**Capítulo 8**  
**Contratos Inteligentes**

Este capítulo introduz contratos inteligentes. Este conceito não é novo; no entanto, com o advento da tecnologia blockchain, o interesse nessa ideia foi reavivado. Os contratos inteligentes são agora uma área contínua e intensa de pesquisa no espaço da blockchain. Muitas blockchains surgiram que oferecem suporte a contratos inteligentes.

Devido a benefícios como aumento de segurança, confiabilidade, descentralização, economia de custos e transparência que os contratos inteligentes podem trazer para muitas indústrias (especialmente a indústria financeira), os contratos inteligentes são vistos como uma tecnologia que tem o potencial de mudar bilhões de vidas.

Neste capítulo, abordaremos os seguintes tópicos:

* Introdução aos contratos inteligentes
* Contratos Ricardianos
* Modelos de contrato inteligente
* Oráculos
* Implantação de contratos inteligentes
* O DAO
* Avanços na tecnologia de contratos inteligentes

**Introdução aos contratos inteligentes**  
A sociedade humana é baseada em contratos sociais. Historicamente, estamos acostumados a acordos e contratos feitos entre indivíduos ou empresas. Historicamente, tivemos contratos escritos em pedra, madeira e papel. Então, na era do computador, surgiram contratos digitais. No entanto, os contratos digitais são centralizados e não confiáveis. Uma entidade poderosa poderia forçar sua saída do contrato prematuramente, influenciar o contrato e suas partes de uma forma ou de outra, ou não cumprir sua parte do acordo.

Além disso, eles são apenas legalmente executáveis, e partes mal-intencionadas às vezes podem enganar o sistema legal. No entanto, com o advento da blockchain, contratos descentralizados, mais seguros e automaticamente executáveis passaram a existir, onde nenhuma parte única pode desviar ou exercer poder para influenciar a execução do contrato. Como você pode imaginar, tal tecnologia pode mudar, para melhor, a vida de bilhões de pessoas.

**Definições**  
Nick Szabo teorizou pela primeira vez os contratos inteligentes na década de 1990, em um artigo chamado *Formalizing and Securing Relationships on Public Networks*. Essa teoria foi apresentada quase 20 anos antes do verdadeiro potencial e benefícios dos contratos inteligentes serem apreciados, ou seja, antes da invenção do Bitcoin e do subsequente desenvolvimento de outras plataformas de blockchain mais avançadas, como o Ethereum.

Os contratos inteligentes são descritos por Szabo da seguinte maneira:

Um contrato inteligente é um protocolo de transação eletrônica que executa os termos de um contrato. Os objetivos gerais são satisfazer condições contratuais comuns (como termos de pagamento, garantias, confidencialidade e até mesmo execução), minimizar exceções tanto maliciosas quanto acidentais, e minimizar a necessidade de intermediários confiáveis. Os objetivos econômicos relacionados incluem reduzir perdas por fraude, arbitragens e custos de execução, e outros custos de transação.

A funcionalidade de contrato inteligente foi implementada de forma limitada no Bitcoin em 2009. O Bitcoin oferece suporte a uma linguagem de script restrita chamada *Script*, que permite a transferência de bitcoins entre usuários. No entanto, esta não é uma linguagem Turing-completa e não oferece suporte ao desenvolvimento de programas arbitrários. Pode ser considerada como uma calculadora de função limitada com apenas operações aritméticas simples, enquanto os contratos inteligentes podem ser considerados computadores de uso geral que suportam a escrita de qualquer programa.

A seguir está minha tentativa de fornecer uma definição geral abrangente de um contrato inteligente:

Um contrato inteligente é um programa de computador seguro e imparável que representa um acordo que é automaticamente executável e aplicável.

Dissecar essa definição revela que um contrato inteligente é, fundamentalmente, um programa de computador escrito em uma linguagem que um computador ou máquina-alvo pode entender. Ele também exige alguns requisitos para que um contrato inteligente funcione de forma eficaz.

**Propriedades**  
Um contrato inteligente possui as seguintes propriedades:

* **Automaticamente executável**: É autoexecutável em uma blockchain quando certas condições satisfazem instruções codificadas sem exigir qualquer intervenção.
* **Aplicável (executável)**: Isso significa que todos os termos contratuais são realizados conforme especificado e esperado, mesmo na presença de adversários. Preferencialmente, os contratos inteligentes não devem depender de nenhum método tradicional de execução. Em vez disso, devem funcionar com o princípio de que o código é a lei, o que significa que não há necessidade de um árbitro ou de uma terceira parte para aplicar, controlar ou influenciar a execução de um contrato inteligente.
* **Seguro**: Isso significa que os contratos inteligentes são à prova de adulteração (ou resistentes à adulteração) e operam com garantias de segurança. A blockchain subjacente geralmente fornece essas garantias de segurança; no entanto, a linguagem de programação do contrato inteligente e o próprio código do contrato devem ser corretos, válidos e verificados.
* **Determinístico**: A característica determinística garante que os contratos inteligentes sempre produzam a mesma saída para uma entrada específica. Essa propriedade permite que um contrato inteligente seja executado por qualquer nó em uma rede e atinja o mesmo resultado. Se o resultado diferir mesmo que ligeiramente entre os nós, então um consenso não pode ser alcançado, e todo o paradigma de consensos distribuídos na blockchain pode falhar. Além disso, também é desejável que a própria linguagem de contrato seja determinística, garantindo assim a integridade e a estabilidade dos contratos inteligentes.
* **Semanticamente sólido**: Isso significa que são completos e significativos tanto para pessoas quanto para computadores.
* **Imparável**: Isso significa que adversários ou condições desfavoráveis não podem afetar negativamente a execução de um contrato inteligente. Quando os contratos inteligentes são executados, eles completam sua performance de forma determinística em um tempo finito.

Pode-se argumentar que as quatro primeiras propriedades são exigidas como mínimo, enquanto as duas últimas podem não ser necessárias ou aplicáveis em alguns cenários e podem ser relaxadas. Por exemplo, um contrato de derivativos financeiros talvez não precise ser semanticamente sólido e imparável, mas deve ao menos ser automaticamente executável, aplicável, determinístico e seguro.

Os contratos inteligentes são autoexecutáveis, ao contrário de legalmente executáveis. Essa ideia pode soar como o sonho de um libertário, mas é totalmente possível e está em linha com o verdadeiro espírito dos contratos inteligentes.

As plataformas de blockchain desempenham um papel vital ao fornecer a rede subjacente necessária, com garantias de segurança exigidas para executar contratos inteligentes.

Um exemplo é algumas funções matemáticas em JavaScript, que podem produzir resultados diferentes para a mesma entrada em navegadores diferentes e podem, por sua vez, levar a vários bugs. Esse cenário é inaceitável em contratos inteligentes, pois se os resultados forem inconsistentes entre os nós, então um consenso nunca será alcançado.

Por outro lado, uma escritura de propriedade precisa ser semanticamente sólida e completa; portanto, para que ela seja implementada como um contrato inteligente, a linguagem em que está escrita deve ser compreendida tanto por computadores quanto por pessoas.

Ainda assim, ela proporcionará benefícios mais significativos a longo prazo se as propriedades de segurança e de ser imparável forem incluídas na definição de contrato inteligente. Essa inclusão permitirá que os pesquisadores se concentrem nesses aspectos desde o início e ajudará a construir fundamentos sólidos sobre os quais mais pesquisas poderão ser baseadas.

Há também uma sugestão de alguns pesquisadores de que os contratos inteligentes não precisam ser automaticamente executáveis; em vez disso, eles podem ser chamados de *automatizáveis*, devido à entrada manual humana necessária em alguns cenários. Por exemplo, a verificação manual de um prontuário médico por um profissional médico qualificado pode ser necessária. Em tais casos, abordagens totalmente automatizadas podem não funcionar da melhor forma. Embora seja verdade que, em alguns casos, a intervenção e o controle humanos são úteis, eles não são necessários. Para que um contrato seja realmente inteligente, na minha opinião, ele deve ser totalmente automatizado. Certas entradas que precisam ser fornecidas por pessoas podem e devem também ser automatizadas. Oráculos podem ser usados para esse propósito. Discutiremos oráculos com mais detalhes mais adiante neste capítulo.

**Aplicação no mundo real**  
Os contratos inteligentes geralmente operam gerenciando seu estado interno usando um modelo de máquina de estados fornecido pela blockchain subjacente. Isso permite o desenvolvimento de uma estrutura prática para programar contratos inteligentes, onde o estado de um contrato inteligente é avançado com base em alguns critérios e condições predefinidos.

Há também um debate contínuo sobre se o código de computador é aceitável como um contrato convencional em um tribunal de justiça. Um contrato inteligente é diferente, na forma, da prosa jurídica tradicional, embora represente e imponha todas as cláusulas contratuais exigidas. Ainda assim, um tribunal de justiça não entende código de computador. Esse dilema levanta várias questões sobre como um contrato inteligente pode ter validade legal: ele pode ser desenvolvido de tal forma que seja prontamente aceitável e compreensível em um tribunal? É possível implementar resolução de disputas dentro do próprio código? Além disso, requisitos regulatórios e de conformidade são outros tópicos que precisam ser abordados antes que os contratos inteligentes possam se tornar tão eficientes quanto os documentos legais tradicionais.

A legalidade dos contratos inteligentes é incerta em muitas jurisdições. Ainda assim, um desenvolvimento recente interessante neste campo tornou criptoativos e contratos inteligentes válidos sob a lei inglesa, ao reconhecer criptoativos como propriedade comercializável e contratos inteligentes como acordos executáveis. Esse anúncio foi feito pela UK Jurisdiction Taskforce (UKJT) do Lawtech Delivery Panel. A Suíça também tem um ambiente favorável para ativos digitais criptográficos. Outros países “cripto-amigáveis” incluem Singapura, El Salvador, Porto Rico e Malta.

Mais informações sobre isso, incluindo a declaração legal completa, estão disponíveis aqui:  
<https://technation.io/news/uk-takes-significant-step-in-legal-certainty-for-smart-contracts-and-cryptocurrencies/>

Mesmo que os contratos inteligentes sejam chamados de "inteligentes", eles apenas fazem o que foram programados para fazer. Essa própria característica dos contratos inteligentes garante que eles produzam a mesma saída toda vez que são executados. A natureza determinística dos contratos inteligentes é altamente desejável em plataformas de blockchain devido aos requisitos de consistência.

Agora, isso dá origem a um problema no qual surge uma grande lacuna entre o mundo real e o mundo da blockchain. Nessa situação, a linguagem natural não é compreendida pelo contrato inteligente e, da mesma forma, o código de computador não é aceitável no mundo natural. Assim, surgem algumas questões: como contratos do mundo real podem ser implantados em uma blockchain? Como essa ponte entre o mundo real e o mundo dos contratos inteligentes pode ser construída?

Essas questões abrem várias possibilidades, como tornar o código do contrato inteligente legível não apenas por máquinas, mas também por pessoas. Se humanos e máquinas puderem entender o código escrito em um contrato inteligente, ele pode se tornar aceitável em situações legais, em oposição a ser apenas um pedaço de código que ninguém entende, exceto programadores. Essa propriedade desejável é uma área madura para pesquisa, e um esforço significativo de pesquisa tem sido despendido nessa área para responder perguntas sobre a semântica, o significado e a interpretação de contratos inteligentes.

Já foi feito algum trabalho para descrever contratos em linguagem natural formalmente, combinando tanto o código do contrato inteligente quanto os contratos em linguagem natural, vinculando termos contratuais a elementos compreensíveis por máquinas. Isso é alcançado usando uma linguagem de marcação chamada LKIF (*Legal Knowledge Interchange Format*), que é um esquema XML para representar teorias e provas. Foi desenvolvido sob o projeto Estrella em 2008.

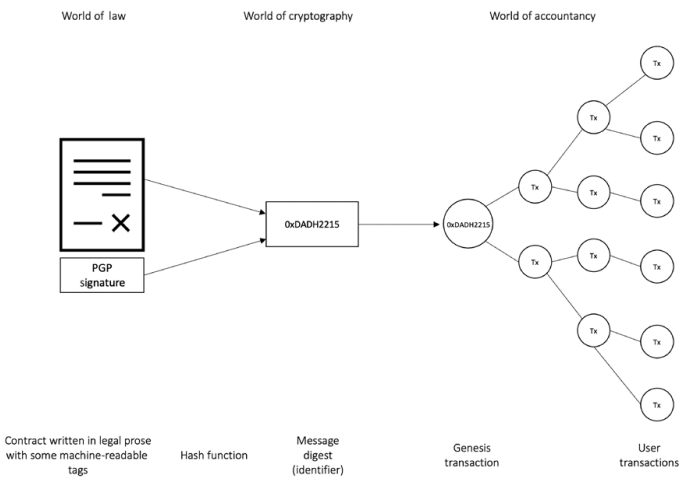
Ian Grigg abordou essa questão da interpretação com sua invenção dos contratos Ricardianos, que introduziremos na próxima seção.

**Contratos Ricardianos**  
Contratos Ricardianos foram inicialmente utilizados em um sistema de negociação e pagamento de títulos chamado Ricardo. A ideia fundamental por trás desse contrato é escrever um documento que seja compreendido e aceito tanto por um tribunal quanto por um software de computador. Os contratos Ricardianos abordam o desafio da emissão de valor pela internet, identificando o emissor e capturando todos os termos e cláusulas do contrato em um documento, tornando-o aceitável como um contrato legalmente vinculativo.

Um contrato Ricardiano possui várias das seguintes propriedades:

* É um contrato oferecido por um emissor aos detentores
* É um direito valioso mantido pelos detentores e gerenciado pelo emissor
* Pode ser facilmente lido por pessoas (como um contrato em papel)
* Pode ser lido por programas (analisável, como um banco de dados)
* É digitalmente assinado
* Contém as chaves e informações do servidor
* Está aliado a um identificador único e seguro

Na prática, os contratos são implementados produzindo um único documento que contém os termos do contrato em prosa legal e as tags legíveis por máquina necessárias. Esse documento é assinado digitalmente pelo emissor usando sua chave privada. Esse documento é então condensado por meio de uma função de resumo de mensagem para produzir um hash pelo qual o documento pode ser identificado. Esse hash é então usado e assinado pelas partes durante a execução do contrato para vincular cada transação ao hash identificador, servindo, portanto, como evidência de intenção. Isso é representado no próximo diagrama e geralmente é chamado de modelo “gravata-borboleta” (*bowtie model*):



**Figura 8.1: O modelo de gravata-borboleta de contrato Ricardiano**

A informação anterior é baseada na definição original de Ian Grigg em  
<http://iang.org/papers/ricardian_contract.html>

O diagrama mostra vários elementos:

1. O Mundo da Lei está no lado esquerdo, de onde o documento se origina. Este documento é um contrato escrito em prosa legal com algumas tags legíveis por máquina.
2. Este documento é então condensado (hash).
3. O resumo de mensagem resultante é usado como um identificador em todo o Mundo da Contabilidade, conforme mostrado no lado direito do diagrama.

O elemento do Mundo da Contabilidade representa quaisquer sistemas de contabilidade, negociação e informação que estão sendo usados nos negócios para executar várias operações comerciais. A ideia por trás deste fluxo é que o resumo de mensagem gerado ao condensar o documento seja usado primeiro em uma chamada transação gênese (*genesis transaction*), ou transação inicial, e depois seja usado em todas as transações como um identificador durante a execução operacional do contrato. Dessa forma, cria-se uma ligação segura entre o contrato escrito original e cada transação no Mundo da Contabilidade.

Um contrato Ricardiano é diferente de um contrato inteligente no sentido de que um contrato inteligente não inclui nenhum documento contratual e é focado puramente na execução do contrato. Um contrato Ricardiano, por outro lado, está mais preocupado com a riqueza semântica e a produção de um documento que contenha a prosa legal contratual. A semântica de um contrato legal pode ser dividida em dois tipos:

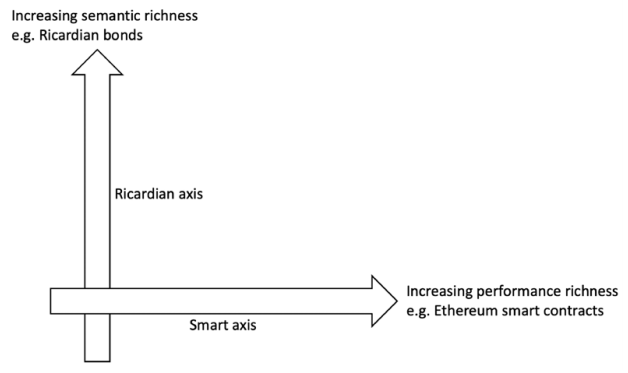
* A **semântica operacional** define a execução, correção e segurança do contrato.
* A **semântica denotacional** está relacionada ao verdadeiro “significado legal” do contrato, ou seja, o acordo jurídico.

Faz sentido categorizar contratos inteligentes com base na diferença entre as semânticas, mas é melhor considerar um contrato inteligente como uma entidade independente que é capaz de codificar prosa legal e código (lógica de negócios).

No Bitcoin, uma implementação simples de contratos inteligentes básicos (lógica condicional) pode ser observada, totalmente orientada à execução e desempenho do contrato, enquanto um contrato Ricardiano está mais voltado à produção de um documento que seja compreendido por humanos, com algumas partes compreensíveis por um programa de computador.

Alguns pesquisadores diferenciaram entre código de contrato inteligente e contratos legais inteligentes (*smart legal contracts*), onde um contrato inteligente está apenas preocupado com a execução do contrato.

Isso pode ser visto como semântica legal versus desempenho operacional (*semantics versus performance*), conforme mostrado no diagrama a seguir:



**Figura 8.2: Diagrama explicando que desempenho versus semântica é uma questão ortogonal, conforme descrito por Ian Grigg; está ligeiramente modificado para mostrar exemplos de diferentes tipos de contratos em ambos os eixos**

O diagrama mostra que contratos Ricardianos são mais ricos semanticamente, enquanto contratos inteligentes são mais ricos em desempenho.

Um contrato inteligente é composto por ambos os elementos (desempenho e semântica) incorporados juntos, o que completa o modelo ideal de um contrato inteligente.

Um contrato Ricardiano pode ser representado como uma tupla de três objetos, a saber: *prosa*, *parâmetros* e *código*. *Prosa* representa o contrato legal em linguagem natural, *código* representa o programa que é uma representação compreensível por computador da prosa legal, e *parâmetros* conectam as partes apropriadas do contrato legal ao código equivalente.

Esse conceito foi inicialmente proposto por Ian Grigg em seu artigo *On the intersection of Ricardian and Smart Contracts*. O artigo está disponível em:  
<https://iang.org/papers/intersection_ricardian_smart.html>

Contratos Ricardianos foram implementados em muitos sistemas, como o CommonAccord (<http://www.commonaccord.org>) e o OpenBazaar (<https://github.com/OpenBazaar>).

Agora que entendemos o que são contratos Ricardianos, vejamos o conceito de **modelos de contrato inteligente**. Estes foram construídos com base na ideia dos contratos Ricardianos, e visam apoiar a gestão de todo o ciclo de vida de contratos legais inteligentes.

**Modelos de contrato inteligente**  
Contratos inteligentes podem ser implementados em qualquer setor onde sejam necessários, mas os casos de uso mais populares se relacionam ao setor financeiro. Isso ocorre porque a blockchain encontrou muitos casos de uso na indústria financeira primeiro e, portanto, despertou enorme interesse de pesquisa na indústria financeira muito antes de outras áreas. Trabalhos recentes no espaço de contratos inteligentes, específicos para o setor financeiro, propuseram a ideia de modelos de contrato inteligente. A ideia é construir modelos padrão que forneçam uma estrutura para apoiar acordos legais para instrumentos financeiros.

O artigo também sugeriu que Linguagens Específicas de Domínio (*Domain-Specific Languages* – DSLs) deveriam ser construídas para apoiar o design e a implementação de modelos de contrato inteligente. Uma linguagem chamada *Common Language for Augmented Contract Knowledge* (CLACK) foi proposta, e pesquisas começaram a ser desenvolvidas sobre essa linguagem. Essa linguagem é destinada a ser muito rica e espera-se que ofereça uma grande variedade de funções, desde o suporte à prosa legal até a capacidade de ser executada em múltiplas plataformas e funções criptográficas.

Clack et al. também realizaram trabalhos para desenvolver modelos de contrato inteligente que apoiem contratos inteligentes legalmente executáveis. Essa proposta foi discutida em seu artigo de pesquisa *Smart Contract Templates: essential requirements and design options*.

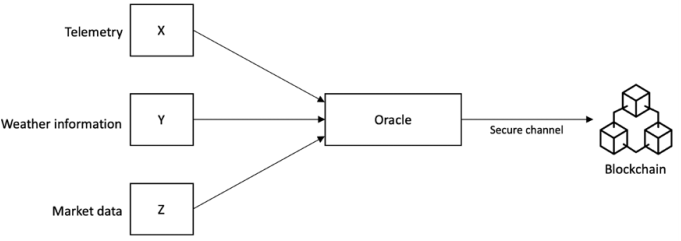
O objetivo principal deste artigo é investigar como a prosa legal poderia ser vinculada ao código usando uma linguagem de marcação. Ele também cobre como acordos legais inteligentes podem ser criados, formatados, executados e serializados para armazenamento e transmissão. Este trabalho está em andamento e permanece uma área aberta para mais pesquisas e desenvolvimento.

Contratos na indústria financeira não são um conceito novo, e várias DSLs já estão em uso na indústria de serviços financeiros para fornecer uma linguagem específica para um domínio específico. Por exemplo, há DSLs disponíveis que apoiam o desenvolvimento de produtos de seguros, representam derivativos de energia, ou estão sendo usados para construir estratégias de negociação.

Christopher D. Clack et al. propuseram essa ideia em seu artigo publicado em 2016, chamado *Smart Contract Templates: foundations, design landscape and research directions*.  
O artigo está disponível em: <https://arxiv.org/pdf/1608.00771.pdf>

**Oráculos**  
Oráculos são um componente essencial do ecossistema de contratos inteligentes e blockchain. A limitação dos contratos inteligentes é que eles não podem acessar dados externos porque blockchains são sistemas fechados, sem qualquer acesso direto ao mundo real. Esses dados externos podem ser necessários para controlar a execução de alguma lógica de negócios no contrato inteligente, por exemplo, o preço de uma ação de um produto financeiro que é necessário pelo contrato para liberar pagamentos de dividendos. Em tais situações, oráculos podem ser usados para fornecer dados externos aos contratos inteligentes. Um oráculo pode ser definido como uma interface que entrega dados de uma fonte externa aos contratos inteligentes. Oráculos são entidades confiáveis que usam um canal seguro para transferir dados fora da cadeia (*off-chain*) para um contrato inteligente.

O diagrama a seguir mostra um modelo genérico de um oráculo e ecossistema de contrato inteligente:



**Figura 8.3: Um modelo genérico de um ecossistema de oráculo e contrato inteligente com várias fontes de dados**

Dependendo da indústria e dos requisitos do caso de uso, oráculos podem fornecer diferentes tipos de dados, variando de relatórios meteorológicos, notícias do mundo real e ações corporativas a dados vindos de um dispositivo da Internet das Coisas (*IoT*).

Uma lista de alguns dos casos de uso comuns de oráculos é mostrada aqui:

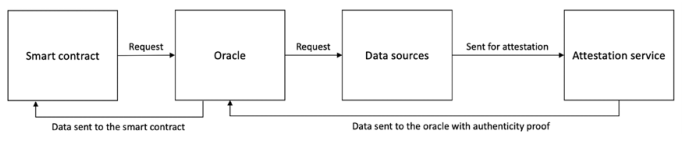
| **Tipo de dado** | **Exemplos** | **Caso de uso** |
| --- | --- | --- |
| Dados de mercado | Cotações ao vivo de instrumentos financeiros: taxas de câmbio, desempenho, precificação e dados históricos de commodities, índices, ações, títulos e moedas | Aplicativos descentralizados relacionados a serviços financeiros, por exemplo, exchanges descentralizadas e DeFi |
| Eventos políticos | Resultados de eleições | Mercados de previsão |
| Informações de viagem | Programação e atrasos de voos | Aplicativos descentralizados de seguros |
| Informações climáticas | Dados de enchentes, temperatura e chuva | Aplicativos descentralizados de seguros |
| Esportes | Resultados de partidas de futebol, críquete e rúgbi | Mercados de previsão |
| Telemetria | Dispositivos de hardware IoT, dados de sensores, localização de veículos e dados de rastreamento de veículos | Aplicativos descentralizados de seguros; gestão de frotas |

Há diferentes métodos utilizados por oráculos para escrever dados em uma blockchain, dependendo do tipo de blockchain utilizado. Por exemplo, em uma blockchain Bitcoin, um oráculo pode escrever dados em uma transação específica, e um contrato inteligente pode monitorar essa transação na blockchain e ler os dados. Outros métodos incluem armazenar os dados obtidos no armazenamento de um contrato inteligente, que pode então ser acessado por outros contratos inteligentes na blockchain via requisições entre contratos, dependendo da plataforma. Por exemplo, no Ethereum, isso pode ser alcançado usando chamadas de mensagens.

A mecânica padrão de como os oráculos funcionam é apresentada a seguir:

1. Um contrato inteligente envia uma solicitação de dados a um oráculo.
2. A solicitação é executada, e os dados necessários são requisitados da fonte. Existem vários métodos para solicitar dados da fonte. Esses métodos normalmente envolvem invocar APIs fornecidas pelo provedor de dados, chamar um serviço web, ler de um banco de dados (por exemplo, em casos de uso de integração empresarial onde os dados necessários podem existir em um sistema legado local da empresa), ou solicitar dados de outra blockchain. As fontes podem ser qualquer provedor de dados externo fora da cadeia, na internet ou em uma rede empresarial interna.
3. Os dados são enviados a um notário para gerar prova criptográfica (geralmente uma assinatura digital) dos dados solicitados para provar sua validade (autenticidade). Geralmente, o TLSNotary é usado para esse propósito (<https://tlsnotary.org>). Outras técnicas incluem *Android proofs*, *Ledger proofs* e provas assistidas por hardware confiável, que explicaremos em breve.
4. Os dados com a prova de validade são enviados ao oráculo.
5. Os dados solicitados com sua prova de autenticidade podem ser opcionalmente salvos em um sistema de armazenamento descentralizado, como o Swarm ou o IPFS, e podem ser usados pelo contrato inteligente/blockchain para verificação. Isso é especialmente útil quando as provas de autenticidade são grandes e enviá-las aos contratos inteligentes solicitantes (armazená-las na cadeia) não é viável.
6. Por fim, os dados, com a prova de validade, são enviados ao contrato inteligente.

Esse processo pode ser visualizado no seguinte diagrama:



**Figura 8.4: Um fluxo de dados genérico de oráculo**

O diagrama anterior mostra o fluxo de dados genérico de uma solicitação de dados de um contrato inteligente para o oráculo. O oráculo então solicita os dados da fonte de dados, que são enviados ao serviço de atestação para notarização. Os dados são enviados ao oráculo com prova de autenticidade. Por fim, os dados são enviados ao contrato inteligente com prova criptográfica (prova de autenticidade) de que os dados são válidos.

Devido aos requisitos de segurança, os oráculos também devem ser capazes de assinar digitalmente ou atestar digitalmente os dados para provar que os dados são autênticos. Essa prova é chamada de prova de validade ou prova de autenticidade.

Contratos inteligentes se inscrevem em oráculos. Contratos inteligentes podem puxar dados dos oráculos ou os oráculos podem empurrar dados para os contratos inteligentes. Também é necessário que os oráculos não possam manipular os dados que fornecem e devem ser capazes de fornecer dados factuais. Mesmo que os oráculos sejam confiáveis (devido à prova de autenticidade dos dados associada), ainda é possível que, em alguns casos, os dados estejam incorretos devido a manipulação ou falha no sistema. Portanto, os oráculos não devem ser capazes de modificar os dados. Essa validação pode ser fornecida usando vários esquemas de prova criptográfica. A seguir, introduziremos diferentes mecanismos para produzir prova criptográfica da autenticidade dos dados.

**Provas assistidas por software e rede**

Como o nome sugere, esses tipos de prova fazem uso de software, protocolos de rede, ou uma combinação de ambos para fornecer provas de validade. Um dos principais exemplos de tais provas é o **TLSNotary**.

**TLSNotary**

O TLSNotary é uma tecnologia desenvolvida para ser usada principalmente no projeto *PageSigner* (<https://old.tlsnotary.org/pagesigner.html>) para fornecer notarização de páginas web. Esse mecanismo também pode ser usado para fornecer os serviços de segurança exigidos aos oráculos. Esse protocolo fornece uma prova irrefutável a um auditor de que tráfego web específico foi trocado entre um cliente e um servidor. Ele é baseado no *Transport Layer Security* (TLS), que é um mecanismo de segurança padrão que permite comunicação segura e bidirecional entre hosts. É amplamente utilizado na internet para proteger sites e permitir tráfego HTTPS.

Uma discussão sobre os detalhes internos desse protocolo está além do escopo deste livro. Leitores interessados podem ler o documento de padrões em <http://www.ietf.org/rfc/rfc2246.txt>. O link fornecido é apenas para a versão 1.0 do TLS, pois o TLSNotary só oferece suporte ao TLS versão 1.0 ou 1.1.

A ideia central por trás do uso do TLSNotary é utilizar o recurso do protocolo de handshake do TLS, que permite a divisão da chave mestra TLS em três partes. Cada parte é alocada para o servidor, o auditado e o auditor. O provedor de serviços de oráculo (<https://provable.xyz>) torna-se o auditado, enquanto uma instância da Amazon Web Services (AWS), que é segura e restrita, serve como o auditor.

**Mecanismo baseado em TLS-N**

Esse mecanismo é um dos desenvolvimentos mais recentes nesse espaço. O TLS-N é uma extensão do TLS que fornece garantias seguras de não repúdio. Esse protocolo permite que você crie provas de conteúdo de uma sessão TLS que preservam a privacidade e não são interativas. Oráculos baseados em TLS-N não precisam confiar em nenhum hardware de terceiros como o Intel SGX ou um serviço do tipo TLSNotary para fornecer provas de autenticidade de conteúdo da web (dados) para a blockchain. Em contraste com o TLSNotary, esse esquema funciona com o mais recente padrão TLS 1.3, que permite maior segurança. Mais informações sobre esse protocolo estão disponíveis em: <https://eprint.iacr.org/2017/578.pdf>

**Provas assistidas por dispositivo de hardware**

Como o nome sugere, essas provas dependem de alguns elementos de hardware para fornecer prova de autenticidade. Em outras palavras, exigem hardware específico para funcionar. Diferentes mecanismos se enquadram nessa categoria, e os apresentaremos brevemente a seguir.

**Prova Android**

Essa prova depende da atestação por software do SafetyNet do Android e da atestação por hardware para criar um dispositivo comprovadamente seguro e auditável. O SafetyNet valida que um aplicativo Android genuíno está sendo executado em um dispositivo de hardware seguro, íntegro e não adulterado. A atestação por hardware valida que o dispositivo possui a versão mais recente do sistema operacional, o que ajuda a prevenir quaisquer explorações que existiam devido a vulnerabilidades nas versões anteriores do sistema operacional. Esse dispositivo seguro é então usado para buscar dados de fontes terceiras, garantindo conexões HTTPS à prova de adulteração. O próprio uso de um dispositivo comprovadamente seguro fornece a garantia e confiança (isto é, a prova de autenticidade) de que os dados são autênticos.

**Prova Ledger**

A prova Ledger depende das carteiras de hardware de criptomoedas construídas pela empresa Ledger (<https://www.ledger.com>). Duas carteiras de hardware — Ledger Nano S e Ledger Blue — podem ser usadas para essas provas. O principal propósito desses dispositivos é serem carteiras de criptomoedas seguras. No entanto, devido à segurança e flexibilidade fornecidas por esses dispositivos, eles também permitem que desenvolvedores construam outros aplicativos para esse hardware. Esses dispositivos executam um sistema operacional específico chamado *Blockchain Open Ledger Operating System* (BOLOS), que, por meio de diversas APIs em nível de kernel, permite a atestação do dispositivo e do código para fornecer um ambiente comprovadamente seguro.

O ambiente seguro fornecido pelo dispositivo também pode provar que os aplicativos que podem ter sido desenvolvidos por provedores de serviços de oráculo e que estão sendo executados no dispositivo são válidos, autênticos e estão de fato sendo executados no Ambiente de Execução Confiável (*Trusted Execution Environment* – TEE) do dispositivo Ledger. Esse ambiente, suportado tanto pela atestação de código quanto de dispositivo, fornece um ambiente que permite executar um aplicativo de terceiros em um ambiente de Ledger seguro e verificável, a fim de fornecer prova de autenticidade dos dados. Atualmente, esse serviço é usado pela Provable, um serviço de oráculo, para fornecer números aleatórios não adulterados a contratos inteligentes.

Mais informações sobre o SafetyNet e atestação por hardware estão disponíveis aqui:  
<https://developer.android.com/training/safetynet>  
<https://developer.android.com/training/articles/security-key-attestation.html>

As provas Ledger, Android e TLSNotary são utilizadas em oráculos Provable. A documentação oficial desses métodos pode ser encontrada em: <http://docs.provable.xyz>

Atualmente, como esses dispositivos não se conectam diretamente à internet, os dispositivos Ledger não podem ser usados para buscar dados da internet.

**Provas assistidas por hardware confiável**

Esse tipo de prova faz uso de hardware confiável, como TEEs (*Trusted Execution Environments* — Ambientes de Execução Confiáveis). Um exemplo importante de tal dispositivo de hardware é o **Intel SGX**. Uma abordagem geral utilizada nesse cenário é confiar nas garantias de segurança de uma execução segura e confiável fornecida pelo elemento ou enclave seguro do dispositivo TEE.

Um exemplo notável de prova assistida por hardware confiável é o **Town Crier** (<https://www.town-crier.org>), que fornece um feed de dados autenticado para contratos inteligentes. Ele usa o Intel SGX para fornecer uma garantia de segurança de que os dados solicitados vieram de uma fonte confiável existente.

O Town Crier também oferece um serviço de confidencialidade, que permite executar consultas confidenciais. A consulta para a solicitação de dados é processada dentro do Enclave SGX, que fornece uma garantia de execução confiável, e os dados solicitados são transmitidos usando uma conexão de rede segura via TLS, o que fornece garantias adicionais de integridade dos dados.

Um problema já pode ser visto aqui: a questão da confiança. Com os oráculos, estamos efetivamente confiando a uma terceira parte a tarefa de nos fornecer os dados corretos. E se essas fontes de dados se tornarem maliciosas, ou simplesmente por falha começarem a fornecer dados incorretos aos oráculos? E se o próprio oráculo falhar ou a fonte de dados parar de enviar dados? Esse problema pode então comprometer todo o modelo de confiança da blockchain. Esse fenômeno é chamado de **problema do oráculo na blockchain** (*Blockchain oracle problem*). Como confiar em uma terceira parte quanto à qualidade e autenticidade dos dados que ela fornece? Essa questão é especialmente real no mundo financeiro, onde os dados de mercado devem ser precisos e confiáveis.

Há várias formas propostas para superar esse problema. Essas soluções vão desde confiar em uma terceira parte reputada até o uso de oráculos descentralizados. Já discutimos algumas das técnicas de atestação anteriormente; no entanto, uma terceira parte, por falha genuína ou intenção maliciosa, ainda pode fornecer dados incorretos. Mesmo que sejam atestados posteriormente, os dados reais em si não são garantidos como corretos. Pode ser aceitável, para o projetista de um contrato inteligente em determinado caso de uso, aceitar dados de um oráculo fornecidos por uma terceira parte grande, reputada e confiável. Por exemplo, a fonte dos dados pode ser uma agência meteorológica confiável ou um sistema de informações de aeroporto que transmite diretamente os atrasos de voo, o que pode fornecer certo nível de confiança. No entanto, o problema da centralização permanece.

A tecnologia Intel SGX é desenvolvida pela Intel, que fornece um TEE baseado em hardware.  
Mais informações sobre isso estão disponíveis aqui: <https://software.intel.com/en-us/sgx>  
Ela é usada por diversos provedores de serviços de oráculo, como o Town Crier e o iExec (<https://docs.iex.ec/use-cases/iexec-doracle#the-iexec-solution-the-decentralized-oracle-doracle>)

Deve-se observar que, em todas as técnicas de prova mencionadas anteriormente, há muitos tipos de recursos usados para fornecer garantias de segurança, incluindo hardware, rede e software. A categorização apresentada aqui é baseada no elemento principal — hardware ou software — que desempenha um papel crítico no mecanismo de segurança geral para fornecer essa segurança.

**Tipos de oráculos de blockchain**

Com base na evolução da blockchain nos últimos anos, diversos tipos de oráculos de blockchain surgiram. Apresentamos anteriormente algumas informações de forma informal sobre eles, mas agora os definiremos formalmente.

Há vários tipos de oráculos de blockchain, variando de simples oráculos de software até oráculos descentralizados e assistidos por hardware mais complexos. De forma geral, podemos categorizar os oráculos em duas grandes classes: **oráculos de entrada** (*inbound oracles*) e **oráculos de saída** (*outbound oracles*).

**Oráculos de entrada (*Inbound oracles*)**

Esta classe representa os oráculos que recebem dados provenientes de serviços externos e os alimentam para o contrato inteligente. A seguir, discutiremos oráculos de software, de hardware e outros tipos de oráculos de entrada.

**Oráculos de software**

Esses oráculos são responsáveis por obter informações de serviços online na internet. Esse tipo de oráculo é normalmente utilizado para obter dados como informações meteorológicas, dados financeiros (por exemplo, preços de ações), informações de viagem e outros tipos de dados de provedores terceirizados. A fonte de dados também pode ser um sistema interno da empresa, que pode fornecer alguns dados específicos do domínio corporativo. Esses tipos de oráculos também podem ser chamados de oráculos padrão ou simples.

Um problema com esses oráculos é que, como se baseiam simplesmente em uma fonte online de dados, como um site, se essa fonte não estiver mais disponível ou tiver falhado por algum motivo, então o fornecimento de dados para o contrato inteligente na blockchain também falhará. No entanto, a redundância pode ser usada para tornar uma fonte de dados única tolerante a falhas.

**Oráculos de hardware**

Esse tipo de oráculo é utilizado para obter dados de fontes de hardware, como dispositivos IoT ou sensores. Isso é útil em casos de uso como contratos inteligentes relacionados a seguros, onde sensores de telemetria fornecem certas informações — por exemplo, velocidade e localização de veículos. Essas informações podem ser alimentadas no contrato inteligente responsável por avaliar pedidos de indenização e pagamentos, para decidir se aceita ou rejeita uma solicitação.

Esses oráculos são úteis em qualquer situação onde sejam necessários dados do mundo real provenientes de dispositivos físicos. No entanto, essa abordagem exige um mecanismo no qual os dispositivos de hardware sejam à prova de adulteração ou resistentes à adulteração. Esse nível de segurança pode ser alcançado fornecendo prova criptográfica (não repúdio e integridade) dos dados de um dispositivo IoT e um mecanismo antiadulteração no dispositivo, que o torne inutilizável se houver qualquer tentativa de adulteração.

O problema do oráculo na blockchain pode ser definido formalmente como o conflito de confiança entre oráculos presumivelmente confiáveis (fontes de dados terceiras confiáveis) e uma blockchain completamente confiável e sem necessidade de confiança (*trustless*).

**Oráculos de computação (*Computation oracles*)**

Esses oráculos permitem que cálculos intensivos sejam realizados fora da cadeia (*off-chain*). Como a blockchain não é adequada para realizar operações computacionalmente pesadas, uma blockchain (ou seja, um contrato inteligente em uma blockchain) pode solicitar que os cálculos sejam realizados em uma infraestrutura computacional de alto desempenho fora da cadeia e obtenha os resultados verificados de volta por meio de um oráculo. O uso de um oráculo, neste caso, fornece garantias de integridade e autenticidade dos dados.

Um exemplo de tal oráculo é o **Truebit** (<https://truebit.io>). Ele permite que um contrato inteligente envie tarefas de computação para oráculos, que são eventualmente concluídas por mineradores em troca de um incentivo. A ideia fundamental é descarregar operações intensivas em computação para um sistema fora da cadeia e, quando a computação estiver concluída, os resultados serem postados de volta à blockchain via oráculo.

**Oráculos baseados em agregação (*Aggregation-based oracles*)**

Neste cenário, um único valor é obtido a partir de várias fontes. Por exemplo, esse valor pode ser o preço de um instrumento financeiro, e pode ser arriscado confiar apenas em uma única fonte. Para mitigar esse problema, vários provedores de dados podem ser utilizados, onde todos esses feeds são inspecionados e, finalmente, o valor de preço relatado pela maioria dos feeds pode ser selecionado. A suposição aqui é que, se a maioria das fontes relatar o mesmo valor de preço, então é provável que ele seja correto.

O mecanismo de colagem depende do caso de uso: às vezes é simplesmente a média de vários valores, às vezes é tomada a mediana de todos os valores, às vezes é a média aritmética (*mean*), e às vezes é o valor máximo. Independentemente do mecanismo de agregação, o requisito essencial aqui é obter um valor que seja válido e autêntico, que eventualmente alimente o sistema.

É necessário ter cuidado aqui porque um único nó malicioso pode distorcer os resultados de um valor médio relatando um número muito grande ou muito pequeno. Normalmente, valores medianos são usados na prática, pois se a maioria dos nós for honesta, o valor mediano pode ser razoavelmente confiável. Claro, se a maioria dos nós for maliciosa, então nem mesmo um valor mediano pode ser confiável.

Dois ótimos exemplos de oráculos baseados em agregação que agregam feeds de preços (também chamados de *price feed oracles*) são o **MakerDAO** (<https://makerdao.com/en/>) e os oráculos de feed de preço (<https://developer.makerdao.com/feeds/>), que coletam dados de preço de múltiplas fontes externas e fornecem um preço mediano ETHUSD ao MakerDAO.

**Oráculos orientados por sabedoria coletiva (*Crowd wisdom-driven oracles*)**

Essa é outra forma de abordar o problema do oráculo na blockchain, onde uma única fonte não é confiável. Em vez disso, múltiplas fontes públicas são utilizadas para deduzir, ao final, os dados mais apropriados. Em outras palavras, resolve-se o problema onde uma única fonte de dados pode não ser tão confiável ou precisa quanto se espera. Se houver apenas uma fonte de dados, ela pode ser pouco confiável e arriscada de se depender totalmente.

Neste caso, para garantir a credibilidade dos dados fornecidos por fontes terceiras aos oráculos, os dados são obtidos de várias fontes. Essas fontes podem ser usuários do sistema ou mesmo membros do público em geral que tenham acesso e conhecimento de certos dados — por exemplo, um evento político ou esportivo em que membros do público conhecem os resultados e podem fornecer os dados necessários. Da mesma forma, esses dados podem ser obtidos de diversos sites de notícias diferentes.

Os dados então podem ser agregados e, se um número suficientemente alto de informações idênticas for recebido de múltiplas fontes, então há maior probabilidade de que os dados estejam corretos e possam ser confiáveis.

**Oráculos descentralizados (*Decentralized oracles*)**

Outro tipo de oráculo, que surgiu principalmente devido aos requisitos de descentralização, é o oráculo descentralizado. Lembre-se de que, em todos os tipos de oráculos discutidos até agora, existem alguns requisitos de confiança colocados em uma terceira parte confiável. Como plataformas blockchain como o Bitcoin e o Ethereum são totalmente descentralizadas, espera-se que os serviços de oráculo também sejam descentralizados. Dessa forma, podemos resolver o problema do oráculo na blockchain.

Esse tipo de oráculo pode ser construído com base em um mecanismo distribuído. Também pode-se imaginar que os oráculos podem obter dados de outra blockchain, que é regida por um consenso distribuído, garantindo assim a autenticidade dos dados. Por exemplo, uma instituição executando sua própria blockchain privada pode publicar seu feed de dados por meio de um oráculo que, então, pode ser consumido por outras blockchains.

Um oráculo descentralizado essencialmente permite que informações fora da cadeia sejam transferidas para uma blockchain sem depender de uma terceira parte confiável.

A ideia central por trás do oráculo da **Augur** é a de oráculos apoiados por sabedoria coletiva, nos quais informações sobre um evento são adquiridas a partir de múltiplas fontes e agregadas no resultado mais provável. As fontes, no caso da Augur, são repórteres financeiramente motivados que são recompensados por relatórios corretos e penalizados por relatórios incorretos.

**Oráculos inteligentes (*Smart oracles*)**

Uma ideia de oráculo inteligente também foi proposta pela **Ripple Labs (Codius)**. O white paper original está disponível em:  
<https://github.com/codius/codius-wiki/wiki/White-Paper#from-oracles-to-smart-oracles>

Oráculos inteligentes são entidades semelhantes aos oráculos, mas com a capacidade adicional de executar código de contrato. Os oráculos inteligentes propostos pelo Codius são executados usando o **Google Native Client**, que é um ambiente isolado (*sandboxed*) para execução de código nativo x86 não confiável.

**Oráculos de saída (*Outbound oracles*)**

Esse tipo, também chamado de *oráculo reverso*, é usado para enviar dados da blockchain (dos contratos inteligentes) para o mundo externo. Há dois cenários possíveis aqui: o primeiro é quando a blockchain de origem é uma produtora de dados, como métricas da própria blockchain, que são necessárias para outra blockchain.

Os dados reais de alguma forma precisam ser enviados para outro contrato inteligente em outra blockchain. O segundo cenário é aquele em que um dispositivo de hardware externo precisa realizar alguma atividade física em resposta a uma transação *on-chain*. No entanto, observe que esse tipo de cenário não precisa necessariamente de um oráculo, porque o dispositivo de hardware externo pode receber um sinal como resultado do evento do contrato inteligente.

Por outro lado, pode-se argumentar que se o dispositivo de hardware estiver operando em uma blockchain externa, então, para obter dados da cadeia de origem para a cadeia de destino, certamente serão necessárias algumas garantias de segurança que a infraestrutura de oráculo pode fornecer. Outra situação é aquela em que precisamos integrar sistemas legados empresariais com a blockchain. Nesse caso, o oráculo de saída será capaz de fornecer dados da blockchain para os sistemas legados existentes. Um exemplo é a liquidação de uma negociação realizada em uma blockchain, que precisa ser reportada a sistemas de liquidação e relatórios back-end legados.

**Oráculos criptoeconômicos (*Cryptoeconomic oracles*)**

Esses oráculos são fundamentalmente **exchanges descentralizadas**, que são contratos inteligentes que permitem aos usuários negociar pares de ativos digitais *on-chain*. Como essas exchanges descentralizadas precisam manter preços precisos e atualizados para todos os ativos digitais, elas também podem ser usadas para servir como oráculos para fornecer o preço correto e mais recente de um ativo.

Oráculos criptoeconômicos fornecem **garantias econômicas de correção**. Na prática, isso significa que qualquer entidade adversária que tente manipular o sistema a seu favor, talvez introduzindo um preço falso no sistema, perderá seus fundos como penalidade devido às regras codificadas no sistema.

Agora que discutimos os diferentes tipos de oráculos, vamos apresentar os diferentes **provedores de serviços** que oferecem esses serviços. Vários provedores oferecem serviços de oráculo para blockchain, alguns dos quais apresentamos a seguir.

**Serviços de oráculo para blockchain (*Blockchain oracle services*)**

Diversos serviços online estão agora disponíveis que fornecem serviços de oráculo. Estes também podem ser chamados de **plataformas de oráculo como serviço** (*oracle-as-a-service*). Todos esses serviços têm como objetivo permitir que um contrato inteligente obtenha com segurança os dados fora da cadeia (*off-chain*) de que precisa para executar e tomar decisões:

* **Town Crier**: <https://www.town-crier.org>
* **Provable**: <https://provable.xyz>
* **Witnet**: <https://witnet.io>
* **Chainlink**: <https://chain.link>
* **Projeto Realitio**: <https://realit.io>
* **Truebit**: <https://truebit.io>
* **iExec**: <https://iex.ec>

Outro serviço disponível em <https://smartcontract.com/> também permite a criação de oráculos para Ethereum, Bitcoin e Town Crier. Ele permite que contratos inteligentes se conectem a aplicações e que dados sejam alimentados a partir de fontes fora da cadeia (*off-chain*) para os contratos inteligentes.

Há muitos serviços de oráculo disponíveis atualmente, e é difícil cobrir todos eles aqui. Uma seleção aleatória é apresentada na lista anterior.

Agora que cobrimos oráculos em detalhes, vamos olhar novamente para os contratos inteligentes e ver como eles podem ser **implantados** em um nível fundamental.

**Implantando contratos inteligentes (*Deploying smart contracts*)**

Contratos inteligentes podem ou não ser implantados em uma blockchain, mas faz sentido fazê-lo em uma blockchain devido à segurança e ao mecanismo de consenso descentralizado fornecido por ela. O **Ethereum** é um exemplo de uma plataforma de blockchain que oferece suporte nativo ao desenvolvimento e implantação de contratos inteligentes. Abordaremos o Ethereum com mais detalhes no **Capítulo 9 — Arquitetura do Ethereum**. Contratos inteligentes em uma blockchain Ethereum normalmente fazem parte de um aplicativo descentralizado maior (*DApp*).

Em comparação, em uma blockchain Bitcoin, os bloqueios temporais de transações (*timelocks*), como o campo nLocktime, o CHECKLOCKTIMEVERIFY (CLTV) e o operador de script CHECKSEQUENCEVERIFY na transação Bitcoin, podem ser vistos como habilitadores de uma versão simples de contrato inteligente. Esses bloqueios permitem que uma transação seja bloqueada até um tempo específico ou até um número de blocos, fazendo cumprir um contrato básico no qual uma transação só pode ser desbloqueada se certas condições (tempo decorrido ou número de blocos) forem atendidas. Por exemplo, é possível implementar condições como: "Pagar a parte X N bitcoins após 3 meses". No entanto, isso é bastante limitado e deve ser visto apenas como um exemplo de contrato inteligente básico.

Além do exemplo mencionado acima, a linguagem de script do Bitcoin, embora limitada, pode ser usada para construir contratos inteligentes básicos. Um exemplo de contrato inteligente básico é financiar um endereço Bitcoin que pode ser gasto por qualquer um que demonstre um ataque de colisão de hash. Esse foi um concurso anunciado no fórum Bitcointalk, onde bitcoins foram definidos como recompensa para quem conseguisse encontrar colisões de hash (discutimos esse conceito no **Capítulo 3 — Criptografia Simétrica**) para funções de hash. Esse desbloqueio condicional de bitcoins unicamente mediante a demonstração de um ataque bem-sucedido é um tipo básico de contrato inteligente.

Essa ideia foi apresentada no fórum Bitcointalk, e mais informações podem ser encontradas em:  
<https://bitcointalk.org/index.php?topic=293382.0>

Várias outras plataformas de blockchain oferecem suporte a contratos inteligentes, tais como:

* **Monax**
* **Lisk**
* **Counterparty**
* **Stellar**
* **Hyperledger Fabric**
* **Axoni Core**
* **Neo**
* **EOSIO**
* **Solana**
* **Polkadot**
* **Avalanche**
* **Tezos**

Contratos inteligentes podem ser desenvolvidos em várias linguagens, sejam DSLs (linguagens específicas de domínio) ou GPLs (linguagens de propósito geral). O requisito crítico, no entanto, é **determinismo**, que é muito importante porque é vital que, independentemente de onde o código do contrato inteligente seja executado, ele produza o mesmo resultado toda vez e em qualquer lugar.

Em blockchains onde uma GPL, como o **Rust**, é usada como linguagem de programação para contratos inteligentes, quaisquer recursos não determinísticos são removidos antes que essa linguagem seja implementada como uma linguagem de contrato inteligente no protocolo da blockchain. Por exemplo, o **Solana** usa a linguagem Rust como linguagem de contrato inteligente, mas com todos os recursos não determinísticos removidos.

Mais detalhes sobre isso podem ser encontrados em:  
<https://docs.solana.com/developing/on-chain-programs/developing-rust#restrictions>

Esse requisito da natureza determinística dos contratos inteligentes também implica que o código de contrato inteligente deve estar absolutamente livre de erros (*bug-free*). A validação e verificação de contratos inteligentes é uma área ativa de pesquisa, e uma discussão detalhada sobre esse tema será apresentada no **Capítulo 17 — Escalabilidade** e no **Capítulo 19 — Segurança em Blockchain**.

Diversas linguagens foram desenvolvidas para construção de contratos inteligentes, como a **Solidity**, que é executada na **Ethereum Virtual Machine (EVM)**. Vale notar que existem plataformas que já suportam linguagens convencionais para desenvolvimento de contratos inteligentes, como o Solana, que suporta Rust. Outro exemplo proeminente é o **Hyperledger Fabric**, que suporta **Golang**, **Java** e **JavaScript** para desenvolvimento de contratos inteligentes. Outro exemplo é o **EOSIO**, que permite escrever contratos inteligentes em **C++**.

A **segurança** é de suma importância para contratos inteligentes. No entanto, muitas vulnerabilidades foram descobertas em plataformas de blockchain populares e nas linguagens de desenvolvimento de contratos inteligentes relevantes. Essas vulnerabilidades resultaram em alguns incidentes de alto perfil, como o ataque ao **DAO**.

**O DAO**

A **Organização Autônoma Descentralizada (DAO)**, iniciada em abril de 2016, foi um contrato inteligente escrito para fornecer uma plataforma de investimento. Devido a um erro no código, chamado **falha de reentrada** (*reentrancy bug*), ela foi hackeada em junho de 2016. O equivalente a aproximadamente **3,6 milhões de ether** foi drenado do DAO para outra conta.

Embora o termo “hackeado” seja usado aqui, ele não foi realmente hackeado. O contrato inteligente fez exatamente o que foi programado para fazer, mas, devido às suas vulnerabilidades, o atacante conseguiu explorá-lo. Pode-se considerar que foi um comportamento não intencional (um bug) que os programadores do DAO não previram. Esse incidente resultou em um **hard fork** na blockchain do Ethereum, que foi introduzido para recuperar-se do ataque.

O ataque ao DAO explorou uma vulnerabilidade (bug de reentrada) no código do DAO em que era possível sacar tokens repetidamente do contrato inteligente do DAO antes que o contrato tivesse a chance de atualizar seu estado interno, indicando quantos tokens do DAO haviam sido retirados. O invasor conseguiu sacar os tokens. No entanto, antes que o contrato pudesse atualizar seu estado, o invasor retirava os tokens novamente. Esse processo foi repetido várias vezes, mas, ao final, apenas uma retirada foi registrada pelo contrato inteligente, e o contrato também perdeu o registro de todas as retiradas repetidas.

A noção de que **o código é a lei**, ou que contratos inteligentes são **imparáveis**, deve ser vista com certo ceticismo, pois a implementação desses conceitos ainda não está madura o suficiente para merecer confiança plena e inquestionável. Isso é evidente a partir dos eventos posteriores ao incidente do DAO, onde a fundação Ethereum conseguiu interromper e alterar a execução do DAO ao introduzir um hard fork na blockchain Ethereum. Embora esse hard fork tenha sido introduzido por razões legítimas, ele vai contra o verdadeiro espírito de descentralização, imutabilidade e da noção de que o código é a lei.

Consequentemente, a resistência contra esse hard fork resultou na criação do **Ethereum Classic**, onde muitos usuários decidiram continuar minerando na cadeia antiga.

Essa cadeia é a blockchain Ethereum original, não bifurcada (*non-forked*), que ainda contém o DAO. Pode-se dizer que, nesta cadeia, o código ainda é a lei.

Há tópicos de mensagens e anúncios interessantes relacionados a esse evento, que os leitores podem achar informativos e até mesmo divertidos:

* Carta aberta do atacante do DAO: <https://pastebin.com/CcGUBgDG>
* Anúncio do desenvolvedor principal do Ethereum: <https://twitter.com/avsa/status/745313647514226688>
* Especificação do hard fork: <https://blog.slock.it/hard-fork-specification-24b889e70703>

**Avanços na tecnologia de contratos inteligentes**

Desde a primeira disponibilização dos contratos inteligentes na blockchain Ethereum, desenvolvimentos significativos foram feitos para criar novas linguagens de programação de contratos e melhorar as linguagens existentes.

**Solana Sealevel**

A **execução paralela** de contratos inteligentes é uma área significativa de pesquisa para melhorar a **escalabilidade** da blockchain. A blockchain **Solana** introduziu o **Sealevel**, uma tecnologia para a execução paralela de contratos inteligentes, que ajuda a aumentar a escalabilidade e o desempenho da blockchain. O Sealevel é um ambiente de execução (*runtime*) que permite a execução paralela de contratos inteligentes nos núcleos de GPU disponíveis para um validador.

Mais informações sobre o Solana Sealevel estão disponíveis em:  
<https://medium.com/solana-labs/sealevel-parallel-processing-thousands-of-smart-contracts-d814b378192>

**Linguagem de Modelagem de Ativos Digitais (*Digital Asset Modeling Language — DAML*)**

**DAML** é uma nova linguagem funcional baseada em Haskell para blockchain que oferece recursos inovadores como:

* privacidade em subtransações
* processamento paralelo de transações
* escalabilidade
* poda de livro razão (*ledger pruning*)
* alta disponibilidade

O foco do DAML está nos fluxos de negócios distribuídos, permitindo que os desenvolvedores concentrem-se mais em programá-los do que se preocupar com a criptografia subjacente e outros detalhes complexos da arquitetura da blockchain. Dessa forma, os mecanismos subjacentes são abstraídos, e os desenvolvedores podem focar mais nos fluxos de negócios e nos problemas reais da empresa.

O DAML é projetado para construir aplicações **componíveis** sobre um modelo abstrato de livro razão DAML. Os contratos inteligentes DAML são armazenados em um livro razão. Para executar, criar ou ler do livro razão DAML, utiliza-se a **API do Livro Razão DAML** (*DAML Ledger API*).

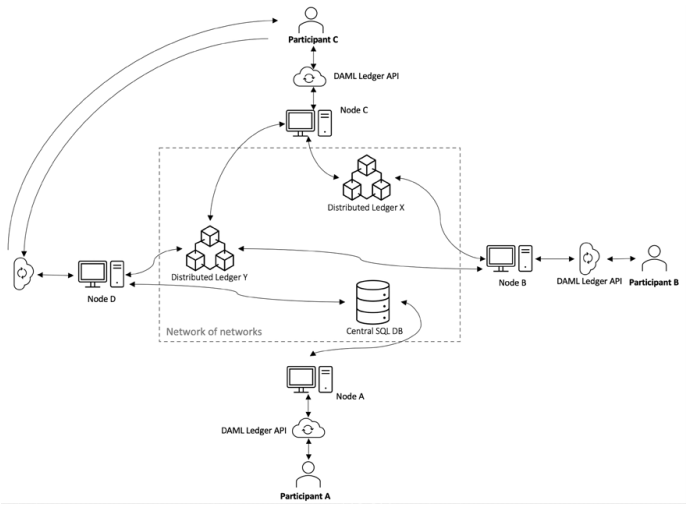
O modelo de livro razão DAML simplifica a programação, e o desenvolvedor não precisa pensar sobre o livro razão subjacente. Em vez disso, os desenvolvedores podem desenvolver o código contra o modelo de livro razão, e posteriormente os implementadores podem selecionar o livro razão real para implantação, seja ele **Corda**, **VMware**, **Hyperledger Fabric**, ou até mesmo um banco de dados centralizado padrão como o **PostgreSQL**. Além disso, ele permite fluxos de trabalho multipartidários ao fornecer às partes um livro razão compartilhado virtual que codifica o estado atual de seus contratos compartilhados, escritos em DAML.

Devido às complexidades de negócios, processos complexos e fluxos de trabalho excessivamente amplos, os modelos atuais são ineficientes. Além disso, a reconciliação entre esses sistemas é bastante desafiadora, principalmente devido à falta de interoperabilidade. Normalmente, a resposta é automatizar os processos de negócios, mas a complexidade central ainda existe e pode resultar em problemas posteriormente. Além disso, o código de infraestrutura geralmente está entrelaçado com as aplicações de negócios, o que resulta em soluções frágeis que carecem de interoperabilidade e portabilidade.

O DAML aborda essas ineficiências em fluxos de trabalho complexos atuais ao **isolar automaticamente a lógica de negócios do código de infraestrutura**. Além disso, o DAML permite a construção de aplicações componíveis sobre um modelo abstrato de livro razão DAML, tornando fácil o desenvolvimento de aplicações **interoperáveis** e **componíveis**. Normalmente, diferentes redes não conseguem se comunicar entre si, mesmo que utilizem as mesmas tecnologias, resultando em redes isoladas. Às vezes, essas redes acabam sendo capazes de se comunicar apenas por meios rudimentares, como compartilhamento de arquivos CSV ou XML.

No entanto, com o DAML, as aplicações podem se comunicar facilmente entre si, independentemente da plataforma subjacente, utilizando o Livro Razão DAML. Por exemplo, uma rede **Hyperledger Fabric** pode transacionar **atomicamente** com uma rede **Quorum**. Além disso, essas redes podem trocar informações com aplicações DAML que estejam sendo executadas em servidores de banco de dados centralizados, como o PostgreSQL, dentro de uma empresa.

O diagrama abaixo mostra esse conceito:



**Figura 8.5: Topologia de livro razão interoperável usando DAML**

No diagrama anterior, é mostrada uma topologia onde múltiplos livros razão interoperam com múltiplos participantes usando a API de livro razão DAML como uma rede de redes.

O DAML integra-se com diversos livros razão distribuídos, como **VMware Blockchain**, **Hyperledger Besu** e **Hyperledger Fabric**. É uma linguagem **portável** e ajuda a alcançar **interoperabilidade**, pois pode ser executada em um motor de execução DAML que pode ser integrado a qualquer blockchain ou até mesmo a um banco de dados centralizado. Portanto, em essência, o mesmo código funcionaria tanto para um banco de dados centralizado quanto para uma blockchain.

O DAML é construído com **privacidade** em mente, o que permite rastreamento e autorização em cada etapa do fluxo de trabalho; essencialmente, cada contrato inteligente possui sua própria privacidade definida.

Mais informações sobre DAML estão disponíveis em:  
<https://www.digitalasset.com/developers>

Outros avanços nas linguagens de contratos inteligentes incluem o desenvolvimento de ferramentas e métodos para analisar a **segurança** dos contratos. Essas ferramentas ajudam a encontrar padrões de código vulneráveis em contratos inteligentes. Além disso, esforços foram feitos para usar **métodos formais** para verificar contratos inteligentes. Também foram desenvolvidas linguagens que são adequadas à verificação formal, como a linguagem de programação **Move**.

Discutiremos isso em mais detalhes no **Capítulo 17 — Escalabilidade** e no **Capítulo 19 — Segurança em Blockchain**.

**Resumo**

Este capítulo começou introduzindo o histórico dos contratos inteligentes, seguido por uma discussão detalhada sobre a definição de contrato inteligente. Como não há um consenso sobre uma definição padrão, tentamos apresentar uma definição que abrange a essência dos contratos inteligentes.

Também foi apresentada uma introdução aos **contratos Ricardianos**, e a diferença entre contratos Ricardianos e contratos inteligentes foi explicada, destacando que os contratos Ricardianos estão focados na **definição** de um contrato, enquanto os contratos inteligentes estão orientados para a **execução** real de um contrato.

Discutiu-se o conceito de **modelos de contrato inteligente**, uma área de intensa pesquisa atualmente tanto na academia quanto na indústria. Foram discutidas ideias sobre a possibilidade de criar **DSLs de alto nível** para a criação de contratos inteligentes ou modelos de contratos inteligentes.

Nas seções posteriores, os **oráculos** foram introduzidos, seguidos por uma breve discussão sobre o **DAO**, juntamente com suas questões de segurança e implicações para contratos inteligentes.

Por fim, discutimos os **avanços em contratos inteligentes**, incluindo linguagens, execução paralela, DAML, e verificação formal.