FACULDADE DE EDUCAÇÃO TECNOLÓGICA DO ESTADO DO RIO DE JANEIRO - FAETERJ

CURSO DE TECNÓLOGO EM ANÁLISE DE SISTEMAS

**GILMAR RIBEIRO SANTANA**

**SISTEMA DE CONTROLE DE PONTO DE FUNCIONÁRIOS BASEADO TECNOLOGIA BLOCKCHAIN**

**RIO DE JANEIRO – RJ**

**2023**

**SISTEMA DE CONTROLE DE PONTO DE FUNCIONÁRIOS BASEADO TECNOLOGIA BLOCKCHAIN**

Trabalho de Conclusão de Curso – TCC - apresentado ao Curso de Tecnólogo de Análise de Sistemas da Faculdade de Educação Tecnológica do Estado do Rio de Janeiro – FAETERJ – como pré-requisito para a obtenção do grau de Tecnólogo em Análise de Sistemas, sob orientação do Prof. André Neves.

**RIO DE JANEIRO – RJ**

**2023**

**SUMÁRIO**

**[1 INTRODUÇÃO 3](#_Toc26855)**

**[2 PROBLEMA 4](#_Toc13019)**

**[3 JUSTIFICATIVA 4](#_Toc7378)**

**[4 OBJETIVO GERAL 10](#_Toc13670)**

**[5 OBJETIVOS ESPECÍFICOS 12](#_Toc7796)**

**[6 FUNDAMENTAÇÃO TEÓRICA 13](#_Toc26720)**

**[6.1 BLOCKCHAIN ..... 13](#_Toc25353)**

**[6.1.1 HISTÓRIA ............ 13](#_Toc29363)**

**[6.1.2 FUNCIONAMENTO 14](#_Toc27118)**

**[6.1.3 CRIPTOGRAFIA E VALIDAÇÃO 16](#_Toc25090)**

**[6.1.4 MECANISMO DE HASH 24](#_Toc12076)**

**[6.1.5 ÁRVORE DE MERKLE 33](#_Toc31600)**

**[6.1.6 REDE DESCENTRALIZADA 37](#_Toc16267)**

**[6.1.7 PROTOCOLO DE CONSENSO 40](#_Toc27638)**

**[6.1.7.1 PROOF OF WORK (PoW) – PROVA DE TRABALHO 41](#_Toc25134)**

**[6.1.7.2 PROOF OF STAKE (PoS) – PROVA DE PARTICIPAÇÃO 43](#_Toc7183)**

**[6.1.7.3 PROOF OF AUTORITY (PoA) – PROVA DE AUTORIDADE 46](#_Toc11259)**

**[6.1.8 ESTRUTURA DO BLOCO 46](#_Toc18939)**

**[6.2 EVOLUÇÃO DA WEB 47](#_Toc13153)**

**[6.3 SOLUÇÕES BLOCKCHAIN 51](#_Toc26349)**

**[6.4 SMART CONTRACTS 55](#_Toc3902)**

**[6.5 ETHEREUM ........ 56](#_Toc22240)**

**[6.5.1 ETHEREUM VIRTUAL MACHINE (EVM) 58](#_Toc25065)**

**[6.5.2 CONTAS .............. 61](#_Toc28652)**

**[6.5.3 TRANSAÇÕES .... 63](#_Toc31160)**

**[6.5.4 PATRICIA MERKLE TREE 64](#_Toc15687)**

**[7 METODOLOGIA CIENTÍFICA 65](#_Toc6769)**

**[7.1 QUESTIONÁRIO 66](#_Toc27486)**

**[8 SOLUÇÕES UTILIZADAS NO MERCADO 73](#_Toc24455)**

**[8.1 MODELO DE NEGÓCIO 74](#_Toc11690)**

**[9 ESPECIFICAÇÕES DO PROJETO 75](#_Toc25885)**

**[9.1 REGRAS DE NEGÓCIO 75](#_Toc13350)**

**[9.2 MODELAGEM DE PROCESSOS 77](#_Toc5652)**

**[9.3 REQUISITOS 80](#_Toc13723)**

**[9.4 CASOS DE USO 82](#_Toc12223)**

**[9.4.1 UC00 - Geral 83](#_Toc32025)**

**[9.4.2 UC01 - Manter Administrador 84](#_Toc20456)**

**[9.4.3 UC02 - Manter Empregador 85](#_Toc7248)**

**[9.4.4 UC03 - Manter Funcionário 87](#_Toc18509)**

**[9.4.5 UC04 - Emitir Relatório 88](#_Toc12715)**

**[9.4.6 UC05 - Marcar Ponto 89](#_Toc2754)**

**[9.5 DIAGRAMA DE CLASSE 90](#_Toc26718)**

**[9.6 DIAGRAMA DE SEQUÊNCIA 95](#_Toc23443)**

**[9.7 CONSIDERAÇÕES NO DESENVOLVIMENTO COM SOLIDITY 96](#_Toc18887)**

**[9.8 ARQUITETURA DO SISTEMA 98](#_Toc9190)**

**[10 TECNOLOGIA UTILIZADA NO APLICATIVO 98](#_Toc29381)**

**[11 POTENCIALIDADES DE MERCADO 98](#_Toc19493)**

**[12 CRONOGRAMA DA PESQUISA 100](#_Toc5374)**

**[13 REFERÊNCIAS 100](#_Toc1719)**

# 1 INTRODUÇÃO

A tecnologia blockchain tem se apresentado como uma das soluções mais disruptivas e inovadoras da atualidade. Palavras como criptomoedas, NFC, e redes descentralizadas já começam a ser mais difundidas e despertam curiosidade sobre o que esses termos significam e qual sua utilidade.

Neste contexto de inovação, a tecnologia blockchain, também conhecida como tecnologia de livro razão distribuído (Decentralized Ledger Technology – DLT), traz as bases para este novo mundo que é ferramenta eficaz para atingir os alvos da Web 3.0.

Inicialmente, a tecnologia blockchain foi utilizada para atender um novo produto, um novo mercado, uma nova solução financeira: as criptomoedas, mais especificamente, o Bitcoin, sendo a primeira e até hoje a mais sólida e conhecida das criptomoedas. Hoje, essa tecnologia disruptiva já é utilizada em diversos cenários trazendo um grau a mais de segurança, sigilo e velocidade em muitos outros negócios.

Já não são isoladas as soluções na área de educação, logística, engenharia social, IoT, inteligência artificial, entre outras, além da área financeira, é claro, que utilizam esta inovação.

Assim sendo, o presente trabalho também faz uso desta tecnologia apresentando um sistema e controle de ponto de funcionários armazenando os dados e as mudanças de estado em uma rede pública que utiliza a tecnologia de livro razão distribuído, ora chamada, blockchain.

# 2 PROBLEMA

Na empresa em que trabalho utilizamos um sistema de controle de ponto que efetua os registros através do uso de aparelho de leitura biométrica e certa vez, olhando meus bolsos no fim do dia, encontrei 4 comprovantes referentes àquele dia de trabalho. Imediatamente pensei, 4 comprovantes por dia aplicado a 22 dias úteis no mês, isso gera 88 comprovantes por mês e 1056 comprovantes por ano. Agora imaginem um funcionário com 5 anos de empresa, que não é algo incomum, seriam 5280 comprovantes.

Diante deste cenário, percebi o quão inviável é para qualquer funcionário gerenciar tais registros, sem falar que os comprovantes são gerados utilizando papel térmico e que se não tiverem um armazenamento adequado, longe do calor e luz intensa, podem se apagar. Por outro lado, o empregador muitas vezes é contestado em juízo e dentro da própria rotina diária por funcionários que se sentem lesados pelos registros, havendo, então, desconfiança por ambas as partes.

Diante de um conflito de interesses tão grande haveria uma solução viável capaz de atender os preceitos técnicos de confiança, segurança e imutabilidade que fosse ponto pacífico entre as duas partes (empregado e empregador) e que fornecesse os meios necessários para que o juízo trabalhista também se sinta confortável em dirimir tais tipos de conflitos? Esta pergunta é a que o presente trabalho pretende responder.

# 3 JUSTIFICATIVA

Conforme estabelecido pela Lei da Liberdade Econômica[[1]](#footnote-0) as empresas com mais de 20 trabalhadores devem fazer uso de um mecanismo para controle de ponto, mecanismo no qual serão realizadas as anotações da jornada de trabalho, com marcações de início e fim da jornada de trabalho bem como o início e fim da pausa.

Como requisitos mínimos e padronização destes dispositivos, foi introduzido no ordenamento jurídico a portaria 671/2021[[2]](#footnote-1) do Ministério do Trabalho e Previdência (MTP) que entre várias disposições, trata sobre a anotação da hora de entrada e de saída em registro manual, mecânico ou eletrônico. Estas disposições atualizam as portarias 1510/2009[[3]](#footnote-2) e 373/2011[[4]](#footnote-3), ambas do Ministério do Trabalho e Emprego (MTE). A primeira trata sobre o uso de relógio eletrônico de ponto (REP), definindo configurações mínimas, informações obrigatórias e aspectos de segurança do equipamento, já a segunda, atendendo à modernização tecnológica, reconhece os programas e aplicativos utilizados para este fim.

A portaria 671/2021 utiliza a Seção IV Subseção I para definir os meios eletrônicos para o controle da jornada, reconhecendo, então, três categorias de dispositivos para este fim, a saber:

1. Registrador Eletrônico de Ponto Convencional (REP-C);
2. Registrador Eletrônico de Ponto Alternativo (REP-A);
3. Registrador Eletrônico de Ponto Via Programa (REP-P).

Em seu artigo 76, a portaria define o REP-C:

Art. 76. O REP-C é o equipamento de automação monolítico, identificado pelo seu número de fabricação e cujo modelo possui certificado de conformidade especificado no art. 90, utilizado exclusivamente para o registro de jornada de trabalho e com capacidade para emitir documentos decorrentes da relação do trabalho e realizar controles de natureza fiscal trabalhista, referentes à entrada e à saída de empregados nos locais de trabalho.

Na sequencia, em seu artigo 77, é definido o REP-A:

Art. 77. O REP-A é o conjunto de equipamentos e programas de computador que tem sua utilização destinada ao registro da jornada de trabalho, autorizado por convenção ou acordo coletivo de trabalho.

E para terminar a classificação, em seu artigo 78 é definido o último tipo de registrador, o REP-P:

Art. 78. O REP-P é o programa (software) executado em servidor dedicado ou em ambiente de nuvem com certificado de registro nos termos do art. 91, utilizado exclusivamente para o registro de jornada e com capacidade para emitir documentos decorrentes da relação do trabalho e realizar controles de natureza fiscal trabalhista, referentes à entrada e à saída de empregados nos locais de trabalho.

O Anexo IX deste mesmo documento estabelece os requisitos técnicos aos quais o REP-P deve atender, entre as quais elenca em seu item 6 **armazenamento com redundância, alta disponibilidade e confiabilidade**, conceitos que estão intimamente ligados e até mais que isso, fazem parte da essência da tecnologia blockchain. As demais determinações são de ordem mais formal definindo as informações que devem ser registradas e o formato apresentado, como por exemplo, permitir a identificação do empregador e trabalhador, possuir relógio com sincronismo com a hora de Brasília, identificação da pessoa que faz o registro ou alterações no sistema, entre outras.

As regras estabelecidas acima são totalmente perfeitamente implementáveis na solução proposta, conforme será exposto adiante, além disso, as exigências de imutabilidade, identificação e segurança estabelecidos na portaria são, por essência, implementados na tecnologia blockchain, conforme, também, será exposto em momento oportuno no decorrer dessa obra.

Diante da importância do tema, é fundamental apresentar uma solução capaz de atender a esta demanda de mercado com critérios de segurança e integridade que impossibilitem a adulteração de dados para proteger a sociedade; e a tecnologia blockchain tem se mostrado, em muitos cenários, como solução viável e segura para repositório de dados de forma descentralizada, com persistência de histórico de operações e identificação de transação permitindo sua rastreabilidade.

Sendo o blockchain uma tecnologia de Livro Razão Distribuído materializado através de uma cadeia de blocos onde o bloco da transação atual tem a referência do bloco que fez a transação anterior e assim sucessivamente, uma transação só é validada se houver um histórico e um referência a um bloco anterior válido, e para que esta transação seja válida é preciso aprovação de mais de 50% dos integrantes da rede blockchain[[5]](#footnote-4), ou seja, ao descentralizarmos o repositório das informações e criarmos mecanismos de validação, dificultamos que uma pessoa sozinha faça modificações indevidas nos dados da cadeia[[6]](#footnote-5). Desta maneira, um registro de fim de jornada de trabalho, por exemplo, seguindo a ordem de precedência de blocos da cadeia, só pode ser gerado para um trabalhador que, minimamente, tenha sido registrado dentro do sistema do empregador e iniciado sua jornada de trabalho.

Através da implantação de uma solução utilizando a tecnologia blockchain, onde todos os dados do empregador e empregado bem como seus registros de ponto com início, fim e pausa da jornada, é possível acompanhar toda a cadeia.

Com esta abordagem, um agente fiscalizador qualquer pode consultar quantos empregados registrados há no sistema, os registros de um empregado específico, o histórico de todas as marcações de todos os empregados em uma data específica, em um período de datas ou até mesmo todo o ciclo de vida da empresa. Perceba que uma vez registrado na blockchain tais informações, elas ficam lá independentemente de mudança de fornecedor de software ou até mesmo a extinção da empresa, não dependendo de um banco de dados próprio da empresa ou funcionário para se ter o registro gravado.

Além do histórico idôneo de jornada de trabalho, outras ferramentas tecnológicas como os smart contracts[[7]](#footnote-6), que são contratos eletrônicos auto executáveis podem ser inseridos nesta cadeia, promovendo ações diversas como bloqueio do empregado em caso de abandono de emprego, notificações ao empregador em caso de falta, etc.

Desde o início das operações com blockchain, em especial com a mais famosa das criptomoedas, bitcoin[[8]](#footnote-7), muitas soluções tem sido propostas para aprimorar a segurança em operações e persistência de dados, e o setor público não fica de fora deste cenário onde temos propostas como a gravação de operações policiais e registro em blockchain[[9]](#footnote-8), combate à corrupção e lavagem de dinheiro[[10]](#footnote-9) além de diversas outras estratégias[[11]](#footnote-10) apresentadas por SILVA e MARQUES em seu trabalho de pesquisa sobre as propostas desta tecnologia para o setor público.

A proposta deste trabalho entra, então, em consonância com as mais recentes estratégias de uso desta tecnologia para solução de problemas conhecidos. O autor da obra reconhece o caractere ainda em evolução da tecnologia mencionada e que ela não é a resposta para todos os problemas e muito menos há ausência de limitações e cenários para implantação desta solução, e realmente há cenários em que uma solução com uma autoridade central é mais vantajosa do que outra que usa autoridade descentralizada, como a blockchain, onde o artigo[[12]](#footnote-11) de Sing Kuang Lo, Xiwei Xu, Yin Kia Chiam e Qinghua Lu em conferência sobre a engenharia de sistemas complexos, da IEEE, em 2017 mostra esta análise. A figura abaixo, retirada do artigo supra mencionado, mostra um fluxograma muito útil na tomada de decisão sobre a viabilidade de utilização de uma arquitetura de livro razão descentralizada, como a blockchain.

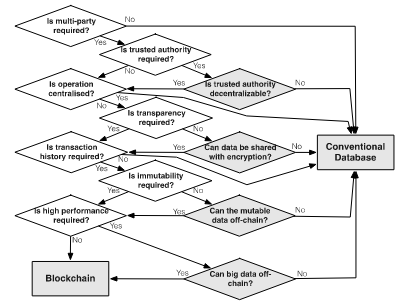
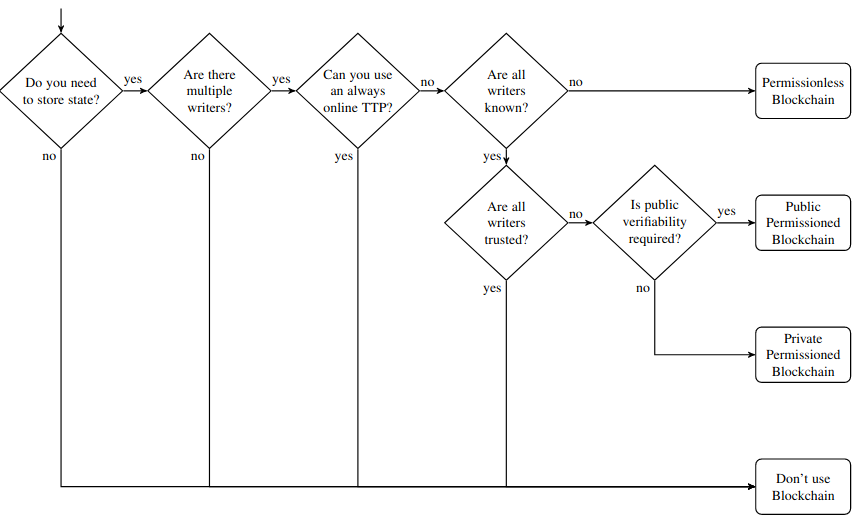


Figura 1 – Framework de avaliação para uma estrutura blockchain. [LO, XU, CHIAM, LU] 2017. P. 159

De igual forma, o trabalho[[13]](#footnote-12) apresentado por WÜST e GERVAIS mostra um fluxograma para tomada de decisão de um projeto envolvendo, ou não, a tecnologia blockchain. Em resumo, os autores propõem o uso apenas em situação em que não temos uma confiança em terceiros e estes são, também, os responsáveis pela governança de um sistema.

Figura 2 – Framework de avaliação para uma estrutura blockchain. [WÜST e GERVAIS] 2018. P. 3

# 4 OBJETIVO GERAL

Criar uma solução em blockchain para registrar os eventos marcação de ponto para os empregados de uma empresa, desde o seu registro até a emissão de relatórios com uso de livro razão descentralizado utilizando a plataforma **Polygon**[[14]](#footnote-13), solução de segunda camada da **Ethereum[[15]](#footnote-14)**,para desenvolver e hospedar esta solução com a implementação de smart contracts.

# 5 OBJETIVOS ESPECÍFICOS

Este projeto tem como objetivos de domínio:

1. Demonstrar as vantagens de um sistema utilizando uma estratégia de registro de livro razão descentralizado em face a uma aplicação tradicional centralizada.
2. Apresentar o funcionamento atual do sistema de controle de ponto tradicional e seus pontos vulneráveis.
3. Criar uma aplicação web para consultar e registrar informações referentes ao registro de ponto de funcionários de uma empresa.

E como objetivos técnicos:

1. Criar uma infraestrutura de blockchain utilizando a plataforma **Solidity** através do ambiente **HardHat**[[16]](#footnote-15).Criar uma Web API em **Solidity**[[17]](#footnote-16) para interagir com a plataforma **Polygon**.
2. Criar uma **Single Page Application** (SPA)[[18]](#footnote-17) utilizando a tecnologia **React**[[19]](#footnote-18) para apresentar funcionar como interação dos usuários com o sistema de livro razão descentralizado.
3. Criar uma solução em blockchain para registrar as marcações de ponto dos funcionários de uma empresa, desde o registro da empresa e funcionários, bem como a emissão de relatórios.

# 6 FUNDAMENTAÇÃO TEÓRICA

## 6.1 BLOCKCHAIN

O que é blockchain? Como surgiu? Por que é tão importante? Hoje milhares de projetos usam blockchain, alguns revolucionários e outros nem tanto, mas é essa estrutura de registros que permite a descentralização das informações e que vai continuar mudando a nossa vida nos próximos anos.

### 6.1.1 HISTÓRIA

Conforme menciona ESCOBAR[[20]](#footnote-19), em 1991, quase vinte anos antes do Bitcoin, STUART HABER e SCOTT STORNETTA, que trabalhavam na Xerox, criaram uma nova forma de trazer segurança às informações. Seu objetivo era criar um sistema de registros digitais que fosse imutável, inspirados pelo trabalho de CHAUM DAVID de 1979 que propunha um sistema de cofres criptográficos. Sua motivação vinha de um escândalo com um artigo de biologia que tinha sido fraudado na época. Nesta fraude, os artigo foi alterado utilizando uma tinta especial e a dupla pensou que se foi possível falsificar um artigo em papel, em mídia digital isso seria muito mais fácil, bastava apertar a tecla Delete. Para resolver isso, eles imaginaram blocos de informação atrelados uns aos outros de forma imutável, onde nada pode ser editado ou excluído.

Em 1992 eles incluíram criptografia nesse mecanismo de registros e até 2008 essa invenção não tinha nome e nenhum caso de uso real. Mas a tecnologia ganhou visibilidade quando foi publicado o *White Paper[[21]](#footnote-20)* de NAKAMOTO, SATOSHI, o grande teórico que influenciou a criação do projeto Bitcoin, que apareceu a palavra blockchain como o termo é reconhecido hoje. Ele se referia blocos de informação – Blocks - e dados em cadeia – Chain - usando funções de hash. De tanto que a palavra Block a palavra Chain apareceram no Paper, foi natural unir essas duas palavras e chamar esse sistema de Blockhcain.

### 6.1.2 FUNCIONAMENTO

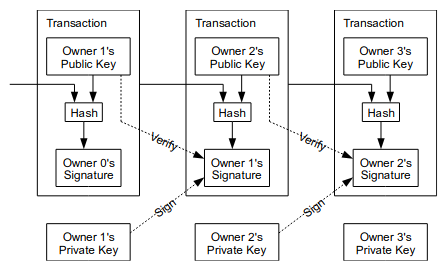
Em tradução literal, **blockchain** significa corrente de blocos. São blocos de informações atrelados uns aos outros. É como tecido digital, você não consegue puxar o bloco do meio da cadeia sem afetar os blocos seguintes, assim como quando você puxa o fio de um tecido de uma roupa você altera toda a costura. Blockchain é como uma costura digital e se um bloco for alterado, todo mundo vai perceber que tem algo errado naquele ponto da cadeia. Depois de registrado na blockchain, essas informações ficam gravadas nessa cadeia de blocos e é por isso que o blockchain também serve como uma linha do tempo onde os fatos não podem ser modificados, por isso é uma rede imutável e irreversível.

Para modificar os blocos de uma cadeia, tomando como base o exemplo do Bitcoin, a única chance que você tem é naqueles primeiros momentos em que os blocos recentes foram criados. É possível tentar reverter esses blocos de informação, mas isso precisaria de poder computacional absurdo para invadir pelo menos 51% da rede e uma janela de tempo muito pequeno, em torno de 10 a 30 minutos, isso gastaria também Bilhões de Dólares e ainda com a chance de não conseguir fazer um ataque furtivo, no máximo reverter, talvez, uma transação da *main pool*, e não o bloco inteiro.

Esse mecanismo de proteção contra modificações é implantado com o uso de um protocolo de consenso fazendo com que a cada transação realizada os outros nós da rede avaliem o bloco e pelo consenso da maioria, 50% + 1, reconheçam-no como válido. Caso houver divergência, esse “desempate” é resolvido na transação seguinte, fazendo com que a cadeia mais longa prevaleça sobre a cadeia de blocos menor, por isso que depois de 6 blocos criados, ou em média uma hora, se diz que as informações se tornam imutáveis. Por que a cada bloco criado é maior a dificuldade e o custo para reverter informações e nesse sentido a blockchain do Bitcoin é a mais longa e a que exige maior poder computacional para ser invadida para reverter blocos, sendo considerada por muitos rede mais segura até hoje.

Essas informações são inseridas na rede por computadores superpotentes chamados de mineradores. Eles fazem cálculos matemáticos para resolver uma espécie de quebra-cabeça criptográfico e quando encontram a resposta certa a rede valida resposta, registra o bloco na blockchain e os mineradores recebem Bitcoin como recompensa por terem emprestado seu poder computacional para rodar a rede. Esse mecanismo se chama prova de trabalho ou **Proof of Work** (PoW) na rede Bitcoin e destrava as novas moedas do protocolo, conforme os blocos de informação são criados e funciona como um jogo, onde o computador que resolver o problema criptográfico primeiro recebe os bitcoins da rodada e os mineradores ficam competindo entre si o tempo todo para ver quem chega primeiro no cálculo e recebe os bitcoins do próximo bloco. Só vão ser criados 21 milhões de bitcoins até o ano de 2140, e isso gera escassez e valor à moeda.

Cada bloco contém informações sobre as transações financeiras feitas na rede tipo, endereço A enviou 2 bitcoins endereço B, e também tem uma marcação de tempo, um carimbo de data e hora, chamado **TimeStamp**. Todos esses dados formam o conteúdo de cada bloco e são misturados de forma aleatória e transformados em um hash. Hash é um código criptografado que identifica tudo que tá dentro daquele bloco de informação e é a partir do Hash que a mágica funciona. Depois de criado o Hash 1 do bloco 1 ele vai ser inserido junto com conteúdo do próximo bloco, o bloco 2, que também será misturado de forma aleatória e vai formar justamente o Hash 2, por isso o Hash 2 resume todo o conteúdo do seu Bloco e também do bloco anterior porque o Hash 1 foi inserido dentro do bloco 2 e assim sucessivamente o Hash 3 vai ser o resumo criptográfico do bloco 3 que tem também o seu conteúdo o Hash do bloco 2 anterior.

Figura 3 – Cadeia de blocos [[22]](#footnote-21)

A grande percepção trazida pelo artigo de NAKAMOTO foi a resolução do problema do gasto duplo, impedindo que um mesmo bem pudesse ser transacionado mais de uma vez, gerando escassez e valor e a mudança do centro gravitacional da confiança, onde antes ficava na mão de um terceiro confiável que validasse as operação de troca/compra e passando agora para o indivíduo, que através da criptografia, protocolo de consenso e descentralização da rede tem a autonomia.

### 6.1.3 CRIPTOGRAFIA E VALIDAÇÃO

Foi mencionado anteriormente que o blockchain utiliza a criptografia como elemento de segurança em sua estrutura. Criptografia nada mais é que uma forma de se ocultar a informação de pessoas não autorizadas. Mas além de ocultar estas informações, no caso, o conteúdo do bloco, é necessário que a rede do blockchain identifique o proprietário, o responsável por aquela transação, e através do uso da criptografia assimétrica conseguimos atingir esses dois objetivos: ocultar os dados de pessoas não autorizadas e garantir a autenticidade.

Podemos dividir as técnicas de criptografia em dois grandes grupos, os que utilizam **chaves simétricas** e os que utilizam **chaves assimétricas**. Resumidamente, os métodos que fazem uso de chaves simétricas utilizam a mesma chave para ocultar (criptografar) e revelar (decriptografar) a mensagem original, fazendo uso de recursos de substituição e permutação. Nesta abordagem, é preciso enviar a senha para o destinatário de forma segura, para que somente a pessoa alvo tenha acesso à chave e consiga revelar a mensagem oculta.

Inicialmente, as técnicas de chaves simétricas faziam uso de algoritmos que podiam ser calculados a mão, passando então por uma grande mudança ao fazer uso do rotor eletromecânico, produzindo, então, algoritmos extremamente complexos, e com o advento dos computadores esta complexidade aumentou ainda mais, mas ainda permaneciam os problemas de privacidade e autenticação que as chaves simétricas não conseguiam atender eficazmente.

Para atender estas lacunas, as técnicas de **chaves assimétricas** foram introduzidas. Basicamente, esta técnica faz uso de duas chaves (diferente da chave simétrica, que usa apenas uma), uma **chave pública** e a outra **chave privada**, em que a pública é divulgada através de um meio público e a privada fica de posse de seu proprietário. Somente o conjunto desse par de chaves é capaz de ocultar e revelar adequadamente a mensagem original.

Importante ressaltar que não se pode declarar que as técnicas de chaves assimétricas são superiores às de chaves simétricas. O verdadeiro diferencial entre uma técnica e outra é o tamanho e complexidade da chave utilizada. Também não se pode falar que uma técnica substitui a outra, pois para cada cenário há uma técnica que melhor se adapta e ainda há a possibilidade de criar soluções que fazem uso de técnicas de chaves simétricas e chaves assimétricas trabalhando juntas.

O trabalho de STALLINGS[[23]](#footnote-22) mostra o funcionamento da criptografia de chave pública. Nesta abordagem, se pretende atacar os dois problemas mais difíceis associados à criptografia simétrica, a situação de dois participantes terem que compartilhar a mesma chave entre eles e o problema da autenticação de mensagens.

O compartilhamento da mesma chave entre dois sujeitos compromete a essência da criptografia pois abre a possibilidade da quebra de sigilo da comunicação por um terceiro que por ações como suborno, ameaça ou uma invasão bem sucedida da comunicação conseguiria acessar um conteúdo de forma indevida originariamente destinada apenas aos integrantes daquele canal de comunicação. No mesmo sentido dos problemas envolvendo a comunicação com chaves simétricas, se na troca de documentos físicos de papel há situações onde é importante ter a certeza de quem enviou o documento como com a presença de uma assinatura manuscrita, este mesmo comportamento pode ser transferido para os cenários de troca de documentos de forma digital, onde a presença de uma assinatura identificando de forma única um participante do canal de comunicação é capaz de ser o diferencial entre uma transação válida ou não.

Exatamente neste sentido é que as chaves assimétricas conseguem trazer robustez a uma comunicação criptografada, além de prover mecanismos de segurança diferenciados e a autenticidade. O quadro abaixo mostra, em linhas gerais, as diferenças entre o sistema de criptografia convencional e criptografia de chave pública.



Figura 4 - STALLINGS, William. Criptografia e segurança de redes. 184 p.

Um sistema criptográfico de chave pública possui 5 componentes básicos:

* **Texto claro:** mensagem ou dados que são alimentados no algoritmo como entrada
* **Algoritmo de Criptografia**: processamento matemático e computacional que transforma o texto claro
* **Chaves pública e privada**: par de chaves que são usadas para criptografia e descriptografia.
* **Texto cifrado**: mensagem codificada produzida como saída do resultado do processamento entre texto claro, algoritmo e chave (pública ou privada)
* **Algoritmo de descriptografia**: utilizando o texto cifrado e a chave, entrega como resultado o texto claro

As etapas essenciais deste processo são:

* Cada usuário gera um par de chaves a ser utilizado para criptografia e descriptografia das mensagens
* Cada usuário coloca uma das duas chaves em um registro ou arquivo público acessível
* Se Carlos deseja enviar mensagem confidencial para Tales, Carlos criptografa usando a chave pública de Tales
* Quando Tales recebe a mensagem, ele a descriptografa usando sua chave privada.

As duas imagens abaixo mostram a dinâmica do uso da criptografia de chave pública para fins de sigilo e autenticação. Na primeira imagem, há uma mensagem que é criptografada utilizando a chave pública de Alice, assim, se a chave privada de Alice for mantida em sigilo, apenas Alice conseguirá fazer a descriptografia da mensagem e ver o texto claro através do uso de sua chave privada. Com esta abordagem se atinge o sigilo da mensagem, garantindo que apenas o autor e o receptor devido tenham conhecimento do conteúdo

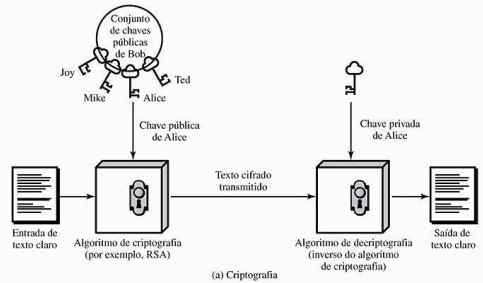


Figura 5 - STALLINGS, William. Criptografia e segurança de redes. 184 p.

Na imagem seguinte vemos o processo inverso, agora utilizando a chave privada de Bob. A partir de uma mensagem de texto claro é feita a criptografia utilizando a chave privada de Bom. Após o envio da mensagem, aqueles que tiverem a chave pública de Bob poderão ver o conteúdo da mensagem. Com esta abordagem se atinge o objetivo da autenticidade, pois independente de quem acesse a mensagem, como tem a chave pública de Bob, automaticamente, tem a certeza de que a mensagem teve Bob como remetente.

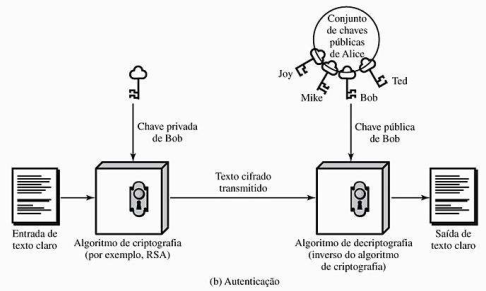


Figura 6 - STALLINGS, William. Criptografia e segurança de redes. 184 p.

Vale ressaltar que várias combinações podem ser utilizadas dentro deste cenário, inclusive com chaves pública e privada para criptografar e descriptografar, atingindo, então, os alvos de sigilo e autenticação. A imagem abaixo traz exatamente este cenário onde uma mensagem original é criptografada com a chave privada de **A**, em seguida passa por uma nova criptografia com a chave pública de **B** e é transmitida até seu destino. No destino, **B** faz a primeira descriptografia com sua chave privada em seguida uma nova descriptografia com a chave pública de **A** entrega novamente o texto claro.

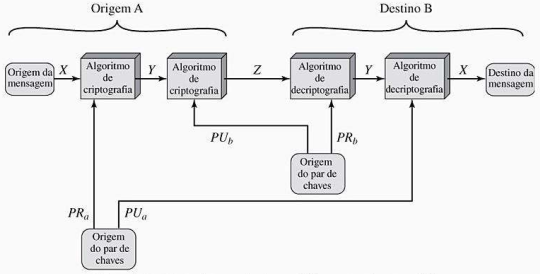
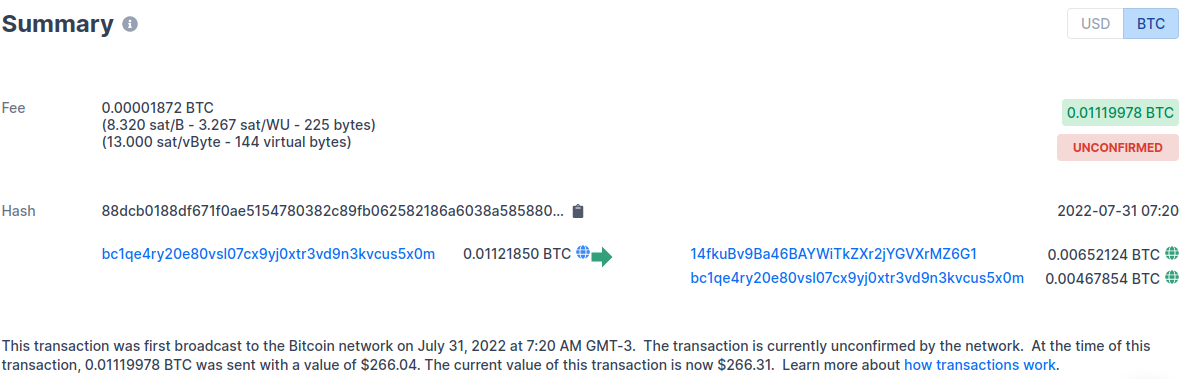


Figura 7 - STALLINGS, William. Criptografia e segurança de redes. 186 p.

Um outro cenário possível, ainda, é a utilização de uma combinação de sistemas de criptografia de chave simétrica em conjunto com outro de chave assimétrica. Enfim, a combinação entre estes elementos são inúmeras e a melhor estratégia deve estar adequada a cada projeto em particular.

Isto posto, tipicamente, uma rede blockchain registra a autenticidade das transações através dos endereços delas e estes endereços nada mais são do que a chave pública de cada uma das pessoas que interagiram na rede, deixando ali sua identidade, sua impressão digital. O site Blockchain.com mostra as transações realizadas na rede do blockchain. Nele é possível verificar os detalhes de uma transação na rede do Bitcoin, com seus registros de endereço de origem e destino, Hash, hora e valor. A imagem abaixo mostra esse detalhe.

Figura 8 – Transação na rede Bitcoin. [[24]](#footnote-23)



Na figura acima temos algumas informações relevantes:

* Em verde – o Hash da transação ocorrida;
* Em laranja – a data da transação;
* Em roxo – o valor da transação;
* Em vermelho – o endereço de origem do bitcoin;
* Em azul – o endereço de destino do bitcoin.

Os endereços de origem e destino da transação não identificam propriamente um proprietário mas sim uma carteira, e esta carteira possui um proprietário. Desta maneira é que são rastreadas todas as transações do blockchain, identificando seus endereços de origem e destino.

Importante ressaltar que os registros na rede blockchain podem ser verificados por qualquer pessoa garantindo a publicidade e verificação das transações, mas a identidade, nome, CPF, RG, sexo, idade, CNPJ ou qualquer outro identificador civil e pessoal da pessoa que fez a transação ficam resguardados, garantindo a privacidade dos utilizadores da rede. É possível identificar, sim, a carteira que movimentou a transação, mas não é possível identificar pessoalmente o proprietário da carteira. Qualquer pessoa com o endereço dessa transação ([https://www.blockchain.com/btc/tx/88dcb0188df671f0ae5154780382c89fb062582186a6038a585880945f3cbbc0](https://www.blockchain.com/btc/tx/88dcb0188df671f0ae5154780382c89fb062582186a6038a585880945f3cbbc0" \o "https://www.blockchain.com/btc/tx/88dcb0188df671f0ae5154780382c89fb062582186a6038a585880945f3cbbc0)) ocorrida na rede poderá confirmar tais dados assim como foi apresentado na figura acima.

### 6.1.4 MECANISMO DE HASH

Conforme menciona STALLINGS[[25]](#footnote-24), um mecanismo de autenticação de mensagens atua em dois níveis, no primeiro, ele funciona como um autenticador, ou seja, um valor que identifica tal mensagem como autêntica; já no segundo nível, atua como um mecanismo de não repúdio, fazendo conhecer o verdadeiro autor da mensagem. Neste sentido, uma **Função de Hash** relaciona uma mensagem de qualquer tamanho a um valor de tamanho fixo que serve como autenticador.

Conceitualmente, uma função de hash pode ser matematicamente representado pela expressão:

***h = H(M)***

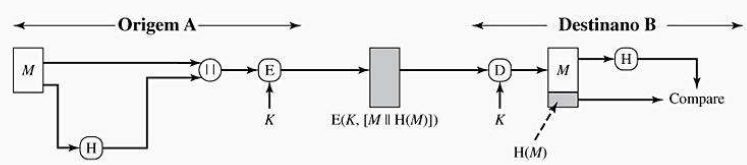
*Sendo:*

***h →*** *valor de hash resultante*

***M →*** *mensagem de comprimento variável*

***H(M) →*** *função de hash de comprimento fixo*

Nestes termos, um código Hash não usa uma chave, sendo uma função apenas de mensagem de entrada. Aliada à criptografia, uma função hash pode atuar como solução para critérios de sigilo, autenticidade da mensagem, não repúdio e controle de erros. A figura abaixo ilustra este exemplo de aplicação, onde **A** compartilha sua chave pública com **B**, que ao receber a mensagem precisa confirmar se a mesma não foi alterada durante o envio e se **A** é realmente o emissor de tal mensagem.

 Figura 9 – STALLINGS, William. Criptografia e segurança de redes. 234 p.

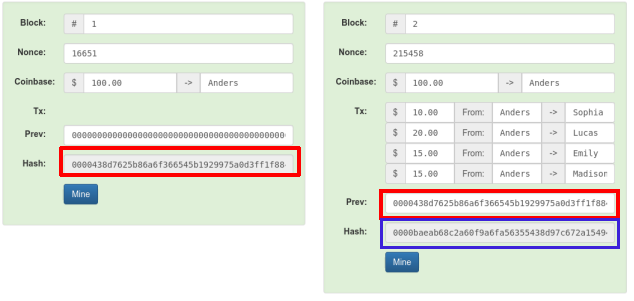
Ainda de acordo com STALLINGS, uma função de Hash **H** precisa ter as seguintes propriedades:

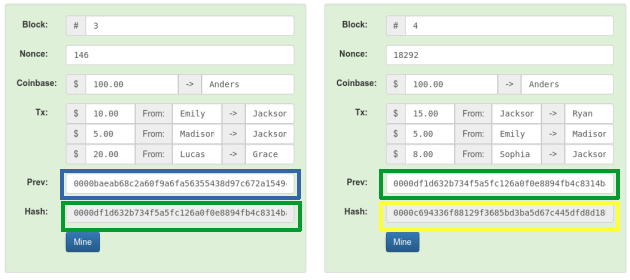
1. **H** pode ser aplicado a uma entrada de qualquer tamanho;
2. **H** produz uma saída de tamanho fixo;
3. **H(M)** é relativamente fácil de calcular para qualquer **M**;
4. Para um valor de **h** dado, deve ser computacionalmente inviável encontrar **M**, atendendo o princípio da **resistência à primeira inversão** ou p**ropriedade unidirecional**;
5. Para um bloco **M**, deve ser computacionalmente inviável encontrar **M’≠ M** tal que **H(M’) = H(M)**, atendendo à **resistência à segunda inversão** ou r**esistência fraca a colisões**;
6. Deve ser computacionalmente inviável encontrar o par ordenado **(M, M’)** tal que **H(M) = H(M’)**, atendendo à **resistência a colisões** ou **resistência forte a colisões**.

No caso específico do Bitcoin, a função Hash utilizada é a **SHA-256**, que independentemente do tamanho da entrada, entrega como resultado de **256 bits** que contém **64 símbolos** alfanuméricos. O código abaixo representa um exemplo de aplicação de uma função Hash que para a mensagem **Blockchain is innovative** encontra como resultado o seguinte Hash: **0a94aa3f0c84ccc061bef0eb59903903732800eed9828c2f5c04c19ff3f329e9.**

Figura 10 – Função Hash 256. [https://andersbrownworth.com/blockchain/hash](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

Para ilustrar esse mecanismo de funcionamento do Hash, o site [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase) possui um simulador onde mostra uma cadeia com 5 blocos gerenciada por uma rede com 3 nós. Em cada um desses nós há uma cópia da cadeia de blocos, de forma que se algum fraudador tentar mudar a cadeia de blocos administrada por seu nó, será desmascarado pela simples comparação de sua cadeia com as demais cadeias de blocos da rede. Vamos considerar a cadeia de 5 blocos representada pelos blocos abaixo:





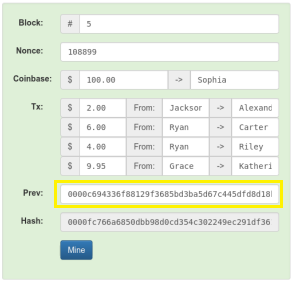


Figura 11 – Cadeia de blocos do primeiro nó. [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

A cada bloco notamos a informação do hash do bloco atual e a informação do hash do bloco anterior, fazendo a referência direta entre blocos, sendo o primeiro bloco um bloco especial, reconhecido como bloco gênese, que no campo de hash anterior tem o valor preenchido com zeros, e para cada bloco, para que seu hash seja considerado válido é preciso realizar as operações matemáticas necessárias para que seu hash seja iniciado com 4 zeros (**0000**).

Cada um desses blocos tem um conjunto de transações, e o hash é o resultado matemático do fruto dessas operações e do bloco anterior. Como resultado, este mesmo conjunto de blocos é registrado nos outros três nós da cadeia. Note que os hash dos blocos são iguais em qualquer um dos três nós, evidenciando o registro descentralizado dos dados em qualquer um dos nós da rede.

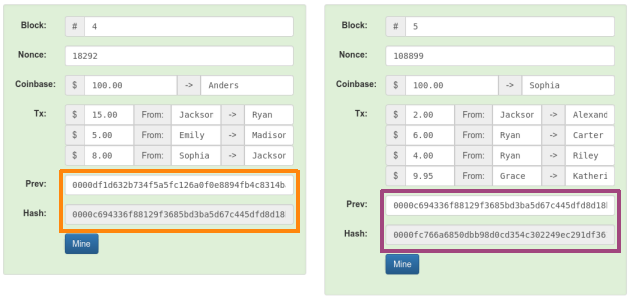


Figura 12 – Blocos 4 e 5 do segundo nó. [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

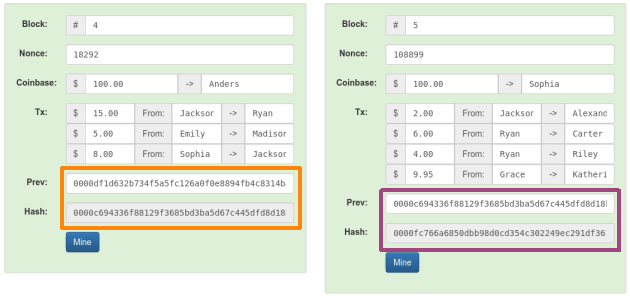


Figura 13 – Blocos 4 e 5 do terceiro nó. [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

Se em nossa simulação de rede blockchain modificarmos qualquer dado de um bloco, isso comprometerá toda a cadeia de blocos na sequência, evidenciando a manipulação. Para exemplificar, vamos alterar uma transação do bloco3 do segundo nó. Vamos modificar a transação realizada entre Madison e Jackson de **5,00** para **5,01.** Note que o seu hash será modificado, devido à mudança de um dado do bloco. Uma mudança pequena, é verdade, mas suficiente para denunciar que algo foi modificado.

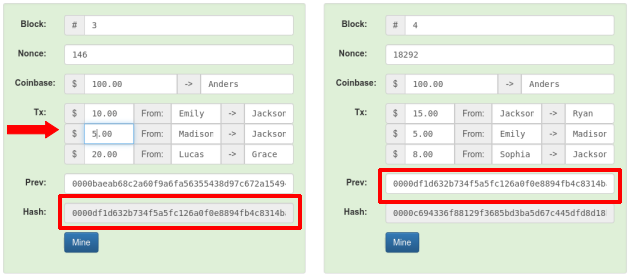


Figura 14 – Blocos 3 e 4 originais do segundo nó. https://andersbrownworth.com/blockchain/coinbase

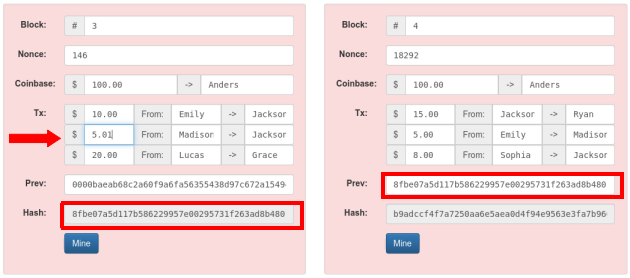


Figura 15 – Blocos 3 e 4 modificados do segundo nó. [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

Perceba que mesmo com a pequena alteração aplicada de **0,01,** o bloco já teve o seu hash modificado, onde antes da alteração era **0000df1d...**, passando para **8fbe...**. Essa modificação altera todos os blocos seguintes da cadeia, fazendo com que seja identificada, de imediato, uma mudança.



Figura 16 – Blocos 4 e 5 afetados pela modificação do bloco 3 do segundo nó. [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

Ao compararmos estes mesmos blocos nos outros nós da rede percebemos que eles não foram alterados.

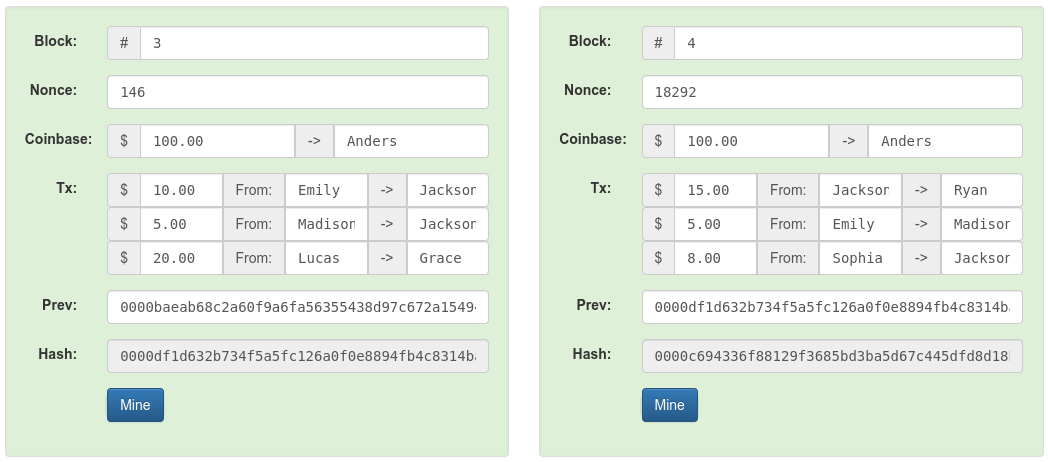


Figura 17 – Blocos 3 e 4 do primeiro nó. [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

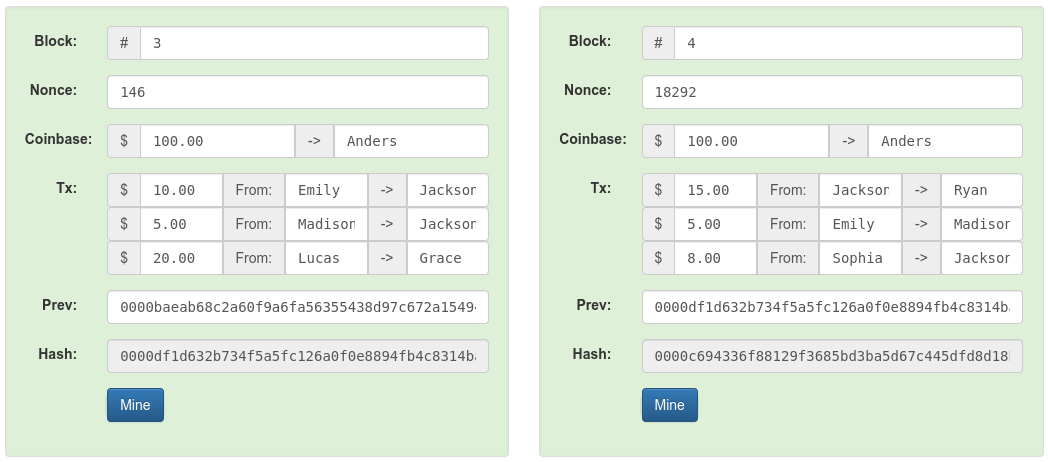


Figura 18 – Blocos 3 e 4 do terceiro nó. [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

Ainda quanto ao segundo nó, mais especificamente o bloco alterado, mesmo após minerado, com o Hash de seu bloco corrigido, os blocos 4 e 5 desta cadeia continuarão denunciando a modificação.

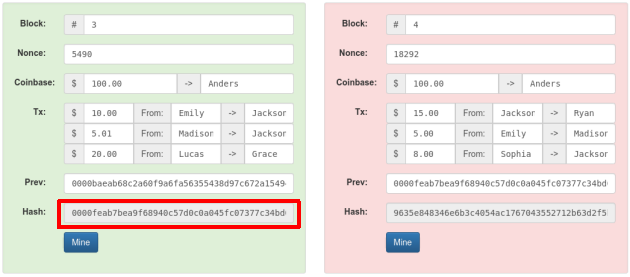


Figura 19 – Blocos 3 e 4 do segundo nó após a mineração do bloco 3. [https://andersbrownworth.com/blockchain/coinbase](https://andersbrownworth.com/blockchain/coinbase" \o "https://andersbrownworth.com/blockchain/coinbase)

Note que agora, o bloco se inicia com **0000** sendo esta sequência de 4 zeros o conceito validador para os blocos desta cadeia, diferente do valor de hash que tinha antes (**8fbe...**), mas como o bloco 4 tem em seu registro o hash do bloco 3 anterior à manipulação realizada, apesar de o bloco 3 estar validado nesta cadeia os blocos 4 e 5 denunciam que houve alguma alteração em um bloco (no bloco 3).

Se o fraudador da rede em nosso exemplo quisesse ter sucesso efetivo em seu golpe ele teria que gastar processamento para minerar os blocos 4 e 5 da rede do segundo nó e ainda os blocos 3, 4 e 5 da cadeia do primeiro ou terceiro nó para que sua manipulação de dados atinja mais de 50% da rede, e tudo isso num curto espaço de tempo, pois se os nós 1 e 2 realizassem uma nova transação, todo o seu trabalho seria em vão, pois precisaria modificar o novo bloco inserido na sua rede e mais os outros blocos dos nós adjacentes. Esse mecanismo faz com que os mineradores sejam muito mais estimulados a manterem a rede íntegra do que a corrompê-la, pois é perfeitamente rastreável o ponto onde a fraude, ou pelo menos, tentativa dela ocorreu.

### 6.1.5 ÁRVORE DE MERKLE

Até aqui creio que tenha ficado claro que o bloco guarda um conjunto de transações e estas transações vão se acumulando no tempo e os blocos se relacionam entre si tendo um único bloco pai, exceto no caso do primeiro bloco da cadeia que é conhecido como bloco gênese , que por ser o primeiro, não possui seu bloco pai, tendo no campo da informação do Hash do bloco anterior um conjunto com 64 zeros.[[26]](#footnote-25)

A função Hash é capaz de grandes trunfos e na rede blockchain ela é muito bem utilizada para fazer a checagem de grande quantidade de dados organizados. A estratégia utilizada faz uso de uma estrutura chamada **Merkle Tree** (Árvore de Merkle) relacionando seus identificadores em uma estrutura única em forma de dados (KUNTZ).[[27]](#footnote-26)

Proposta por Ralph Merkle em 1979, essa estrutura agrupa nós de forma hierárquica, de forma que os identificadores de cada nó é formado a partir de seus nós filhos de forma recursiva até chegar ao identificador único dos nós, identificado como **Merkle Root** (Raiz de Merkle).

A Árvore de Merkle acaba trazendo algumas vantagens, entre elas, num cenário onde há grande número de transações, não se faz necessário enviar o Hash de cada uma das transações para que uma transação seja validada, economizando volume de dados trafegados entre os nós da rede.

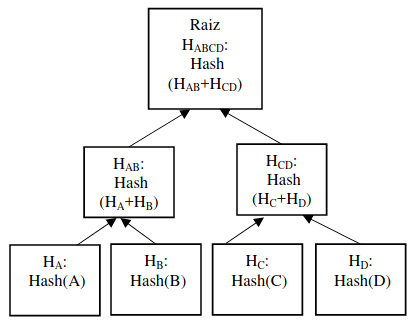


Figura 20 – Árvore de Merkle para 4 transações A, B, C e D.[[28]](#footnote-27)

Na figura acima é mostrada a árvore de Merkle para um conjunto de 4 transações, **A**, **B**, **C** e **D**. Vamos adiantar aqui um conceito da rede descentralizada que é de nós fracos e nós fortes. Em linhas gerais, podemos entender que os nós fracos guardam apenas a raiz de Merkle enquanto os nós fortes guardam todas as transações. Isto posto, se um nó fraco precisa confirmar com o nó forte se a transação **A** está naquele nó, o nó forte não precisa transmitir os Hashs das 4 transações, bastando apenas transmitir 2 transações, sendo o resultado de

**n log2**

**com n → número de transações do bloco**



Figura 21 – Árvore de Merkle para 4 transações A, B, C e D.

Da mesma forma que a figura 20, a figura 21 também traz a Árvore de Merkle para 4 transações, mas agora com os detalhes de cor para poder entender melhor a dinâmica de funcionamento. Os quadrados em cinza são as transações, em laranja o Hash de cada uma delas, em vermelho o agrupamento e em verde a Raiz de Merkle, sendo esta raiz o resultado da operação de Hashs sucessivos para cada um dos elementos antecessores.

Continuando nossa busca de confirmação da presença da transação A em um nó forte, se o nó forte enviar o Hash de **B** e o Hash **H(CD)**, como o nó fraco tem em seu escopo a transação **A**, logo ele conseguirá chegar ao Hash **H(AB)** que combinado ao Hash **H(CD)** enviado, conseguirá montar sua raiz e confirmar a presença da transação **A**.



Figura 22 – Envio de 2 Hashs para 4 transações.

Se fossem 8 transações, não seria necessário enviar 7 Hashs para essa confirmação, bastaria enviar 3 Hashs (**8log2**) para esta confirmação.

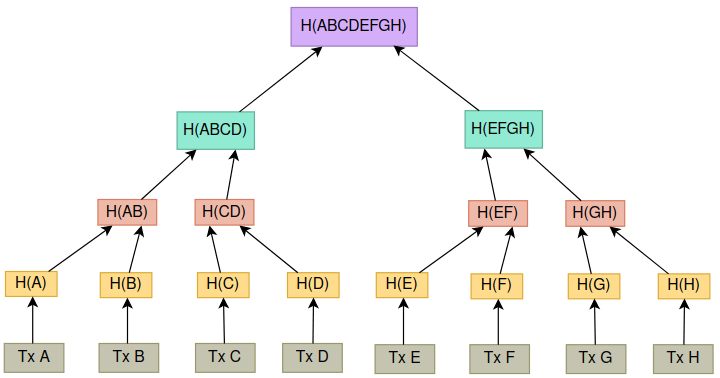


Figura 23 – Envio de 3 Hashs para 8 transações.

Notamos aqui, que de forma bem inteligente, com o envio dos Hashs **H(B)**, **H(CD)** e **H(EFGH)** é perfeitamente possível remontar a Raiz de Merkle para validar um conjunto de transações, economizando tráfego na rede. Se fossem 100 transações, da mesma forma, seguindo a fórmula matemática de **100 log2**, chegaríamos ao resultado de 7 Hashs enviados, num universo de 100, para remontar a Raiz de Merkle. Muito inteligente, não?!!!

### 6.1.6 REDE DESCENTRALIZADA

O blockchain também pode ser considerado um sistema de registros distribuídos funcionando em uma rede ponto-a-ponto (Peer-to-Peer Network). Todos os computadores que ajudam a processar a rede possuem cópias de todos os blocos de informação que já foram criados e qualquer informação alterada, seja uma vírgula, um espaço, um acento, um valor por menor que seja, qualquer coisa, vai mudar o hash do bloco e não vai bater com toda a cadeia de Hashs já consolidada. A rede não vai aceitar uma mudança porque não fecha com a cópia do blockchain que todos os nós tem salvo no seus computadores. É por isso que o bloco é um mecanismo extremamente resistente e inteligente.

É retirada a dependência de um centralizador para armazenar informação. Os dados ficam distribuídos de forma descentralizada com várias cópias. Para uma blockchain ser destruída seria necessário destruir todas as cópias que existem pois, se uma cópia ainda estiver disponível, a rede vai sobreviver a partir dessa única cópia.

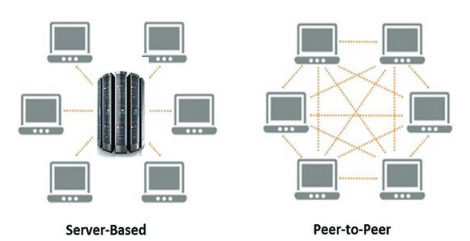


Figura 24 – Diferença entre rede descentralizada e rede centralizada.[[29]](#footnote-28)

Esse sistema garante que nenhum dado seja perdido. Todos podem verificar a veracidade dos registros. Tudo é feito em tempo real e de forma sincronizada e transparente. Além disso, qualquer pessoa pode auditar a rede, o que aumenta também a segurança, eficiência, confiança nos registros e elimina a necessidade de terceiros ou intermediários em uma única fonte de informação e confiança.

Uma rede ponto-a-ponto é formada por **NODES** (NÓS) e esta é uma peça fundamental neste grande quebra cabeça. O nó na rede blockchain é o ponto onde há uma cópia dos dados da rede blockchain. Muito mais que um repositório de dados, o nó exerce papéis podendo acumular funções ou se especializar em uma delas.

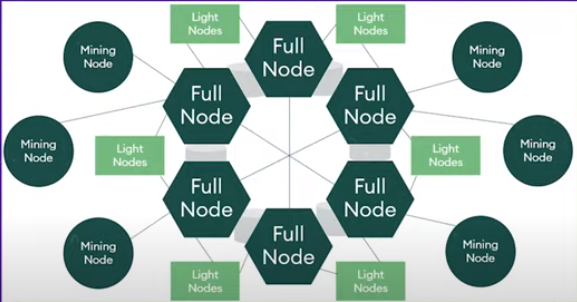


Figura 25 – Tipos de nó na rede blockchain[[30]](#footnote-29)

Entre os diferentes tipos de nós, alguns se destacam, entre eles:

1. Full Nodes (Nós Completos) – Contém todo o histórico e informações relacionadas a cada bloco, desde a primeira transação na rede. Uma de suas características mais distintivas é a verificação da autenticidade da assinatura em cada transação do bloco.
2. Lightweight Nodes (Nós Fracos) – Conhecido também como verificação de pagamento simples, ao invés de armazenar informações completas, os nós fracos contém informações relacionadas a um bloco anterior específico ao qual está conectado. Muitas vezes eles utilizam os nós completos par acessar a rede para verificar transações e recuperar informações. Eles se ligam a nós completos funcionando como pontes.
3. Mining Nodes (Nós de mineração) - Estes nós são responsáveis pela tarefa de minerar. Como resultado de seu trabalho, são gerados novos blocos.

O grande trunfo do blockchain é eliminar a necessidade de intermediários, e uma rede que descentraliza confiança, só poderia ter o seu primeiro caso de uso relacionado a dinheiro. O sistema financeiro é onde a gente mais depende de intermediários, tanto para criar moedas e as políticas monetárias com os bancos centrais, quanto para transmitir a moeda para a população, através dos bancos comerciais, só que como bem disse NAKAMOTO, a nossa história está repleta de violações essa confiança, seja pelos bancos centrais manipulando as políticas monetárias inflacionando as próprias moedas, seja via bancos comerciais criando mecanismos que dificultam o acesso das pessoas ao seu próprio dinheiro. Você já tentou sacar valores altos da sua conta bancária e precisou passar pelo gerente, né? Isso mostra o quanto a gente não é livre para movimentar o próprio dinheiro, precisa sempre pedir permissão e fica dependente desses intermediários liberarem ou não. O valor do Bitcoin, talvez o case de maior sucesso envolvendo a tecnologia blockchain, está na descentralização e na gama de tecnologias utilizadas como blockchain, Proof of Work e criptografia que tornam a rede segura.

Se não houver uma moeda, se resolver usar só blockchain, e utilizar uma autoridade Central decidindo quais são as ações ou quais blocos são válidos ou não, lá se vão todas as propriedades boas da descentralização.

De forma geral, uma rede blockchain pública precisa respeitar 5 princípios, todos alinhados com a Web 3.0:

1- **neutralidade** - não importa quem você é, onde você mora, qual a sua aparência, qual o seu partido político, nada sobre você vai te impedir de usar rede, a sua transação vai ser propagada independentemente de qualquer característica sua, o que não acontece no sistema financeiro tradicional, se você não fornecer seu endereço, um CPF válido, você não consegue ter uma conta bancária, por exemplo, você acaba sendo excluído do sistema.

2- **não territorialidade** - assim como a internet não é de nenhum país, está em todos os lugares, a rede não fica presa a nenhum território, a nenhum país, não pertence a nenhuma nação, é de todos os países e ao mesmo tempo não possui um órgão centralizador específico.

3- **abertura** - qualquer pessoa pode acessar, é só baixar um aplicativo ou qualquer outro meio tecnológico adequado e se conectar na rede.

4 – **resistência à censura** - não tem como parar a rede porquê alguém ficou incomodado com seu conteúdo ou porquê tem algo ali que a incomoda, é uma forma de expressão, um direito humano.

5- **publicidade** – os dados estão disponíveis para todo mundo verificar, isso significa que ninguém pode trapacear ou falsificar estas informações.

### 6.1.7 PROTOCOLO DE CONSENSO

Sendo uma rede onde os dados são armazenados em diversos nós, garantir que a rede blockchain mantenha a consistência dos dados sem a utilização de uma autoridade central, o que comprometeria um de seus pilares, é um desafio.[[31]](#footnote-30)

Neste contexto, os protocolos de consenso surgem como uma ferramenta de controle da blockchain, definindo qual o comportamento da cadeia no processo de adição de um novo bloco, seja validando e distribuindo este bloco entre os nós da cadeia, seja rejeitando inserções de blocos indevidas, protegendo a rede contra ataques[[32]](#footnote-31).

As propostas para atender a esta necessidade são várias, dependendo da proposta da rede, alvos técnicos e filosofia adotada. Seguindo referências de KUNTZ e ZHENG e autores, apresento abaixo os principais protocolos de consenso utilizados na atualidade.

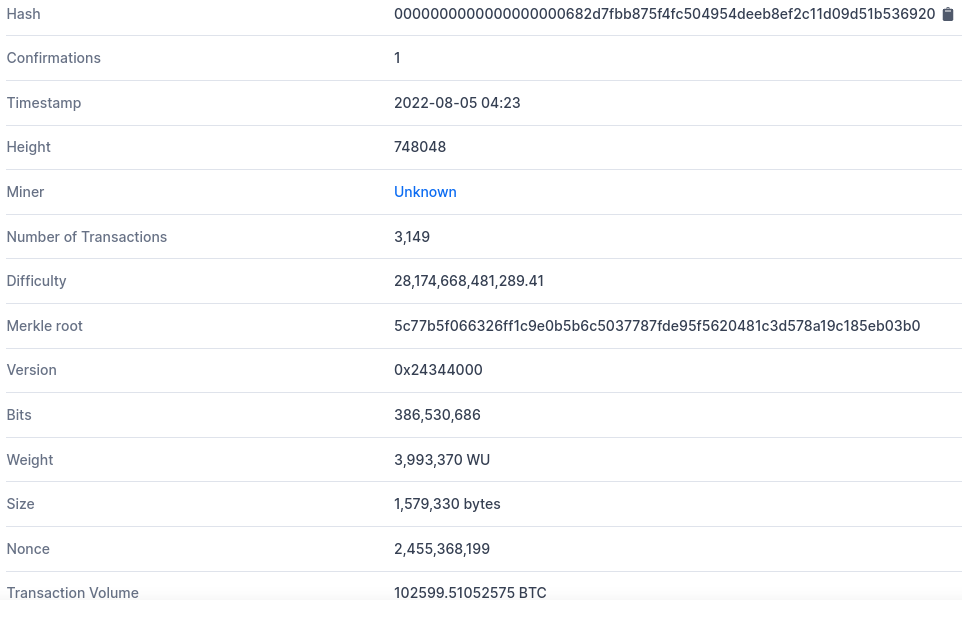
#### 6.1.7.1 PROOF OF WORK (PoW) – PROVA DE TRABALHO

Utilizando o mecanismo de mineração, os nós da rede competem entre si para resolver um desafio matemático para decidir qual vai ser o nó vencedor para incluir um novo bloco na rede. Este é o mecanismo utilizado na blockchain do Bitcoin.

O mecanismo chave no bloco é o seu Hash, e o Hash de um bloco precisa atender certas características para ser reconhecido como válido e por consequência, um bloco válido. Duas características são principais no trabalho de mineração: um processo matemático factível, mas de difícil realização e de fácil verificação. Na busca de um Hash criptográfico válido, o minerador deve encontrar um número único, um valor numérico de 4 bytes unsigned, reconhecido como **nonce** (number once) que aliado às transações do bloco (**raiz de Merkle**) e o **TimeStamp** produza um Hash que atende as condições especificadas, denominada **target**, alvo.

Este trabalho torna-se mais difícil ainda visto que o TimeStamp do bloco se modifica a cada segundo fazendo com que os testes de Hash tenham validade apenas naquele segundo específico. Uma vez encontrado o nonce, qualquer nó da rede poderá validar o bloco com aquele nonce dado. É como aquele famoso jogo “Onde Está Wally”, do autor inglês Martin Handford, onde num cenário com muitas, muitas, muitas figuras, os participantes deveriam encontrar o rapaz com camisa listrada e óculos redondo, Wally. Uma vez que um dos participantes do jogo encontra Wally e mostra para os outros, fica fácil deles comprovarem se o primeiro está certo ou não.

A figura abaixo retirada do site [https://www.blockchain.com](https://www.blockchain.com/" \o "https://www.blockchain.com/) mostra os detalhes de um bloco da rede Bitcoin.

Figura 26 – Bloco da rede Bitcoin. [https://www.blockchain.com/btc/block/748048](https://www.blockchain.com/btc/block/748048" \o "https://www.blockchain.com/btc/block/748048)

Podemos destacar, entre várias, algumas informações relevantes:

* Em vermelho – o Hash do bloco composto por 19 zeros a esquerda.
* Em amarelo – o TimeStamp, a data do bloco.
* Em verde – a raiz de Merkle, o Hash de todas as transações do bloco.
* Em laranja – o nonce do bloco

A quantidade de zeros à esquerda no Hash representa a dificuldade do bloco. Se considerarmos o SHA-256, que gera um Hash de 64 caracteres hexadecimais (4 bits x 64 caracteres = 256) temos um cenário de 16⁶⁴ Hashs, aproximadamente 10⁷⁷ valores diferentes. Se criarmos uma blockchain que tem como regra, um target de 10 zeros à esquerda isso reduziria a nossa possibilidade de encontrar um Hash válido de 16⁶⁴ ⁻ ¹⁰ = 16⁵⁴, algo em torno de 10⁶⁵. Ainda parece um número grande, mas se calcularmos a probabilidade de acerto (10⁶⁵ / 10⁷⁷) encontramos o resultado de 10⁻¹², ou seja, um número bem pequeno. Na figura 26, de exemplo da rede Bitcoin, o target com 19 zeros indica um cenário de dificuldade de 10⁵⁴ / 10⁷⁷ = 10⁻²³.

Após encontrar o nonce válido para a dificuldade estabelecida, o minerador é recompensado financeiramente e o bloco é distribuído para os outros nós da rede, que em posse do nonce válido validam o bloco e o incluem em sua cadeia. Na situação onde há um empate, ou seja, dois nós distintos encontram o nonce válido no mesmo TimeStamp, o protocolo de consenso espera a próxima inserção de blocos na cadeia e aquela cadeia que inserir o próximo bloco válido é reconhecida como a cadeia mais longa, e portanto, a cadeia válida para continuar a crescer na rede blockchain, desempatando o cenário anterior.

#### 6.1.7.2 PROOF OF STAKE (PoS) – PROVA DE PARTICIPAÇÃO

A principal diferença entre PoS e PoW é que no PoS não há prova de mineração através daquele cálculo matemático que explicamos anteriormente, mas sim prova de validação. Umas das críticas ao PoW é a carga energética necessária para validação dos blocos pelos mineradores e a baixa escalabilidade da rede, e no protocolo Proof of Stake a validação é feita com uma espécie de “sorteio” daquele que vai validar o próximo bloco na rede.

Para que o participante, usuário validador, nó, participe deste processo de consenso é preciso que ele aloque moedas na rede numa operação especial indicando que deseja validar blocos na rede. Aqueles que desejam participar deste processo e alocam moedas na transação de validação formam um comitê que entre várias premissas para a decisão do vencedor, tem a variável da quantidade de moeda alocada pelo participante, fazendo com que quanto maior a quantidade de moeda alocada, maior seja o seu peso na decisão.

Todo este processo de decisão deve acontecer dentro de uma janela de tempo chamada de **slot** (fenda) e havendo consenso entre os validadores o bloco é aprovado e inserido na cadeia. Um único bloco pode ser inserido em um slot e um conjunto de slots é conhecido como **epoch** (época). Um comitê é válido apenas dentro de uma época.

Em seu mecanismo de aprovação, o PoS recompensa os validadores que votaram a favor da validação do bloco com a volta de suas moedas aportadas inicialmente na transação e mais um valor financeiro. Caso o bloco seja rejeitado, ou seja, não seja considerado como válido para inserção na rede, aqueles que votaram positivamente para a sua inserção não recebem de volta suas moedas aportadas e acabam tendo perda financeira.

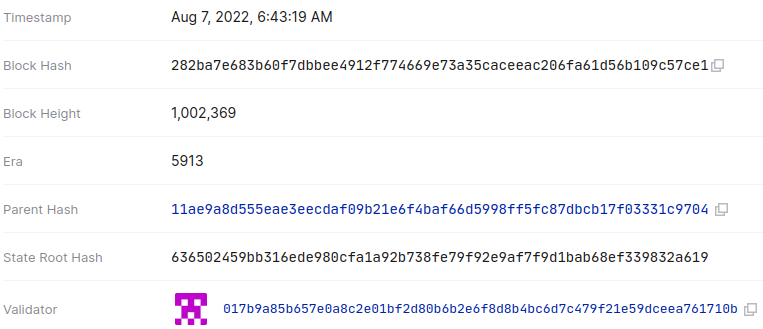


Figura 27 – Bloco da rede Casper. https://cspr.live/block/282ba7e683b60f7dbbee4912f774669e73a35caceeac206fa61d56b109c57ce1

Na figura 27 vemos um bloco da rede Casper, que utiliza Proof of Stake. Nele, encontramos informações relevantes referentes ao bloco inserido:

* TimeStamp – é a data do bloco, data de inserção na rede;
* Block Hash – é o Hash identificador do bloco, que o qualifica unicamente na rede;
* Era – é o **epoch** do bloco;
* Parent Hash – é o seu bloco anterior da cadeia de blocos;
* Validator – é a identificação, endereço, do nó validador.

Como vantagens sobre o protocolo Proof of Work, o Proof os Stake apresenta:

1. Menores custos associados a equipamentos e estrutura física.
2. Maior velocidade, permitindo mais transações em menor tempo.
3. Menor gasto energético, pois não utiliza mineração.
4. É mais descentralizado que o PoW, visto que vários atores participam do processo de validação.

Se por um lado, a fragilidade do PoW está em um ataque à maioria dos nós da rede, a fragilidade do PoS estaria em um participante que tivesse em seu poder uma grande monta de moeda corrente da rede para fazer seus aportes de participação em comitê de validação. As discussões são muitas entre as diferenças entre as redes que utilizam PoW e PoS, não existindo uma melhor ou pior, mas cenários onde uma proposta se adéqua mais que outra. Se uma proposta prioriza segurança e imutabilidade, o PoW, a princípio, se apresenta como mais adequado, se a busca for por maior escalabilidade e velocidade, sem deixar de lado, é claro, a segurança, o PoS, a princípio, se apresenta como melhor proposta.

Não é incomum uma rede iniciar seu trabalho utilizando um determinado protocolo de consenso e depois mudar para outro protocolo. Este é o caso da rede Ethereum. Apesar de iniciar a operar oficialmente utilizando o protocolo Proof of Work, ela está migrando suas operações para o protocolo Proof of Stake (**The merge**[[33]](#footnote-32)).

#### 6.1.7.3 PROOF OF AUTORITY (PoA) – PROVA DE AUTORIDADE

O Proof of Autority (PoA) é um protocolo de consenso com funcionamento muito parecido com o PoS, com a diferença que no PoA os validadores são agentes previamente selecionados e muitas vezes necessitando algum tipo de validação externa documental ou governamental, mas seguindo a mesma ideia de consenso estabelecida no PoS.

Essa característica quebra um dos pilares da blockchain pública que é o seu caráter descentralizado, na medida em que há um agente central que define quem pode e quem não pode participar da rede, fazendo uma solução típica para blockchains privadas. A critério de curiosidade, o Real Digital, que é uma proposta de **Carteira Digital de Banco Central** (CDBC) está apontando para uso de um protocolo de PoA.

### 6.1.8 ESTRUTURA DO BLOCO

A esta altura, mencionar que o bloco é o elemento central de toda a estrutura deveria ser redundante, e a sua formatação depende das especificações da rede em uso, mas tipicamente, o bloco é composto por duas estruturas fundamentais, o cabeçalho e o corpo. No cabeçalho ficam as metainformações e no corpo o histórico de transações.[[34]](#footnote-33)

O cabeçalho é uma estrutura de 80 bytes distribuídos da seguinte forma:

1. 4 bytes para sua identificação e regra para validação do bloco (Block Version);
2. 32 bytes para armazenar o Hash do bloco atual (Merkle Tree Root Hash);
3. 4 bytes para o TimeStamp;
4. bytes para representar a dificuldade de mineração (nBits);
5. 4 bytes para representar o Nonce;
6. 32 bytes para armazenar o Hash do bloco anterior (Parent Block).

Já o corpo do bloco contém um contador de transações (Transaction Counter) e as transações do bloco (TX), propriamente ditas. O número máximo de transações e a quantidade delas depende do tamanho do bloco e do tamanho das transações.[[35]](#footnote-34)

A imagem abaixo mostra a organização de todos esses elementos do bloco

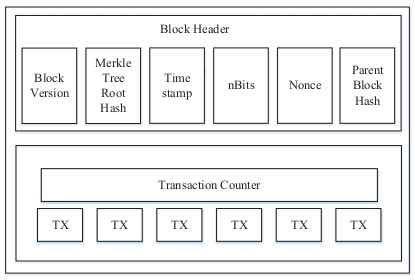


Figura 28 – Estrutura de um bloco na rede blockchain.[[36]](#footnote-35)

## 6.2 EVOLUÇÃO DA WEB

Importante trabalho apresentado por OLIVEIRA[[37]](#footnote-36), MAZIERO e ARAÚJO mostra a evolução da Web até a nova Web 3.0. Neste trabalho os autores apresentam a internet como um recurso provido pela informática que conecta pessoas, sendo a Web 3.0 uma evolução deste recurso, onde os elementos de inteligência e organização são ampliados.

É difícil precisar um marco zero, um dia D, se assim podemos falar, sobre o início da internet, mas, sem dúvida, os trabalhos desenvolvidos nos idos de 1991 e protagonizados por Tim Berners-Lee nos laboratórios do CERN (*Conseil Européen pour la Recherche Nucléaire*) na Suíça, são grandes marcos desta tecnologia.

Na esteira da evolução, a Web 1.0 (década de 1990) se caracterizava com recursos de conectividade entre os ambientes e pessoas mas com baixa interação, onde elementos como links e páginas estáticas eram a essência. Em seguida, a Web 2.0 (início dos anos 2000) trouxe elementos de maior colaboração e mídias sociais, onde a relação entre as pessoas e comunidades ganhou destaque e a interação entre seus participantes é o foco. Com a Web 3.0 - iniciada em 2001 por artigo de Tim Berners-Lee, ainda em desenvolvimento - se propõe uma internet mais inteligente e voltada para as máquinas, criando um conteúdo mais propício à leitura feita por estas, Web Semântica.

Os autores continuam sua obra e descrevem as características da Web Semântica, sendo então uma técnica para construção da Web voltada para a leitura do conteúdo por não humanos, assim, utilizando a linguagem XML2 ao invés de HTML, é possível ampliar o poder de indexação dos elementos. Com o poder de indexação ampliado, é possível filtrar o conteúdo da rede de forma cada vez mais personalizado, ampliando a experiência de uso e a especialização das informações e comunidades participantes.

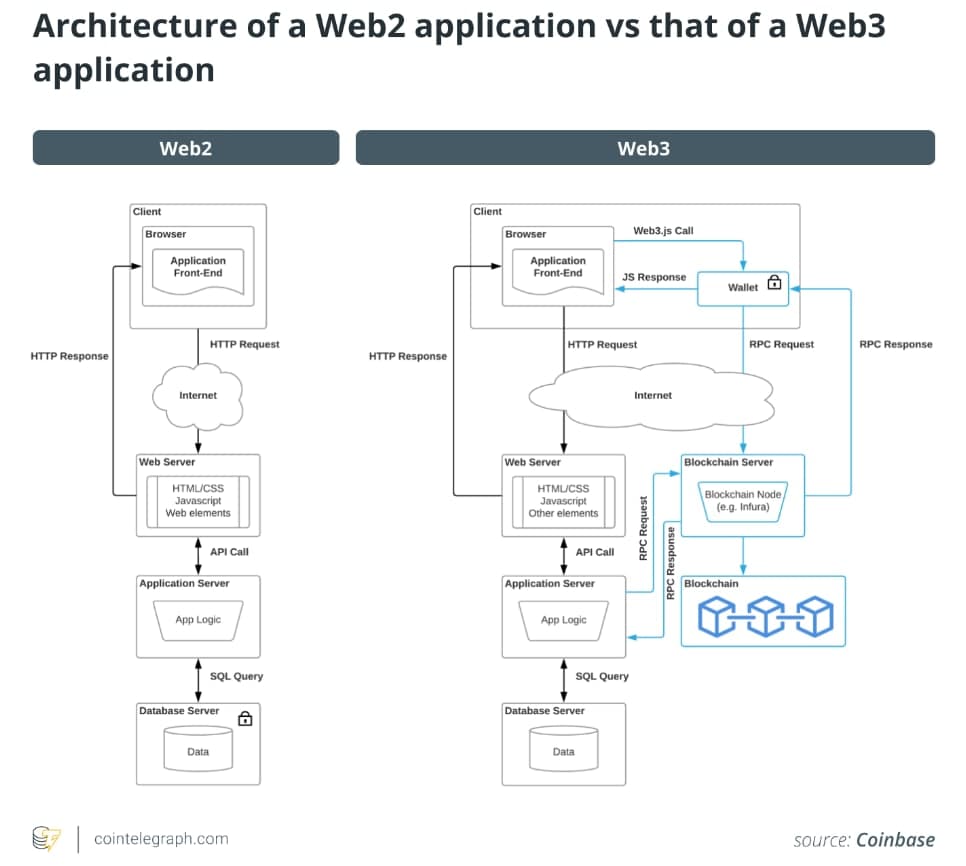
Esta nova estrutura conta com a introdução, ao lado dos dados propriamente ditos, de metadados e ontologia, marcando a semântica, significado, daquele dado apresentado e sua ontologia, seu fim, objetivo.

Além disso, a Web 3.0 se destaca por uma nova proposta, onde o foco não são mais os grandes *players* como Google, Meta, Apple, entre outros, mas o indivíduo, assim sendo, novos negócios e filosofias que fazem o compartilhamento de dados (sejam eles um vídeo, um documento ou até mesmo valor monetário) de forma independente da participação de um terceiro para validação ganha espaço e coloca o poder novamente na mão das pessoas e não das grandes companhias.

Outro famoso artigo escrito por ZAGO[[38]](#footnote-37) faz, também, essa comparação entre as diferentes fases da Web e cita algumas vantagens esperadas com a descentralização proposta pela Web 3.0:

1. Ausência de um ponto de controle central – governos ou entidades perdem a capacidade de bloquear sites e serviços e não há um indivíduo central podendo controlar a identidade das pessoas.
2. Propriedade de dados – usuários finais recuperaram o controle dos dados e fazem uso de criptografia.
3. Redução de ataques *hackers* e vazamento de dados – como os dados são descentralizados e distribuídos, para que os dados fossem comprometidos, um ataque teria que afetar todos os nós da rede e aliado à tecnologia de criptografia, o sigilo dos dados fornece uma camada de proteção a mais.
4. Interoperabilidade – Possibilidade de aplicações poderem *rodar* em diferentes dispositivos e plataformas, de maneira agnóstica, não presa a um sistema operacional em específico e não fazendo diferenciação entre plataformas iOs ou Android, por exemplo.[[39]](#footnote-38)
5. Ausência de permissão para participar da blockchain – tratando aqui das blockchains públicas[[40]](#footnote-39). Qualquer pessoa pode criar um endereço e interagir com a rede, não havendo barreiras geográficas, gênero ou qualquer outra restrição social ou demográfica.
6. Funcionamento ininterrupto – Suspensão de conta e ataque de negação de serviço (DDOS) são dramaticamente reduzidos, pois não há um concentrador central, um ponto único de falha, os nós da rede garantem redundância de dados e proporcionam maior garantia da continuidade do serviço.

Pelas diferenças apresentadas, fica claro que as soluções propostas para a Web2 e Web3 se mostram bem diferentes e essa diferença acaba se desdobrando na forma como a arquitetura pra cada uma destas duas soluções tradicionalmente se desenvolve. Em artigo publicado em 2018 TEKISALP na **coinbase[[41]](#footnote-40)** apresenta essas diferenças e a figura abaixo, mostra as diferenças arquiteturais em uma típica aplicação baseada em Web2 comparada a outra aplicação Web3[[42]](#footnote-41).

Figura 29 – Aplicação Web2 vs aplicação Web3.

## 6.3 SOLUÇÕES BLOCKCHAIN

Como mencionado na introdução desta obra, há diversas soluções no mercado que utilizam a tecnologia blockchain, e para tentar ilustrar este panorama, serão apresentadas nesta seção algumas alternativas jé em mercado que fazem uso dessa tecnologia.

Para começar, a obra de MORAIS e LINS[[43]](#footnote-42) traz projetos na área de educação. Neste trabalho, os autores começam reconhecendo que o documento mais importante no sistema educacional é o diploma e que para evitar fraudes, a tecnologia blockchain se apresenta como solução eficaz para evitar diplomas falsos e adulterações, garantindo a idoneidade das informações da instituição e alunos, permitindo que eles sejam validados a qualquer momento por um interessado.

Já existem ferramentas no mercado voltadas para esta aplicação, onde podemos

destacar a Blockcerts, desenvolvida pelo MIT e a TrueRec, criada pelo grupo SAP. Outras propostas são: Smart Contracts na plataforma Ethereum, o UniCert, uma solução que usa validação em uma base de dados local em PostgrSQL e o registro do hash em uma blockchain, o SmartCert que utiliza a plataforma Ethereum e o CredenceLedger, outra aplicação baseada na rede Ethereum.

Muita soluções além do registro de diplomas podem ser gerenciadas com a tecnologia blockchain. Histórico escolar, o desenvolvimento da aprendizagem, dados científicos, pagamentos, entre outros, são soluções em que esta tecnologia pode ser utilizada de forma bastante segura e garantindo a validação por qualquer interessado.

Vale lembrar que para o desenvolvimento de uma solução em blockchain é possível adotar uma arquitetura pública ou privada, onde esta permite a entrada apenas de pessoas autorizadas e aquela é aberta ao público em gera. Cada uma destas propostas tem suas vantagens e desvantagens e deve ser adotada aquela que fornece melhor alternativa ao projeto.

Outros cuidados que devem ser levados em consideração na implementação de uma solução blockchain para registro de informações educacionais é a capacidade da rede local para lidar com essa nova carga de dados, olhando aqui tanto para a capacidade e segurança necessária para tal, a capacidade de se integrar com sistemas legados, e a capacidade de integração de forma segura com ferramentas de aprendizagem educacional externas.

Na mesma esteira tecnológica, NIWA apresenta um sistema de voto eletrônico baseado em blockchain. Em seu documento, o autor faz um estudo dos sistemas de votação passando pelo tradicional sistema de papel e em seguida apresenta os sistemas com votação eletrônica. Há as chamadas urnas de 1ª geração (DRE – *Direct Recording Eletronic voting machine*), onde os votos são registrados na memória de um equipamento eletrônico e posteriormente contados pelo administrador do sistema e o desenvolvedor do software, sendo este o modelo utilizado no Brasil. Há, ainda, as urnas de 2ª geração (IVVR – *Independent Voter Verifiable Record),* que imprimem um comprovante em papel possibilitando a auditoria contábil dos votos e a de 3ª geração (*End-to-End verifiable*)*,* que contam com dispositivos de rádio frequência possibilitando a conferência do eleitor independente de software.

Na sequência é apresentada a motivação de um novo sistema que inclui, principalmente, as falhas denunciadas por: CUNHA[[44]](#footnote-43) e demais anomalias que poderiam ser melhoradas denunciadas por ARRIAL[[45]](#footnote-44), TSE[[46]](#footnote-45) e RAMALHO[[47]](#footnote-46). E como solução, é apresentado um sistema de votação que utilize a tecnologia *blockchain*, sendo: “um banco de dados criado através de uma rede distribuída, descentralizada e com criptografia de chaves público-privada e algoritmos de hash seguros”[[48]](#footnote-47).

## 6.4 SMART CONTRACTS

Como proposta de nosso trabalho, iremos utilizar a estratégia de smart contracts para desenvolver a solução. Mas esta estratégia é válida? É reconhecida? O Direito a reconhece como instrumento viável? Na busca dessas respostas o trabalho de CARVALHO e ÁVILA nos ajuda a responder.

Em seu artigo, os autores discorrem sobre a teoria dos contratos e definem os smart contracts como nada mais que contratos codificados (representados em código de programação) que tem execução automática e autônoma.

Ao definir contrato, em sua forma tradicional, os autores trazem a visão deste conceito sob a ótica de renomados juristas e destaca como definição:

“contrato é negócio jurídico que necessita obrigatoriamente de ao menos duas partes para compor a relação”[[49]](#footnote-48).

Mas ressalta que apesar da liberdade de contrato entre as partes, este contrato deve estar alinhado com preceitos legais estabelecidos no ordenamento.

Já quanto aos Contratos Inteligentes, conhecidos também como *Smart Contracts*, inferem a origem do termo a Nick Szabo, na obra *Smart Contracts: Building Blocks for Digital Free Markets”* de 1996. Nesta obra, Nick Szabo apresenta a verificabilidade, o acompanhamento, a privacidade e a exigibilidade como melhorias nos contratos trazidas pelos Contratos Inteligentes.

Após tratar sobre contratos, os autores abordam a tecnologia chave para esses Contratos Inteligentes, Blockchain, sendo esta, uma cadeia de blocos organizados através de uma rede *Peer-to-Peer* (Ponto-a-Ponto), que funciona como um banco de dados descentralizado das transações e dados dos blocos, organizados cronologicamente e criptografados através de *hash*.

Como exemplos de soluções que fazem uso desse novo modelo de contratos são apresentados cenários como na fabricação e venda de veículo, financiamento de automóvel e soluções como *OriginalMy*, que registra prova de autenticidade em custos muito mais competitivos que cartórios notariais, e *OpenLaw*, solução na área jurídica que fornece modelos de contratos que podem ser adaptados por seus signatários e que também fazem uso da tecnologia blockchain.

Com essa nova tecnologia a celeridade, a certeza do cumprimento e a solução da falta de confiança entre as partes são peças chave que colocam os Contratos Inteligentes como alternativa eficaz para adquirir, extinguir, modificar ou executar direitos e obrigações.

Insta mencionar o consenso entre as partes para pactuarem desta forma, sendo, então, a tecnologia blockchain alternativa viável, mas não única fonte para todos os modos de pactuação de contratos. Superadas as devidas ressalvas, este novo modelo, já nem tão novo assim, se apresenta como modelo viável e eficaz para atender diversos cenários onde a celeridade, objetividade e certeza de cumprimento se apresentam na realidade dos contratos.

## 6.5 ETHEREUM

Ethereum é uma blockchain pública não permissionada, de código aberto e de uso geral, [[50]](#footnote-49) idealizada por Vitalik Buterin e lançada oficialmente em 2015. Como diferencial, permite que clientes interajam com a rede através de linguagens de programação para a execução de Smart Contracts. Enquanto a rede Bitcoin se concentra em se consolidar como uma rede de pagamentos, a rede Ethereum se posiciona como um Marketplace.

Essas características possibilitaram que esta rede fosse palco para receber diversas aplicações nos mais diversos segmentos, desde o setor financeiro, saúde, moda, entretenimento, etc. Alguns destaques sobre a rede Ethereum estão sobre a sua utilização para soluções DeFi (Decentralized Finances), DAO (Decentralized Autonomous Organization), NFT (Non-Fungible Tokens) entre outras.

Os dados da rede Ethereum impressionam. De acordo com dados da State of the dapps[[51]](#footnote-50), com dados de maio de 2022, há 2970 projetos utilizando a rede Ethereum, mais de 50 milhões de Smart Contracts utilizando sua rede[[52]](#footnote-51) e mais de 1.6 bilhão de transações em sua rede[[53]](#footnote-52).

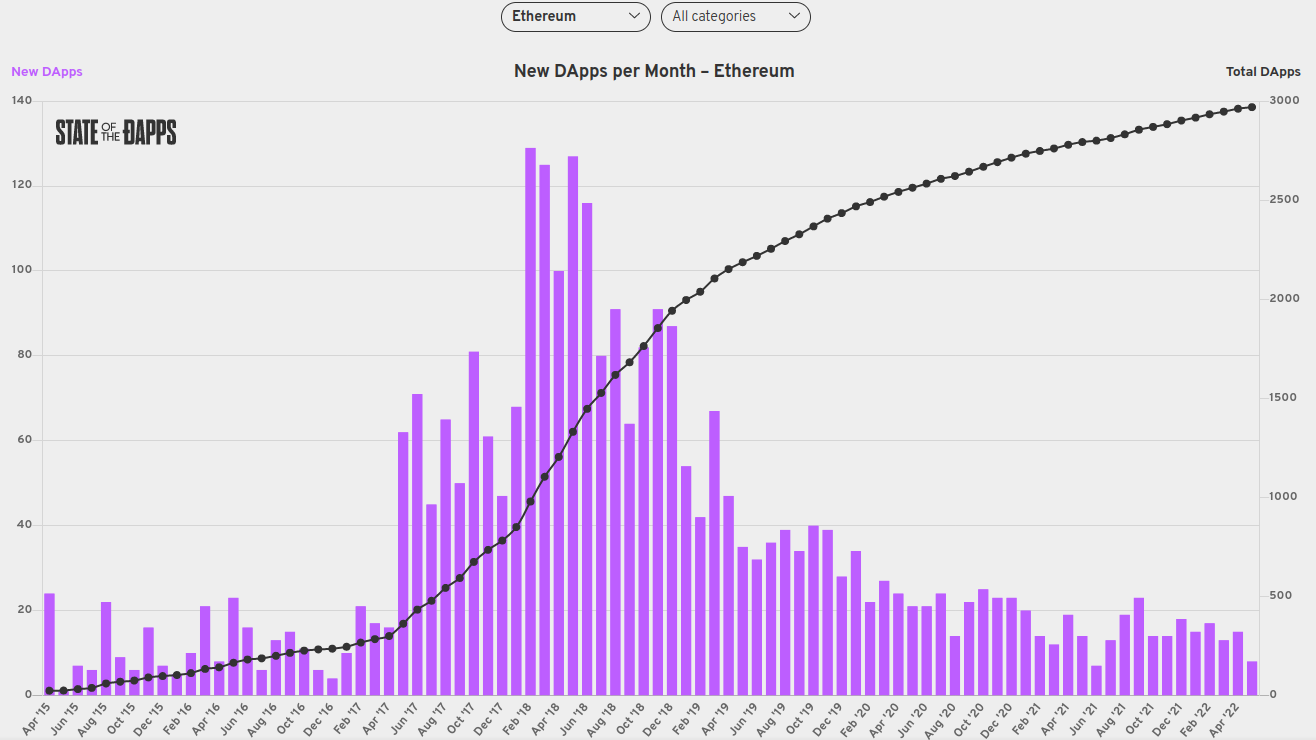


Figura 30 – Projetos utilizando Ethereum. 50

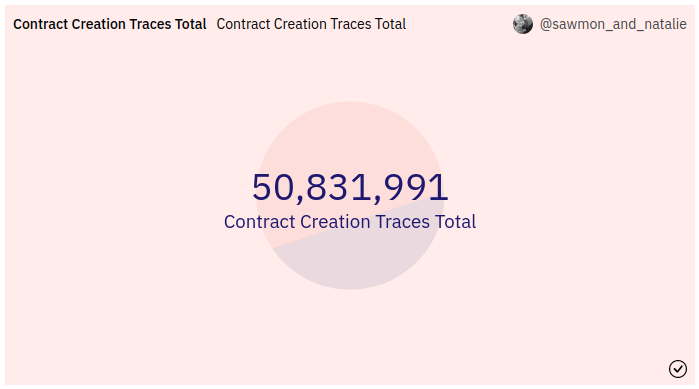


Figura 31 – Smart Contracts utilizando a rede Ethereum.51

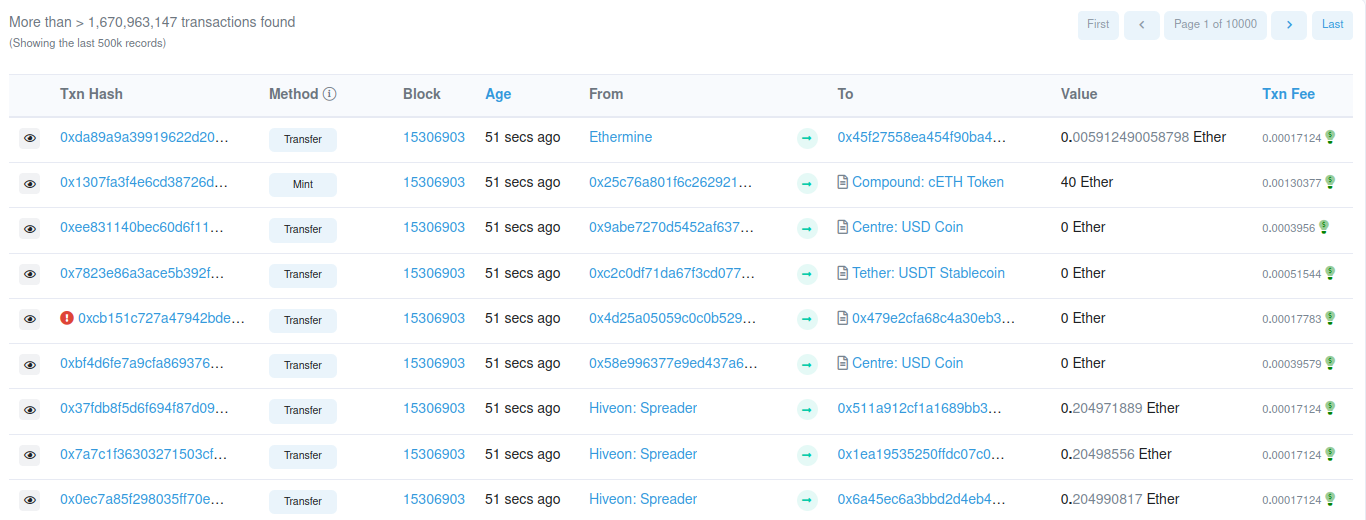
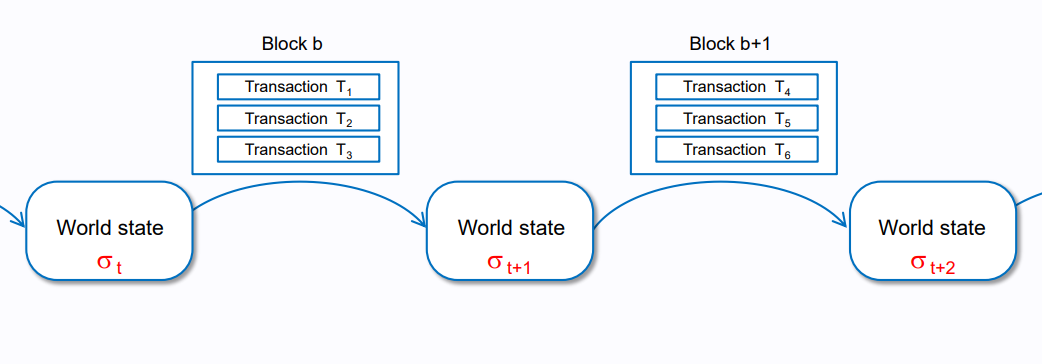


Figura 32 – Smart Contracts utilizando a rede Ethereum.52

### 6.5.1 ETHEREUM VIRTUAL MACHINE (EVM)

A Máquina Virtual do Ethereum (Ethereum Virtual Machine - EVM) é uma máquina virtual nos padrões Turing-Complete, podendo executar qualquer programa expresso em código. No contexto da blockchain, a propriedade Turing-Complete é importante porquê permite que contratos inteligentes sejam criados e executados na plataforma. O ponto de contato entre a EVM e os contratos inteligentes se dá nos *bytecodes,* que são códigos gerados pelo processo de compilação e carregados na EVM. Através dos *bytecodes* a EVM reconhece o conjunto de instruções, entradas e saídas de um contrato inteligente.

A EVM é ainda, uma máquina de estados distribuída onde cada nó da rede executa esta máquina e para isso, há protocolos próprios para controlar estes estados em diferentes pontos da rede. O estado no Ethereum é uma grande estrutura de dados que não tem apenas contas e saldos, como no caso do Bitcoin, mas também um estado da máquina, que pode mudar de bloco pra bloco. Se o blockchain pode ser traduzido em uma cadeia de blocos a EVM pode ser traduzida em uma cadeia de transações executadas em uma máquina de estados.

Figura 33 – Cadeia de estados na EVM.[[54]](#footnote-53)

A EVM se comporta como uma função matemática em que de acordo com a entrada, ele produz uma saída determinística. Portanto, é bastante útil descrever mais formalmente o Ethereum como tendo uma **função de transição de estado[[55]](#footnote-54)**:

***Y (S, T) = S’***

*Sendo:*

***Y →*** *Uma função de transição*

***S →*** *Um estado inicial*

***T →*** *Um conjunto de transações válidas*

***S’ →*** *Um novo estado final de saída*

Se olharmos para a estrutura da EVM poderemos perceber algo bem semelhante a uma estrutura computacional como conhecemos hoje, mas há um elemento importante nessa estrutura que age como um agente de custo das operações que é o Gas. O Gas é uma expressão computacional diretamente ligada ao esforço computacional dispendido para realizar uma operação solicitada à EVM. Se a operação for muito custosa, mais Gas será exigido, se for menos custosa, menos Gas será gasto. Em linhas gerais, podemos entender o Gas como uma espécie de pedágio para as operações da EVM. Essa característica é importante pois estimula os desenvolvedores a escreverem códigos cada vez mais otimizados para fazerem uso da EVM da forma mais otimizada possível.

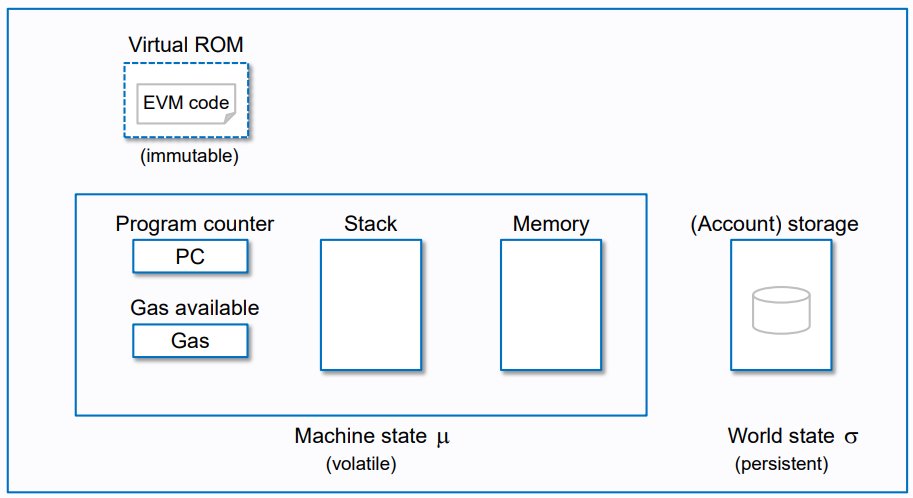


Figura 34 – Cadeia de estados na EVM.[[56]](#footnote-55)

### 6.5.2 CONTAS

Outro elemento importante na estrutura do Ethereum é a conta. Conta, no mundo Ethereum é um objeto que se relaciona diretamente a um estado da EVM. Há dois tipos de conta: Conta de proprietário externa (Externally Owned Accout –EOA) e Conta de Contrato (Contract Account – CC). Ambas são representadas por uma identificação de 20 dígitos em hexadecimal e esta identificação designa o endereço desta conta. As contas de proprietário são associadas a uma pessoa e são independentes da EVM, já as contas de contrato são auto gerenciadas pela EVM. Somente contas de proprietário externa iniciam uma transação.

Enquanto as contas de proprietário externa possuem apenas duas informações associadas a ela que são a quantidade de moeda nativa associada a essa conta, chamada de **balance**, e um número decimal usado para o controle de transações referenciadas a esta conta que inicial em zero e é incrementado a cada transação, chamado de **nonce**, as Contas de Contrato possuem, além de **balance**  e **nonce**, também código que a EVM pode executar (o código programado no contrato - bytecode) e variáveis de estado que armazenam dados. As contas de proprietário são controladas pela sua chave privada, ao passo que as contas de contrato são controladas pela EVM.

Outro detalhe importante que diferencia esses dois tipos de conta é que a conta de proprietário externa é composta por um par de chaves pública e privada (para maiores detalhes reveja a sessão 6.1.3 ondefalamos sobre criptografia e validação) que identificam o proprietário de uma conta (identificação) e o autor de uma transação (não repúdio).

Quando o usuário da blockchain cria uma conta será gerado uma chave privada aleatória composta por 64 caracteres hexadecimal e a partir dela, utilizando um algorítimo de assinatura digital de curva eliptica será gerad a chave pública desta conta. Seu endereço público é formado com base nessa chave pública sendo os últimos 20 bytes do hash Keccak-256 da chave pública e adicionado **0x** no início.[[57]](#footnote-56)

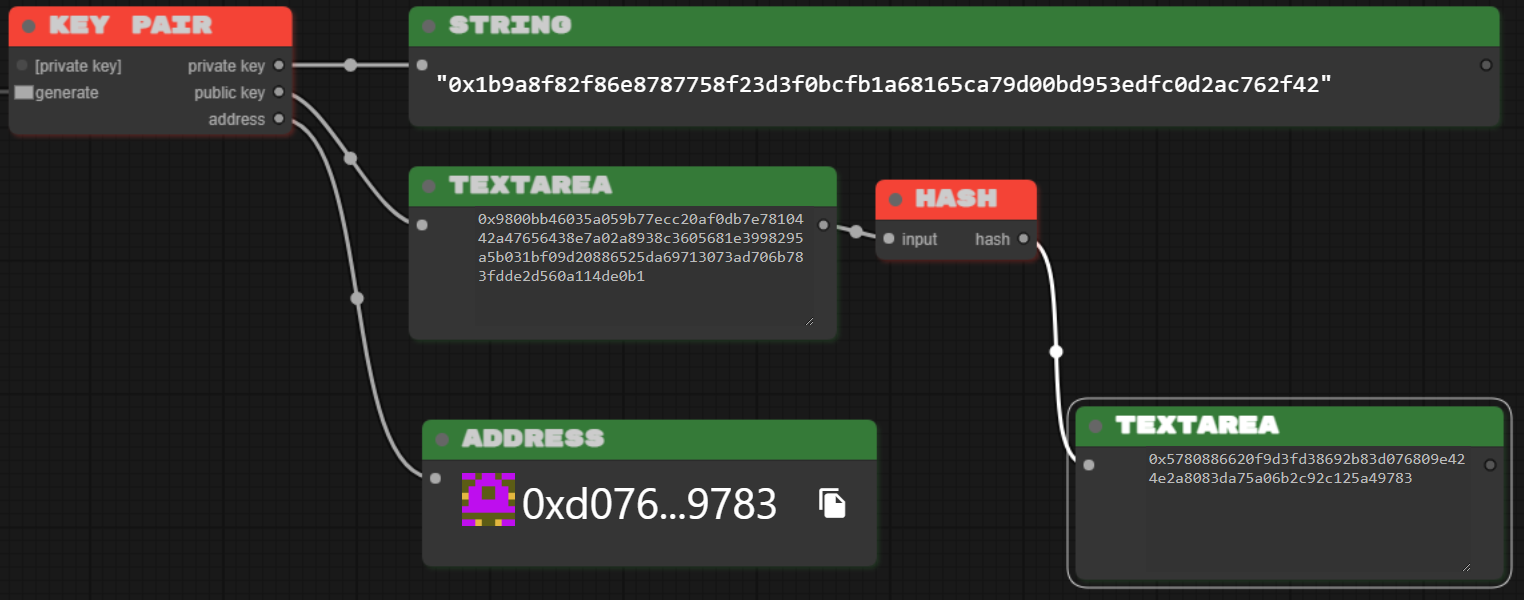


Figura 35 – Chave privada, pública e endereço de uma conta.[[58]](#footnote-57)

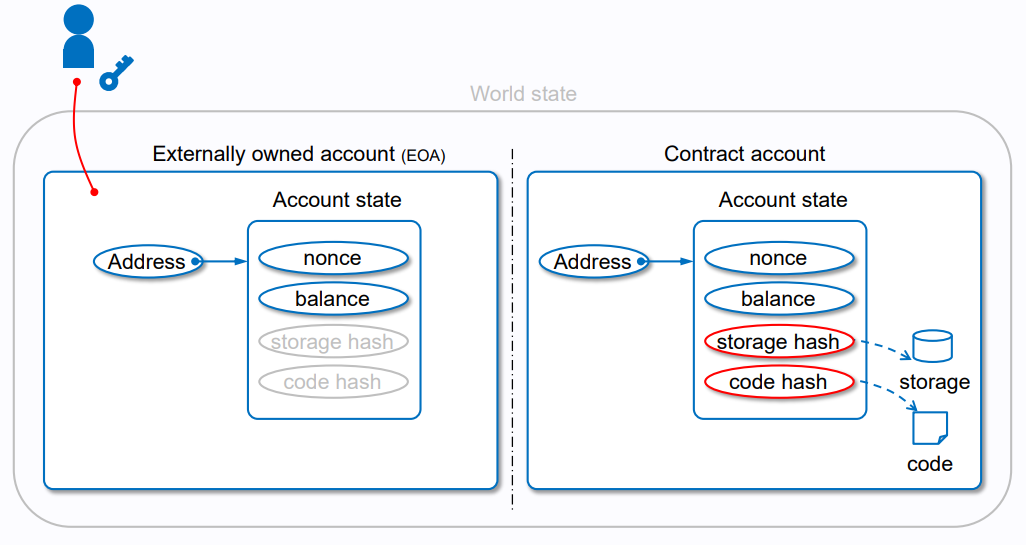


Figura 36 – Cadeia de estados na EVM.[[59]](#footnote-58)

### 6.5.3 TRANSAÇÕES

Transações são instruções submetidas por um ator externo criptograficamente assinadas que alteram o estado da blockchain.

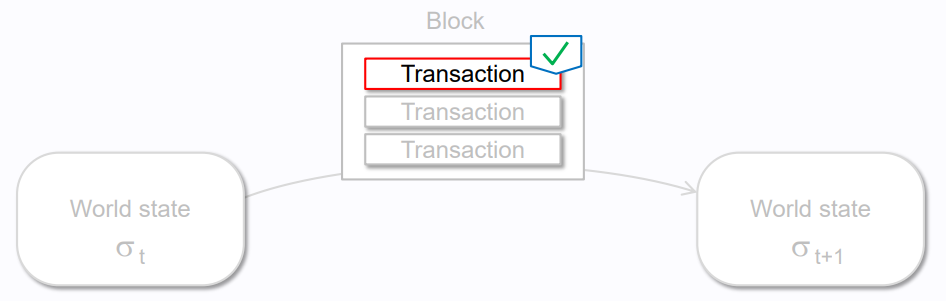


Figura 37 –Transação na EVM.[[60]](#footnote-59)

As transações que alteram o estado da blockchain devem ser replicadas para toda a rede. Essas transações são solicitadas por uma conta e registradas em um bloco. Qualquer nó pode transmitir uma solicitação para que uma transação seja executada na EVM; depois que isso acontecer, um validador executará a transação e a propagará para toda a rede.

Uma transação enviada a EVM possui as seguintes informações:

* **Recipient**: se for uma conta externa, esta receberá o valor; se for um contrato, executará o código do contrato;
* **Signature**: o identificador do remetente mediante o uso de sua chave privada e mensagem;
* **Nonce**: um contador de incremento sequencial que indica o número de transações da conta;
* **Value**: quantidade de ETH (moeda do Ethereum) a ser transmitida para o recipient;
* **Data**: campo adicional que possui dados arbitrários;
* **GasLimit**: quantidade máxima de gas que pode ser consumida pela transação;
* **MaxPriorityFeePerGas**: quantidade máxima de gas a ser incluída como gorjeta para o validador;
* **MaxFeePerGas**: quantidade máxima de gas a ser paga pela transação.

Existem três tipos básicos de transações no Ethereum:

* **Transações regulares**: transação entre contas externas;
* **Transações de criação de contrato**: transações sem um endereço de destino onde são enviados os dados do código do contrato ;
* **Transações de execução de contrato**: transações que chamam alguma função de um contrato criado na rede Ethereum.

Importante frisar que uma transação é atômica na EVM. Uma transação não pode ser dividida, nem interrompida. A estrutura blockchain como um todo pode ser considerado como uma estrutura que possui propriedades ACID (Atomicidade, Consistência, Isolamento e Durabilidade).

### 6.5.4 PATRICIA MERKLE TREE

Em sessões anteriores falamos sobre a utilidade da Merkle Tree para guardar hashs das transações, mas como a rede Ethereum tem algumas complexidades a mais como dados para armazenamento, é interessante utilizar uma outra estrutura de dados capaz de atender a esta demanda, e a Patricia Merkle Tree tem esse propósito.

A Patricia Merkle Tree (PMT), também conhecida como Trie-based Merkle Tree, é uma estrutura de dados que combina as árvores de Patricia e as árvores de Merkle. É usada principalmente em sistemas distribuídos e blockchains para garantir a integridade dos dados e a segurança das transações. A árvore de Patricia é uma estrutura de dados de árvore digital que armazena chaves em seus nós internos, em vez de apenas em suas folhas. Isso permite uma rápida pesquisa de prefixo, que é útil em sistemas de indexação de palavras, por exemplo. Já as árvores de Merkle são árvores binárias hash-based, onde cada nó folha é o hash de uma transação e cada nó interno é o hash dos seus nós filhos.

A PMT combina essas duas estruturas, onde cada nó folha da árvore de Patricia armazena um hash do conteúdo correspondente, enquanto os nós internos armazenam hashes das suas subárvores. Assim, a PMT permite que sejam verificadas as provas de autenticidade e integridade das transações e dos dados armazenados na árvore.

Dessa forma, a Patricia Merkle Tree oferece uma solução eficiente para a validação de dados em sistemas distribuídos e blockchains, permitindo que se verifique se as informações armazenadas em diferentes nós são iguais sem a necessidade de transferir grandes quantidades de dados pela rede.

# 7 METODOLOGIA CIENTÍFICA

O presente trabalho trata de uma pesquisa de natureza aplicada por quê o produto apresentado nesta obra destina-se a solucionar o problema pertinente ao armazenamento de informações centralizadas na gestão de controle de ponto. Quanto à abordagem, será de caráter qualitativo porquê envolve menos quantidade de dados e aprofunda melhor as questões envolvidas.

Sendo assim, a forma como será desenvolvida esta pesquisa será de natureza descritiva, uma vez que serão apresentados diversos diagramas como os de caso de uso, classe e sequência, entre outros, definidos pela UML[[61]](#footnote-60). Além disso, com relação aos procedimentos de pesquisa, será realizado um estudo de caso do tema problematizado acima.

## 7.1 QUESTIONÁRIO

Na busca do entendimento do mercado e cenário buscado no atendimento à demanda de criação da solução, foi adotada a estratégia de criar um questionário com 20 perguntas, sendo 10 delas voltadas para os funcionários e 10 delas voltadas para os gestores que cuidam da parte de RH (Recursos Humanos).

Neste cenário, foi criado um pequeno sistema web para registro das questões e armazenamento das respostas utilizando arquitetura MVC conforme abaixo:

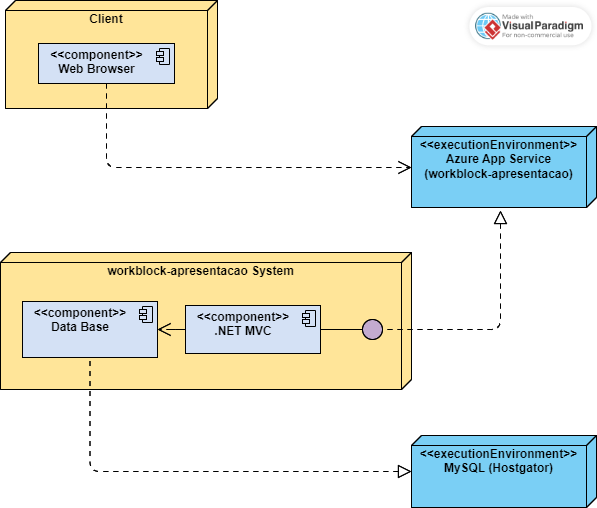


Figura 38 –Diagrama de representação do projeto web para questionário.

Esta aplicação apresenta duas partes distintas, sendo elas: uma página web estática com uma apresentação sobre o projeto, e um questionário, onde de acordo com o perfil desejado (profissional ou gestor de RH) são apresentadas as perguntas do questionário. A URL do projeto pode ser acessada no seguinte endereço: [https://workblock-apresentacao.azurewebsites.net/.](https://workblock-apresentacao.azurewebsites.net/." \o "https://workblock-apresentacao.azurewebsites.net/.)



Figura 39 – Página de apresentação do projeto. [https://workblock-apresentacao.azurewebsites.net/.](https://workblock-apresentacao.azurewebsites.net/." \o "https://workblock-apresentacao.azurewebsites.net/.)

.

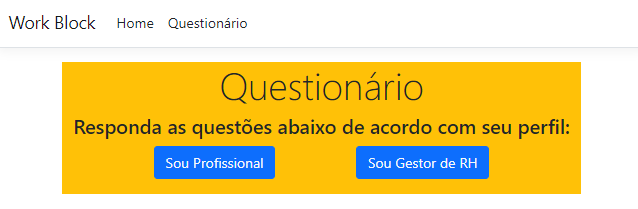


Figura 40 – Página para questionário. [https://workblock-apresentacao.azurewebsites.net/Questionario.](https://workblock-apresentacao.azurewebsites.net/." \o "https://workblock-apresentacao.azurewebsites.net/.)

Após a divulgação da proposta e questionário em rede social, entre os profissionais foram obtidos os resultados conforme abaixo:

**Respostas referente a profissionais.**

Analisando os dados notamos que o uso do registro de ponto é algo plenamente natural e de ampla utilização, sendo a opção mais adotada o registro de ponto utilizando meio eletrônico para tal. Dentre os entrevistados foi notado que mais da metade já enfrentaram problemas com seu registro de ponto, mesmo que de forma não frequente, aliado há isso, há o fato de cerca de apenas 1/4 dos entrevistados terem acesso aos seus registros diretamente, fazendo então com que ampla maioria ache relevante um sistema com mais recursos como uso por aplicativo ou reconhecimento facial e cerca de um pouco mais da metade não confie plenamente no sistema adotado para registro de ponto. Já quanto à adoção de uma tecnologia blockchain, aproximadamente metade dos entrevistados apoia a ideia e nenhum deles reconhece essa proposta como contrária à segurança dos registros e otimização na operação do sistema.

De modo geral, percebemos com os resultados que a utilização de um meio eletrônico e a adoção da tecnologia blockchain para este fim são apoiadas. Logicamente, a pesquisa poderia ser ampliada para um universo maior de entrevistados para um resultado mais conclusivo, mas esse levantamento inicial já nos mostram sinais interessantes no apoio de nosso projeto.

Figura 41 – Resultado questão 1 do perfil Profissional.

Figura 42 – Resultado questão 2 do perfil Profissional.

Figura 43 – Resultado questão 3 do perfil Profissional.

Figura 44 – Resultado questão 4 do perfil Profissional.

Figura 45 – Resultado questão 5 do perfil Profissional.

Figura 46 – Resultado questão 6 do perfil Profissional.

Figura 47 – Resultado questão 7 do perfil Profissional.

Figura 48 – Resultado questão 8 do perfil Profissional.

Figura 49 – Resultado questão 9 do perfil Profissional.

Figura 50 – Resultado questão 10 do perfil Profissional.

**Respostas referente a profissionais.**

Analisando os dados notamos .....

# 8 SOLUÇÕES UTILIZADAS NO MERCADO

Conforme mencionado na seção 3 desta obra, o Registro Eletrônico de Ponto (REP) é regulado pela portaria 671/2021 do MTP e reconhece três tipos de soluções para registro eletrônico destas operações, REP-C, REP-A e REP-P. Essas soluções sã desenvolvidas por grandes *players* do mercado, inclusive, com um deles, a DIMEP, fundada em 1936 e desenvolvendo relógios de ponto desde 1959.

Figura 51 – Exemplos de equipamentos físicos para registro de ponto.



Figura 52 – Exemplo de software para registro de ponto.

De uma forma geral, podemos dividir esses equipamentos em dois grandes grupos: um deles funcionando como um equipamento independente (stand alone) e o outro formado com software de gestão podendo fazer uso, ou não, de algum equipamento físico para os registros. Independente de qual dos dois tipos de solução adotada, todas elas fazem uso de uma arquitetura centralizada, sofrendo de todos os problemas típicos de aplicações desse tipo conforme já mencionamos nas seções anteriores.

## 8.1 MODELO DE NEGÓCIO

O modelo de negócio utilizado para este tipo de produto conta basicamente com players de mercado que atuam no segmento de controle de acesso e controle de ponto. Alguns deles fornecem soluções integradas para estes dois cenários e outros não, mas não necessariamente são implementadas as soluções de um só fabricante. Em minha experiência profissional já me deparei em campo com empresas que utilizam sistema de controle de acesso e controle de ponto nas mais diversas associações.

Neste modelo de negócio o player pode vender, alugar ou comodatar o equipamento, podendo, ainda, oferecer um contrato de manutenção mensal para atendimentos ao equipamento e ao software de gestão.

# 9 ESPECIFICAÇÕES DO PROJETO

Após apresentado todo o cenário do projeto, passaremos para a fase de modelagem e especificação da solução.

## 9.1 REGRAS DE NEGÓCIO

Como fruto de determinações legais e dinâmica de uso da solução, decrevemos abaixo as regras de negócio que orientarão o desenvolvimento.

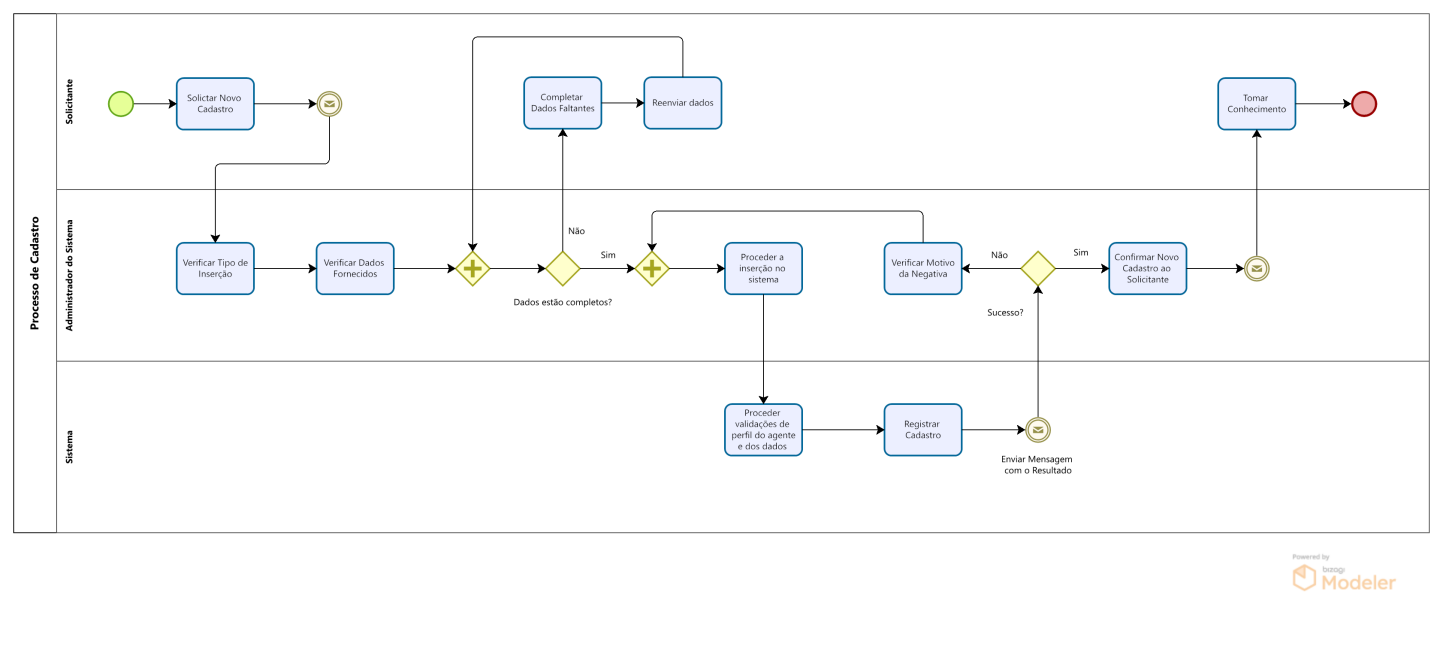
* **Regra de tipo de perfi**l: O sistema deve possuir perfil de administrador (gestor) e funcionário.
* **Regra de privilégio de gestor**: Somente a pessoa com perfil de gestor poderá cadastrar as informações do empregador, do funcionário, inserir novos gestores, fazer buscas no sistema e emitir relatórios (salvo condições especiais onde o funcionário poderá ver alguns de seus relatórios). Somente administradores ativos poderão interagir com o sistema.
* **Regra de segurança**: Somente pessoas registradas no sistema poderão interagir com o mesmo para alterar e visualizar dados.
* **Regra de criação de administrador**: O administrador ao ser criado deve possuir uma identificação única (id), nome, número de documento e o estado (ativo ou inativo).
* **Regra de criação de empregador**: O empregador ao ser criado deve possuir informação que o identifique de forma única (id), nome, número de documento e endereço.
* **Regra de criação de funcionário**: O funcionário ao ser criado deve possuir informação que o identifique de maneira única (id), nome, número de documento, jornada de trabalho, empregador e o estado (ativo ou desativado).
* **Regra de criação de dados**: Qualquer dado novo ao ser inserido no sistema deve ser registrado a identificação de quem fez tal adição.
* **Regra de alteração de dados**: Os dados de qualquer entidade do sistema ao serem modificados (nome, documento, etc) devem ser registrados a identificação de quem fez tal modificação.
* **Regra de horário**: o sistema ao ser implantado deve aceitar o ajuste de fuso horário para adequação de acordo com o fuso horário desejado.
* **Regra de tipo de ponto**: Devem haver quatro marcações de ponto: início da jornada, início da pausa, fim da pausa, fim da jornada; e somente perfil funcionário poderá registrar ponto.
* **Regra de marcação geral de ponto**: O ponto deve ser marcado mediante identificação do funcionário no sistema e posterior escolha do ponto a ser marcado, com o sistema, de forma automática, reconheça o horário e registre o ponto, devendo o ponto registrar o funcionário, o tipo de ponto marcado e o horário. Somente funcionários ativos marcarão pontos.
* **Regra de marcação de inicio de jornada**: O início da jornada deverá ser obrigatoriamente a primeira marcação do dia, não permitindo que seja feita uma marcação de outro tipo em seu lugar.
* **Regra de marcação de início da pausa**: A marcação de início da pausa só poderá ser realizada se houver um registro de marcação de início de jornada no mesmo dia.
* **Regra de marcação de fim de pausa**: A marcação de fim de pausa só poderá ser realizada se houver registros de início de jornada e início de pausa anteriores a ela no mesmo dia.
* **Regra de marcação de fim de jornada**: A marcação de fim de jornada só poderá ser marcada se houver registro de marcação de início de jornada no mesmo dia. Caso haja um registro de início de pausa e não haja fim de pausa o sistema deverá se comportar da seguinte forma:
  + Se o intervalo entre o início da pausa e a marcação de fim de jornada for **inferior a 1h**, o sistema deverá registrar primeiro o fim da pausa e logo em seguida o fim da jornada.
  + Se o intervalo entre o início da pausa e a marcação de fim de jornada for **superior a 1h**, o sistema deverá registrar primeiro o fim da pausa com um período de 1h entre o início da pausa e o fim da pausa e logo em seguida o fim da jornada.
* **Regra de emissão de relatório por gestor**: Apenas gestor poderá emitir relatórios fazendo busca por qualquer funcionário e qualquer data ou período.
* **Regra de emissão de relatório por funcionário**: O funcionário registrado poderá emitir relatórios apenas sobre seu registro.

## 9.2 MODELAGEM DE PROCESSOS

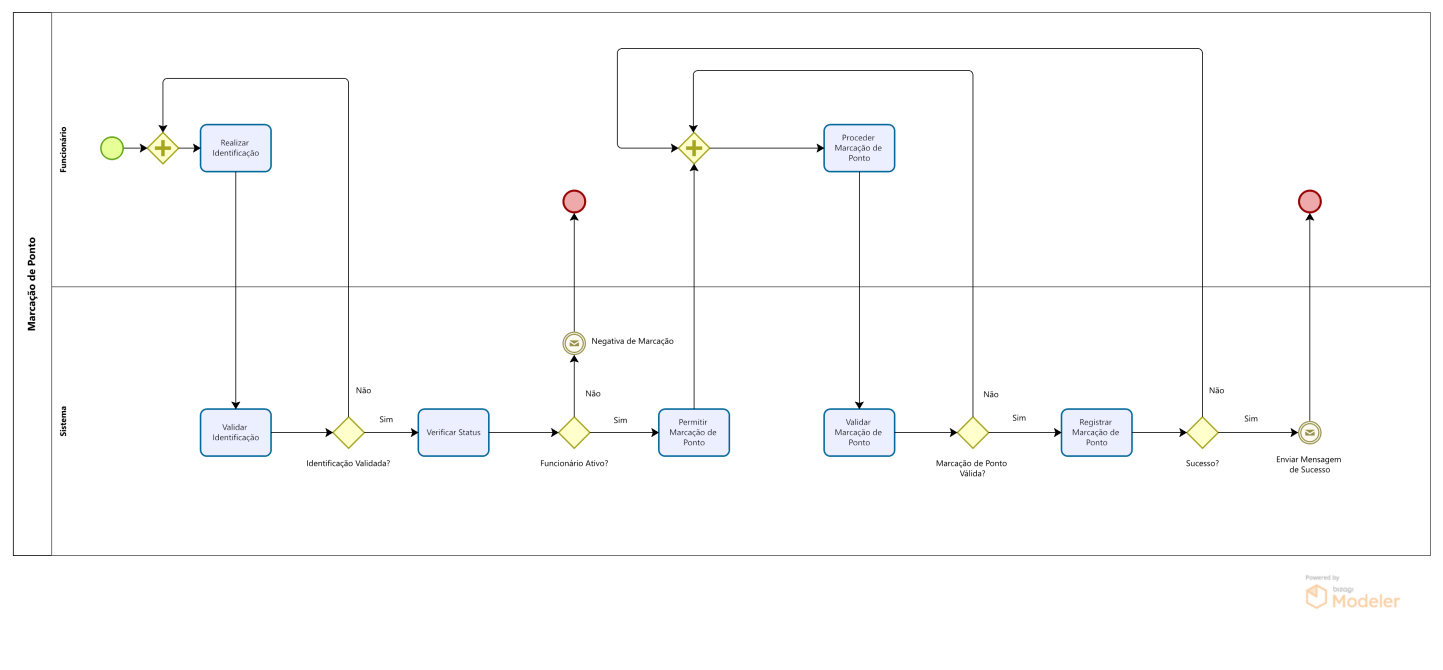
Observando a proposta do sistema observamos três processos distintos que descrevem a operação do sistema, a saber:

* Processo de Cadastro: responsável por cadastrar os perfis e utilizadores do sistema;
* Processo de Marcação de Ponto: responsável pelo funcionário realizar sua marcação de ponto;
* Processo de Emissão de Relatório: responsável pela emissão dos relatórios por parte do perfil administrador e do funcionário.

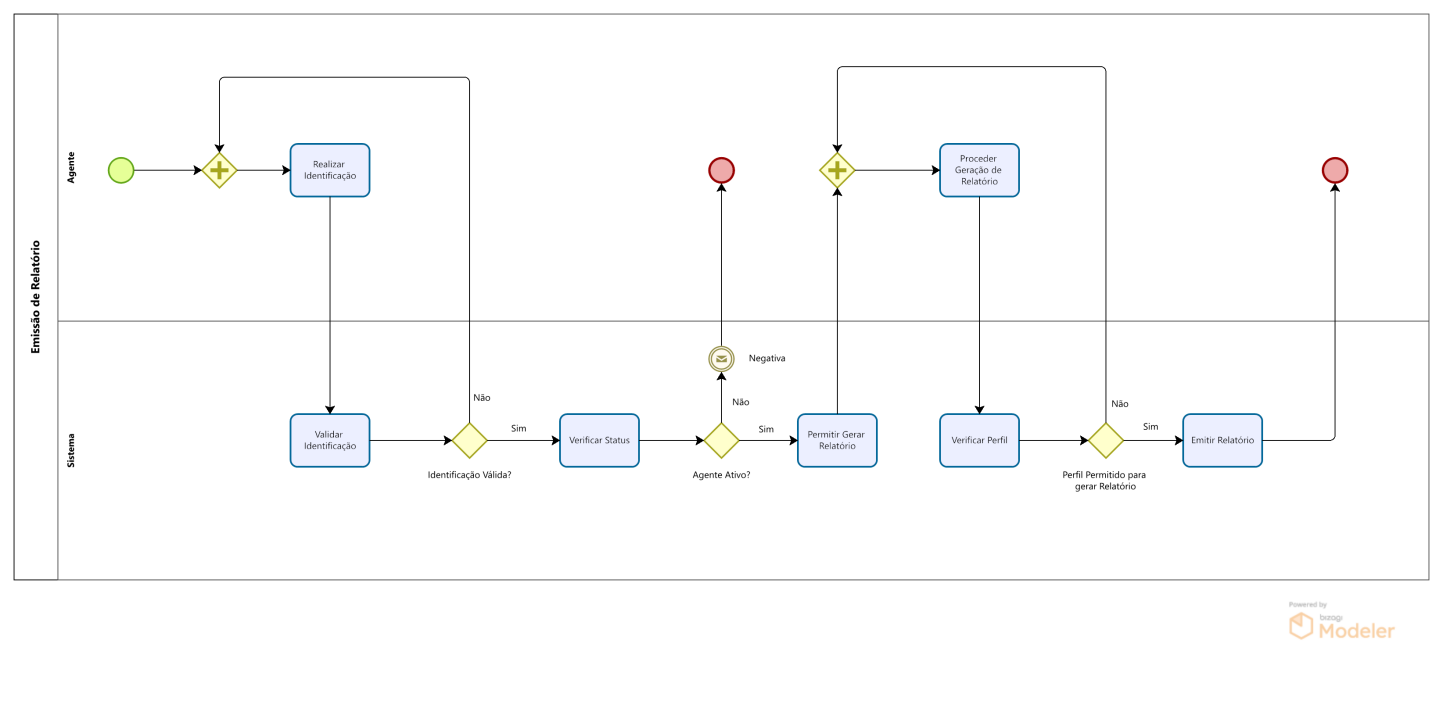
Na imagem abaixo vemos o processo de cadastro onde um solicitante faz o pedido de um novo cadastro ao administrador e após a validação das informações fornecidas, o administrador procede com a gravação do novo cadastro (administrador, empregador ou funcionário) e retorna com a confirmação para o solicitante.



Na imagem seguinte descrevemos o processo de marcação de ponto onde um funcionário após realizar sua identificação e realizar sua validação, a marcação de ponto é realizada.



E no último processo, temos a emissão de relatórios, que pode ser feita por administrador ou funcionário, onde o funcionário pode emitir seus relatórios e o administrador pode emitir relatório de qualquer funcionário e em qualquer período de data. Da mesma forma que os anteriores, é feita identificação e validação das ações e perfis.



## 9.3 REQUISITOS

Após reconhecer as regras de negócio e desenhar os processos envolvidos estamos aptos a descrever os requisitos do nosso sistema, assim temos a tabela abaixo:

|  |  |
| --- | --- |
| **REQUISITOS FUNCIONAIS** | |
| **REF** | **DESCRIÇÃO** |
| **RF01** | O sistema deverá incluir pelo menos dois perfis de usuário, um sendo gestor e o outro funcionário, que realizará a marcação de pontos. |
| **RF02** | O sistema deverá ser capaz de identificar de forma única e inequívoca cada usuário que utiliza o sistema. |
| **RF03** | Somente o perfil administrador poderá cadastrar e atualizar participantes do sistema, devendo sempre registrar o autor dessas ações. |
| **RF04** | O sistema poderá aceitar múltiplos usuários de tipo gestor e funcionário, mas cada um terá sua identificação única. |
| **RF05** | O sistema deverá cadastrar o empregador com os dados de nome, documento e endereço legal, todos de caráter obrigatório. |
| **RF06** | O sistema deverá cadastrar funcionário com os dados de nome, documento, início e fim da jornada de trabalho, seu empregador associado e status (ativo ou inativo), todos de caráter obrigatório. |
| **RF07** | O sistema deverá cadastrar o gestor com os dados de nome, documento e status (ativo ou inativo), todos de caráter obrigatório. |
| **RF08** | O sistema deverá ser capaz de registrar o fuso horário em que será utilizado. |
| **RF09** | O sistema deverá reconhecer 4 tipos de marcação de ponto: início e fim da jornada e início e fim da pausa. |
| **RF10** | O sistema deverá permitir que apenas funcionários ativos realizem a marcação de ponto. |
| **RF11** | A primeira marcação do dia deverá ser, sempre, início da jornada e na sequência, início da pausa, fim da pausa e fim da jornada, não admitindo inversão de qualquer dessa ordem nem replicações do mesmo tipo de marcação de ponto durante o dia, mas podendo admitir a ausência das marcações de pausa. |
| **RF12** | Se houver marcação de início de pausa, obrigatoriamente, deverá ser registrada a marcação de fim da pausa. |
| **RF13** | Se durante a marcação de fim de jornada houver ocorrência de início de pausa sem uma marcação de fim de pausa correspondente no mesmo dia, caso o intervalo entre o início da pausa e o fim da jornada seja maior que 1 hora, deverá ser registrada uma marcação de fim da pausa 1 hora após o início da jornada e posteriormente a marcação do fim da jornada do dia, caso esse intervalo seja menor que 1 hora, as marcações de fim de pausa e fim da jornada serão marcadas juntas. |
| **RF14** | Todo registro de marcação de ponto deverá conter de forma inequívoca o funcionário, o tipo da marcação e o horário. |
| **RF15** | O sistema deverá permitir a emissão de relatórios realizados por funcionário ou gestor onde, o gestor poderá emitir relatório de marcação de ponto realizado por qualquer funcionário em qualquer dia e o funcionário apenas as suas marcações. |
| **RF16** | O registros de marcação de ponto devem estar disponíveis a qualquer tempo para visualização de seus relatórios, independente do horário do dia. |

|  |  |
| --- | --- |
| **REQUISITOS NÃO FUNCIONAIS** | |
| **REF** | **DESCRIÇÃO** |
| **RNF01** | O sistema deverá ser desenvolvido em ambiente web para que possa ser utilizado por qualquer funcionário, dentro ou fora da empresa. |
| **RNF02** | Devem ser utilizadas soluções Open Source em seu desenvolvimento. |
| **RNF03** | Devem ser respeitadas as determinações da LGPD. |
| **RNF04** | O sistema deverá funcionar em ambiente independente da empresa de forma que mesmo que a sede da empresa esteja fechada ou inoperante por qualquer motivo seja possível a marcação do ponto. |
| **RNF05** | Deverão ser desenvolvidos testes unitários para validar cada uma das operações do sistema, seja de algum tipo de cadastro, marcações de ponto ou emissão de relatórios. |
| **RNF06** | O desenvolvimento do sistema deve ser feito em diferentes camadas de forma que as regras de negócio possam ser evoluídas de forma independente da interface web utilizada. |
| **RNF07** | Os dados devem ser persistidos em uma estrutura perene que garanta a gravação individual de cada registro e independente, alinhando-se às propriedades ACID. |

## 9.4 CASOS DE USO

Casos de uso são ferramentas importantes que nos ajudam a perceber e definir a forma de interação dos usuários com o sistema proposto. De acordo com a modelagem dos processos, reconhecemos três principais tipos de interação com o sistema a saber: Cadastro, Marcação de Ponto e Emissão de Relatório, assim, apresentamos abaixo caso de uso com sua representação.

Durante nosso desenvolvimento serão utilizadas algumas nomenclaturas que nos ajudarão a simplificar, organizar e reconhecer os elementos no documento de análise, assim sendo, temos:

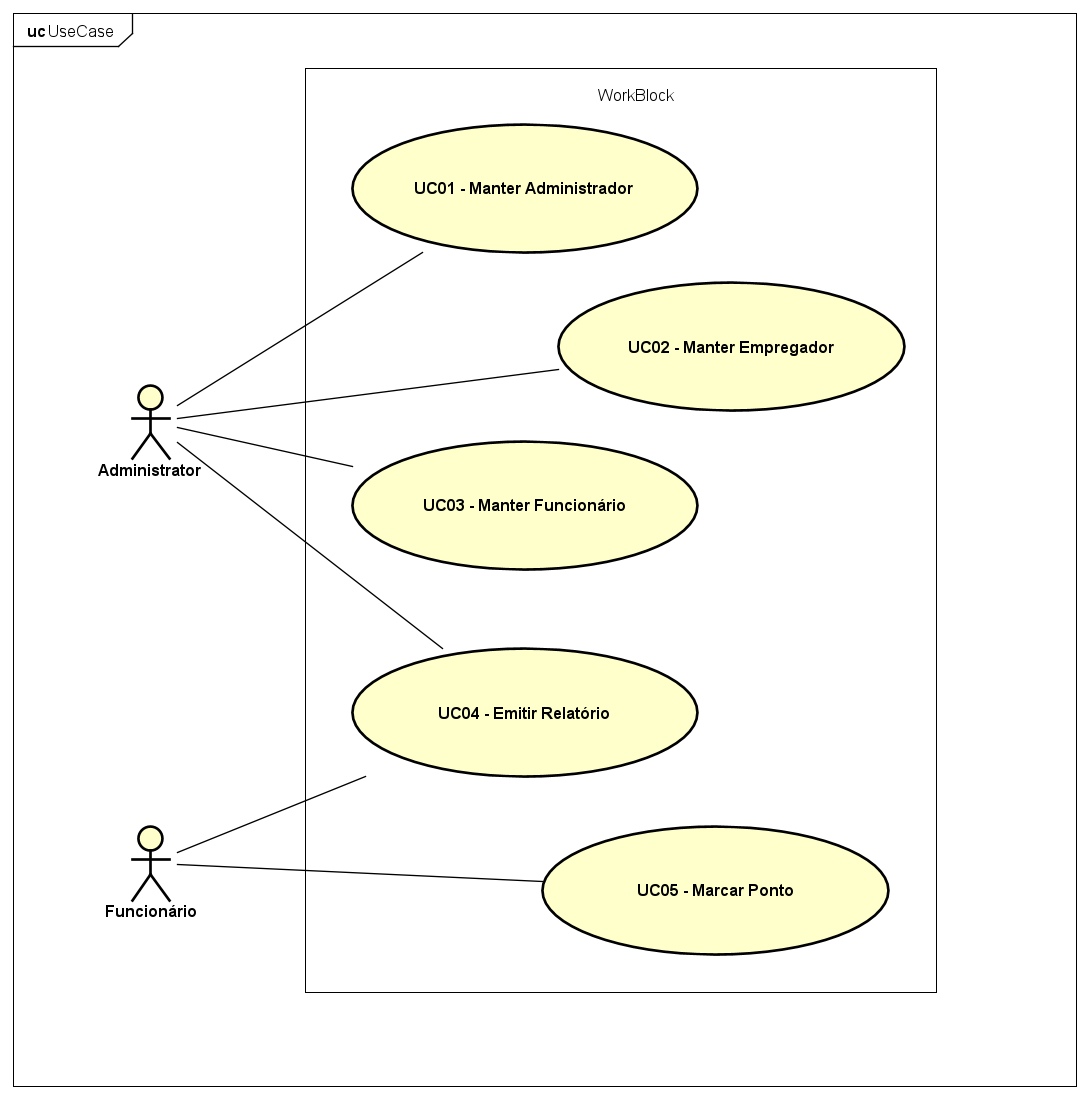
**UCXX**: Use Case (Caso de Uso) – Descrição de caso de uso. O **XX** representa uma sequência numérica de dois dígitos, sequencial e único para cada caso de uso, começando em 00. Exemplo: UC01 – Caso de uso 01.

**UCXX-AY:** Auxiliar Diagram (Diagrama Auxiliar) – Diagrama auxiliar de um determinado caso de uso para detalhar algum elemento ou grupo de elementos ou ações. O **XX** representa uma sequência numérica de dois dígitos, sequencial e única para o caso de uso em questão. O **A** é elemento fixo definindo que se trata de um Auxiliar. O **Y** representa uma sequência numérica de um dígito e única indicando o número do diagrama auxiliar começando em 1. Exemplo: UC00-A1 – Diagrama Auxiliar 1 do caso de uso 00.

Para o nosso sistema foram relacionados os seguintes casos de uso:

|  |  |  |
| --- | --- | --- |
| UC | NOME | DESCRIÇÃO |
| 00 | Geral | Visão geral do sistema |
| 01 | Manter Administrador | Visão das ações de cadastro de administrador |
| 02 | Manter Empregador | Visão das ações de cadastro de Empregador |
| 03 | Manter Funcionário | Visão das ações de cadastro de funcionário |
| 04 | Emitir Relatório | Visão das ações para emissão de relatório |
| 05 | Marcar Ponto | Visão das ações para marcação de ponto |

### 9.4.1 UC00 - Geral



**Sumário:** Estrutura geral e relacionamentos entre atores e caso de uso.

**Atores:**

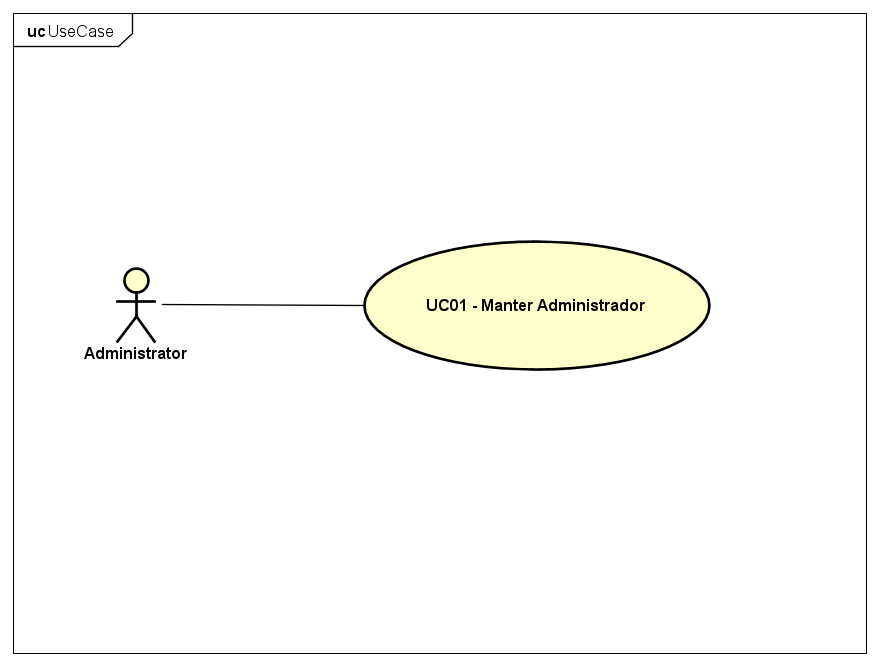
**Administrador:** Ator que ocupa posição de gestão cadastrando dados e emitindo relatórios. Este ator se relaciona com os casos de uso:

* UC01 - Manter Administrador,
* UC02 - Manter Empregador,
* UC03 - Manter Funcionário,
* UC04 - Emitir Relatório

**Funcionário:** Ator que faz uso do sistema na ação de marcação de ponto e ocasionalmente emite seus relatórios de marcação de ponto. Este ator se relaciona com os casos de uso:

* UC04 - Emitir Relatório,
* UC05 - Marcar Ponto

### 9.4.2 UC01 - Manter Administrador



**Sumário:** O ator Administrador mantém administrador no sistema.

**Ator principal:** Administrador

**Pré-condição:**

1. Ator cadastrado no sistema e ativo

**Pós-condições:**

1. Administrador mantido no sistema e apto para executar ações.

**Fluxo Principal:**

1. O Ator principal coleta as informações necessárias para cadastro no sistema. Informações necessárias para cadastro:

i - nome completo.

i.a - Se pessoa física, o nome registrado no documento oficial, se pessoa jurídica, a razão social.

ii - numero de documento oficial válido.

ii.a - Se pessoa física, preferencialmente CPF, se pessoa jurídica, preferencialmente CNPJ.

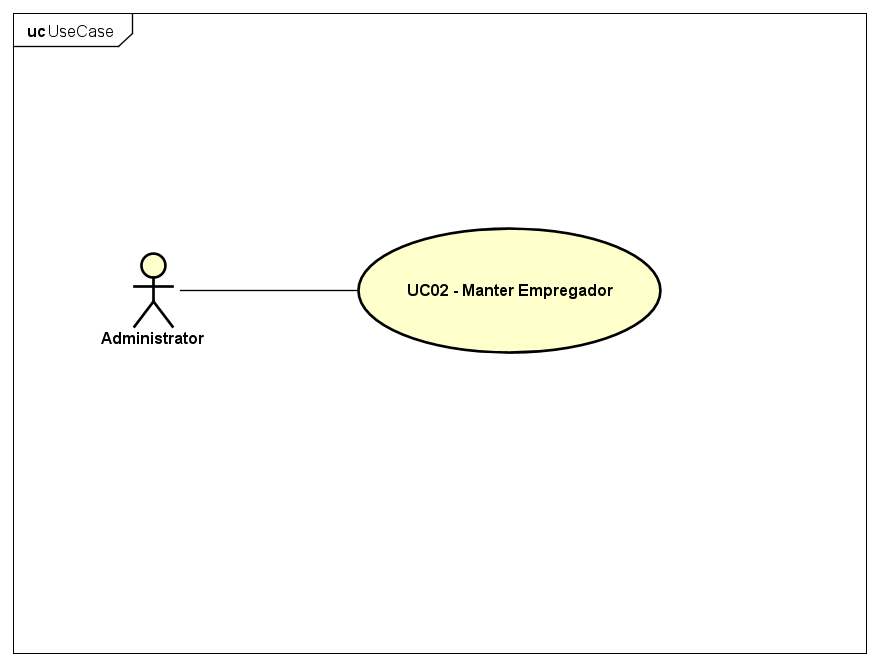
**Fluxo Alternativo [AF01-01]:**

1. Ator pode buscar administradores cadastrados no sistema.

**Fluxo Alternativo [AF01-02]:**

1. Ator pode atualizar dados de administradores cadastrados no sistema.

### 9.4.3 UC02 - Manter Empregador



**Sumário:** O ator Administrador mantém empregador no sistema.

**Ator principal:** Administrador

**Pré-condição:**

1. Ator cadastrado no sistema e ativo

**Pós-condições:**

1. Empregador mantido no sistema.

**Fluxo Principal:**

1. O Ator principal coleta as informações necessárias para cadastro no sistema. Informações necessárias para cadastro:

i - nome completo.

i.a - Se pessoa física, o nome registrado no documento oficial, se pessoa jurídica, a razão social.

ii - numero de documento oficial válido.

ii.a - Se pessoa física, preferencialmente CPF, se pessoa jurídica, preferencialmente CNPJ.

iii - endereço, localização, do empregador

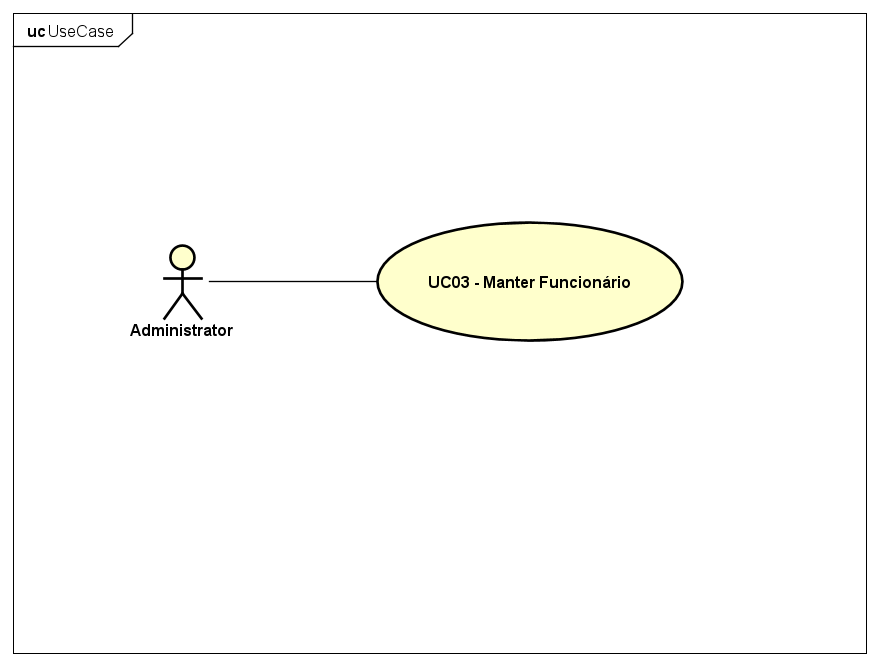
**Fluxo Alternativo [AF02-01]:**

1. Ator pode buscar empregadores cadastrados no sistema.

**Fluxo Alternativo [AF02-02]:**

1. Ator pode atualizar dados de empregadores cadastrados no sistema.

### 9.4.4 UC03 - Manter Funcionário



**Sumário:** O ator Administrador mantém funcionário no sistema.

**Ator principal:** Administrador

**Pré-condição:**

1. Ator cadastrado no sistema e ativo

**Pós-condições:**

1. Funcionário mantido no sistema.

**Fluxo Principal:**

1. O Ator principal coleta as informações necessárias para cadastro no sistema. Informações necessárias para cadastro:

i - nome completo registrado no documento oficial.

ii - numero de documento oficial válido, preferencialmente PIS ou CPF.

iii - hora de início e fim da jornada de trabalho do funcionário.

iv - referência do empregador

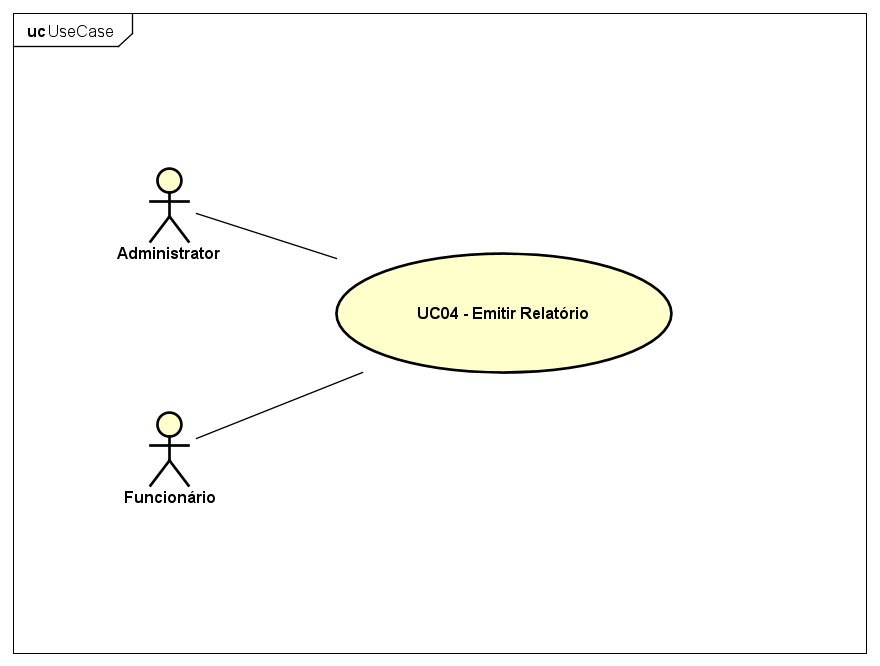
**Fluxo Alternativo [AF02-01]:**

1. Ator pode buscar funcionários cadastrados no sistema.

**Fluxo Alternativo [AF02-02]:**

1. Ator pode atualizar dados de funcionários cadastrados no sistema.

### 9.4.5 UC04 - Emitir Relatório



**Sumário:** Os atores emitem relatório de marcação de ponto.

**Ator principal:** Administrador

**Ator secundário:** Funcionário

**Pré-condição:**

1. Atores cadastrados no sistema e ativos.

2.a) Ator secundário consulta apenas os seus registros

**Pós-condições:**

1. Relatório desejado emitido pelo sistema.

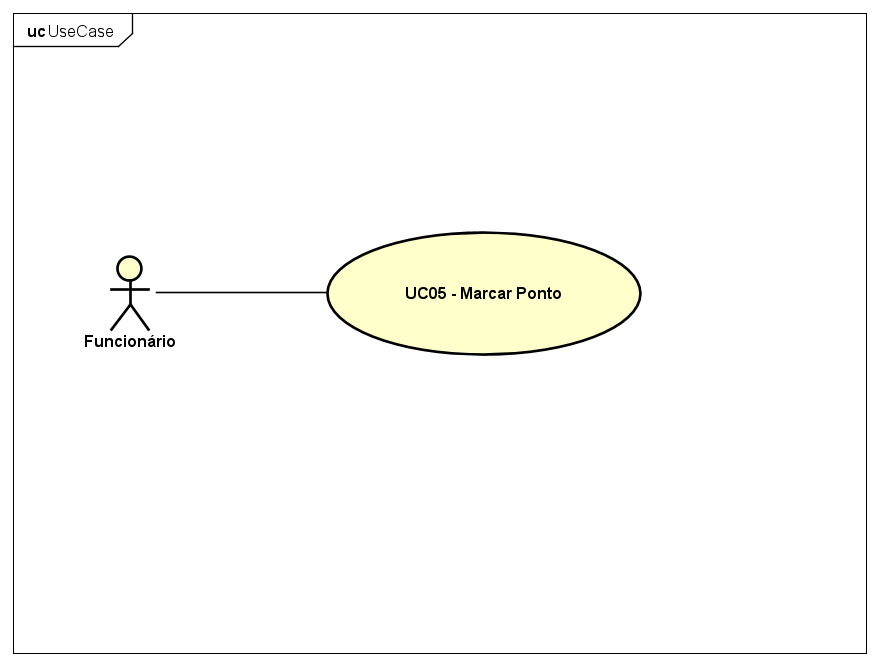
**Fluxo Principal:**

1. Os Atores entram com informação do funcionário e data buscada
2. O sistema emite o relatório desejado.

i - Se o agente for o Ator Secundário, o sistema valida para que a busca seja permitida apenas para os dados referentes a seu registro.

ii - Se o agente for o Ator Principal, o sistema permite a busca de registros de qualquer funcionário e qualquer data.

### 9.4.6 UC05 - Marcar Ponto



**Sumário:** O ator faz a marcação de ponto.

**Ator principal:** Funcionário

**Pré-condição:**

1. Ator cadastrado no sistema e ativo.

**Pós-condições:**

1. Marcação de ponto registrada sistema.

**Fluxo Principal:**

1. Os Ator escolhe o ponto a ser marcado
2. O sistema valida a solicitação de marcação de ponto
3. O sistema registra a marcação de ponto realizada.

## 9.5 DIAGRAMA DE CLASSE

O Diagrama de Classe descreve as entidades do sistema com seus atributos e comportamentos e seus relacionamentos entre si. Na imagem abaixo temos a representação do modelo de domínio do sistema.

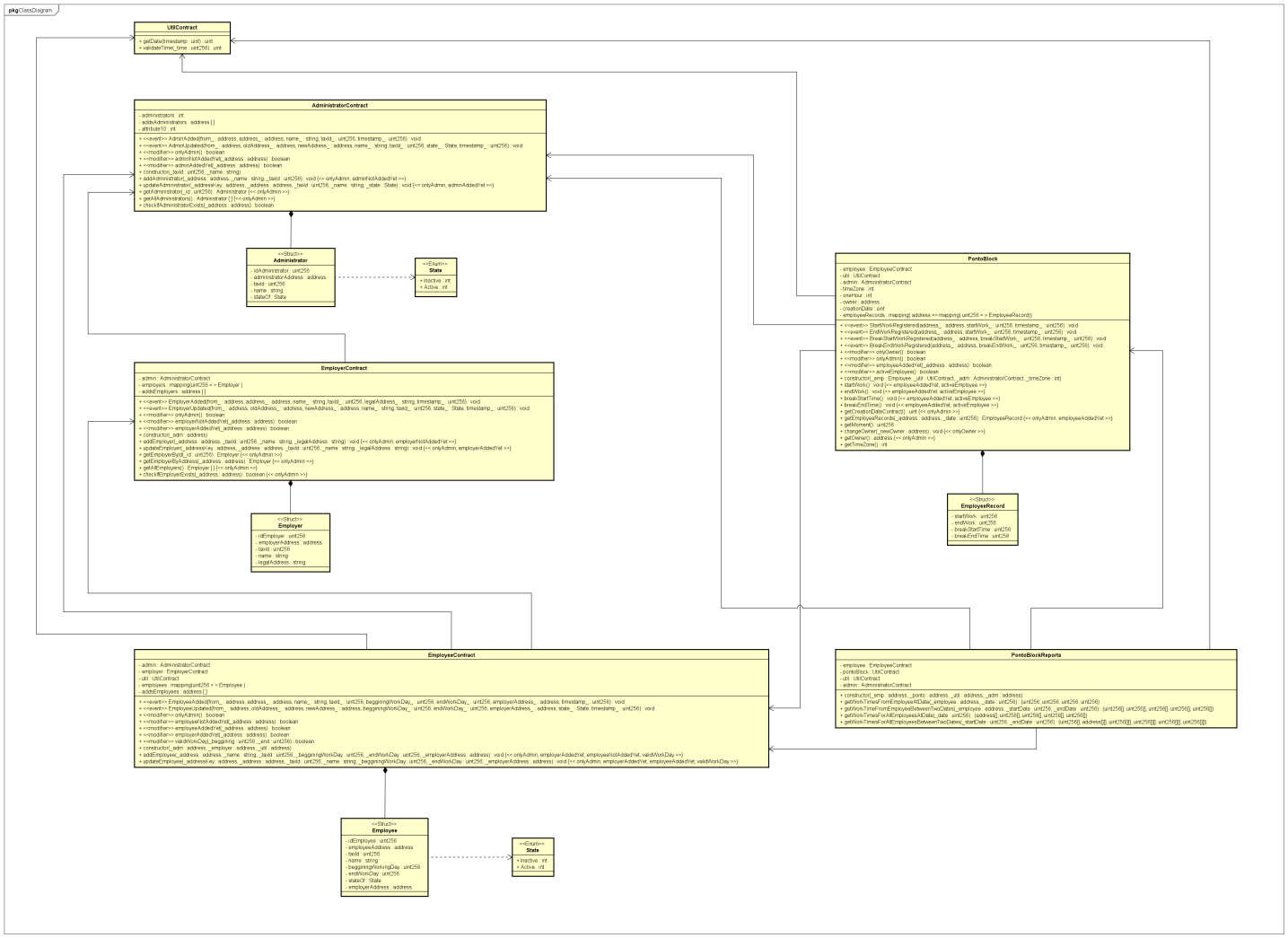
Durante nosso desenvolvimento serão utilizadas algumas nomenclaturas que nos ajudarão a simplificar, organizar e reconhecer os elementos no documento de análise, assim sendo, temos:

**CDXX**: Class Diagram (Diagrama de Classe) – Diagrama de Dlasse. O **XX** representa uma sequência numérica de dois dígitos, sequencial e único para cada diagrama de classe, começando em 00. Exemplo: CD01 – Diagrama de Classe 01.

Para o nosso sistema foram relacionados os seguintes diagramas de classe:

|  |  |  |
| --- | --- | --- |
| CD | NOME | DESCRIÇÃO |
| 00 | Contracts | Relacionamento entre as os Smart Contracts |
| 01 | UtilContract | Smart Contract UtilContract |
| 02 | AdministratorContract | Smart Contract AdministratorContract |
| 03 | EmployerContract | Smart Contract EmployerContract |
| 04 | EmployeeContract | Smart Contract EmployeeContract |
| 05 | PontoBlock | Smart Contract PontoBlock |
| 06 | PontoBlockReports | Smart Contract PontoBlockReports |

### 9.5.1 CD00 - Contracts



**Sumário:** Estrutura geral e relacionamentos entre os Smart Contracts.

**Entidades:**

**UtilContract:** Representação do Smart Contract UtilContract.

**Administrator:** Struct que representa as propriedades do administrador.

**State:** Enum que representa o status, inativo ou ativo.

**EmployerContract:** Representação do Smart Contract EmployerContract.

**Employer:** Struct que representa as propriedades do empregador.

**EmployeeContract:** Representação do Smart Contract EmployeeContract.

**Employee:** Struct que representa as propriedades do funcionário.

**PontoBlock:** Representação do Smart Contract PontoBlock.

**EmployeeRecord:** Struct que representa as propriedades do registro das gravações de ponto.

**PontoBlockReports:** Representação do Smart Contract PontoBlockReports.

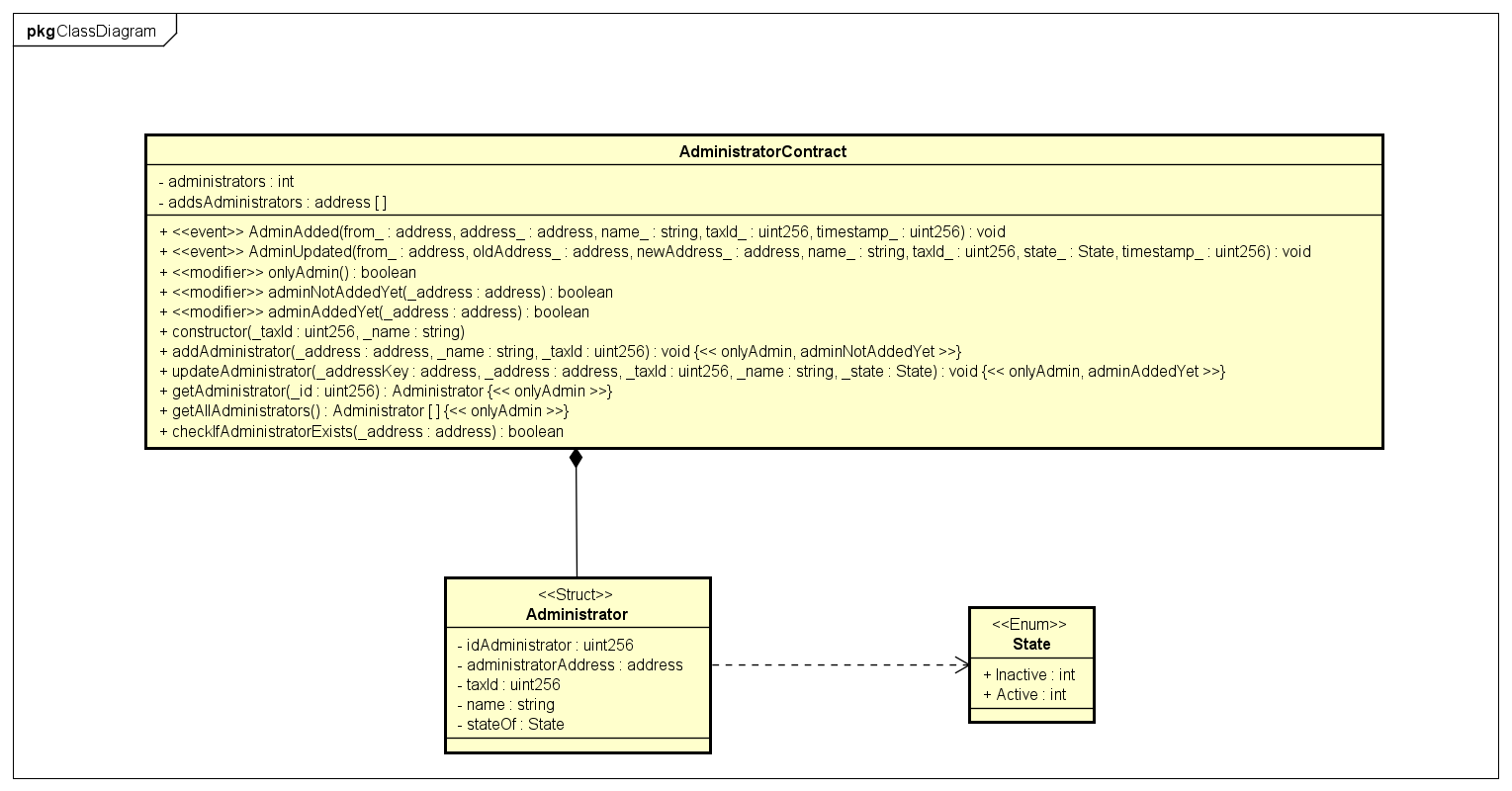
### 9.5.2 CD01 - UtilContract



O primeiro contrato desenvolvido é o **UtilContract**. Este contrato é responsável por executar funções genéricas que não tem uma relação específica a um contrato, mas funções de caráter geral.

As funções neste contrato são **getDate** e **validateTime**. A função **getDate** tem como parâmetro um *timestamp* e retorna um *uint* no formato AAAAMMDD onde, AAAA representa o ano de 4 dígitos, MM representa o ano com 2 dígitos e DD representa o dia com 2 dígitos. Já a função **validateTime** é responsável com base em um *uint256* de entrada no formato hhmm onde, hh representa a hora e mm representa o minuto, retorna um booleano informando se o formato passado é valido ou não.

### 9.5.3 CD02 - AdministratorContract



O próximo contrato desenvolvido é o **AdministratorContract**. Este contrato é responsável manter os administradores do sistema. Vale lembrar que os contratos também são conhecidos por suas contas, assim como os usuários, assim sendo, este contrato guarda as contas (de usuário e de contrato) com perfil administrador. Para as contas de usuário, a referência ao **AdministratorContract** se faz necessária porque há algumas operações como adicionar funcionário, exibir relatório, entre outras, que só podem ser feitas por agente permissionado. E quanto aos contratos, como há relações entre as funções de um contrato com outro, eles também devem ser colocados nesta categoria.

Em sua composição há um Struct para armazenar os dados do administrador com:

* o *id* sendo um *uint256* que guarda um sequencial a partir de zero,
* o endereço do administrador sendo um *address*,
* o taxId representando o documento do administrador sendo um *uint256*,
* o nome, sendo uma *string,*
* o estado, representando a situação do administrador podendo ser inativo ou ativo.

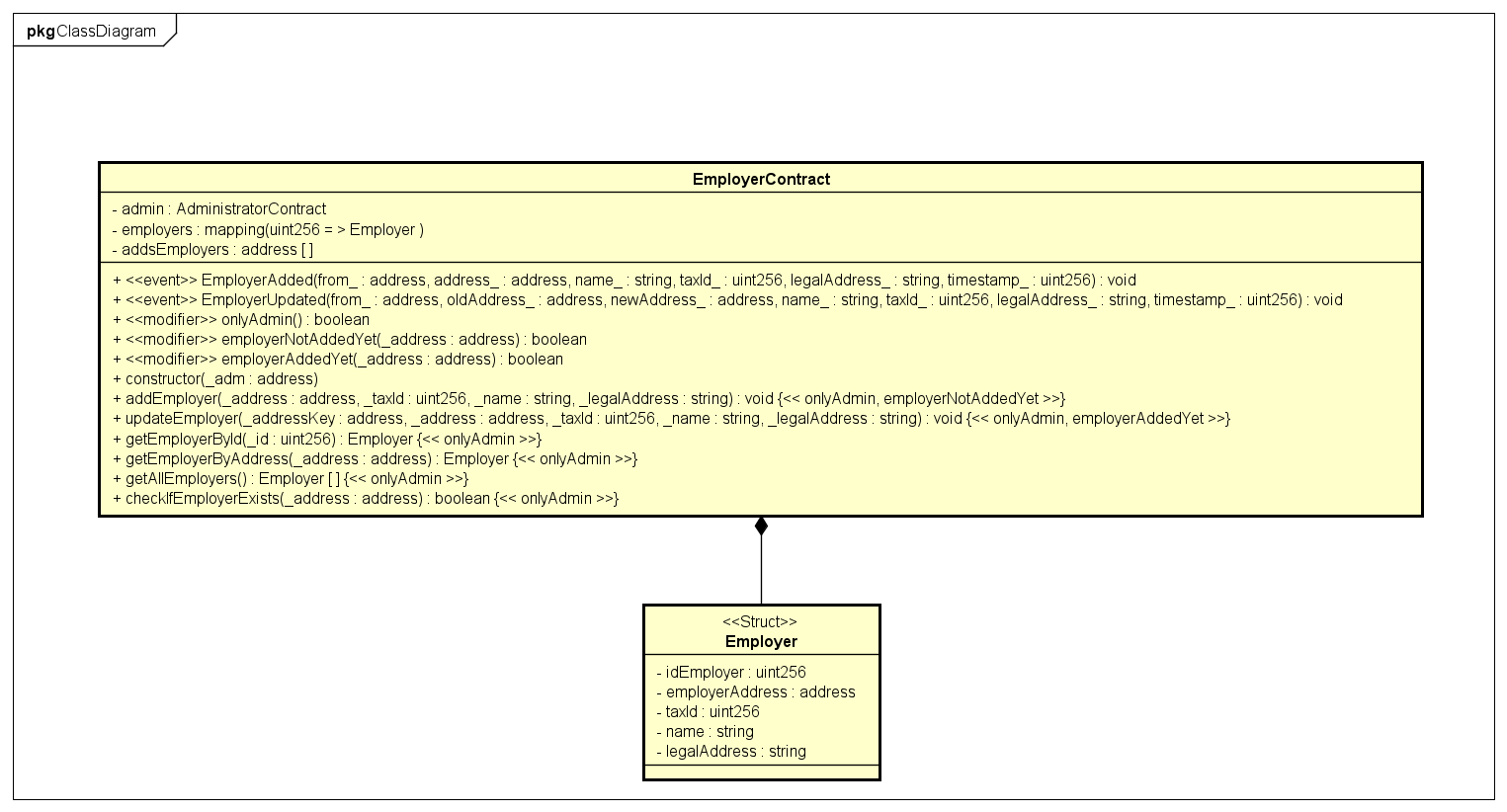
Há também o Enum para representar a situação, Inativo ou Ativo.

Duas variáveis, um array de *address*  e um *mapping* de *uint256* para o Struct de Administrador para armazenar a lista de Administradores.

Quanto aos métodos, há 11 funções neste contrato, sendo duas funções de evento, três funções modificadoras, um construtor e 5 funções de manipulação da entidade, sendo:

* Evento *AdminAded*, sendo uma função especial, tipo evento, que atua como um log onde após um administrador ser adicionado registra esse log com os dados do agente da ação (from\_), o endereço do administrador adicionado (address\_), nome (name\_), documento (taxId\_) e timestamp da gravação (timestamp\_);
* Evento *AdminUpdated*, sendo uma função especial, tipo evento, que atua como um log onde após um administrador ser atualizado, registra esse log com os dados do agente da ação (from\_), o antigo endereço do administrador (oldAddress\_), novo endereço do administrador (newAddress\_), nome atualizado (name\_), documento atualizado (taxId\_), estado (state\_) e timestamp da gravação (timestamp\_);
* Modificador *onlyAdmin*, sendo uma função que faz uma validação verificando se o agente da ação está incluso na coleção de administradores. Seu resultado positivo permite que a função seja executada;
* Modificador *adminNotAddedYet*, sendo uma função que faz uma validação verificando se o endereço fornecido como parâmetro ainda não foi inserido na coleção de administradores. Seu resultado positivo, permite que a função seja executada;
* Modificador *adminAddedYet*, sendo uma função que faz uma validação verificando se o endereço fornecido como parâmetro já foi inserido na coleção de administradores. Seu resultado positivo, permite que a função seja executada;
* O *constructor*, que tem como parâmetros um *uint256* para guardar o documento do administrador e uma *string* para guardar seu nome;
* O *addAdministrator*, que tem como parâmetros um *address*, uma *string* e um *uint256* para armazenar o endereço, nome e documento do administrador, respectivamente;
* O *updateAdministrator*, tendo como parâmetros dois *address* para buscar o administrador e o seu novo endereço, *uint256* para seu documento, *string*, nome, e *State* para situação;
* O *getAdministrator*, que com base em um *uint256* busca um administrador no *mapping* administrators;
* O *getAllAdministrators*, que retorna um array com todos os administradores;
* Por último, *checkIfAdministratorExists*, que com base em um *address* dado, verifica se ele existe no *array* addsAdministrators.

### 9.5.4 CD03 - EmployerContract



O **EmployerContract** é responsável manter os empregadores do sistema.

Em sua composição há um Struct para armazenar os dados do empregador com:

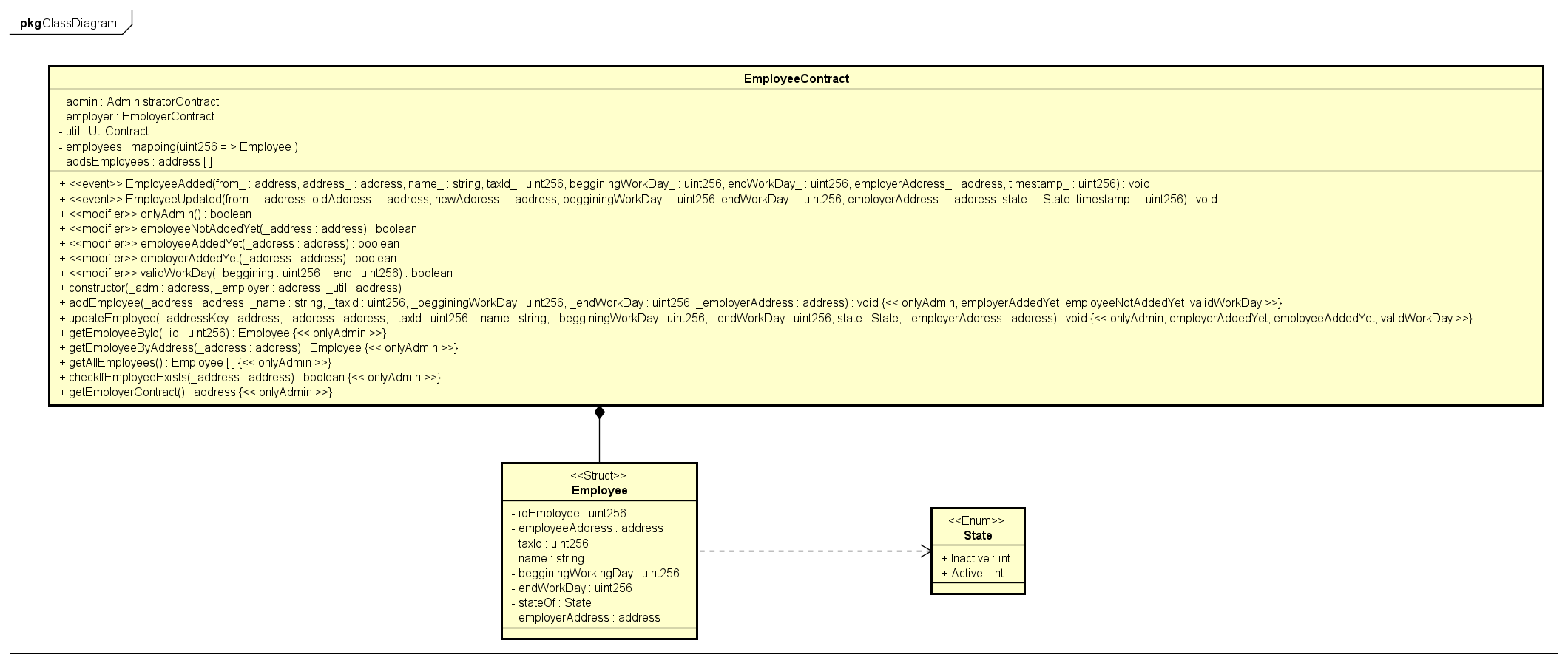
* o *id* sendo um *uint256* que guarda um sequencial a partir de zero,
* o endereço do empregador, sendo um *address*,
* o taxId representando o documento do empregador, sendo um *uint256*,
* o nome, sendo uma *string,*
* o legalAddress, representando o endereço físico de registro do empregador.

Duas variáveis, um array de *address*  e um *mapping* de *uint256* para o Struct de Employer para armazenar a coleção de Employers, e admin, para referenciar o AdministratorContract, compõem as variáveis deste Smart Contract.

Quanto aos métodos, há 12 funções neste contrato, sendo duas funções de evento, três funções modificadoras, um construtor e 6 funções de manipulação da entidade, sendo:

* Evento *EmployerAded*, sendo uma função especial, tipo evento, que atua como um log onde após um empregador ser adicionado, registra esse log com os dados do agente da ação (from\_), o endereço do empregador adicionado (address\_), nome (name\_), documento (taxId\_); endereço legal (legalAddress\_) e timestamp da gravação (timestamp\_);
* Evento *EmployerUpdated*, sendo uma função especial, tipo evento, que atua como um log onde após um empregador ser atualizado, registra esse log com os dados do agente da ação (from\_), o antigo endereço do empregador (oldAddress\_), novo endereço do empregador (newAddress\_), nome atualizado (name\_), documento atualizado (taxId\_), endereço legal (legalAddress\_) e timestamp da gravação (timestamp\_);
* Modificador *onlyAdmin*, sendo uma função que faz uma validação verificando se o agente da ação está incluso na coleção de administradores. Seu resultado positivo permite que a função seja executada;
* Modificador employer*NotAddedYet*, sendo uma função que faz uma validação verificando se o endereço fornecido como parâmetro ainda não foi inserido na coleção de Employers. Seu resultado positivo, permite que a função seja executada;
* Modificador employer*AddedYet*, sendo uma função que faz uma validação verificando se o endereço fornecido como parâmetro já foi inserido na coleção de Employers. Seu resultado positivo, permite que a função seja executada;
* O *constructor*, que tem como parâmetros um *address* para referenciar o endereço do contrato AdministratorContract;
* O *addEmployer*, que tem como parâmetros um *address*, um *uint256, duas* *string* para armazenar o endereço, documento, nome e endereço legal do empregador, respectivamente;
* O *updateEmployer*, tendo como parâmetros dois *address* para buscar o administrador e o seu novo endereço, *uint256* para seu documento e duas *string* para o nome e endereço legal, atuando na atualização dos dados do empregador;
* O *getEmployerById*, que com base em um *uint256* busca um empregador na coleção de Employers;
* O *getEmployerByAddress*, que com base em um *address* busca um empregador na coleção de Employers;
* O *getAllEmployers*, que retorna um array com todos os empregadores;
* Por último, *checkIfEmployerExists*, que com base em um *address* dado, verifica se ele existe no *array* addsEmployers.

### 9.5.5 CD04 - EmployeeContract



O **EmployeeContract** é responsável manter os funcionários do sistema.

Em sua composição há um Struct para armazenar os dados do funcionário com:

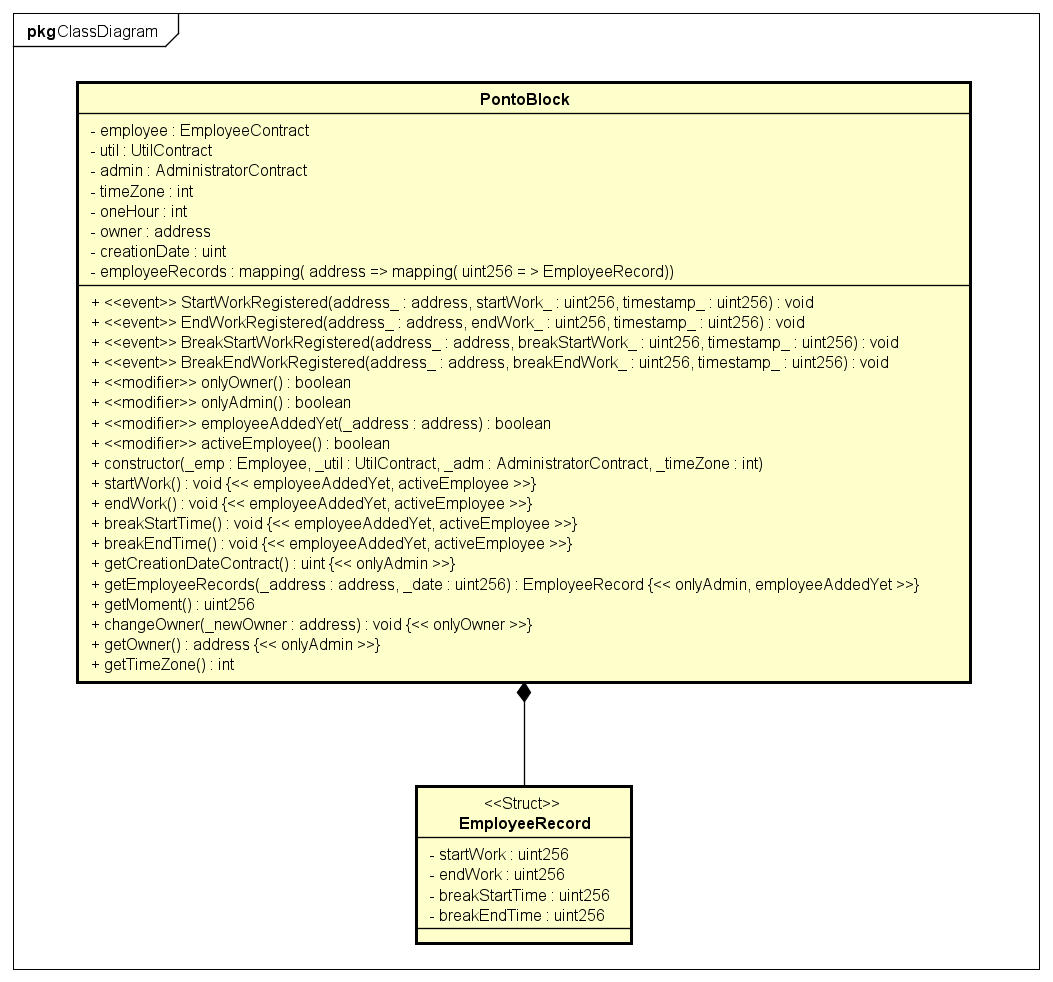
* o *id* sendo um *uint256* que guarda um sequencial a partir de zero,
* o endereço do funcionário, sendo um *address*,
* o taxId representando o documento do funcionário, sendo um *uint256*,
* o nome, sendo uma *string,*
* o início da jornada de trabalho, sendo um *uint256*,
* o fim da jornada de trabalho, sendo um *uint256*,
* O estado, inativo ou ativo,
* Endereço do empregador, sendo um *address* representando o endereço do empregador.

Cinco variáveis, um array de *address*  e um *mapping* de *uint256* para o Struct de Employee para armazenar a coleção de Employees, admin, para referenciar o AdministratorContract, employer, para referenciar o EmployerContract, e util, para referenciar o UtilContract, compõem as variáveis deste Smart Contract.

Quanto aos métodos, há 15 funções neste contrato, sendo duas funções de evento, cinco funções modificadoras, um construtor e 7 funções de manipulação da entidade, sendo:

* Evento *EmployeeAded*, sendo uma função especial, tipo evento, que atua como um log onde após um funcionário ser adicionado, registra esse log com os dados do agente da ação (from\_), o endereço do funcionário adicionado (address\_), nome (name\_), documento (taxId\_); início da jornada de trabalho (begginingWorkDay\_), fim da jornada (endWorkDay\_), endereço do empregador (employerAddress\_) e timestamp da gravação (timestamp\_);
* Evento *EmployeeUpdated*, sendo uma função especial, tipo evento, que atua como um log onde após um funcionário ser atualizado, registra esse log com os dados do agente da ação (from\_), o antigo endereço do funcionário (oldAddress\_), novo endereço do funcionário (newAddress\_), nome atualizado (name\_), documento atualizado (taxId\_), início da jornada de trabalho (begginingWorkDay\_), fim da jornada (endWorkDay\_), endereço do empregador (employerAddress\_), estado (state\_) e timestamp da gravação (timestamp\_);
* Modificador *onlyAdmin*, sendo uma função que faz uma validação verificando se o agente da ação está incluso na coleção de administradores. Seu resultado positivo permite que a função seja executada;
* Modificador employee*NotAddedYet*, sendo uma função que faz uma validação verificando se o endereço fornecido como parâmetro ainda não foi inserido na coleção de Employees. Seu resultado positivo, permite que a função seja executada;
* Modificador employee*AddedYet*, sendo uma função que faz uma validação verificando se o endereço fornecido como parâmetro já foi inserido na coleção de Employees. Seu resultado positivo, permite que a função seja executada;
* Modificador employer*AddedYet*, sendo uma função que faz uma validação verificando se o endereço fornecido como parâmetro já foi inserido na coleção de Employers. Seu resultado positivo, permite que a função seja executada;
* Modificador *validWorkDay*, sendo uma função que faz uma validação verificando se os dois parâmetros \_beginning e \_end compõem um horário válido. Seu resultado positivo, permite que a função seja executada;
* O *constructor*, que tem como parâmetros três *address* para referenciar o endereço dos contratos AdministratorContract, EmployerContract e UtilContract;
* O *addEmployee*, que tem como parâmetros dois *address*, uma *string* e três *uint256* para armazenar o endereço do funcionário e do empregador, nome, documento, início e fim da jornada de trabalho do funcionário;
* O *updateEmployee*, tendo como parâmetros dois *address* para buscar o funcionário e o seu novo endereço, *uint256* para seu documento, uma string para o nome, dois *uint256* para início e fim da jornada de trabalho, um *address* para endereço do empregador e *State* para o estado, atuando na atualização dos dados do funcionário;
* O *getEmployeeById*, que com base em um *uint256* busca um funcionário na coleção de Employees;
* O *getEmployeeByAddress*, que com base em um *address* busca um funcionário na coleção de Employees;
* O *getAllEmployees*, que retorna um array com todos os funcionários;
* O *checkIfEmployeeExists*, que com base em um *address* dado, verifica se ele existe no *array* addsEmployees;
* *getEmployerContract*, que fornece o endereço do EmployerContract.

### 9.5.6 CD05 - PontoBlock



O **PontoBlock** é responsável manter os registros de marcação de ponto dos funcionários.

Em sua composição há um Struct para armazenar os dados da marcação de ponto com:

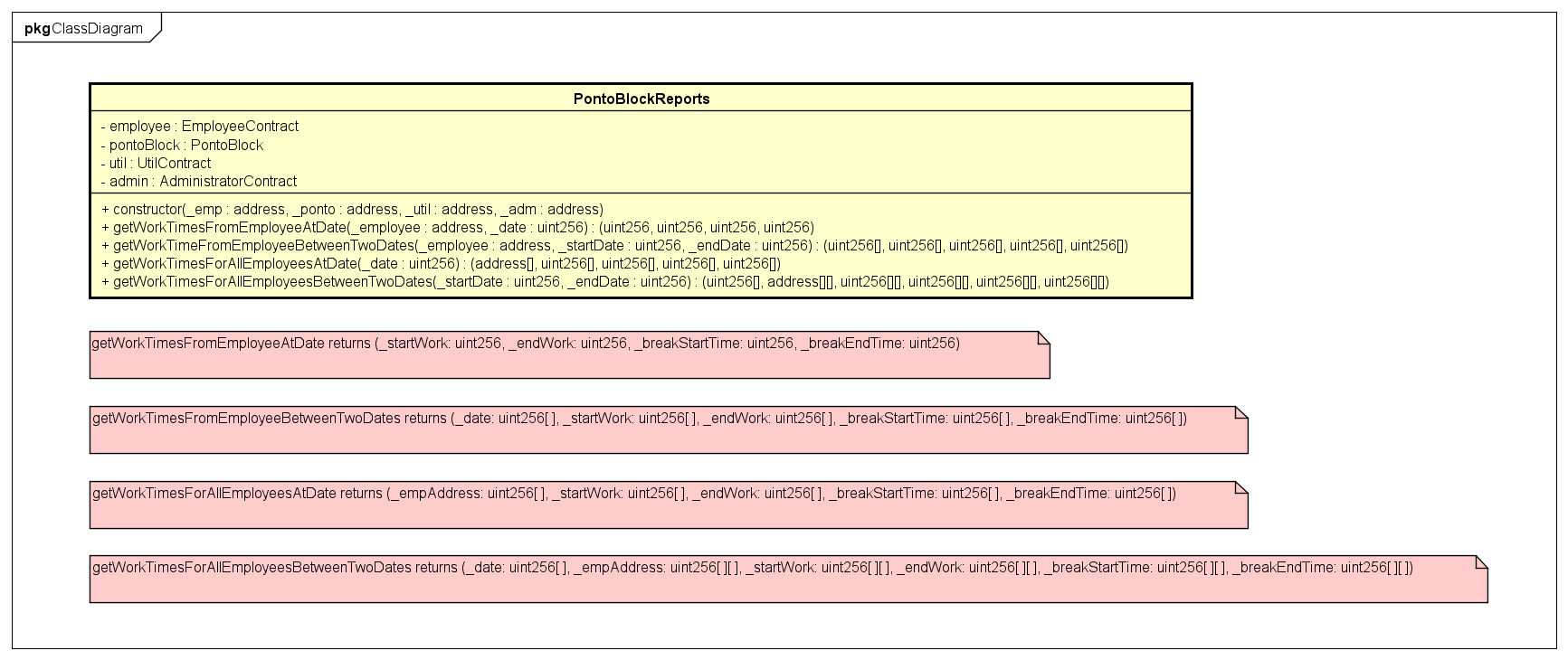
* o startWork, sendo um *uint256* que guarda o timestamp da hora da marcação de ponto do início da jornada,
* o endWork, sendo um *uint256* que guarda o timestamp da hora da marcação de ponto do fim da jornada,
* o breakStartTime, sendo um *uint256* que guarda o timestamp da hora da marcação de ponto do início da pausa,
* o breakEndTime, sendo um *uint256* que guarda o timestamp da hora da marcação de ponto do fim da pausa.

Oito variáveis, um mapping de *address* de Employee que aponta para um mapping de *uint256* para armazenar uma data (YYYYMMDD) que aponta para o Struct EmployeeRecord, um *uint* para armazenar a data de criação do contrato, um *address* para o endereço do dono do contrato, *int* para a representação de 1h, *int* para o Time Zone, admin, util e employee para a referência dos contractos AdministratorContract, UtilContract e EmployeeContract, respectivamente, compõem as variáveis deste Smart Contract.

Quanto aos métodos, há 19 funções neste contrato, sendo quatro funções de evento, quatro funções modificadoras, um construtor e 10 funções de manipulação da entidade, sendo:

* Evento *StartWorkRegistered*, sendo uma função especial, tipo evento, que atua como um log onde após a marcação de início de jornada, registra esse log com os dados do funcionário (address\_), timestamp da marcação (startWork\_) e timestamp da gravação (timestamp\_);
* Evento *EndWorkRegistered*, sendo uma função especial, tipo evento, que atua como um log onde após a marcação de fim de jornada, registra esse log com os dados do funcionário (address\_), timestamp da marcação (endWork\_) e timestamp da gravação (timestamp\_);
* Evento *BreakStartWorkRegistered*, sendo uma função especial, tipo evento, que atua como um log onde após a marcação de início da pausa, registra esse log com os dados do funcionário (address\_), timestamp da marcação (breakStartWork\_) e timestamp da gravação (timestamp\_);
* Evento *BreakEndWorkRegistered*, sendo uma função especial, tipo evento, que atua como um log onde após a marcação de fim da pausa, registra esse log com os dados do funcionário (address\_), timestamp da marcação (breakEndWork\_) e timestamp da gravação (timestamp\_);
* Modificador *onlyOwner*, sendo uma função que faz uma validação verificando se o agente da ação é o dono do contrato. Seu resultado positivo permite que a função seja executada;
* Modificador *onlyAdmin*, sendo uma função que faz uma validação verificando se o agente da ação está incluso na coleção de administradores. Seu resultado positivo permite que a função seja executada;
* Modificador employee*AddedYet*, sendo uma função que faz uma validação verificando se o endereço fornecido como parâmetro já foi inserido na coleção de Employees. Seu resultado positivo, permite que a função seja executada;
* O *constructor*, que tem como parâmetros três *address* para referenciar o endereço dos contratos AdministratorContract, EmployeeContract e UtilContract e um *int* para referenciar o Time Zone do contrato;
* O *startWork* é a função que faz o registro da marcação do início da jornada;
* O *endWork* é a função que faz o registro da marcação de fim da jornada;
* O *breakStartTime* é a função que faz o registro de marcação de início da pausa;
* O *breakEndTime* é a função que faz o registro de marcação de fim da pausa;
* O *getCreationDateContract* fornece a data em que o contrato foi criado;
* O *getEmployeeRecords*, que com base em um *address* e um *uint256* dados, retorna a Struct EmployeeRecord;
* *getMoment*, que fornece o endereço o timestamp ajustado com o Time Zone;
* *changeOwner*, fornece o endereço do dono do contrato;
* E por último, *getTimeZone*, que retorna um *int* para o Time Zone.

### 9.5.7 CD06 - PontoBlockRecords



O **PontoBlockReports** é responsável retornar os registros das marcações de ponto com base na pesquisa por endereços dos funcionários e a data.

* Em sua composição há quatro variáveis para referenciar os contratos EmployeeContract, PontoBlock, UtilContract e AdministratorContract.

Quanto aos métodos, há 5 funções neste contrato, sendo um construtor e 4 funções de manipulação da entidade, sendo:

* O *constructor*, que tem como parâmetros quatro *address* para referenciar o endereço dos contratos AdministratorContract, EmployeeContract e UtilContract e PontoBlock;
* O *getWorkTimesFromEmployeeAtDate*, que tem como parâmetros um *address* para referenciar o endereço de um funcionário e um *uint256* para referenciar uma data e retorna quatro *uint256* para representar o timestamp para cada uma das marcações de ponto do funcionário no dia;
* O *getWorkTimesFromEmployeeBetweenTwoDates*, que tem como parâmetros um *address* para referenciar o endereço de um funcionário e dois *uint256* para referenciar uma data início e uma data fim e retorna cinco array de *uint256* para representar o dia e o timestamp para cada uma das marcações de ponto do funcionário no intervalo de dias;
* O *getWorkTimesForAllEmployeesAtDate*, que tem como parâmetros um *uint256* para referenciar uma data e retorna um array de *address* e quatro array de *uint256* para representar os endereços dos funcionários os timestamps das marcações de ponto do dia;
* O *getWorkTimesForAllEmployeesBetweenTwoDates*, que tem como parâmetros dois *uint256* para referenciar uma data início e uma data fim e retorna um array de *uint256* para representar o dia, uma matriz *address* de duas dimensões para representar os endereços no dia, e quatro matriz de *uint256* de duas dimensões para representar o timestamp de cada uma das marcações de ponto do funcionário no intervalo de dias.

## 9.6 DIAGRAMA DE SEQUÊNCIA

O diagrama de sequência mostra a troca de informações entre os objetos do sistema e como nossa solução tem uma estrutura particular com solidity, considero importante representar a relação entre as estruturas, muito em especial por conta da participação das funções auxiliares modificadoras que validam algumas ações e eventos que registram os logs.

## 9.7 CONSIDERAÇÕES NO DESENVOLVIMENTO COM SOLIDITY

O core de nossa solução está representada nos smart contracts, que são os elementos responsáveis pela gestão dos recursos da aplicação, fazendo com que todos os dados sejam registrados na blockchain e posteriormente lidos e modificados.

Algumas considerações iniciais devem ser apresentadas quando lidamos com uma solução em blockchain utilizando a linguagem **Solidity**. A primeira delas é que as operações de leitura e gravação de dados da blockchain tem um custo e a otimização do código influencia diretamente no custo destas operações. Se tomarmos com exemplo um determinado valor para representar uma data com dia mês e ano, poderíamos ter três propostas para tal:

1. **Representar a data propriamente dita:** no caso da linguagem **Solidity**, como não possui os tipos **Date**, **Time** e **DateTime**, representá-la com o Unix Timestamp utilizando o tipo básico da linguagem **uint**.
2. **Representar a data como string:** neste caso, a linguagem possui o tipo string, então poderíamos representá-la com o formato **2023/03/23**.
3. **Representar a data como uint:** aqui, seria um misto das duas anteriores, reresentando a data em formato como string mas utilizando **uint**, ficando **20230323.**

Todas as três propostas acima conseguiriam representar a nossa data de exemplo, mas como mencionado, a **EVM** tem o fator de custo das operações na blockchain, mas para efeito de custo a opção utilizando *uint*é a menos custosa, fazendo então que descartemos a opção 2. Com as opções 1 e 3 sobre a mesa resta a dúvida sobre qual utilizar, e em nossa proposta optamos pela opção 3 pois gravando a data nesse formato conseguimos utilizar uma um tipo básico da linguagem menos custoso para a **EVM** e já temos uma espécie de formatação, algo que otimiza algumas de nossas operações, fazendo também mais uma opção pela operação menos custosa pois se utilizássemos o timestamp, em alguns cenários teríamos que fazer algumas conversões para comparar datas.

Na mesma visão de otimização de performance dos smart contracts, há uma pequena diferenciação com relação a uma classe tradicional do modelo de orientação a objetos. Enquanto as classes do modelo OO é comum serem divididas em atributos e métodos, nos smart contracts também podemos fazer esse mesmo tipo de divisão, mas para sua operação, se agruparmos as propriedades em um **Struct**, sua operação fica menos custosa para a EVM, daí, termos um **Struct** para representar um funcionário e não seus atributos discretos como nome, endereço e documento dentro do smart contract.

Outro detalhe importante é que, no contexto de desenvolvimento utilizando smart contracts, a relação entre os contratos se diferencia do tradicional modelo de orientação à objetos, onde as relações entre contrato funcionam de forma parecida como uma relação entre interfaces. Isto posto, começo, abaixo, a descrição da modelagem do sistema para cada um dos contratos.

O desenvolvimento de smart contracts tem um paradigma muito particular para seu desenvolvimento.... Falar sobre o require

## 9.8 ARQUITETURA DO SISTEMA

Após ultrapassadas as etapas de conhecimento dos aspectos legais, soluções atuais de mercado, perspectivas para o desenvolvimento de uma solução com uma arquitetura inovadora como o blockchain e modelagem de regras de negócio, requisitos, processos e modelagem, começaremos a criar as definições para desenvolvimento do software para a solução de sistema de controle de ponto proposta.

Nossa proposta apresenta duas camadas bem distintas à saber:

1. **Camada de Contratos:** Camada onde estão os smart contracts e todos os recursos para testes unitários e publicação dos mesmos na blockchain.
2. **Camada Web:** Camada onde está a visão do usuário responsável pela interação entre o usuário e os contratos.

Na camada de contratos é utilizada a ferramenta **HardHat** para fazer compilação, testes, publicação e verificação dos contratos na rede. Já na camada web a tecnologia base é o framework React para interação do usuário.

Entre as particularidades de um desenvolvimento de uma solução voltada para a tecnologia blockchain é a utilização de uma linguagem de programação particular para este fim, neste caso, **Solidity**. Esta linguagem tem inspiração no **TypeScript** mas tem suas características e tipos próprios. Associados a ela há diversas bibliotecas que ajudam no seu desenvolvimento, em especial com a tecnologia **Node.**

# 10 TECNOLOGIA UTILIZADA NO APLICATIVO

Desenvolvimento da obra desde a modelagem do sistema com diagramas em UML até a distribuição das soluções arquiteturais e de implantação utilizadas em back e frontend.

A solução proposta, como um projeto inicial, tem plenas condições de se adequar, fazendo pequenos ajustes, para que se enquadre nas regras estabelecidas pela portaria do MTE .

# 11 POTENCIALIDADES DE MERCADO

Na proposta que expomos aqui, entendemos que seria muito interessante um modelo onde a cobrança seria feita contemplando duas etapas:

1. Implantação do contrato inteligente e toda a infraestrutura necessária para iniciar o serviço do cliente com publicação dos smart contracts e dos serviços de consumo da aplicação para gestão e utilização do sistema.
2. Uma cobrança por cada registro feito no sistema (entrada, saída, início da pausa e fim da pausa).

Com levantamento inicial utilizando ambiente de teste para desenvolvimento da aplicação conseguimos custos bem interessantes tanto para a publicação do smart contract na rede de testes MUMBAI da POLYGON, como para a utilização das funcionalidade. Considerando a cotação da moeda nativa da rede POLYGON, MATIC para Real Brasileiro (BRL) a:

**1 MATIC = 5,77 BRL**

Temos o seguinte quadro:

|  |  |  |
| --- | --- | --- |
| DEPLOY | | |
| CONTRATO | MATIC | BRL |
| UTIL | 0,0009 | 0,005193 |
| ADMINISTRATOR | 0,0041 | 0,023657 |
| EMPLOYER | 0,0042 | 0,024234 |
| EMPLOYEE | 0,0042 | 0,024234 |
| PONTO BLOCK | 0,0048 | 0,027696 |
| PONTO BLOCK REPORTS | 0,0052 | 0,030004 |
| **TOTAL** | **0,0234** | **0,135018** |

|  |  |  |
| --- | --- | --- |
| UTILIZAÇÃO DO SISTEMA | | |
| MARCAÇÕES DE PONTO | MATIC | BRL |
| INÍCIO DE JORNADA | 0,0002 | 0,001154 |
| INÍCIO DE PAUSA | 0,0003 | 0,001731 |
| FIM DA PAUSA | 0,0005 | 0,002885 |
| FIM DA JORNADA | 0,0006 | 0,003462 |
| **TOTAL** | **0,0016** | **0,009232** |

|  |  |  |
| --- | --- | --- |
| EVOLUÇÃO DE CUSTOS | | |
|  | MATIC | BRL |
| CUSTO DIÁRIO POR FUNCIONÁRIO | 0,0016 | 0,009232 |
| FUNCIONÁRIO MÊS (22 DIAS) | 0,0352 | 0,203104 |
| 100 FUNCIONÁRIOS MÊS | 3,52 | 20,3104 |
| 100 FUNCIONÁRIOS ANO | 42,24 | 243,7248 |

|  |  |
| --- | --- |
| PREVISÃO DE FATURAMENTO | BRL |
| EMPRESA COM 100 FUNCIONÁRIOS (ASSINATURA MENSAL) | 180,00 |
| FATURAMENTO ANUAL | 2.160,00 |
| CUSTO ANUAL | - 243,72 |
| SALDO ANUAL | 1.916,28 |

De acordo com as tabelas acima, notamos custos bem competitivos para a implantação do sistema. No cenário proposto, num universo de 100 funcionários em uma empresa conseguimos o resultado de R$ 1.916,28.

# 12 CRONOGRAMA DA PESQUISA

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Etapas | 2022 | | | | | | |
|  | jul | ago | set | out | nov | dez |  |
| Entrega do projeto ao professor Orientador |  |  |  |  |  |  |  |
| Leitura e fichamento da literatura |  |  |  |  |  |  |  |
| Analise crítica do material |  |  |  |  |  |  |  |
| Revisão de literatura concluída |  |  |  |  |  |  |  |
| Construção do aplicativo |  |  |  |  |  |  |  |
| Entrega do aplicativo |  |  |  |  |  |  |  |
| Discussão dos resultados |  |  |  |  |  |  |  |
| Entrega da monografia ou artigo |  |  |  |  |  |  |  |
| Entrega do trabalho a banca 15 antes da apresentação |  |  |  |  |  |  |  |
| Apresentação publica |  |  |  |  |  |  |  |

# 13 REFERÊNCIAS

**ARRIAL**, T. **Estudo Técnicos** - junho de 2018. **Confederação Nacional de Municı́pios, 2018**. Disponı́vel em: <https://www.cnm.org.br/cms/biblioteca/Eleitorado-2018.pdf>. Acesso em: 20 de junho de 2022.

**Assessoria Detran e PJC – MT**. **Servidores do Detran são presos por fraudes e falsificação de documentos**. Governo de Mato Grosso. 28 de junho de 2016. Disponível em: <http://www.mt.gov.br/-/4395709-servidores-do-detran-sao-presos-por-fraudes-e-falsificacao-de-documentos>. Acesso em: 21 de maio de 2022.

**ATTARAN,** Mohsen e **GUNASEKARAN,** Angappa. **Applications of Blockchain Technology in Business - Challenges and Opportunities**. p. 14. Texas: Springer, 2019.

**BISPO**, Fábio. **Participação de servidores em falsificação de carros de luxo é investigada.** Terra. 17 de julho de 2019. Disponível em: <https://www.terra.com.br/noticias/brasil/cidades/participacao-de-servidores-em-falsificacao-de-carros-de-luxo-e-investigada,1e380bed31b2f7717ef585719af9365djt0n7aq9.html>. Acesso em 21 de maio de 2022.

**BITCOIN**. **What is Bitcoin**. Disponível em: <https://bitcoin.org/en/faq#general>. Acesso em: 21 de maio de 2022.

**CARVALHO,** Carla Arigony de; **ÁVILA**, Lucas Veiga. **A TECNOLOGIA BLOCKCHAIN APLICADA AOS CONTRATOS INTELIGENTES.** Revista Em Tempo, [S.l.], v. 18, n. 01, p. 156 - 176, dec. 2019. ISSN 1984-7858. Disponível em: <https://revista.univem.edu.br/emtempo/article/view/3210>. Acesso em: 19 june 2022.

**CLEAR SALE.** **Com aumento na procura por carros, setor automotivo se destaca no Mapa da Fraude 2021.** 02 de fevereiro de 2022. Disponível em: <https://www.ecommercebrasil.com.br/noticias/setor-automotivo-mapa-da-fraude-2021/>. Acesso em: 26 de março de 2022.

**CRUZ,** Elaine Patrícia**. PF e PRF agem contra quadrilha que clonou viaturas do Exército**. 24 de fevereiro de 2022. Disponível em: <https://agenciabrasil.ebc.com.br/geral/noticia/2022-02/pf-e-prf-agem-contra-quadrilha-que-clonou-viaturas-do-exercito>. Acesso em: 26 de março de 2022.

**CUNHA**, S. S. d. et al. **Relatório sobre o Sistema Brasileiro de Votação Eletrônica.** COMITÊ MULTIDISCIPLINAR INDEPENDENTE, 2014. Disponível em: <http://www.brunazo.eng.br/voto-e/textos/CMind-1-Brasil-2010.pdf>. Acesso em 20 de junho de 2022.

**DAVIDSON**, Al. **Increasing trust in criminal evidence with blockchains. MOJ Digital & Technology.** 2 de novembro de 2017. Disponível em: <https://mojdigital.blog.gov.uk/2017/11/02/increasing-trust-in-criminal-evidence-with-blockchains/>. Acesso em: 21 de maio de 2022.

**ENCCLA-2020**. **Estratégia Nacional de Combate à Corrupção e Lavagem de Dinheiro. Blockchain no setor público: Guia de conceitos e usos potenciais.** Agosto de 2020. Disponível em: <http://enccla.camara.leg.br/acoes/arquivos/resultados-enccla-2020/blockchain-no-setor-publico-guia-de-conceitos-e-usos-potenciais>. Acesso em: 21 de maio de 2022.

**ESCOBAR,** Matheus Garcia. **Contextualização e Introdução ao Blockchain**. 2021. Disponível em: <https://www.ufsm.br/pet/sistemas-de-informacao/2021/11/29/contextualizacao-e-introducao-ao-blockchain/>. Acesso em: 22 de maio de 2022.

**ETHEREUM.**  Disponível em: <https://ethereum.org/pt-br/>. Acesso em: 21 de maio de 2022.

**FACHINI**, Tiago. Projuris. Disponível em: <https://www.projuris.com.br/smart-contract/>. Acesso em: 21 de maio de 2022.

**KUNTZ,** João. **Blockchain Ethereum – Fundamentos de Arquitetura, desenvolvimento de contratos e aplicações.** São Paulo: Casa do Código, 2022.

**LARA**, Walace. **Justiça afasta 85 servidores do Detran de SP suspeitos de participar de fraude que clonava veículos do Exército.** G1. 24 de fevereiro de 2022. Disponível em:<https://g1.globo.com/sp/sao-paulo/noticia/2022/02/24/justica-afasta-85-servidores-do-detran-de-sp-suspeitos-de-participar-de-fraude-que-clonava-veiculos-do-exercito.ghtml>. Acesso em: 21 de maio de 2022.

**MORAIS**, Anderson Melo de, **LINS**, Fernado Antonio Aires. **Uso de Blockchain na Educação: Estado da arte e desafios em aberto**. 2020. Disponível em: <https://www.nucleodoconhecimento.com.br/tecnologia/uso-de-blockchain>. Acesso em: 03 de abril de 2022.

**NAKAMOTO,** Satoshi. Bitcoin: **A Peer-to-Peer Eletronic Cash System**. 2008 Bitcoin.org. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Acesso em: 21 de maio de 2022.

**NIWA**, Henrique; "**UM SISTEMA DE VOTO ELETRÔNICO BASEADO EM BLOCKCHAIN**", p. 2877-2893 . In: Anais do XIX Simpósio de Pesquisa Operacional & Logística da Marinha. São Paulo: Blucher, 2020. Disponível em: <https://www.proceedings.blucher.com.br/article-details/um-sistema-de-voto-eletrnico-baseado-em-blockchain-34623>. Acesso em: 20 de junho de 2022.

**O,** S. K., **XU**, X., **CHIAM**, Y. K. and **LU**, Q., "**Evaluating Suitability of Applying Blockchain**," 2017 22nd International Conference on Engineering of Complex Computer Systems (ICECCS), 2017, pp. 158-161, doi: 10.1109/ICECCS.2017.26. Disponível em <https://sci-hub.se/10.1109/ICECCS.2017.26>. Acesso em 18 de maio de 2022.

**OLIVEIRA**, F. R.; **MAZIERO**, R. C.; **ARAÚJO**, L. S. de. U**M ESTUDO SOBRE A WEB 3.0: evolução, conceitos, princípios, benefícios e impactos.** Revista Interface Tecnológica, [S. l.], v. 15, n. 2, p. 60–71, 2018. DOI: 10.31510/infa.v15i2.492. Disponível em: <https://revista.fatectq.edu.br/interfacetecnologica/article/view/492>. Acesso em: 8 jun. 2022.

**RAMALHO,** R. **Prazo para eleitor pedir voto em trânsito termina nesta quinta.** 2018. Disponı́vel em: <https://g1.globo.com/politica/eleicoes/2018/noticia/2018/08/22/termina-nesta-quinta-feira-prazo-para-eleitor-pedir-voto-em-transito.ghtml>. Acesso em: 20 de junho de 2022.

**RODRIGUES**, Carlo Kleber da Silva. **Uma análise simples de eficiência e segurança da Tecnologia Blockchain. Revista de Sistemas e Computação**, Salvador, v. 7, n. 2, p. 147-162, jul./dez. 2017. Disponível em: <https://repositorio.uniceub.br/jspui/bitstream/235/11373/1/Uma%20an%c3%a1lise%20simples%20de%20efici%c3%aancia%20e%20seguran%c3%a7a%20da%20Tecnologia%20Blockchain.pdf>. Acesso em: 31 de julho de 2022.

**RODRIGUES**, Galtiery. **Maior adulterador de chassis do Brasil é preso em Goiás, diz polícia**. Metrópoles. 25 de março de 2022. Disponível em:<https://www.metropoles.com/brasil/maior-adulterador-de-chassis-do-brasil-e-preso-em-goias-diz-policia>. Acesso em: 21 de maio de 2022.

**SCHULTZ**, Felix. **Entenda 5 vantagens do Blockchain para empresas**. Milvus. 11 de janeiro de 2019. Disponível em: <https://blog.milvus.com.br/entenda-5-vantagens-do-blockchain-para-as-empresas/>. Acesso em: 21 de maio de 2022.

**SILVA**, Euber Chaia Cotta e, e **MARQUES**, Rodrigo Moreno. **Blockchain no setor público: uma revisão sistemática de literatura**. AtoZ:novaspráticaseminformaçãoeconhecimento,10(3),1-11,set./dez.2021. Disponível em: <https://revistas.ufpr.br/atoz/article/view/79903/44241>. Acesso em: 21 de maio de 2022.

**STALLINGS,** William. **Criptografia e segurança de redes**. p. 181. 4. ed. São Paulo: Pearson Prentice Hall, 2008.

**TSE.** **Eleições 2018: confira as datas do calendário eleitoral**. 2018. Disponı́vel em: <https://g1.globo.com/politica/eleicoes/2018/noticia/eleicoes-2018-datas.ghtml>. Acesso em: 20 de junho de 2022.

**TAKENOBU, T. – Ethereum EVM Illustrated**. Disponível em: <https://takenobu-hs.github.io/downloads/ethereum\_evm\_illustrated.pdf>. Acessado em: 11 de março de 2023.

**TEKISALP,** Emre. **Understanding Web3 – A User Controlled Internet.** Disponível em: <https://www.coinbase.com/blog/understanding-web-3-a-user-controlled-internet>. Acessado em 10 de março de 2023.

**UML – UNIFIED MODELING LANGUAGE**. Disponível em: <https://www.uml.org/what-is-uml.htm>. Acesso em: 19 de junho de 2022.

**WÜST**, Karl e **GERVAIS**, Arthur. **Do you need a blockchain?** 2018. Disponível em: <https://www.law.berkeley.edu/wp-content/uploads/2018/08/Do-you-need-a-Blockchain-Karl-Wust-and-Arthur-Gervais.pdf>. Acesso em: 21 de maio de 2022.

**ZAGO,** Mateo. **Why the Web 3.0 Matters and you should know about it.** 2018. Disponível em: <https://medium.com/@Matzago/why-the-web-3-0-matters-and-you-should-know-about-it-a5851d63c949>. Acesso em: 08 de junho de 2022.

**ZHENG,** ZIBIN, **XIE**, Shaoan, **DAI**, Hongning, **CHEN**, Xiangping, **WANG**, Huaimin, "**An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends**," *2017 IEEE International Congress on Big Data (BigData Congress)*, 2017, pp. 557-564, doi: 10.1109/BigDataCongress.2017.85.

1. **Lei de Liberdade Ecoômica – Lei Nº 13.874 de 20 de setembro de 2019, Art. 74 parágrafo segundo**. Disponível em: <http://www.planalto.gov.br/ccivil\_03/\_ato2019-2022/2019/lei/L13874.htm>. Acessado em: 10 de março de 2023. [↑](#footnote-ref-0)
2. **Portaria Nº 671/2021 MTP**. Disponível em: <https://www.in.gov.br/en/web/dou/-/portaria-359094139>. Acessado em: 10 de março de 2023. [↑](#footnote-ref-1)
3. **Portaria Nº 1.510/2010 MTE**. Disponível em: <https://www.trt2.jus.br/geral/tribunal2/ORGAOS/MTE/Portaria/P1510\_09.html>. Acessado em: 10 de março de 2023. [↑](#footnote-ref-2)
4. **Portaria Nº 373/2011 MTE**. Disponível em: <https://www.trt2.jus.br/geral/tribunal2/ORGAOS/MTE/Portaria/P373\_11.html>. Acessado em: 10 de março de 2023. [↑](#footnote-ref-3)
5. **SCHULTZ**, Felix. **Entenda 5 vantagens do Blockchain para empresas**. Milvus. 11 de janeiro de 2019. Disponível em: <https://blog.milvus.com.br/entenda-5-vantagens-do-blockchain-para-as-empresas/>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-4)
6. Apesar de matematicamente possível, devido ao processamento computacional dispensado é praticamente impossível alteração de um dado da cadeia blockchain enquanto os nós honestos forem a maioria da rede. **NAKAMOTO**, Satoshi. **Bitcoin: A Peer-to-Peer Eletronic Cash System**. Bitcoin.org. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-5)
7. Conforme define **FACHINI**, Tiago: “Smart Contract é um tipo de contrato digital, que usa tecnologia blockchain de ponta para garantir a auto execução das cláusulas, sempre que as condições contratuais previstas são atendidas.” Projuris. Disponível em: <https://www.projuris.com.br/smart-contract/>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-6)
8. Bitcoin is a consensus network that enables a new payment system and a completely digital money. It is the first decentralized peer-to-peer payment network that is powered by its users with no central authority or middlemen. From a user perspective, Bitcoin is pretty much like cash for the Internet. Bitcoin can also be seen as the most prominent [triple entry bookkeeping system](http://financialcryptography.com/mt/archives/001325.html" \o "http://financialcryptography.com/mt/archives/001325.html) in existence. **Bitcoin.org**. **What is Bitcoin**. Disponível em: <https://bitcoin.org/en/faq#general>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-7)
9. **DAVIDSON**, All. **Increasing trust in criminal evidence with blockchains**. MOJ Digital & Technology. Disponível em: <https://mojdigital.blog.gov.uk/2017/11/02/increasing-trust-in-criminal-evidence-with-blockchains/>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-8)
10. **ENCCLA-2020**. **Estratégia Nacional de Combate à Corrupção e Lavagem de Dinheiro. Blockchain no setor público: Guia de conceitos e usos potenciais**. Agosto de 2020. Disponível em: <http://enccla.camara.leg.br/acoes/arquivos/resultados-enccla-2020/blockchain-no-setor-publico-guia-de-conceitos-e-usos-potenciais>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-9)
11. **SILVA**, Euber Chaia Cotta e, e **MARQUES**, Rodrigo Moreno. **Blockchain no setor público: uma revisão sistemática de literatura**. AtoZ:novaspráticaseminformaçãoeconhecimento,10(3),1-11,set./dez.2021. Disponível em: <https://revistas.ufpr.br/atoz/article/view/79903/44241>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-10)
12. **O**, S. K., **XU**, X., **CHIAM**, Y. K. and **LU**, Q., "**Evaluating Suitability of Applying Blockchain**," 2017 22nd International Conference on Engineering of Complex Computer Systems (ICECCS), 2017, pp. 158-161, doi: 10.1109/ICECCS.2017.26. [↑](#footnote-ref-11)
13. **WÜST**, Karl e **GERVAIS**, Arthur. **Do you need a blockchain?** 2018. Disponível em: <https://www.law.berkeley.edu/wp-content/uploads/2018/08/Do-you-need-a-Blockchain-Karl-Wust-and-Arthur-Gervais.pdf>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-12)
14. **Polygon** é uma solução de segunda camada da rede Ethereum utilizada para conectar e escalonar projetos em blockchain, sobretudo aqueles relacionados e compatíveis com a rede Ethereum. Disponível em: <https://coinext.com.br/criptomoedas/polygon>. Acessado em: 10 de março de 2023. [↑](#footnote-ref-13)
15. **Ethereum** é uma plataforma descentralizada que faz uso da tecnologia de TLD (Tecnologia de Livro Razão

    Distribuído) onde, além de possuir sua própria criptomoeda (ETH), é possível executar contratos inteligentes, criada em 2014 pelo russo-canadense Vitalik Buterin. Disponível em: <https://ethereum.org/pt-br/>. Acessado em: 21 de maio de 2022. [↑](#footnote-ref-14)
16. **Hardhat** is a development environment for Ethereum software. It consists of different components for editing, compiling, debugging and deploying your smart contracts and dApps, all of which work together to create a complete development environment. Hardhat Runner is the main component you interact with when using Hardhat. It's a flexible and extensible task runner that helps you manage and automate the recurring tasks inherent to developing smart contracts and dApps. [↑](#footnote-ref-15)
17. **Solidity** é uma linguagem orientada a objetos de alto nível, onde é possível implementar smart contracts. [↑](#footnote-ref-16)
18. **Single Page Application** (SPA) é uma estratégia de desenvolvimento web onde uma única página contem todas as outras páginas do site. [↑](#footnote-ref-17)
19. **React** é uma biblioteca Javascript para criar interfaces de usuário. [↑](#footnote-ref-18)
20. **ESCOBAR,** Matheus Garcia. **Contextualização e Introdução ao Blockchain**. 2021. Disponível em: <https://www.ufsm.br/pet/sistemas-de-informacao/2021/11/29/contextualizacao-e-introducao-ao-blockchain/>. Acesso em: 22 de maio de 2022. [↑](#footnote-ref-19)
21. **NAKAMOTO,** Satoshi. Bitcoin: **A Peer-to-Peer Eletronic Cash System**. 2008 Bitcoin.org. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-20)
22. **NAKAMOTO,** Satoshi. Bitcoin: **A Peer-to-Peer Eletronic Cash System**. p. 2. 2008 Bitcoin.org. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Acesso em: 21 de maio de 2022. [↑](#footnote-ref-21)
23. **STALLINGS,** William. **Criptografia e segurança de redes**. p. 181. 4. ed. São Paulo: Pearson Prentice Hall, 2008. [↑](#footnote-ref-22)
24. https://www.blockchain.com/btc/tx/88dcb0188df671f0ae5154780382c89fb062582186a6038a585880945f3cbbc0 [↑](#footnote-ref-23)
25. **STALLINGS,** William. **Criptografia e segurança de redes**. p. 227. 4. ed. São Paulo: Pearson Prentice Hall, 2008. [↑](#footnote-ref-24)
26. **ZHENG,** Zibin, **XIE,** Shaoan, **DAI**, Hongning, **CHEN**, Xiangping, **WANG**, Huaimin. "**An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends**," 2017 IEEE International Congress on Big Data (BigData Congress), 2017, pp. 557-564, doi: 10.1109/BigDataCongress.2017.85. [↑](#footnote-ref-25)
27. **KUNTZ,** João. **Blockchain Ethereum – Fundamentos de Arquitetura, desenvolvimento de contratos e aplicações.** p. 22. São Paulo: Casa do Código, 2022. [↑](#footnote-ref-26)
28. **RODRIGUES**, Carlo Kleber da Silva. **Uma análise simples de eficiência e segurança da Tecnologia Blockchain. Revista de Sistemas e Computação**, Salvador, v. 7, n. 2, p. 147-162, jul./dez. 2017. Disponível em: <https://repositorio.uniceub.br/jspui/bitstream/235/11373/1/Uma%20an%c3%a1lise%20simples%20de%20efici%c3%aancia%20e%20seguran%c3%a7a%20da%20Tecnologia%20Blockchain.pdf>. Acesso em: 31 de julho de 2022. [↑](#footnote-ref-27)
29. **ATTARAN,** Mohsen e **GUNASEKARAN,** Angappa. **Applications of Blockchain Technology in Business -   
    Challenges and Opportunities**. p. 14. Texas: Springer, 2019. [↑](#footnote-ref-28)
30. **BSC Army. Node and how to Run It.** Disponível em: <https://bscarmy.com/node-and-how-to-run-it/>. Acesso em: 01 de agosto de 2022. [↑](#footnote-ref-29)
31. **ZHENG,** ZIBIN, **XIE**, Shaoan, **DAI**, Hongning, **CHEN**, Xiangping, **WANG**, Huaimin, "**An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends**," *2017 IEEE International Congress on Big Data (BigData Congress)*, 2017, 559 p., doi: 10.1109/BigDataCongress.2017.85. [↑](#footnote-ref-30)
32. **KUNTZ,** João. **Blockchain Ethereum – Fundamentos de Arquitetura, desenvolvimento de contratos e aplicações.** 26p.São Paulo: Casa do Código, 2022. [↑](#footnote-ref-31)
33. **The merge**. Disponível em: <https://ethereum.org/en/upgrades/merge/>. Acessado em: 10 de março de 2023. [↑](#footnote-ref-32)
34. **KUNTZ,** João. **Blockchain Ethereum – Fundamentos de Arquitetura, desenvolvimento de contratos e aplicações.** p. 21. São Paulo: Casa do Código, 2022. [↑](#footnote-ref-33)
35. **ZHENG,** Zibin, **XIE,** Shaoan, **DAI**, Hongning, **CHEN**, Xiangping, **WANG**, Huaimin. "**An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends**," 2017 IEEE International Congress on Big Data (BigData Congress), 2017, pp. 558, doi: 10.1109/BigDataCongress.2017.85. [↑](#footnote-ref-34)
36. **ZHENG,** Zibin, **XIE,** Shaoan, **DAI**, Hongning, **CHEN**, Xiangping, **WANG**, Huaimin. "**An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends**," 2017 IEEE International Congress on Big Data (BigData Congress), 2017, pp. 558, doi: 10.1109/BigDataCongress.2017.85. [↑](#footnote-ref-35)
37. **OLIVEIRA**, F. R.; **MAZIERO**, R. C.; **ARAÚJO**, L. S. de. U**M ESTUDO SOBRE A WEB 3.0: evolução, conceitos, princípios, benefícios e impactos.** Revista Interface Tecnológica, [S. l.], v. 15, n. 2, p. 60–71, 2018. DOI: 10.31510/infa.v15i2.492. Disponível em: <https://revista.fatectq.edu.br/interfacetecnologica/article/view/492>. Acesso em: 8 jun. 2022. [↑](#footnote-ref-36)
38. **ZAGO,** Mateo. **Why the Web 3.0 Matters and you should know about it.** 2018. Disponível em: <https://medium.com/@Matzago/why-the-web-3-0-matters-and-you-should-know-about-it-a5851d63c949>. Acesso em: 08 de junho de 2022. [↑](#footnote-ref-37)
39. Vale ressaltar que, apesar do objetivo cada vez maior do agnosticismo buscado pelas plataformas blockchain, ainda há certas limitações de interoperabilidade mesmo entre as diferentes soluções blockchain. Como exemplo de barreira ainda não superada, podemos citar a barreira de executar smart contracts da rede Ethereum dentro da rede Solana, por exemplo. No ano em que escrevo esta obra, inciativas bem avançadas e promissoras neste sentido de interoperabilidade entre plataformas já tem sido desenvolvidas, mas ainda em fase de testes. Como estamos falando de tecnologias relativamente novas, é perfeitamente factível esperar uma padronização no setor que consiga nos levar a este alvo. [↑](#footnote-ref-38)
40. As blockchains públicas se diferenciam das blockchains privadas exatamente por este caráter de abertura para participação. Enquanto as públicas não fazem qualquer restrição para a entrada de um participante na rede, as blockchains privadas controlam seus participantes. Esta última abordagem tem sido utilizada principalmente por grandes empresas para integrar dentro de seu guarda-chuva tecnológico seus fornecedores e clientes para que haja um controle de toda a cadeia produtiva de modo fim-a-fim proporcionando automatização de processos e rastreabilidade de eventos, com agentes participando de todo o processo da cadeia e outros adentrando e saindo apenas em momentos específicos. Exemplo de plataforma que faz esta abordagem é a Hyperledger Fabric, iniciativa da Linux Foundation e amplamente utilizada pela IBM. Famoso vídeo do caso de parceria entre a gigante da tecnologia IBM e a gigante da logística Maersk utilizando blockchain como solução pode ser vista neste endereço eletrônico: <https://www.youtube.com/watch?v=tdhpYQCWnCw>. [↑](#footnote-ref-39)
41. **coinbase - [https://www.coinbase.com/](https://www.coinbase.com/" \o "https://www.coinbase.com/)** [↑](#footnote-ref-40)
42. **TEKISALP,** Emre. **Understanding Web3 – A User Controlled Internet.** Disponível em: <https://www.coinbase.com/blog/understanding-web-3-a-user-controlled-internet>. Acessado em 10 de março de 2023. [↑](#footnote-ref-41)
43. **MORAIS**, Anderson Melo de, **LINS**, Fernado Antonio Aires. **Uso de Blockchain na Educação: Estado da arte e desafios em aberto**. 2020. Disponível em: <https://www.nucleodoconhecimento.com.br/tecnologia/uso-de-blockchain>. Acesso em: 03 de abril de 2022. [↑](#footnote-ref-42)
44. **CUNHA**, S. S. d. et al. **Relatório sobre o Sistema Brasileiro de Votação Eletrônica.** COMITÊ MULTIDISCIPLINAR INDEPENDENTE, 2014. Disponível em: <http://www.brunazo.eng.br/voto-e/textos/CMind-1-Brasil-2010.pdf>. [↑](#footnote-ref-43)
45. **ARRIAL**, T. **Estudo Técnicos** - junho de 2018. Confederação Nacional de Municı́pios, 2018. Disponı́vel em: <https://www.cnm.org.br/cms/biblioteca/Eleitorado-2018.pdf>. [↑](#footnote-ref-44)
46. **TSE.** **Eleições 2018: confira as datas do calendário eleitoral**. 2018. Disponı́vel em: <https://g1.globo.com/politica/eleicoes/2018/noticia/eleicoes-2018-datas.ghtml>. [↑](#footnote-ref-45)
47. **RAMALHO,** R. **Prazo para eleitor pedir voto em trânsito termina nesta quinta.** 2018. Disponı́vel em: <https://g1.globo.com/politica/eleicoes/2018/noticia/2018/08/22/termina-nesta-quinta-feira-prazo-para-eleitor-pedir-voto-em-transito.ghtml>. [↑](#footnote-ref-46)
48. **Niwa**, Henrique; "**UM SISTEMA DE VOTO ELETRÔNICO BASEADO EM BLOCKCHAIN**", p. 2879 . In: Anais do XIX Simpósio de Pesquisa Operacional & Logística da Marinha. São Paulo: Blucher, 2020. Disponível em: <https://www.proceedings.blucher.com.br/article-details/um-sistema-de-voto-eletrnico-baseado-em-blockchain-34623>. Acesso em: 20 de junho de 2022. [↑](#footnote-ref-47)
49. **CARVALHO,** Carla Arigony de; **ÁVILA**, Lucas Veiga. **A TECNOLOGIA BLOCKCHAIN APLICADA AOS CONTRATOS INTELIGENTES.** Revista Em Tempo, [S.l.], v. 18, n. 01, p. 159, dec. 2019. ISSN 1984-7858. Disponível em: <https://revista.univem.edu.br/emtempo/article/view/3210>. Acesso em: 19 june 2022. [↑](#footnote-ref-48)
50. **KUNTZ,** João. **Blockchain Ethereum – Fundamentos de Arquitetura, desenvolvimento de contratos e aplicações.** 54p.São Paulo: Casa do Código, 2022. [↑](#footnote-ref-49)
51. [https://www.stateofthedapps.com/stats/platform/ethereum#new](https://www.stateofthedapps.com/stats/platform/ethereum" \l "_blank" \o "https://www.stateofthedapps.com/stats/platform/ethereum#_blank) [↑](#footnote-ref-50)
52. [https://dune.com/sawmon\_and\_natalie/smart-contracts-on-ethereum](https://dune.com/sawmon_and_natalie/smart-contracts-on-ethereum" \o "https://dune.com/sawmon_and_natalie/smart-contracts-on-ethereum) [↑](#footnote-ref-51)
53. [https://etherscan.io/txs](https://etherscan.io/txs" \o "https://etherscan.io/txs) [↑](#footnote-ref-52)
54. **TAKENOBU, T. – Ethereum EVM Illustrated**. Disponível em: <https://takenobu-hs.github.io/downloads/ethereum\_evm\_illustrated.pdf>. Acessado em: 10 de março de 2023. [↑](#footnote-ref-53)
55. **Máquina virtual do Ethereum (EVM)**. Disponível em: <https://ethereum.org/pt-br/developers/docs/evm/>. Acessado em: 11 de março de 2023. [↑](#footnote-ref-54)
56. **TAKENOBU, T. – Ethereum EVM Illustrated**. Disponível em: <https://takenobu-hs.github.io/downloads/ethereum\_evm\_illustrated.pdf>. Acessado em: 11 de março de 2023. [↑](#footnote-ref-55)
57. **Contas Ethereum**. Disponível em: <https://ethereum.org/pt-br/developers/docs/accounts/>. Acessado em: 11 de março de 2023. [↑](#footnote-ref-56)
58. **Chave privada, pública e endereço de uma conta**. Na geração dos dados de uma conta a geração da chave privada é completamente aleatória e após aplicação do algoritmo se obtém a chave pública. Perceba que o hash tem como entrada uma chave pública e como saída, pegando os últimos 20 dígitos e acresentando 0x no início temos o endereço de uma conta – Contribuição de eth.build. Disponível em: <https://sandbox.eth.build/> . Acessado em: 11 de março de 2023. [↑](#footnote-ref-57)
59. **TAKENOBU, T. – Ethereum EVM Illustrated**. Disponível em: <https://takenobu-hs.github.io/downloads/ethereum\_evm\_illustrated.pdf>. Acessado em: 11 de março de 2023. [↑](#footnote-ref-58)
60. **TAKENOBU, T. – Ethereum EVM Illustrated**. Disponível em: <https://takenobu-hs.github.io/downloads/ethereum\_evm\_illustrated.pdf>. Acessado em: 11 de março de 2023. [↑](#footnote-ref-59)
61. **UML – UNIFIED MODELING LANGUAGE**. Linguagem de Modelagem Unificada ajuda a especificar, visualizar e documentar modelos de sistema de software, incluindo sua estrutura e arquitetura, como forma de organizar todos esses requisitos. UML é marca registrada de **OMG.** Adaptado. Disponível em: <https://www.uml.org/what-is-uml.htm>. Acessado em: 19 de junho de 2022. [↑](#footnote-ref-60)