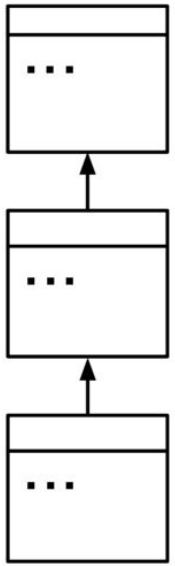
IMD0913
ETHEREUM E SMART CONTRACTS

... primeiro, uma pergunta:

O que torna o Bitcoin tão especial?

Backbone do Bitcoin

identidades criptográficas



blockchain

consenso *trustless*

O que podemos reaproveitar do Bitcoin?

Pseudônimos, identidades criptográficas que permitem prestação de contas

Decisões democráticas através do algoritmo de consenso, que não necessita de confiança

Imutabilidade da estrutura de dados

Não censurável, não pode ser controlado por nenhuma entidade

Distribuído, sem ponto central de falha

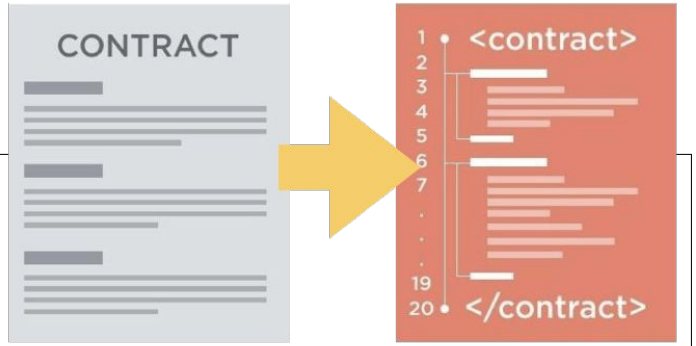
Smart Contracts

con·tract

(noun) /'käntrakt/

1. a written or spoken agreement … that is intended to be enforceable by law.

Smart Contracts



smart con·tract

(noun) /smärt ‘käntrakt/

1. code that **facilitates**, **verifies**, or **enforces** the negotiation or execution of a digital contract.
   a. **Trusted entity** must run this code

Ethereum Beginners Use Learn Developers Languages

Ethereum

Ethereum is a global, open-source platform for decentralized applications.

On Ethereum, you can write code that controls digital value, runs exactly as programmed, and is accessible anywhere in the world.

→ Beginners

- Completely new to Ethereum?

→ Use

- What can I do with Ethereum today?

Ethereum 101

Ethereum is a technology that lets you send cryptocurrency to anyone for a small fee. It also powers applications that everyone can use and no one can take down.

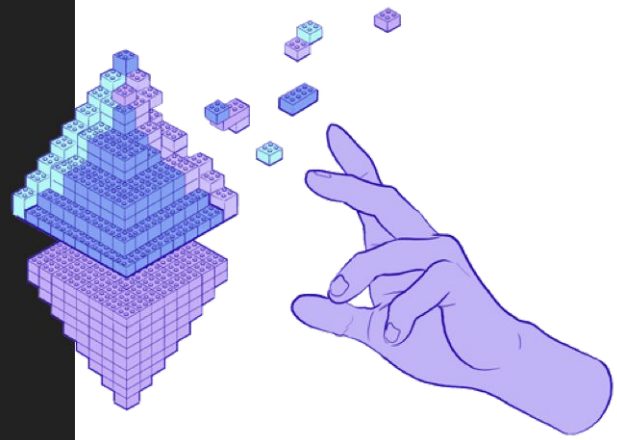
It's *the world's programmable blockchain*.

Ethereum builds on Bitcoin's innovation, with some big differences.

Both let you use digital money without payment providers or banks. But Ethereum is programmable, so you can also use it for lots of different digital assets – even Bitcoin!

This also means Ethereum is for more than payments. It's a marketplace of financial services, games and apps that can't steal your data or censor you.

So step into the bazaar and give it a try...



O que é Ethereum?

Ethereum é uma plataforma descentralizada projetada para executar contratos inteligentes (*smart contracts*)

Como um computador distribuído para executar código

Máquina de estados distribuída - transações mudam o estado global

transações == função de transição de estados

Ethereum tem um "ativo nativo" chamado **ether** (ETH)

É uma criptomoeda!

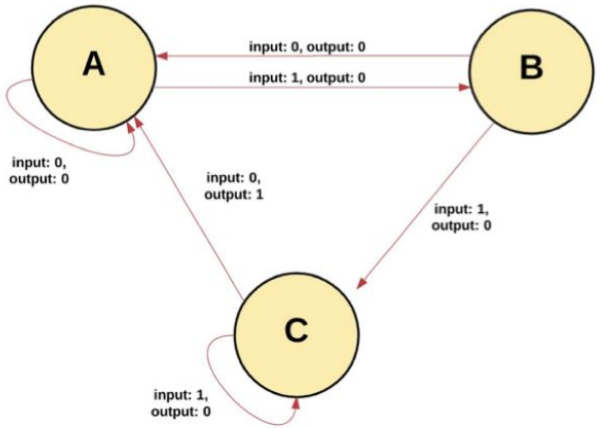
O que é Ethereum?

Para o Ethereum um *blockchain* é uma máquina de estados criptograficamente segura, singleton e compartilhada

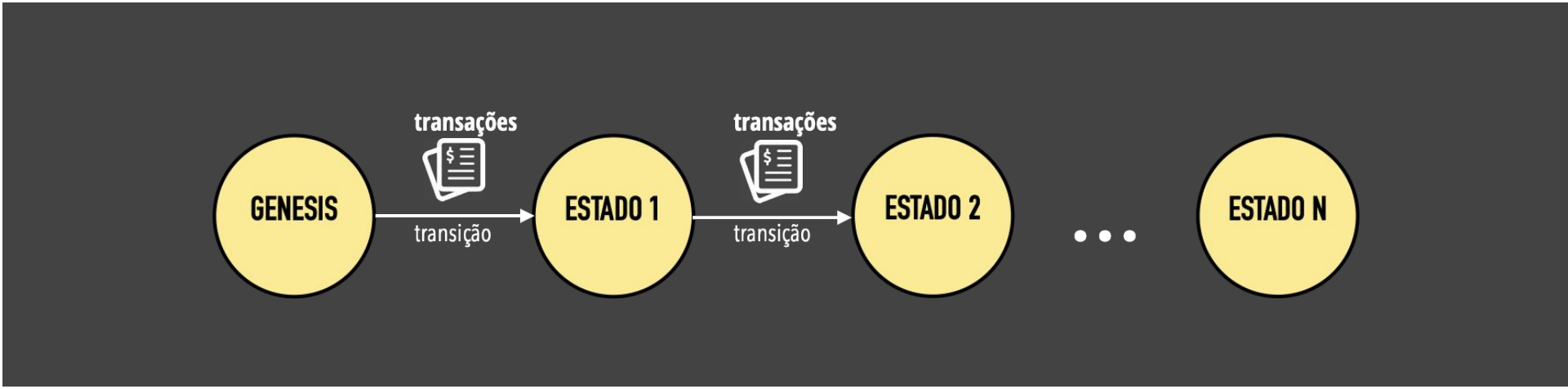
Criptograficamente seguro: Não poderia criar transações falsas

Singleton: instância única da máquina de estados para todas as transações criadas no sistema (“verdade global”)

Compartilhada: Estado armazenado nesta máquina é compartilhado e aberto para todos



O que é Ethereum?



Comparação

BITCOIN

O **padrão de ouro** do *blockchain*

Ativo: bitcoin

- Propósito primário do *blockchain* do Bitcoin

Simples e robusto

Baseado em pilha, linguagem de *scripting* simples, não é Turing-completa

Baseado em **UTXO**

Provavelmente irá manter o PoW

ETHEREUM

Plataforma de *blockchain* para **Smart Contracts**

Ativo: ether

- Financiar a computação
- Alinhar incentivos

Complexo e rico em funcionalidades

Linguagem de *scripting* Turing-completa

Baseado em **contas**

PoW → PoS

O que é Ethereum?

Ether é subdivido em unidades menores, e a menor se chama **wei**

O valor internamente no Ethereum é representando em wei

Valor (em *wei*)	Grandeza	Nome comum	Nome SI
1	1	wei	Wei
1,000	10^3	Babbage	Kilowei ou femtoether
1,000,000	10^6	Lovelace	Megawei ou picoether
1,000,000,000	10^9	Shannon	Gigawei ou nanoether
1,000,000,000,000	10^{12}	Szabo	Microether ou micro
1,000,000,000,000,000	10^{15}	Finney	Milliether ou milli
1,000,000,000,000,000,000	**10^{18}**	**Ether**	**Ether**
1,000,000,000,000,000,000,000	10^{21}	Grand	Kiloether
1,000,000,000,000,000,000,000,000	10^{24}		Megaether

Vamos começar a interagir com a rede Ethereum?

Começando pela carteira **MetaMask**

https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn

www.metamask.io

MetaMask

Instale o MetaMask no Chrome/Brave

Crie uma conta (defina a senha)

Anote as 12 palavras (*mnemonica*) para restaurar carteira

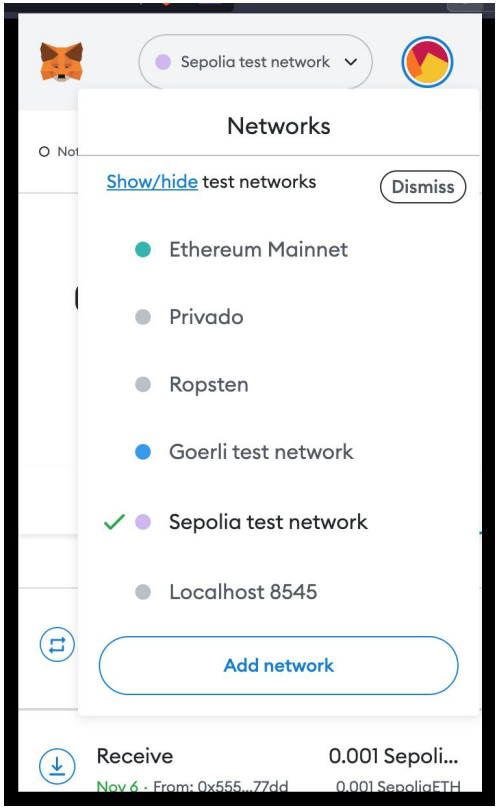
Mude a rede da principal para rede de teste **Sepolia**

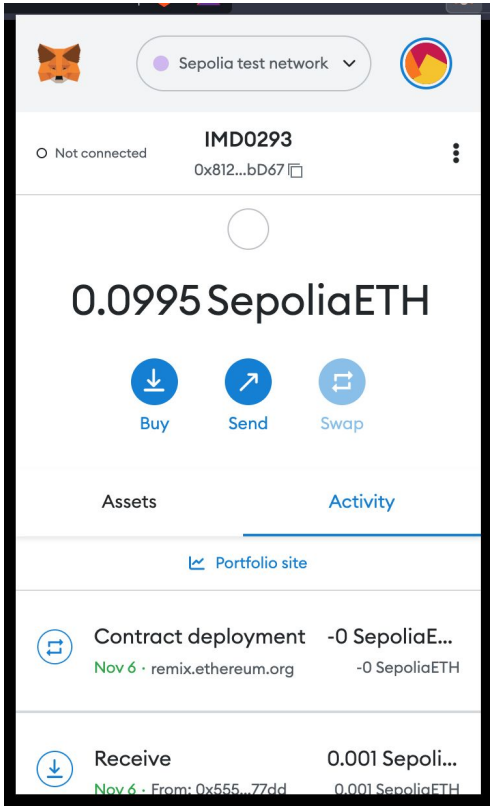
Adquira 0.05 ether através de um faucet (https://faucet.sepolia.dev/)

Pesquise também por outros Faucets para a rede Sepolia

Verifique essa transação do Etherscan (block explorer)

MetaMask


Sepolia test network
Networks
Show/hide test networks
Dismiss
Ethereum Mainnet
Privado
Ropsten
Goerli test network
Sepolia test network
Localhost 8545
Add network
Receive
0.001 Sepoli...


Sepolia test network
Not connected
IMD0293
0x812...bD67
0.0995 SepoliaETH
Buy
Send
Swap
Assets
Activity
Portfolio site
Contract deployment
-0 SepoliaE...
Nov 6 · remix.ethereum.org
-0 SepoliaETH
Receive
0.001 Sepoli...

Contas Ethereum

O estado global do Ethereum compreende vários pequenos objetos (**contas**) que podem interagir uns com os outros através de uma estrutura de passagem de mensagens

Uma conta tem um estado e um identificador de 20 bytes (160 *bits*):

`0x91fff4cbd6159a527ca4dcce2e3937431086c662`

Dois tipos de contas:

- Propriedade externa
- Contrato

Contas Ethereum

Contas de propriedade externa
Externally Owned Accounts (EOAs)

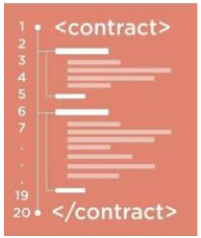
Propriedade de uma entidade externa (pessoa, organização, etc.)

Controlado por chave privada

Pode enviar transações para transferir ether ou desencadear código de contrato

Contém:

- Endereço
- Saldo de *ether*



Contas de contrato
Contract Accounts

"Propriedade" de um contrato

Controlado pelo código

Execução do código através de transações ou através de chamadas de funções (*msg*)

Contém:

- Endereço
- Código associado ao contrato
- Armazenamento persistente

Componentes de uma conta Ethereum

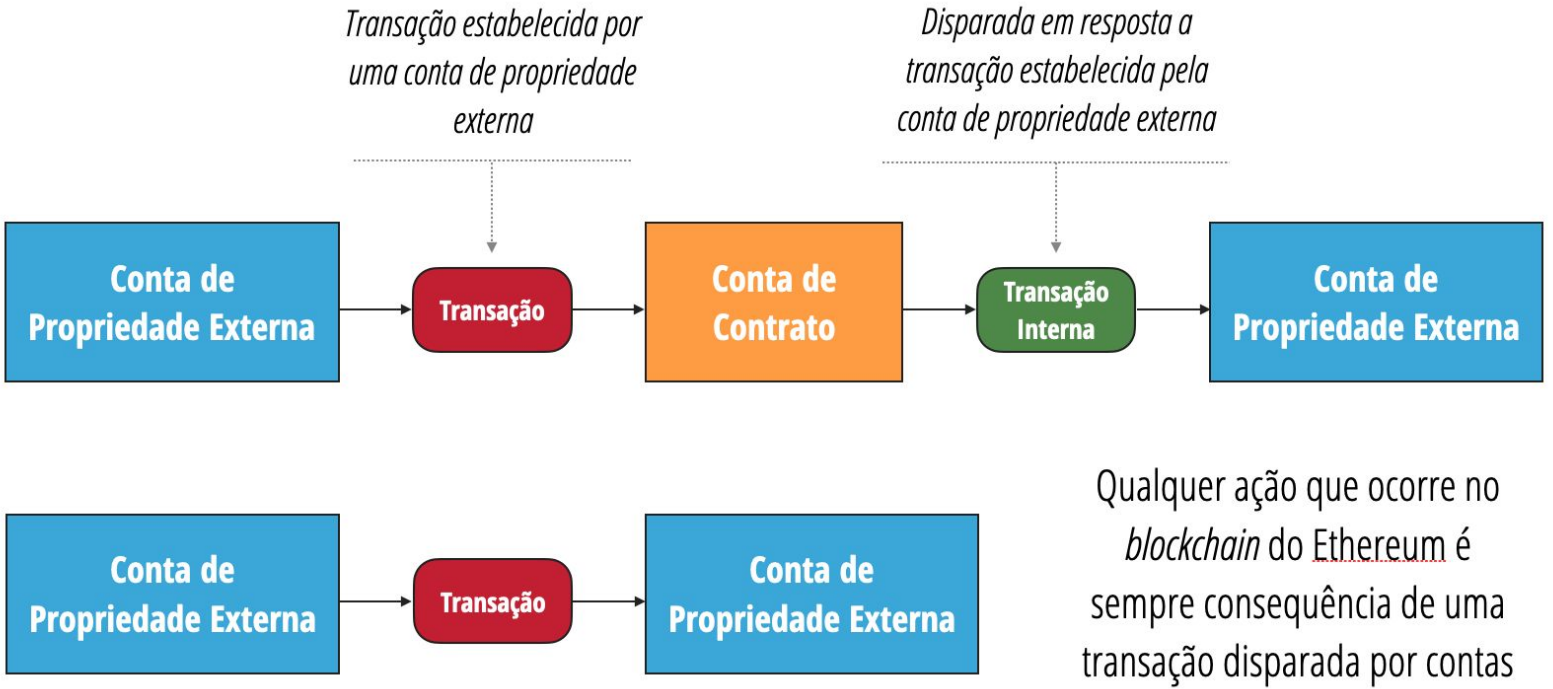
nonce: número de transações enviadas (se de *propriedade externa*) ou número de contratos criados (se de *contrato*)

balance: Saldo em Wei desse endereço (1e18 Wei = 1 ETH)

storageRoot: O *hash* do nó raiz de uma árvore Merkle-Patricia (MPT). Essa estrutura codifica o *hash* dos conteúdos de armazenamento dessa conta, e é vazio por padrão

codeHash: O *hash* do código do contrato da conta de contrato; para contas de propriedade externa, é vazio

Contas de propriedade externa ***vs*** contas de contrato



Qualquer ação que ocorre no *blockchain* do Ethereum é sempre consequência de uma transação disparada por contas de propriedade externa

Todas as contas == estado da rede

Estado de todas as contas é o estado da rede Ethereum

Toda a rede Ethereum concorda no balanço atual, estado de armazenamento, código de contrato, etc. de todas as contas

Estado da rede Ethereum é atualizado com todo bloco

Um bloco pega o estado anterior e produz um novo estado da rede

Todo nó tem que concordar com esse novo estado

Contas interagem com a rede, outras contas, outros contratos e estado dos contratos **através de transações**

Por que usar contas?

Economizar espaço

Nós só precisam atualizar o saldo de cada conta ao invés de armazenar todos os UTXOs

Mais intuitivo

Smart contracts são mais fáceis de programar quando transferimos entre saldos de conta *vs* atualização constante do conjunto de UTXOs para verificar o saldo

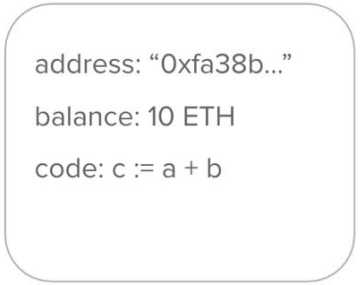
Comparação

BITCOIN

Bob é dono das chaves privadas para um conjunto de UTXOs

ETHEREUM

Alice é dona da chave privada de uma conta



Smart Contracts do Ethereum

Smart Contracts no Ethereum são como agentes autônomos que residem dentro de uma rede Ethereum

- Reagem ao mundo externo quando “cutucados” por transações (que chamam funções específicas)
- Tem controle direto sobre:
  - Saldo de *ether* interno
  - Estado interno do contrato

Smart Contracts do Ethereum

Smart Contracts do Ethereum

Contratos no Ethereum geralmente servem quatro propósitos:

Armazenar e manter dados

Dados representam algo útil para usuários ou outros contratos

Ex: um token de alguma moeda ou uma associação a alguma organização

Gerenciar contrato ou relação entre usuários sem relação de confiança

Ex: contratos financeiros, garantias, seguros, ...

Prover funções para outros contratos

Servindo como uma biblioteca de *software*

Autenticação complexa

Ex: Acesso através de multiassinatura M-de-N

Smart Contracts do Ethereum

```
pragma solidity >=0.4.16 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

Smart Contracts do Ethereum

```
contract Betting {
    address public owner;
    address public gamblerA, gamblerB, oracle;
    uint[] outcomes;
    struct Bet {
        uint outcome;
        uint amount;
        bool initialized;
    }

    mapping (address => Bet) bets;
    mapping (address => uint) winnings;
    ...
    function makeBet(uint _outcome) payable returns (bool) { ... }
    function makeDecision(uint _outcome) oracleOnly { ... }
    function withdraw(uint withdrawAmount) returns (uint remainingBal) { ... }
}
```

Smart Contracts do Ethereum

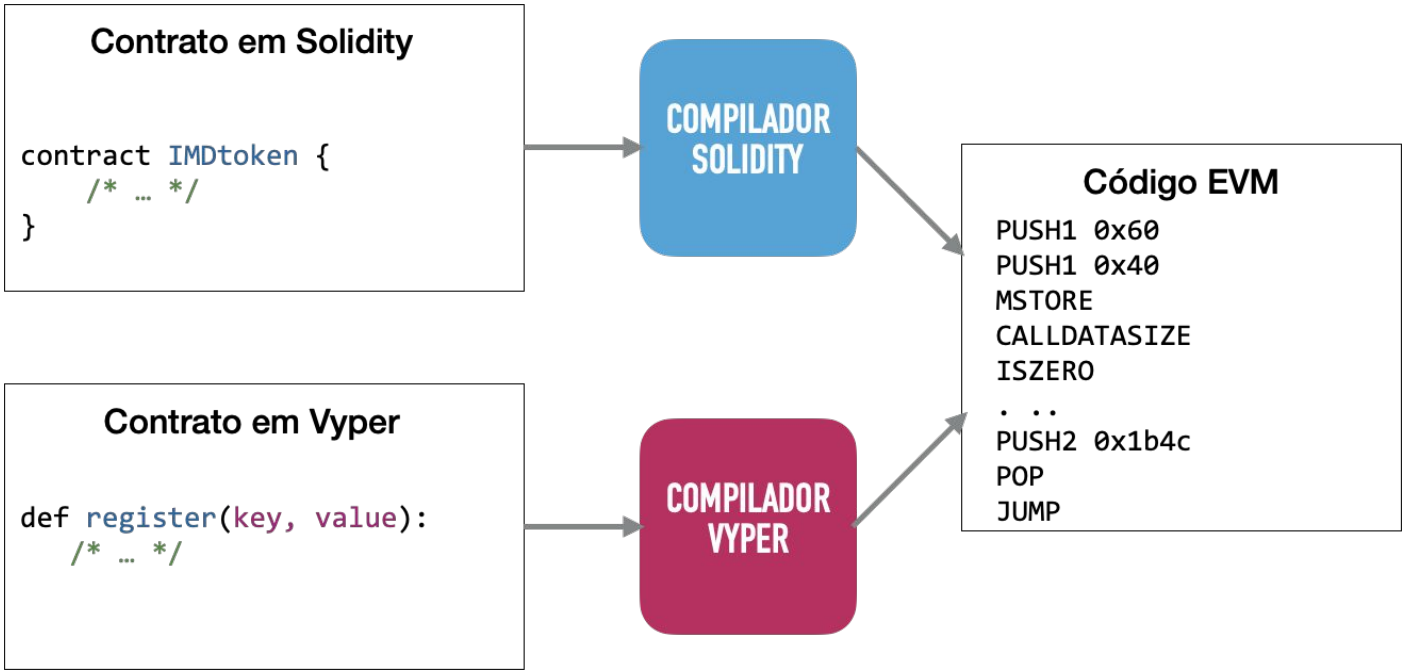
Ethereum é um "**computador distribuído**": todo nó validador processa transações, executa *smart contracts*, e então atinge um consenso sobre o novo estado da rede

Algoritmo de consenso do Ethereum é o **Proof-of-Stake**

Consenso da rede remove a necessidade do terceiro confiável

Acordos *peer-to-peer* que vivem no *blockchain* para sempre

Ethereum Virtual Machine (EVM)



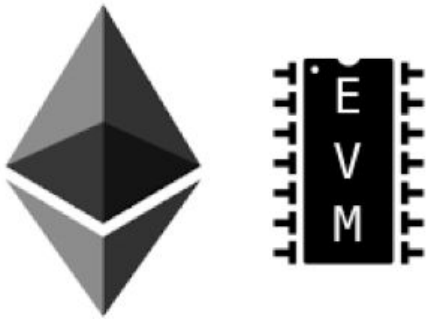
Ethereum Virtual Machine (EVM)

A EVM é um “mini computador” que executa código de contratos

Código de contrato que de fato é executado em cada nó é código EVM

> Código EVM: baixo nível, linguagem baseada em *bytecode*, como o Java e a JVM

Todo nó Ethereum roda a EVM



Ethereum Virtual Machine (EVM)

Problema imediato:

E se nosso contrato conter um *loop* infinito?

```
function foo() {
    while (true) {
        /* loop infinito! */
    }
}
```

Todo nó da rede ficaria preso executando o *loop* para sempre!

Devido ao **problema da parada**, é impossível determinar de antemão se um contrato irá terminar sua execução

Ataque de negação de serviço!

EVM: gas e taxas

Solução do Ethereum:

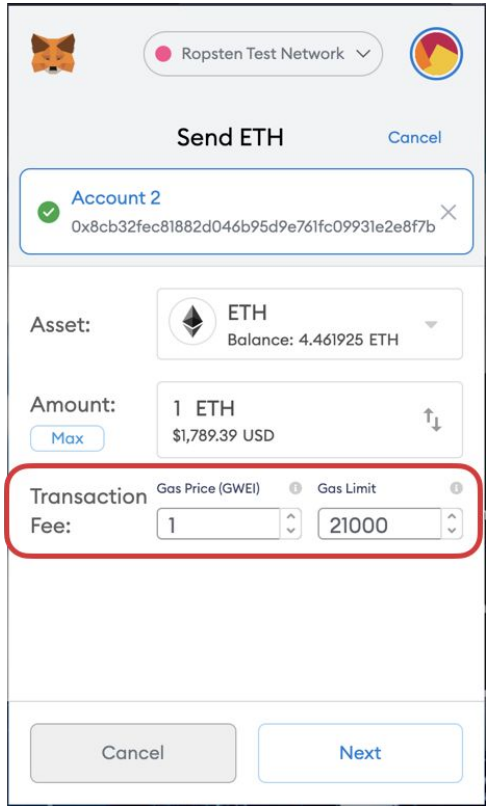
Todo contrato requer ***gas***, que é o "combustível" para executar contratos

Todo *op-code* da EVM requer *gas* para executar

Toda transação especifica:

- o `startgas`, ou a máxima quantidade (limite) de *gas* que está disposta a consumir;
- o `gasprice`, ou a taxa, em *ether*, está disposta a pagar por unidade de *gas*.

O preço da unidade de *gas* é medido em *gwei* (1e9 wei)



EVM: gas

No início da transação:

> `startgas` * `gasprice` são subtraídos da conta de quem envia (quem está "cutucando" o contrato)

Se o contrato é **executado com sucesso**...

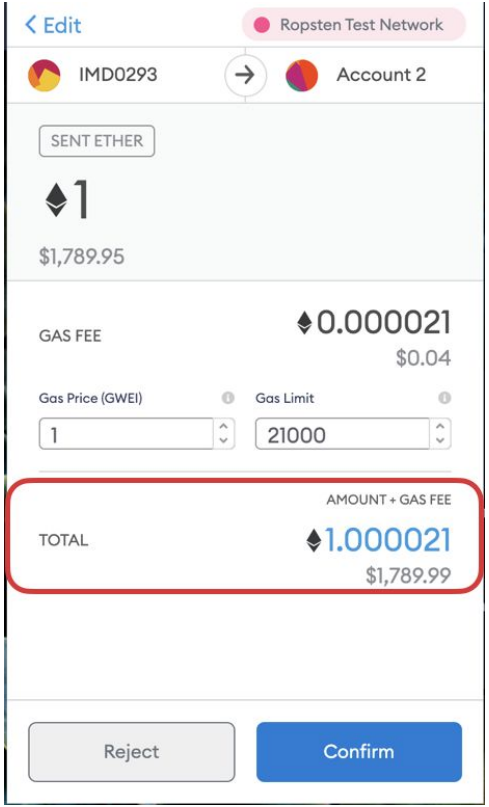
> O *gas* restante é reembolsado para quem enviou

Se a execução do contrato **ficar sem *gas* antes de terminar**...

> `startgas` * `gasprice` não é reembolsado

Comprar *gas* = comprar poder computacional distribuído e *trustless*

Um usuário mal intencionado procurando lançar um ataque DoS precisará fornecer *ether* suficiente para financiar o ataque



EVM: gas

Exemplo: startgas: 50.000, gasprice: 20 gwei

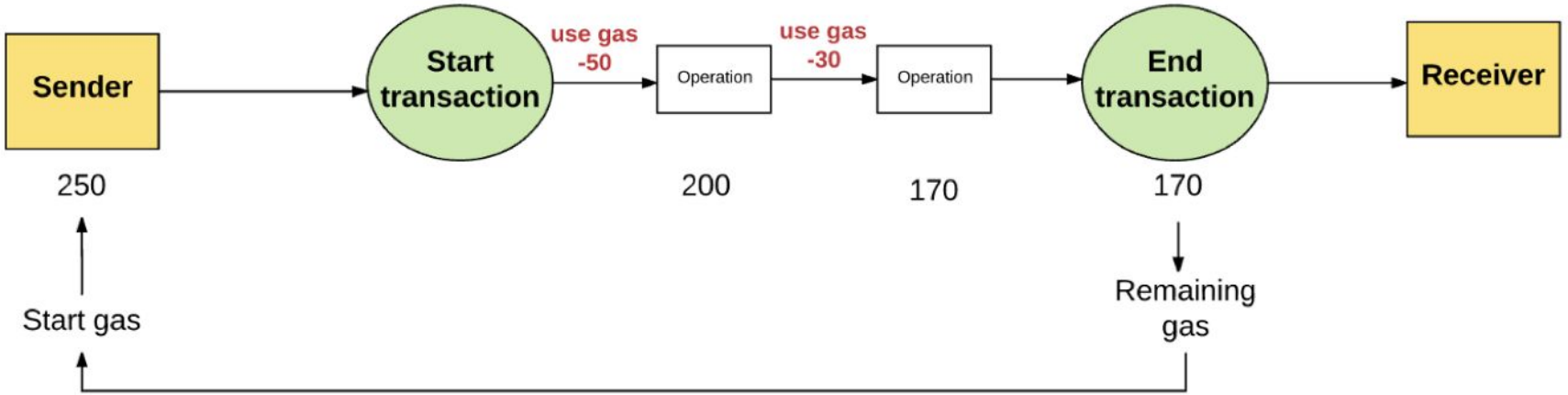
Máxima taxa (*fee*) de transação: 50.000 * 20 gwei = 1.000.000.000.000.000 = 0,001 Ether

Como o startgas é o limite máximo de uso de *gas* que quem está enviando está disposto a pagar, se ele tiver saldo suficiente em sua conta para cobrir esse máximo e se o startgas é suficiente para executar a transação, a transação será executada

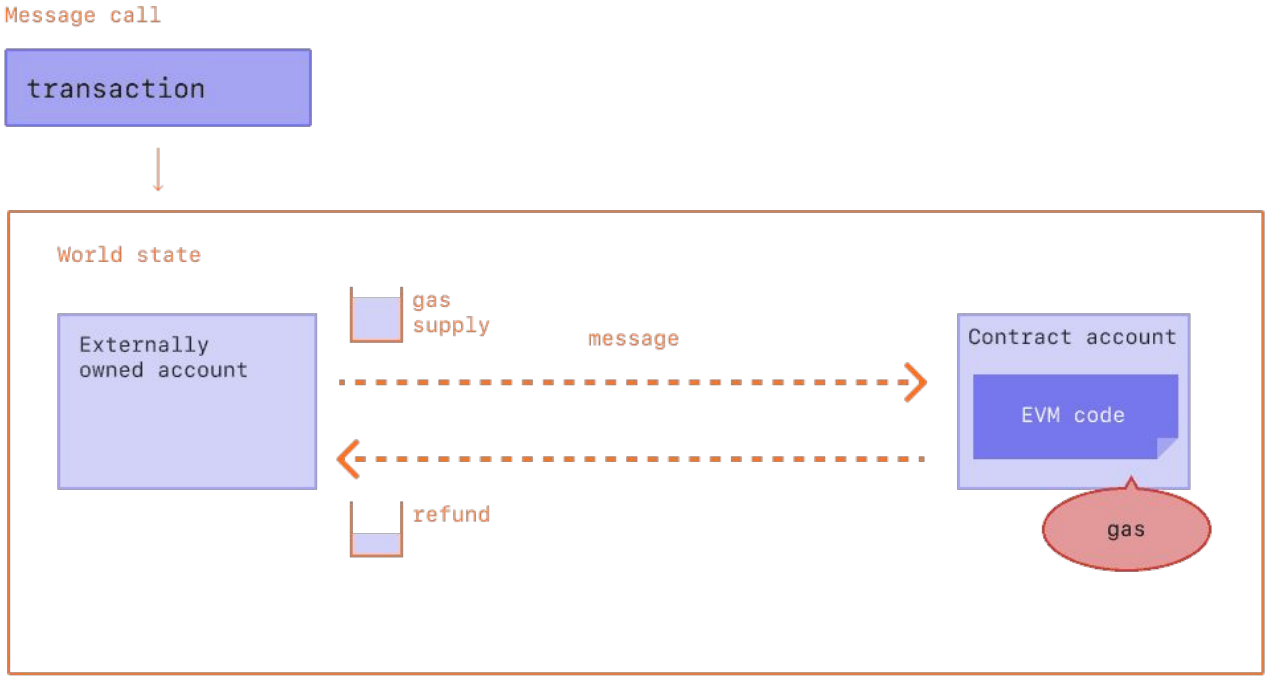
```
startGas == gasLimit == gasSend
```

***Gas* não utilizado é devolvido a quem enviou**

EVM: gas


Sender
Start transaction
use gas -50
Operation
use gas -30
Operation
End transaction
Receiver
250
200
170
170
Start gas
Remaining gas

EVM: gas


Message call
transaction
World state
Externally
owned account
gas
supply
message
Contract account
EVM code
gas
refund

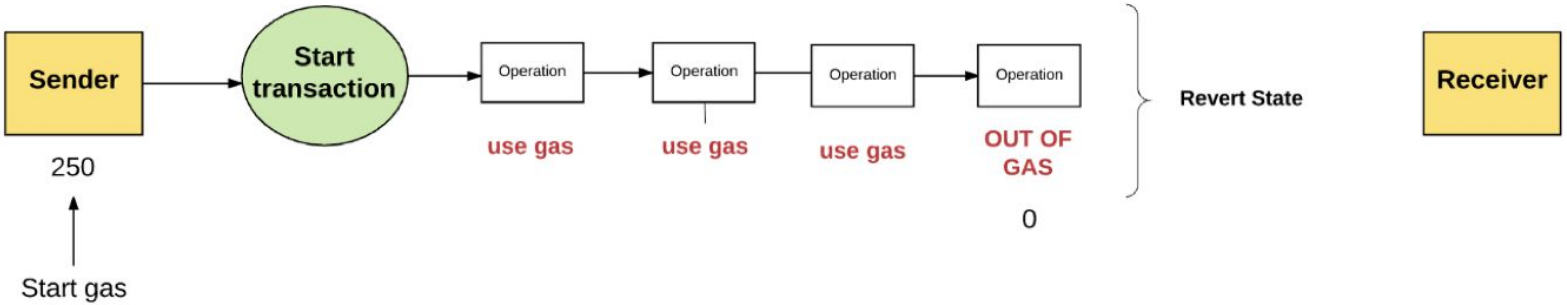
EVM: gas

Sem gas para executar a transação?

Transação fica sem *gas* e é considerada inválida

Mudanças de estado são revertidas

Como a computação pela rede já foi feita, nenhum *gas* é reembolsado

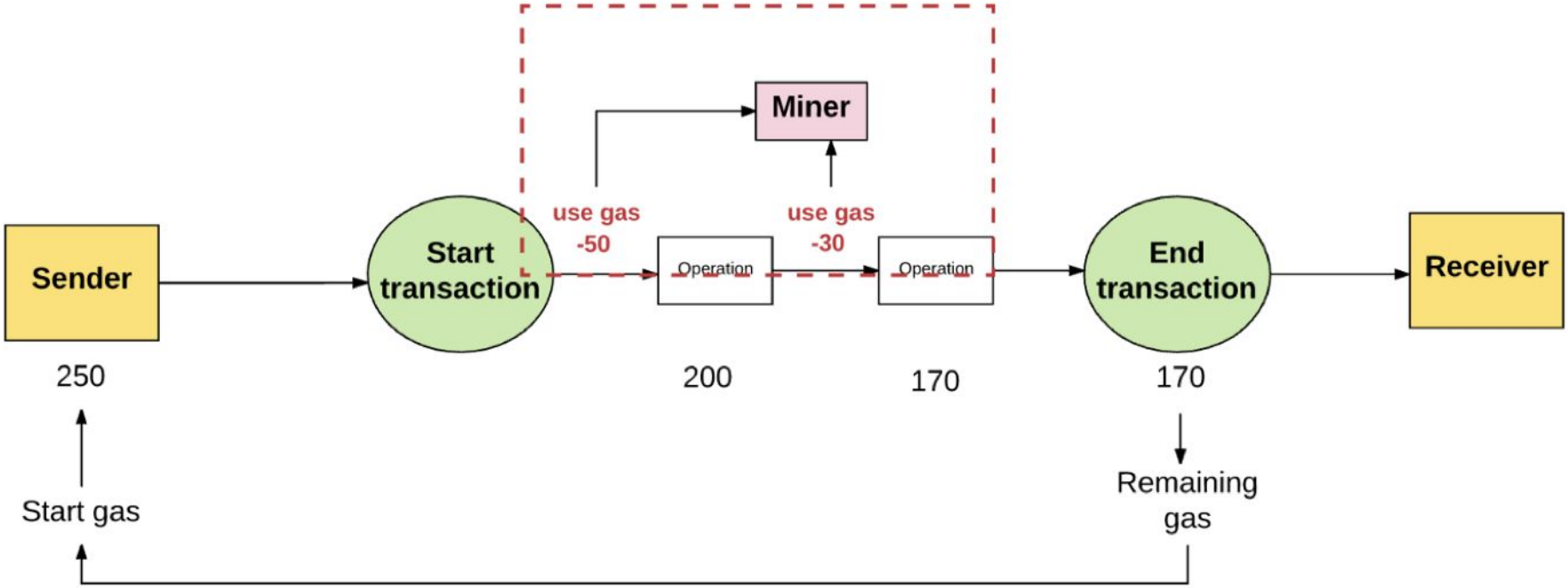


EVM: gas

Para onde a taxa (*fee*) de transação vai?

- Para o endereço do validador que propôs o bloco, em troca de seu esforço e computação realizado (é uma recompensa!)
- Quanto maior o preço do *gas* (`gasprice`) você está disposto a pagar, maior a recompensa do validador
  - E é mais provável que a transação seja incluída em um bloco, pois validadores escolhem quais transações irão validar ou ignorar

EVM: gas


Miner
use gas
-50
use gas
-30
Sender
Start transaction
Operation
Operation
End transaction
Receiver
250
200
170
170
Start gas
Remaining gas

EVM: gas

Gas também é usado para pagar por armazenamento, proporcional ao menor múltiplo de 32 *bytes* utilizado

Aumentar o armazenamento implica também no aumento do estado do Ethereum em todos os nós

Portanto existem incentivos para mante-lo pequeno

Se uma transação tem um passo que limpa uma entrada que está sendo armazenada, a taxa é reembolsada (*refund*)

Questão

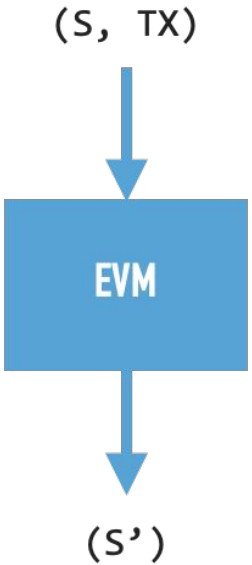
Lembra que *gas* é utilizado para pagar para que seu *smart contract* seja validado pelo computador distribuído. **Quanto *gas* você precisa fornecer?**

A) O suficiente para uma pessoa executar o código

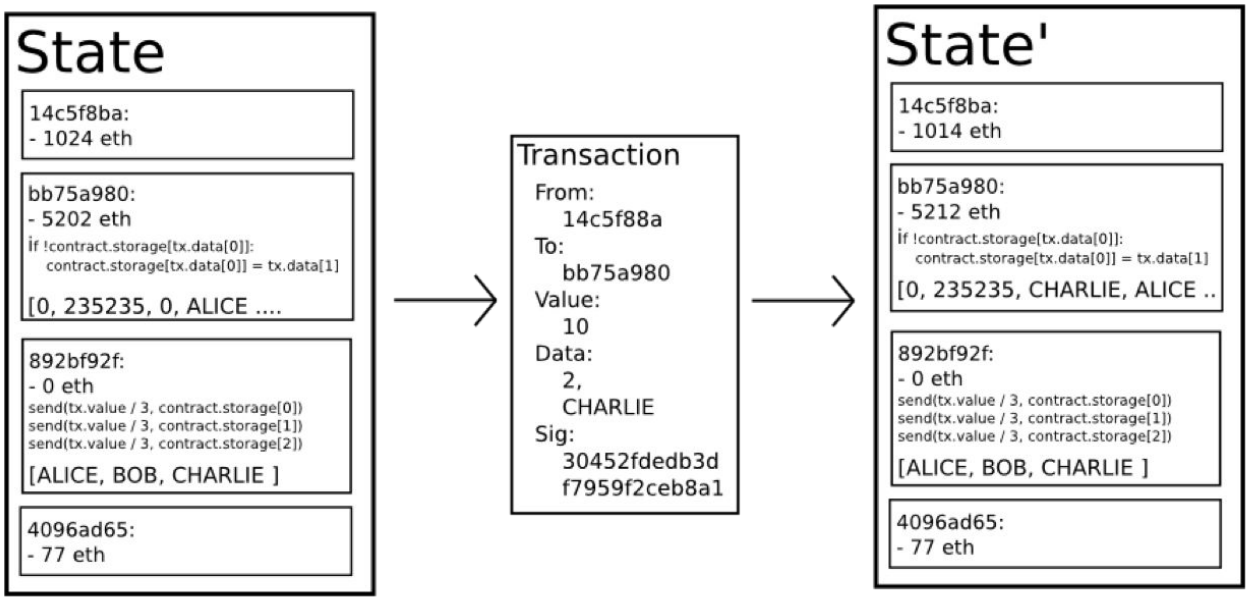
B) O suficiente para todo os nós *full* da rede executar o código

C) O suficiente para validadores delegados executarem o código

Função de transição de estados

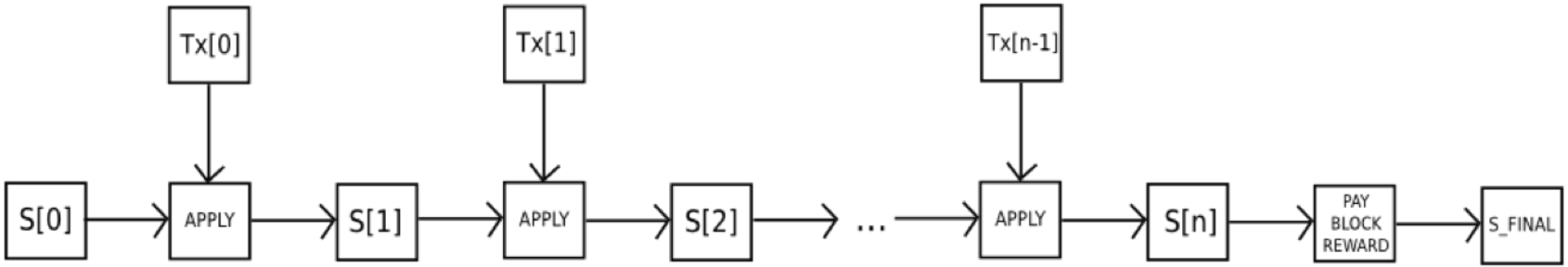


Função de transição de estados



Ethereum Whitepaper

Função de transição de estados



EVM: gas

Ethereum não é sobre otimizar a eficiência da computação

Seu processamento paralelo é **paralelismo redundante**

- **Maneira de atingir o consenso** no sistema sem precisar de terceiros confiáveis

Execução de contratos são replicados redundantemente através dos nós

- Caro, devagar e uso intensivo de memória
- De certa maneira é um **incentivo a não usar *blockchain*** para computação que pode ser feita sem *blockchain*!

Questão

Como você decidiria entre usar uma solução **centralizada** ou **descentralizada**?

Comparação

Use *blockchain* se:

houver necessidade de um banco de dados compartilhado com múltiplos usuário com permissão de escrita

as partes não podem confiar entre elas, e nenhum terceiro confiável ou autoridade está disponível

tem interesse em tolerância a falhas, imutabilidade dos dados e resistência a censura

Use solução centralizada se:

o banco de dados não precisa ser compartilhado, ou é compartilhado entre partes que se confiam

precisa ser mantido confidencial

deve manipular grandes quantidades de dados e/ou dados complexos

requer edição dos dados

tem interesse em velocidade e eficiência

Transações Ethereum

Transações mudam o estado de uma conta dentro do estado global - de um estado para outro

Três tipos: **regular**, **execução de contrato** e **criação de contrato**

Transações Ethereum

Dentre os componentes de uma transação Ethereum, podemos destacar:

nonce: número de transações enviadas pela conta

gasprice: quantidade de *wei* que quem envia está disposto a pagar por unidade de *gas* para executar a transação

startgas: quantidade máxima de *gas* quem envia está disposto a pagar para executar a transação, estabelecido antes de qualquer computação ter sido realizada

to: endereço do destinatário

value: quantidade de *wei* que será transferida de quem envia para quem é o destinatário

assinatura: assinatura que identifica quem envia a transação

data

em transações de criação de contrato: O fragmento de código EVM que é utilizado para inicializar uma nova conta de contrato

em transações de execução de contrato: os dados de entrada (por exemplo, parâmetros) da chamada de mensagem

Transações Ethereum

As transações de **execução de contrato** e também de **criação de contrato** sempre são inicializadas por contas de **propriedade externa**

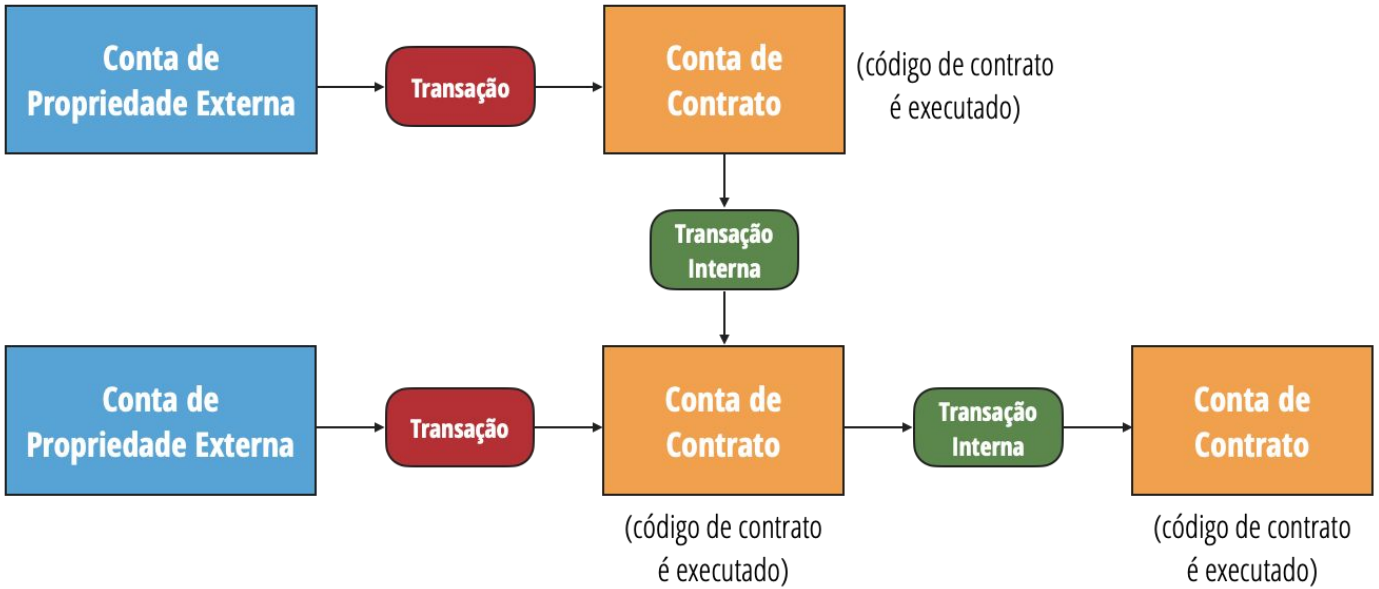
> Transações são a ponte entre o mundo externo e o estado interno do Ethereum

Contratos que existem dentro do escopo global do Ethereum podem se comunicar com outros contratos através de **mensagens** (transações internas)

Podemos imaginar mensagens como sendo similares a transações, exceto que elas não são geradas por contas de propriedade externa, somente por contratos

Transações Ethereum

Quando um contrato envia uma transação interna para outro contrato, o código associado que existe do contrato alvo é executado



Transações Ethereum

Mensagens (transações internas) não contém `startgas`

`startgas` determinado pelo criador (externo) da transação original

> Portanto o `startgas` que a conta de propriedade externa definiu deve ser suficiente para alimentar a transação e qualquer outra sub-execução que ocorra como resultado dessa transação