

# Blockchain 3.0

## Primeiros Passos



# Minha Trajetória

16 Anos Desenvolvendo Software

Delphi, Java, C#, Node, Python,  
Blockchain Hyperledger Fabric

Atuação na Área da Saúde Pública  
e Privada

Atuação na Área de Seguros e  
Previdência

Graduado em C.C  
(Faculdade Ruy Barbosa)

Mestre em Eng. de Sistemas  
(IFBA - PPGESP)

Doutorando em C.C  
(UFBA – PGCOMP)

# Objetivo

Compartilhar e expandir conhecimentos para facilitar os primeiros passos no desenvolvimento de blockchain 3.0 na plataforma Hyperledger. Demonstrar um setup funcional e adaptável as necessidades do negócio de cada participante.



# Calma Calabreso!

Aprender Blockchain não é como uma corrida de 5 km e sim como uma maratona, lembremos que estamos saindo do paradigma Cliente/Servidor para o P2P com vários outros conceitos associados.



# SUMÁRIO

O que é Blockchain?

Solução?

Hyperledger

Hands-on

Gerações

Tipos de Blockchain

HL Fabric e Componentes

Links Úteis

# Mas... o que é Blockchain?

A blockchain é um livro-razão seguro, compartilhado e distribuído que facilita o processo de gravação e rastreamento de recursos sem a necessidade de confiança em uma autoridade central, permitindo que duas partes comuniquem-se e troquem recursos em uma rede (P2P), na qual decisões são tomadas pela maioria, e não por uma única entidade [Salman et al. 2019]. Este livro-razão, conhecido também como DLT (Distributed Ledger Technology) é composto de transações assinadas criptograficamente e agrupadas em blocos. Cada bloco está ligado ao bloco anterior e as validações das transações utilizam um mecanismo de consenso.

# Algumas Características

- Descentralização – Rede P2P
- Resiliência – Funcional com 51% dos participantes
- Não Repúdio – Transações assinadas criptograficamente
- Auditabilidade – Não existe exclusão total dos dados

# Gerações



Blockchain 1.0

Início - 2009

(Satoshi Nakamoto - Bitcoin)



Blockchain 2.0

Início - 2013

Contratos  
Inteligentes  
(Ethereum)



Blockchain 3.0

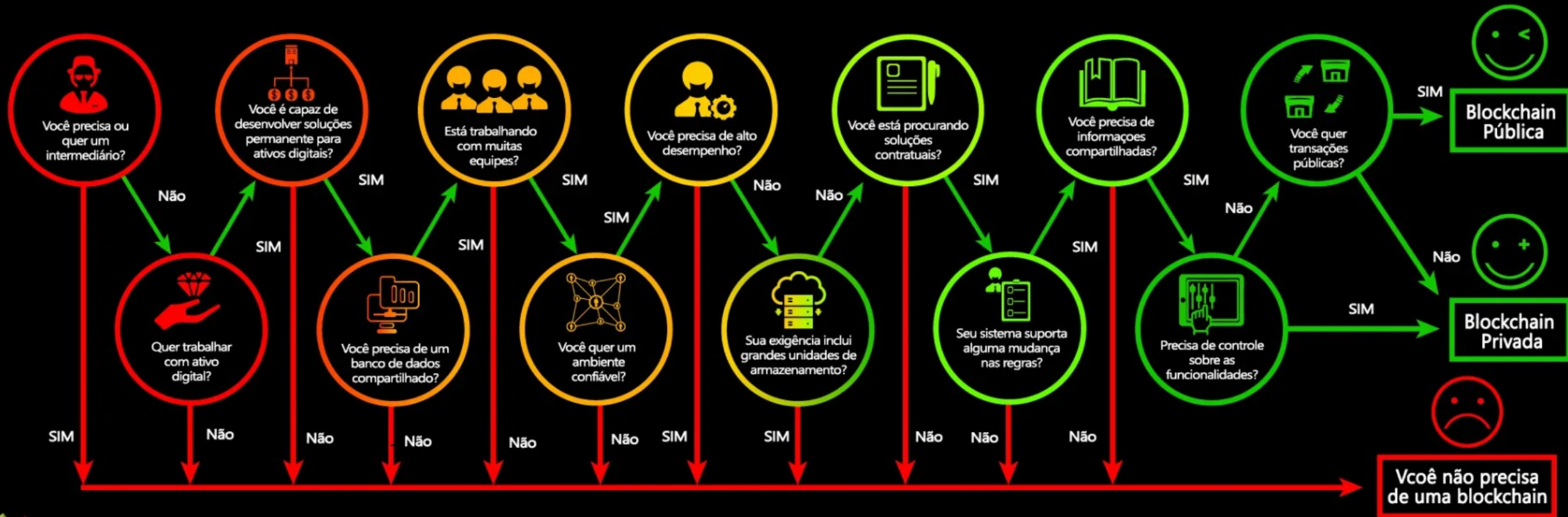
Início - 2015

Aplicações em Diversas  
Áreas (Hyperledger)



# Solução?

## VOCÊ PRECISA DE UMA BLOCKCHAIN?



**101 Blockchains**

Created by 101blockchains.com

# Tipos de Blockchain

## Pública

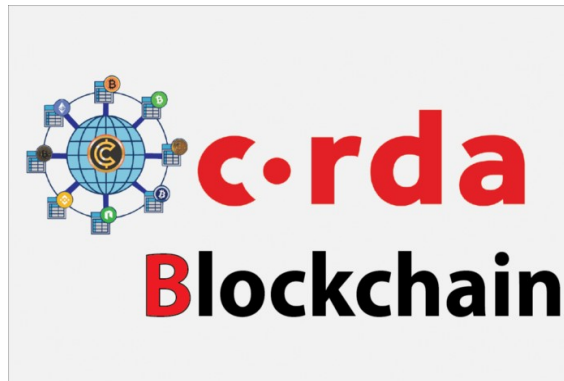
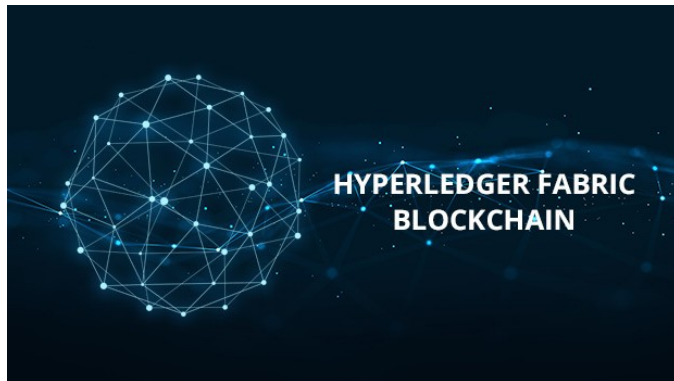
Também conhecida como **blockchain aberta**, é definida por **uma rede aberta e descentralizada**, na qual qualquer pessoa pode participar e realizar transações, bem como ter acesso aos registros e informações armazenados na rede.



# Tipos de Blockchain

## Privada

Originou-se de um **movimento natural de interesse de governos, instituições financeiras e bancos após a explosão da utilização das moedas digitais**. As blockchains privadas são **redes restritas e controladas que possuem organização central por uma entidade ou organização específica**. Algumas empresas que utilizam: Receita Federal, RNDS, IBM...)



# Tipos de Blockchain

## **Híbrida**

Esse modelo permite a configuração de um sistema público/privado, desde que haja permissão por parte do sistema público. Com isso, a organização pode controlar quem tem acesso aos dados públicos e quem pode acessar dados específicos armazenados na área privada. Boa aplicabilidade para entes regulatórios.

# Tipos de Blockchain

## Consórcio

É uma rede **semi privada**, também conhecida como blockchain autorizada. Surgindo da necessidade de **descentralizar informações e manter a transparência das operações**, ela permite que diferentes membros da organização central colaborem nas operações da tecnologia, desde que validados; sem o risco de ter apenas uma entidade no controle da rede.

# Hyperledger



**HYPERLEDGER**

## Distributed Ledgers



**HYPERLEDGER  
BESU**

Java-based  
Ethereum client



**HYPERLEDGER  
BURROW**

Permissionable smart  
contract machine (EVM)



**HYPERLEDGER  
FABRIC**

Enterprise-grade DLT  
with privacy support



**HYPERLEDGER  
INDY**

Decentralized identity



**HYPERLEDGER  
IROHA**

Mobile application focus



**HYPERLEDGER  
SAWTOOTH**

Permissioned & permissionless  
support; EVM transaction family

## Libraries



**HYPERLEDGER  
ARIES**



**HYPERLEDGER  
QUILT**



**HYPERLEDGER  
TRANSACTION**



**HYPERLEDGER  
URSA**

## Tools



**HYPERLEDGER  
AVALON**



**HYPERLEDGER  
CACTUS**



**HYPERLEDGER  
CALIPER**



**HYPERLEDGER  
CELLO**



**HYPERLEDGER  
EXPLORER**

## Domain-Specific



**HYPERLEDGER  
GRID**



**HYPERLEDGER  
LABS**

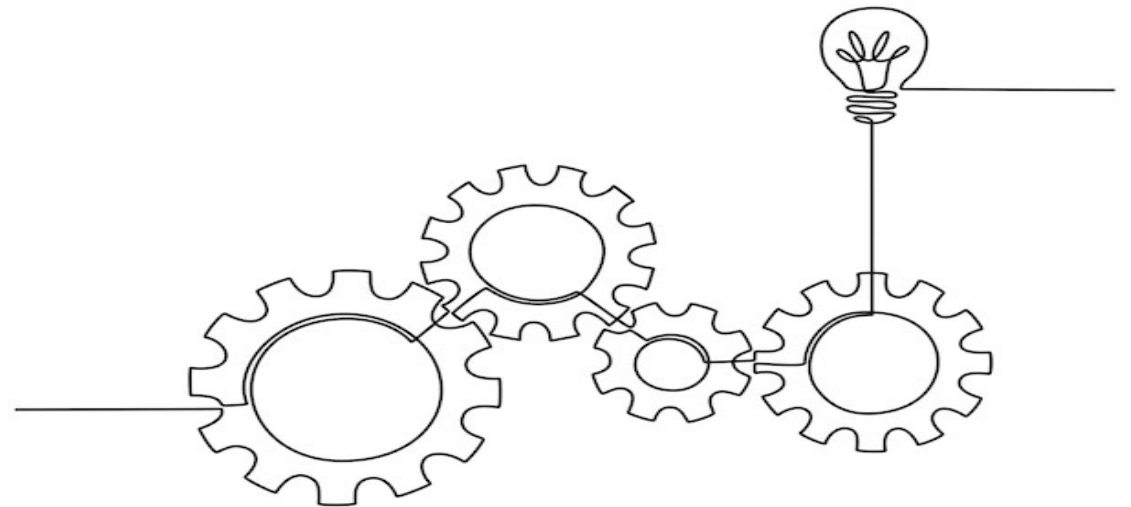




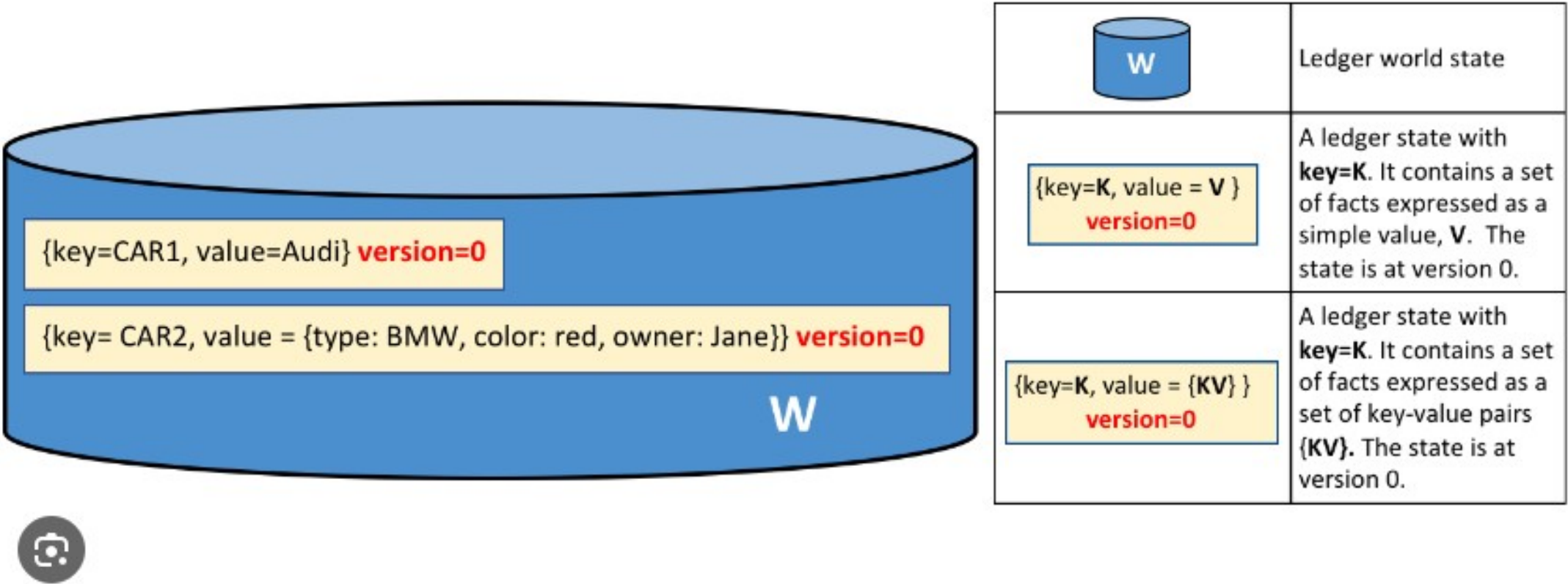
O Hyperledger Fabric é um framework para desenvolvimento de blockchains permissionadas e empresariais (GREVE et al., 2018).

## Componentes

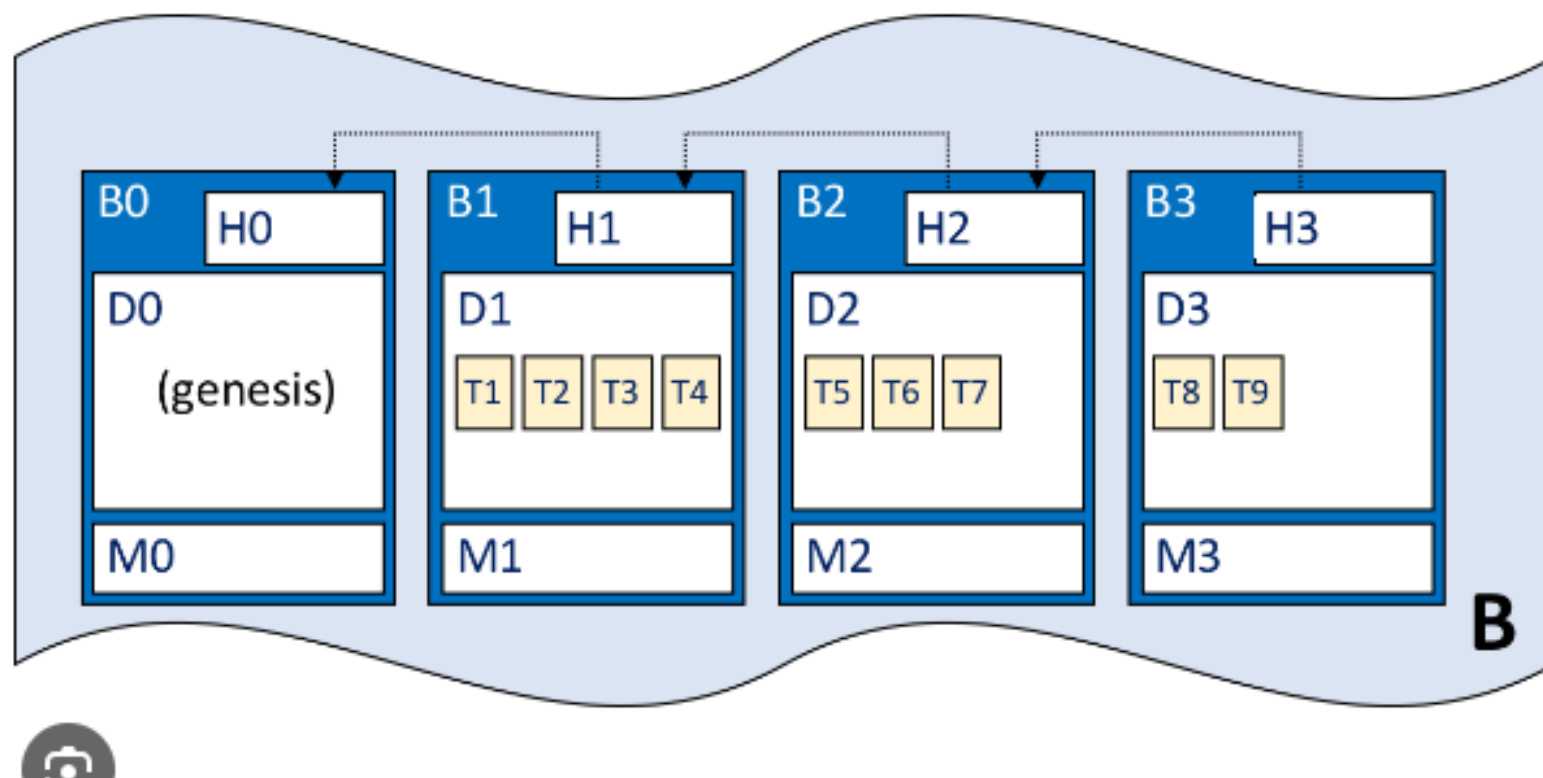
- Peers
- Membership Service Provider (MSP)
- Ledger (Estado Global e Blockchain)
- Canal
- Smart Contract e Transações
- Consenso e Ordenação
- Protocolo de mensageria (Gossip)


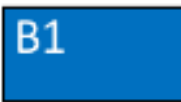
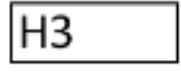
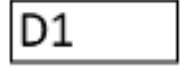

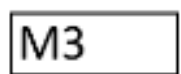
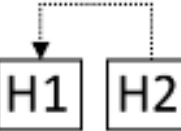


# Componentes HL Fabric – Ledger World State

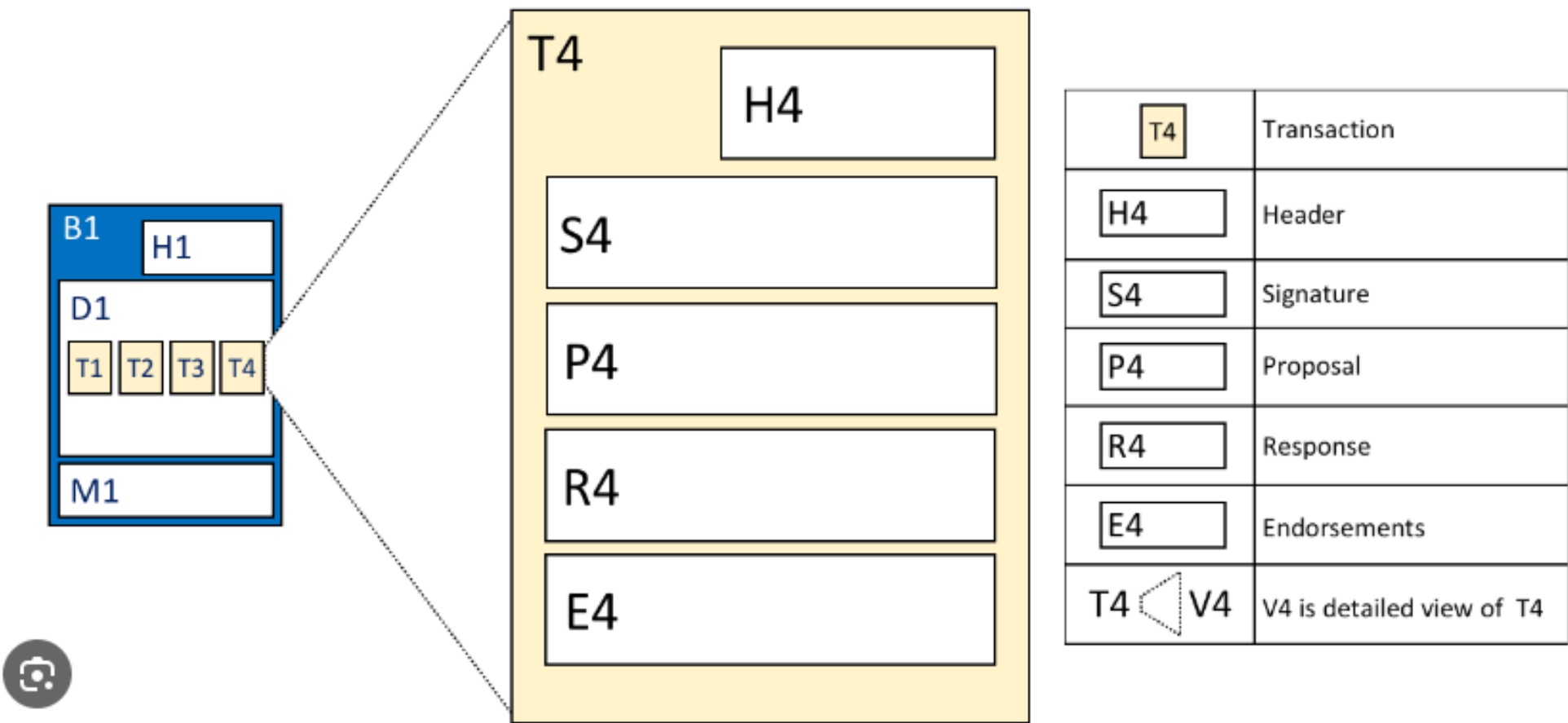


# Componentes HL Fabric – Ledger Blockchain

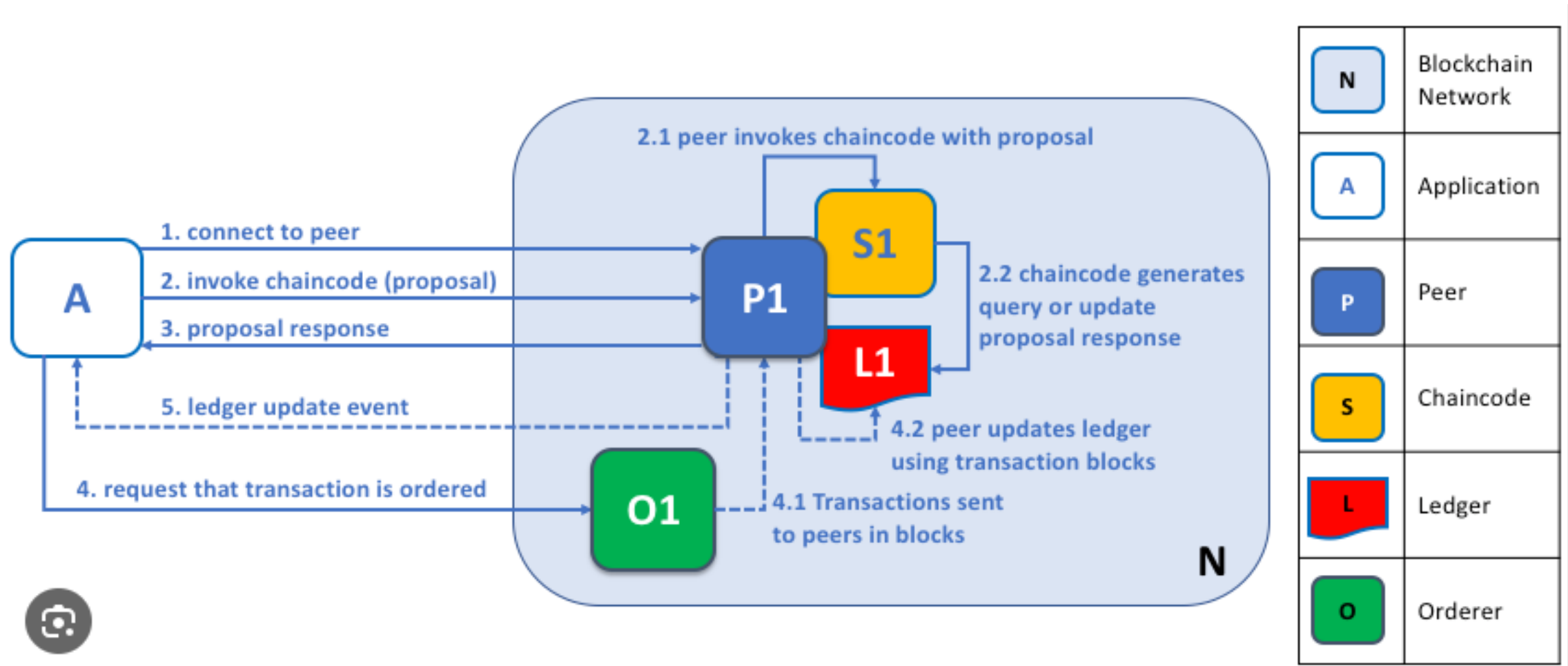


 B	Blockchain
 B1	Block
 H3	Block header
 D1	Block data
 T5	Transaction
 M3	Block metadata
	H2 is chained to H1

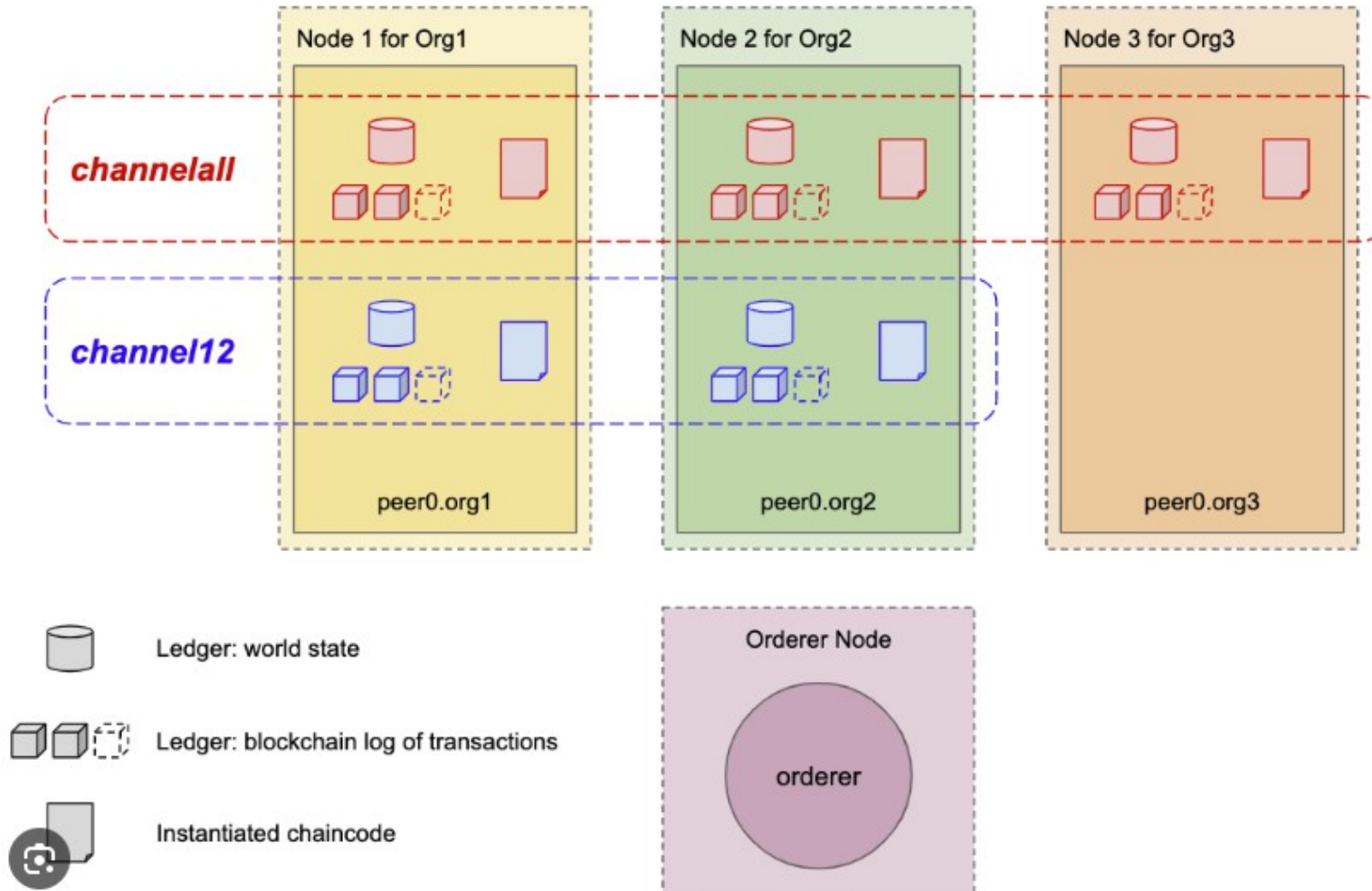
# Componentes HL Fabric – Ledger Blockchain



# Componentes HL Fabric - Peers

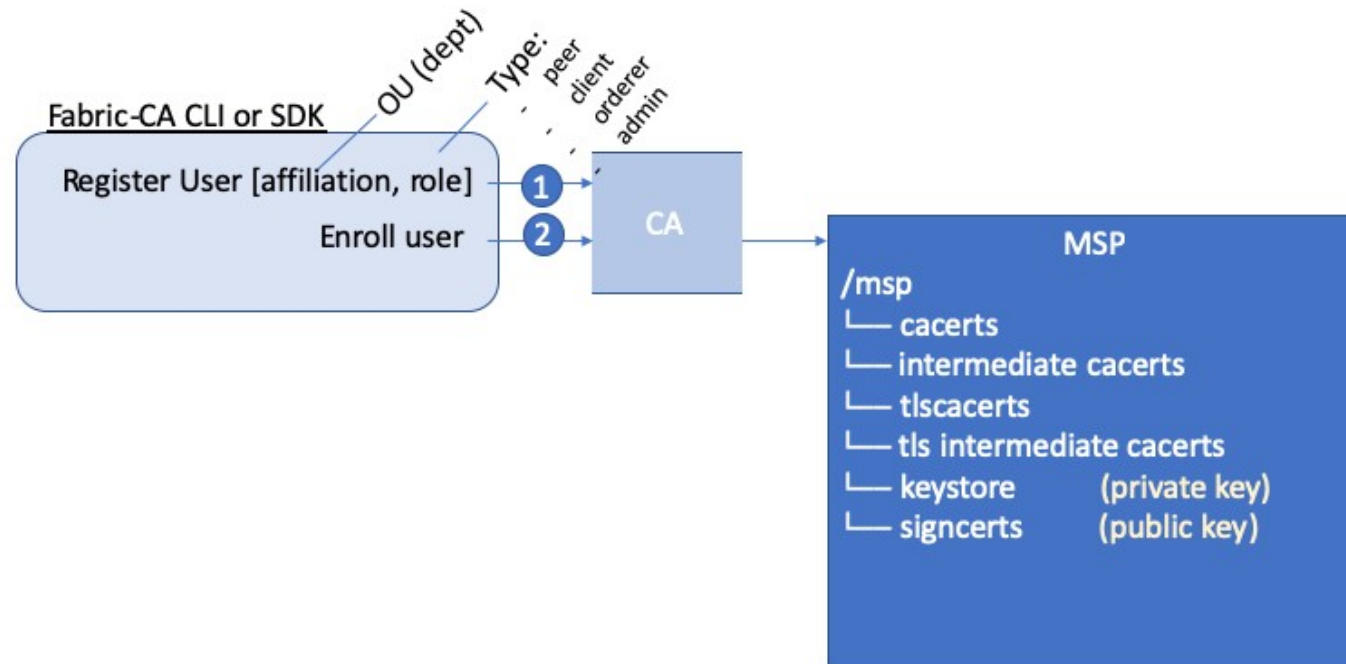


# Componentes HL Fabric - Canal



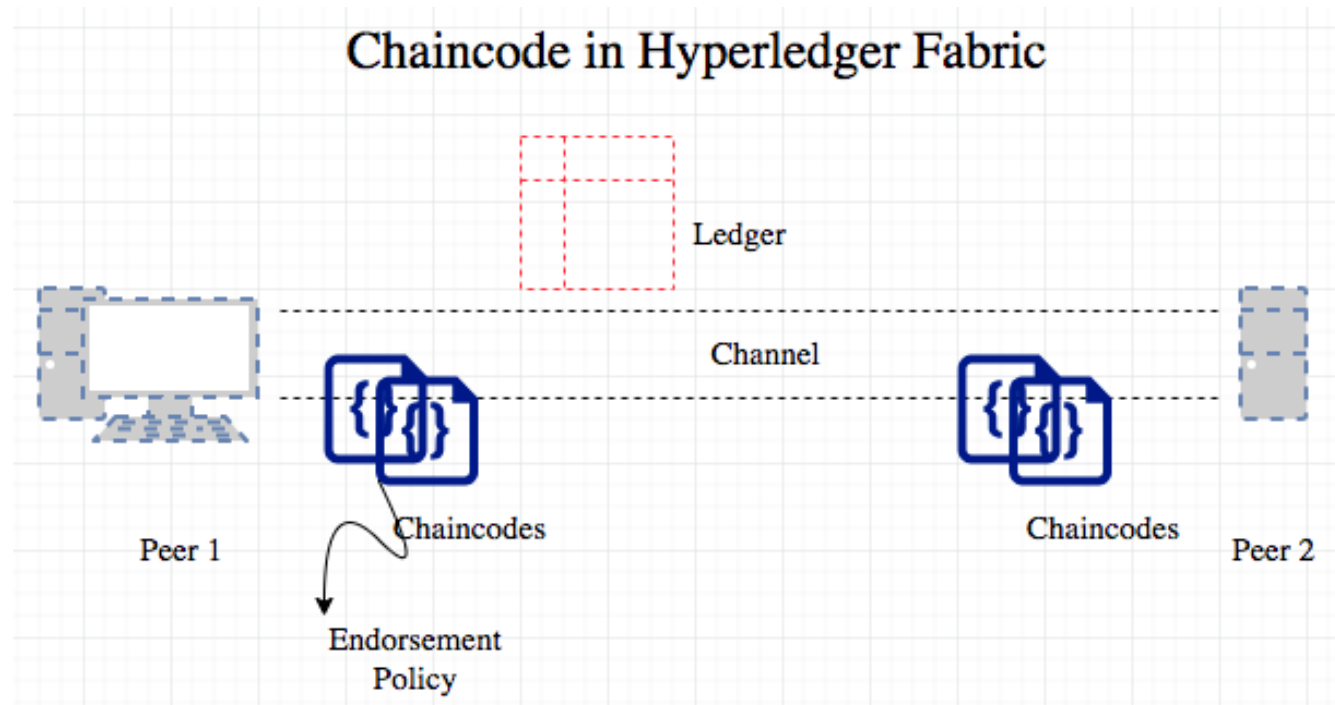


# Componentes HL Fabric - MSP



# Componentes HL Fabric – Chaincode

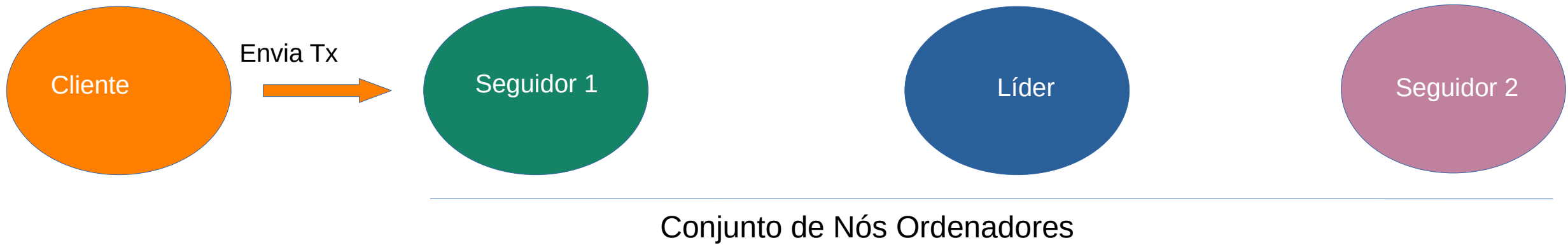
O contrato inteligente é conhecido como *Chaincode* e pode ser escrito nas linguagens: Go, Java, e Node.js. Para realizar a interação com o *chaincode*, são realizadas transações. Existem dois tipos de transações: *Invoke* e *Query*. A transação do tipo *invoke* executa uma função que altera o estado do *ledger*, enquanto a transação *query* realiza uma consulta.



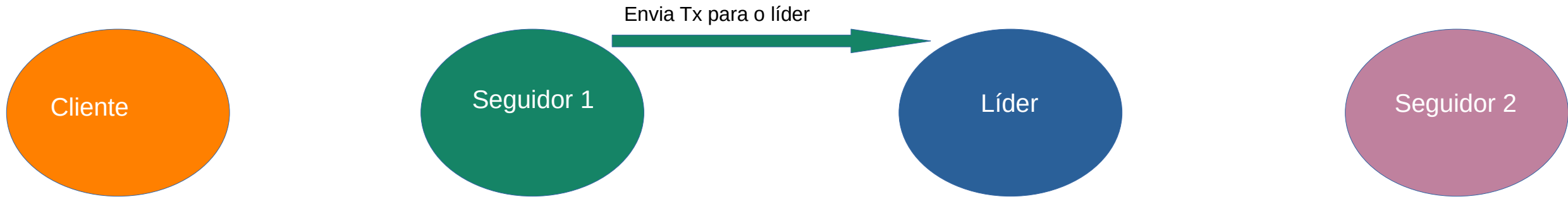
# Protocolos de Consenso

O consenso pode ocorrer de diferentes maneiras, seja com o uso de algoritmos de loteria, como o Proof of Work (PoW) (Bitcoin), ou através do uso de métodos baseados em votação. Dentre os mecanismos baseados em votação destacam-se os modelos Crash Fault Tolerant (CFT) e Byzantine Fault Tolerant (BFT). Enquanto o mecanismo CFT assume que alguns nós responsáveis pelo consenso podem falhar, o mecanismo BFT admite que além de falhar, alguns nós podem agir de maneira maliciosa. Segundo (MIERS et al., 2020), a escolha adequada à solução irá depender das características da rede. RAFT é CFT e SmartBFT (BFT)

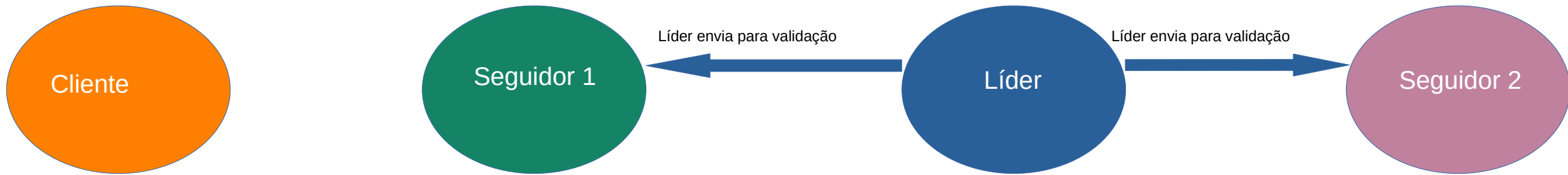
# Protocolo de Consenso Raft e Mensageria Gossip



# Protocolo de Consenso Raft

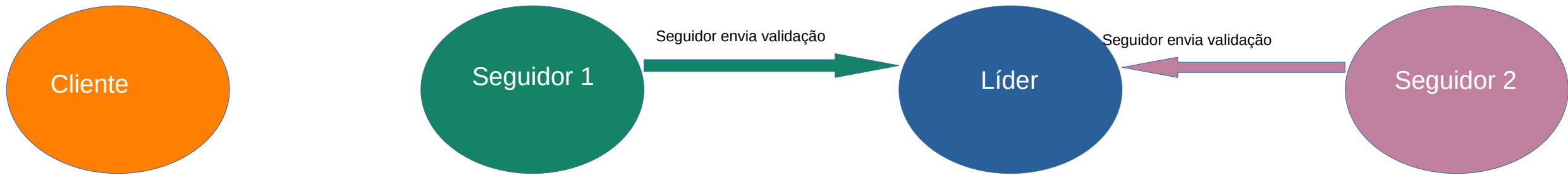


# Protocolo de Consenso Raft

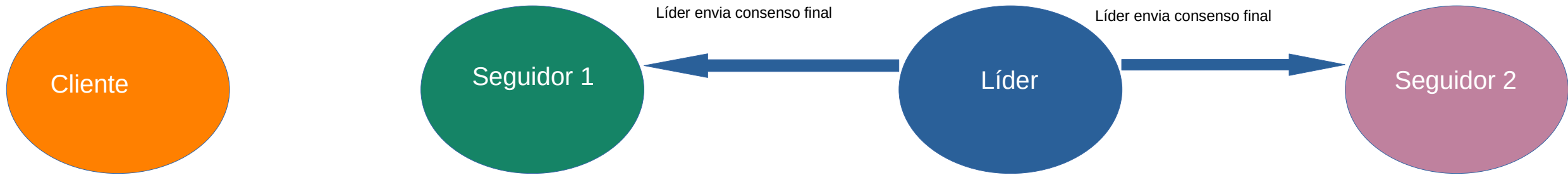




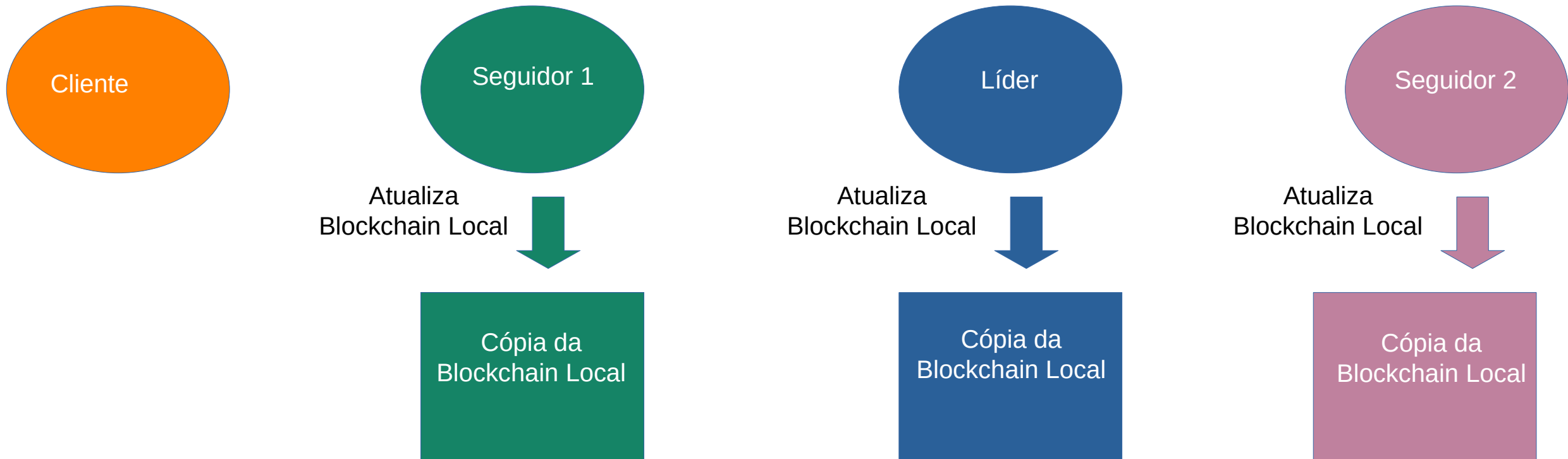
# Protocolo de Consenso Raft



# Protocolo de Consenso Raft



# Protocolo de Consenso Raft



# Configurações da Rede

*Arquivo **core.yaml**: Configurações **gerais da rede** (Arquivo base **utilizado por vários binários**)*

```
# BCCSP (Blockchain crypto provider): Select which crypto implementation or
# library to use
BCCSP:
  Default: SW
  # Settings for the SW crypto provider (i.e. when DEFAULT: SW)
  SW:
    # TODO: The default Hash and Security level needs refactoring to be
    # fully configurable. Changing these defaults requires coordination
    # SHA2 is hardcoded in several places, not only BCCSP
    Hash: SHA2
    Security: 256
    # Location of Key Store
    FileKeyStore:
      # If "", defaults to 'mspConfigPath'/keystore
      KeyStore:
    # Settings for the PKCS#11 crypto provider (i.e. when DEFAULT: PKCS11)
    PKCS11:
      # Location of the PKCS11 module library
      Library:
      # Token Label
      Label:
      # User PIN
      Pin:
      Hash:
      Security:

# Path on the file system where peer will find MSP local configurations
mspConfigPath: msp
```

# Configurações da Rede

Arquivo **configtx.yaml**: Para definir organizações, peers, orders, canal, consorcio e o bloco gênese. (Binário **configtxgen**)

```
#####
#
# CHANNEL
#
# This section defines the values to encode into a config transaction or
# genesis block for channel related parameters.
#
#####
Channel: &ChannelDefaults
  # Policies defines the set of policies at this level of the config tree
  # For Channel policies, their canonical path is
  # /Channel/<PolicyName>
  Policies:
    # Who may invoke the 'Deliver' API
    Readers:
      Type: ImplicitMeta
      Rule: "ANY Readers"
    # Who may invoke the 'Broadcast' API
    Writers:
      Type: ImplicitMeta
      Rule: "ANY Writers"
    # By default, who may modify elements at this config level
    Admins:
      Type: ImplicitMeta
      Rule: "MAJORITY Admins"

  # Capabilities describes the channel level capabilities, see the
  # dedicated Capabilities section elsewhere in this file for a full
  # description
  Capabilities:
    <<: *ChannelCapabilities
```





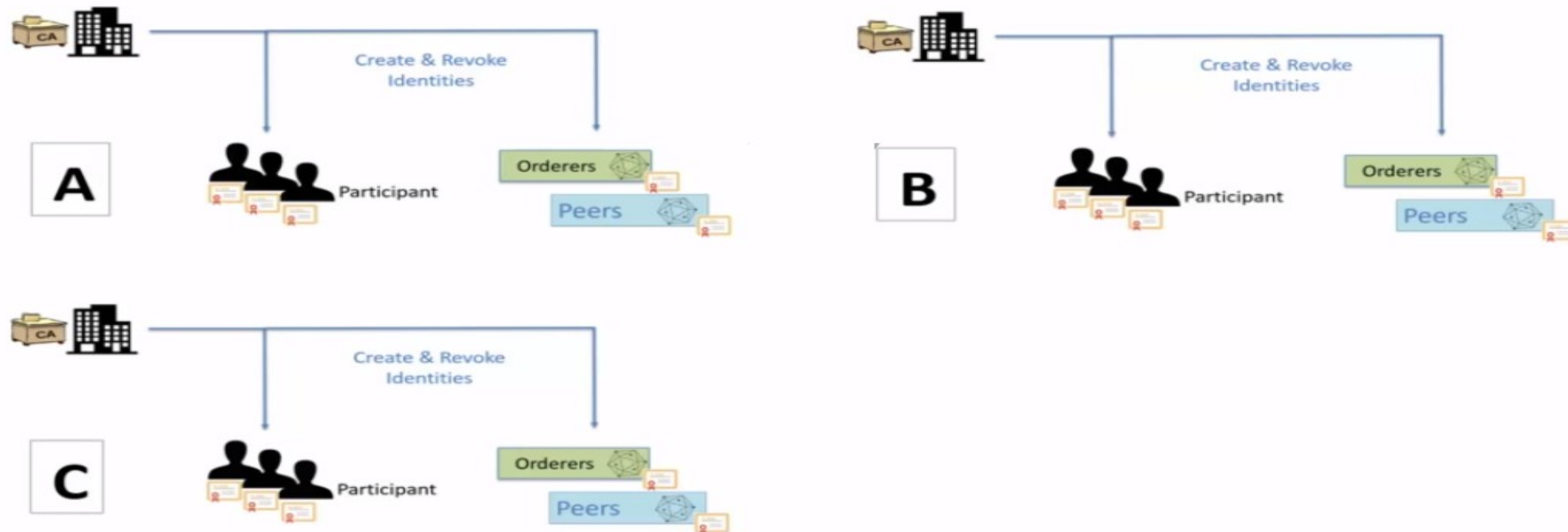
# Membership Service Provider (MSP)



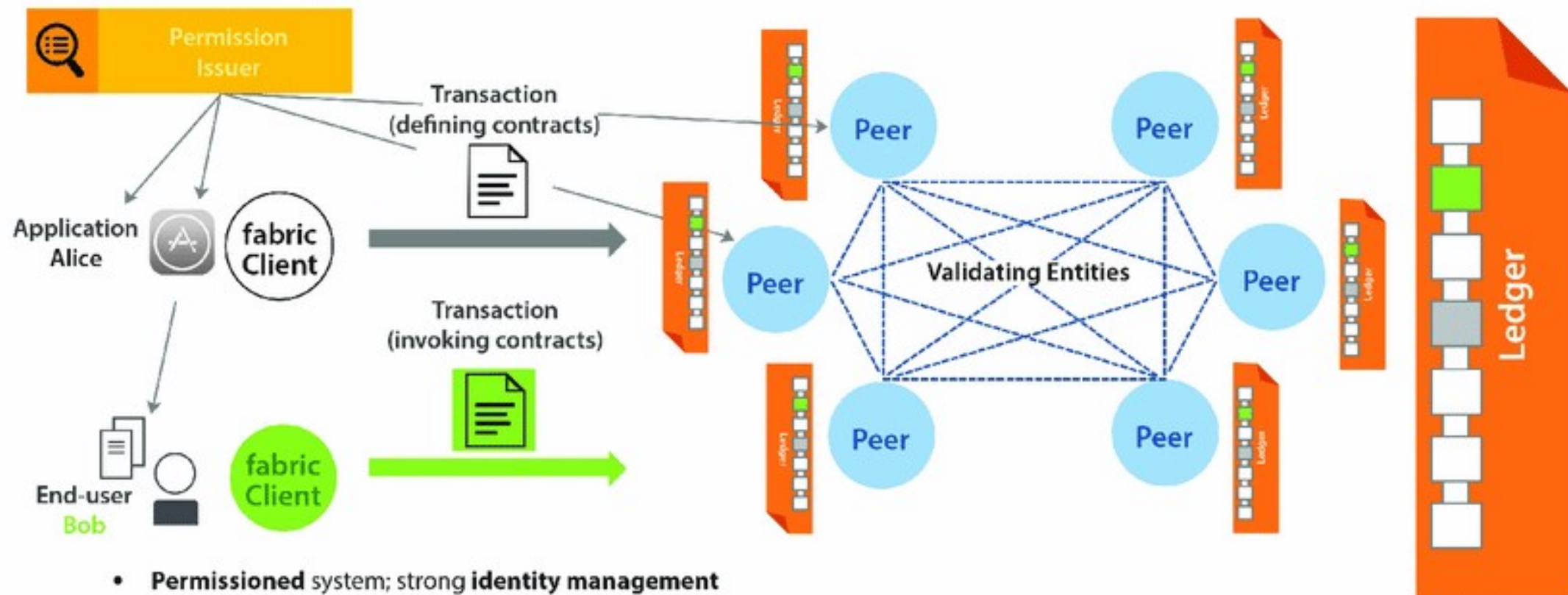
## Decentralized identity management

LET'S RECAP.

- Members manage identities within their organization



# Hyperledger-fabric model



- **Permissioned** system; strong **identity management**
- Distinct roles of **users**, and **validators**
- Users **deploy** new pieces of code (chaincodes) and **invoke** them through **deploy & invoke** transactions
- Validators evaluate the effect of a transaction and reach consensus over the new version of the **ledger**
- **Ledger** = total order of transactions + hash (global state)
- **Pluggable consensus** protocol, currently PBFT & Sieve

# Hands-on





# Links Úteis:

## **Artigo Satoshi Nakamoto**

<https://bitcoin.org/bitcoin.pdf>

[https://bitcoin.org/files/bitcoin-paper/bitcoin\\_pt\\_br.pdf](https://bitcoin.org/files/bitcoin-paper/bitcoin_pt_br.pdf)

## **Hyperledger**

<https://hyperledger-fabric.readthedocs.io/en/latest/index.html>

<https://wiki.hyperledger.org/display/fabric/Hyperledger+Fabric+Roadmap>

# Contatos

## **Email**

[marioaugustosantos@gmail.com](mailto:marioaugustosantos@gmail.com)

## **Linkedin**

[www.linkedin.com/in/mário-augusto-santos](https://www.linkedin.com/in/mário-augusto-santos)