MORE

Hyperledger Indy is a distributed ledger, purpose-built for decentralized identity. It provides tools, libraries, and reusable components for creating and using independent digital identities rooted on blockchains or other distributed ledgers for interoperability.

» LEARN MORE

Hyperledger Fabric

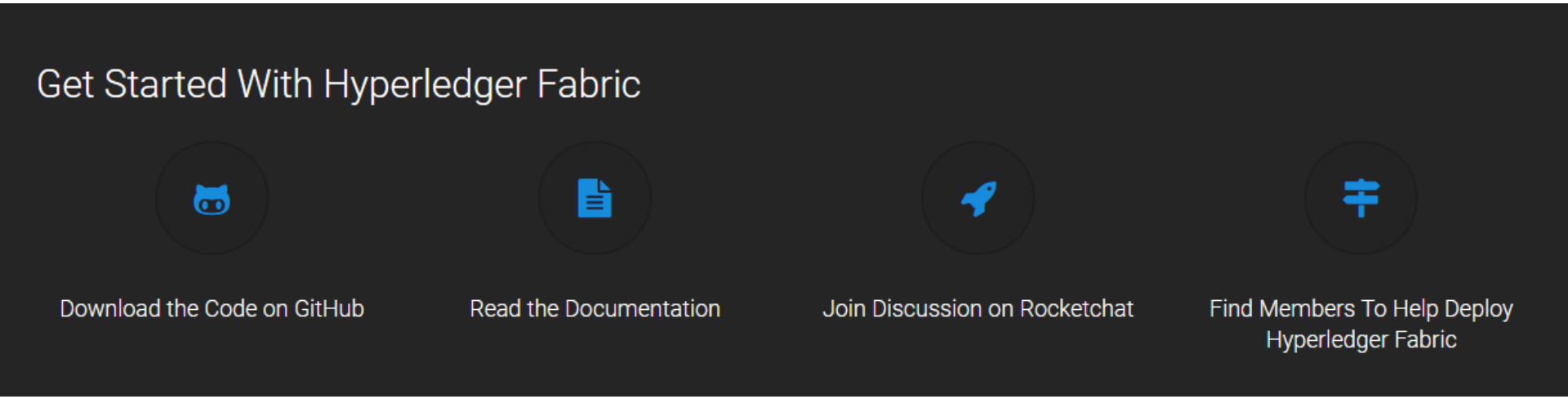
HYPERLEDGER FABRIC

Hyperledger Fabric fue la primera propuesta para una base de código, que combina el trabajo previo realizado por Digital Asset Holdings, el libconsensus de Blockstream y OpenBlockchain de IBM.

Arquitectura modular, que permite que los componentes como el consenso y los servicios de membresía sean plug-and-play.

Hyperledger Fabric es revolucionario al permitir que las entidades realicen transacciones confidenciales sin pasar información a través de una autoridad central.

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



Hyperledger Iroha

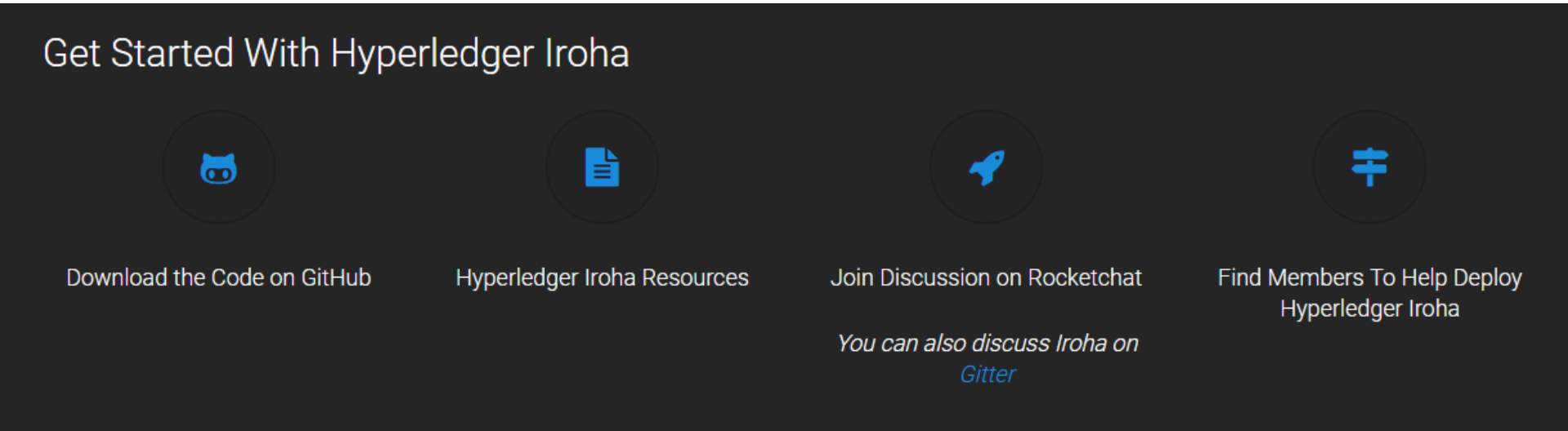
Hyperledger Iroha es un Framework de blockchain aportado por Soramitsu, Hitachi, NTT Data y Colu.

Hyperledger Iroha está diseñado para ser simple y fácil de incorporar en proyectos de infraestructura que requieren tecnología ledger distribuida.

Hyperledger Iroha hace hincapié en el desarrollo de aplicaciones móviles con bibliotecas cliente para Android e iOS, lo que lo diferencia de otros marcos de Hyperledger.

Escrito en C++
10+ contribuidores
1k+ commits
Sin caso de uso exitoso

https://hyperledger.org/projects/iroha



Hyperledger Sawtooth

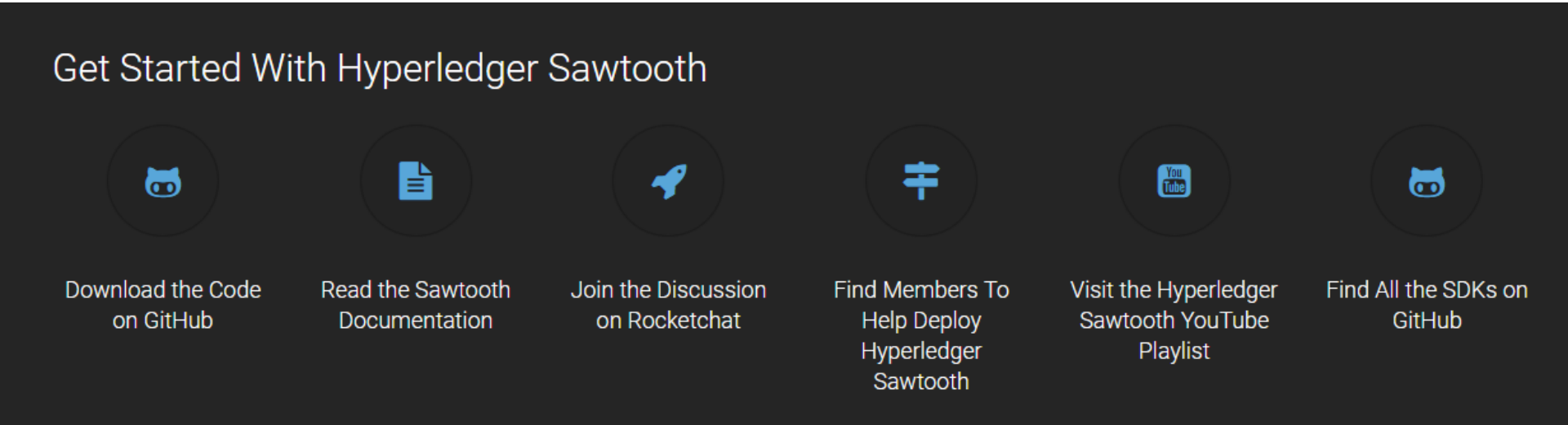
Hyperledger Sawtooth, **contribuido por Intel**, es un framework de blockchain que utiliza una plataforma modular para construir, desplegar y ejecutar ledgers distribuidos.

Las soluciones de contabilidad distribuidas construidas con Hyperledger Sawtooth pueden utilizar varios algoritmos de consenso basados en el tamaño de la red.

De forma predeterminada, utiliza el **algoritmo de consenso** Prueba de tiempo transcurrido (Proof of Elapsed Time - PoET), que proporciona la escalabilidad de la cadena de bloques de Bitcoin sin el alto consumo de energía.

- Escrito en Python
- 20+ contribuidores
- 2k+ commits
- Sin casos de uso exitosos

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



Hyperledger Burrow

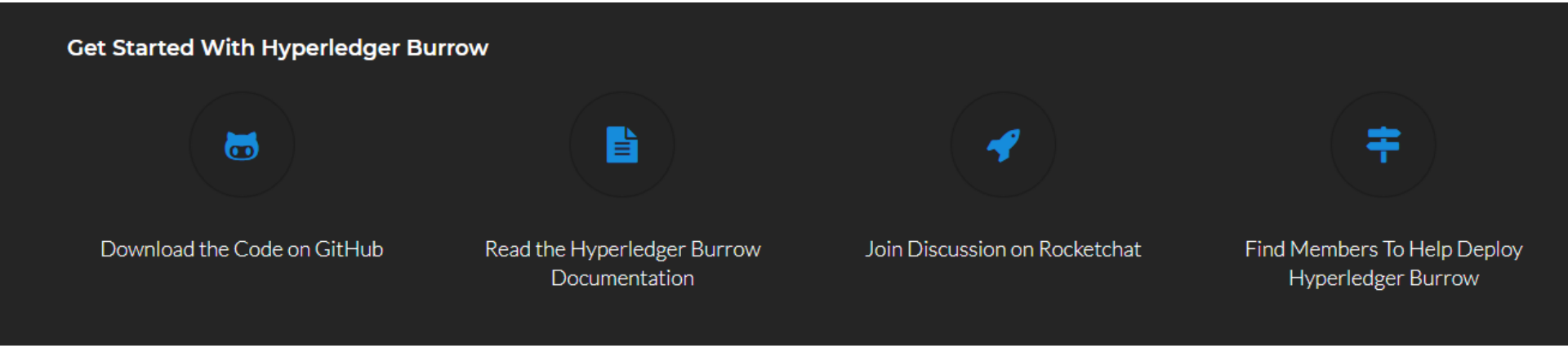
Hyperledger Burrow, **antes conocido como Monax**, es un framework de blockchain que utiliza una plataforma modular para construir, desplegar y ejecutar ledgers distribuidos.

Hyperledger Burrow proporciona un proyecto modular de blockchain con un intérprete de contratos inteligentes permisionado y desarrollado para la especificación de la máquina virtual Ethereum (EVM).

Alta velocidad de minado y los Smart Contracts pueden ejecutarse con Solidity.

Escrito en Go
8 contribuidores
1,4k+ commits
Sin casos de uso exitosos

https://hyperledger.org/projects/hyperledger-burrow

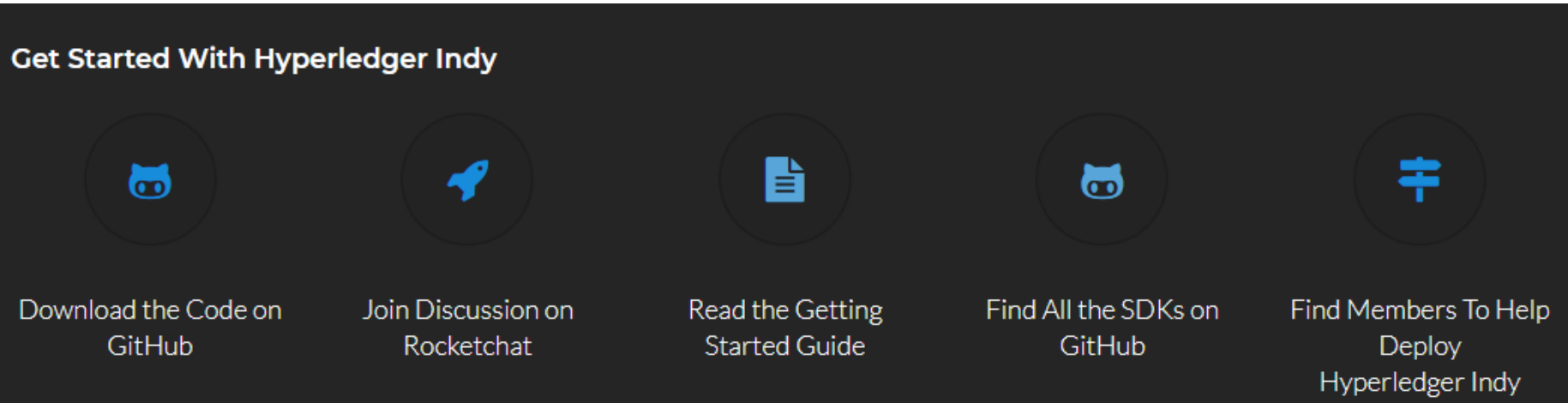


Hyperledger Indy

Hyperledger Indy es un ledger distribuido especialmente diseñado para la identidad descentralizada. El objetivo de Hyperledger Indy es lograr esto desarrollando un conjunto de *""(...) **características descentralizadas de identidad** y artefactos que son independientes de cualquier libro particular y permitirán la interoperabilidad a través de cualquier DLT que los soporte"*

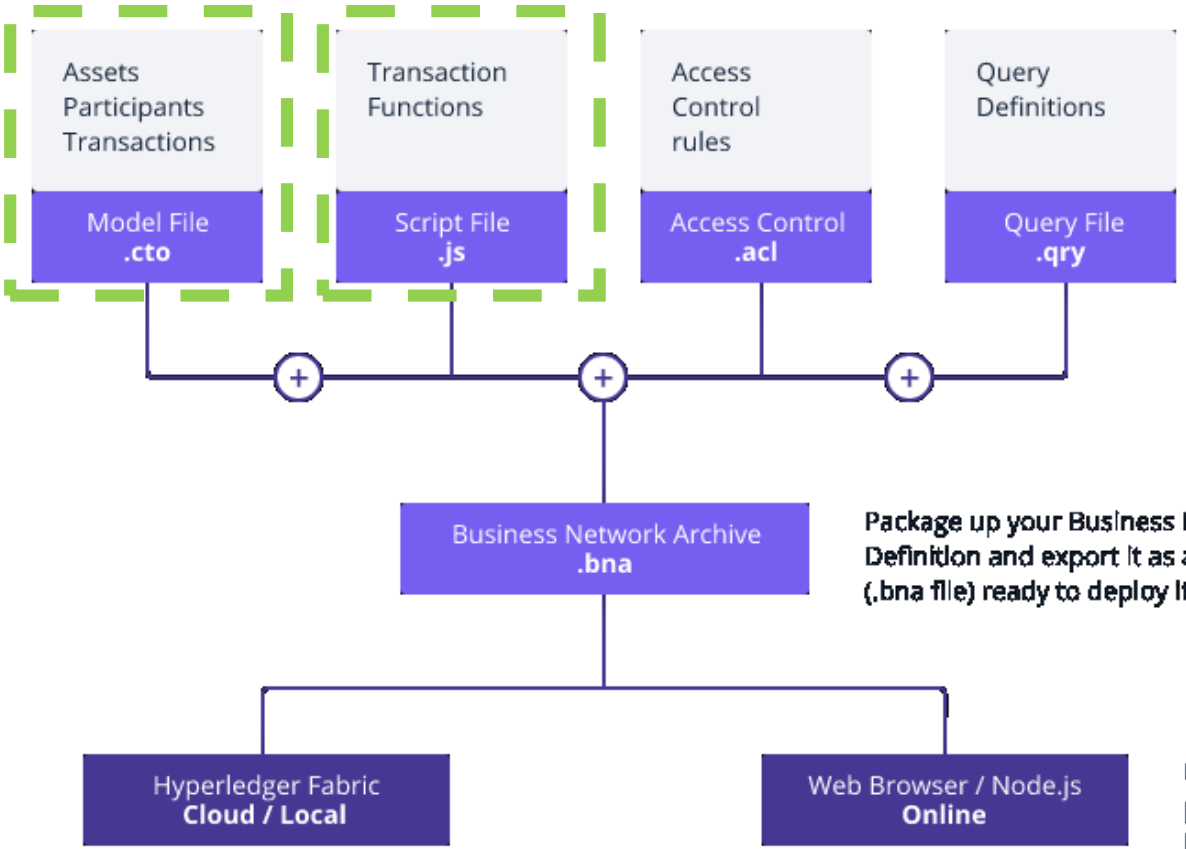
- Hyperledger Indy permite a las personas administrar y controlar sus identidades digitales.
- Contribuido por la Fundación Sovrin
- Hyperledger Indy les permite a las empresas almacenar indicadores de identidad, en lugar de que las empresas almacenen grandes cantidades de datos personales de personas

https://www.hyperledger.org/projects/hyperledger-indy



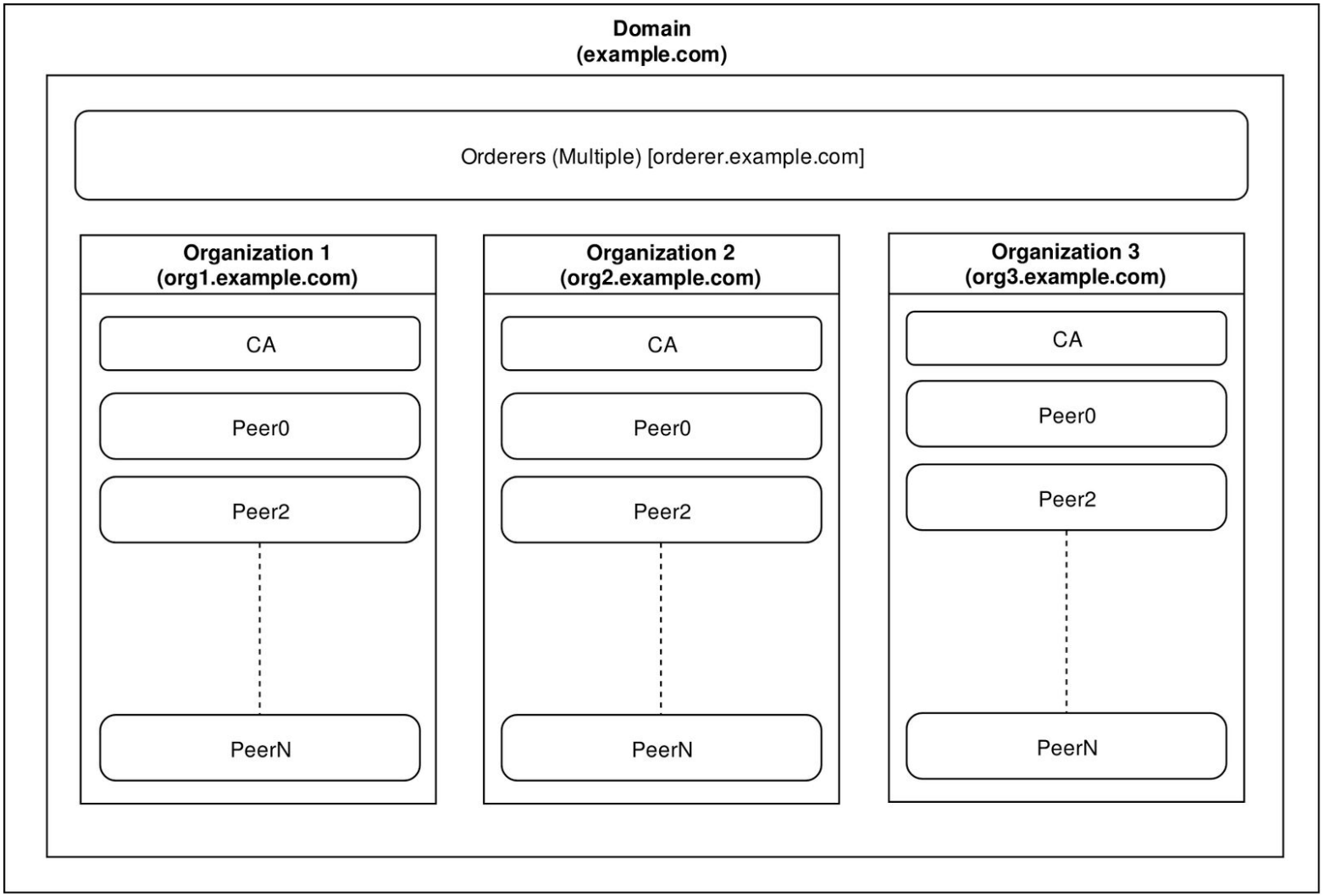
< Composer >

Hyperledger Composer

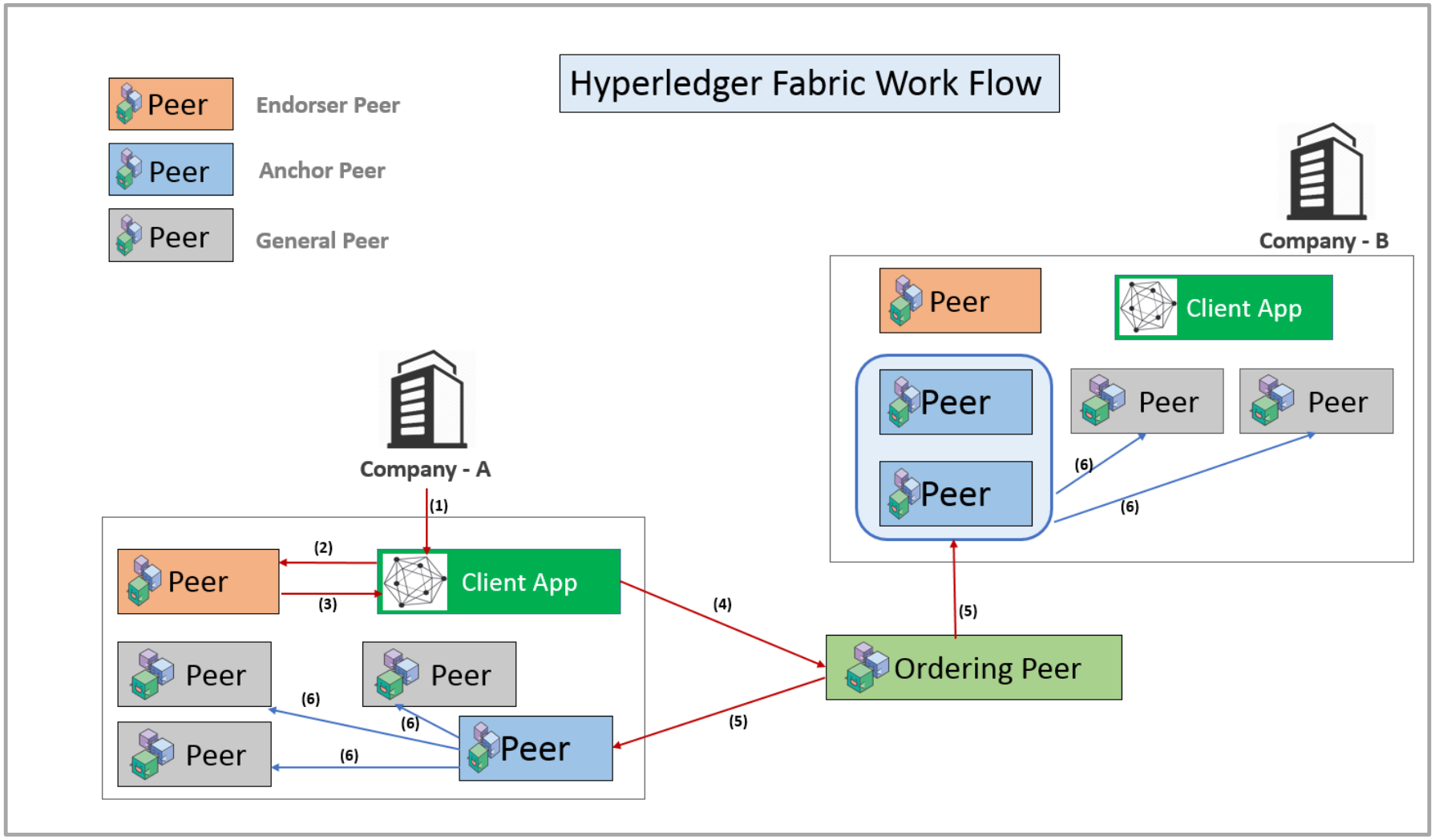


< Fabric >

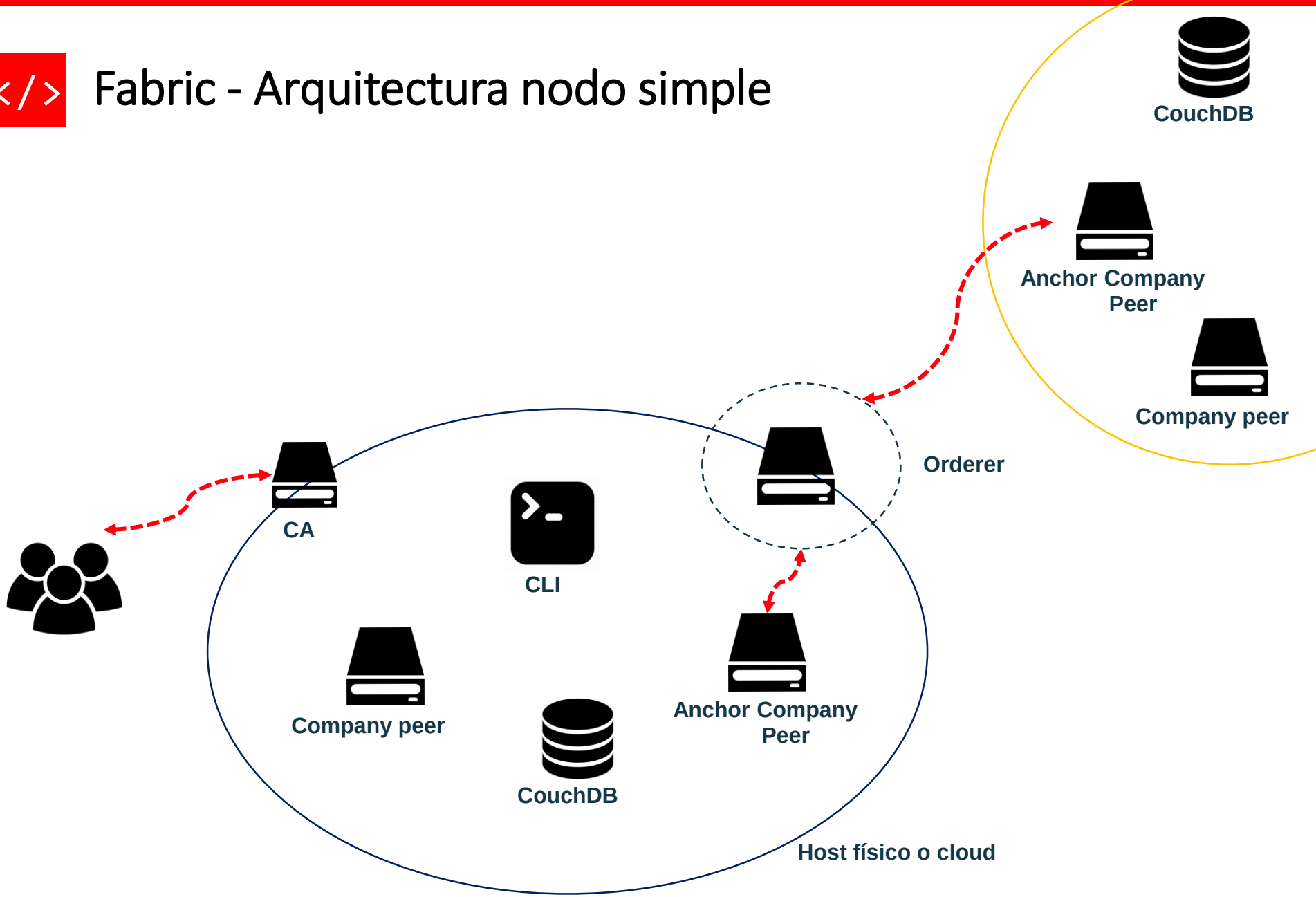
Fabric - Arquitectura



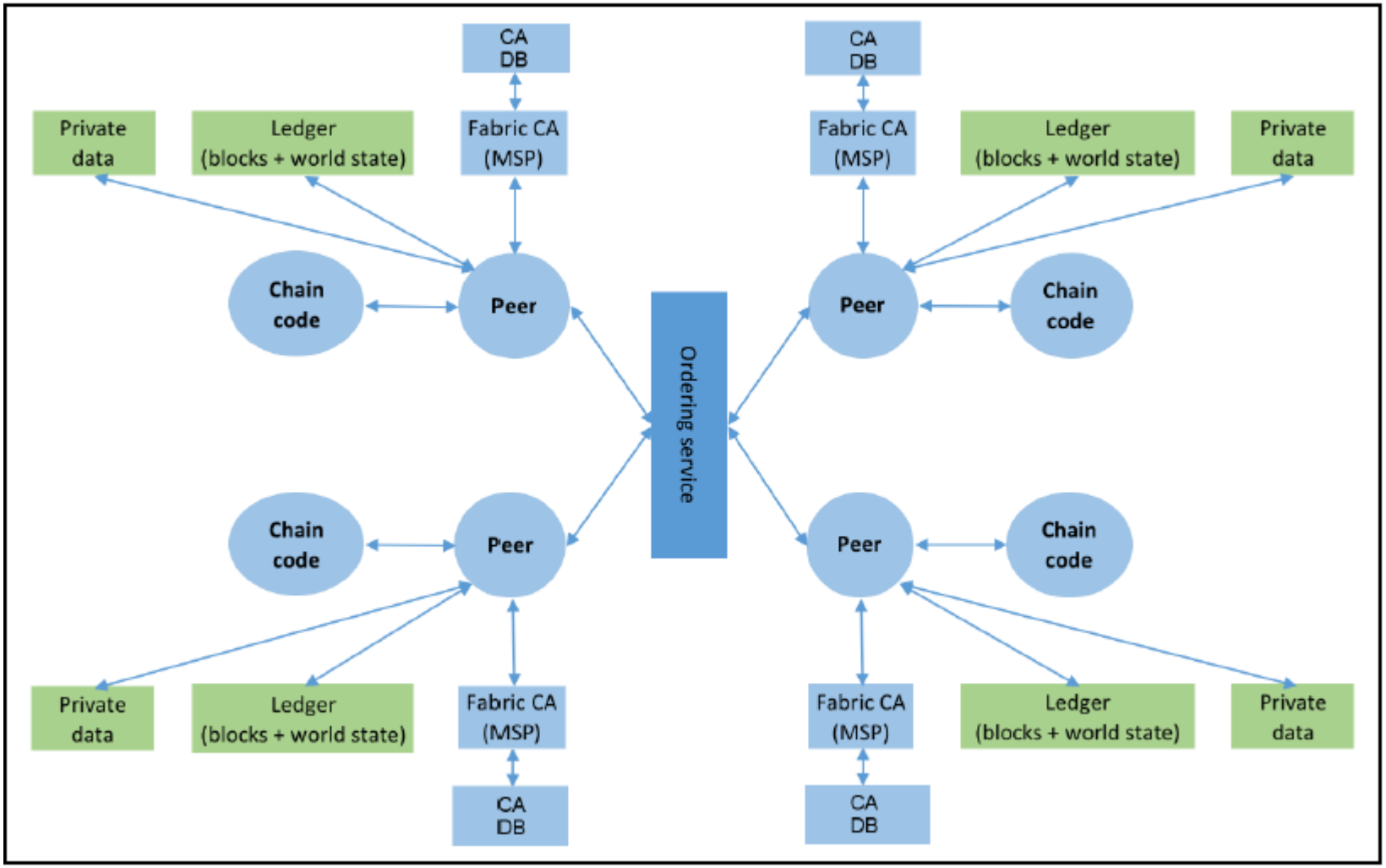
Fabric - WorkFlow



Fabric - Arquitectura nodo simple



Fabric – Arquitectura ejemplo



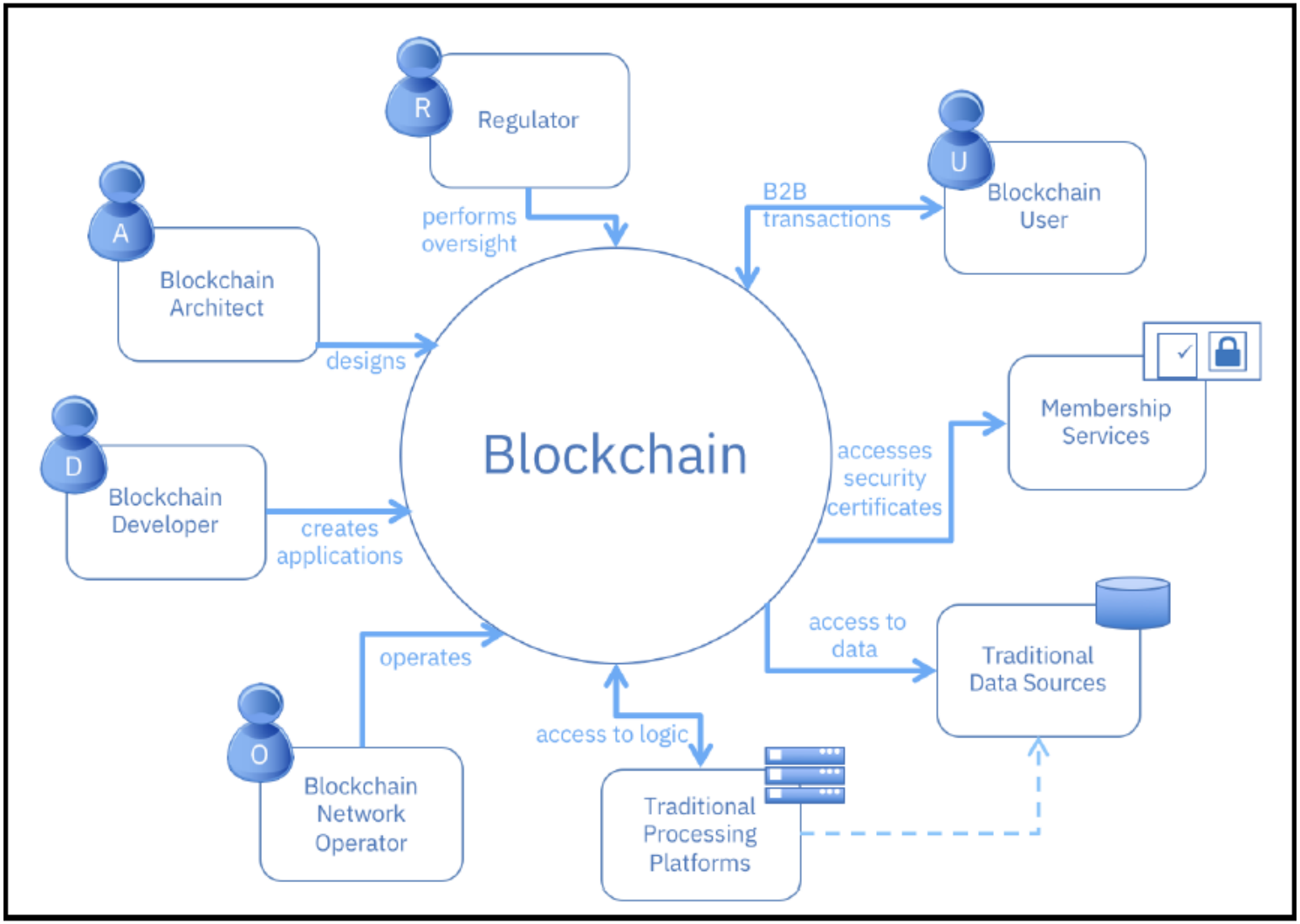
Fabric en proyectos en producción

Visión de un proyecto blockchain con Hyperledger Fabric:

Actores en la red blockchain: una blockchain es una infraestructura basada en red. Aquí se aplican construcciones de diseño, desarrollo, implementación, administración y soporte centradas en la red. Por lo tanto, es vital comprender a los diversos actores y sus roles que interactúan con la red blockchain para diversos fines, tales como gestión, soporte, usuarios comerciales, reguladores, etc.

- Desarrolladores
- Administradores
- Dueños del producto
- Auditores
- Usuarios de negocio

Fabric


R
Regulator
performs oversight
A
Blockchain Architect
designs
B2B transactions
U
Blockchain User
D
Blockchain Developer
creates applications
Blockchain
accesses security certificates
Membership Services
access to data
Traditional Data Sources
operates
O
Blockchain Network Operator
access to logic
Traditional Processing Platforms

Fabric

Interoperabilidad: este principio se basa en la interoperabilidad hacia versiones anteriores y NO en la interoperabilidad entre los diversos sistemas blockchain de Hyperledger como Sawtooth, Burrow, etc.

Enfocado a soluciones seguras: aquí la seguridad empresarial es primordial, de ahí el enfoque en la seguridad y no solo en la abstracción criptográfica, sino en la interacción entre componentes y la estructura que rige la naturaleza permisiva de blockchains permisionados. La mayoría de las industrias que se embarcan en el blockchain permisionada son industrias reguladas por unos procesos definidos.

Enfoque agnóstico sobre el Token: los proyectos Hyperledger no usan criptoactivos, criptomonedas, tokens o construcciones similares a monedas como mecanismos de incentivo para establecer sistemas de confianza. Si bien existe una noción de tokenización de activos que representa un activo físico, virtual o desmaterializado, la tokenización de activos es un concepto muy diferente de un token que se genera en el sistema como una virtualización de economía de incentivos.

APIs fáciles de usar: el objetivo es garantizar que los sistemas blockchain no solo tengan acceso a middleware empresarial, sino también acceso a redes comerciales, participantes existentes y nuevos sistemas sin exponer los detalles de las redes comerciales impulsadas por blockchain.

Fabric – Chaincodes

El término ampliamente utilizado, **Smart Contract**, se conoce como "**chaincode**" en Hyperledger Fabric.

Es la lógica de ejecución que codifica las reglas programadas para las transacciones que se vayan a realizar en Fabric. Chaincode (actualmente escrito en Go) **es instalado e instanciado en los pares de un canal por un miembro debidamente autorizado**.

Los usuarios finales invocan chaincode a través de una aplicación del lado del cliente que interactúa con un par de la red. Esta aplicación del lado cliente puede ser el CLI. Chaincode ejecuta transacciones de red, que si se validan, se anexan al ledger y modifican el world state de la blockchain.

Chaincode es un programa, escrito en **Go**, **node.js** y eventualmente en otros lenguajes de programación como **Java**, que implementa una interfaz prescrita. Chaincode **se ejecuta en un contenedor de Docker** aislado del proceso de homologación de pares. Chaincode inicializa y administra el estado del libro mayor a través de transacciones enviadas por las aplicaciones.

< /Instalación Hyperledger Fabric>

Fabric – Manos a la obra

Construiremos nuestra primera red de Hyperledger Fabric en el servidor cloud que nos facilita DevAcademy.

Para ello, deberemos instalar unos **pre-requisitos** al igual que hicimos en la instalación de Hyperledger Composer y posteriormente descargarnos un repositorio del GitHub de Hyperledger ya que haremos la instalación mediante la configuración que nos facilitan.

IMPORTANTE: **La versión que vamos a usar de Hyperledger Fabric es la 1.1** con lo que habrá que tener muy en cuenta la versión de los repositorios o de las herramientas de encriptación que nos descargamos. Han de estar acorde con la versión de Fabric.

< Pre-requisitos >

Fabric – Instalar pre-requisitos

Vamos a checkear que tenemos instalados los siguientes componentes:

Curl

```
curl --version
```

Docker -> V 17.60.2 o +

```
docker --version
```

Docker Compose -> V 1.14 o +

```
docker-compose --version
```

Go -> V 1.9.x

```
go version
```

Node -> 8.9.x //Solo para cuando usemos el SDK de Fabric

Python -> 2.7

```
python --version
```

< Descargar repositorio & binarios>

Fabric – Repositorio & binarios

El repositorio que nos bajemos, debe de estar siempre por debajo de la instalación de GO, en la carpeta “src”, se ejecutarán ficheros en GO y por ello debe de estar bajo su directorio.

```
cd work/src
```

```
git clone -b master https://github.com/hyperledger/fabric-samples.git
```

```
cd fabric-samples
```

```
git checkout v1.1.0
```

```
curl -sSL https://goo.gl/6wtTN5 | bash -s 1.1.0
```

< Arrancar el primer ejemplo>

Fabric – Primer ejemplo

Bien, una vez levantado nuestra primera red de Hyperledger Fabric (de un modo muy sencillo) vamos a ir paso a paso para comprender lo que sucedía dentro del script replicándolo nosotros mismos paso por paso.

Crypto-generator: Cryptogen consume un archivo **crypto-config.yaml** que contiene la topología de red y nos permite generar un conjunto de certificados y claves para las Organizaciones y los componentes que pertenecen a esas Organizaciones. **A cada organización se le asigna un certificado raíz exclusivo (ca-cert)** que vincula componentes específicos (pares y orderers) a esa organización. Al asignar a cada organización un certificado CA único, estamos imitando una red típica en la que un miembro participante utilizaría su propia autoridad certificadora. **Las transacciones y las comunicaciones dentro de Hyperledger Fabric están firmadas por la clave privada de una entidad (keystore), y luego se verifican mediante una clave pública (signcerts).**

Fabric – Primer ejemplo

Ver el archivo “crypto-config.yaml”

```
# ---------------------------------------------------------------------------
# "OrdererOrgs" - Definition of organizations managing orderer nodes
# ---------------------------------------------------------------------------
OrdererOrgs:
  # ---------------------------------------------------------------------------
  # Orderer
  # ---------------------------------------------------------------------------
  - Name: Orderer
    Domain: example.com
    # ---------------------------------------------------------------------------
    # "Specs" - See PeerOrgs below for complete description
    # ---------------------------------------------------------------------------
    Specs:
      - Hostname: orderer
# ---------------------------------------------------------------------------
# "PeerOrgs" - Definition of organizations managing peer nodes
# ---------------------------------------------------------------------------
PeerOrgs:
  # ---------------------------------------------------------------------------
  # Org1
  # ---------------------------------------------------------------------------
  - Name: Org1
    Domain: org1.example.com
    EnableNodeOUs: true
    # ---------------------------------------------------------------------------
```

Este archivo lo ejecuta la herramienta cryptogen que se encuentra dentro de la carpeta “bin” que se encuentra dentro de “fabric-samples”.

Fabric – Primer ejemplo

Configtxgen: consume el archivo configtx.yaml que contiene las definiciones para la red de ejemplo. Hay tres miembros: un Orderer Org (OrdererOrg) y dos Peer Orgs (Org1 y Org2) administrando y manteniendo cada uno dos peers. Este archivo también especifica un consorcio "SampleConsortium" que consiste en la conexión entre los peers de cada organización.

Ver el archivo "configtx.yaml"

Este archivo lo ejecuta las herramientas configtxgen y cryptogen que se encuentran dentro de la carpeta "bin" que se encuentra dentro de "fabric-samples".

```
Profiles:

    TwoOrgsOrdererGenesis:
        Capabilities:
            <<: *ChannelCapabilities
        Orderer:
            <<: *OrdererDefaults
            Organizations:
                - *OrdererOrg
            Capabilities:
                <<: *OrdererCapabilities
        Consortiums:
            SampleConsortium:
                Organizations:
                    - *Org1
                    - *Org2
    TwoOrgsChannel:
        Consortium: SampleConsortium
        Application:
            <<: *ApplicationDefaults
            Organizations:
                - *Org1
                - *Org2
            Capabilities:
                <<: *ApplicationCapabilities
```

Fabric – Primer ejemplo

El primer paso será generar los “artefactos” del proyecto, esto se traduce a que generaremos todos los ficheros de configuración de canales y material criptográfico del proyecto.

```
./byfn.sh -m generate

//darle a “y” cuando nos pregunte si queremos generar los artefactos.
```

```
Generating certs and genesis block for with channel 'mychannel' and CLI timeout of '10' seconds and CLI delay of '3' seconds
Continue? [Y/n] y
proceeding ...
/home/hyperledger/work/src/fabric-samples/first-network/../bin/cryptogen

##########################################################
##### Generate certificates using cryptogen tool #########
##########################################################
+ cryptogen generate --config=./crypto-config.yaml
org1.example.com
org2.example.com
+ res=0
+ set +x

/home/hyperledger/work/src/fabric-samples/first-network/../bin/configtxgen
##########################################################
#########  Generating Orderer Genesis block ##############
##########################################################
+ configtxgen -profile TwoOrgsOrdererGenesis -outputBlock ./channel-artifacts/genesis.block
2018-07-22 08:21:58.425 UTC [common/tools/configtxgen] main -> INFO 001 Loading configuration
2018-07-22 08:21:58.436 UTC [msp] getMspConfig -> INFO 002 Loading NodeOUs
2018-07-22 08:21:58.437 UTC [msp] getMspConfig -> INFO 003 Loading NodeOUs
2018-07-22 08:21:58.437 UTC [common/tools/configtxgen] doOutputBlock -> INFO 004 Generating genesis block
2018-07-22 08:21:58.437 UTC [common/tools/configtxgen] doOutputBlock -> INFO 005 Writing genesis block
+ res=0
+ set +x
```

< Ejercicio Universidades>

Fabric – Ejercicio Universidades

Pintar arquitectura

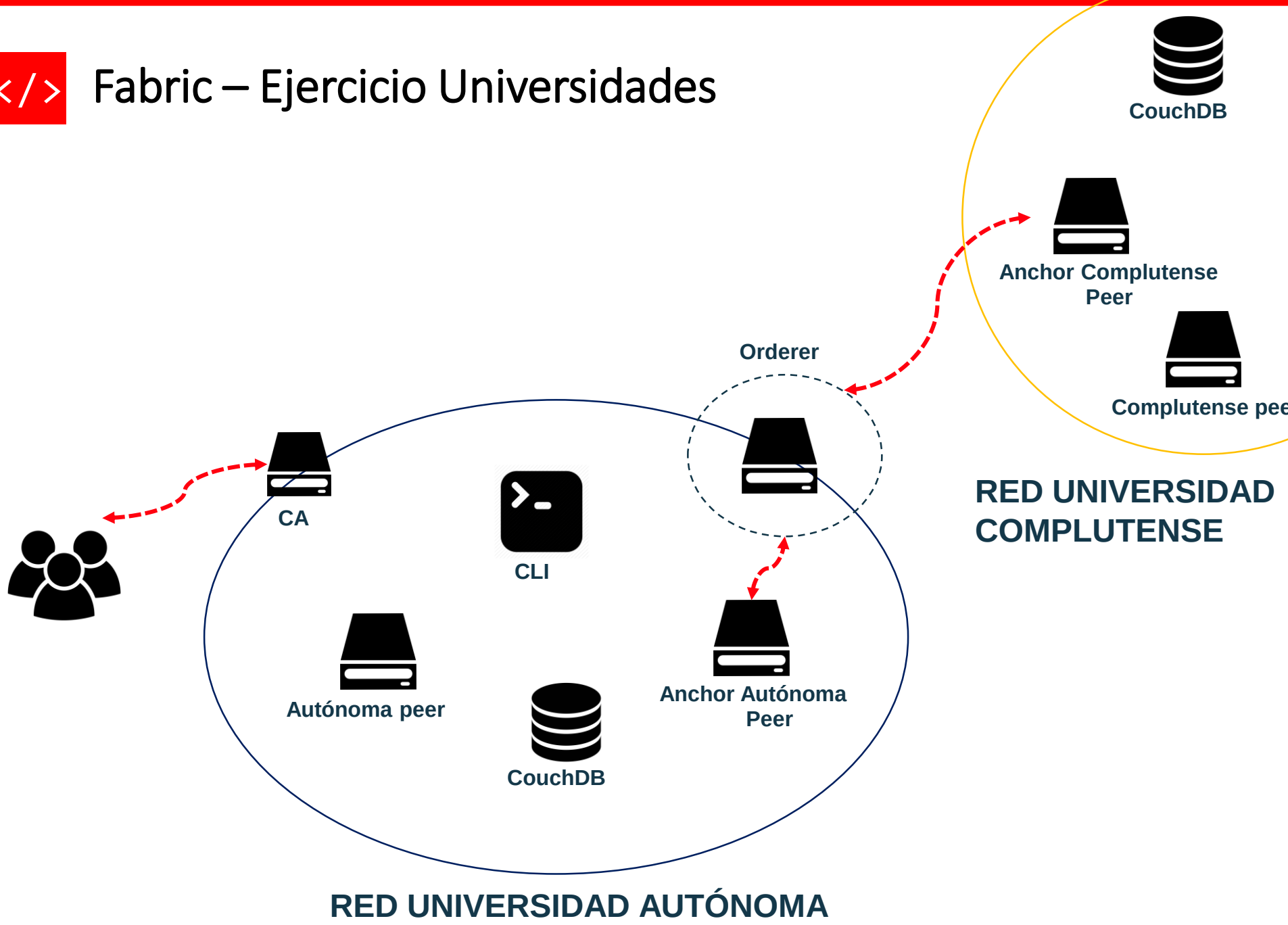
Explicar dónde iría la lógica

¿Quiénes serían los participantes?

¿Quiénes son los activos?

¿Podríamos escalarlo a más universidades que accedan posteriormente?

Fabric – Ejercicio Universidades



< ¡Manos a la obra!>

Fabric – Ejercicio Universidades

Cambiar el archivo de configuración crypto-config.yaml

```
vi crypto-config.yaml
```

```
OrdererOrgs:
  # ---------------------------------------------------------------------------
  # Orderer
  # ---------------------------------------------------------------------------
  - Name: Orderer
    Domain: universidades.com
    # ------------------------------------------------------------------------
    # "Specs" - See PeerOrgs below for complete description
    # ------------------------------------------------------------------------
    Specs:
      - Hostname: orderer
# ---------------------------------------------------------------------------
# "PeerOrgs" - Definition of organizations managing peer nodes
# ---------------------------------------------------------------------------
PeerOrgs:
  # ---------------------------------------------------------------------------
  # Org1
  # ---------------------------------------------------------------------------
  - Name: Autonoma
    Domain: autonoma.universidades.com
    EnableNodeOUs: true

    Template:
      Count: 1

    Users:
      Count: 1
  # ---------------------------------------------------------------------------
  # Org2: See "Org1" for full specification
  # ---------------------------------------------------------------------------
  - Name: Complutense
    Domain: complutense.universidades.com
    EnableNodeOUs: true
    Template:
      Count: 1
    Users:
      Count: 1
```

Fabric – Ejercicio Universidades

Después cambiaremos el archivo de configuración configtx.yaml

```
vi configtx.yaml
```

```
################################################################################
Profiles:

    UniversidadesOrdererGenesis:
        Capabilities:
            <<: *ChannelCapabilities
        Orderer:
            <<: *OrdererDefaults
            Organizations:
                - *OrdererOrg
            Capabilities:
                <<: *OrdererCapabilities
        Consortiums:
            UniversidadesConsortium:
                Organizations:
                    - *Autonoma
                    - *Complutense
    UniversidadesChannel:
        Consortium: UniversidadesConsortium
        Application:
            <<: *ApplicationDefaults
            Organizations:
                - *Autonoma
                - *Complutense
            Capabilities:
                <<: *ApplicationCapabilities
```

```
################################################################################
Organizations:

    # SampleOrg defines an MSP using the sampleconfig.  It should never be used
    # in production but may be used as a template for other definitions
    - &OrdererOrg
        # DefaultOrg defines the organization which is used in the sampleconfig
        # of the fabric.git development environment
        Name: OrdererOrg

        # ID to load the MSP definition as
        ID: OrdererMSP

        # MSPDir is the filesystem path which contains the MSP configuration
        MSPDir: crypto-config/ordererOrganizations/universidades.com/msp

    - &Autonoma
        # DefaultOrg defines the organization which is used in the sampleconfig
        # of the fabric.git development environment
        Name: AutonomaMSP

        # ID to load the MSP definition as
        ID: AutonomaMSP

        MSPDir: crypto-config/peerOrganizations/autonoma.universidades.com/msp

        AnchorPeers:
            # AnchorPeers defines the location of peers which can be used
            # for cross org gossip communication.  Note, this value is only
            # encoded in the genesis block in the Application section context
            - Host: peer0.autonoma.universidades.com
              Port: 7051
```

Fabric – Ejercicio Universidades

En la carpeta general del proyecto “universidades” editar fichero “Docker-compose-cli.yaml”

```yaml
peer0.autonoma.universidades.com:
  container_name: peer0.autonoma.universidades.com
  extends:
    file:  base/docker-compose-base.yaml
    service: peer0.autonoma.universidades.com
  networks:
    - universidades
  environment:
    - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
    - CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=couchdb0:5984
    # The CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME and CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD
    # provide the credentials for ledger to connect to CouchDB.  The username and password must
    # match the username and password
    - CORE_LEDGER_STATE_COUCHDBCONFIG
    - CORE_LEDGER_STATE_COUCHDBCONFIG
  depends_on:
    - couchdb0

cli:
  container_name: cli
  image: hyperledger/fabric-tools:$IMAGE_TAG
  tty: true
  stdin_open: true
  environment:
    - GOPATH=/opt/gopath
    - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
    #- CORE_LOGGING_LEVEL=DEBUG
    - CORE_LOGGING_LEVEL=INFO
    - CORE_PEER_ID=cli
    - CORE_PEER_ADDRESS=peer0.autonoma.universidades.com:7051
    - CORE_PEER_LOCALMSPID=AutonomaMSP
    - CORE_PEER_TLS_ENABLED=true
    - CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/autonoma.universidades.com/peers
    - CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/autonoma.universidades.com/peers/
    - CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/autonoma.universidades.com/p
    - CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/autonoma.universidades.com/users
  working_dir: /opt/gopath/src/github.com/hyperledger/fabric/peer
  command: /bin/bash
  volumes:
      - /var/run/:/host/var/run/
      - ./../chaincode/:/opt/gopath/src/github.com/chaincode
      - ./crypto-config:/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
      - ./scripts:/opt/gopath/src/github.com/hyperledger/fabric/peer/scripts/
      - ./channel-artifacts:/opt/gopath/src/github.com/hyperledger/fabric/peer/channel-artifacts
  networks:
    - universidades

peer0.complutense.universidades.com:
```

Fabric – Ejercicio Universidades

Crear material crypto:

```
../bin/cryptogen generate --config=./crypto-config.yaml

export FABRIC_CFG_PATH=$PWD

../bin/configtxgen -profile UniversidadesOrdererGenesis -outputBlock ./channel-artifacts/genesis.block

export CHANNEL_NAME=universidadeschannel  && ../bin/configtxgen -profile UniversidadesChannel -
outputCreateChannelTx ./channel-artifacts/channel.tx -channelID $CHANNEL_NAME

../bin/configtxgen -profile UniversidadesChannel -outputAnchorPeersUpdate ./channel-
artifacts/AutonomaMSPanchors.tx -channelID $CHANNEL_NAME -asOrg AutonomaMSP

../bin/configtxgen -profile UniversidadesChannel -outputAnchorPeersUpdate ./channel-
artifacts/ComplutenseMSPanchors.tx -channelID $CHANNEL_NAME -asOrg ComplutenseMSP
```

```
hyperledger@cursoblockchain:~/work/src/fabric-samples/universidades$ ../bin/cryptogen generate --config=./crypto-config.yaml
autonoma.universidades.com
complutense.universidades.com
hyperledger@cursoblockchain:~/work/src/fabric-samples/universidades$ export FABRIC_CFG_PATH=$PWD
hyperledger@cursoblockchain:~/work/src/fabric-samples/universidades$ ../bin/configtxgen -profile UniversidadesOrdererGenesis -
2018-07-22 14:53:06.534 UTC [common/tools/configtxgen] main -> INFO 001 Loading configuration
2018-07-22 14:53:06.545 UTC [msp] getMspConfig -> INFO 002 Loading NodeOUs
2018-07-22 14:53:06.546 UTC [msp] getMspConfig -> INFO 003 Loading NodeOUs
2018-07-22 14:53:06.546 UTC [common/tools/configtxgen] doOutputBlock -> INFO 004 Generating genesis block
2018-07-22 14:53:06.546 UTC [common/tools/configtxgen] doOutputBlock -> INFO 005 Writing genesis block
hyperledger@cursoblockchain:~/work/src/fabric-samples/universidades$ export CHANNEL_NAME=universidadeschannel  && ../bin/confi
nnel-artifacts/channel.tx -channelID $CHANNEL_NAME
2018-07-22 14:53:12.880 UTC [common/tools/configtxgen] main -> INFO 001 Loading configuration
2018-07-22 14:53:12.891 UTC [common/tools/configtxgen] doOutputChannelCreateTx -> INFO 002 Generating new channel configtx
2018-07-22 14:53:12.891 UTC [msp] getMspConfig -> INFO 003 Loading NodeOUs
2018-07-22 14:53:12.892 UTC [msp] getMspConfig -> INFO 004 Loading NodeOUs
2018-07-22 14:53:12.919 UTC [common/tools/configtxgen] doOutputChannelCreateTx -> INFO 005 Writing new channel tx
hyperledger@cursoblockchain:~/work/src/fabric-samples/universidades$ ../bin/configtxgen -profile UniversidadesChannel -outputA
nnelID $CHANNEL_NAME -asOrg AutonomaMSP
2018-07-22 14:53:18.878 UTC [common/tools/configtxgen] main -> INFO 001 Loading configuration
2018-07-22 14:53:18.889 UTC [common/tools/configtxgen] doOutputAnchorPeersUpdate -> INFO 002 Generating anchor peer update
2018-07-22 14:53:18.890 UTC [common/tools/configtxgen] doOutputAnchorPeersUpdate -> INFO 003 Writing anchor peer update
hyperledger@cursoblockchain:~/work/src/fabric-samples/universidades$ ../bin/configtxgen -profile UniversidadesChannel -outputA
channelID $CHANNEL_NAME -asOrg ComplutenseMSP
2018-07-22 14:53:28.859 UTC [common/tools/configtxgen] main -> INFO 001 Loading configuration
2018-07-22 14:53:28.869 UTC [common/tools/configtxgen] doOutputAnchorPeersUpdate -> INFO 002 Generating anchor peer update
2018-07-22 14:53:28.870 UTC [common/tools/configtxgen] doOutputAnchorPeersUpdate -> INFO 003 Writing anchor peer update
hyperledger@cursoblockchain:~/work/src/fabric-samples/universidades$
```

Fabric – Ejercicio Universidades

Arrancar el fichero Docker Compose junto con el del CouchDB:

```
docker-compose -f docker-compose-cli.yaml -f docker-compose-couch.yaml up -d
```

```
fe735c3da899        hyperledger/fabric-peer:latest
utes       0.0.0.0:10051->7051/tcp, 0.0.0.0:10053->7053/tcp   peer1.complutense.universidades.com
fdb0b770cd12        hyperledger/fabric-peer:latest
utes       0.0.0.0:7051->7051/tcp, 0.0.0.0:7053->7053/tcp     peer0.autonoma.universidades.com
356ba16be87e        hyperledger/fabric-peer:latest
utes       0.0.0.0:8051->7051/tcp, 0.0.0.0:8053->7053/tcp     peer1.autonoma.universidades.com
ec1a725564f1        hyperledger/fabric-peer:latest
utes       0.0.0.0:9051->7051/tcp, 0.0.0.0:9053->7053/tcp     peer0.complutense.universidades.com
d75f1424e07c        hyperledger/fabric-couchdb
utes       4369/tcp, 9100/tcp, 0.0.0.0:8984->5984/tcp         couchdb3
b9407afbc682        hyperledger/fabric-couchdb
utes       4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp         couchdb0
d61bf8fa75f5        hyperledger/fabric-couchdb
utes       4369/tcp, 9100/tcp, 0.0.0.0:6984->5984/tcp         couchdb1
aab223916f47        hyperledger/fabric-orderer:latest
utes       0.0.0.0:7050->7050/tcp                             orderer.universidades.com
37f99923e230        hyperledger/fabric-couchdb
utes       4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp         couchdb2
```

```
docker exec –it cli bash
```

Ya estamos dentro de la terminal. Vamos a configurar el entorno.

Fabric – Ejercicio Universidades

Configuración:

```
export CHANNEL_NAME=universidadeschannel

peer channel create -o orderer.universidades.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/channel.tx --tls
--cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/universidades.com/orderers/order
er.universidades.com/msp/tlscacerts/tlsca.universidades.com-cert.pem

peer channel join -b universidadeschannel.block

CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/complutense
.universidades.com/users/Admin@complutense.universidades.com/msp
CORE_PEER_ADDRESS=peer0.complutense.universidades.com:7051 CORE_PEER_LOCALMSPID="ComplutenseMSP"
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/complut
ense.universidades.com/peers/peer0.complutense.universidades.com/tls/ca.crt peer channel join -b
universidadeschannel.block

peer channel update -o orderer.universidades.com:7050 -c $CHANNEL_NAME -f ./channel-
artifacts/AutonomaMSPanchors.tx --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/universidades.com/orderers/order
er.universidades.com/msp/tlscacerts/tlsca.universidades.com-cert.pem

CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/complutense
.universidades.com/users/Admin@complutense.universidades.com/msp
CORE_PEER_ADDRESS=peer0.complutense.universidades.com:7051 CORE_PEER_LOCALMSPID="ComplutenseMSP"
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/complut
ense.universidades.com/peers/peer0.complutense.universidades.com/tls/ca.crt peer channel update -o
orderer.universidades.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/ComplutenseMSPanchors.tx --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/universidades.com/orderers/order
er.universidades.com/msp/tlscacerts/tlsca.universidades.com-cert.pem
```

<¿Y el Chaincode?>

Fabric – Ejercicio Universidades

Instalar e instanciar el Chaincode:

```
peer chaincode install -n univcc -v 1.0 -p github.com/chaincode/universidades/go/

peer chaincode instantiate -o orderer.universidades.com:7050 --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/universidades.com/orderers/order
er.universidades.com/msp/tlscacerts/tlsca.universidades.com-cert.pem -C $CHANNEL_NAME -n univcc -v 1.0 -c
'{"Args":["initLedger"]}' -P "OR ('AutonomaMSP.peer','ComplutenseMSP.peer')"

peer chaincode invoke -o orderer.universidades.com:7050 --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/universidades.com/orderers/order
er.universidades.com/msp/tlscacerts/tlsca.universidades.com-cert.pem -C $CHANNEL_NAME -n univcc -c
'{"Args":["initLedger"]}'

peer chaincode query -C $CHANNEL_NAME -n univcc -c '{"Args":["queryAllAlumnos"]}'
```

```
root@23580d2ce1fe:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode query -C $CHANNEL_NAME
2018-07-22 17:21:33.050 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc
2018-07-22 17:21:33.050 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
Query Result: [{"Key":"ALUMNO0", "Record":{"carrera":"TYPE1","dateCreate":"01/06/2010","idAlumno":"0","nomb
TYPE2","dateCreate":"01/03/2011","idAlumno":"1","nombre":"Alberto","universidad":"Derecho"}},{"Key":"ALUMNO
"Laura","universidad":"ADE"}},{"Key":"ALUMNO3", "Record":{"carrera":"TYPE4","dateCreate":"11/01/2011","idA
":{"carrera":"TYPE3","dateCreate":"22/02/2012","idAlumno":"4","nombre":"Maria","universidad":"Informatica"}
2018-07-22 17:21:33.063 UTC [main] main -> INFO 003 Exiting.....
root@23580d2ce1fe:/opt/gopath/src/github.com/hyperledger/fabric/peer#
```

Fabric – Ejercicio Universidades

Editar nuestro Chaincode

```
peer chaincode install -n univcc -v 1.1 -p github.com/chaincode/universidades/go/

peer chaincode upgrade -o orderer.universidades.com:7050 --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/universidades.com/orderers/order
er.universidades.com/msp/tlscacerts/tlsca.universidades.com-cert.pem -C $CHANNEL_NAME -n univcc -v 1.1 -c
'{"Args":["init"]}'  -P "OR ('AutonomaMSP.peer','ComplutenseMSP.peer')"

peer chaincode query -C $CHANNEL_NAME -n univcc -c
'{"Args":["queryAlumno","{\"selector\":{\"nombre\":\"Alberto\"}}"]}'
```

```
root@23580d2celfe:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode query -C $CHANNEL_NAME -n univcc -c '{"Args":["queryAlumno","{\
2018-07-22 17:43:32.826 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc
2018-07-22 17:43:32.826 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
Query Result: [{"Key":"ALUMNO1", "Record":{"carrera":"TYPE2","dateCreate":"01/03/2011","idAlumno":"1","nombre":"Alberto","universidad":"Derecho"}}]
2018-07-22 17:43:32.840 UTC [main] main -> INFO 003 Exiting.....
```

< /Curiosidades de Hyperledger Fabric>

Fabric – Curiosidades

¿Dónde crees que se guardan físicamente los datos de Hyperledger Fabric?

¿Y cómo?

¿Y si borro de 1 peer la información qué pasa?

Si no uso Docker, ¿qué hago?

¿Cómo crear redes físicas en distintos equipos?

¿Alguna vez has visto colapsado un Fabric?

< / Muchas gracias!>