第5章　智 能 合 约

智能合约是一种将合约条款直接写入代码中的自我执行协议，通过自动化流程，减少中介，提高合约执行过程的透明度，具有彻底改变各个行业的潜力。本章将深入探讨智能合约的原理、设计和实际应用，然后介绍智能合约的局限性，如它们对区块链网络的依赖和重入攻击等潜在漏洞，以及支持智能合约执行的以太坊虚拟机、EOS 和超级账本 Fabric 等各种平台与环境，最后介绍创建可跨不同区块链网络的、无缝通信的互操作智能合约的挑战与机遇。

5.1　智能合约的力量：自执行协议

本节将深入讨论智能合约的原理、方法及实际应用。包括智能合约的图灵完备性和局限性、基于智能合约构建的去中心化应用，以及智能合约的现实世界用例供应链管理与 DeFi 等。

5.1.1　智能合约的图灵完备性和局限性

图灵机是一种抽象的计算模型，它被认为是现代计算机的理论基础，用于研究算法及可计算性。图灵机的组件包括：磁带，无限长的单元格带，每个单元格都能容纳有限字母表中的单个符号；读写头，可以沿磁带左右移动、读取当前单元格中的符号并将新符号写入单元格的设备中；有限状态机，基于当前状态和从磁带读取的符号确定机器下一步操作的控制单元。

图灵机的工作原理：首先进行初始化，机器以特定的初始状态启动，输入数据写在磁带上；其次进行读写，读写头读取其下方的符号，有限状态机根据当前状态和读取的符号确定下一步操作，这可能涉及写入新符号、左右移动读写头或更改机器的状态；最后是重复前面的步骤，直到达到停止状态。

作为在区块链上执行的可编程协议，智能合约改变了人们与去中心化应用的交互方式。在理解智能合约的能力与限制方面，图灵完备性的概念起着关键作用。智能合约的图灵完备性示意图如图 5-1 所示。

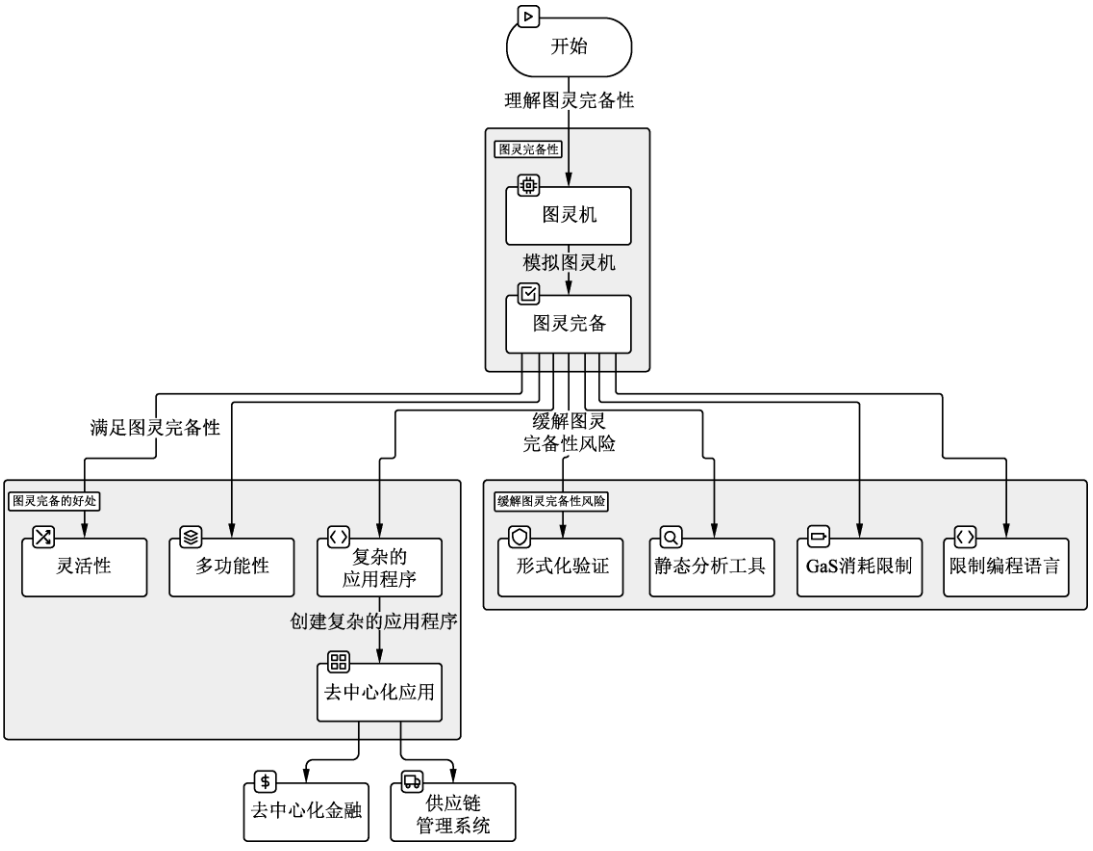

图 5-1　智能合约的图灵完备性示意

1．图灵完备性：强大的计算能力

图灵机能执行任何可以用算法描述的计算，是计算的理论模型。若一种编程语言能模拟图灵机，那么它就被认为是图灵完备的。对智能合约而言，图灵完备意味着理论上可以使用智能合约创建任意复杂程度的应用程序，而底层虚拟机如以太坊虚拟机，可以执行任何类型的计算。

2．图灵完备的好处

智能合约满足了图灵完备性，具备灵活性、多功能性。智能合约赋能开发者创立具备条件逻辑、循环及状态管理的复杂应用程序。基于智能合约能开发多种去中心化应用，包括去中心化金融（DeFi）、智能供应链管理系统等。

3．图灵完备的挑战

虽然图灵完备的好处很多，但是也引入了挑战和限制。

- 复杂性与安全性，图灵完备的语言可能增加了开发的复杂度。如果没有经过细致的设计和测试，它的灵活性可能导致漏洞产生。

- 性能与可扩展性，复杂的智能合约有可能消耗大量的计算资源，可能影响区块链网络的性能及可扩展性。这有可能导致交易费用增加、执行时间变长。
- Gas 费用，许多区块链网络引入了 Gas 的概念，以应对资源密集型操作的风险。Gas 代表智能合约执行所需的计算资源。为避免资金耗尽并预防合约被撤销，开发者应仔细管理 Gas 消耗。

图灵完备系统可以表达不可判定性问题，即存在一些问题，无法通过有限步骤的算法来确定其答案，这将导致智能合约陷入无限循环或不良状态中。

4．缓解图灵完备性的风险

为了解决图灵完备性相关风险，可以采取如下策略：

- 智能合约的正确性和安全性可采用形式化验证技术来验证；
- 合约中的潜在问题、漏洞及低效之处，可利用静态分析工具自动识别；
- 为了确保网络的可持续运行，可设置适当的 Gas 消耗限制；
- 一些区块链网络通过限制使用特定的编程语言来限制智能合约的复杂性并降低漏洞的风险。

图灵完备是一个强大的概念，赋能构建复杂的智能合约。但是也引入了多种潜在挑战，如复杂性、安全性及不可判定性等。充分权衡这些因素，并且采取相应策略后，能降低执行风险并充分发挥图灵完备性的优势。

5.1.2　基于智能合约构建去中心化应用

去中心化应用是一种基于区块链的应用，利用智能合约的自动化流程执行协议并进行交互。基于智能合约构建去中心化应用的示意图如图 5-2 所示。

1．去中心化应用的开发原则

- 去中心化应用必须运行在去中心化的网络上，消除对中心化机构的依赖，确保其强大的容错性、审查抗性及透明性；
- 去中心化应用代码库应当是开源、公开可访问的，以促进社区的发展；
- 许多去中心化应用利用代币作为激励来促进交易，并在生态系统内创造经济价值；
- 应优先考虑用户资金和数据安全性，采取健壮的编码，遵循最佳安全实践。

2．去中心化应用的开发方法

去中心化应用开发通常使用的编程语言有 Solidity、Vyper 或 Rust 等。这些编程语言赋能开发者用智能合约来编写应用程序的规则和逻辑；

去中心化应用需要一个友好的用户及交互界面，包括直观的界面、清晰的说明和无缝的体验；去中心化应用使用区块链的 API 和库与底层区块链网络通信，执行交易、检索数

据和与其他智能合约交互；去中心化应用依赖区块链的共识机制来确保交易和数据的安全性、完整性；去中心化应用通常利用经济模型来激励参与、分配奖励和管理生态系统，这涉及代币经济学、治理机制及其他经济激励措施；去中心化应用的代币经济，涉及代币的供应、分配和效用，即确定初始代币供应量、代币生成规则及在代币生态系统中如何使用等。

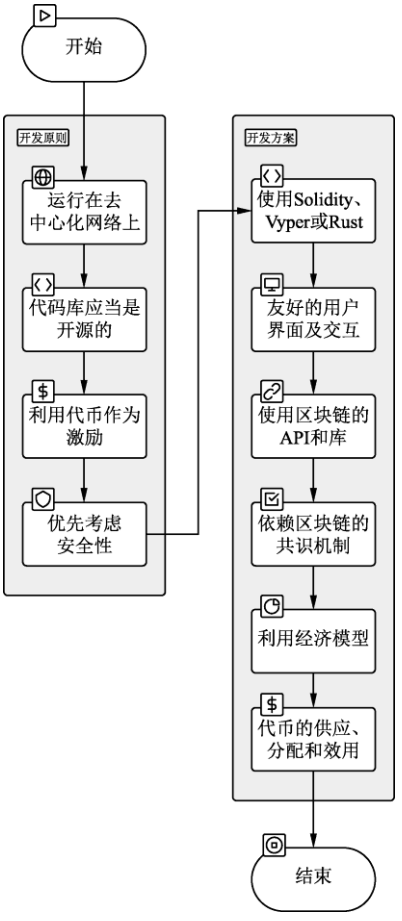

图 5-2　基于智能合约构建去中心化应用流程

3．成功的去中心化应用

一些成功的去中心化应用有：

- 去中心化金融（DeFi），Aave、Compound 和 Uniswap 等平台，提供去中心化借贷、借款和交易；
- CryptoKitties、Decentraland 等引入了新的游戏和虚拟现实空间模型，基于区块链技术实现所有权、稀缺性及互操作性；
- 在供应链管理中，去中心化应用跟踪商品的移动，保障供应链的透明度及真实性；
- 去中心化应用提供去中心化的身份解决方案，帮助用户控制自己的个人数据并减少对中心化机构的依赖。

去中心化应用是一种可能会彻底改变各行业的革命性技术。深入剖析它的原理和方法后，能够创建具有影响力的应用程序，充分发挥区块链技术的优势。

5.1.3　智能合约用例：供应链管理和去中心化金融

在供应链管理等领域，智能合约有广泛的应用。智能合约在提高效率、透明度及安全性等方面改善了传统业务流程，下面将充分阐述。

1．供应链管理

智能合约用于跟踪产品从生产到消费的整个生命周期，保证供应链的透明度及可追溯性，有助于防止伪造、欺诈性交易以及不符合标准的产品进入市场；智能合约跟踪库存水平，触发补货订单并保障及时交付，有助于减少库存成本，避免缺货、超卖的情况发生；智能合约自动处理支付、结算业务，减少纸质文书工作，加快了流程，提高了供应链效率；

智能合约可以自动执行合同条款，保障各方履行义务，有助于减少争议纠纷，提高供应链的可靠性。智能合约在供应链管理中的应用如图 5-3 所示。

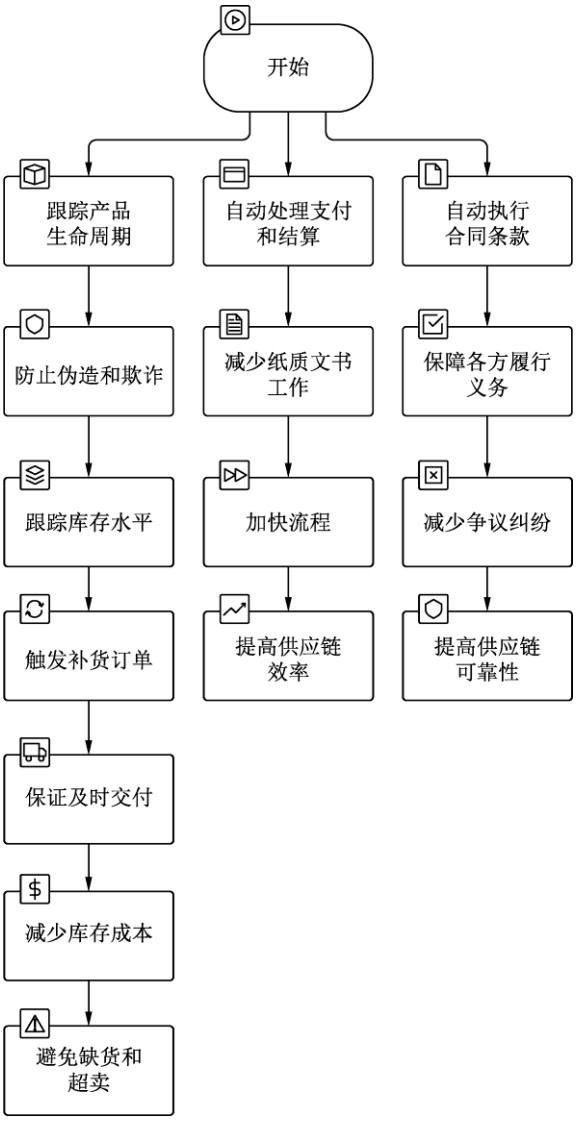

图 5-3　智能合约在供应链管理中的应用

区块链技术可以用于供应链金融，以改善现金流、降低融资成本。智能合约可以跟踪货物的运输、交付过程，依据预定条件触发付款，这使得供应商可以更快获得付款，减少资金周转时间且降低坏账风险。

2．去中心化金融

去中心化金融（DeFi）平台利用智能合约创建去中心化的借贷市场，用户在无中间人的情况下直接借贷、借款；去中心化交易所使用智能合约实现点对点的加密货币交易。智能合约用于创建、交易金融衍生品，如期权、期货和永续合约，为投资者提供了新的投资

机会及风险管理工具；智能合约可进行自动化的投资管理，如基于预设条件的资产配置及再平衡。智能合约应用于去中心化金融的执行流程如图 5-4 所示。

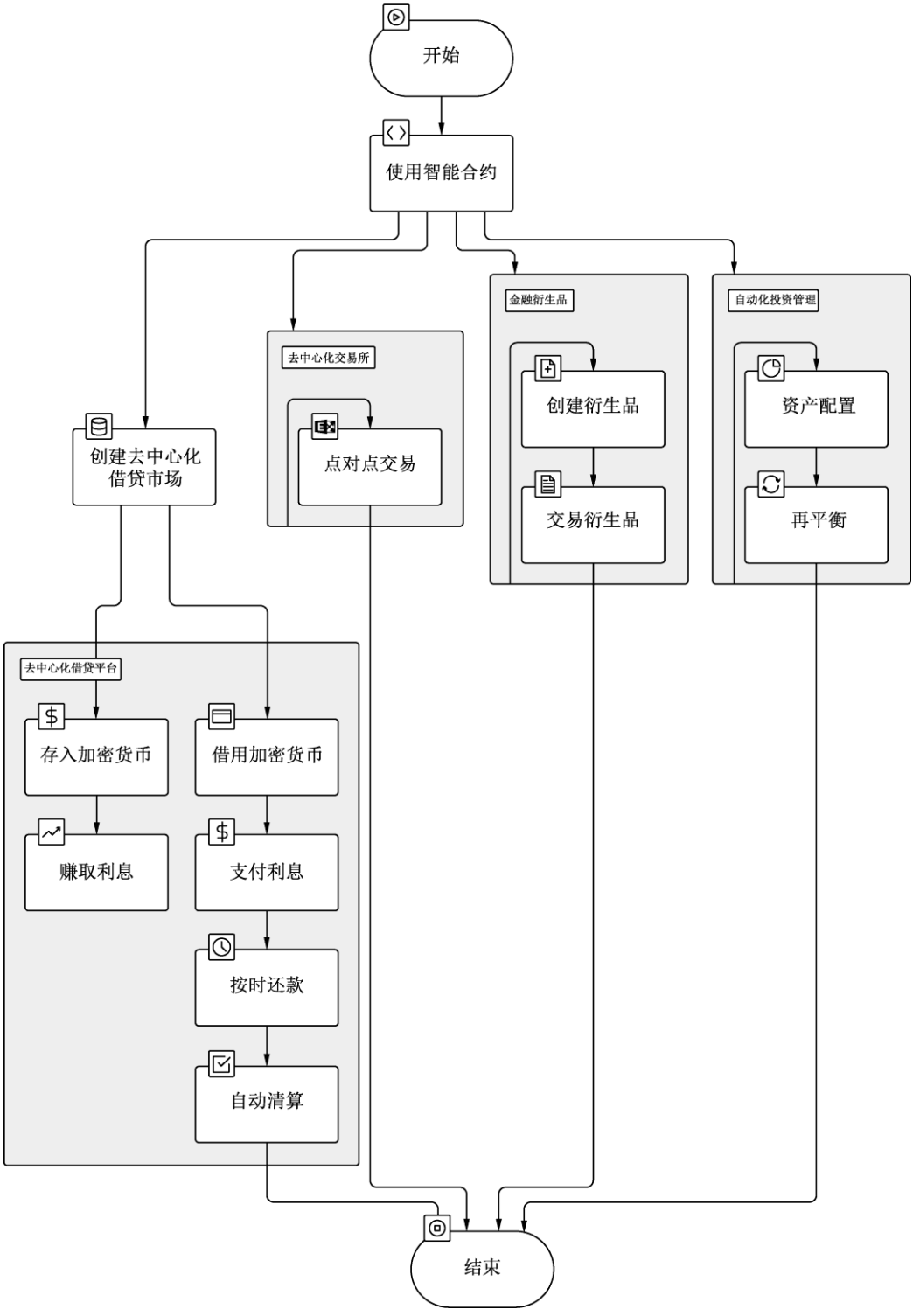

图 5-4　智能合约应用于去中心化金融的执行流程

领先的去中心化借贷平台：Aave 和 Compound。用户可以将加密货币存入平台来赚取利息。反之，用户也可以借用加密货币来支付利息。为保护贷方利益，智能合约能确保借款人按时还款，并自动清算。

随着区块链技术的发展，我们有希望看到更多的创新性智能合约应用。

5.2　智能合约设计与开发

本节将介绍智能合约的设计和开发的关键内容，如 Solidity 编程语言、安全事项以及稳健性测试等，掌握这些原则或方法，能够充分利用智能合约的潜力，构建创新、可靠的去中心化应用。

5.2.1　用于创建智能合约的 Solidity 编程语言

Solidity 是一种专门为在以太坊区块链上开发智能合约而设计的高级语言，它是一种静态类型编程语言。Solidity 合约被编译成字节码，其能在以太坊虚拟机上执行。以太坊虚拟机托管在连接到区块链的以太坊节点上。

1．Solidity的关键特性

- 静态类型，Solidity 执行严格的数据类型，有助于防止编程错误并提高代码可读性；
- 合约能从其他合约继承过来，允许代码重用及模块化；
- 支持创建和使用库，这些库是可重用代码集合，能导入其他合约中；
- 修饰符是自定义条件，用于在函数中控制访问权限和执行逻辑；
- 事件可以从合约中发出，以通知其他合约或外部应用程序合约状态的变化；
- 存储用于保存区块链上的持久数据，而内存用于保存函数调用期间的临时数据。

2．编写与部署Solidity合约

编写与部署 Solidity 合约流程如图 5-5 所示。

（1）创建一个新的.sol 文件来编写合约代码。

（2）用 contract 关键字定义一个新的合约。

（3）定义变量、函数，声明变量来存储数据，定义函数与合约交互。

（4）编写合约逻辑，应用 Solidity 语法，在函数中实现所需的逻辑。

（5）使用 Solidity 编译器，把合约代码编译成字节码。

（6）通过交易，将编译后的字节码部署到以太坊区块链上。这会在区块链上创建一个新的合约地址。

Solidity 是以太坊区块链上开发智能合约的强大工具。通过 Solidity，可开发出新的合

约并支撑各种去中心化应用。

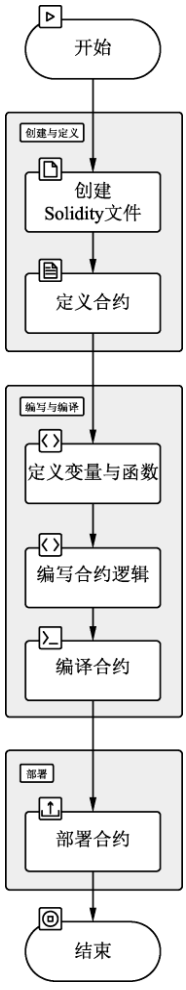

图 5-5　编写与部署 Solidity 合约流程

5.2.2　安全注意事项：重入攻击和缓解措施

重入攻击是智能合约的一个安全方面的威胁，它利用智能合约可以在同一交易中调用其他合约的能力，即合约在执行过程中可能会调用其他合约的函数从而被恶意合约利用，导致攻击者能够反复调用原合约并篡改其状态。重入攻击可能会导致资金损失或出现其他意料之外的结果。

1．重入攻击

当一个合约调用另一个合约时，有发生重入攻击的潜在可能。恶意合约是指在目标合约（目标合约是被攻击的对象）的状态更新之前，再次调用目标合约函数，对目标合约多

次操控，如多次提取资金。

比如一个简单的目标合约，它包含 withdraw 函数，该函数允许用户从余额中提取资金。而恶意合约循环将会调用 withdraw 函数所在的目标合约，在 withdraw 函数完成之前可能会多次提取资金。

2．缓解重入攻击

可以使用多种技术缓解重入攻击的风险。缓解重入攻击的技术执行流程如图 5-6 所示。

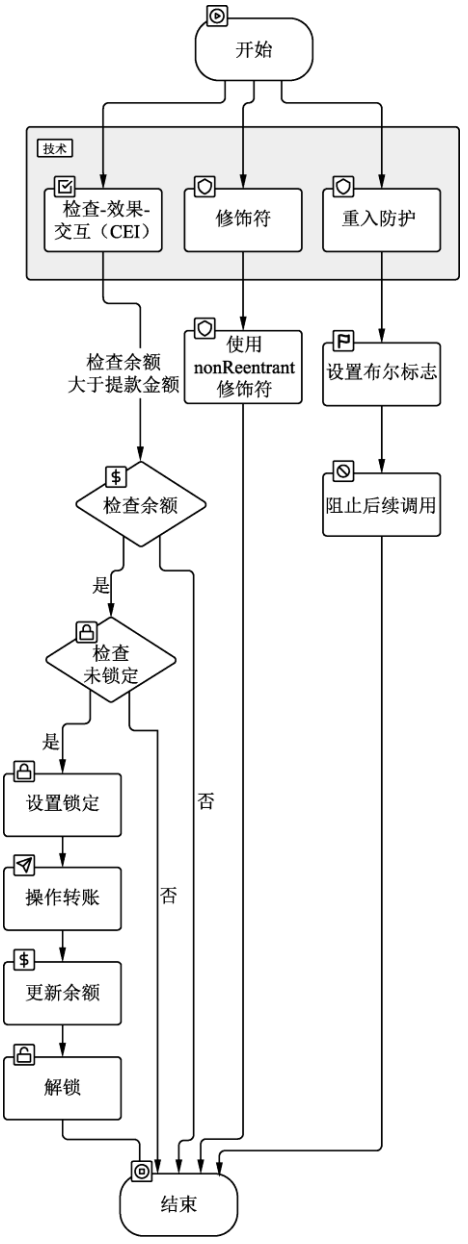

图 5-6　缓解重入攻击的技术执行流程

- 检查-效果-交互（CEI）模式要求按照这种方式执行函数：最先执行必要的检查代码，再执行操作合约状态的代码，最后执行与其他合约、外部账户交互的代码，即首先检查状态，再更新状态效果，稍后交互。通过在转账操作之前进行检查，如先检查余额大于提款金额，然后检查是否未锁定。然后使用设置锁定（如 locked = true）来防止重入，锁定后，进行转账操作，更新余额后解锁。若检查时已锁定，则不得再进行转账操作，防止重入。

Solidity 提供了应用在函数中强制执行某些条件的修饰符。例如，可以使用 nonReentrant 修饰符来防止同一事务中递归调用函数。

重入防护是布尔标志，用于跟踪函数当前是否正在执行。如果设置了标志，则会阻止对函数的后续调用。

通过了解重入攻击的原理并实施适当的缓解、预防技术，研发者可以显著降低漏洞风险，并确保智能合约应用的完整性。

5.2.3 测试和审计智能合约的稳健性

测试和审计是保证智能合约安全、可靠和其功能的重要流程。

1．测试技术

- 单元测试，单独测试智能合约的某个函数、组件，从微观层面识别错误；
- 集成测试，评估合约不同组件间的交互，保障合约作为一个整体而运行；
- 模糊测试，生成随机输入，测试合约漏洞及意外行为；
- 安全测试，专门识别安全漏洞，如重入攻击、整数溢出及拒绝服务攻击等；
- Gas 估算，测试执行智能合约所要消耗的 Gas 量，预防产生意外成本，保障交易成功执行。智能合约测试序列图如图 5-7 所示。

2．审计技术

- 手动审查代码，由经验丰富的研发者审查代码，识别潜在的漏洞等；
- 自动进行代码分析，使用自动化工具分析合约代码，发现常见漏洞；
- 形式化验证，使用数学方法证明合约行为的正确与否；
- 安全审计，由审计公司进行独立的安全性审计，对智能合约的安全性予以全面评估。

运用测试和审计工具，如 Solidity 测试框架、Mythril 静态分析工具等提高效率。

对于智能合约的健壮性、安全性，彻底的测试和审计是必不可少的。通过结合各种技术和工具，识别潜在漏洞，从而在智能合约应用中建立用户和参与者的信任和信心。

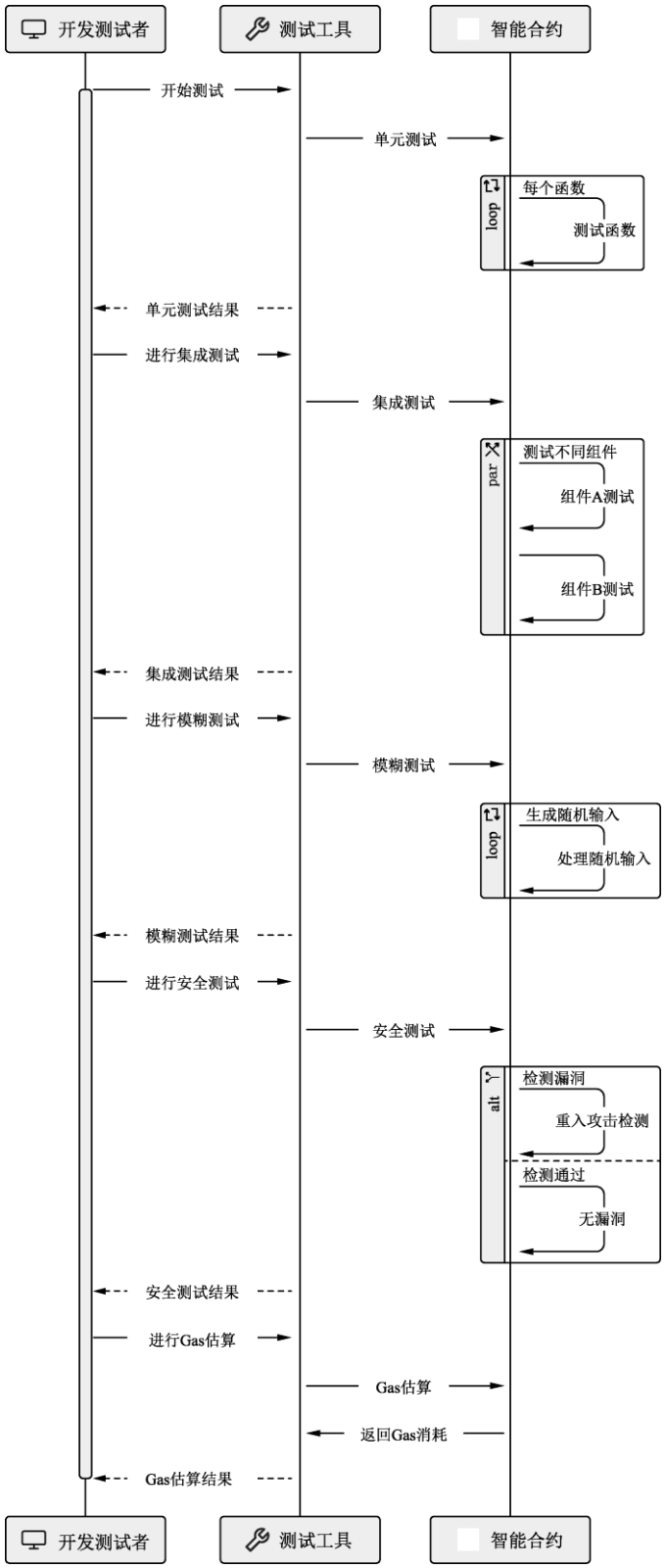

图 5-7　智能合约测试序列图

5.3 智能合约平台和执行环境

智能合约需要平台和环境来执行并与区块链交互，包括以太坊虚拟机、EOS 和超级账本 Fabric 平台及互操作性，本节将介绍智能合约平台的多方面内容。对于利用智能合约的研发人员和企业来说，了解这些概念、原理及方法至关重要。

5.3.1 以太坊虚拟机和 Gas 费用

以太坊虚拟机用于执行智能合约，是以太坊区块链的基本组件。它提供了一个沙箱环境，智能合约能在其中和区块链或其他合约交互。了解以太坊虚拟机和 Gas 费用，对于有效的智能合约开发非常重要。

1．以太坊虚拟机

以太坊虚拟机是基于堆栈的虚拟机，在确定性状态机上运行，在以太坊虚拟机上执行 Solidity 或其他兼容的编程语言生成的字节码。以太坊虚拟机提供了一组预定义的操作码，用于各种执行操作，如算法、逻辑运算及数据存储等。

2．以太坊虚拟机的关键组件

以太坊虚拟机的关键组件如图 5-8 所示。

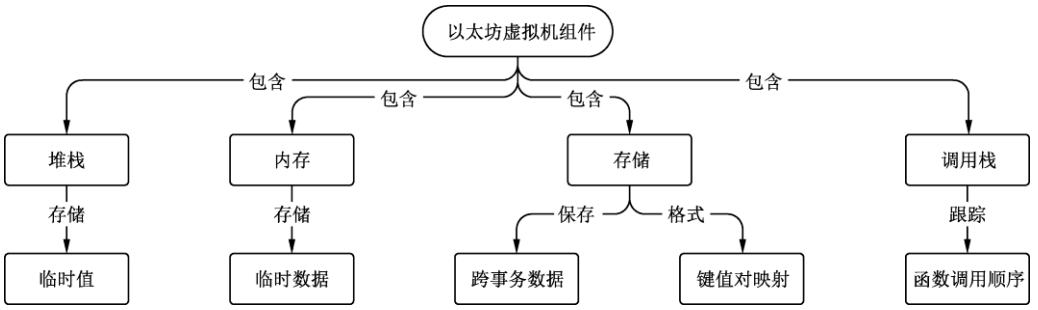

图 5-8 以太坊虚拟机关键组件

堆栈用于在执行操作码时存储临时值。内存用于存储仅在当前事务中需要的临时数据。存储用于保存跨事务可访问的持久数据。调用栈，用于跟踪在程序执行过程中各个函数的调用顺序。以太坊账户分为两种类型，其中：外部账户由私钥控制，可以发起交易，用于持有以太币，并与智能合约交互；合约账户用于存储合约的代码和状态，由部署的合约代码定义，被动响应外部调用。以太坊区块链上的每个账户都有其状态，如账户余额、存储等。余额即账户所拥有的以太币数量；存储是一个键值对的映射，键、值都是 256 位

的数字，用于保存合约的状态变量，如投票权及合约的内部状态等。

3．Gas费用

Gas 是衡量以太坊区块链上交易消耗的计算资源的单位。

以太坊交易成本模型：以太坊单个交易的总费用与基础费、Gas 用量及其单价相关，计算公式如下：

$$T = b + (g \times p)$$

总费用 = 基础费 + (Gas 用量 × 单价)

其中，T 代表总费用，交易的最终成本 Gas 即需要支付的以太币数量；b 为基础费用，是网络对每笔交易收取的固定费用；g 为 Gas 交易消耗的 Gas 数量，即计算资源的消耗量；p 为 Gas 单价，即每单位 Gas 的价格。

4．高级主题

- 预编译合约：以太坊提供了一组预编译合约，其他合约可以调用预编译合约，这些预编译合约提供了某些函数的优化实现，如加密操作。
- 状态转换：EVM 通过从一种状态转换到另一种状态来执行交易。
- 以太坊虚拟机操作码：有数百个可用的操作码，每个操作码都有其功能和 Gas 成本。例如：PUSH1 表示把一个字节的数据压入栈顶，即将其从指令中复制到栈顶；ADD 表示将栈顶的两个数字相加并将结果压入栈顶。

了解以太坊虚拟机和 Gas 费，对于开发高效、安全的去中心化应用至关重要。仔细考虑 Gas 成本并优化智能合约代码，可以保障交易成功执行且成本合理。

5.3.2　替代智能合约平台：EOS 和超级账本 Fabric

虽然以太坊是智能合约的主导平台，但是也出现了几个替代平台，它们各有其特点和优势。

1．EOS：高性能平台

EOS 是一个旨在实现高交易吞吐量和可扩展性的区块链平台。它利用委托权益证明（DPoS）共识机制及独特的架构并行执行交易。

2．EOS的关键特征

EOS 使用委托权益证明来选择一组验证交易并创建新区块的块生产者，验证者按照预定的轮次顺序轮流出块；EOS 能同时处理多个交易，可显著提高吞吐量及可扩展性；EOS 采用基于账户的模型，每个账户都有自己的余额和独立的存储模型；EOS 使用 WebAssembly 进行智能合约的执行，与以太坊虚拟机相比，WebAssembly 的灵活性和性能更高。EOS 的

关键特征如图 5-9 所示。

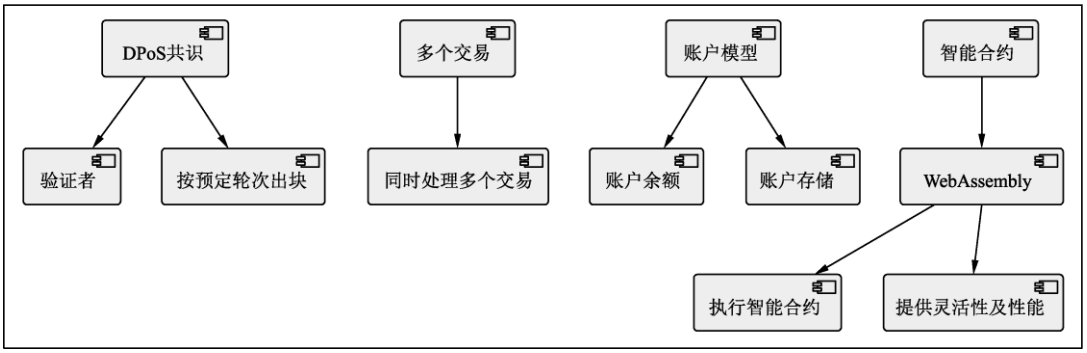

图 5-9　EOS 的关键特征

3．超级账本Fabric：许可型区块链

超级账本 Fabric 是一个面向企业用例的许可型区块链平台。它高度模块化、可定制，允许组织根据其特定需求定制平台。

超级账本 Fabric 只有授权参与者才能加入网络；模块化架构提供包括共识机制、成员服务及链码执行环境等在内的多个模块化组件；链码是 Fabric 上的智能合约，它用 Go 或 Java 语言编写，在沙箱环境中执行；Fabric 支持各种插件，如身份管理、隐私和合规性等。

选择 EOS 还是 Fabric，是由去中心化应用的特定需求决定的。如果高吞吐量的公共区块链是一种强需求，则 EOS 是合适的选择；超级账本 Fabric 更适合需要许可型网络且对平台有更多控制的场景。

替代基于以太坊智能合约开发的两个方案 EOS 和超级账本 Fabric 各有优势，适用于不同的用例。在权衡平台的可扩展性、许可及开发者支持等因素后，研发者能为去中心化应用选择合适的平台。

5.3.3　互操作智能合约与跨链通信

不同区块链之间的互操作性是区块链被广泛使用所面临的挑战。

1．互操作性的原则

互操作智能合约需要在不同区块链平台之间安全地交换数据、信息，需要具备在不同区块链之间转移资产等能力，包括转移代币、NFT 等；互操作智能合约需要在共识机制、验证规则方面保持一致，采用不同共识算法的区块链需要找到一种兼容方式，如通过侧链、跨链桥来实现；互操作智能合约需要考虑安全及隐私，保护用户数据并防止恶意攻击。

2．互操作性的方法

采用跨链协议、侧链、桥接、原子交换等方法实现不同区块链之间的互操作。

3．互操作性的未来

- 预言机网络，向智能合约提供链下数据，使智能合约能与现实世界交互；
- 异构网络，互操作解决方案需要权衡具有不同共识机制、代币经济及治理模型的异构网络；
- 隐私与保密，保护用户隐私，确保在区块链上交互的数据的保密性是关键因素。

互操作智能合约、跨链通信对于释放区块链技术的潜力至关重要。利用多种方法解决面临的挑战，可以创建一个更加高效、互联的区块链网络和去中心化应用生态系统。

5.4　智能合约实践

下面编写并编译 Solidity 智能合约。

1．设置开发环境

（1）安装 Node.js 和 NPM，确保你的系统上安装了 Node.js 及 Node 的包管理器 NPM。可以从如下网站下载并安装 https://nodejs.org/en。

（2）创建项目目录，给你的项目建立一个新目录。初始化 Node.js 项目，打开终端，导航到项目目录，然后运行以下命令：

```
npm init -y
```

2．编写Solidity智能合约

在开发过程中，可使用一个集成开发环境（IDE），如 Visual Studio Code 搭配 Solidity 插件。

创建 Solidity 文件，在项目目录下创建一个新的.sol 文件，如 MyContract.sol。

编写合约代码，使用 Solidity 语法定义合约的变量、函数及逻辑。下面是一个简单的例子。

```
// SPDX-License-Identifier: MIT
// Author: Xingxiong Zhu, Email: zhuxx@pku.org.cn
pragma solidity ^0.8.0;

contract SimpleStorage {
    uint storedData;                    // 声明一个存储整数数据的变量
```

```
    function set(uint x) public {        // 定义一个公共函数
        storedData = x;                  // 将传入的值存储到 storedData 变量中
    }

    function get() public view returns (uint) {     // 定义一个公共视图函数
        return storedData;               // 返回存储在 storedData 变量中的值
    }
}
```

3．安装Solidity编译器solc

打开终端，运行下面的命令安装 Solidity 编译器 solc。

```
npm install -g solc
```

这会将 solc 安装到全局环境，方便在任意目录下使用。

4．编译合约

```
solcjs MyContract.sol --bin --abi

MyContract_sol_SimpleStorage.bin
MyContract_sol_SimpleStorage.abi
```

solcjs 是 Solidity 编译器的一个版本，可以用它编译 MyContract.sol 文件。在终端中运行下面的命令编译文件：

```
solcjs MyContract.sol --bin --abi

MyContract_sol_SimpleStorage.bin
MyContract_sol_SimpleStorage.abi
```

编译后创建了两个文件，一个是.bin 文件，包含编译后的字节码，另一个是.abi 文件，其是合约的应用程序二进制接口。

5．在线智能合约开发环境

Remix IDE 是一个线上的以太坊智能合约开发环境，网址是 https://remix.ethereum.org/。它不需要在本地安装，可以直接在浏览器中进行 Solidity 合约的编写、编译、部署和调试。

5.5 小　　结

本章介绍了智能合约的基本原理、设计事项及其实际应用，以及如何通过优化自动化流程、提高交易透明度、消除中介来改变传统的协议模式。

理论上，智能合约能执行图灵机可以执行的任何计算，这使得其功能众多且强大。智能合约是去中心化应用的关键构建块，可建立值得信赖、透明及安全的系统。它已经在供

应链、金融等多个领域得到了应用。

Solidity 是一种专门为在以太坊区块链上编写智能合约而设计的高级编程语言。设计安全的智能合约对于预防重入攻击等漏洞至关重要。适当的测试、审计，对于识别和缓解潜在风险必不可少，有助于保障智能合约的可靠性和健壮性。

以太坊虚拟机是以太坊智能合约的执行环境，为执行字节码提供了沙盒环境。EOS 和超级账本 Fabric 为智能合约开发提供了不同的特性和权衡。

实现不同区块链平台之间的互操作性，在区块链间的无缝通信和资产转移至关重要。

智能合约代表区块链技术领域的重大进步，为开展新业务及自动化流程提供了新的模式，可以利用其强大的功能研发去中心化解决方案。随着技术不断发展，未来将会看到更多的智能合约突破性应用。

5.6　习　　题

一、选择题

1．智能合约的主要功能是什么？（　　）

A．在区块链上存储数据　　B．在区块链上执行代码

C．创建新的加密货币　　D．促进点对点交易

2．以下哪个不是智能合约的限制？（　　）

A．无法访问外部数据　　B．重入攻击等潜在漏洞

C．缺乏图灵完备性　　D．对区块链网络的依赖

3．以下哪个是基于智能合约构建的去中心化应用的示例？（　　）

A．社交媒体平台　　B．加密货币交易所

C．云存储服务　　D．文字处理器

二、论述题

1．在智能合约的背景下论述图灵完备性对智能合约的能力及限制有哪些影响。

2．比较以太坊虚拟机、EOS 和超级账本 Fabric 等平台在架构、共识机制和用例方面有哪些主要区别？

3．论述创建跨不同区块链平台互操作的智能合约的挑战和机遇，这些挑战的潜在解决方案是什么？

4．智能合约如何用于提高供应链管理中的透明度和可追溯性？

5．论述智能合约的未来及其在各个行业中的应用潜力。