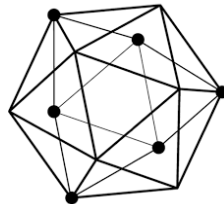


# Módulo 2

## Hyperledger Fabric

# Hyperledger Foundation



## Código Aberto

esforço colaborativo com o objetivo de melhorar as **tecnologias em blockchain** entre diversos segmentos empresariais

Administrado pela **Linux Foundation**, sendo o projeto de maior velocidade de crescimento na história da fundação.

**Colaboração global** entre empresas financeiras, bancos, IoT, logística, indústrias e tecnológicas

# Hyperledger Fabric



Framework de código aberto para criação de redes Blockchain permissionadas

Governança distribuída

Contratos inteligentes (chaincodes) em múltiplas linguagens

Algoritmos de consenso hierárquico

Consenso por contrato inteligente

Não requer cryptomoeda

# Hyperledger Fabric

Redes de negócio (Business network) para múltiplas organizações

Privacidade nativa com channels e private data collection

Cada channel possui um ledger distribuído

Cada rede de negócio pode possuir vários channels

Cliente (api) possui papel fundamental para a operação da rede.

Escalável (nós, orgs, channels, chaincodes)

# Hyperledger Fabric

- Asset – elemento compartilhado na rede
- Node – equipamento da rede
- Ledger – livro razão das transações
- CA – Autoridade Certificadora
- Channel – ledger para comunicação entre partes da rede.
- Chaincode – Contrato Inteligente
- Membership Service Provider (MSP)– identificação dos participantes da rede.
- Consenso – hierarquia da rede.
- Client – entidade que externa que se comunica com a rede blockchain
- Business network

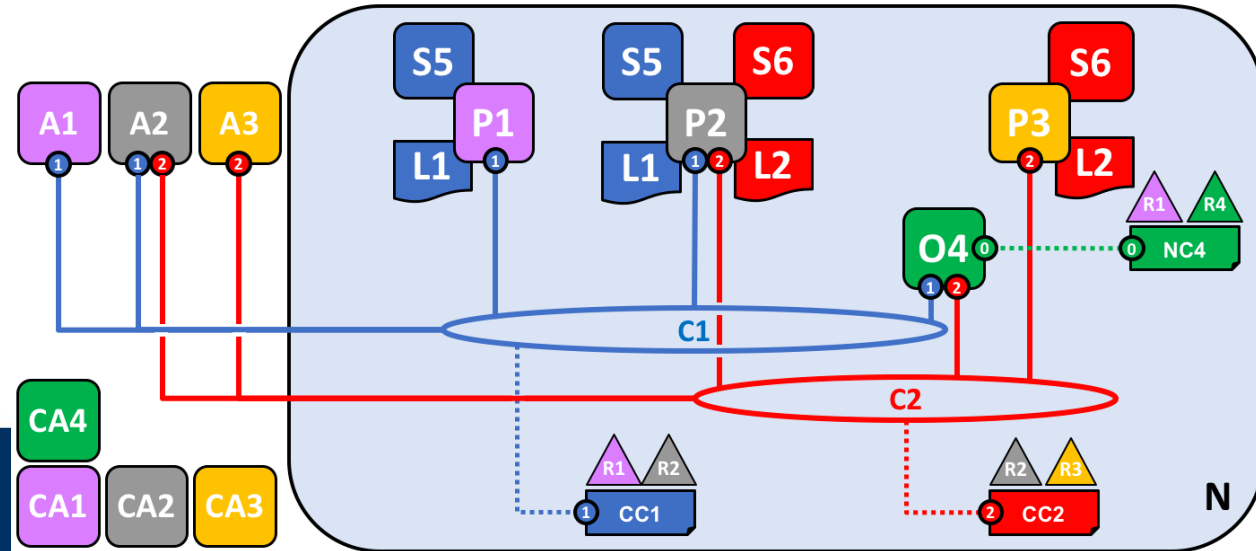
# Exemplo de uma rede Hyperledger Fabric

- **R1..R4:** Organizações 1 a 4
- **C1, C2:** Channels 1, 2
- **P1..P3:** Peers das Orgs 1..3
- **S5, S6:** Chaincodes
- **A1..A3:** Clientes

**CA1..C4:** CAs das Orgs

**L1, L2:** Ledgers 1, 2

**O4:** Ordering Service



# Asset

- Podem ser elementos tangíveis (imóveis, equipamentos) ou intangíveis (documentos contratuais, propriedade intelectual)
- Podem ser **binários** ou **JSON**.
- Elemento presente nos estados da rede.
- Análogo a uma tabela BD
- Criado/modificado/lido dentro de um channel.
- Indexado por uma chave primária ou composta.

# Asset

```
// Description of a book
var Book = assets.AssetType{
  Tag: "book",
  Label: "Book",
  Description: "Book",
  Props: []assets.AssetProp{
    {
      IsKey: true, // Primary Key
      Tag: "title",
      Label: "Book Title",
      DataType: "string",
      Writers: []string{"org2MSP"}, // This means only org2 can create the asset (others can edit)
    },
    {
      Tag: "currentTenant",
      Label: "Current Tenant",
      DataType: "->person", /// Reference to another asset
    },
    {
      Tag: "genres",
      Label: "Genres",
      DataType: "[]string", // String list
    },
    {
      Tag: "published",
      Label: "Publishment Date",
      DataType: "datetime", // Date property
    },
  },
}
```








# Nodes (Nós)

- Nodes, ou nós são os equipamentos que armazenam os ledgers e smartcontracts.
- Existem os seguintes tipos tipos básicos
- **Peer** – armazena uma cópia ledger. Pertence a uma Organization
- **Ordering node** – recebe as transações e executa o consenso e ordena a gravação de novos blocos na rede.

# Peer

- Tipo de node com as seguintes funções:
- Armazenar uma cópia do channel (ledger)
- Executar um chaincode
- Representar uma organização.
- **Anchor peer** – Comunica com outras organizações e orderers para sincronizar o channel.
- **Endorsement peer** – além do channel, armazena o chaincode e executa as transaction proposal provenientes dos HL Fabric Clients.

# Anotomia Peer

Peer		CouchDb ou LevelDb  
MSPs		
Ledger		
Chaincodes(endorsing)		

# Orderer

- Tipo de node com as seguintes funções:
- Armazenar channels de uma rede.
- Pertence a uma organização.
- Identificar e validar as propriedades de um channel (permissões de **leitura, escrita ou administração** para uma organização)
- Identificar os **chaincodes** de um channel (camada de Application) e suas características inclusive o endorsing policy de um chaincode (consenso)
- Receber as pacote de transações assinadas, validar (endorsing policy) e acrescentar a transação em um novo bloco.
- Ordenar as transações recebidas e gerar um novo bloco no channel e enviar para os Anchor Peers.

# Ledger

- Ledger imutável e privado.
- Sequência cronológica de transações.
- Registra transações de configuração ou de aplicação.
- Transações são criadas com operações de criação, atualização e deleção de Assets.

# Channel

- Um channel representa a comunicação entre duas ou mais partes através de um ledger.
- Um channel é definido pelos seguintes elementos:
  - Organizações participantes (members)
  - Peers de comunicação das organizações (anchors peers)
  - Um Ledger
  - Chaincodes que podem modificar ou visualizar o ledger
  - Ordering service nodes

# CA – Certification Authority

- Permite o permissionamento da rede
- Public Key Infrastructure ou PKI
- Certificados Digitais
- Chaves Públicas e Privadas
- Autoridades Certificadoras
- Lista de Revogação de Certificados
- Membership Service Provider ou **MSP**

# MSP – Membership Service Provider

- Membership Service Provider
- Abstração do permissionamento
- **Componentes** criptográficos dos elementos do HL Fabric
- Organizações
- Peers
- Orderers
- Usuários



# Chaincode

- Programa para realizar **propostas** de alteração do estado dos assets.
- Recebe parâmetros de entrada (transaction proposal) e retorna **transaction proposal response**.
- Possui uma política de endosso (*Endorsing Policy*)
- Não atualiza o ledger ou o estado.
- **Linguagens disponíveis**
  - Java
  - NodeJs
  - **GoLang**

# Endorsing Policy

- Regra composta pelas definições das assinaturas necessárias para executar uma transação.
- A *endorsing policy* é atribuída a um *chaincode*
- As assinaturas são obtidas após a execução das transações pelos *chaincodes* dentro dos *endorsing peers*.
- A sintaxe da endorsing policy é uma regra lógica entre as organizações.
- Exemplos:
  - `AND('Org1', 'Org2', 'Org3')`
  - `OR(AND('Org1', 'Org2'), AND('Org1', 'Org3'), AND('Org2', 'Org3'))`

# Client

- Elemento que passa as entradas de dados do mundo exterior.
- Se comunica com o Blockchain através de um peer.
- Utiliza um SDK.
- **SDKs** disponíveis:
- Java
- Node
- GoLang

# Private Data Collection

- Um *channel* pode conter informações acessíveis apenas a certas orgs.
- Essas informações ficam gravadas externamente ao *peer* dentro de bases do *CouchDb* em um banco de dados transiente.
- Dentro do *ledger* ficam registradas apenas os hashes das informações.
- Esse conceito é chamado de ***Private Data Collection***.
- As organizações que podem acessar a informação privada ou alterada são definidas em uma *Private Data Collection Policy*.
- Uma Private Data Collection pode ter tempo de vida.

# Business Network

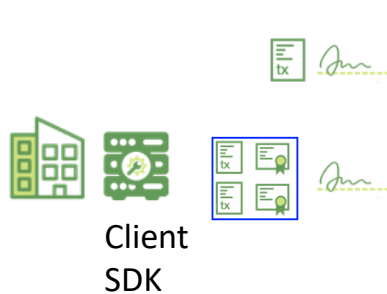
- Os participantes de uma Business Network podem transacionar entre si utilizando componentes Hyperledger Fabric.
- Conjunto de:
  - CAs
  - Channels (peers, chaincodes, orgs, MSPs)
  - Orderers
  - Clients (APIs)



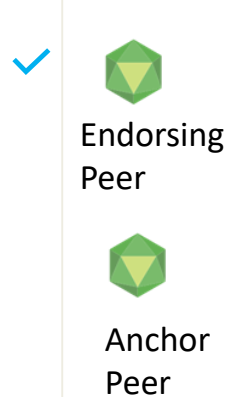
# **Como o Hyperledger Fabric funciona?**

## **Transaction Flow**

## ORGANIZAÇÃO A



## ORGANIZAÇÃO B



## ORGANIZAÇÃO C (ORDERER)



# 1. Client cria uma transação e envia para os endorsing peers

- CLIENT cria uma PROPOSE MESSAGE
- PROPOSE MESSAGE deve ser enviada para os ENDORSING PEERS necessários para o ENDORSING POLICY do CHAINCODE.
- **PROPOSE MESSAGE** contém:
  - clientID
  - chaincodeID
  - txPayload
  - Timestamp
  - clientSig



## 2. Endorsing peer simula uma transação e assina digitalmente

- Utilizando as informações do payload da PROPOSE MESSAGE, ENDORSING PEER executa o CHAINCODE.
- Resultado do CHAINCODE é assinado digitalmente pelo ENDORSING PEER, gerando uma transação assinada.
- ENDORSING PEER retorna a transação assinada para o CLIENT.

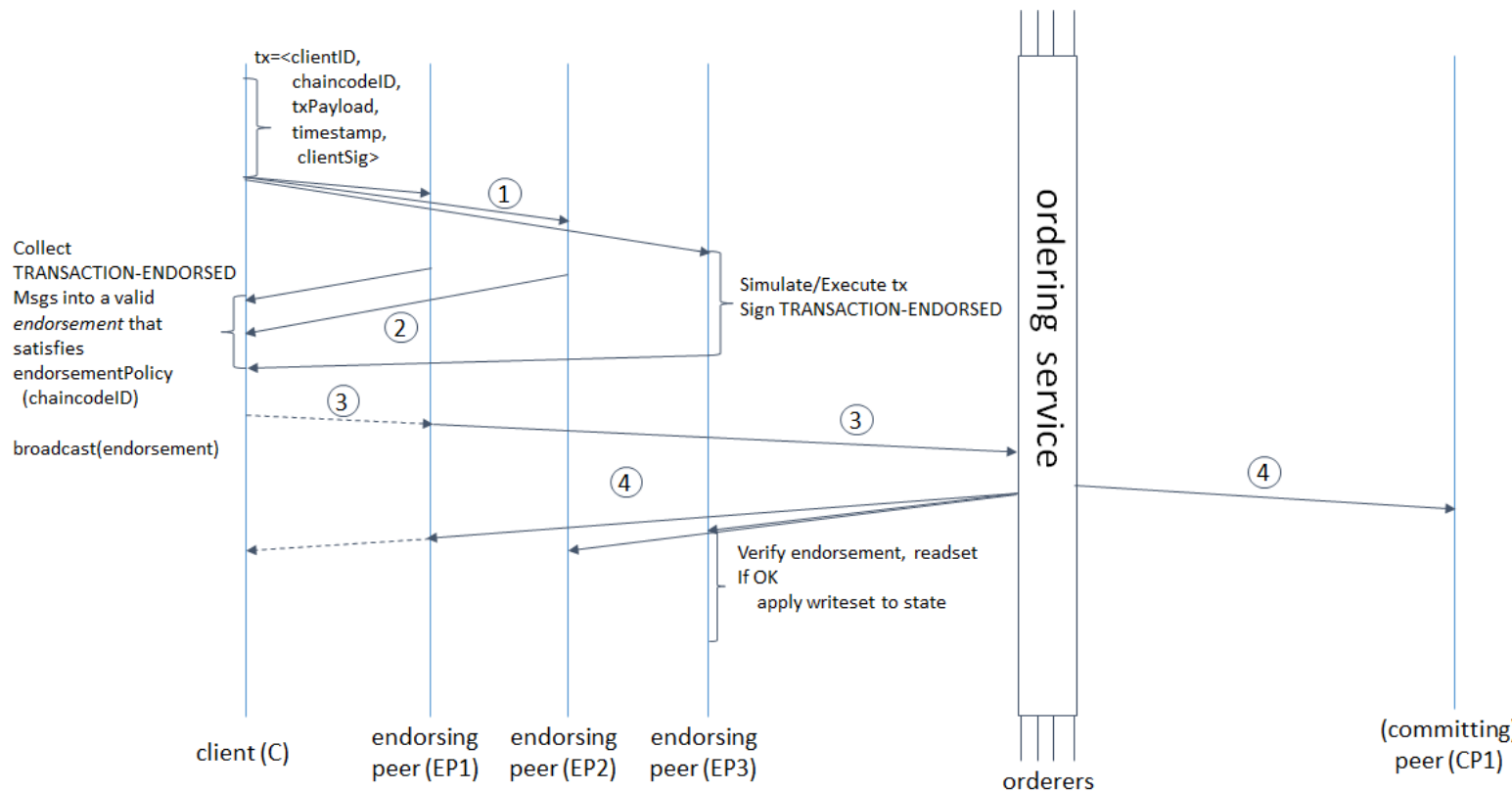
### 3. Client recebe as transações assinadas e envia para o Ordering Service

- Todas as transações assinadas pelos ENDORSING PEERS são recebidas pelo CLIENT.
- A transações assinadas geram um pacote de ENDORMENT.
- O ENDERSOMENT é então enviado para o ORDERING SERVICE.

## 4. Ordering Service entrega o bloco para os peers

- Após receber o ENDORSEMENT, o ORDERING SERVICE valida a transação de acordo com o ENDORSING POLICY do CHAINCODE.
- Caso todos os ENDORSING PEERS necessários para a execução do CHAINCODE tenham assinado a transação, essa é entregue pelo ORDERING SERVICE a todos os PEERS.
- Todos os PEERS passam a ter um novo STATE

# Fluxo de uma transação Hyperledger Fabric





## **Exemplo de Fluxo**

# A vende legumes para B

- Client A compra e vende legumes do Client B
- Consenso (Endorsing Policy) do chaincode estabelece que ambos A e B devem assinar as transações.
- A e B possuem um peer cada um para enviar as transações:
- Pedido de compra de legumes
- Aceite de compra de legumes
- Rejeição de compra de legumes



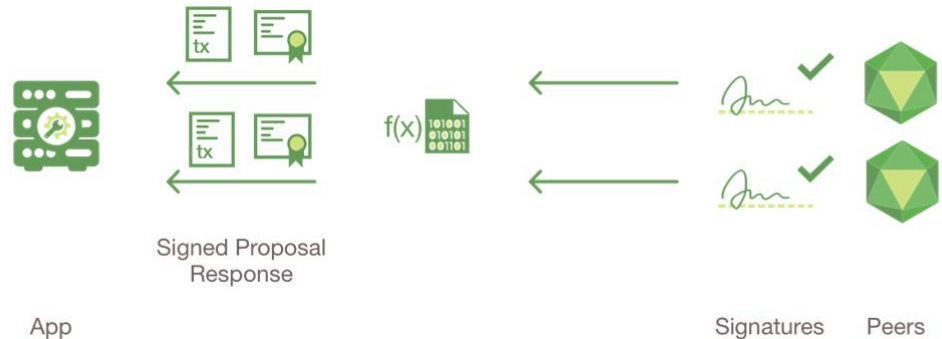
# 1. Client inicia uma transação

- Client A se comunica com um endorsing peer através de um SDK.
- Um proposta de transação (transaction proposal) é criada no Client A e enviada para os endorsing peers A e B.



## 2. Endorsing peer verifica assinatura e executa a transação

- Endorsing peers verificam:
- Formato da transação
- Transação não foi realizada anteriormente
- Client está autorizado pedir a transação.
- Chaincode é executado dentro do Endorsing Peer
- Endorsing peer retorna a transação para o Client





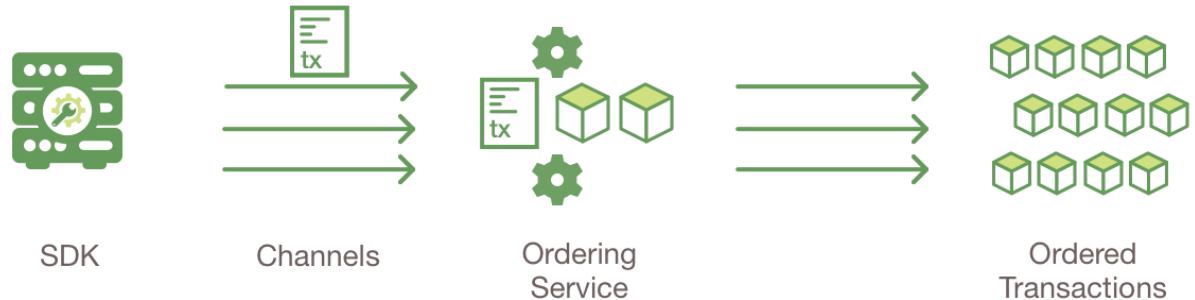
### 3. Client verifica as respostas

- Client A recebe as respostas de todos os Endorsing Peers A e B
- Client A verifica se as respostas dos Chaincodes executadas em A e B são iguais.



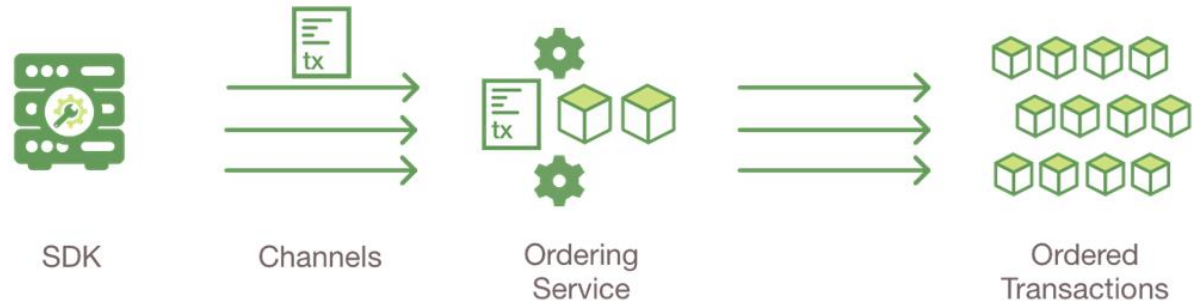
## 4. Client monta a transação de endorsement

- Client A monta a transação de Endorsement
- Client A envia a transação para o Ordering Service.
- Ordering Service valida a transação de acordo com o Endorsing Policy.
- Transação é agrupada em um Bloco, em ordem cronológica para ser acrescentada no Channel



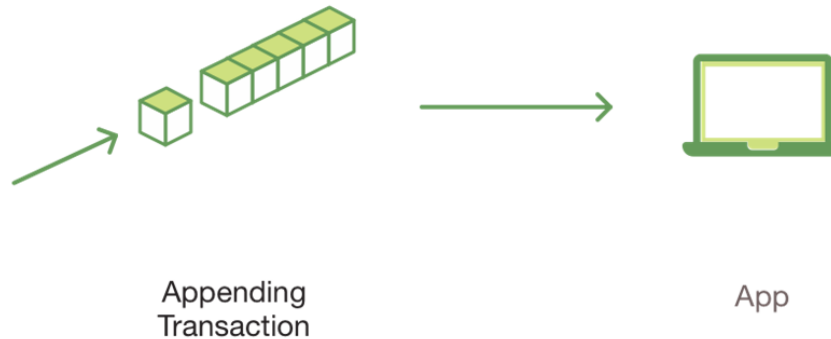
## 5. Transação é validada e agrupada em um Bloco

- Client A monta a transação de Endorsement
- Client A envia a transação para o Ordering Service.
- Ordering Service valida a transação de acordo com o Endorsing Policy.
- Transação é agrupada em um Bloco, em ordem cronológica para ser acrescentada no Channel

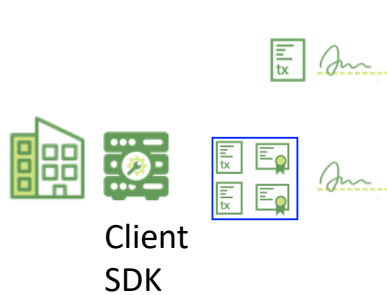


## 6. Ledger atualizado

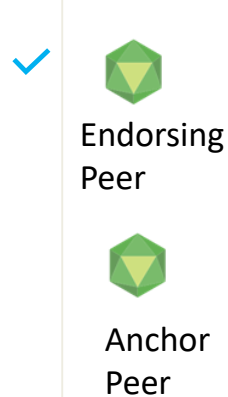
- Os Peers atualizam o seu Ledger.
- Um evento é gerado para o Client A informando do sucesso do registro da transação.
- O Ledger se encontra em um novo World State



## ORGANIZAÇÃO A



## ORGANIZAÇÃO B



## ORGANIZAÇÃO C (ORDERER)



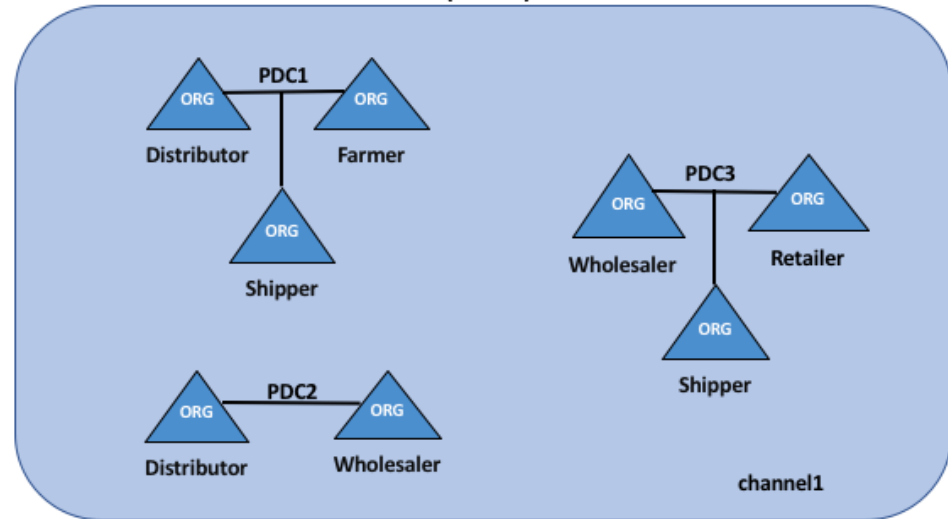
# Private Data Collections

- Caso o transação de um *chaincode* faça uso de PDC, o fluxo da transação possui etapa extra para geração de um TRANSACTION PROPOSAL RESPONSE.
- Transação possui leitura ou escrita em um PDC então:
- Gravação da informação privada gravada em base de *state* separada dentro do *CouchDB*.
- Replicação das informações privadas entre *peers* das organizações.
- PDC possui um Policy para definir as orgs que podem ter acesso a informação privada.
- *Transaction flow* de uma transação que usa PDC é mais complexo.

# Exemplo de um channel com PDCs

- PDC1: Distributor, Farmer and Shipper
- PDC2: Distributor and Wholesaler
- PDC3: Wholesaler, Retailer and Shipper

Private data collections  
(PDC)



# Tarefa

**PRÓXIMO MÓDULO:**

**3**

**Hyperledger Fabric  
(parte 2)**



**GoLedger**