Universidade Estadual de Londrina

TATIANE SOARES FERREIRA

# EM BUSCA DE UMA VERIFICAÇÃO MAIS RIGOROSA PARA CONTRATOS INTELIGENTES

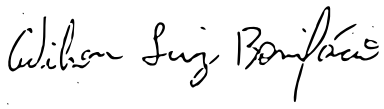
**LONDRINA**
**2023**

TATIANE SOARES FERREIRA

# EM BUSCA DE UMA VERIFICAÇÃO MAIS RIGOROSA PARA CONTRATOS INTELIGENTES

Versão Preliminar de Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Adilson Luiz Bonifácio



**LONDRINA**
**2023**

FERREIRA, T. S.. **Em busca de uma verificação mais rigorosa para contratos inteligentes**. 2023. 30f. Trabalho de Conclusão de Curso – Versão Preliminar (Bacharelado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina, 2023.

## RESUMO

A tecnologia tem sido muito importante no desenvolvimento humano, desempenhando um papel fundamental na resolução de problemas complexos. Neste contexto, este projeto explora a verificação rigorosa de contratos inteligentes. O objetivo é proporcionar maior segurança e confiabilidade aos contratos inteligentes, implementando um sistema com suporte dos modelos de redes de Petri.

**Palavras-chave**: Verificação. Redes de Petri. Redes de Petri Coloridas. CPN. Contratos Inteligentes

FERREIRA, T. S.. **Search of a More Rigorous Verification for Smart Contracts**. 2023. 30p. Final Project – Draft Version (Bachelor of Science in Computer Science) – State University of Londrina, Londrina, 2023.

# ABSTRACT

This project aims to enhance the security and reliability of smart contracts by implementing a system that transforms them into Petri Net models for comprehensive verification, with the hope of contributing to the development of safer and more reliable smart contracts. Technology has been essential in human progress, with a fundamental role in solving complex problems. In this context, this project explores the rigorous verification of smart contracts. It aims to provide more security and reliability for smart contracts, implementing a system supported by Petri net models.

**Keywords**: Verification. Petri Nets. Colored Petri Nets. CPN. Smart Contracts

# LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| CPN | Colored Petri Nets |

# SUMÁRIO

# 1 INTRODUÇÃO

A tecnologia é formada pelo conjunto de conhecimentos, habilidades, processos e dispositivos criados e utilizados pelos seres humanos, sendo assim, o presente projeto visa aprofundar a compreensão de um domínio tecnológico de grande impacto na atualidade global: a tecnologia blockchain, os contratos inteligentes e o Ethereum.

Neste cenário, a tecnologia blockchain surge como uma das inovações mais promissoras e revolucionárias, onde inicialmente ganhou destaque como a infraestrutura responsável pelo grande avanço das criptomoedas, como o Bitcoin, mas sua utilidade se estende para muito além disso. A blockchain é uma espécie de registro digital descentralizado, que mantém um histórico imutável e transparente de todas as transações realizadas em uma rede, ou seja, todas as transações são registradas em blocos interconectados e protegidas por criptografia, fazendo com que se torne virtualmente à prova de adulterações.

Uma das aplicações mais notáveis da blockchain é a criação de contratos inteligentes. Tais contratos tem como principal objetivo revolucionar as transações e acordos comerciais, eliminando intermediários e garantindo a execução automática de acordos, visto que esses contratos são programas autoexecutáveis que podem ser programados para realizar automaticamente ações específicas quando condições predefinidas são atendidas, eliminando assim a necessidade de intermediários. Portanto, a blockchain é uma peça fundamental em um cenário tecnológico em constante evolução, com o potencial de revolucionar várias áreas da economia global.

No entanto, à medida que os contratos inteligentes e a blockchain ganham popularidade, surgem também preocupações sobre quão seguro e confiável é usá-los. Uma falha em um contrato inteligente pode ter consequências significativas, um exemplo disso é o ataque DAO (Decentralized Autonomous Organization) uma organização virtual baseada em Ethereum. Este ataque é um dos mais conhecidos, visto que foi um grande incidente na história das criptomoedas e dos contratos inteligentes, onde o invasor explorou uma vulnerabilidade ligada às funções de fallback e à propriedade de reentrância no contrato DAO. A vulnerabilidade permitiu o hacker fazer vários pedidos de saque de forma rápida, antes mesmo que o contrato pudesse atualizar o saldo das contas, passando 60 milhões de dólares para seu controle.

Sendo assim, há uma necessidade de ser realizada uma verificação e validação rigorosa nos contratos, a fim de aumentar sua segurança e permitir que os usuários tenham mais confiança na tecnologia que estão usando. Este projeto visa contribuir na segurança e

confiabilidade dos contratos inteligentes evitando que mais ataques como DAO aconteçam, e permitindo uma verificação abrangente por meio da transformação em modelos de Redes de Petri. Com isso, é esperado fornecer uma base sólida para o desenvolvimento futuro de contratos inteligentes mais seguros e confiáveis.

O restante desse projeto está organizado da seguinte forma. A Seção 2 descreve todos os assuntos que estarão envolvidos no desenvolvimento deste projeto como: blockchain, cotratos inteligentes, Ethereum, Solidity e o modelo que será usado de Rede de Petri . Já os objetivos dessa proposta estão descritos na Seção **??**. Os procedimentos e o cronograma de estudo, criação e desenvolvimento são apresentados na Seção **??**. A Seção **??** lista algumas das contribuições esperadas com a implementação do objetivo descrito. A proposta termina com uma lista de referências usadas na elaboração desse projeto.

# 2 FUNDAMENTAÇÃO

Esta seção fornece a fundamentação teórica necessária para o desenvolvimento do projeto proposto. Os conceitos sobre a tecnologia Blockchain, contratos inteligentes, a plataforma Ethereum e a linguagem Solidity, bem como os modelos de redes de Petri são descritos a seguir.

## 2.1 Blockchain

A tecnologia Blockchain surgiu, inicialmente, como um banco de dados descentralizado usando criptografia. O sistema garante confiabilidade, sem a necessidade de entidades terceiras de verificação, na realização de transações [26].

Apesar de sua grande eficácia, o conceito de Blockchain se estabeleceu amplamente apenas a chegada do Bitcoin [20], uma implementação de transações com criptomoedas. Essas transações são agrupadas em estruturas de tamanho restrito chamadas de blocos, com um timestamp compartilhado, chave pública do portador da criptomoeda e uma assinatura produzida pelo usuário detentor da criptomoeda, com chave privada [17].

Uma das base da tecnologia blockchain é o hash, que pode ser considerado uma string criptografada da string original. Um hash é uma representação segura e única de um conjunto de dados, ou seja, uma sequência de caracteres gerada a partir de dados digitais usando um algoritmo de hash, com o objetivo de garantir a integridade e segurança dos dados armazenados na rede [19].

A imutabilidade dos dados numa Blockchain é então garantida pela dificuldade de se alterar os dados de um bloco, pois para ocorrer uma alteração nos dados de um bloco é necessário que o valor calculado da hash seja válido para todos os blocos seguintes ao bloco alterado, o que é muito difícil de acontecer. A verificação dos blocos também deve ser realizada pelos usuários para garantir a segurança da rede. No bitcoin o método *proof-of-work* é usado para que os contribuintes criem uma hash válida para um certo bloco. Quando ocorre a criação da primeiro hash válida uma recompensa é enviada ao proprietário. Já um outro método, chamado *proof-of-stake*, o hash é calculado de forma aleatória, e quando um usuário selecionado aprova um bloco inválido, as criptomoedas depositadas por ele são recolhidas. Além disso, o usuário que investem mais moedas têm maiores chances de criarem um bloco [17].

Uma Blockchain pode ser representada por uma tabela com três colunas, onde a

primeira possui o horário e data em que ocorreu a transação. A segunda coluna armazena os detalhes da transação, tais como endereços das partes envolvidas, quantidade de ativos a ser transferido, assinatura digital, entre outras informações. Já a terceira coluna armazena um hash atual mais o hash da transação anterior. Cada linha da tabela representa uma transação diferente [8].

Um ponto importante no funcionamento de uma blockchain diz respeito aos mineradores. Os mineradores são participantes da rede encarregados em manter a integridade e segurança da rede. Além disso, esses participantes são responsáveis por validar as transações, criar novos blocos e garantir que o funcionamento da blockchain esteja em conformidade com as regras do protocolo específico da blockchain. Portanto, quando um novo registro é inserido na blockchain, o hash calculado na última transação, ou seja, aquele computado no último bloco pelos mineradores, é compartilhado com todas as entidades ou participantes envolvidos na operação [8].

A Figura 1 ilustra o funcionamento de uma Blockchain. Inicialmente, quando um usuário solicita uma transação, um bloco que representa essa transação é criado, contendo todas as informações fornecidas pelo usuário. Após sua criação, o bloco é anunciado e difundido para todos os nós que fazem parte da rede, ou seja, para os nós que participam da transação. Em seguida, ocorre uma validação por parte desses nós. Se todas as informações estiverem corretas de acordo com as regras da rede, o novo bloco é adicionado à Blockchain.

Para concluir o processo de inserção do bloco, uma verificação final sobre as informações contidas no bloco e a verificação de conformidade com as regras da rede são realizadas. Se tudo estiver em conformidade, a transação é executada. Esse ciclo de criação de blocos, validação e adição à Blockchain é fundamental para garantir a segurança e a integridade das transações na rede.

## 2.2 Contratos

Um dos pontos fundamentais nos negócios atualmente é o uso de contratos com o intuito de se firmar acordos entre duas ou mais partes. Para que um arranjo seja válido de forma judicial, é necessário que haja certos requisitos, como: (I) o acordo, uma oferta séria e voluntária, expressa por uma parte à outra sem influência externas ou restrições, podendo ser feita apenas por pessoa física ou jurídica; (II) a consideração, é o interesse dos envolvidos na execução do acordado, de forma que cada um deva oferecer, dar ou prometer algo ao outro, sendo isto indispensável para a validação jurídica; (III) competência e capacidade das partes, onde uma pessoa incapaz de compreender o contrato não pode finalizá-lo; (IV) a finalidade legal e o objeto presente no contrato devem estar dentro da lei. Portanto, um contrato é tido

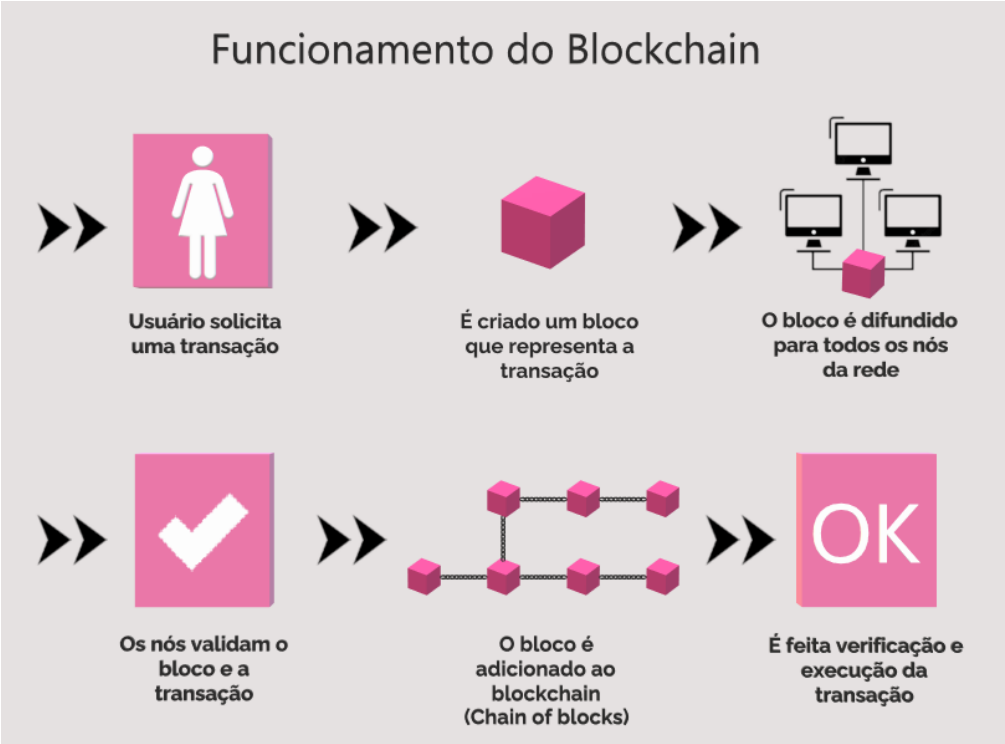



Figura 1 – Como funciona o blockchain

como nulo ou anulável caso algum destes aspectos esteja em falta, onde o último tem efeito, porém pode ser rescindido pelo tribunal [14].

De forma geral, um contrato é um ato institucional, onde são declarados arranjos jurídicos por partes relacionadas. O conteúdo presente num contrato pode ser analisado por diferentes elementos, e de acordo com o objetivo de cada indivíduo, visto que há contratos que: (I) impõe a necessidade de obrigações a serem cumpridas na relação de um indivíduo para com o outro; (II) a criação de liberdades à uma das partes; (III) transferência de direito; (IV) ações e penalidades, caso as condições impostas no contrato não sejam cumpridas [14].

Para a formação deste arranjo, em um primeiro momento é necessário haver negociação e formação dos termos que interessam ambas as partes. Qualquer pessoa jurídica tem a possibilidade de celebrar contratos, propondo uma oferta, no qual deve-se obter uma aceitação para então ter a conclusão. Já num segundo momento deve haver um armazenamento e notorização deste, para que possa ser desempenhado, uma vez que esteja de acordo com os objetivos e leis estabelecidas. Caso uma das partes descumpra o dever que lhe foi imposto, as outras partes têm o direito de rescindir o contrato, além de ser possível desistir ou até mesmo buscar uma indenização, compensatória ou punitiva, ou fazer uma renegociação dos termos.

### 2.2.1 Contratos eletrônicos

Com o avanço tecnológico os processos contratuais e os próprios contratos foram sendo adaptados e implementados de forma a facilitar e oferecer melhorias aos usuários, surgindo o conceito de contrato eletrônico ou *e-contrato* [14]. Um e-contrato é um contrato que pode ser manipulado e processado por sistemas de computadores.

Além disso, um contrato eletrônico, de forma semelhante aos contratos físicos, consiste em três elementos principais: (I) oferta, que compreende os termos e condições apresentados pela parte que criou o contrato; (II) aceitação, que consiste na aprovação do contato por todas as partes envolvidas, confirmada através da assinatura eletrônica do contrato; (III) consideração, que envolve o cumprimento de todos os termos e acordos estabelecidos no contrato [9].

Qualquer ação, desde clicar em "Eu concordo"nos termos de serviço de um aplicativo até utilizar uma assinatura eletrônica para formalizar um contrato de compra ao adquirir uma residência, é considerada como a conclusão de um contrato eletrônico. Mesmo que essa forma de assinatura possa não parecer tão formal quanto a assinatura em papel, os contratos eletrônicos possuem a mesma validade legal e são igualmente vinculativos quando administrados corretamente [9].

Os contratos eletrônicos, diferente dos contratos em papel, oferece várias vantagens significativas, incluindo assinaturas mais rápidas, um controle de versões centralizado, permitindo a atualização simultânea para todas as partes em um único local e também, uma maior segurança devido às medidas de proteção contra ameaças como roubo, sabotagem e falsificação, implementadas tanto em nível digital quanto físico [9].

A criação de um e-contrato pode ser realizada por meio de softwares especializados, e-mails, processadores de texto ou diversas outras abordagens. No contexto empresarial, é frequente o uso de um software de gestão de contratos, pois esse tipo de sistema possibilita a criação e administração centralizada de todos os contratos e dados relacionados num único local [6].

### 2.2.2 Contratos inteligentes

Contrato inteligente[1] é um tipo de contrato que vem sendo amplamente utilizado nas tecnologias atuais. Surgiu com o intuito de ajudar o comércio reduzindo custos e disputas através de uma estrutura legal tecnológica [24].

Os contratos inteligentes podem ser definidos como um protocolo de transação auto-

[1] do inglês, *Smart contract*

matizado, no qual é executado os termos de um contrato, sendo que os contratos inteligentes permitem que um indivíduo transforme as cláusulas contratuais em código executável, reduzindo riscos externos e limitando a participação dos envolvidos no acordo. Sendo assim, um contrato inteligente é um acordo feito entre duas ou mais pessoas, entidades ou negócios, onde os termos e condições são acordados automaticamente, mesmo que não haja completa confiança mútua [3].

Apesar de seu surgimento na década de 1990, os contratos inteligentes não prosperaram, visto que havia uma necessidade da monitoração dos termos e execução dos contratos codificados por uma terceira parte, possibilitando o risco de algum dos envolvidos não cumprir suas obrigações. Porém, ao se juntar os contratos inteligentes com a tecnologia de blockchain uma nova porta se abriu, e a blockchain se tornou a plataforma mais comum e popular. Esta junção entre o contrato inteligente e o blockchain dá a possibilidade aos usuários de usufruírem de forma eficaz e vantajosa deste conceito, pois os próprios usuários se tornam parte do processo de análise, descartando a necessidade de uma terceira entidade confiável para este trabalho [22].

Um contrato inteligente baseado em blockchain, em sua essência, é apenas um código executável para automatizar a análise dos termos de um acordo, que pode trazer taxas de transação mais baixas comparado com outros sistemas, no qual é necessário um trabalho terceirizado para garantir que os termos de um contrato estejam sendo cumpridos [2].

Além disso, um contrato inteligente pode ser separado em duas categorias: (I) o código do contrato inteligente, onde há um blockchain que o armazena, verifica e executa, dependendo totalmente da linguagem de programação na qual o contrato foi implementado, e das ferramentas disponibilizadas pelo blockchain; (II) contrato legal inteligente, ou seja, um código com o intuito de incrementar ou substituir contratos legais, onde há a necessidade de intervenção jurídica, política ou empresarial para sua formação, não dependendo da tecnologia a ser utilizada [2].

## 2.3 Uma Plataforma Blockchain

O Ethereum (ETH), sendo uma tecnologia blockchain, tem transformado a maneira de criar e executar contratos inteligentes. Nesta seção, será explorado os fundamentos do Ethereum e Solidity, a linguagem de programação utilizada para escrever os contratos inteligentes.

### 2.3.1 Ethereum

Ethereum é uma plataforma Blockchain [5] que proporciona maior liberdade na escrita de contratos inteligentes. Um ponto importante deste sistema é a existência de uma linguagem Turing completa própria para descrever contratos, ou seja, qualquer tipo de cálculo é suportado pelo Ethereum, fornecendo uma camada abstrata, no qual qualquer pessoa pode criar as suas regras, formatos e funções utilizando contratos inteligentes [25].

A estrutura base desta plataforma são as contas, que possuem um endereço de 20 bytes como identificador e divididas em: (I) contas de propriedade externa, controladas por chaves privadas e não possui um código associado; (II) contas de contrato, controladas pelo código dos seus respectivos contratos, que ao receber uma mensagem pode então criar outros contratos ou mensagens como saída. O Ethereum é estruturado da seguinte forma: (I) nonce, número de transações enviadas ou de contratos criado para garantir a unicidade; (II) saldo de ether, criptomoeda utilizada no Ethereum; (III) código hash, onde é armazenado o valor do hash utilizando um código de contrato, e é executado caso algum endereço receba uma mensagem; (IV) storageRoot, armazena o hash da conta, ou seja, armazena o conteúdo da conta numa estrutura de dados [25].

O Ethereum executa as instruções de um contrato inteligente traduzindo-os para sua máquina virtual, a EVM (Ethereum Virtual Machine). Uma das funcionalidades do ambiente é cobrar um valor em Wei (Wei representa a menor fração de Ether, um Ether é o mesmo que 1018 Wei) ou ether para que uma transação possa ser finalizada. Outra funcionalidade é quando ocorre a execução de uma operação por um contrato, e o valor para a execução não é suficiente. Nesse caso, qualquer alteração realizada é revertida, e qualquer quantia restante é devolvida ao usuário inicial [17].

A rede Ethereum tem sua base na modificação do protocolo GHOST (Greedy Heaviest Observed Subtree) [25]. As trocas de dados na rede podem ser de duas formas. Primeiramente, as transações que possuem o intuito principal de transferir valores de uma conta para outra, ou a execução de contratos inteligentes, que são armazenadas na rede Ethereum. Já as mensagens permitem a comunicação e coordenação entre os contratos inteligentes. Quando um contrato recebe uma mensagem, funções específicas podem ser executadas com base nas informações passadas. Ao contrário do primeiro tipo de troca de dados, as mensagens não são registradas como transações independentes na Blockchain, sendo utilizadas apenas internamente entre os contratos, desempenhando um papel crucial na implementação de aplicativos descentralizados.

### 2.3.2 Solidity

Solidity é uma linguagem de programação Turing completa de alto nível, criada para o desenvolvimento de contratos inteligentes da plataforma Ethereum. A sintaxe de Solidity se assemelha muito a C++, Python e JavaScript, linguagens tipadas e que suportam herança, polimorfismo, além de bibliotecas e tipos complexos externos. Um contrato em Solidity é um código como uma classe na programação orientada a objetos, com variáveis e funções, sendo algumas destas especiais, tais como, msg, block e tx.

As funções especiais em Solidity servem para acessar informações sobre uma transação de invocação, responsável pela inicialização de um contrato inteligente, e o estado atual do bloco na Blockchain. Essas informações permitem a recuperação de um endereço de origem, a quantidade de Ether e os dados que são enviados por uma transação de invocação [27].

Um contrato inteligente em Solidity, em geral, é compilado para um programa bytecode EVM (Ethereum Virtual Machine), formado por uma coleção de código e dados que residem num endereço Blockchain específico [16].

A Figura 2 ilustra um código simples em Solidity. No início de cada contrato é necessário que a versão desejada seja definida, pois caso não haja uma versão pré estabelecida o compilador leva em consideração a versão mais atual. Essa ausência da versão pode causar alguns problemas, já que as atualizações são realizadas de acordo com as versões específicas. Na segunda linha um tipo personalizado `enum` é definido para a variável `EstadoContrato` com dois valores possíveis para o contrato:“Ativo"e “Inativo". Outra variável, chamada `Pessoa` é declarada como uma `struct`, com os campos `CPF`, `idade` e `nome`, que devem armazenar as informações relevantes das pessoas envolvidas neste contrato especifico.

A descrição do contrato se inicia na linha 11 com a palavra reservada `contract`. Essa declaração define o escopo e as características do contrato implementado. Além disso, dentro desse escopo são definidas as variáveis de estado, funções, eventos e outros elementos que formam a lógica de um contrato inteligente.

As funções do exemplo são simples, para fins de ilustração apenas. A função `setPessoa()`, na linha 17, define os valores da estrutura `Pessoa`, passando os parâmetros `novo_CPF, nova_idade` e `enovo_nome`;

As linhas 13 e 14 são instanciadas os objetos `EstadoContrato public estado` e `Pessoa public pessoa`. Além disso, o contrato inclui um construtor, denotado pela função `constructor()`. No exemplo, o construtor é usado para inicializar o estado do contrato como “Ativo".

A linguagem solidity permite o uso de muitos outros tipos de variáveis, classificadas

```
1   pragma solidity ^0.8.0;
2
3   enum EstadoContrato { Ativo, Inativo }
4
5   struct Pessoa {
6       int CPF;
7       int idade;
8       string nome;
9   }
10
11  contract MeuContratoInteligente {
12
13      EstadoContrato public estado;
14      Pessoa public pessoa;
15
16      constructor() {
17          estado = EstadoContrato.Ativo;
18      }
19
20      function setPessoa(int novo_CPF, int nova_idade, string memory novo_nome)
            public {
21
22          pessoa = Pessoa(novo_CPF, nova_idade, novo_nome);
23      }
24
25      function getCPF() public view returns (int) {
26          return pessoa.CPF;
27      }
28
29      function getIdadePessoa() public view returns (int) {
30          return pessoa.idade;
31      }
```

Listing 2.1 – Exemplo de contrato inteligente em Solidity

Figura 2 – Exemplo de código Solidity.

em três grupos [17]: (I) locais, declaradas dentro do escopo de uma função; (II) globais, declaradas fora do escopo de qualquer função específica, tornando-as acessíveis a partir de qualquer lugar dentro do contrato. Em geral, as variáveis globais são usadas para armazenar informações que devem ser compartilhadas entre as diferentes partes de um contrato ou incluir informações relacionadas a Blockchain e as transações, como o bloco atual, o remetente da transação, etc; (III) estado, definidas pelo usuário, declaradas fora do escopo de função, e armazenadas permanentemente na Blockchain juntamente ao contrato.

## 2.4 O Formalismo de Redes de Petri

As Redes de Petri são um formalismo poderoso e versátil, comumente utilizado para modelagem de sistemas dinâmicos e concorrentes. Sendo assim, nesta seção, sera discutido a respeito dos formalismos das Redes de Petri Ordinárias e Redes de Peri coloridas, explorando a estrutura, conceitos e aplicações práticas.

### 2.4.1 Redes de Petri Ordinárias

Os modelos de Redes de Petri, propostas por Carl Adam Petri [21], descrevem adequadamente sistemas distribuídos e concorrentes, onde diferentes componentes estão interconectados, ou são realizados de forma simultânea e independentes uns dos outros. Assim, tais sistemas são descritos por Redes de Petri através e mecanismos de comunicação síncronos e assíncronos, além das restrições de compartilhamento de recursos [13].

**Definição 1** *Uma Rede de Petri (RdP) é definida por* $R = (P, T, A, O, K, C)$*, onde na Tabela 1 está descrito o que representa cada termo [2].*

- $P = \{\ p_1, p_2, \ldots, p_n\}$ *é um conjunto finito de lugares;*
- $T = \{t_1, t_2, \ldots t_q\}$ *é um conjunto finito de transições;*
- $A \subseteq (P \times T) \cup (T \times P)$ *é um conjunto finito de arcos;*
- $O : A \to \{1, 2, \ldots\}$ *é a função peso associada aos arcos;*
- $K : P \to \{0, 1, 2, \ldots\}$ *é a marcação inicial.*

Uma RdP, graficamente, pode ser representa da seguinte forma: (I) lugar (place), é um círculo; (II) transição (transition), representado por retângulos ou barras verticais; (III) arcos, são representados como setas, conectando lugares a transições e transições a lugares. Além disso, cada lugar pode ter em seu interior fichas (tokens), representadas por pontos. Nesse caso, o local é definido como marcado, descrevendo o estado global coletivo de um sistema [13].

A representação gráfica de uma rede de Petri básica é composta por dois elementos principais: uma entidade ativa conhecida como "transição"(representada por uma barra) e outra entidade passiva denominada "lugar"(representada por um círculo). Os lugares correspondem às variáveis de estado da rede, enquanto as transições representam as ações executadas pelo sistema. Esses dois elementos são interconectados por meio de setas direcionadas, que podem ser únicas ou múltiplas.

A Figura 3 ilustra uma RdP básica [11].

Os ativos de uma Rede de Petri são representados pelas fichas explicadas acima, esses ativos também são chamados de marcadores, possibilitam que uma transição seja disparada para então realizar uma ação, dessa forma, para se habilitar uma transição é necessário que haja marcadores suficientes em seu lugar de início. Para que a Rede de Petri funcione corretamente ela depende da marcação inicial imposta, visto que, quando um lugar possui

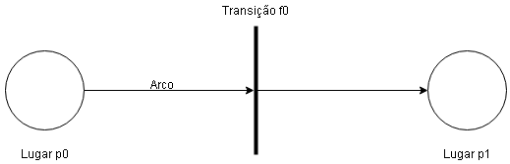



Figura 3 – Grafo de uma Rede de Petri básica

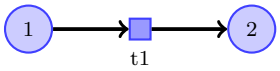



Figura 4 – Exemplo de Sequenciamento

marcadores capazes de habilitar uma transição, os ativos são consumidos do lugar de entrada de acordo com o peso do arco (valor determinado no arco) e a marca é produzida no lugar de saída. Caso os ativos não tenham peso pré estabelecido de forma explícita, por convenção é dito que o peso é 1 [17].

Para exemplificar esta definição, a seguir será mostrado um exemplo que representa o ano letivo de uma universidade, no qual o ano letivo começa com o primeiro período letivo, seguido das primeiras férias, logo após, tem-se o segundo período letivo, e por fim as férias novamente. Sendo assim, a Figura 5 representa o esquema proposto [11].

A descrição deste exemplo pode ser tida por:

$R_{AnoLetivo} = (P, T, I, O, K)$, onde

- O conjunto de lugares $P$ é $P = \{Periodo1, Ferias1, Periodo2, Ferias2\}$;
- O conjunto de transições $T$ é
  $T = \{GozarFerias1, RetornarPeriodo2, GozarFerias2, RetornarPeriodo1\}$;
- O conjunto de bags de entrada $I$ é
  $I = \{I(GozarFerias1) = [Periodo1], I(RetornarPeriodo2) = [Ferias1],$
  $I(GozarFerias2) = [Periodo2], I(RetornarPeriodo1) = [Ferias2]\}$;
- O conjunto de bags de saída $O$ é
  $O = \{O(GozarFerias1) = [Ferias1], O(RetornarPeriodo2) = [Periodo2],$
  $O(GozarFerias2) = [Ferias2], O(RetornarPeriodo1) = [Periodo1]\}$;

De um ponto de vista matricial a estrutura de uma Rede de Petri é representada por uma quádrupla $R =< P, T, Pre, Post >$, a Tabela 2 explica cada um dos termos [7]:

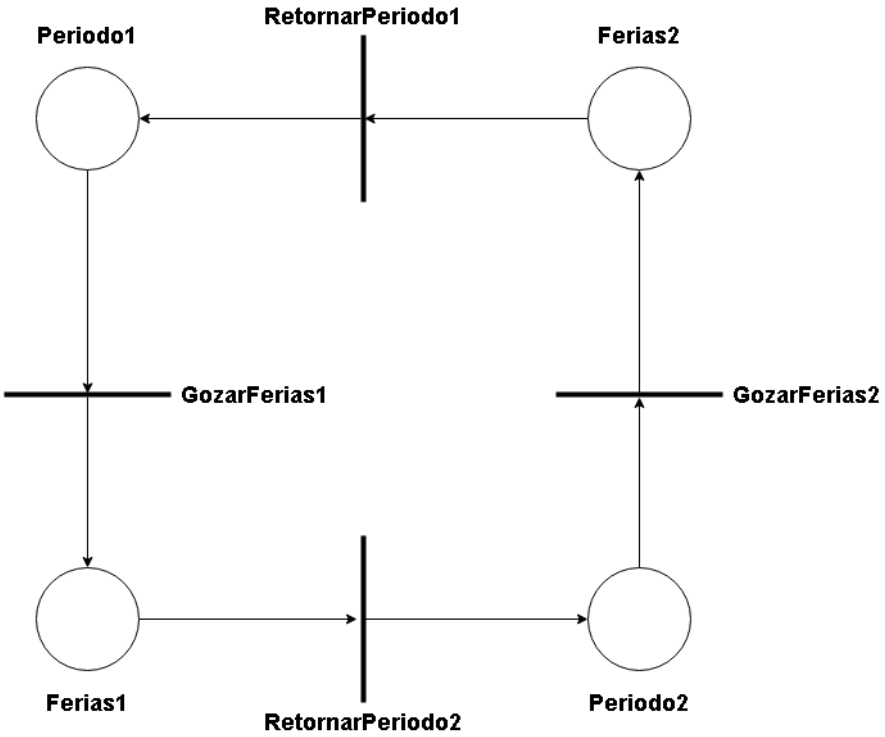



Figura 5 – Ano Letivo Representado por grafo em Redes de Petri

| **Conjunto** | **Descrição** |
|---|---|
| $P$ | Representa um conjunto finito de lugares de dimensão n |
| $T$ | Representa um conjunto finito de transições de dimensão m |
| $Pre : P \times T \rightarrow IN$ | é a aplicação de entrada (lugares precedentes ou incidência anterior), com IN sendo o conjunto dos n´umeros naturais |
| $Post : P \times T \rightarrow IN$ | é a aplicação de saída (lugares seguintes ou incidência posterior), com IN sendo o conjunto dos n´umeros naturais |

Tabela 1 – Descrição das variáveis e conjuntos da definição matricial.

A seguir é definido um exemplo de Rede de Petri onde há uma quádrupla $R =< P, T, Pre, Post >$ que possui [7]:

- $P = p1, p2, p3$;
- $T = a, b, c, d$;

Os valores das aplicações de entrada e saída são dados por:

- $Pre(p2, c) = 3$ e $Pre(p1, b) = Pre(p2, a) = Pre(p3, d) = 1$;
- $Post(p2, d) = 3$ e $Post(p1, a) = Post(p2, b) = Post(p3, c) = 1$;

Ao se utilizar um Rede de Petri é possível ter uma análise detalhada de um sistema modelado, o que torna mais fácil identificar pontos de falha ou fazer uma otimização no funcionamento. Além disso, as Redes de Petri são capazes de representar sistemas mais complexos, nesse tipo de caso, uma rede complexa pode ser composta por redes mais básicas. Algumas redes básicas podem ser combinadas para criar uma complexa este estas [Danilo]: (I) sequencial, esta rede se inicia em um lugar que representa uma condição, seguida de uma transição, ou seja, a ação, que ao ser disparada avança para um novo lugar; (II) distribuída, é muito semelhante a sequencial, porém ao realizar a ação, o avanço é feito para dois lugares de saída simultaneamente; (III) junção é o inverso da distribuição, visto ue os lugares de entrada devem estar disponíveis para simultaneamente habilitar a ação, para então gerar recursos no lugar de saída apenas; (IV) não-determinística, nesta rede um único lugar definido como entrada esta marcado para habilitar duas ações, no entanto os recursos desta marcação é suficiente para disparar apenas uma delas [17].

### 2.4.2 Redes de Petri coloridas

As Redes de Petri coloridas têm como propósito otimizar o tamanho de um modelo, tornando mais viável a individualização dos tokens por meio da atribuição de cores, permitindo a representação de diferentes processos ou recursos em uma única rede. As cores não são apenas referentes a tonalidades ou padrões, mas podem também representar tipos de dados complexos, onde a terminologia “colorida"é utilizada para destacar a capacidade de se distinguir os tokens da rede [11]. Neste sentido, as Redes de Petri Coloridas são uma extensão das Redes de Petri convencionais que torna mais simples a representação de múltiplos recursos em sistemas mais complexos. A distinção principal entre as redes convencionais e as Redes de Petri Coloridas reside na capacidade desta última de atribuir valores por meio de marcadores coloridos [17].

Uma RdP Colorida pode ser dividida em três partes, de acordo com a sua composição: (I) estrutura, o grafo dirigido composto por dois tipos de vértices, lugares, representados graficamente por círculos ou elipses, e transições, representadas por retângulos; (II) declarações, formadas pela especificação de conjuntos de cores e a declaração de variáveis; (III) inscrições, que estão associadas a componente da rede. Lugares possuem três tipos de inscrições: nomes, conjuntos de cores e a marcação inicial; Já as transições têm dois tipos de inscrições: nomes e expressões guarda; Por fim, os arcos possuem apenas um tipo de inscrição, representado por meio de expressões [11].

Para definir formalmente uma Rede de Petri Colorida é necessário a noção de multiconjuntos. Um multiconjunto $m$ sobre um conjunto não vazio $M$ é definido por uma função $m : M \to \mathbb{N}$, e representada pelo somatório $\sum_{x \in M} m(x)'s$, onde $m(x)$ é o número de elementos $x$ no multiconjunto. Assuma, por exemplo, o conjunto $M = \{a, b, c\}$. Ao adicionar o elemento $a$ à $M$, o conjunto $M$ ainda é dado por $\{a, b, c\}$. No entanto, $m$ é um multiconjunto sobre $M$ dado por $m = \{a, b, c\}$, ao adicionar o elemento $a$ à $m$, o resultado é $m = \{a, a, b, c\}$, contendo duas ocorrências do elemento $a$ [15].

Operações de adição, subtração, multiplicação escalar e comparação são definidas sobre multiconjuntos [17]. Para dois multiconjuntos $m_1$ e $m_2$ sobre $M$, temos que:

- $m_1 + m_2 = \sum_{x \in M}(m_1(x) + m_2(x))'s$
- $m_1 - m_2 = \sum_{x \in M}(m_1(x) - m_2(x))'s$
- $nm_1 = \sum_{x \in M}(nm_1(x))'s$, com $n \in \mathbb{N}$
- $m_1 \neq m_2 \iff \exists x \in M, m_1(x) \neq m_2(x)$
- $m_1 \leq m_2 \iff \forall x \in M, m_1(x) \leq m_2(x)$

Para exemplificar, consideremos $M = x, y, z$. Nesse contexto, as seguintes expressões representam multiconjuntos sobre $M$: $1'x + 4'y$ e $2'x + 2'y + 1'z$.

As expressões de variáveis também precisam ser definidas sobre multiconjuntos. Uma expressão lógica é denotada por $expr$, e o conjuntos de variáveis associadas é dada por $Var(expr)$. Além disso, a avaliação de uma expressão pode ser realizada quando valores são atribuídos a essas variáveis. Assim, cada variável $v \in Var(expr)$ é substituída pelo valor correspondente de um conjunto de tipos $Type(v)$, resultando na avaliação $expr(g)$. Por exemplo, se $expr : x = y + 1$, então $Var(expr)$ é $\{x, y\}$ e a avaliação é obtida substituindo cada variável pelo seu valor correspondente em $g(v)$ [15] [17].

Com a noção de multiconjuntos estabelecida podemos definir as Redes de Petri Coloridas.

**Definição 2** *Uma Rede de Petri Colorida (RPC) [17] é formalmente definida pela tupla* $N = (L, T, \Sigma, C, G, A, E, I)$*, onde*

- $L = \{l_1, l_2, \ldots, l_n\}$ *é um conjunto finito de lugares;*
- $T = \{t_1, t_2, \ldots, t_q\}$ *é um conjunto finito de transições;*
- $\Sigma$ *é um conjunto finito de tipos distintos chamados de conjunto coloração;*

- $C : L \to \Sigma$ *é uma função de atribuição de cores que associa cada lugar* $l$ *a uma cor* $C(l)$*, onde* $C(l) \subseteq \Sigma$ *e* $\bigcup_{l \in L} C(l) = \Sigma$*;*
- $G : T \to expr$ *é uma função de restrição que associa uma expressão booleana expr a cada transição* $t$*. Logo,* $\forall t \in T$*,* $G(t)$ *é uma expressão, onde* $Type(Var(G(t))) \subseteq \sum$ *e* $Var(expr)$ *é o conjunto das variáveis em expr e* $Type(v)$ *é o conjunto de cores de* $v \in Var(G(t))$*;*
- $A \subseteq (L \times T) \cup (T \times L)$ *é um conjunto finito de arcos;*
- $E : A \to expr$ *é uma função que associa a cada arco* $a \in A$ *uma expressão expr. Logo,* $\forall a \in A$ *temos* $Type(Var(E(a))) = C(l)_{MS}$*, onde* $l$ *é um lugar adjacente ao arco a e* $C(l)_{MS}$ *indica todos os multiconjuntos de* $C(l)$*, e* $Type(Var(E(a))) \subseteq \Sigma$*, indicando que todas as variáveis na expressão assumem valores do conjunto de cores;*
- $I : L \to expr_{fechada}$ *é a função que estabelece a marcação inicial dos lugares em termos de expressões fechadas (sem variáveis), onde* $expr_{fechada}$ *é o conjunto de todas as expressões fechadas tal que* $\forall l \in L : Type(I(l)) = C(l)_{MS}$*.*

A marcação de uma Rede de Petri Colorida é determinada pela função $M$ sobre $L$, em que $M(l)$ representa a soma das cores no local $l$, definida formalmente por

$$M(l) = \sum_{i=1}^{n'} n_i \cdot c_i,$$

onde $n_i$ denota o número de marcas da cor $c_i$ no lugar $l$, e $u$ é o tamanho do conjunto de cores de $l$, ou seja, $u = |C(l)|$.

## 2.5 Verificação de Contratos

A verificação de contratos representa um passo crucial no desenvolvimento de sistemas seguros e confiáveis. Dessa forma, esta seção abordará a respeito da verificação formal, sua importância e consequência, também analisando a validação de Redes de Petri Coloridas.

### 2.5.1 Verificação formal

O avanço nos meios de comunicação trouxe um aumento significativo na interação entre pessoas e empresas por meios de sistemas computacionais, este fato fez com que as áreas de negociação, integração e interoperabilidade enfrentassem desafios, tendo a necessidade de evoluírem os métodos até então utilizados. Neste contexto pode-se analisar a importância dos contratos, que visam a garantia de acordos práticos e confiáveis em transações [4].

Um contrato descrito tanto em linguagem natural, como em linguagens de programação, está sujeito a conflitos por conta de possíveis ambiguidades em suas descrições, o que leva a quebras contratuais. Contratos são garantias de que há regras e deveres a serem cumpridos pelas partes envolvidas, sendo comum que expressem seus interesses e informe as leis que se adequam ao cenário, porém, ao combinar diferentes normas e interesses um conflito normativo (quando as cláusulas de um contrato se contradizem) pode ser gerado no contrato como um todo [4].

O blockchain é um ambiente seguro para execução de contratos inteligentes, simplificando significativamente o processo de desenvolvimento destes. A plataforma mais utilizada para a implementação de contratos inteligentes é o Ethereum, porém, há muitas vulnerabilidades na maioria dos contratos, considerando as limitações da linguagem de programação utilizada e, também, a complexidade do ambiente distribuído [18].

Sendo assim , para amenizar os problemas que podem ser causados por um conflito, e analisar as vulnerabilidades de um contrato, existem métodos capazes de formalizá-los, tendo como base a matemática e automação computacional. Alguns exemplos de representações precisas são: (I) lógicas modais; (II)lógicas temporais; (III)lógica deôntica; (IV) lógica dinâmica. Esses métodos compõem a chamada Verificação Formal, que na área de computação é uma abordagem rigorosa e matemática que verifica a correção de sistemas [4].

Analisando eventos anteriores, é possível ver exemplos destas vulnerabilidades como, o ataque DAO em junho de 2016, um ataque cibernético que explorou uma vulnerabilidade de recursividade no DAO, permitindo o roubo de mais de três milhões de ETH. Tempos depois uma vulnerabilidade de autodestruição surgiu na carteira Parity multising da Parity Technologies(empresa de desenvolvimento de software para blockchain e criptomoedas) em 2017, levando ao desvio de mais de 500 mil ETH, mesmo sendo um acidente não intencional causou um grande prejuízo [12].

Uma análise de segurança realizada por Atzei et al. usando a ferramenta MAIAN revelou que quase 1 milhão de contratos inteligentes estavam em risco A análise identificou 34.200 contratos inteligentes vulneráveis, representando um potencial para perdas significativas. Diante de prejuízos tão impactantes, a questão de como assegurar a segurança e a confiabilidade dos contratos inteligentes tornou-se um foco de atenção crucial [18].

Para elevar a segurança dos contratos inteligentes foram propostos métodos formais e informais. Apesar das técnicas informais permitirem fazer testes para determinados cenários em um contrato inteligente, não são confiáveis, visto que não são adequadas para verificar propriedades específicas que definem sua correção, um exemplo disso é a ausência de vulnerabilidades de overflow de inteiros ou a liberdade de impasses. Tendo esta perspectiva, as

técnicas formais de verificação demonstram mais eficacia [12].

No contexto deste trabalho, os contratos inteligentes da plataforma Ethereum serão o foco da formalização e verificação. Estes contratos podem lidar com diferentes lógicas de negócios para que assim seja aproveitado o máximo da capacidade da plataforma, tornando o Ethereum a maior plataforma de desenvolvimento de blockchain atual. No entanto, assim como sua boa utilidade, há também falhas de seguranças que devem ser tratadas rigidamente para diminuir os ataques e garantir que a descrição do contrato seja coerente além de segura. Com a verificação formal é possível analisar a exatidão dos contratos inteligentes se baseando em métodos formais de validação [23]

### 2.5.2 Verificação em Redes de Petri

Considerando a quantidade e importância dos ativos que circulam nas blockchains, segurança é um quesito importante a ser considerado nas plataformas que o utilizam. Os contratos inteligentes possuem a maioria das vulnerabilidades que possibilitam violações, sendo assim, é crucial haver rigorosas verificações antes que os contratos sejam implantados no sistema [12].

Em [10] é proposto uma solução formal para contratos inteligentes baseando-se em Redes de Petri Coloridas (CPN), visto que estas entregam uma boa descrição da lógica e do comportamento do contrato, além disso, ao se estabelecer conjuntos de diferentes cores é possível aprimorar a descrição e capacidade de abstração de um sistema. O artigo destaca a importância dos contratos e suas vulnerabilidades, propondo então um método automatizado de modelagem CPN, utilizando ferramentas CPN para simulação dinâmica dos modelos lógicos e de execução de contratos inteligentes, validando a eficácia do método proposto realizando experimentos com contratos inseguros.

Também em [18] é proposto um método formal de verificação de contratos inteligentes baseado nas CPNs. Em um primeiro momento é utilizado ferramentas CPN para modelar os contratos e seus atacantes, com esta ferramenta é possível analisar as etapas da execução do contrato. Em um segundo momento, a lógica de temporização de computação aprimorada ASK-CTL é combinada com ferramentas de espaço de estado para efetuar a verificação do modelo, sendo assim, vulnerabilidades em contratos inteligentes considerando o comportamento do usuário podem ser encontradas.

Dessa forma, será utilizado neste trabalho as Redes de Petri Coloridas (CPNs) para a verificação formal dos contratos inteligentes. Com este método de verificação é possível combinar o poder de análise das Redes de Petri com a expressividade das linguagens de programação, tornando-o uma modelagem e verificação adequada para sistemas grandes e

complexos.

# REFERÊNCIAS


[1]

[2] Maher Alharby and Aad Van Moorsel. Blockchain-based smart contracts: A systematic mapping study. *arXiv preprint arXiv:1710.06372*, 2017.

[3] Mohammed AlShamsi, Mostafa Al-Emran, and Khaled Shaalan. A systematic review on blockchain adoption. *Applied Sciences*, 12(9):4245, 2022.

[4] Wellington Aparecido Della Mura. Detecção de conflitos em contratos multilaterais. Master's thesis, Universidade Estadual de Londrina, Londrina, Brasil, 2016.

[5] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 3(37):2–1, 2014.

[6] Contracts365 by Jessica Alden. Everything you need to know about e-contracts. 18/09/2023.

[7] Janette Cardoso and Robert Valette. *Redes de petri*. Editora da UFSC Florianópolis, 1997.

[8] Massimo Di Pierro. What is the blockchain? *Computing in Science & Engineering*, 19(5):92–95, 2017.

[9] DocuSign. Electronic contracts (e-contracts), 2022. 18/09/2023.

[10] Wang Duo, Huang Xin, and Ma Xiaofeng. Formal analysis of smart contract based on colored petri nets. *IEEE Intelligent Systems*, 35(3):19–30, 2020.

[11] Carlos Renato Lisboa Francês. Introdução às redes de petri. *Laboratório de Computação Aplicada, Universidade Federal do Pará*, 2003.

[12] Ikram Garfatta, Kaïs Klai, Mohamed Graïet, and Walid Gaaloul. Model checking of vulnerabilities in smart contracts: A solidity-to-cpn approach. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, pages 316–325, 2022.

[13] Vijay Gehlot. From petri nets to colored petri nets: A tutorial introduction to nets based formalism for modeling and simulation. In *2019 Winter Simulation Conference (WSC)*, pages 1519–1533. IEEE, 2019.

[14] Guido Governatori, Florian Idelberger, Zoran Milosevic, Regis Riveret, Giovanni Sartor, and Xiwei Xu. On legal contracts, imperative and declarative smart contracts, and blockchain systems. *Artificial Intelligence and Law*, 26:377–409, 2018.

[15] Kurt Jensen. *Coloured Petri nets: basic concepts, analysis methods and practical use*, volume 1. Springer Science & Business Media, 1996.

[16] Jiao Jiao, Shuanglong Kan, Shang-Wei Lin, David Sanan, Yang Liu, and Jun Sun. Semantic understanding of smart contracts: Executable operational semantics of solidity. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1695–1712, 2020.

[17] Danilo Yudi Futata Kassuya. Transformação entre contratos inteligentes e redes de petri, 2023.

[18] Zhentian Liu and Jing Liu. Formal verification of blockchain smart contract based on colored petri net models. In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, volume 2, pages 555–560. ieee, 2019.

[19] William Mougayar. *The business blockchain: promise, practice, and application of the next Internet technology*. John Wiley & Sons, 2016.

[20] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, Dec 2008. Accessed: 2015-07-01.

[21] Carl Adam Petri and Wolfgang Reisig. Petri net. *Scholarpedia*, 3(4):6477, 2008.

[22] Andrea M Rozario and Miklos A Vasarhelyi. Auditing with smart contracts. *International Journal of Digital Accounting Research*, 18, 2018.

[23] Tianyu Sun and Wensheng Yu. A formal verification framework for security issues of blockchain smart contracts. *Electronics*, 9(2):255, 2020.

[24] Nick Szabo. Formalizing and securing relationships on public networks. *First Monday*, 2(9), Sep. 1997.

[25] Dejan Vujicic, Dijana Jagodic, and Sinisa Randic. Blockchain technology, bitcoin, and ethereum: A brief overview. pages 1–6, 03 2018.

[26] Maximilian Wohrer and Uwe Zdun. Smart contracts: security patterns in the ethereum ecosystem and solidity. In *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, pages 2–8, 2018.

[27] Maximilian Wohrer and Uwe Zdun. Smart contracts: security patterns in the ethereum ecosystem and solidity. In *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, pages 2–8. IEEE, 2018.