



## INTERNET DAS COISAS, BLOCKCHAIN E CONTRATOS INTELIGENTES APLICADOS À SAÚDE

Jauberth Abijaude  
jauberth@uesc.br

Henrique Serra  
haserra.cic@uesc.br

Rita Barreto  
rita.barreto@ufba.br

Aprígio Bezerra  
aalbezerra@uesc.br

Péricles Sobreira  
pericles.delimasobreira@uqo.ca

Fabíola Greve  
fabiola@ufba.br

1

## AGENDA

Apresentação dos autores

Introdução

Plataforma Ethereum e Contratos Inteligentes

Desafios de Blockchain e IoT na área de Saúde

Pesquisas e Aplicações Recentes

Integração de IoT, Blockchain e Sistemas Web

Como montar um curso de IoT, Blockchain e Sistemas Web

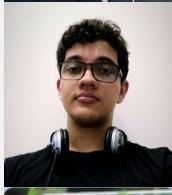
Desafio de Pesquisa e Conclusões

2

## AUTORES



Jauberth Weyll Abijaude  
Doutorando em Ciência da Computação – UFBA  
Professor UESC



Henrique Serra  
Graduando em Ciência da Computação – UESC  
Bolsista de IC



Rita Barreto  
Doutoranda em Ciência da Computação – UFBA



Aprígio Bezerra  
Universidade Autônoma de Barcelona – Espanha  
Professor UESC



Péricles Sobreira  
Doutor em Ciência da Computação pela Université Grenoble Alpes, França  
Professor University of Quebec at Outaouais e Granby College



Fabíola Greve  
Doutora em Informática pela Université Rennes I e Laboratórios IRISA-INRIA, França  
Professora UFBA

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

3

## INTRODUÇÃO

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

4

## INTRODUÇÃO

- Nosso objetivo é auxiliar no desenvolvimento das competências para criar DApps com foco na plataforma Ethereum;
- Vamos apresentar neste seção:
  - Blockchain;
  - IoT;
  - Contratos Inteligentes;
  - Blockchain e IoT;
  - DApps

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

5

5

## HISTÓRICO BLOCKCHAIN



### 1997 – Nick Szabo – Contratos Inteligentes

CIs são uma combinação de protocolos com interfaces de usuário para formalizar e proteger relacionamentos em redes de computadores



### 2008 – Bitcoin – Satoshi Nakamoto

Habilidosa combinação de técnicas para dar suporte à criptomoeda bitcoin



### 2013 – Ethereum – Vitalik Buterin

Transaciona uma criptomoeda e códigos de programação mais avançados



### Novas aplicações e plataformas baseadas em Blockchain

Diversas criptomoedas, protocolos de consenso e aplicações

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

6

6

3

## COMPONENTES DE UMA BLOCKCHAIN

Rede P2P

Mensagens

Protocolos de Consenso

Máquina de Estados

Cadeia de blocos



Blockchain Pública

Mecanismo de incentivo

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

7

## PROPRIEDADES DA BLOCKCHAIN



Descentralização;



Disponibilidade e integridade;



Transparência e Auditabilidade;



Imutabilidade e Irrefutabilidade;



Privacidade e Anonimidade;



Desintermediação;



Cooperação e Incentivos;

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

8

8

## TIPOS DE BLOCKCHAIN

### PÚBLICA

- Não permissionada ou de acesso aberto;
- Acesso anônimo;
- Sem confiança mútua.



**bitcoin**

### PRIVADA

- Permissionada ou federada;
- Acesso autenticado;
- Ambientes corporativos.

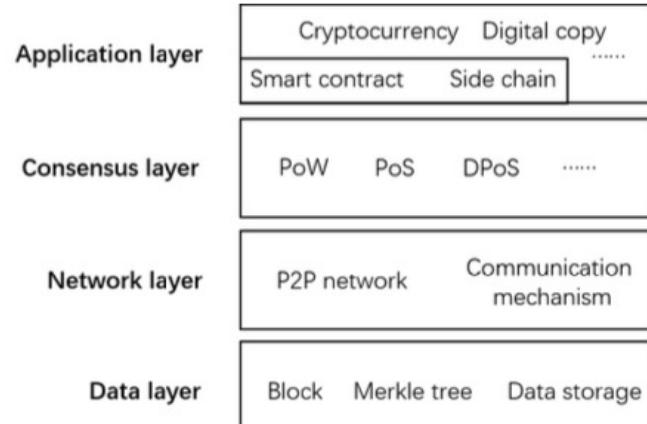


Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

9

9

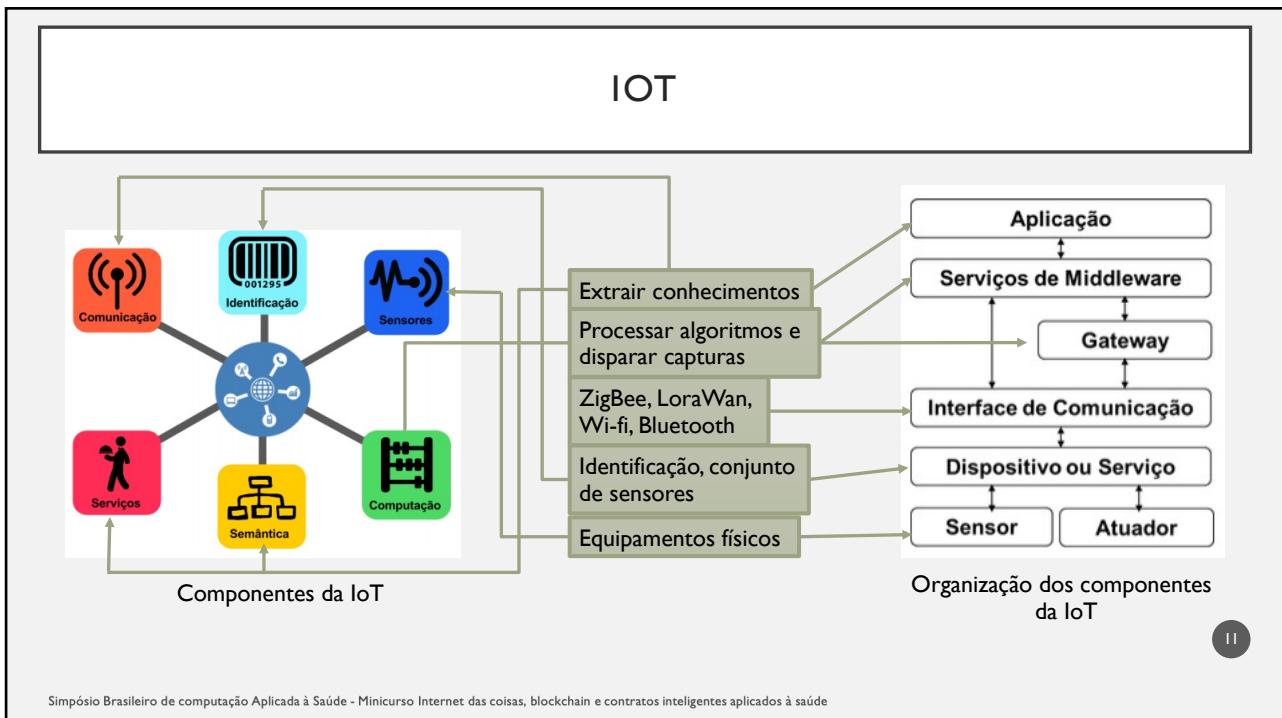
## ARQUITETURA BLOCKCHAIN



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

10

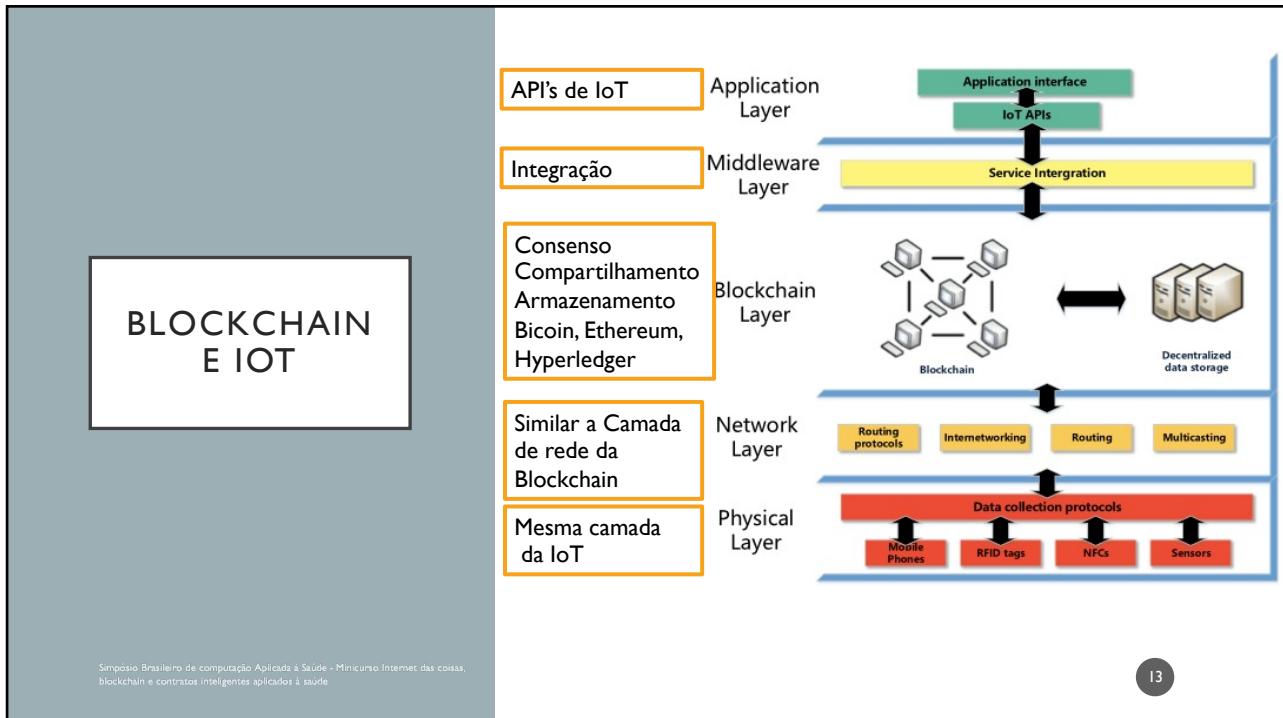
10



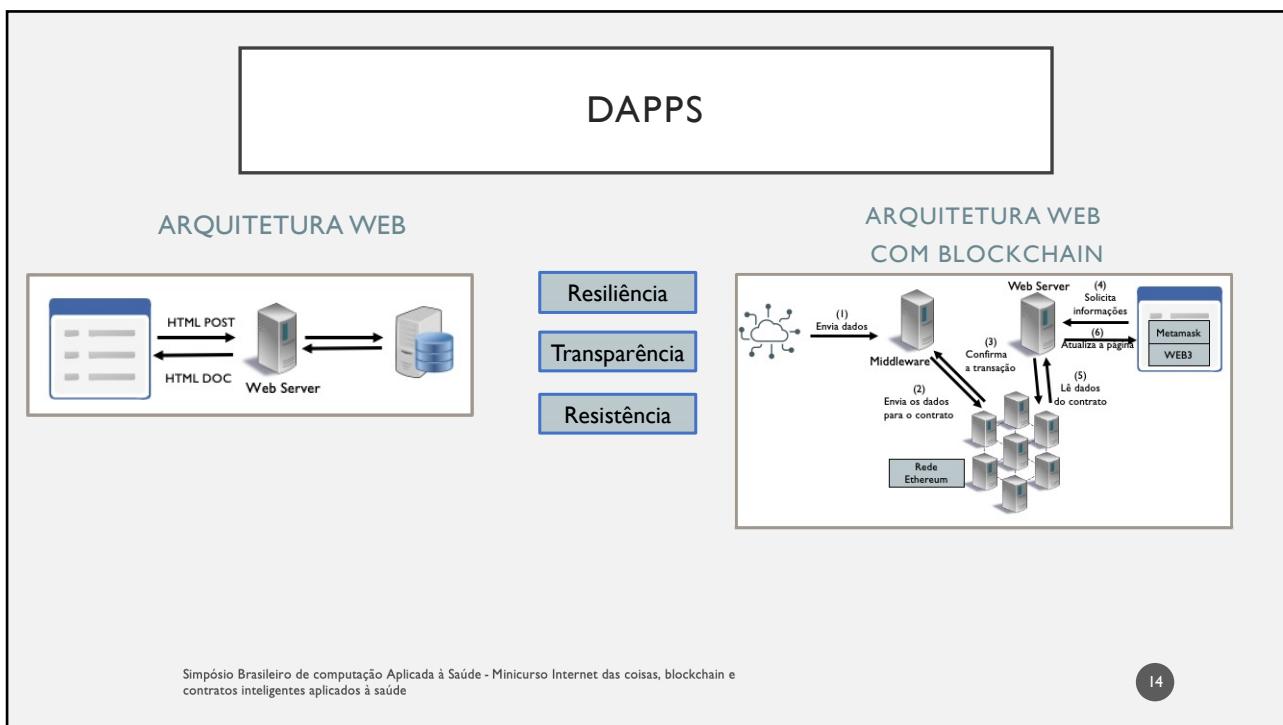
11



12



13



14

## PLATAFORMA ETHEREUM E CONTRATOS INTELIGENTES

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

15

15

## PLATAFORMA ETHEREUM E CONTRATOS INTELIGENTES

- Plataforma Ethereum
  - Redes Ethereum
  - Transações
  - Consenso
- Contratos Inteligentes

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

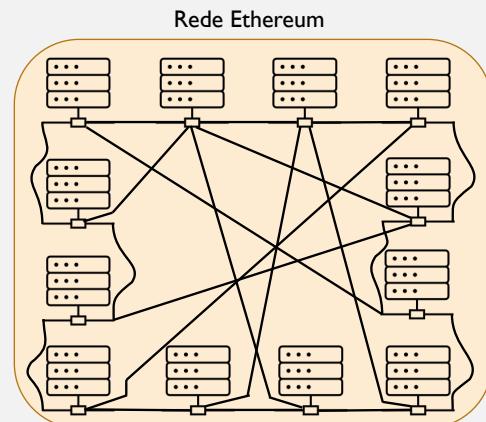
16

16

## REDE ETHEREUM

### Características

- Usada para transferência de dinheiro digital
- Existe mais de uma rede Ethereum
- Executa programas chamados de smart contracts
- Cada nó é uma máquina que possui um cliente Ethereum rodando
- Qualquer um pode rodar um nó – associação de mineradores
- Cada nó pode conter uma cópia completa da blockchain



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

17

## COMPONENTES DA REDE ETHEREUM

Rede P2P - A rede principal Ethereum roda na porta TCP 30303, e o protocolo  $\text{Ξ}Vp2p$ .

Regras de consenso – São definidas na especificação de referência - yellow paper.

Transações - As mensagens incluem: *sender, recipient, value, e data payload*.

Ethereum Virtual Machine (EVM) – Máquina de estado que executa os bytecodes gerados na compilação dos smart contracts.

Estrutura de dados – Armazenado em cada nó (Google's LevelDB). Contém transações e o estado do sistema na estrutura chamada Merkle Patricia Tree.

Algoritmo de Consenso – mesmo consenso do Bitcoin POW.

O Ethereum atualmente usa um algoritmo PoW chamado Ethash.

Clientes Ethereum – Várias implementações interoperáveis. As mais conhecidas são Go-Ethereum (Geth) e Parity.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

18

18

## ESTÁGIOS DE DESENVOLVIMENTO

- **Bloco #0 Frontier - O estágio inicial do Ethereum, durando de 30 de julho de 2015 a março de 2016.**
- Bloco #200.000 Ice Age - Um hard fork para introduzir um aumento exponencial de dificuldade, preparando a transição para o PoS.
- Bloco #1.150.000 Homestead - O segundo estágio do Ethereum, lançado em março de 2016.
- **Bloco #1.192.000 DAO - Um hard fork que reembolsou as vítimas do contrato DAO hackeado e fez com que o Ethereum e o Ethereum Classic se dividissem em dois sistemas concorrentes.**
- Bloco #2.463.000 Tangerine Whistle - Uma hard fork para alterar o cálculo do gás para certas operações de E/S e para limpar o estado acumulado de um ataque de negação de serviço (DoS) que explorou o baixo custo do gás dessas operações.
- Bloco #2.675.000 Spurious Dragon - Um hard fork para lidar com mais vetores de ataque DoS e outra limpeza de estado. Além disso, um mecanismo de proteção contra ataques de repetição.
- **Bloco #4.370.000 Byzantium - Recompensas de mineração é reduzida de 5 para 3 ETH.**
- **Bloco #7.280.000 Constantinople – Mudança significativa na rede, entre elas, ajustes para a implementação do POS e ajustes no custo do gas. (2019).**
- Bloco #12.244.000 Berlim - A atualização Berlim otimizou o custo do gás para certas ações de EVM e aumentou o suporte para vários tipos de transação.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

19

19

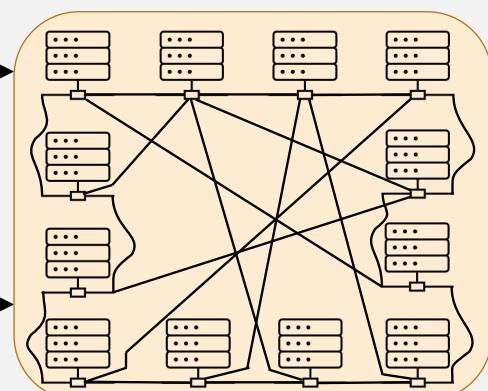
## CONEXÃO COM A REDE ETHEREUM

Para Desenvolvedores  
Java, Android e Javascript

Web3

IPC (Usa o sistema de arquivo local)  
Websockets (mais rápido que HTTP)  
HTTP (suportado pela maioria dos nós)

Rede Ethereum



Para Clientes

Cliente Remoto

Não armazenam a blockchain Ethereum completa;  
São mais rápidos de configurar;  
Requerem menos armazenamento de dados;  
Clientes remotos oferecem um subconjunto da funcionalidade de um cliente completo

Simpósio Brasileiro de Sistemas de Informação - Minicurso Blockchain, Contratos Inteligentes e Sistemas Web: Teoria e Prática

20

20

## CLIENTES ETHEREUM

### Nós Clientes

- Geth (Go)
- Parity (Rust)
- Cpp-ethereum(C++)
- Pyethereum(Python)
- Mantis(Scala)
- Harmony(Java)

### Clientes remotos

- Jaxx
- Status
- Trust Wallet
- Coinbase

### Extensões

- Metamask
- Jaxx
- MyEtherWallet
- Nifty Wallet
- MyCrypto

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

21

21

## CLIENTES REMOTOS

Gerenciar chaves privadas e endereços Ethereum em uma carteira;

Criar, assinar e transmitir transações;

Interagir com contratos inteligentes, usando a carga útil de dados;

Navegar e interagir com DApps;

Oferecer links para serviços externos, como exploradores de blocos;

Converter unidades de ether e recuperar taxas de câmbio de fontes externas;

Injetar uma instância web3 no navegador web como um objeto Javascript;

Usar uma instância web3 fornecida/injetada no navegador por outro cliente;

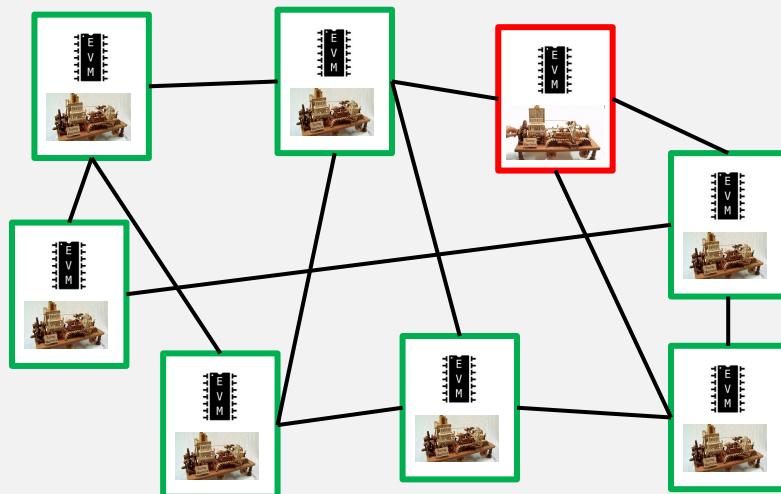
Acessar os serviços RPC em um nó Ethereum local ou remoto.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

22

22

## EVM – ETHEREUM VIRTUAL MACHINE



Implementa uma máquina de turing virtual para processar os contratos Inteligentes.

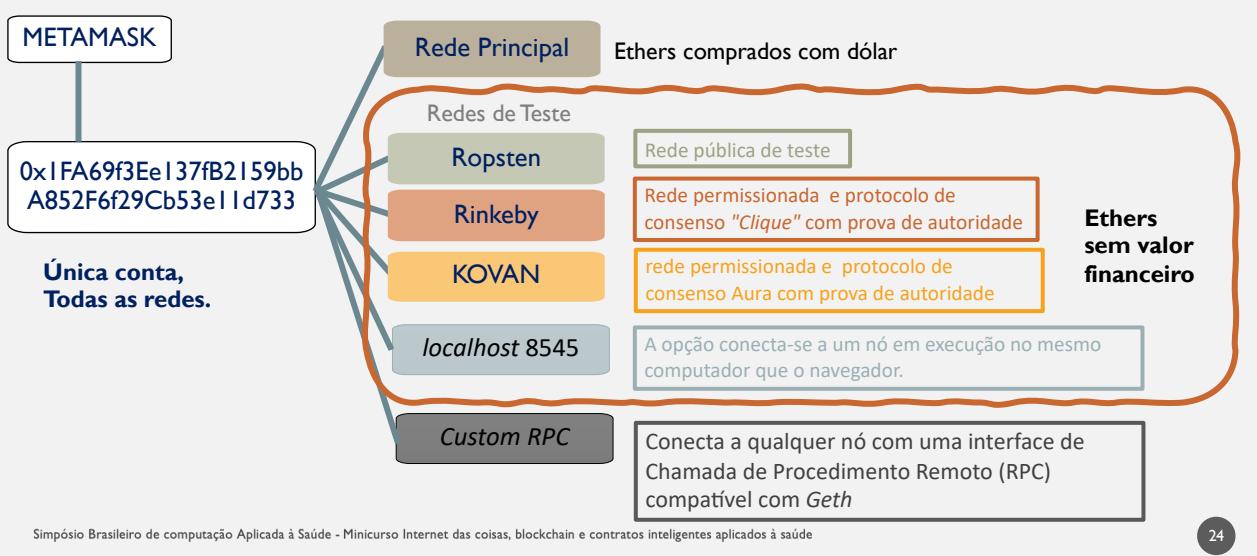
Usa o `gas` para controlar o uso da Máquina de Turing

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

23

23

## REDES ETHEREUM

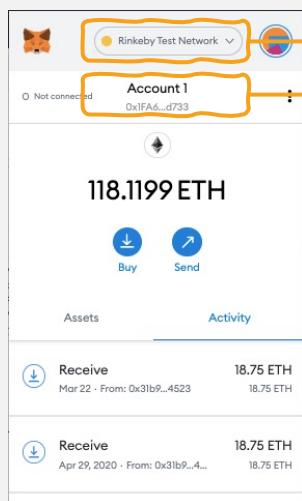


Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

24

24

## METAMASK



Rede ativa

Conta ativa

- Ele fornece acesso a todas as redes da plataforma com uma única conta.
- Instalado como extensão nos principais navegadores
- Ao instalar, você receberá 12 palavras mnemônicas (Guarda-as sob sigilo)
  - A ordem apresentada é importante
  - Usadas para recuperar a sua conta
  - Realizar transações usando a web3
- O procedimento para instalação do Metamask pode ser acessado na página do curso

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

25

## ESTRUTURA DA TRANSAÇÃO

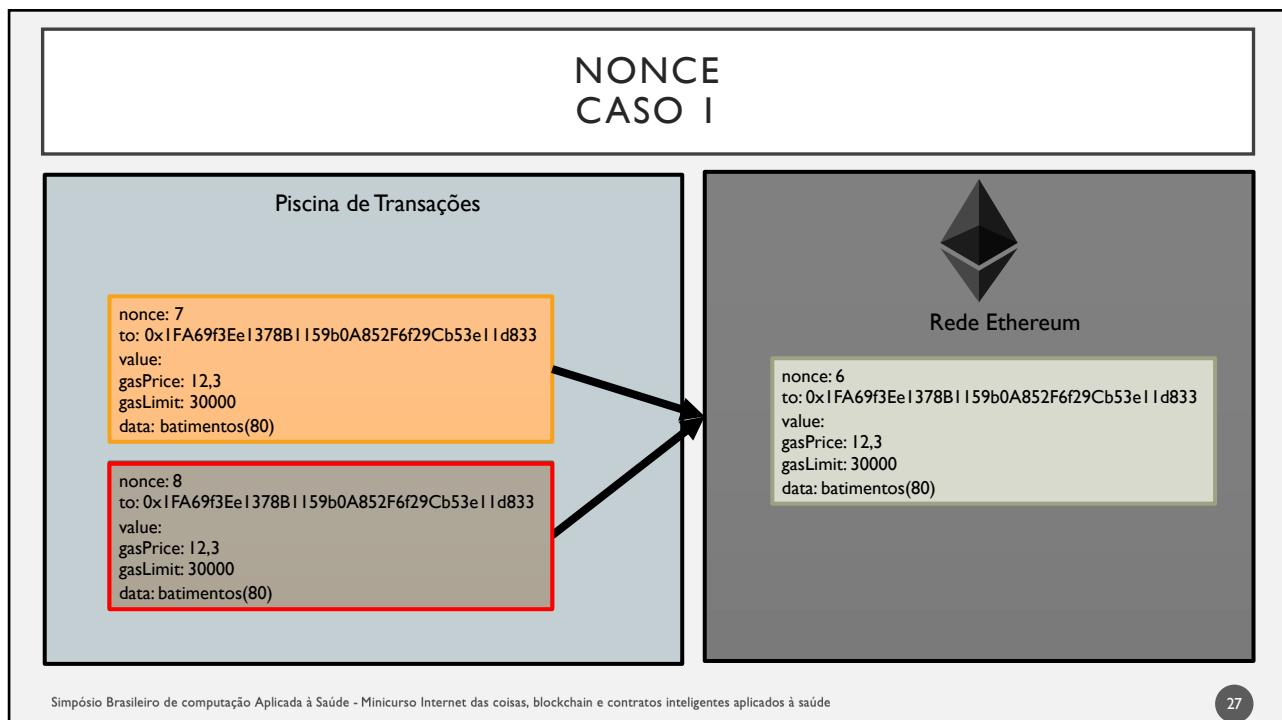
Acesse <https://andersbrownworth.com/> para ver um exemplo

Campo	Descrição
nonce	Número de sequência, emitido pela origem, usado para evitar a reprodução da mensagem
to	Endereço para onde será enviada uma quantidade de ether
value	Quantidade de ether a ser enviada
GasPrice	Valor que será pago por unidade de gas para que a transação seja processada
GasLimit	Quantidade de GAS que esta transação pode consumir
v	Peças criptográficas usadas para a validação do bloco.
r	
s	

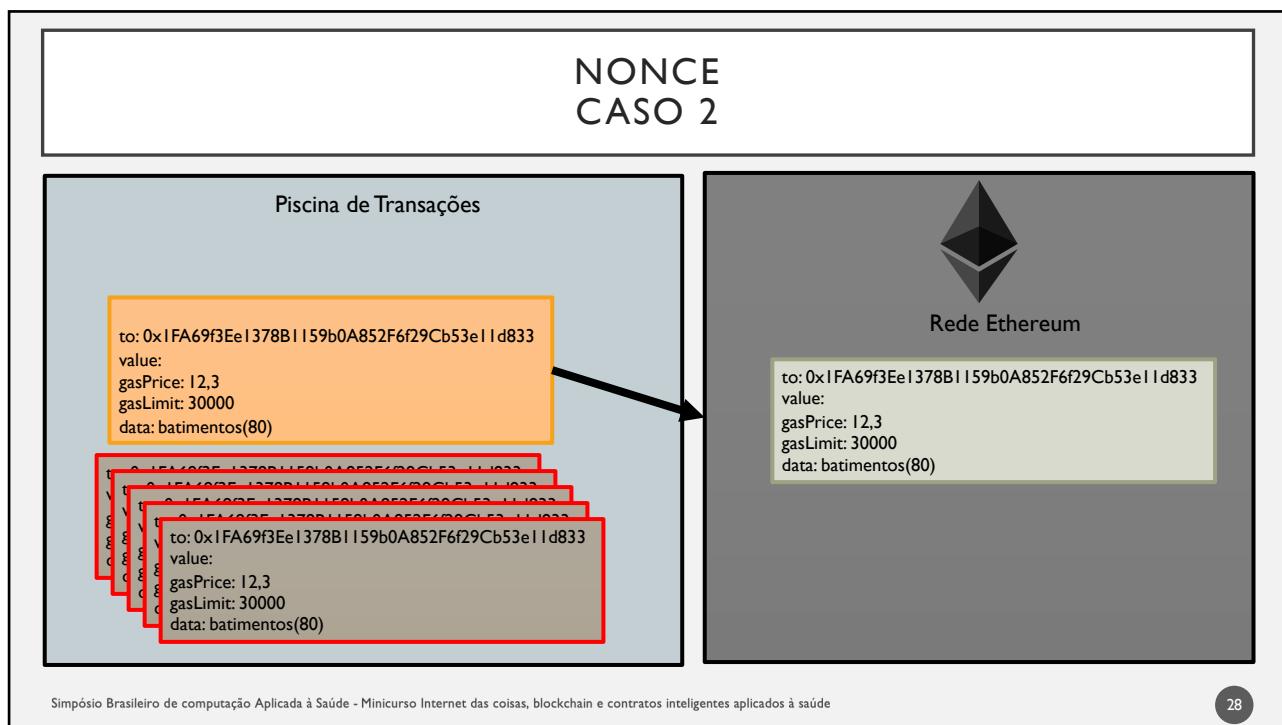
Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

26

26



27



28

## DATA E VALUE

Apenas consome gas

```
nonce: 7
to: 0x1FA69f3Ee1378B1159b0A852F6f29Cb53e11d833
value:
gasPrice: 12,3
gasLimit: 30000
data:
```

Indica um pagamento

```
nonce: 7
to: 0x1FA69f3Ee1378B1159b0A852F6f29Cb53e11d833
value: 2.3
gasPrice: 12,3
gasLimit: 30000
data:
```

Indica uma invocação

```
nonce: 7
to: 0x1FA69f3Ee1378B1159b0A852F6f29Cb53e11d833
value:
gasPrice: 12,3
gasLimit: 30000
data: batimentos(80)
```

Indica uma invocação e um pagamento

```
nonce: 7
to: 0x1FA69f3Ee1378B1159b0A852F6f29Cb53e11d833
value: 0,5
gasPrice: 12,3
gasLimit: 30000
data: pag_exame()
```

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

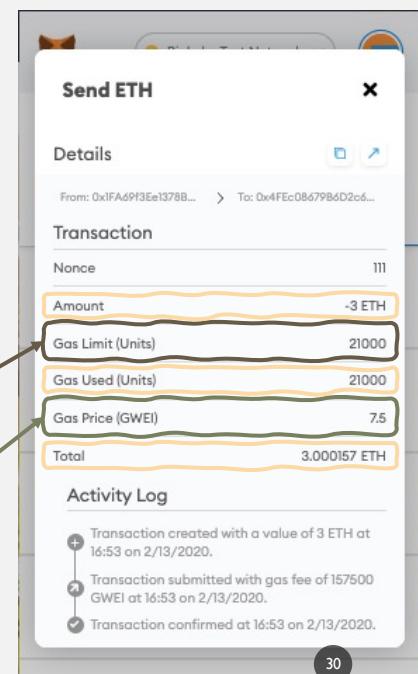
29

## GASPRICE X GASLIMIT

- O **gas** não é ether, é uma moeda separada usada para pagamento das transações na rede
- Possui cotação própria em relação ao ether

**GasLimit** - indica qual o limite de gas a ser consumido pela transação.

**GasPrice** - permite que o emissor da transação defina o preço que está disposto a pagar para adquirir o gas.



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

30

30

## CONSENSO

A Ethereum usa um protocolo de consenso de prova de trabalho (PoW) - Etash.

A prova de trabalho é executada por mineradores, que competem para criar novos blocos, com as transações processadas.

O vencedor compartilha o novo bloco com o resto da rede e ganha novos ethers recém-criados.

A corrida é ganha por quem resolver um quebra-cabeça matemático mais rápido

A rede é mantida segura pelo fato de que você precisaria de 51% do poder de computação da rede para fraudar a cadeia.

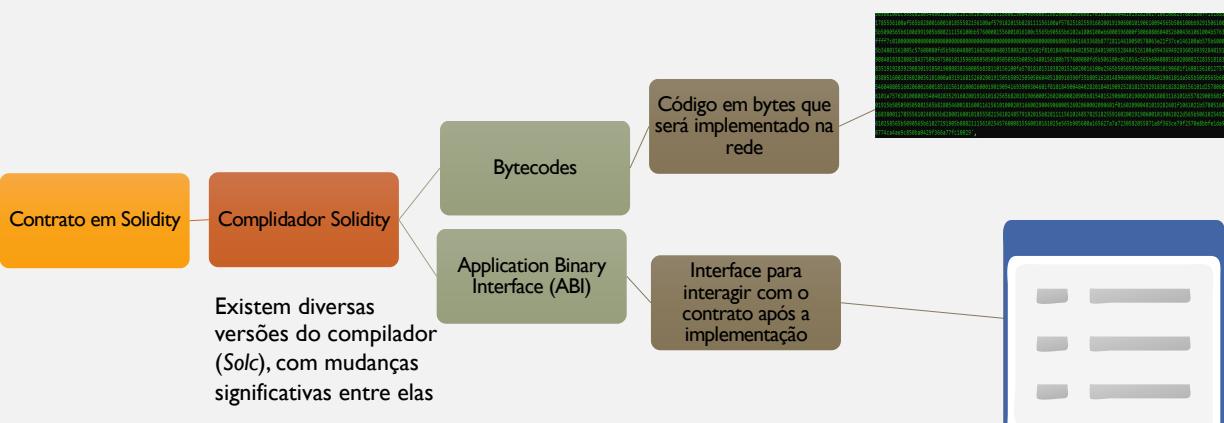
Isso exigiria investimentos tão grandes em equipamentos e energia, que provavelmente você gastaria mais do que ganharia.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

31

31

## CICLO DE DESENVOLVIMENTO DOS CONTRATOS INTELIGENTES

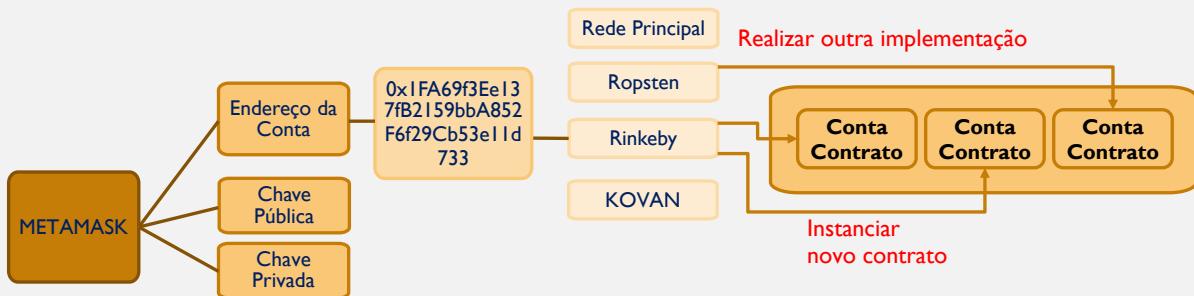


Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

32

32

## CONTRATOS INTELIGENTES



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

33

## TIPOS DE DADOS

Tipo	Descrição
int	Inteiros positivos ou negativos (int) declarados em incrementos de 8 bits (int8, int16, ... int256).
uint	Inteiros positivos declarados em incrementos de 8 bits (uint8, uint16... uint256).
bool	Valor lógico, verdadeiro ou falso, com operadores lógicos ! (não), && (e),    (ou), == (igual) e != (diferente).
fixed/ ufixed	Números de ponto fixo, declarados com (u) fixedMxN em que M é o tamanho em bits (incrementos de 8 até 256) e N é o número de decimais após o ponto (até 18); por exemplo, ufixed32x2.
address	Usado para armazenar endereços Ethereum de 20 bytes. O objeto de endereço tem muitas funções membro úteis, como balance (retorna o saldo da conta) e transfer (transfere ether para uma conta).
byte array (fixed)	Matrizes de bytes de tamanho fixo, declaradas com bytes.
byte array (dynamic)	Matrizes de bytes de tamanho variável, declaradas com bytes ou string.
enum	Tipo definido pelo usuário para enumerar valores discretos enum name {rotulo1, rotulo2...}.
array	Um array de qualquer tipo, fixo ou dinâmico.
struc	Containers de dados definidos pelo usuário para agrupar variáveis struct Car { String year; int color; }.
Mapping	Tabelas de pesquisa de hash para pares chave => mapping (key_type=>value_type).

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

34

34

## LINGUAGEM SOLIDITY

- Quando criamos uma transação e a enviamos para a rede, algumas informações podem ser encapsuladas na mensagem. Estas opções são pré-definidas pelo *Solidity* e agrupadas em 3 categorias:

### Objeto msg :

- msg.sender: endereço que iniciou a chamada de contrato
- msg.value: O valor de *ether* enviado (em *wei*).
- msg.gas: A quantidade de *gas* restante no suprimento de *gas* desse ambiente de execução. Isso foi descontinuado no *Solidity* v0.4.21 e substituído pela função *gasleft()*.
- msg.data: A carga útil de dados desta chamada no contrato.

### Objeto tx :

- tx.gasprice: o preço do gas na transação de chamada.
- tx.origin: O endereço da conta Ethereum de origem para esta transação.

### OBJETO block :

- block.blockhash(blockNumber): O *hash* de um bloco específico. Em desuso e substituído pela função *blockhash()* no *Solidity* v0.4.22.
- block.coinbase: O endereço do destinatário das taxas do bloco atual e da recompensa do bloco.
- block.difficulty: A dificuldade (prova de trabalho) do bloco atual.
- block.gaslimit: A quantidade máxima de *gas* que pode ser gasta em todas as transações.
- block.number: O número do bloco atual.
- block.timestamp: O carimbo de data/hora colocado no bloco atual pelo minerador.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

35

## MANIPULANDO ENDEREÇO DOS CONTRATOS

### address.balance:

- O saldo do endereço, em *wei*.

### address.transfer(quantidade)

- Transfere o valor (em *wei*) para este endereço.

### address.send(quantidade)

- Semelhante ao *transfer*, ele retorna falso em caso de erro.

### address.call(payload):

- pode construir uma chamada de mensagem arbitrária com uma carga de dados.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

36

36

## DECLARAÇÃO DE FUNÇÕES

- `function FunctionName [parâmetros] public|private internal|external [pure|constant|view|payable]`  
[modifier] [return (tipos de retorno)]

*FunctionName* é o nome usado para chamar a função. Uma

função pode ter parâmetros. Os parâmetros vêm após o nome, especificando os argumentos de *fallback*, que são chamadas quando a função não é nomeada.

Os parâmetros vêm após o nome, especificando os argumentos de *fallback*, que d

nomeada. tipos.

O padrão são funções pu

outros contratos, transações dentro do contrato.

As funções com o atributo `internal` só podem ser chamadas dentro do contrato.

As funções com o atributo `external`

mas não podem ser chamadas

As funções *external* são como funções públicas, exceto que não podem ser chamadas de dentro do contrato, a menos que

Uma função *pure* não lê nem grava nenhuma variável, apenas opera argumentos e retorna dados, sem referência a dados armazenados.

As funções *constant* ou *view* não modificam nenhum estado.

As funções *payable* podem aceitar pagamentos.

contrato. Elas não podem ser chamadas por outro contrato ou transações de carteiras Ethereum.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

37

37

### FUNÇÕES CONSTRUTORAS, MODIFICADORAS E AUTODESTRUTIVAS

#### Construtoras

- São executadas apenas uma vez durante a criação do contrato e possuem a palavra chave `constructor()`

#### Modificadoras

- São usadas para criar condições que se aplicam a muitas situações em um contrato. Desviam temporariamente a execução de uma função para uma rotina particular e possuem a palavra chave `modifier()`

#### Autodestrutivas

- As funções de autodestruição possuem a palavra-chave `destroy()` e são utilizadas para destruir o contrato implementado

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

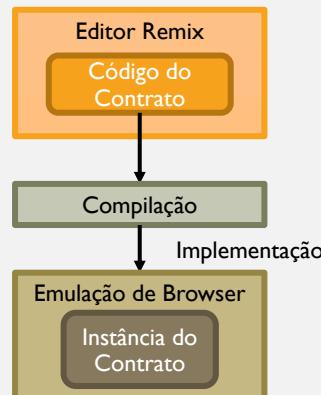
38

38

19

## CONSTRUINDO O PRIMEIRO CONTRATO USANDO O REMIX

- **Manipular mensagens**
- Uma variável pública para receber mensagens
- Uma função para definir uma mensagem inicial
- Uma função para modificar a mensagem inicial



The screenshot shows the Remix IDE interface. On the left, the Solidity Compiler interface is visible, showing the code for the `Inbox.sol` contract. The code defines a public string variable `message` and two functions: `setMessage` and `getMessage`. On the right, the contract details are shown, including the ABI and bytecode.

```

// SPDX-License-Identifier: MIT
pragma solidity ^0.7.4;

contract Inbox {
    string public message;
    constructor(string memory initialMessage) {
        message = initialMessage;
    }
    function setMessage(string memory newMessage) public {
        message = newMessage;
    }
    function getMessage() public view returns(string memory) {
        return message;
    }
}

```

<https://remix.ethereum.org/>

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

39

## INBOX.SOL

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.7.4;
3
4 contract Inbox{
5     string public message;
6     constructor(string memory initialMessage){
7         message = initialMessage;
8     }
9     function setMessage(string memory newMessage) public{
10         message = newMessage;
11     }
12     function getMessage() public view returns(string memory){
13         return message;
14     }
15 }

```

The code for `INBOX.SOL` is shown with annotations explaining its components:

- Line 1: `// SPDX-License-Identifier: MIT` → Identificador da licença do contrato
- Line 2: `pragma solidity ^0.7.4;` → Informa a versão do compilador
- Line 4: `contract Inbox{` → Informa o nome do contrato e declara as variáveis
- Line 5: `string public message;` → Declara o construtor
- Line 6: `constructor(string memory initialMessage){` → Declara a função para modificar o valor da variável
- Line 7: `message = initialMessage;` → Declara a função para ler o valor da variável
- Line 9: `function setMessage(string memory newMessage) public{` → Declara a função para modificar o valor da variável
- Line 12: `function getMessage() public view returns(string memory){` → Declara a função para ler o valor da variável
- Line 13: `return message;` → Declara a função para ler o valor da variável
- Line 15: `}` → Declara a função para ler o valor da variável

Os contratos somente executam funções se elas forem chamadas por uma transação.

Os contratos nunca podem chamar a si próprios ou atuarem em *background*, mas podem chamar outros contratos em cadeia.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

40

40

## PARTICULARIDADES NA EXECUÇÃO DOS CONTRATOS INTELIGENTES

A transação envolvendo Cls contém dois campos: **valor e dados**. Estes valores podem alternadamente serem preenchidos ou não. Quando o endereço de destino de uma transação for relativa a um contrato, a EVM executará o contrato e tentará chamar a função nomeada na carga útil de dados da transação.

As transações são atômicas.

A função getMessage neste exemplo é didática. Todas as variáveis declaradas no contrato, automaticamente, terão uma função get associada no momento da compilação.

Operações de leitura em contratos são gratuitas. Operações de implementação de contratos ou modificação de valores nos contratos são pagas.

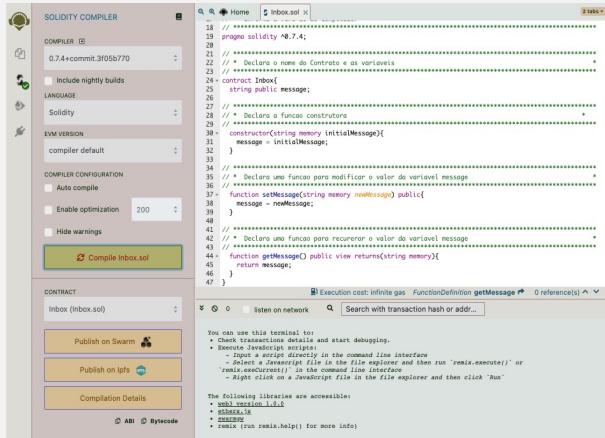
Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

41

41

## AMBIENTE DE DESENVOLVIMENTO

**Usando o Remix**

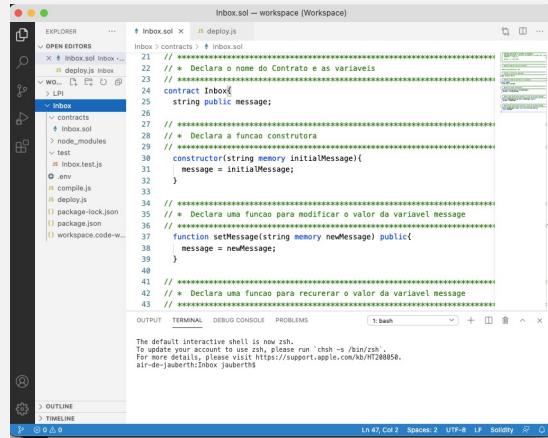


```

Solidity Compiler
COMPILER: 0.7.4+commit.3f05b770
LANGUAGE: Solidity
EVM VERSION: compiler default
COMPILE CONFIGURATION: Auto compile
COMPILE DETAILS: 200
COMPILE: Compile Inbox.sol
CONTRACT: Inbox (Inbox.sol)
  Publish on Swarm
  Publish on Infra
  Compilation Details
    ABI
    Bytecode
  
```

The screenshot shows the Remix interface with the Solidity compiler set to version 0.7.4. The code editor contains the Solidity code for the Inbox contract, which includes a constructor, a setMessage function, and a getMessage function. The Remix interface also shows options for publishing the contract to Swarm or Infra, and for viewing ABI and Bytecode details.

**Usando um ambiente configurado**



```

Inbox.sol — workspace (Workspace)
  contracts
    Inbox.sol
    node_modules
    test
      inbox.test.js
    .concrete
    .deploy
    .node
    .package-lock.json
    .package.json
    .workspace.code-workspace
  
```

The screenshot shows the VS Code interface with the Solidity extension installed. The workspace contains the Inbox.sol file and its associated deployment files. The Solidity extension provides features like code completion and navigation for Solidity contracts within the IDE environment.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

42

42

## PÁGINA DO MINICURSO

- <https://github.com/lifuesc/minicurso-blockchain>
- Códigos, instruções e material complementar
- Sempre em atualização

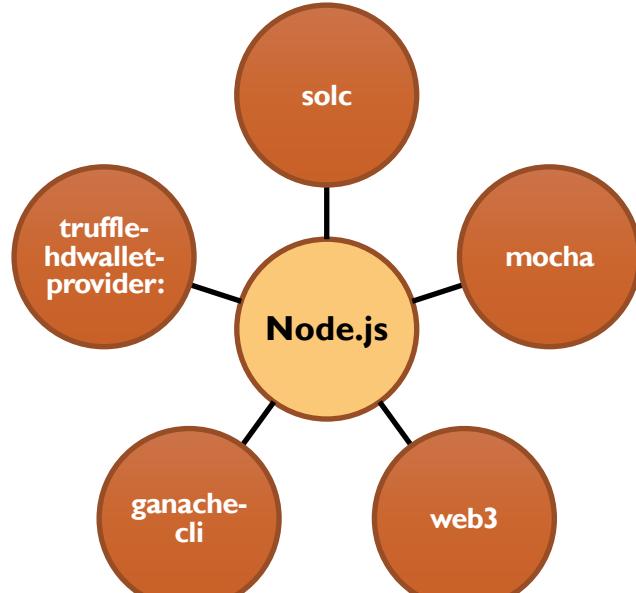


api	Controle de vacina	3 days ago
deploy	Controle de vacina	3 days ago
frontend	Controle de vacina	3 days ago
img	atualizado readme	8 hours ago
tutoriais	adicionando readme e tutoriais	3 days ago
.gitignore	Controle de vacina	3 days ago
README.md	Update README.md	4 minutes ago

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

43

## CONFIGURAÇÃO DE AMBIENTE LOCAL

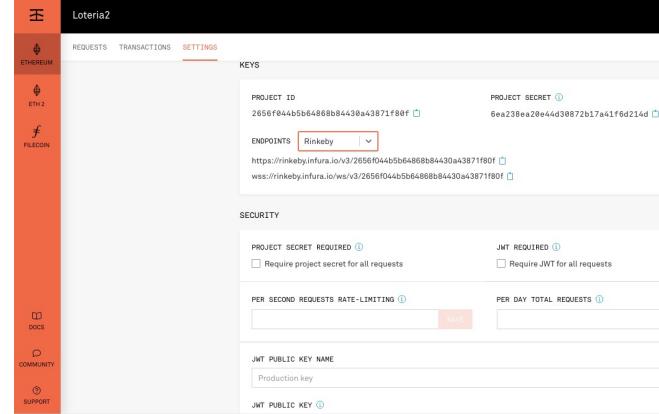


Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

44

44

## ACIONANDO UM NÓ



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

45

## DESAFIOS DA BLOCKCHAIN E IOT NA ÁREA DE SAÚDE

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

46

46

## DESAFIOS DA BLOCKCHAIN E IOT NA ÁREA DE SAÚDE

- Principais requisitos
  - Controle de acesso e não repudiação
  - Interoperabilidade
  - Compartilhamento de dados
  - Mobilidade
- Desafios Técnicos
  - Latência
  - Vazão
  - Consumo de Energia
  - Centralização
  - Privacidade
- Protocolos de Consenso
  - PoW
  - PoS
  - PoX
  - BFT
  - DAG

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

47

## CONTROLE DE ACESSO, AUTENTICAÇÃO E NÃO REPUDIAÇÃO



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

48

48

## INTEROPERABILIDADE E COMPARTILHAMENTO DE DADOS

The diagram shows two hospital buildings connected to a central cloud storage system. A red arrow points from one hospital to the cloud, and a yellow arrow points from the cloud back to the other hospital. A red 'X' is placed over a direct arrow between the two hospitals, indicating that data exchange is not direct but must pass through the central cloud.

**Armazenamento centralizado**  
**Diferenças de padrões entre sistemas**  
**Legislação de proteção de dados**  
**Ausência de Identificador comum**

```

    {"Dados_Pessoais":{ "name": "xxxxxxxx", "endereco": "yyyyyyyy" }, "internado": true, "sexo": "M", "idade": 35, "object": { "tipoSanguíneo": "A+", "Peso": 90 }, "Hemograma":{ "glicose":102, "colesterol":180 }, "Dados_cardiacos":{ "bpm":102, "bpmMax":138, "bpmMin":56 } }
  
```

```

<?xml version="1.0" encoding="UTF-8" ?>
<root>
<name>xxxxxxxx</name>
<internado>true</internado>
<sexo>M</sexo>
<idade>35</idade>
<object>
<tipoSanguíneo>A+</tipoSanguíneo>
<Peso>90</Peso>
</object>
<object>
<glicose>102</glicose>
</object>
</root>
  
```

Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

49

## MOBILIDADE

The icons include a camera, a wristwatch, a bandage, a signal tower, an ambulance, a pregnant woman, and a t-shirt.

Poucos conhecimentos adequados  
 Disponibilidade da rede  
 Autenticação de dados,  
 Confiabilidade  
 Localização.

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

50

## DESAFIOS TÉCNICOS

- Latência**
  - É preciso levar em consideração o tempo de bloqueio das plataformas de blockchain e a aplicação na qual será empregada
- Vazão**
  - O incremento do número de transações e o tempo de bloqueio para plataformas que usam algum tipo de prova podem diminuir a vazão e ser um fator proibitivo para algumas aplicações.
- Consumo energético**
  - Para blockchains que empregam o protocolo PoW, é preciso ponderar o alto consumo energético durante o processo de mineração dos blocos
- Centralização**
  - Alguns protocolos de consenso, apesar dos esforços e mecanismos para manter a justiça, tendem a centralizar os mineradores.
- Privacidade**
  - Os sistemas baseados em blockchain devem estar em conformidade com o Regulamento Geral de Proteção de Dados (GDPR)

aplicados à saúde
51

51

## PROTOCOLOS DE CONSENSO

Protocolos de Consenso							
PoX - Proof of Somethings							
PoW	PoS			BFT	DAG		
DPoS	PoL			PBFT	Tangle		
PoC	PoA			Tendermint	Ripple		
PoB	PoI			Stellar			
PoET							

aplicados à saúde
52

52

## POW – PROOF OF WORK

- Cada nó da rede precisa resolver um desafio computacional para poder propor à rede um bloco de transações
- Dois princípios básicos: *a regra da cadeia mais longa* e *a regra de incentivo*

MedRec	MedBChain	Guo e outros 2018	Conceição e outros 2018
<ul style="list-style-type: none"> <li>• Oferece aos pacientes registros imutáveis e de fácil acesso, gerenciando autenticação, confidencialidade, responsabilidade e compartilhamento</li> </ul>	<ul style="list-style-type: none"> <li>• É um sistema de gerenciamento de dados de saúde centrado no paciente usando blockchain como armazenamento para obter privacidade</li> </ul>	<ul style="list-style-type: none"> <li>• Valida registros eletrônicos encapsulados na blockchain empregando um esquema de assinatura baseada em atributos com várias autoridades para endossar as informações</li> </ul>	<ul style="list-style-type: none"> <li>• Implementam uma arquitetura de informação em larga escala para acessar registros eletrônicos de saúde com base em Contratos Inteligentes.</li> </ul>

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

53

## POS – PROOF OF STAKE

- Considera a porcentagem do número total de moedas (ou recursos) que um nó envolvido na competição detém, e eventualmente considera o tempo que o nó leva com o montante de moedas (ou recursos) para estabelecer uma porcentagem de direito de participação no consenso.

Patel 2019
<ul style="list-style-type: none"> <li>• Framework para compartilhamento de imagens entre domínios que usam blockchain como armazenamento de dados distribuído, para estabelecer um livro-razão de estudos radiológicos e permissões de acesso definidas pelo paciente.</li> </ul>

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

54

54

## PBFT – PRACTICAL BYZANTINE FAULT TOLERANCE

- Pertencem a uma classe que conseguem obter um acordo em um sistema, onde os processadores podem falhar de forma arbitrária, denominado Problema Geral Bizantino, descrito por [Lamport 1982]
- Foi o primeiro algoritmo prático a tolerar falhas bizantinas e adaptou-se para ser usado em ambientes assíncronos
- O BFT-Smart é oferecido pelo Hyperledger Fabric como camada de acordo

Dubovitskaya e outros 2017

- Estrutura para gerenciar e compartilhar dados para atendimento a pacientes com câncer

AuditChain

- Usa o Hyperledger Fabric para resolver problemas de interoperabilidade, conteúdo, estrutura e consolidação de log através do livro razão e contratos inteligentes, padronizando o conteúdo

MedChain

- Armazenamento distribuído para registros eletrônicos de saúde e informações de saúde protegidas

Medicalchain

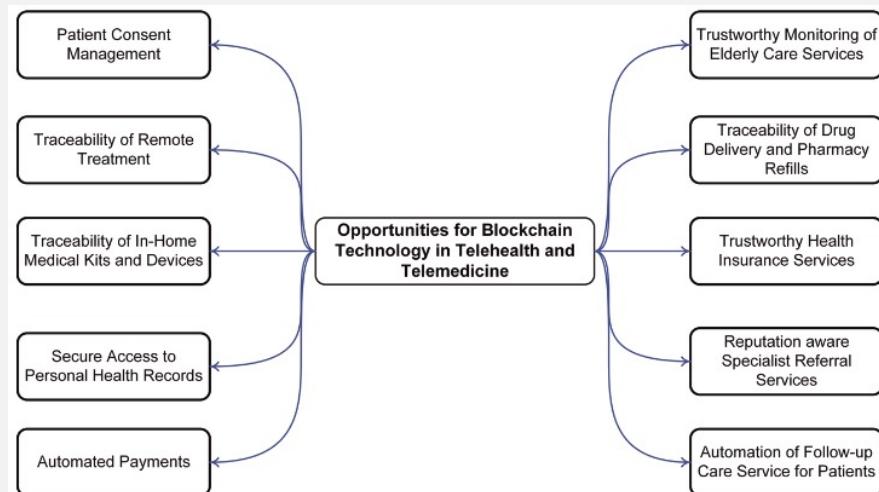
- Permite que o paciente forneça aos profissionais de saúde acesso aos registros e exames médicos de forma auditável, transparente e segura empregando MedTokens .

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

56

56

## PESQUISAS E APLICAÇÕES RECENTES



Fonte: [Ahmad e outros 2021]

60

60

## PESQUISAS E APLICAÇÕES RECENTES EXEMPLOS(2)

### Acesso seguro a registros pessoais de saúde

- Contratos inteligentes podem registrar e autorizar usuários a acessarem informações, em conformidade com a política de consentimento de seus proprietários [Shahnaz e outros 2019, Guo e outros 2019].

### Pagamentos automatizados

- A blockchain pode ser utilizada como plataforma para a realização de pagamentos no setor da tele saúde, através da transferência direta de tokens sem intermediários [Albeyatti 2018, Halamka e outros 2019].

### Monitoramento confiável de serviços de atendimento a idosos

- Pacientes idosos podem permanecer em suas residências e serem monitorados remotamente por biossensores acoplados em seus corpos [Kazmi e outros 2019, Salah e outros 2020].

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

61

## PESQUISAS E APLICAÇÕES RECENTES EXEMPLOS(2)

### Gerenciamento de dados clínicos do paciente

- Informações altamente confidenciais compartilhadas com segurança entre profissionais da área da saúde (médicos, farmacêuticos, hospitais, etc.) [Albeyatti 2018, Saweros e Song 2019]

### Rastreabilidade de tratamento remoto

- O acompanhamento remoto de pacientes utiliza plataformas computacionais que permitem a troca de dados digitalizados que ajudam especialistas na área de saúde a diagnosticar o estado clínico de seus pacientes, auxiliando-os em suas consultas e procedimentos cirúrgicos [Mannaro e outros 2018, Hussien e outros 2021]

### Rastreabilidade de kits e dispositivos médicos residenciais

- A adoção de kits e dispositivos de teste para uso domiciliar auxiliam no tratamento precoce de doenças, reduzindo os custos associados a exames médicos que antes eram realizados apenas em laboratórios e hospitais [Weissman e outros 2018]

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

62

62

## PESQUISAS E APLICAÇÕES RECENTES EXEMPLOS(3)

Rastreabilidade da entrega de medicamentos	Serviços de plano de saúde confiáveis	Serviços especialistas de recomendação baseados em reputação	Automação do serviço de acompanhamento de pacientes
<ul style="list-style-type: none"> <li>Medicamentos em trânsito podem ser rastreados pela farmácia e pelo paciente, que podem acompanhar o trajeto realizado pela encomenda [Thatcher e Acharya 2018].</li> </ul>	<ul style="list-style-type: none"> <li>Cientes que mantenham um seu estilo de vida saudável monitorado por dispositivos inteligentes presentes em academia, ou conectados ao paciente, podem realizar transações na blockchain para a persistência de tais informações [Raikwar e outros 2018, Albeatti 2018]</li> </ul>	<ul style="list-style-type: none"> <li>A telemedicina permite que dados remotos de pacientes possam ser acessados e analisados à distância por especialistas multidisciplinares [Lee 2019].</li> </ul>	<ul style="list-style-type: none"> <li>A tecnologia blockchain pode automatizar este serviço por meio de contratos inteligentes que disparar lembretes ao paciente para o envio de exames no sistema, ou para que ele não se esqueça do dia do encontro remoto com o seu médico [Siyal e outros 2019].</li> </ul>

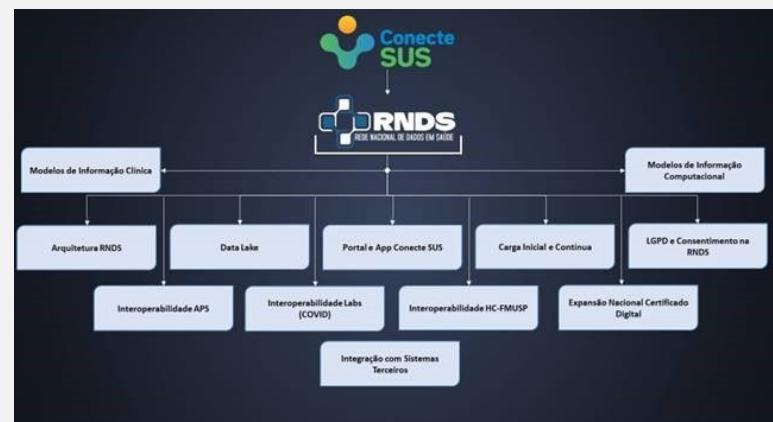
Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

63

63

## REDE NACIONAL DE DADOS EM SAÚDE

- Projeto estruturante do Conecte SUS, programa do Ministério da Saúde (MS) do Governo Federal, cujo objetivo é promover a troca de informações entre os pontos da Rede de Atenção à Saúde, permitindo a transição e continuidade do cuidado nos setores público e privado
- A RNDS é o maior case de blockchain em saúde do mundo com mais de 1 bilhão de transações
- Os documentos clínicos federados estão disponíveis juntamente com a linha do tempo do paciente, a qual está distribuída por meio da tecnologia blockchain
- Enquadrada na Lei no 13.709 - Lei Geral de Proteção a Dados (LGPD)



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

64

64

## DESENVOLVIMENTO DE CONTRATOS INTELIGENTES

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

65

65

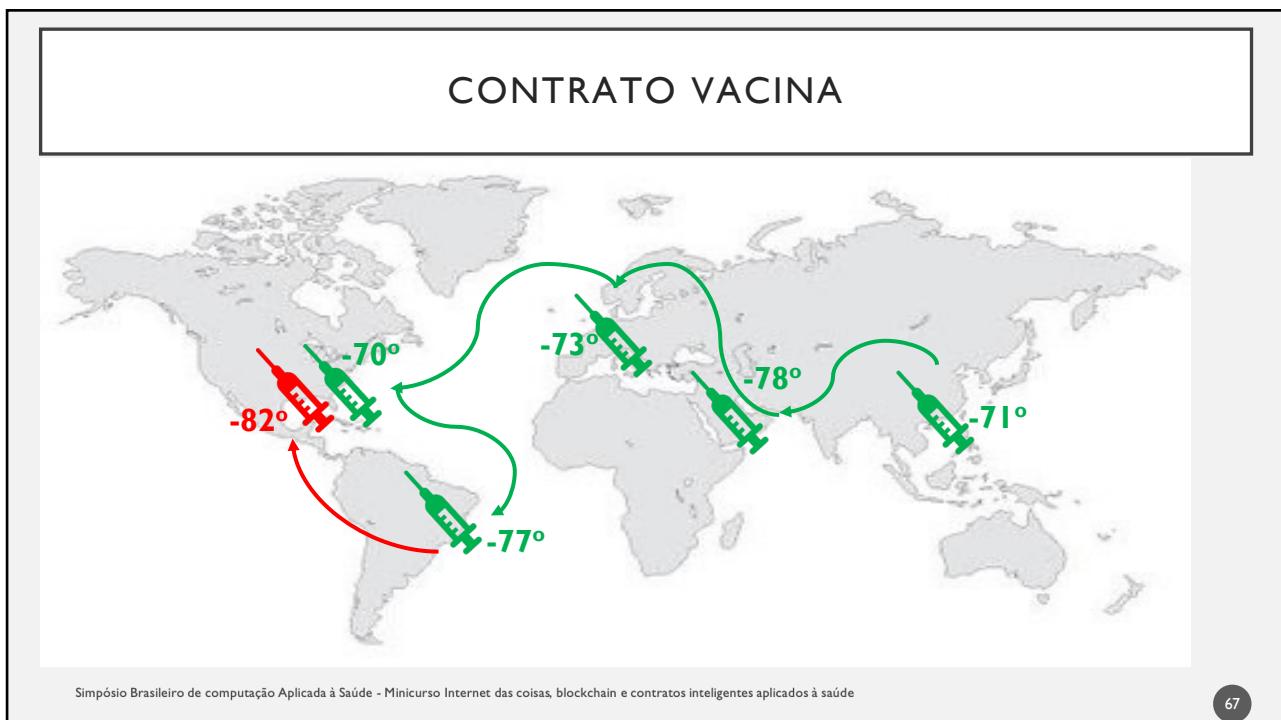
## DESENVOLVIMENTO DE CONTRATOS INTELIGENTES

- Construindo um contrato inteligente
- Configuração de ambiente local

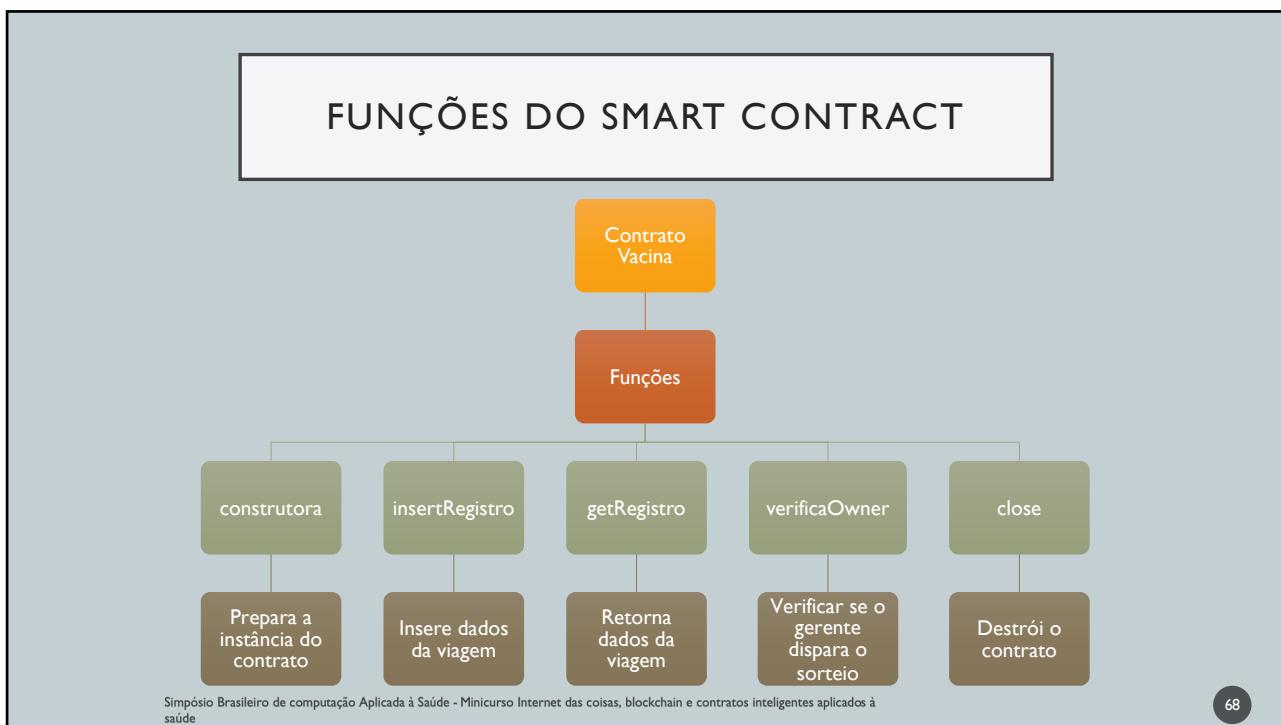
Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

66

66



67



68

## VARIÁVEIS DO SMART CONTRACT

### Structs

- Registros
- perdaVacina

### Variáveis

- owner
- nome
- origem
- destino
- duracao
- condicao
- tempMax
- tempMin

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

69

69

## VACINA.SOL E COMPILE.JS

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.4;
3 pragma abicoder v2;
4
5 // Estrutura de registro
6 struct Registro{
7     int16 value; // Temperatura da vacina
8     uint64 time; // Hora do registro
9     string local; // Local do registro
10 }
11
12 contract Vacina{
13     address public owner; // Proprietário do contrato
14     string public nome; // Nome/tipo da vacina
15     string public origem; // Local de origem da vacina
16     string public destino; // Destino da vacina
17     string public duracao; // Duração prevista
18     bool public condicao = true; // Condição da vacina, se está boa ou não
19     int public tempMax; // Temperatura máxima da vacina
20     int public tempMin; // Temperatura mínima da vacina
21     Registro[] public registros; // Histórico de registros
22     Registro public perdaVacina; // Registro para perda da vacina
23
24     constructor(
25         string memory _nome,
26         string memory _origem,
27         string memory _destino,
28         string memory _duracao,
29         int16 _tempMax,
30         int16 _tempMin
31     )
32     {
33         owner = msg.sender;
34         nome = _nome;
35         origem = _origem;
36         destino = _destino;
37         duracao = _duracao;
38         tempMax = _tempMax;
39         tempMin = _tempMin;
40     }
41
42     function registrar() external {
43         Registro memory novoRegistro;
44         novoRegistro.value = tempMax;
45         novoRegistro.time = uint64(block.timestamp);
46         novoRegistro.local = "Brasil";
47         registros.push(novoRegistro);
48     }
49
50     function consultarHistorico() external view returns (Registro[] memory) {
51         return registros;
52     }
53
54     function perderVacina() external {
55         perdaVacina = Registro();
56     }
57
58     function recuperarVacina() external {
59         require(owner == msg.sender, "Somente o proprietário pode recuperar a vacina");
60         require(perdaVacina.value <= tempMax, "A vacina não pode ser recuperada se estiver com temperatura acima do limite");
61         perdaVacina = Registro();
62     }
63
64     function getOwner() external view returns (address) {
65         return owner;
66     }
67
68     function getNome() external view returns (string) {
69         return nome;
70     }
71
72     function getOrigem() external view returns (string) {
73         return origem;
74     }
75
76     function getDestino() external view returns (string) {
77         return destino;
78     }
79
80     function getDuracao() external view returns (string) {
81         return duracao;
82     }
83
84     function getMaxTemp() external view returns (int) {
85         return tempMax;
86     }
87
88     function getMinTemp() external view returns (int) {
89         return tempMin;
90     }
91
92     function getRegistrados() external view returns (Registro[] memory) {
93         return registros;
94     }
95
96     function getPerdaVacina() external view returns (Registro) {
97         return perdaVacina;
98     }
99
100 }
```

```

1 const path = require("path");
2 const fs = require("fs");
3 const solc = require("solc");
4
5 // ! Alterar aqui
6 // !
7 // Nome da classe do contrato
8 const contractName = "Vacina";
9 // Nome do arquivo do contrato baseado no nome da classe
10 const contractFileName = `Vacina.sol`;
11 // !
12
13 // Pega o diretório do arquivo do contrato
14 const ContractPath = path.resolve(__dirname, "contracts", contractFileName);
15 // Conteúdo do contrato convertido
16 const sourceContract = fs.readFileSync(ContractPath, "utf8");
17
18 // Configuração do SOLC
19 const input = {
20     language: "Solidity",
21     sources: {
22         [contractFileName]: {
23             content: sourceContract,
24         },
25     },
26     settings: {
27         outputSelection: {
28             "*": {
29                 "*": ["*"],
30             },
31         },
32     },
33 };
34
35 const compiledContract = solc.compile(input);
36
37 const bytecode = compiledContract.contracts[contractName].bytecode;
38 const abi = compiledContract.contracts[contractName].abi;
39
40 module.exports = { bytecode, abi };
41
42 // 
```

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

70

70

## VACINA.SOL E COMPILE.JS

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.4;
3 pragma abicoder v2;
4
5 // Estrutura de registro
6 struct Registro{
7     int16 value; // Temperatura da vacina
8     uint64 time; // Hora do registro
9     string local; // Local do registro
10 }
11
12 contract Vacina{
13     address public owner; // Proprietario do contrato
14     string public nome; // Nome/tipo da vacina
15     string public origem; // Local de origem da vacina
16     string public destino; // Destina da vacina
17     string public duracao; // Duração prevista
18     bool public condicao = true; // Condição da vacina, se está boa ou não
19     int public tempMax; // Temperatura maxima da vacina
20     int public tempMin; // Temperatura minima da vacina
21     Registro[] public registros; // Historico de registros
22     Registro public perdaVacina; // Registro para perda da vacina
23
24     constructor(
25         string memory _nome,
26         string memory _origem,
27         string memory _destino,
28         string memory _duracao,
29         int16 _tempMax,
30         int16 _tempMin
31     )
32 }
```

```

1 const path = require("path");
2 const fs = require("fs");
3 const solc = require("solc");
4
5 // ! Alterar aqui
6 // !
7 // Nome da classe do contrato
8 const contractName = "Vacina";
9 // Nome do arquivo do contrato baseado no nome da classe
const contractFileName = `Vacina.sol`;
10 // !
11
12 // Pega o diretório do arquivo do contrato
13 const ContractPath = path.resolve(__dirname, "contracts", contractFileName);
14 // Conteúdo do contrato convertido
15 const sourceContract = fs.readFileSync(ContractPath, "utf8");
16
17 // Configuração do SOLL
18 const input = {
19     language: "Solidity",
20     sources: {
21         [contractFileName]: {
22             content: sourceContract,
23         },
24     },
25     settings: {
26         outputSelection: {
27             "*": {
28                 "*": ["*"],
29             },
30         },
31     },
32 }
```

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

71

71

## DEPLOY.JS

```

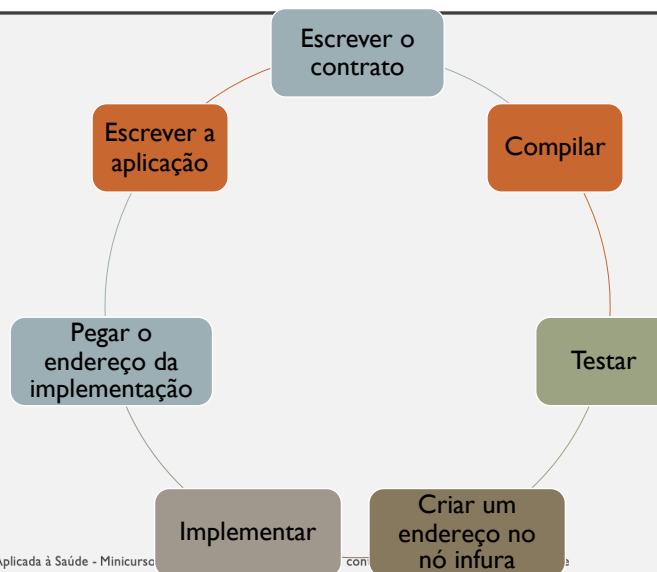
1 require("dotenv").config();
2 const HDWalletProvider = require("@truffle/hdwallet-provider");
3 const Web3 = require("web3");
4 const { abi, evm } = require("./compile");
5
6 // Configuração do provider do infura
7 const provider = new HDWalletProvider({
8     mnemonic: { phrase: process.env.MNEMONIC },
9     providerOrUrl: process.env.RINKEBY_INFURA,
10 });
11
12 const web3 = new Web3(provider);
13
14 const deploy = async () => {
15     // Pega as contas do navegador
16     const accounts = await web3.eth.getAccounts();
17     // Pega a primeira conta que será utilizada no deploy
18     const deploymentAccount = accounts[0];
19     // Gera a chave primária a partir da carteira
20     const privateKey =
21         provider.wallets.accounts[0].toLowerCase().privateKey.toString("hex");
22     // Mostra carteira utilizada para deploy
23     console.log(`Conta usada para o deploy ${accounts[0]}`);
24
25     try {
26         // Prepara uma instância do contrato para assinatura
27         let contract = await new web3.eth.Contract(abi)
28             .deploy({
29                 data: evm.bytecode.object,
30                 // Nome/tipo da vacina, Origem, Destino, Duração, temperatura máxima, temperatura mínima
31                 arguments: ["Vacina Covid-19 Pfizer", "China", "Brasil", "45°", -70, -80],
32             })
33     } catch (error) {
34         console.error(error);
35     }
36 }
```

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

72

72

## CICLO PARA DESENVOLVER UM CONTRATO



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso

73

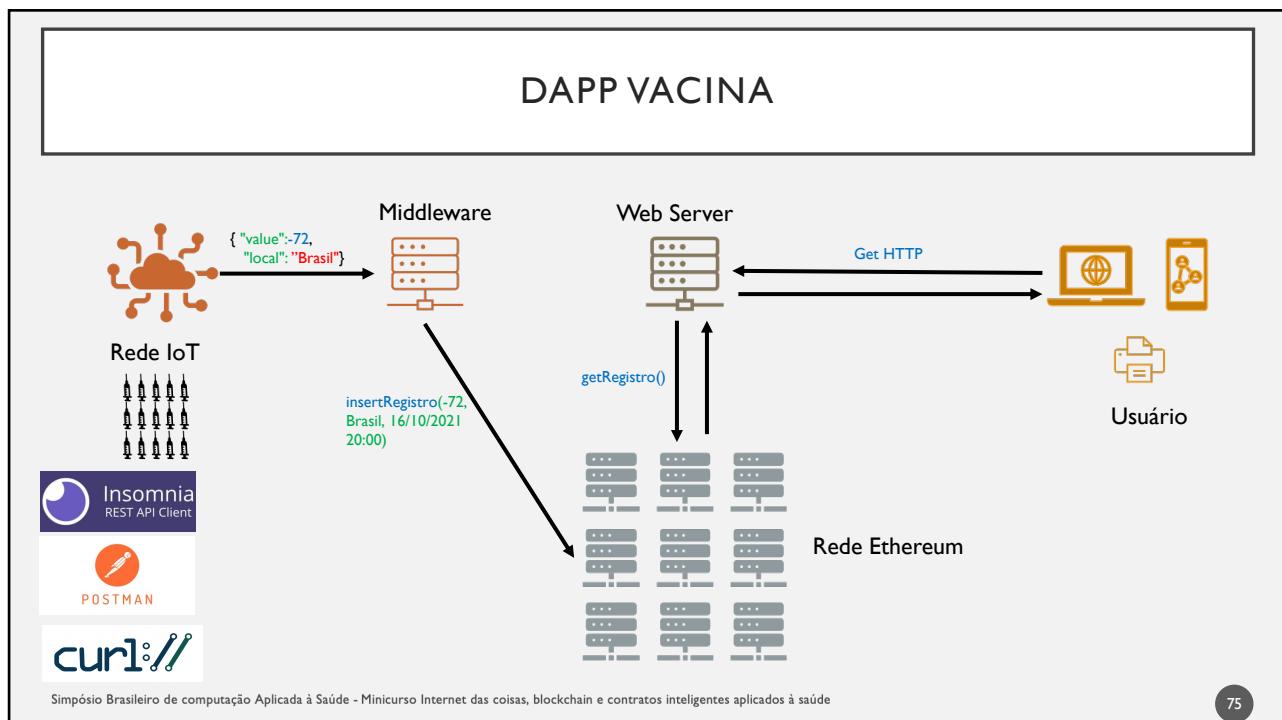
## DESENVOLVIMENTO DE DAPPS

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

74

74

35



75

## CONTRATO DE LOTERIA

### Sistema de Rastreamento de Vacinas

**Vacina Covid-19 Pfizer**

**Contrato:** [0xB5577f5f1967ba2321efC1503e5EaCA82404682e](#)

Origem	Destino
China	Brasil
Estimativa de tempo	Condição da vacina
40h	Imprópria
Temperatura máxima	Temperatura mínima
-70	-80

**Violação de Temperatura**

Temperatura	Local
-82	Nova Iorque
Data	
10/06/2021 11:50:32	

**Dados cadastrados no servidor**

Id	Valor	Time	Local
1	-71	10/06/2021 11:45:36	China
2	-78	10/06/2021 11:47:50	Dubai
3	-73	10/06/2021 11:48:07	Londres
4	-70	10/06/2021 11:48:28	Nova Iorque
5	-77	10/06/2021 11:48:40	São Paulo
6	-82	10/06/2021 11:50:32	Nova Iorque

Selecionar o id que deseja enviar para o contrato

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

76

76

## ARQUIVOS DO MIDDLEWARE

```
server.js x
① sbcas > vacina_v2 > api > server.js > ...
1 // Importa express
2 const express = require("express");
3 // Roda configuração do express e atribui ao app
4 const app = express();
5 // Configuração de acesso a diferentes origens
6 const cors = require("cors");
7 // Porta que irá rodar o servidor
8 const port = 3333;
9
10 // Atribui configuração do cors no express
11 app.use(cors());
12 // Configuração para o express receber requisição json
13 app.use(express.json());
14
15 // Váriavel que contém as leitura
16 const readings = [];
17
18 // Rota de inserção de dados
19 app.post("/insert-data", function (req, res) {
20   // Pega as variáveis no corpo da requisição
21   const { value, local } = req.body;
22   try {
23     const time = new Date().getTime();
24     readings.push({ value, time, local });
25
26     // Retorna status de sucesso
27     res.status(201).json({ message: "Success!" });
28   } catch (error) {
29     // Retorna status de erro e mostra o erro
30     res.status(400).json({ message: "Error", error });
31   }
32});
```

itoss inteligentes  
slicados à saúde

77

77

## ARQUIVOS DO FRONTEND

### src/contracts/vacina.contracts.js

- Contém o endereço do contrato implementado na rede Ethereum e a ABI do contrato.

### src/contrats/web3.js

- Importa a web3 para a aplicação

### src/app.js

- Contém o código javascript que acessa a DApp e cria a tela da aplicação

### src/index.js

- arquivo principal

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes  
aplicados à saúde

78

78

## COMO MONTAR UM CURSO DE DESENVOLVIMENTO WEB COM BLOCKCHAIN E CONTRATOS INTELIGENTES

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

79

79

### COMO MONTAR UM CURSO

Não é uma tarefa trivial

- Solidity
- Front-end
- Back-end

Uma alternativa são as plataformas  
MOOC (*Massive Open Online Course*)

- Coursera
- Udemy
- edX

Curso de Extensão

- Referencias no livro texto

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos  
inteligentes aplicados à saúde

80

80

## EXPERIÊNCIA DOS AUTORES

- 3 cursos - UFBA, UESC e UESB
- Módulo Básico
  - Teoria e conceitos básicos
  - Criação de um contrato simples
  - Criação de uma DApp single-page
- Módulo Avançado
  - Emprego de conceitos de engenharia de software
  - Mais recursos da linguagem solidity
  - DApp multipágina
- Módulo com IoT
  - Uso de IoT para gerar dados
  - Emprego de um middleware como intermediário
  - O Contrato pode gerar alertas
- Hardware
  - Ambiente misto
  - processadores que vão desde Core2 Duo com 4Gb de RAM até Core i7 com 32 Mb de RAM.
- Software
  - Node.js
  - Bibliotecas e Editores gratuitos
- Carga Horária
  - 16h para o módulo Básico
  - 16h para o módulo Avançado
  - 20h para o módulo com IoT

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

81

81

MAIS RECURSOS

CryptoZombies

Ethernaut

Vyper Tutorials

Ethereum Studio

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

82

82

## DESAFIOS E PERSPECTIVAS

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

83

83

## DESAFIOS E PERSPECTIVAS

- Os Desafios organizacionais para a adoção de blockchain esbarra nos sistemas tradicionais de telemedicina
- No entanto, a falta de consciência, imaturidade da tecnologia e indisponibilidade de padrões de segurança e privacidade impedem os atores de empregar plenamente os benefícios da blockchain, inclusive os relacionados aos incentivos monetários para as organizações participantes [Kolan e outros ].
- Os contratos inteligentes ainda possuem riscos de adulteração e segurança e isto pode afetar significativamente o histórico médico de um paciente, como um ataque de vulnerabilidade [Liu e outros 2018];
- O emprego de *sidechains* ou uma camada na névoa pode ajudar a minimizar a taxa de transação [Debe e outros 2019].
- Construir plataformas de blockchain interoperáveis é desafiador devido a vários problemas, como diferenças nas linguagens suportadas e protocolos de consenso das plataformas de blockchain [Herlihy 2018].

Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

84

84

## PERGUNTAS

Dúvidas e mais informações:

**Jauberth Abijaude**  
jauberth@uesc.br

**Henrique Serra**  
haserra.cic@uesc.br



Simpósio Brasileiro de computação Aplicada à Saúde - Minicurso Internet das coisas, blockchain e contratos inteligentes aplicados à saúde

85