

Tutorial

# BLOCKCHAIN PARA PYTHONISTAS

DO ZERO AO SMART CONTRACT

Apresentação:

Maria do Rosário e Pedro Felipe



# PYTHON NORDESTE 2025

# SUMÁRIO



# Doc com todos os links necessários

# Parte I: Um resumo conceitual

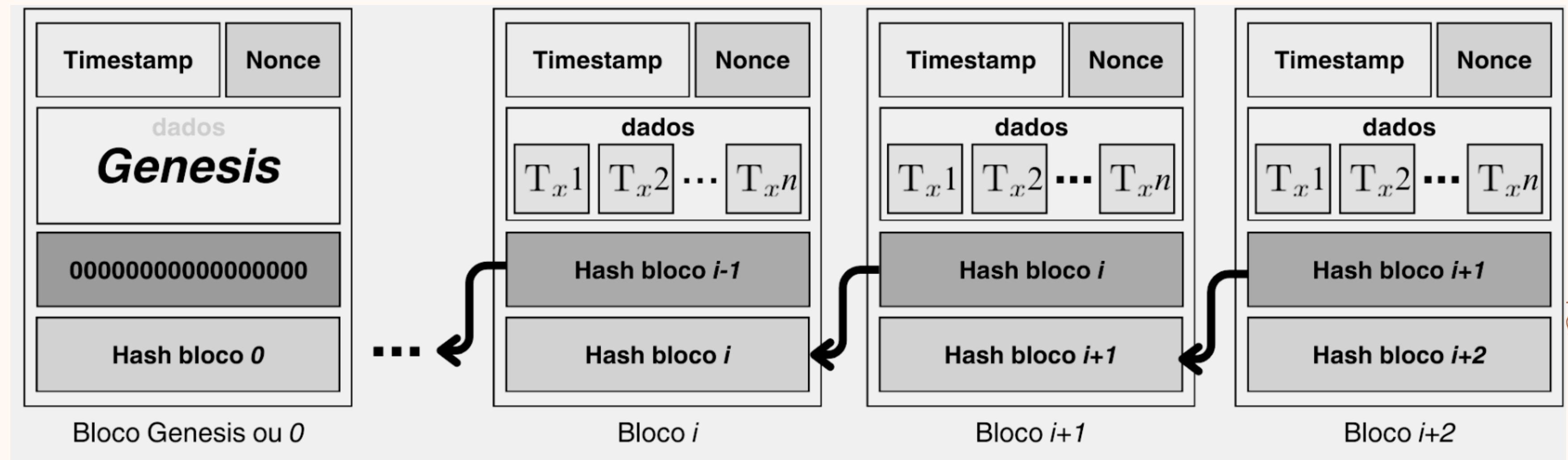
# O QUE É BLOCKCHAIN?

**Acima de qualquer coisa, uma blockchain é uma plataforma de registro de transações.**

- Surgiu para dar suporte à primeira criptomoeda de sucesso: O Bitcoin (Um Sistema de Dinheiro Eletrônico Peer-to-Peer (NAKAMOTO, 2008)).
  - Uma transação mostra a **movimentação** de propriedade de ativos digitais **de um participante para outro** em uma rede blockchain.
  - Junto ao ativo/token, podem ser transferidos outros tipos de dados.
  - Uma **blockchain armazena** todas as suas transações em um **registro estruturado** como uma lista (“cadeia”) de grupos (“blocos”) de transações = block+chain.

## Parte I: Um resumo conceitual

# O QUE É BLOCKCHAIN?



Representação de como transações (campo dados) são armazenadas no registro. **Timestamp** marca data e hora de inserção do bloco à cadeia. **Nonce** é um número único para cada bloco. Cada bloco é conectado ao anterior através de **hashes**.

## Parte I: Um resumo conceitual

# O QUE É BLOCKCHAIN?

- Blockchain é um registro (**livro-razão**) digital **compartilhado, distribuído e à prova de violações** que consiste em dados de transações organizados cronologicamente em um pacote chamado bloco (Kakavand et al., 2017)
- Este livro-razão, que é **transparente e imutável**, pode rastrear todas as transações de seus ativos em uma rede de negócios.
- Um **ativo** é qualquer **coisa com valor** que possa ser **transferida na blockchain**. Eles podem variar do tangível ao intangível e do fungível (indiferenciável) ao não fungível (NFTs) .

## Parte I: Um resumo conceitual

# O QUE É BLOCKCHAIN?

- Uma blockchain na verdade é uma **composição complexa** de várias camadas de tecnologias distintas.
- Segundo Kakavand et al. (2017) em “A Revolução Blockchain”, a tecnologia já foi descrita como sendo, ao mesmo tempo, uma **rede distribuída** e um **banco de dados** equipado com segurança integrada.
- Entretanto, a chegada da rede Ethereum (Buterin, 2013) e a concomitante formalização da tecnologia (Wood, 2019) endossaram a visão de Blockchain também como **uma máquina de estados**.
- Uma blockchain é, ao mesmo tempo, todas essas coisas. E um pouco mais.

# Parte I: Um resumo conceitual

# ELEMENTOS FUNDAMENTAIS

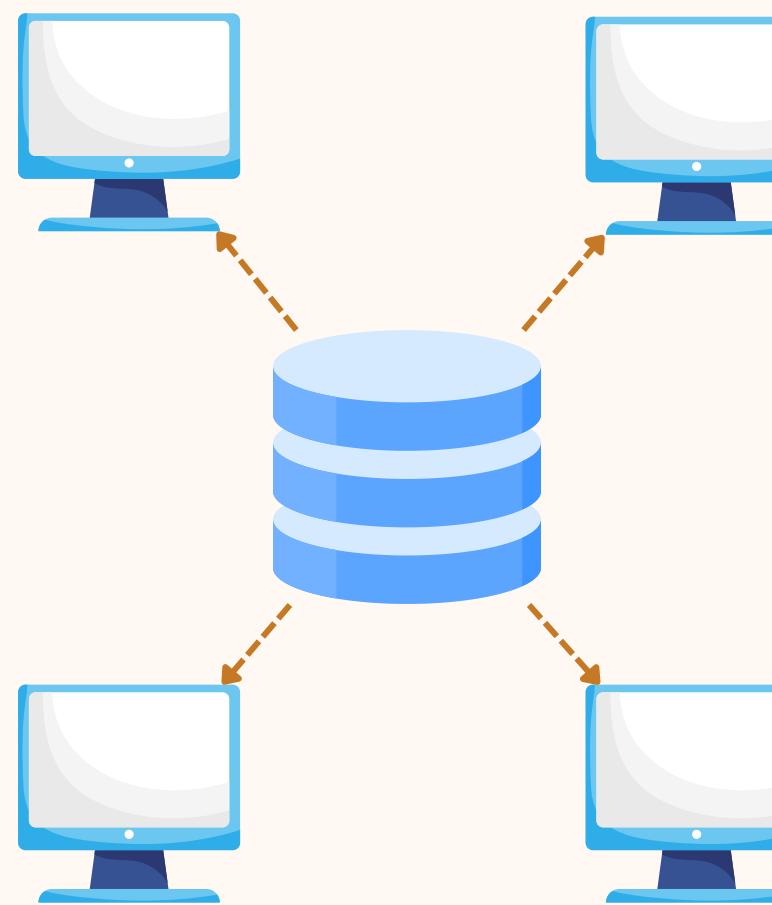
- Uma boa maneira de entender a combinação complexa que é a blockchain é compreender os elementos fundamentais que a compõem.

<b>Livro-Razão</b>	Transparência, Auditabilidade, Imutabilidade e Irrefutabilidade
<b>Distribuição</b>	Descentralização, Replicação e Disponibilidade
<b>Consenso</b>	Desintermediação, Atualidade e Integridade
<b>Criptografia</b>	Identidade, Assinaturas Digitais, Segurança e Privacidade

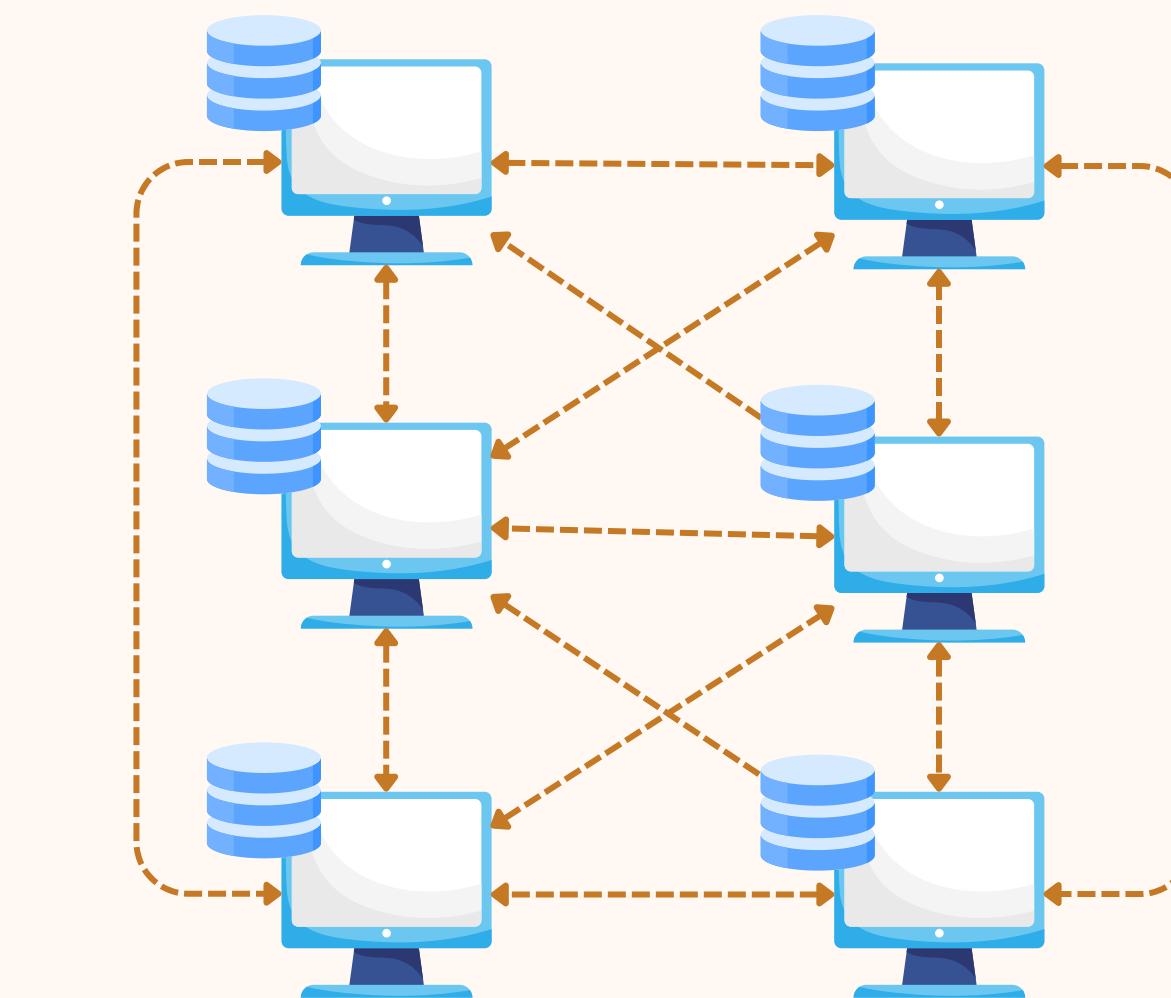
## Descentralização

- Blockchain é um sistema **descentralizado**, mantido por **uma rede peer-to-peer** de nós independentes, cada um mantendo uma cópia do registro. Diferente de um servidor centralizado, mais de um nó pode inserir dados na blockchain.

Base de dados centralizada



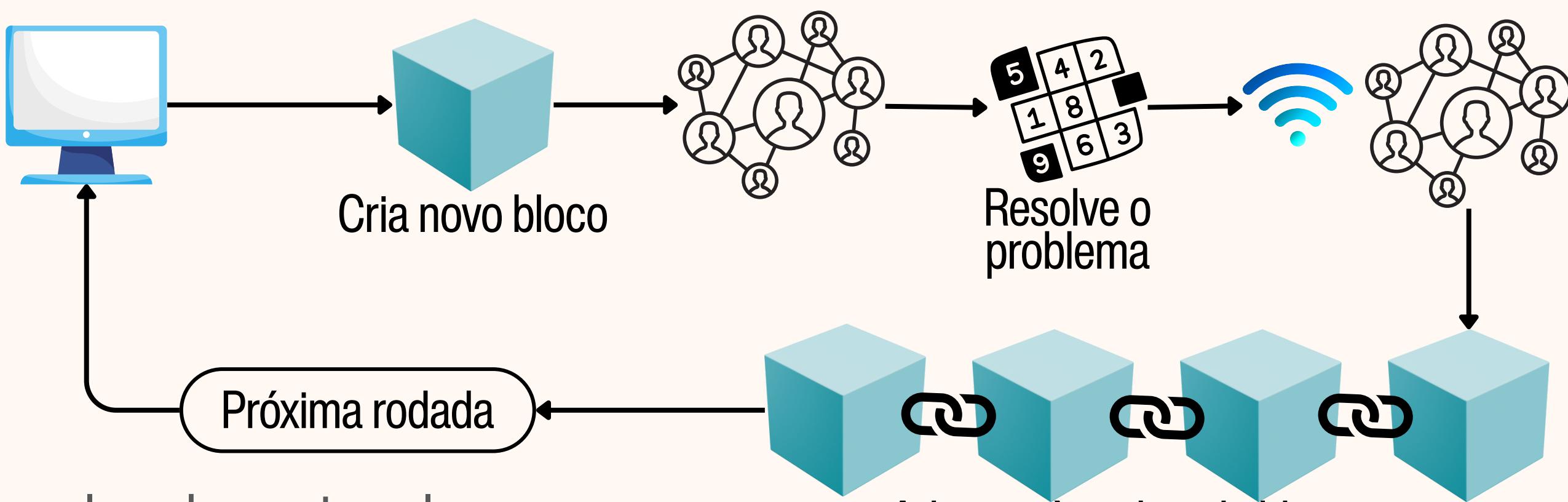
Base de dados da Blockchain



- A descentralização na blockchain é mais do que física.

# Consenso (e Mineração)

- Por ser um sistema distribuído, a blockchain demanda coerência e sincronização entre seus nós. O protocolo de consenso é responsável por indicar a fonte de verdade do ledger e pela ‘eleição’ do próximo bloco a ser adicionado à cadeia.

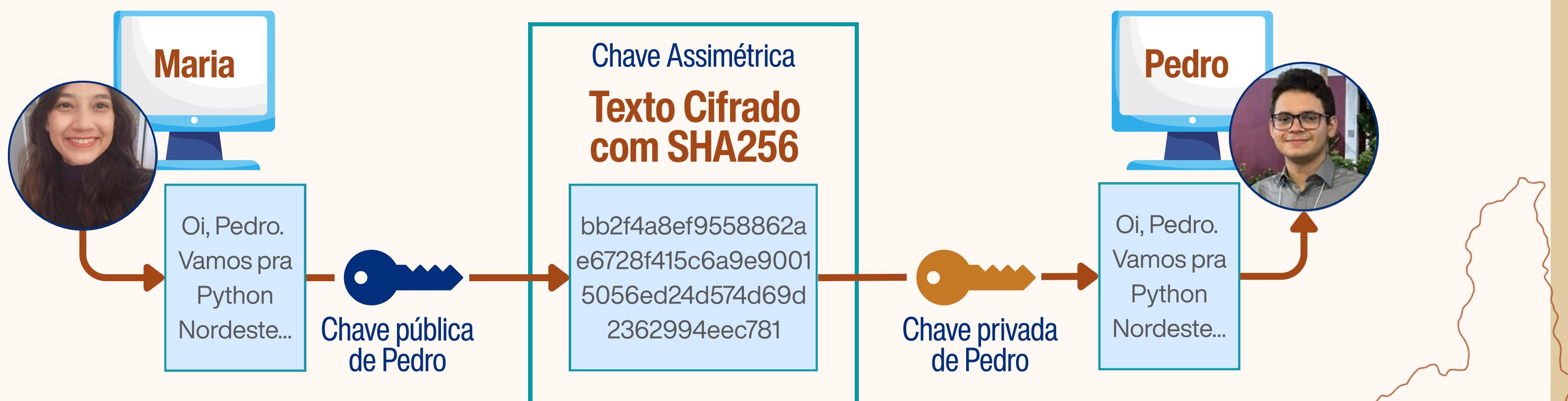


# Alguns exemplos de protocolos:

- Proof of work;
  - Proof of Stake;
  - Proof of Authority;

# Criptografia

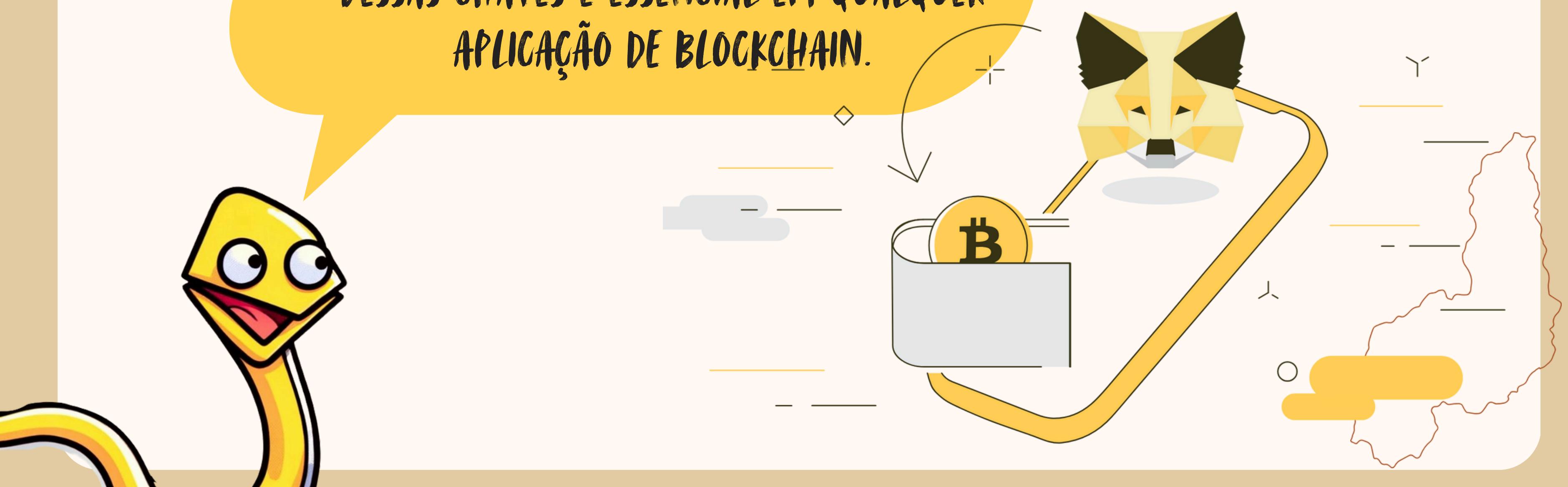
- A Blockchain apoia-se fortemente na Criptografia para satisfazer os requisitos de segurança do sistema e das aplicações. Dentre os recursos mais utilizados, destacam-se os **resumos criptográficos (funções hash)** e as assinaturas digitais (criptografia de **chaves assimétricas**).



- Toda transação com a blockchain precisa ser assinada.

## Criptografia

CADA CONTA BLOCKCHAIN POSSUI UM PAR DE CHAVES PÚBLICA-PRIVADA E O GERENCIAMENTO DESSAS CHAVES É ESSENCIAL EM QUALQUER APLICAÇÃO DE BLOCKCHAIN.



## Tutorial: MetaMask

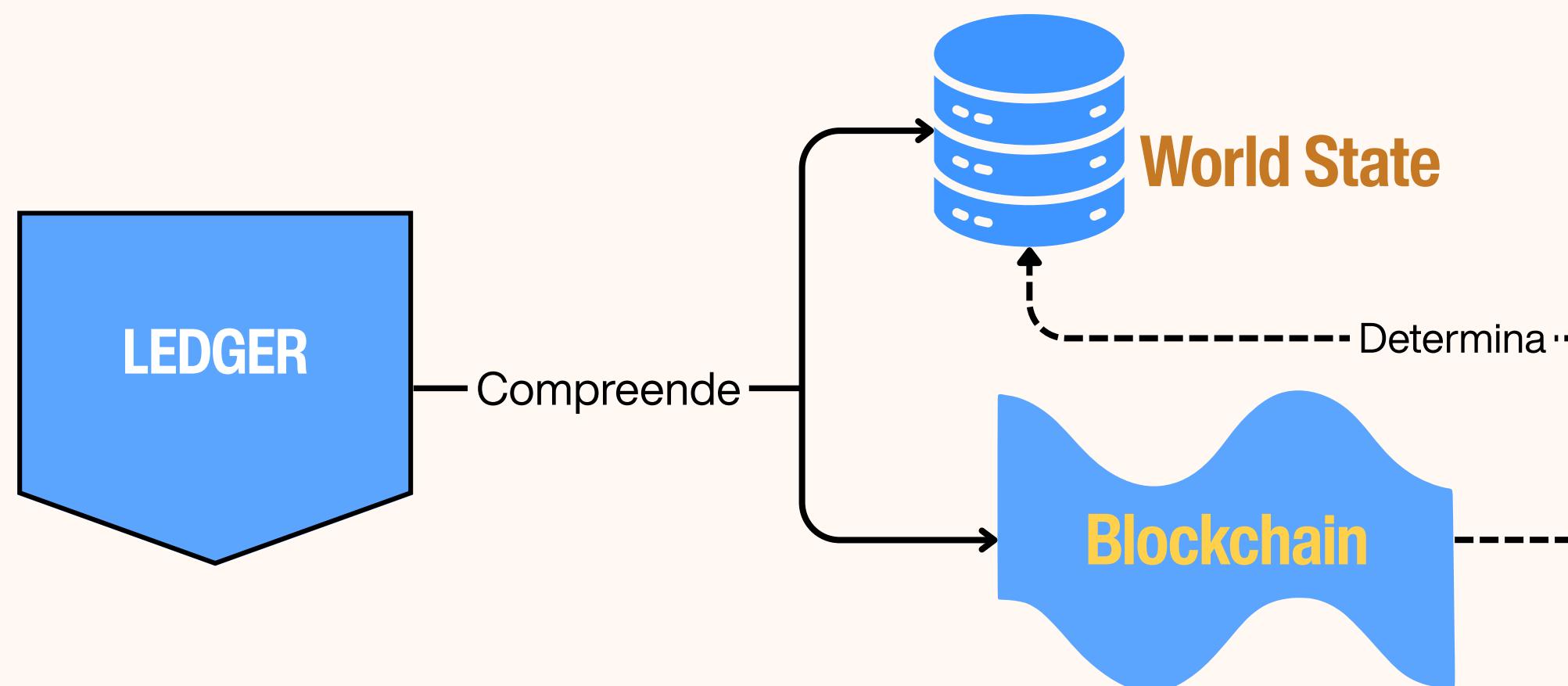
**Objetivo:** Criar uma carteira na MetaMask e adicionar saldo (faucet);

- 1. Fazer login em uma conta google;**
- 2. Acessar <https://metamask.io/>**
- 3. Criar uma carteira com senha que você possa lembrar depois;**
- 4. Por efeito de teste, pule a etapa de segurança;**
- 5. Mude a rede para Sepolia;**
- 6. Acessar <https://cloud.google.com/application/web3/faucet/ethereum/sepolia>**
- 7. Copiar chave pública de sua carteira e usar o Faucet para carregar os tokens;**

## Ledger: o livro-razão

No coração de uma blockchain, o **livro-razão, ou registro**, é um elemento composto por dois componentes interligados:

- **Estado Global (World State)**: um banco de dados que contém o estado atual da rede;
- **blockchain**: é um registro de todas as transações que levaram a esse estado, organizadas em uma estrutura de dados baseada em uma cadeia de blocos.

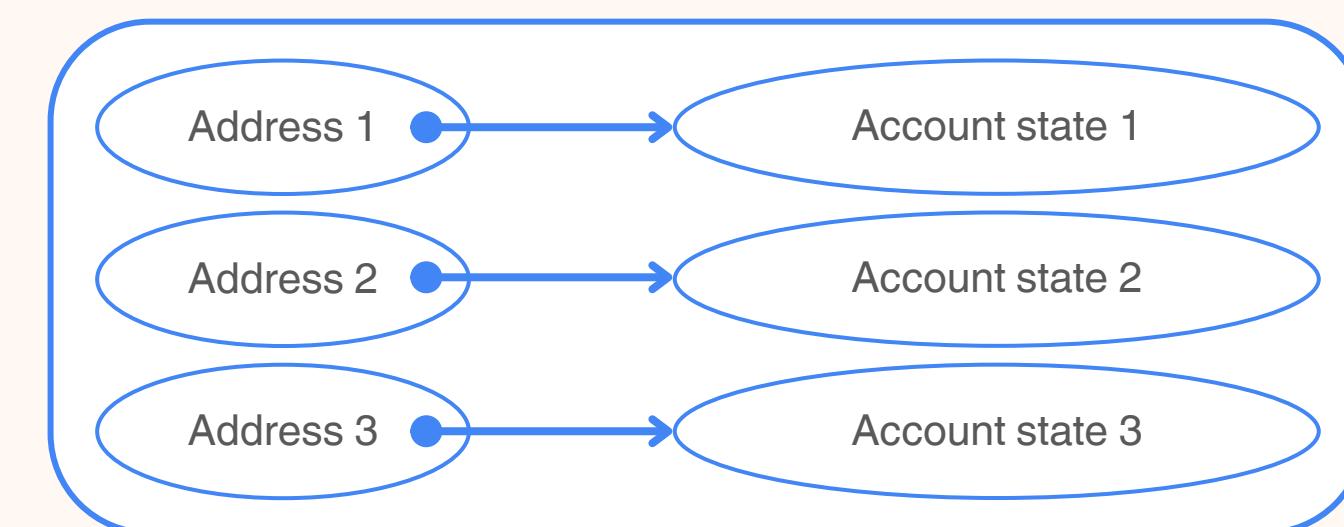


# Ledger: o livro-razão

Nesse contexto, **estado global** em blockchain se refere a um **mapeamento de endereços válidos e dados atuais** (estado) de contas (accounts).

No caso da Bitcoin, é fácil perceber que o ‘estado’ se refere ao saldo total de bitcoins que aquela conta possui.

Já na blockchain Ethereum, o **estado** é **mais complexo** e essa diferença é fundamental para a viabilidade dos contratos inteligentes.



# Mapping view

Address 1	Account state 1
Address 2	Account state 2
..	..
Address n	Account state n

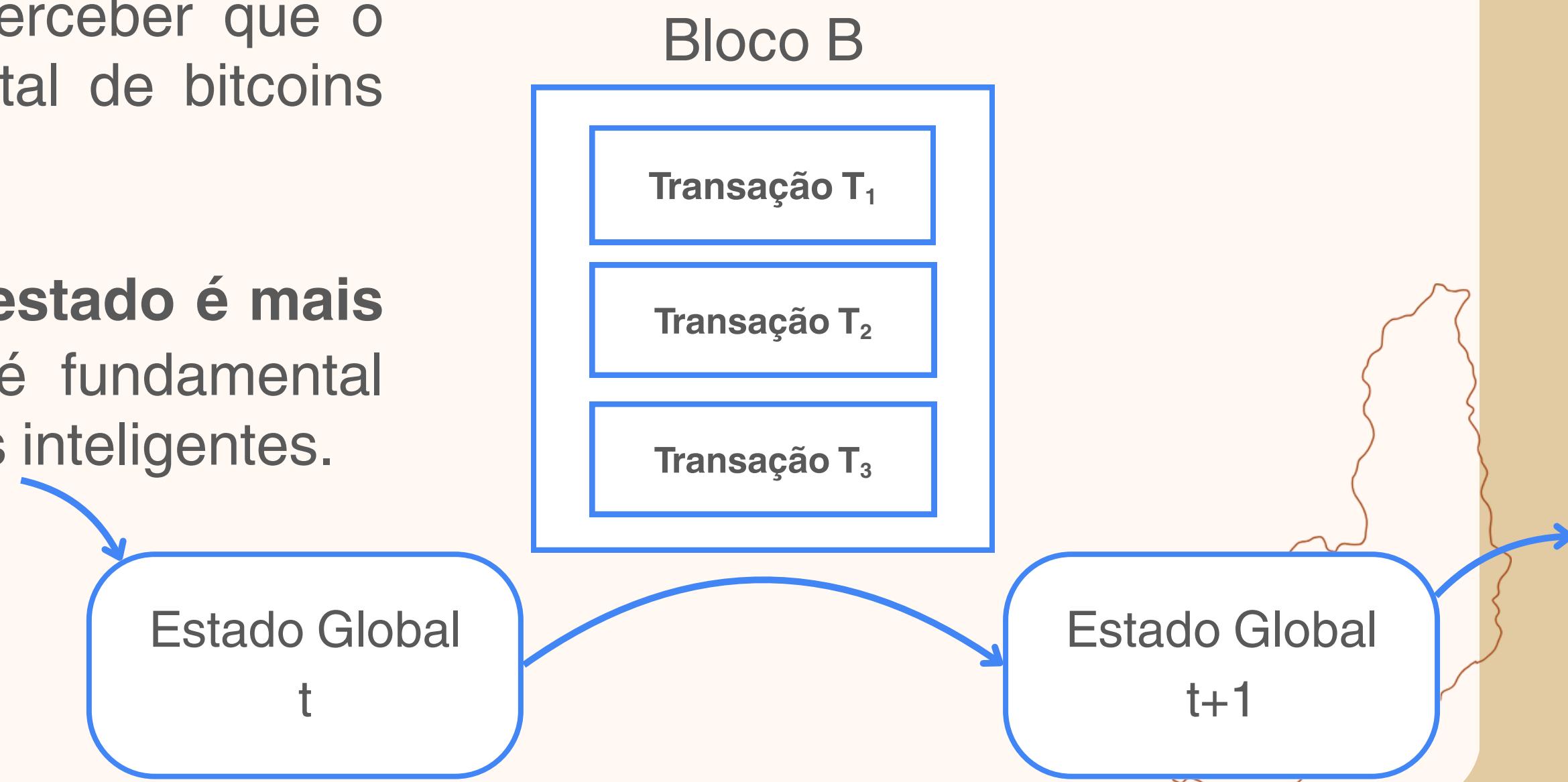
# Table View

# Ledger: o livro-razão

Portanto, uma **blockchain** pode ser vista como uma **cadeia de estados**, em que cada **nova transação** e, portanto, cada **novo bloco** representa uma **transição de estados**.

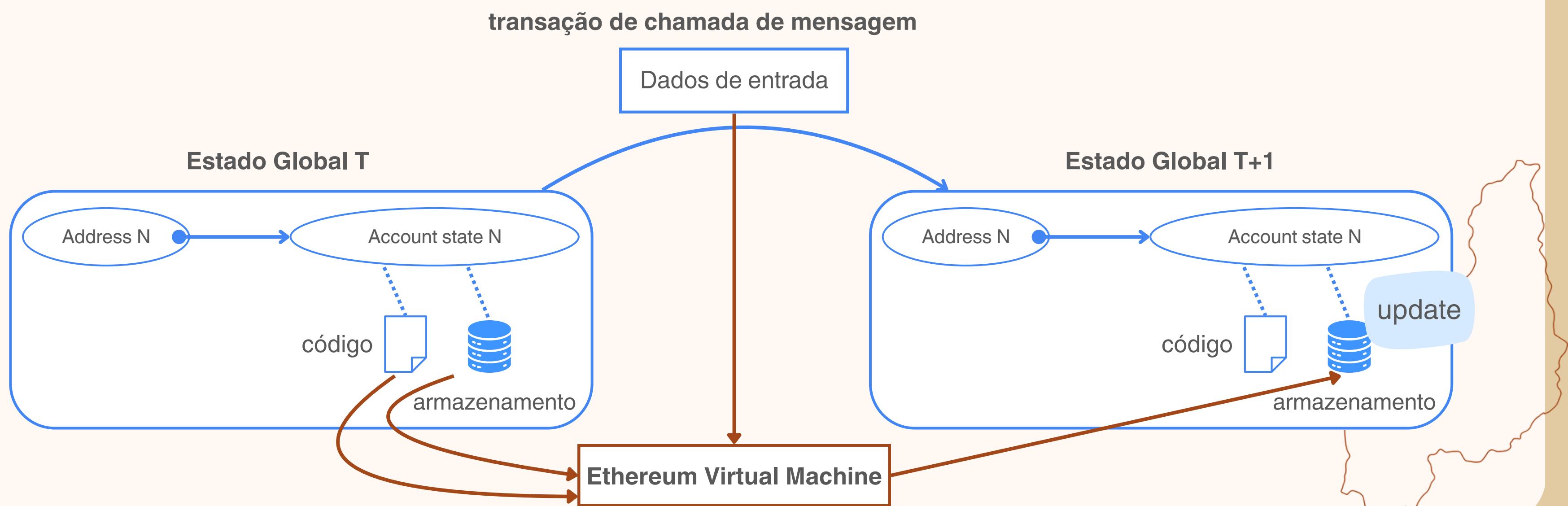
No caso da Bitcoin, é fácil perceber que o ‘estado’ se refere ao saldo total de bitcoins que aquela conta possui.

Já na blockchain Ethereum, o **estado** é mais **complexo** e essa diferença é fundamental para a viabilidade dos contratos inteligentes.



# Estados e EVM

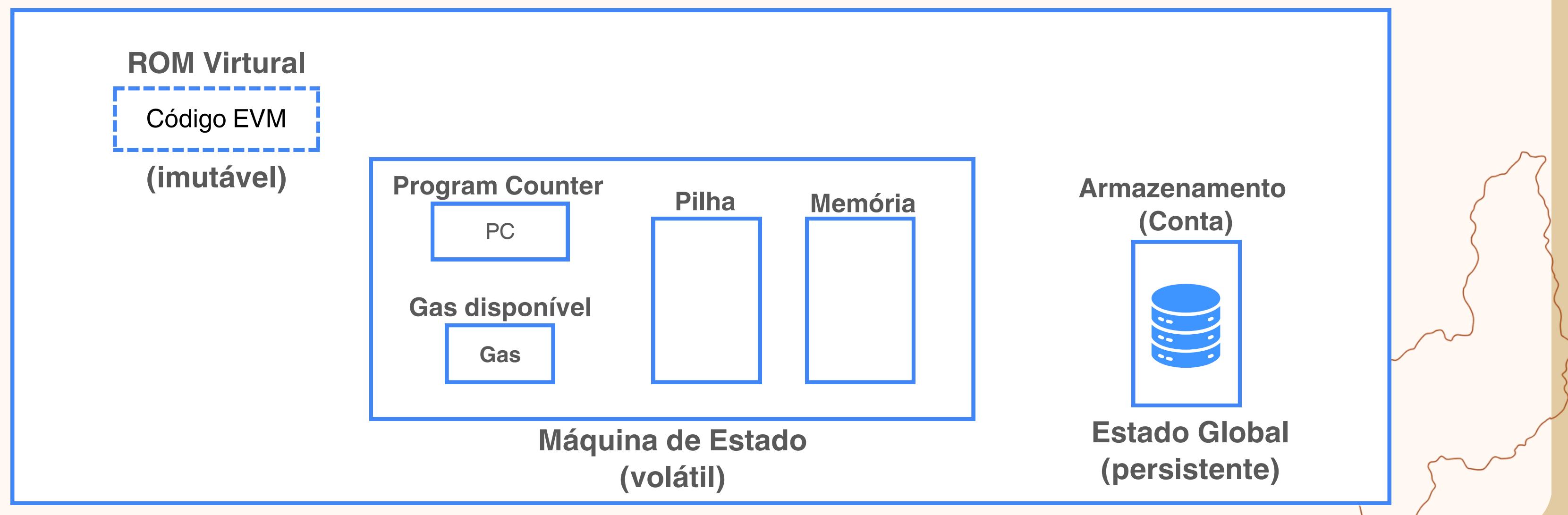
Isso porque a rede Ethereum trouxe consigo uma alteração no protocolo original da blockchain: a **Ethereum Virtual Machine**. De forma que o estado aqui não se refere apenas a saldos, mas também a um **estado de máquina**, que é aceito (por consenso) e copiado por todos os nós da rede.



## Estados e EVM

Ethereum é, portanto, uma blockchain equipada com um computador built-in, além de uma linguagem de programação completa e integrada, capaz de criar programas executáveis complexos (**contratos inteligentes**).

### Ethereum Virtual Machine



## Contas - Accounts

Existem dois tipos de contas:

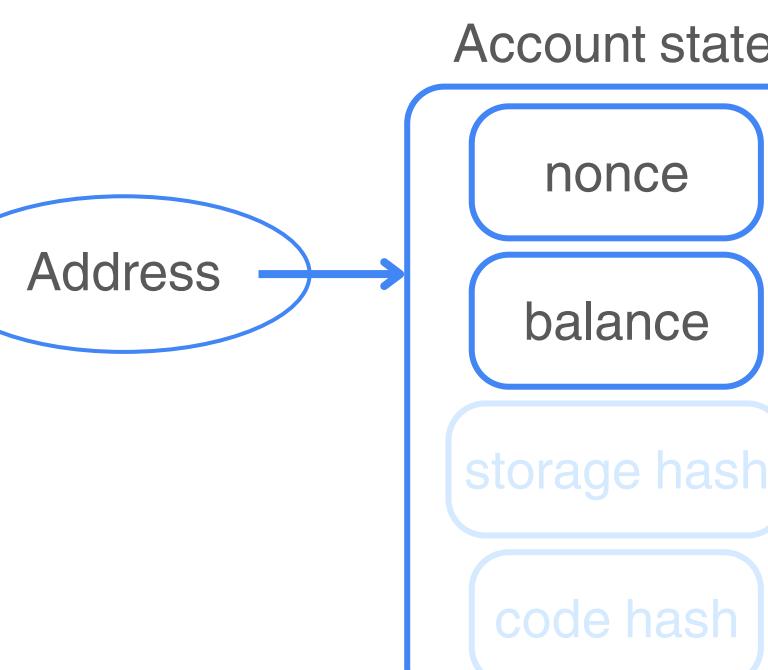
**Contas de atores externos** - são contas pessoais e que dependem da assinatura do dono, através de chave privada.

**Contas de contratos** - são objetos autônomos, contratos inteligentes armazenados na blockchain e que contém código EVM.

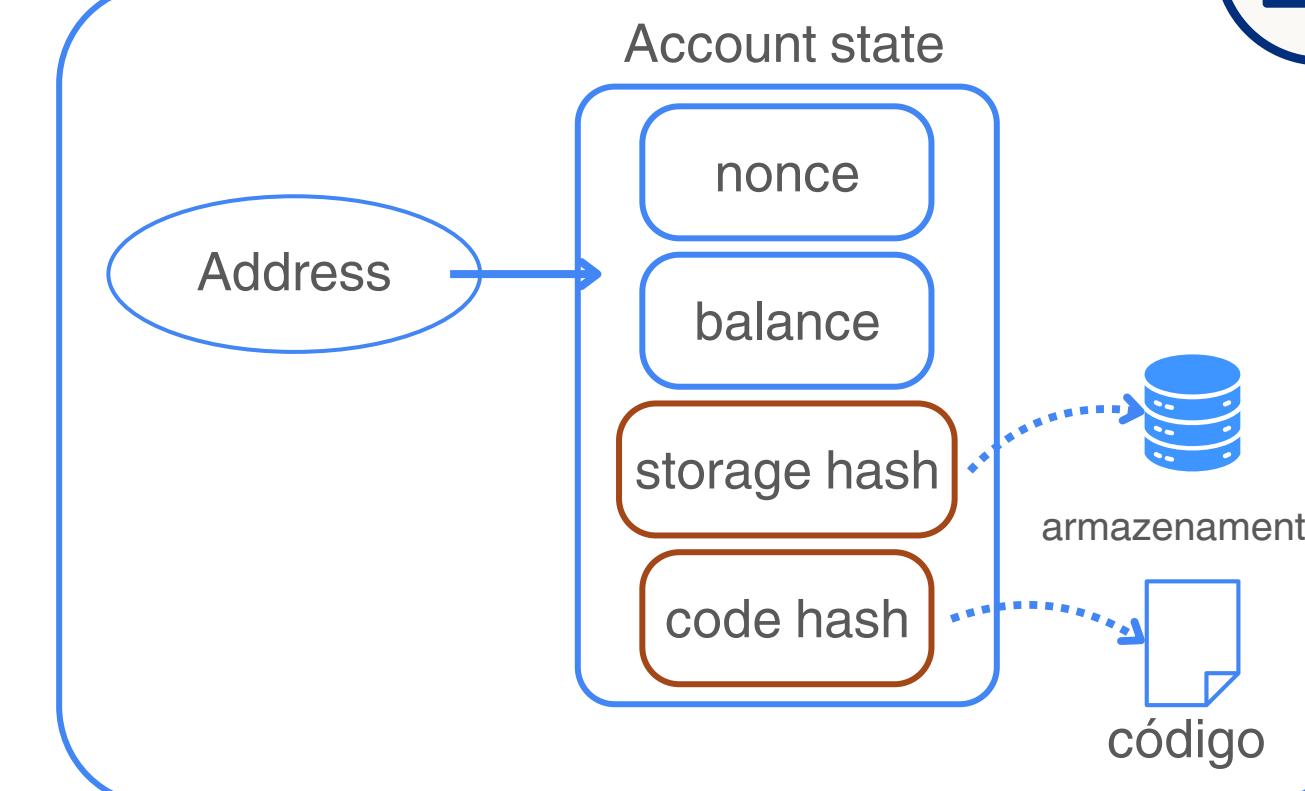


### Estado Global

#### Externally Owned Account (EOA)



#### Account Contract

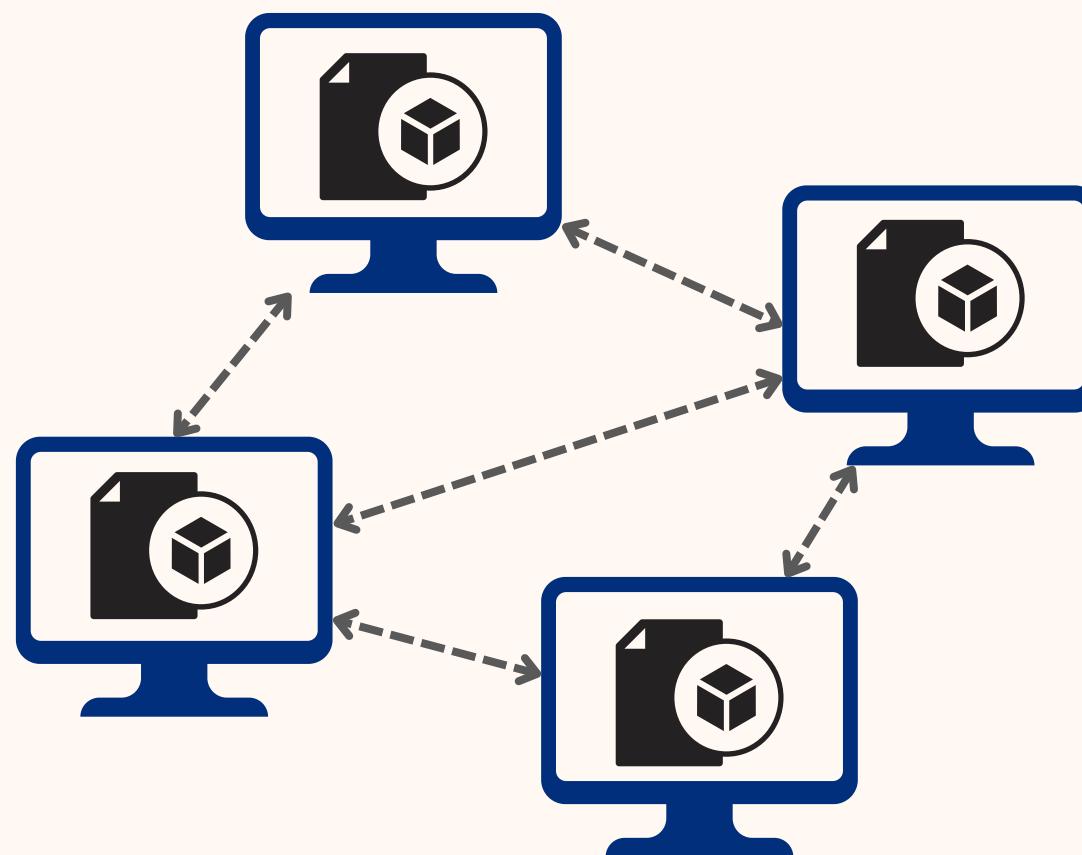


# Na Prática

# COMO INTERAGIR COM A BLOCKCHAIN?



# **Passo 0: Antes de tudo, é preciso escolher a infraestrutura.**



- Essa escolha depende do objetivo, do escopo e das necessidades do caso de uso; Também pode depender dos recursos disponíveis;
  - Blockchains podem ter infraestrutura **pública, privada ou híbrida**. O que as diferencia são as permissões de participação na rede, escrita e leitura, além dos custos, latência e outros aspectos;

## Infraestrutura



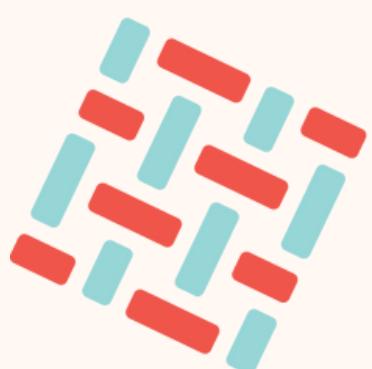
# COMO INTERAGIR COM A BLOCKCHAIN?

- As redes **Bitcoin** e **Ethereum mainet** são exemplos de **redes públicas**, ou seja, **qualquer usuário** pode executar funções de **leitura e escrita**. Em uma blockchain **não-permissionada**, não há uma autoridade central que controle o acesso ou que possa banir usuários ilegítimos.
- Por outro lado, **redes privadas** oferecem a capacidade de restringir o acesso a um grupo específico de participantes, garantindo maior **privacidade e controle** sobre o livro razão. Em ambientes corporativos/industriais, esse padrão é, em geral, mais seguro.

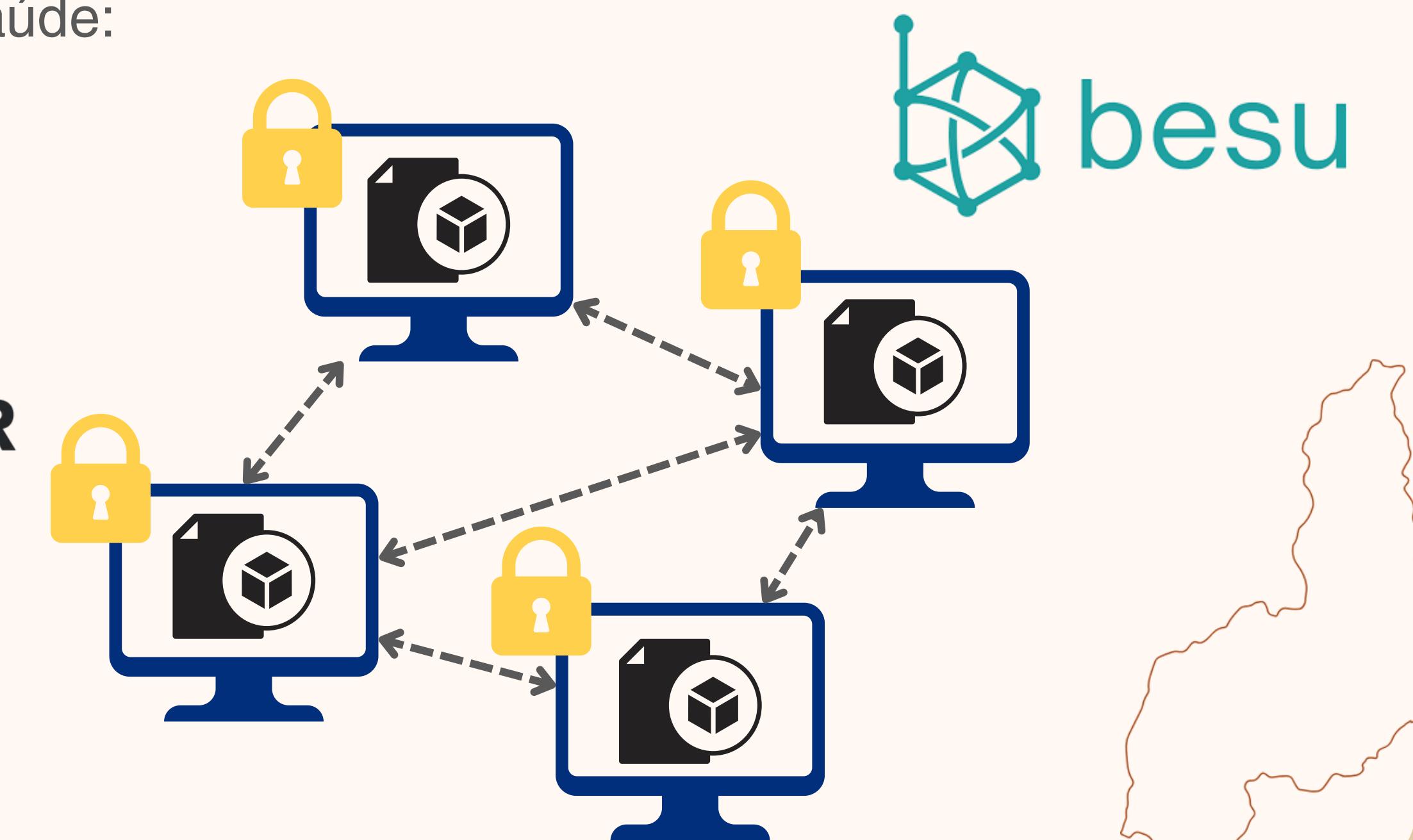
## Infraestrutura

Redes privadas *não são o foco deste tutorial*, mas saiba que existem algumas tecnologias blockchain focadas no desenvolvimento de aplicações para negócios diversos, desde finanças até saúde:

**corda**



**HYPERLEDGER FABRIC**



## Tutorial: Etherscan

**Objetivo:** Acessar o Etherscan e explorar um ledger público;

1. Acessar <https://etherscan.io>
2. No canto superior direito, mude a rede para Sepolia Testnet;
6. Identifique o número do último bloco minerado;
7. Copiar sua **chave pública** de sua carteira MetaMask e busque-a no etherscan;

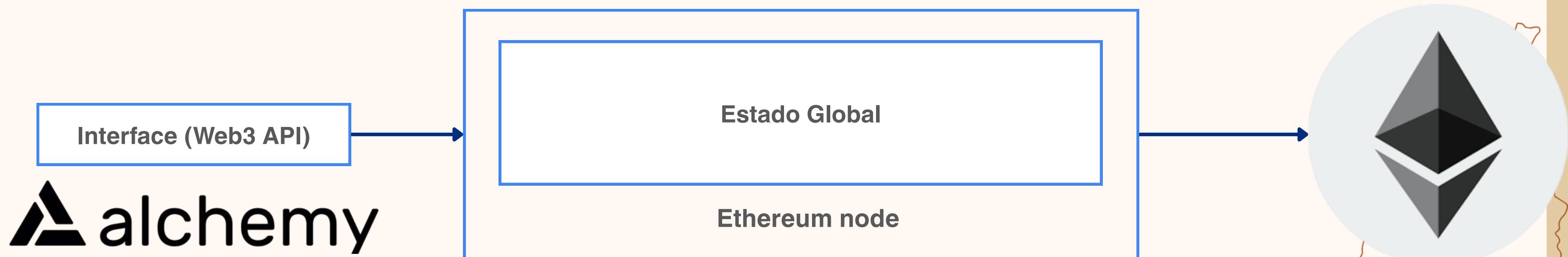
Na prática

# COMO INTERAGIR COM A BLOCKCHAIN?



**Passo 0:** Antes de tudo, é preciso escolher a infraestrutura.

**Passo 1:** Agora, precisaremos de um provedor que nos conecte à rede pública;



## Tutorial: Alchemy

**Objetivo:** Criar conta No Alchemy, obter uma chave de API para interagir com a blockchain;

1. Acessar: <https://www.alchemy.com/>
2. Criar conta;
3. Selecionar a rede Ethereum;
4. Ir em 'My Apps'
5. Encontrar sua chave de API para a rede Sepolia;



# Programação, finalmente



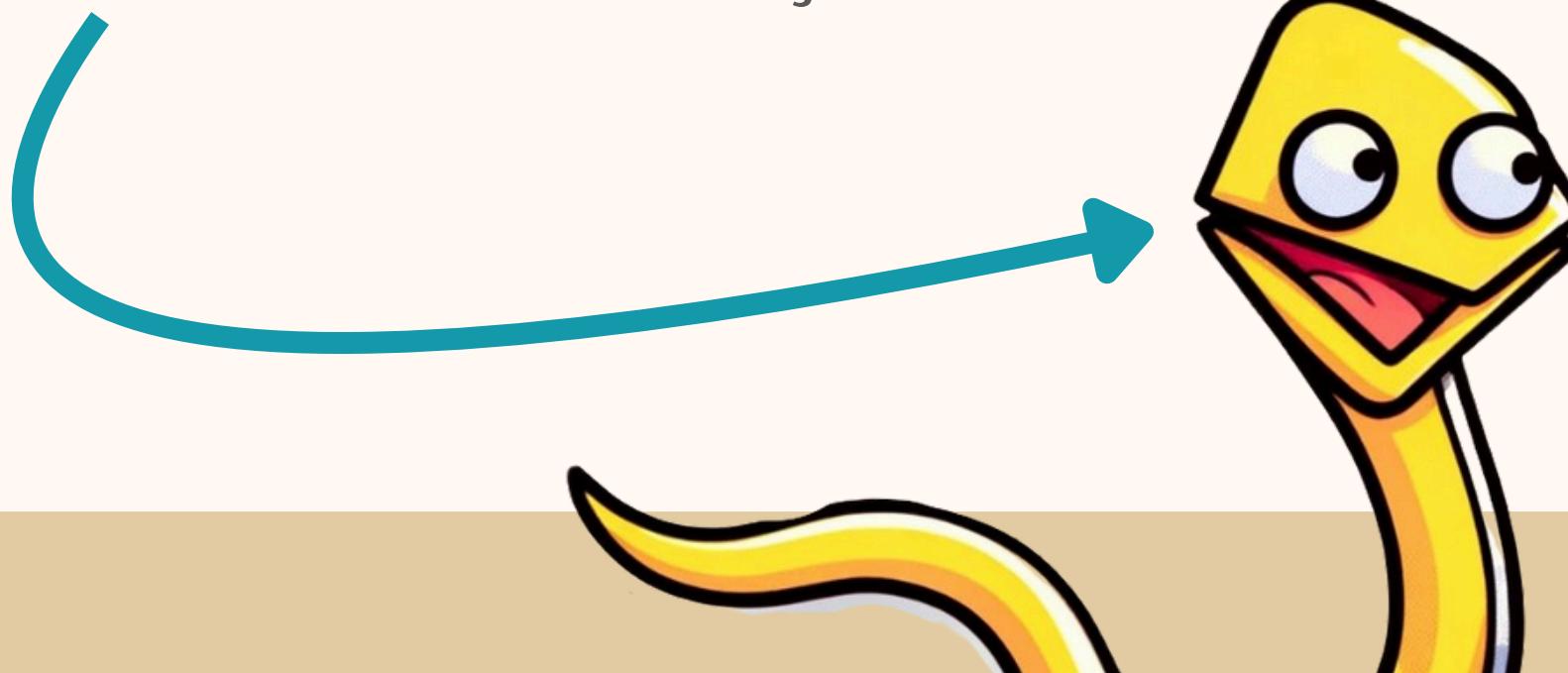
# COMO INTERAGIR COM A BLOCKCHAIN?



## **Passo 0:** Antes de tudo, é preciso escolher a infraestrutura.

**Passo 1:** Agora, precisaremos de um provedor que nos conecte à rede pública;

**Passo 3:** Em seguida, precisaremos interagir de fato com a blockchain, criando e submetendo transações;



# É AQUI QUE ENTRA O PYTHON!

# Interação via Web3.py

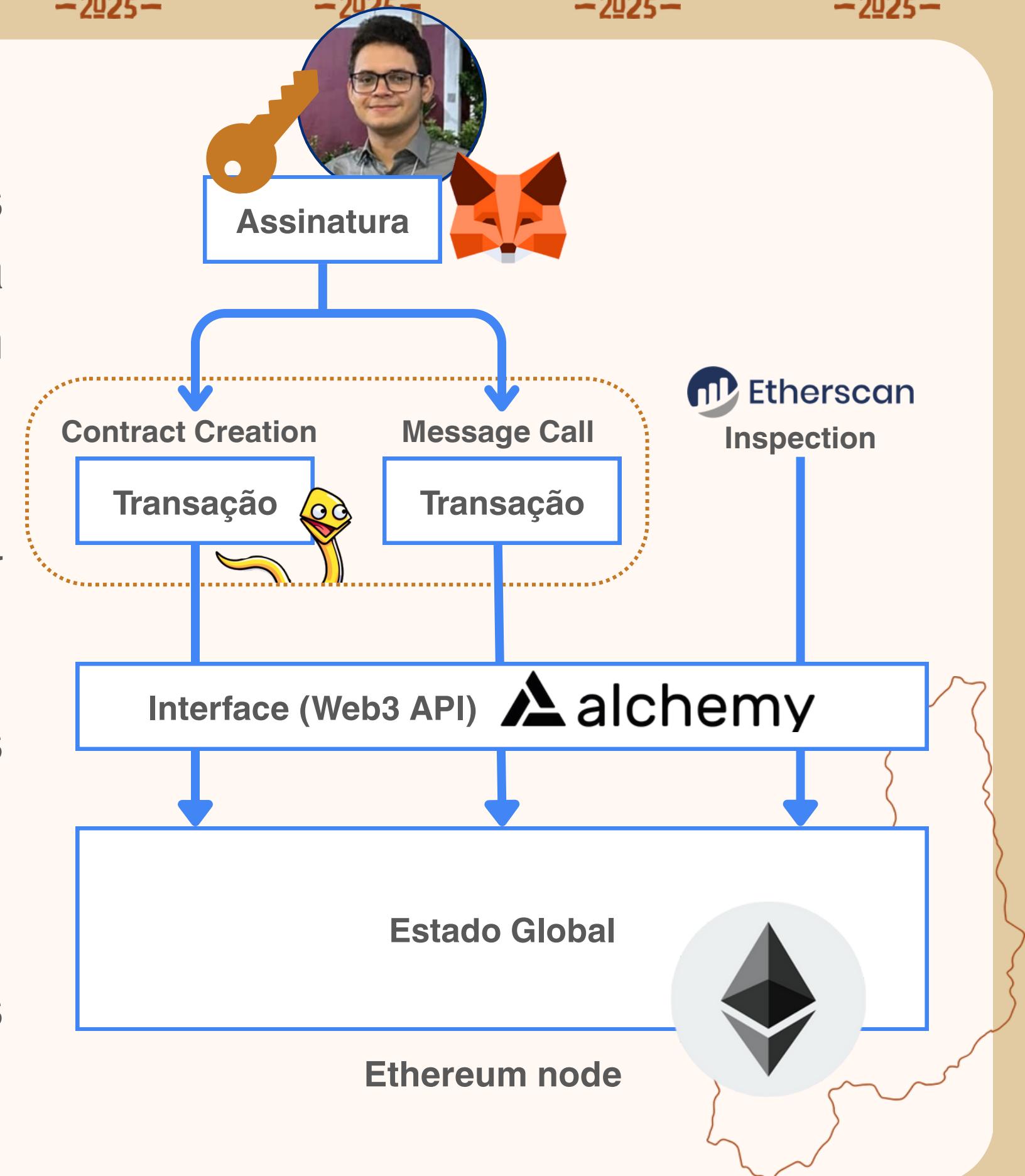
No caso da rede Ethereum (ou outras redes compatíveis com EVM), para **interagir com a blockchain** será necessário ter **acesso a um nó ethereum como provedor**.

As interações acontecem por meio de alguma interface **Web3**.

Linguagens de programação modernas oferecem interfaces de fácil utilização:

# Web3.py | web3.js

Mas antes de colocar a mão no código, alguns ‘últimos’ conceitos importantes...



# Desenvolvimento e Compilação de contratos

Uma vez definida a infraestrutura, contratos podem ser desenvolvidos utilizando a linguagem de programação **Solidity**.

Todo contrato que desejamos adicionar à blockchain precisa ser compilado pelo **compilador Solc**.

Após a compilação do contrato, o compilador retorna uma **interface ABI**, que será utilizada para interagir corretamente com os **métodos do contrato**, e um código **Bytecode** que será executado pela EVM.

Esses dois elementos serão fundamentais para a próxima etapa: o **Deploy**.



## Nosso contrato

```
pragma solidity 0.8.7;

contract VendingMachine {

    mapping (address => uint) public cupcakeBalances;

    constructor() public {
        owner = msg.sender;
        cupcakeBalances[address(this)] = 100;
    }

    function refill(uint amount) public {
        require(msg.sender == owner, "Only the owner can refill.");
        cupcakeBalances[address(this)] += amount;
    }

    function purchase(uint amount) public payable {
        require(msg.value >= amount * 1 ether, "You must pay at least 1 ETH per cupcake");
        require(cupcakeBalances[address(this)] >= amount, "Not enough cupcakes in stock to complete this purchase");
        cupcakeBalances[address(this)] -= amount;
        cupcakeBalances[msg.sender] += amount;
    }
}
```



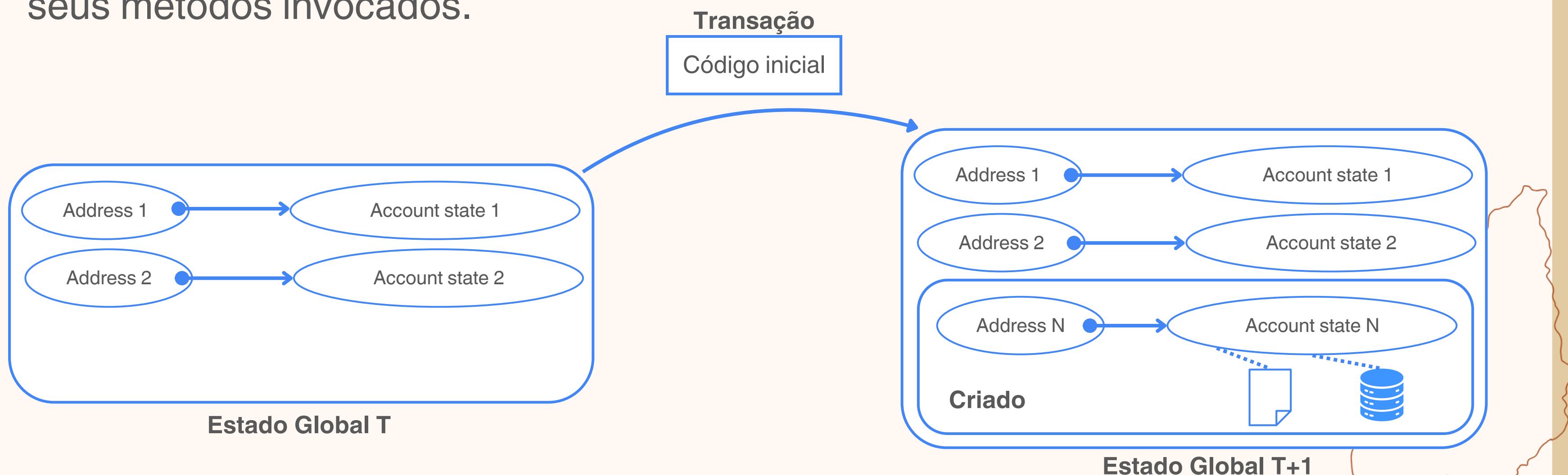
### Introdução aos contratos inteligentes

Uma visão geral dos contratos inteligentes, centrada em suas características e limitações únicas.

 ethereum.org

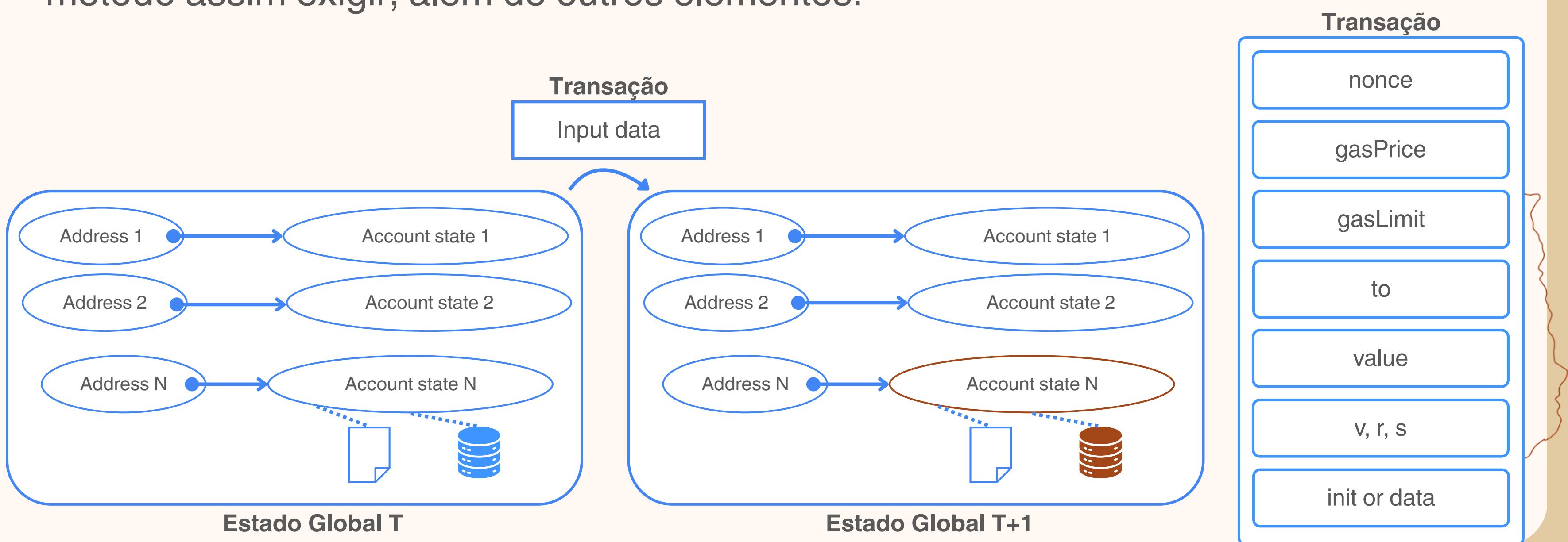
# Deploy

No Deploy, uma **transação** é submetida para **criar o mapeamento do endereço do contrato e seu estado** (código e armazenamento interno). Isso significa que se a transação for bem sucedida o contrato está armazenado na blockchain e já pode ter seus métodos invocados.



# Transações

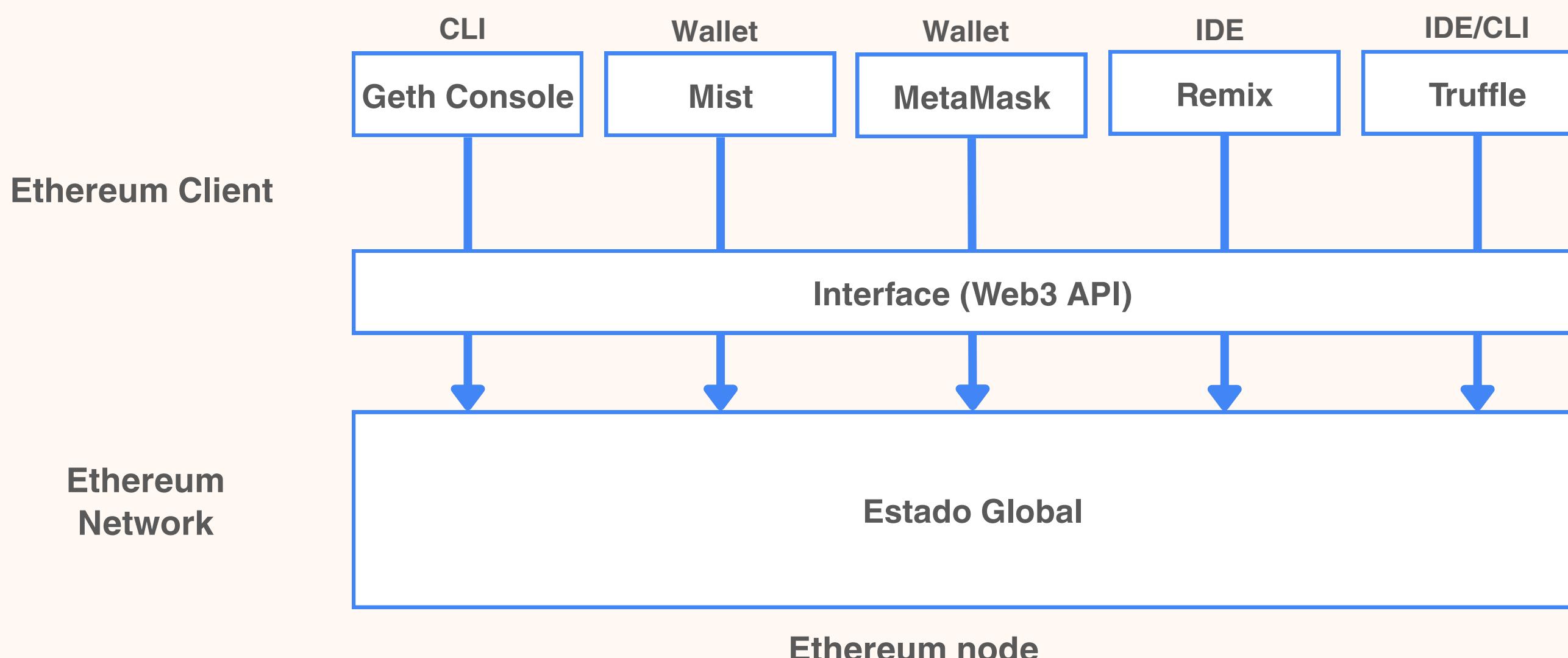
A interação com o contrato também é feita **através da interface**, que deve ser utilizada para **montar transações para o endereço do contrato**. Essas transações são diferentes das anteriores e necessitam fornecer inputs apropriados caso o método assim exigir, além de outros elementos:





## Ferramentas importantes

Resumo de ferramentas importantes para interação com contratos inteligentes em redes Ethereum:





Repositório com projeto completo

[maria.ferreira@ufpi.edu.br](mailto:maria.ferreira@ufpi.edu.br)  
[pedroffda@ufpi.edu.br](mailto:pedroffda@ufpi.edu.br)