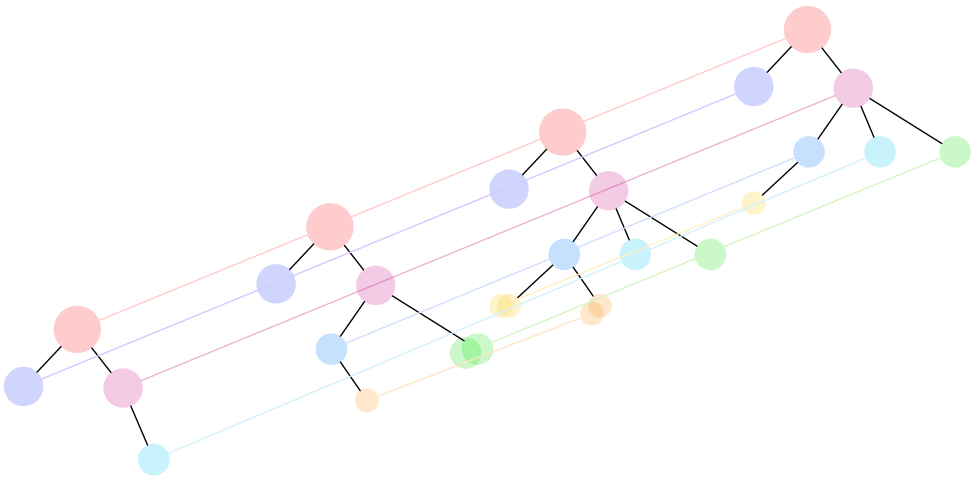
Grid - 一个基于多级侧链的高性能区块链框架



Grid 全球基金会

内部预览版 0629.2

包含任何拨款/融资信息和完整的通信地址。

摘要

本文提出了一种区块链系统框架设计方案，该框架具有以下特征：具有数据分链和垮链传输机制，通过在交易中写入其他链的部分区块信息，实现跨链交互；采用操作系统中分层、子系统的概念，实现区块链各模块间解耦并可单独定制；采用微服务架构；设计了一个区块链最小系统；精简单一链的协议，只处理一种业务，将不同业务分配在不同的链上，解决区块链数据冗杂的问题；扩大智能合约内应用范围，通过智能合约定义共识机制，及开放额外引入国密、抗量子攻击算法、零知识证明等算法的机制；引入单一链合约协议升级机制；具备子链动态加入和离开机制，设计了一种树形侧链结构，将线性的区块链结构扩展为三维树形链结构；设计了 一种代理虚拟矿机的共识机制。本项目通过新型开源基金会进行项目运作。

关键字：三维树形链、分链、跨链、模块化、微服务、最小区块链、合约升级、代理虚拟矿机共识机制

1 现有的区块链系统

目前，区块链技术及其应用处在爆发式增长阶段，许多行业都面临将传统的网络架构向以区块链为基础的网络架构迁移。然而，起源于数字货币的区块链技术本身在跨行业应用场景中仍存在许多问题，这些问题大大限制了区块链技术的应用领域拓展。

1.1 通用区块链 vs 复杂业务场景

当前，区块链技术面临的挑战是，需要满足纷繁复杂的业务应用场景。例如，多方物流供应业务、银行票据留存及溯源业务、多时间节点的复杂交易支付业务等。这些业务在流程表达与执行逻辑方面往往具有不同的特征。

为了满足这些需求，当前一般有两种解决方式可供参考。一种是将业务相关数据及哈希，写入一笔交易输出的 OP_RETURN[1]，存储到区块链中。许多类比特币的区块链系统采用这种方式应对商业需求。这种方式致力对任何业务场景保持兼容，却仅仅把区块链当作一个数据库使用，业务逻辑无法通过区块链处理。另一种方式是基于不同的业务需求硬性抽象出若干种业务模型，针对每一种业务模型，分别编写复杂的智能合约，再将其写入同一条区块链。以以太坊为典型的此种方式确实利用区块链处理了一部分业务逻辑，但这一类的区块链系统仍以通用原则为设计目的，基于通用的共识机制的智能合约编写过程十分复杂，运行效果难以尽如人意。同时，这些各具特

[1] OP_RETURN 是比特币的一个脚本指令，由于该指令能够返回任何数据并且比特币的交易会被记录在区块链中，因而被广泛用于数据存储

点的智能合约都被写入了同一条链，对用户来说，区块链变得复杂、维护成本高昂，业务数据和逻辑流程缺乏有效的组织。

1.2 协议升级困境

一个区块链系统一经发布，后续重大的升级及新特性的引入将变得十分困难。区块链本身的技术复杂性、区块链版本及数据的兼容需求通常导致重大的升级更新举步维艰。区块链系统中包含了大量的业务逻辑、数据及诸多利益相关方，任何的升级改动意图难以在生态社区中获得高效有价值的反馈。典型的例子就是比特币，近几年许多新特性的引入都经历了社区中长期的争执。

1.3 数据冗杂

前文提到，为满足不同业务场景需求，通用性是现有区块链系统的一个重要设计目的。对通用性的追求引入了复杂的协议处理和共识机制。为了满足这些复杂的规则，链上的业务流程数据日趋臃肿杂乱。一条“通用”的链，往往意味着对场景和用户缺乏特征识别，这无疑增加了数据复杂度，导致数据冗杂。

1.4 区块系统膨胀

一个越成功的区块链系统，其维护成本越高。现在运行一个比特币的全节点需要超过 130G 的硬盘空间，以太坊的区块链大小已经超过 180G。区块链的数据永远在增长，一个区块链系统越流行，用户越多，交易越活跃，那么它的数据膨胀就越明显，维护成本就越高，进而形成区块链系统膨胀与区块链维护成本而这件的恶性循环。

1.5 线性处理链式结构的性能瓶颈

一个区块链系统所承载业务量不断增长，其线性处理压力通常面临超过其设计容量的风险，进而产生区块链网络性能瓶颈。在传统的 IT 架构中，数据库分库、分表

或改用分布式架构等方式可最大程度缓解性能瓶颈。而现有的区块链系统中，性能增长困难重重，往往需要牺牲一部分业务处理效率。例如，比特币交易量的增长使得现在的手续费变得越来越昂贵，并且仍有很多交易需要等待很久才能被确认。

另一方面，一个区块中的交易处理没有通过并行的方式增加其运行效率。当一个区块包含大量交易及复杂的只能合约时，线性执行会影响系统的区块铸造及区块验证效率。

1.6 点对点通信支持

现有的区块链系统多基于 P2P[2]广播网络进行通信，而对于点对点数据通信的支持低效而危险。一个典型的例子是，如果一些数据只有有限的用户关心，那么这些数据就应当仅在有限的节点间传播，而非通过 P2P 网络向每一个节点广播。

1.7 跨链交互有待突破性进展

现有区块链技术领域内已有一些在区块链系统间处理相关业务逻辑的尝试，然而数据的跨链交互一直是业界棘手的技术难题。现有跨链交互技术有中心化的实现方案和类 HTLC[3]的实现方案。中心化的实现方案往往面临信任问题、单点故障及单点性能瓶颈问题，应用场景较为局限。类 HTLC 的实现方案只能处理资产互换等特定应用模式，同时对两条链的协议、共识机制有严格要求，具体操作步骤也比较复杂。无论哪一种实现方案，在区块链系统间协议的差异适配及数据标准交互格式定义这两个核心议题上都有待突破性的技术进展。

---

[2] Peer to Peer，一种点对点网络传输技术。

[3] Hashed Timelock Contracts，一种基于比特币脚本的智能合约技术。

2 Grid 的主要设计目标

2.1 提升 TPS[4]

在传统的 IT 架构中，分布式的思想已经成为了打破性能枷锁的流行解决方案。区块链系统也应当支持分布式的并行处理，例如不存在数据竞争的多笔交易应当被同步执行以提升交易处理性能，当一条链的处理能力成为性能瓶颈时，通过构建链的分布并发，分担该链的业务压力，达到性能扩容。

2.2 解决数据冗杂

区块链的设计应面向专注于解决特定业务特点的问题，而非将所有智能合约都在同一条链上做普适处理。新链的创建应面向明确的业务目的，亦即须明确这条链针对怎样的业务定义、执行怎样的业务逻辑、实现怎样的业务价值。进而，新链的协议、共识机制、处理数据的定义、执行业务逻辑的流程摆脱了通用性的桎梏，链内只存在一个经过有效组织的、由协议及共识机制决定的合约集，其数据也会被特定业务需求严格规范，数据冗杂也将极大程度得到解决。

2.3 协议升级

区块链创建伊始，应当清晰明确地定义投票机制的升级策略，藉由共识机制的与时俱进决定新特性的引入，以期避免协议升级困境。

2.4 跨链交互

通过消息方式实现面向现有主流链的跨链交互技术支撑。

---

[4] TPS 每秒交易量

2.5 联盟链模版

区块链技术中，区块数据同步、交易存证溯源、共识机制处理等特性极大地吸引了联盟用户。然而联盟链通常独立存在，而非依赖于另一个生态或与联盟外的业务场景混杂在一起。我们提供了一种类似亚马逊云服务 AMI 市场[5]的模式，用户可以基于一个联盟链模板，快速的创建一条独立的链，独占地处理自己的业务。

2.6 系统膨胀的优化

区块链系统可以定义触发快照的机制。在特定的周期，将当前的有效数据进行一次快照，即可丢弃快照前的交易详细数据，该链之后的交易可以该快照为起始。类似传统 IT 架构数据归档机制的快照机制能够在具体的业务应用中极大缓解区块链系统数据膨胀的问题。

3 Blockchain as OS

如果一个区块链系统需要支撑许多应用系统，那么它就是一个操作系统。

以数字货币为设计目标的比特币作为区块链系统的开山之作，更趋近于一个应用系统。相比之下，以太坊则体现出一些操作系统的特征：用户可以通过智能合约编写自己的“应用程序”，平台以 Solidity 形式提供了“程序设计语言”和“编译器”等。然而，从现代操作系统的角度，以太坊还存在许多不足，如系统组件间缺乏解耦，大多数模块没有开放定制，系统接口定义不完善等。

Linux 内核与众多的 Linux 发行版本构成了庞大而成功的 Linux 家族。Linux 内核解决了操作系统中最基本、最重要、也最耗时的部分，其他开发者根据各自的应用

[5] 亚马逊云服务中，用户可以基于现有的实例镜像快速创建自己的虚拟机实例，也可以将自己制作的镜像开放到社区中。

场景、受众需求，定制了面向不同领域的发行版，使得 Linux 成为最流行的服务器操作系统，支撑各行各业的应用。

我们将这种思想融会到了 Grid 的设计中。首先，我们定义并实现了包含区块链系统最基本功能的 Gird 内核——一个最小化的区块链系统；其次，我们实现了一个基本的“Shell”以期拥有一个基本的“区块链操作系统”交互接口。用户可以直接使用我们提供的完整“区块链操作系统”，也可以基于 Grid 内核，根据特定的需求，快速构建用户定制的区块链系统。进而，用户可以像 Linux 内核裁剪改装一样根据内核各子系统的接口定义，对 Grid 内核实现进行修改，深度定制用户自己的 Grid 内核。

基于这样的思想，区块链系统的定制将达到前所未有的灵活度。开发者可以基于零知识证明重新实现交易规则与跨链消息机制，得到一条“完全匿名”的链——这条链上的交易不可见，这条链本身也不可见，然而其存在又都可以被证明。

另一个例子，开发者可以基于常见的椭圆曲线加密算法之外的其他加密算法，实现区块链系统的账户体系；使用定制的哈希算法，实现区块哈希，脚本哈希等模块。实际上，所有涉及密码学的抽象集都可以被定制化的重新实现，以满足特定安全准则的要求。比如，一个政府军工领域的开发者可以使用国密算法替换全部原有的密码学实现，以满足国家安全审批的需求。开发者可以使用抗量子攻击的算法实现相关模块，帮助用户战胜对量子攻击的恐惧。

4 最小化区块链系统

4.1 区块

区块用于定义一个系统状态，上一个区块到本区块的状态转化由本区块内定义的交易进行转换。

4.2 交易

交易的规则是智能合约。智能合约本质上是一段程序，对于一段相同的输入，应在任何情况下给出一段相同的输出。这样的程序称为智能合约，包含这段程序的原子化执行的结构称为交易。

4.3 账户

账户用于区分数据存储的边界。一般由公钥私钥体系构成。

4.4 P2P 网络通信

节点间数据能够通过底层 P2P 网络进行传输。

4.5 共识机制

对区块链系统内记账权的明确规定。

5 Grid 内核

5.1 实现最小化区块链系统

最小化区块链系统的各部分在 Grid 内核中进行实现，对于智能合约、共识机制、区块头部自定义区域等需要进一步定制的部分，定义相关接口。

5.2 统一的账户体系

比特币系统通过公钥私钥体系引入了账户的概念，其 Pay To Script Hash[6]则是将交易控制权交给了一个智能合约，以太坊区块链系统中定义了内部账户与合约账户，Grid 内核将上述两类账户统一。

[6] Pay to script hash (P2SH)，比特币中的一种支付脚本，将交易发送至一个脚本的哈希，而不是公钥的哈希

5.3 区块内部交易实现并行处理

Grid 对交易静态分析能够获得业务处理将会影响的数据范围，无数据读写冲突的多项业务处理流程能够并行执行，并不影响各项业务流程执行后的结果。区块铸造过程中，铸造方将根据业务处理的互斥性，将处理流程分配到不同的组。组内的业务流程能够以线性处理模式执行，组能够并行执行（见图 1 区块内业务流程分组）。

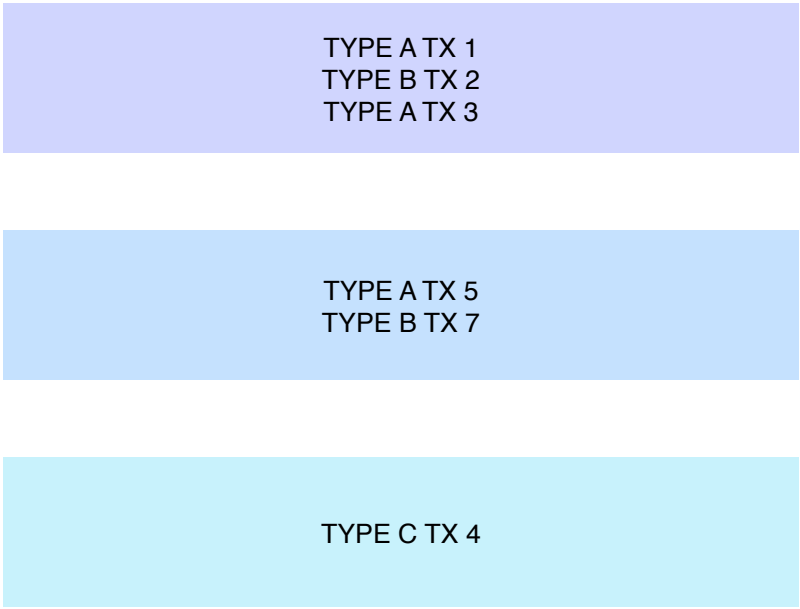


图 1 区块内业务流程分组

对于特殊业务处理流程，其影响的数据范围会随着区块内交易的执行发生变化，难以并行执行。这种情形下 Grid 区块铸造策略是优先执行能够并行的业务流程，在足额手续费用的情况下将这些非并行化业务流程放在一个非并行化组中线性执行，否则区块链铸造者可以拒绝该项业务流程。值得注意的是，当区块铸造者恶意的接受一笔耗时长且无法并行化执行的业务流程时，其他的节点拒绝该区块的概率将增加。

5.4 链合约集

Grid 链合约只存在一个链的初始化过程中被定义的合约集，该合约集命名为创世合约集（Genesis Smart Contract Collection）以向中本聪致敬。链合约集的本质是一些类，这些类定义了链的主要功能、链的共识机制以及合约集的升级策略。

5.5 协议升级

Grid 链的行为通过链合约集来定义，链合约集的升级会引起整个链行为的变化，合约的升级机制由前一个合约集定义。譬如，我们可以定义最近的 100 个区块中有 80%投票支持的情况下，在经过 2000 个区块确认后，新合约集将会替代原合约集，不及时升级合约集的节点将停止工作。

5.6 共识机制插件化

对于特定的业务场景，共识机制对参与者的决策影响巨大。在具有一定信任基础的联盟链中，PBFT[7]是一个多数的选项。这样的共识机制在节点固定且节点数量较少的情况下性能出众。而在低信赖环境下，网络的健壮则一般通过 PoW[8]、PoS[9]、DPoS[10]等共识算法得到保障。

Grid 链的共识机制定义在链合约集中。下文以比特币、点点币[11]的链作为例子进行具体分析。

对于比特币的 PoW 共识机制，因为不需要任何系统状态，仅需要区块头部信息即可验证区块的合法性，因此 PoW 的共识合约不需要任何数据状态信息作为输入。对于点点币的 PoS 共识机制，因为需要一笔未花费交易，因此仅仅提供区块头信息给共识合约是不够的，还需要这个区块 stake 交易[12]的数据，以及验证这笔 stake 交易的输入是一笔合法的未花费交易的数据。建议 Grid 链设计者在设计共识机制时，能够实现

---

[7] PBFT 是 Practical Byzantine Fault Tolerance 的缩写，意为实用拜占庭容错算法。

[8] PoW 是 Proof of Work 的缩写，意为工作量证明。

[9] PoS 是 Proof of Stake 的缩写，意为股权证明。

[10] DPoS 是 Delegated Proof of Stake 的缩写，意为委任权益证明。

[11] PeerCoin，简称 PPC，名字取自 P2P 货币的意思，即点对点货币，因此被翻译为点点币。以权益证明（Proof of Stake，PoS）取代工作量证明（Proof of Work，PoW）来维护网络安全。

[12] 在 PoS 机制中，用于证明铸造者股权的交易称为 stake 交易。

仅读取区块头信息即可执行共识验证的链合约集实现，这样便于对一条链进行快速的合法性验证。

此外，在特定的业务场景下，建议使用定制的共识机制。

5.7 自定义区块头数据

点点币由于在区块头部不包含验证区块合法性的信息，导致一个 stake 区块[13]无法通过其自身验证区块合法性。Grid 内核提供了一种在链初始化过程中，在区块头中自定义数据结构的机制。通过构建一个叶子节点由 Hash（TxId + N + Value）组成的未花费的交易的 Merkle Tree[14]，计算其 Root 存储在区块头中，并附加获得 Stake 奖励的 TxId、N 及 Value 以及 Merkle Tree 验证数据，即可使用区块头验证该区块的有效性。

6 Grid 操作系统

6.1 合约运行

Grid 操作系统能够配置不同的合约运行环境。一个 Grid 链能够配置一种智能合约运行环境。Grid 操作系统将优先支持 Docker，也将原生支持 JAVA、C#、GO、JAVASCRIPT、LUA 等语言。

在 Docker 方案中，Grid 将在容器内部实现一个 RPC 服务来提供合约执行过程中读取全局变量及账户存储的接口实现。在原生语言的方案中，Grid 将会通过相应的 SDK 来提供相应的功能接口实现。

---

[13] 点点币中通过股权证明(PoS)机制铸造的区块。

[14] 一种数据结构中所指的树，其主要特点是所有非叶子节点的值由其子节点值的哈希决定。

6.2 微服务

将智能合约微服务化，实现智能合约的定义不依赖于特定语言。由于共识机制也是在智能合约中定义的，因此共识机制本质就成为了一个服务。

6.3 面向云端

通过微服务的方式能够将 Grid 内核的并行化运行交易的能力扩展至云端，从而实现合约执行云服务化。合约执行速度数量级的提升，有助于区块铸造节点在相对短的周期内完成海量交易接收与验证工作。

由于区块链系统原生提供了数据的一致性约束，在数据存储方面，能够使用性能极高的内存型 Key-Value DB，将热数据放在内存中。通过现有技术应用成熟的分布式数据服务，能够有效提升系统的 IO 性能。

6.4 轻节点

Grid 内通过 Merkle 证明的机制，可以通过业务定制使得一个节点仅处理与自身相关的系统消息，以实现系统节点的轻量化。这样的节点将极大兼容轻桌面、移动终端等运行环境。

6.5 可选模块

6.5.1 数据清理机制

Grid 系统通过快照机制，将系统进行重新初始化，初始化后的状态相当于一个区块在创世区块[15]中追加了原始数据。

6.5.2 数据隧道

数据隧道是点对点的传输的一种实现，其数据并不会记录在区块链中。数据隧道仅用于点对点的加密数据传输。如 A 向 B 购买了一定的数据，通过数

[15] 创世区块：区块链中的第一个区块，通常会定义这条链的初始数据系统参数等。

据隧道，B 发送数据，A 发送资产。这样便于细分交易粒度从而低成本的降低欺诈风险。区块链网络完全有能力实现无中介化的无欺诈交易，但考虑链上传输成本相对较高，Grid 系统提供了数据隧道这一妥协但高效的解决方案。

6.5.3 快速确认模型

当一笔交易指定了能够与其交互的地址，并在一定周期内只与目标地址发生交互时，Grid 系统能够通过接收方持有发送方的授权实现快速的交易确认。以资产发送场景为例，当节点 A 希望与节点 B 建立一个快速确认的交互通道时，A 需要构造一笔交易，该笔交易将一定额度的资产锁定，并声明只会发送给 B。在实际交易过程中，A 将签名好的交易通过数据隧道发送给 B，B 在收到交易后能够立即确认该笔交易，当 B 希望将 A 的承诺兑现时，B 将这些交易打包附加上自己的地址并签名，这些交易的相关资产将被发送到 B 的地址上，A 的剩余资产将返还，通道关闭。

当然，该快速确认模型的数据传输载体不一定是资产，也可以是一些行为。譬如 A 可以通过数据隧道，授权 B 执行某些即时操作。上文资产传输示例的本质是 A 授予 B 拿走自己资产的行为。

6.5.4 代币模块

在资源调用需要进行计费的场景下，或需要对维护区块链稳定的行为作出奖励时，系统需要一个价值载体模块。该模块能够快速表达价值载体所需的相关逻辑。

6.6 定制

Grid 系统通过模块的参数化定义，能够使得开发者快速定制区块链网络。Grid 的设计思想是特定链解决特定的核心问题，通过高度的抽象及模块化架构，能够便于

企业或者创业者快速将自己的区块链业务逻辑落地实施。对于深度用户，可以更加定制化的实现自己链，专注于业务逻辑的实现。

7 Grid

Grid 是 Grid 主链与 Grid 侧链的统称。 它有别于现有的单一链的系统，其本质是由主链主导的，包含多个侧链的生态系统。

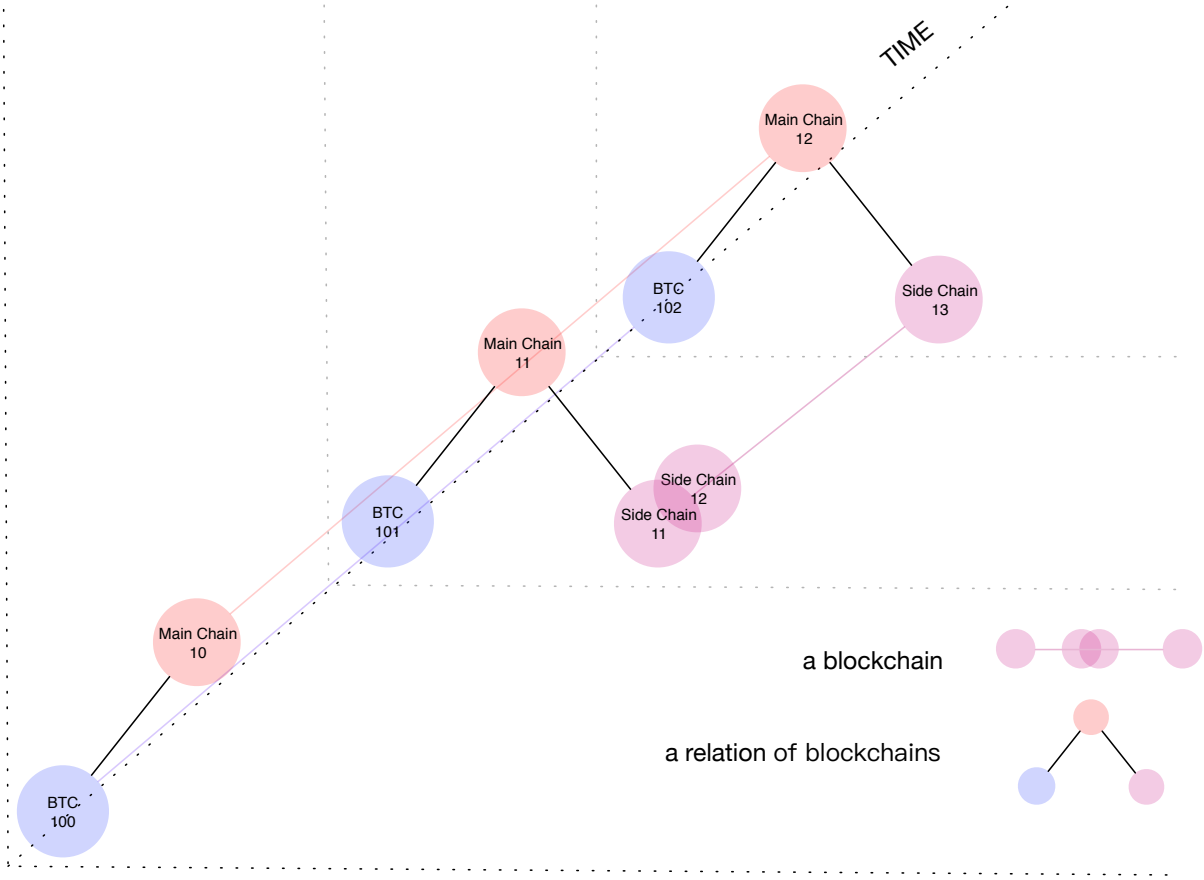


图 2 Grid 系统概览

7.1 核心理念

Grid 通过每一条侧链解决一个具体问题，解决数据冗杂的问题。从传统的一条链能够接受多种合约（图 3 数据冗杂的区块链系统），限制到一条链仅能够实现一个业务边际内的合约，将一条链的业务进行有效的聚焦。

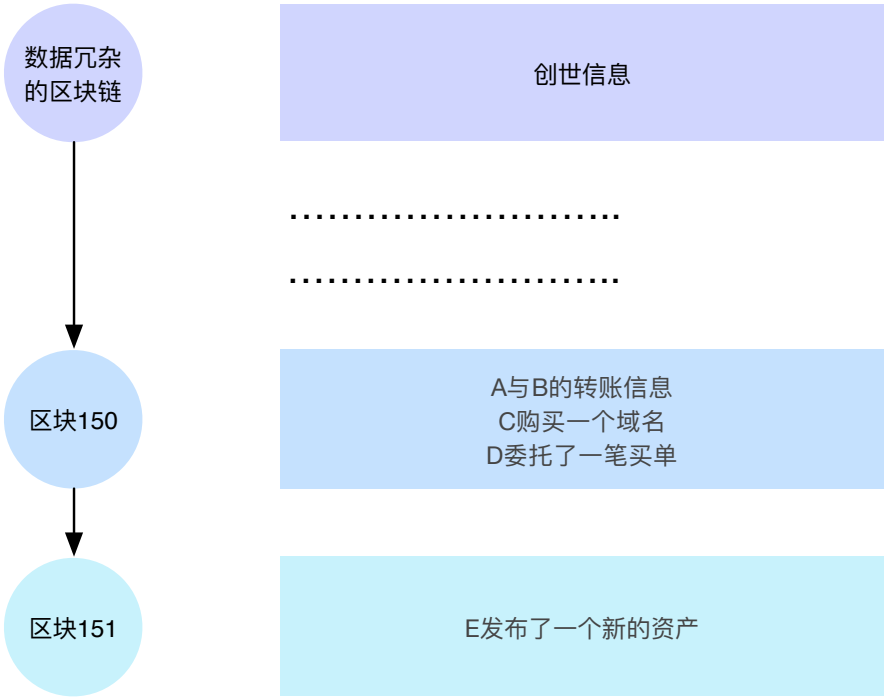


图 3 数据冗杂的区块链系统

7.2 侧链动态索引

Grid 是一个动态的系统，由于一条侧链的交互是通过主链的 Merkle Tree 及外部消息的输入进行的验证，不与其他任何的链发生交互，因此其中的侧链可以被添加，也可以被删除。主链是整个系统边界的索引。在图 2 Grid 系统概览中，我们可以看到主链在区块高度 10 上，其只索引了一个比特币侧链。主链在区块高度 11 上，增加了对一条侧链的索引，这个侧链在索引的时候已经有两个区块了，因此主链直接将侧链的两个区块囊括了进来。

7.3 树形侧链延展

Grid 提供了一个主侧链架构，符合条件的链都可以成为主链或侧链。值得注意的是，因为侧链理论上支持所有类型的链，因此一条主链亦可以作为侧链挂在在另外一条主链上。Grid 因此具有了横向与纵向延展的能力，其主要思想来自于数据库的分库分表，按照业务范围拆分成独立的库，当单表数据量过大时拆分成多个表，在这里对应的是多条业务不同的侧链与侧链的分支侧链。

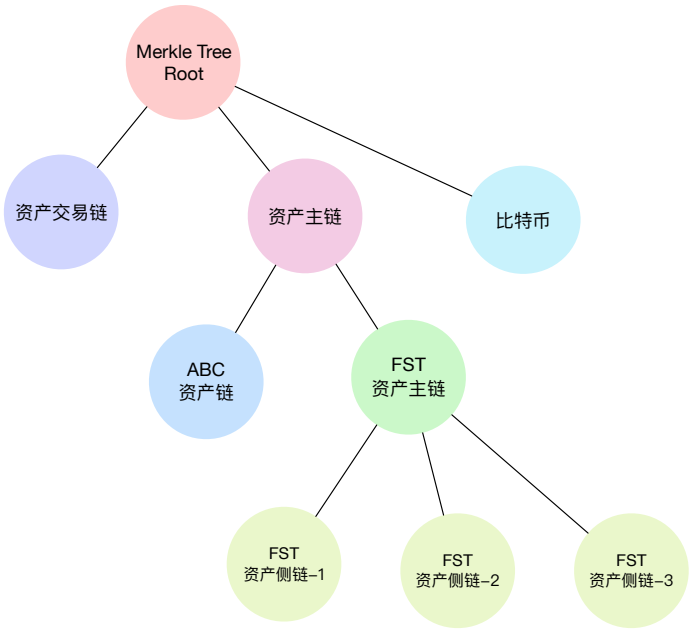


图 4 多级区块链架构

7.4 Grid 经济系统

Grid 作为一个生态系统，需要有一个核心的良性经济系统以激励其可持续地运转。Grid 经济系统将采用代理虚拟矿机权益证明机制。

在 PoS、DPoS 机制中，任何持币用户可以在一个极短的周期内（PoS 有一个锁定期）将其持有的代币销售以退出整个经济生态。而 PoW 的生产是来自于矿机的，矿机的转让受外部资源（电力成本、矿机折损、场地）及人力资源方面的限制，因而矿机持有者退出该经济生态较为困难。PoS、DPoS 机制还存在另外一个棘手的问题，即交易所控制了大量的流通代币，其能够几乎无成本地通过流通代币获取利息。在 PoS、DPoS 机制下，经济系统的博弈并不如 PoW 机制般有众多角色参与。

基于实体矿机的应用实际，Grid 提出了虚拟矿机的概念。

实体矿机在需求量大的情况下，会被快速的制造出来并加入网络，但其增长速度受芯片数量的限制，芯片数量又是通过最近的历史销售情况得出的。实体矿机在运转过程中会消耗一定的电量，并且产生一定的损耗。

虚拟矿机系统每周的供应量是有限的，其最大矿机数量由近期的历史供应量决定。如矿机需求量下滑，最大供应量则随之下降。如矿机需求量超越矿机总量，则下一周期供应量上升。矿机竞购行为是一个匿名拍卖行为，拍卖时需要支付总金额的代币，同时附加支付金额、单价及数量的哈希值。当拍卖结束后，竞拍者需要验证其拍卖信息，该过程须经历特定时间周期。如竞购者中标，则获得相应数量的矿机；如竞购失败则退款；如未进行验证，则罚没竞拍金额。为了鼓励尽早下单，系统将按照提前竞购的天数给予每天 1%的折扣。系统将给予限额内虚拟矿机数量，如超过虚拟矿机限额，则出价高者获胜。

这样的模型，由于购买者在不同时间的成本不同、成本计量方法不同（法币计价、代币计价、比特币计价），出价不同，策略不同，经济系统会产生复杂的博弈过程。

用户购买到虚拟矿机后，即可投票给委托代理人进行记账。代币奖励系统像比特币系统一样，每 4 年减半。矿机的本质其实是另一种特殊的代币。这种特殊的代币如能像其他代币一样自由转移，那么其同样也可以进行二级市场交易。为了限制其流动性，在矿机转移的过程中，矿机会产生 10%的价值损耗。对于半年内转让的矿机，相应的同样会产生折损，其折损规则是线性的，最高 30%。

这个模型是一个代币封顶值拥有一个相对最大值的模型。由于购买矿机会销毁一部分代币，代币封顶值随着矿机增多降低。

8 Grid 主链

Grid 主链是一条由 Grid 操作系统实现的区块链，是整个 Grid 系统的根基。Grid 主链主要由侧链索引系统、以及 Grid 代币（Token）系统构成，其共识机制将使用基于 Grid 代币的代理虚拟矿机共识机制。

8.1 侧链索引系统

侧链索引系统是一个将 Grid 生态内的区块链连接在一起的系统。 Grid 主链会索引两大类侧链，一类侧链是已经有重要应用价值、能够提升 Grid 应用边界的链，譬如比特币。另一类是在 Grid 生态中，基于 Grid 操作系统，能够进一步促进 Grid 经济系统繁荣的侧链，如使用 Grid 代币作为运行费用的侧链。

侧链索引的主要步骤是：主链区块铸造者读取多个侧链信息，并将侧链区块的标识信息整合在一起，构造一个 Merkle Tree，存储这个 Merkle Tree Root 到区块头中。在图 5 侧链索引中，我们最简化的构造了一个侧链区块的 Merkle Tree，在一条侧链要确认一笔交易 TX1 在 BTC 的 1000 区块出现时，只要提供 BTC 的第 1000 区块的 Merkle 证明，以及 TX1 在 BTC Block 1000 的 Merkle 证明，即可完全通过消息的输入及主链的 Merkle Tree Root，证明该笔交易的存在。同样的，在一个具有状态的 Merkle Tree Root 的系统中，如以太坊，我们可以用同样的方式证明一个状态存在于某个区块。

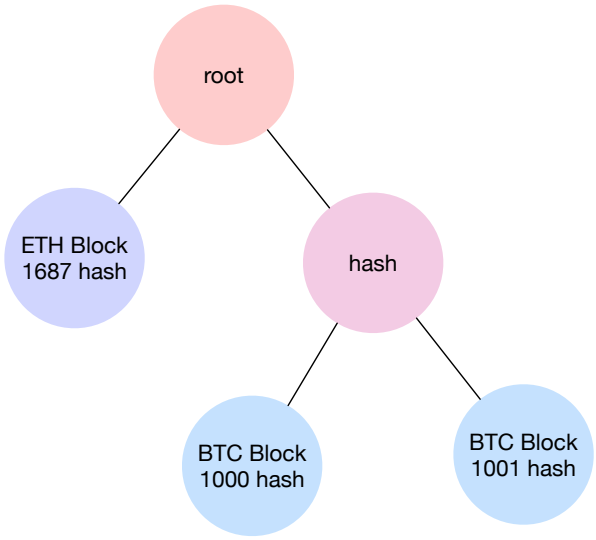


图 5 侧链索引

为了提升验证效率，在构造 Merkle Tree 的过程中，可以不仅仅提供区块的 Hash，还进一步包含区块相对应的交易（图 6 索引交易）及状态（图 7 索引状态）的 Merkle Tree Root。

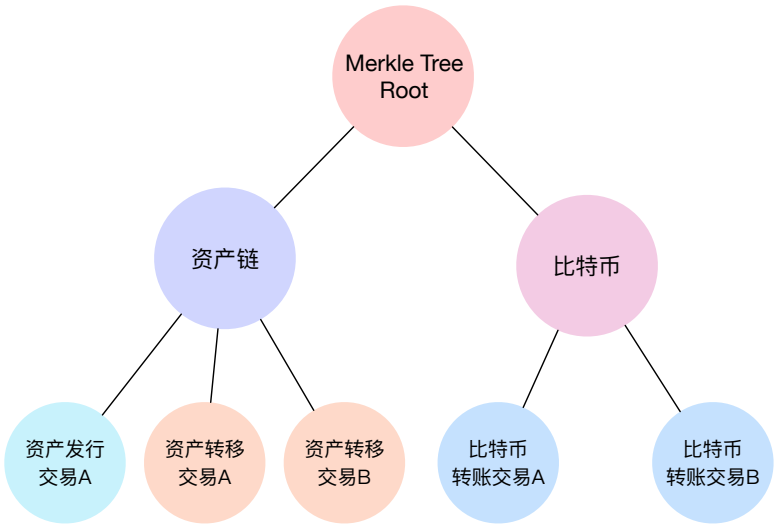


图 6 索引交易

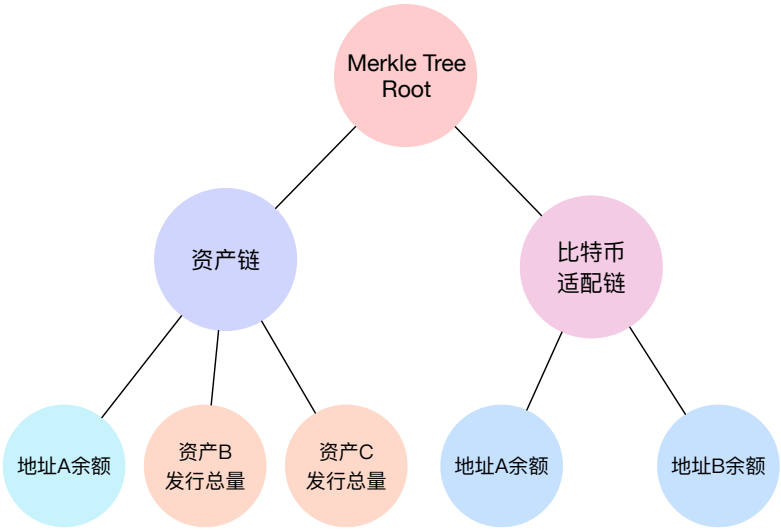


图 7 索引状态

有一个需要讨论的情况，主链在什么时机可以对侧链进行索引。对于一个容易分叉的侧链，主链过早的将其收录，可能会导致一个孤块[16]被收录在区块链中。对于不同的侧链，应选择不同的收录策略。对于比特币这样的网络，从统计角度，在接收到

[16]分离块或孤块是未纳入主区块链的有效区块。它们的成因并不罕见，如两名矿工在几乎相同的时间产出了区块。当然，试图扭转交易数据的攻击也能引发孤块。

一个区块 1 分钟后[17]，如果没有孤块产生，则当前区块不会变成孤块，那么比特币的收录策略即可确定为在比特币区块出现 1 分钟后进行收录。对于 Grid 生态中与主链联合挖矿的侧链，由于其拥有相同的记账人，因此可以对其实时收录。

8.2 Grid 代币系统

Grid 的有序运行需要藉由一个内在的经济激励模块完成。Grid 代币能够在运行了代币模块的 Grid 区块链系统中跨链传输。

由 Grid 主链索引的 Grid 侧链，应当是能够在 Grid 生态下，利用 Grid 代币进行价值传输，并能够通过 Grid 代币向系统支付的链。一个 Grid 侧链在申请被主链索引时，应从 Grid 主链转移一部分 Grid 代币到 Grid 侧链进行锁定，并且在 Grid 侧链收到手续费的情况下，将一部分手续费分配给主链的区块铸造者。一个简单的方式是，侧链使用联合挖矿的方式，将记账权分配给主链的区块铸造者，并且将一部分费用分配给侧链创始人，一部分费用分配给主链。费用比例的动态调整是一个由代币市场动态调整而确定的过程。该设计的主要考虑因素是当主链利益得不到保障时，主链有权停止对侧链的收录，或主链中有提供相同服务的两条侧链处于竞争。

8.3 共识机制

一个稳定高效的区块铸造环境是 Grid 系统运行的良好基础。由于 Grid 区块铸造的主链需要对侧链进行收录，并且 Grid 本身是一个面向云端的企业级服务，因此其运维工作不再像比特币、以太坊一样简单。区块铸造者需要并行同步多条链的信息。虽然 Grid 会将整个云部署过程自动化，但依然面临一个相对高的运维门槛。一个频率稳定且高速的区块铸造策略能够极大程度提升用户体验。

[17] 可以在 https://blockchain.info/zh-cn/orphaned-blocks 查询孤块的产生

比特币的共识机制是 PoW，但其现有运作模式的实质是一个代理机制。矿工会将自己的算力给予矿池，由矿池决定具体的区块铸造行为。基于 PoW、PoS 的共识方案由于铸造节点参与门槛低，区块铸造基于概率而导致时间不均匀等问题（比特币有时会 1 小时甚至更长时间才出块），与 Grid 的设计初衷不相符。

Grid 将采用代理虚拟矿机机制来进行共识。

9 Grid 侧链

被 Grid 主链收录的链称为 Grid 侧链，建议的模式是每一条 Grid 侧链仅以一个核心价值为设计目标。

Grid 建议使用 Grid 操作系统发起新链，开启代币模块，并采用与主链联合挖矿的形式，建立自己的共识机制。为了增进 Grid 生态，被 Grid 主链收录 Grid 侧链应当锁定一部分 Grid 代币并分享一部分手续费给主链。

Grid 侧链如需验证其他侧链上的信息，则需包含 Grid 主链的区块头信息。侧链之间不直接发生关联，验证时使用 Grid 主链提供的 Merkle Tree Root（见图 8 侧链与其他链通过输入消息交互）。

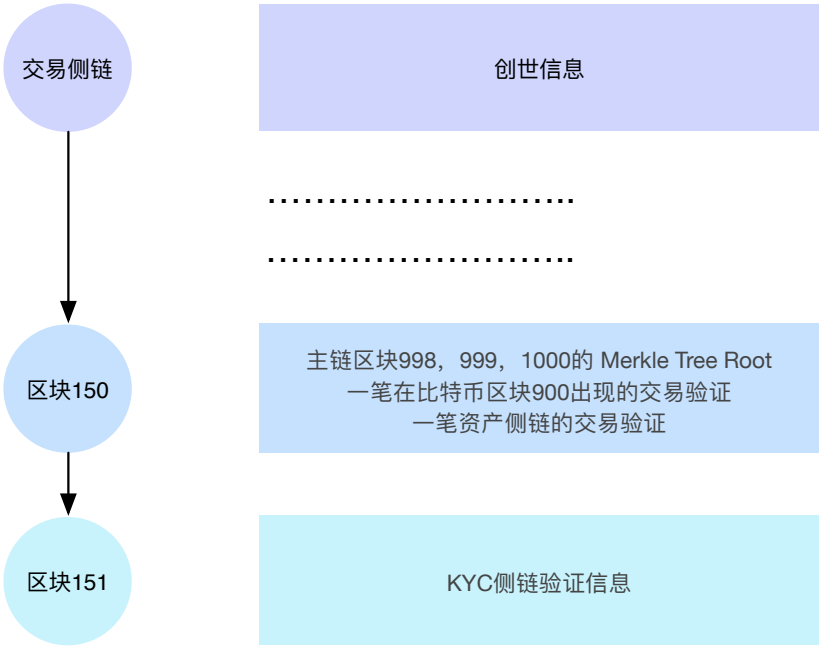


图 8 侧链与其他链通过输入消息交互

9.1 区块链适配器

对于比特币这样的 UTXO[18]系统，其他侧链难以通过简单的输入获取状态，譬如一个地址的可用余额。通过区块链适配器，系统可生成一个与比特币高度相同的并且在头部区块中包含世界状态的 Merkle Tree，有助于其他区快链系统更方便地共享其价值。鉴于比特币的广泛应用及其相对简单的业务逻辑，Grid 有计划实现一条与 Bitcoin 完全兼容的基于 Grid 操作系统的比特币侧链。

9.2 系统内置 Grid 侧链

9.2.1 信息登记认证侧链

登记认证侧链是线上线下行业亟需的价值链。在互联网行业，电商、外卖、网约车等线上线下应用交互已经普及。比特币通过挖矿机制，将现实世界的算力巧妙地融入链上的网络，形成了基础价值的一次有效传递，使比特币经济生态在比特币网络

[18] UTXO 是 Unspent Transaction Output 的缩写，表示一笔未花费的交易输出。比特币系统通过 UTXO 验证交易有效性，是比特币的重要思想之一。

的闭环内完成。进而，供应链金融、物流、存证、征信等业务无可避免的要将其线下机构的信用体系传递到网络中来，不断充实登记认证侧链的通用价值。

9.2.2 数字资产侧链

该链的基本功能是发行数字资产。资产数额较为庞大时，能够迅速挂载在一条独立的链上，若这条独立的链发生性能瓶颈，该链能够再扩展出主侧链结构。

9.2.3 去中心化交易链

在信息登记认证侧链、数字资产侧链以及 BTC 侧链的基础上，我们能够实现一个去中心化交易所的 KYC 与资产充值提现，继而实现挂单、撤单以及成交撮合的逻辑，并据此实现一个去中心化的交易链。

10 一个简单的应用场景

假设我们拥有一条主链，这条主链索引了比特币及一条侧链（该侧链包含主链的 Merkle Tree Root）。下文将阐述如何在该侧链上验证比特币的一笔交易。

首先定义一个私钥，从私钥我们可以推导出比特币的公钥及地址。

侧链收到一条消息，该消息包含比特币交易 Tx，以及到主链 Merkle Tree Root 的验证信息，通过证明该笔 Tx 被主链索引，即能够证明该笔比特币已经支付到了我们定义的私钥上。

一笔系统内资产 X 与比特币资产的交易的实现过程是，B 在侧链内发送一笔交易，该笔交易将锁定 B 的 X 资产，并且检测到 A 发送到比特币到 B 的私钥对应的比特币地址上时，A 有权利将资产 X 转移到 A 的地址。

A 发送了比特币，在检测到主链收录了包含该笔交易并经过了 6 次确认（确认机制可由开发者通过智能合约实现）的情况下，发送一笔交易，该交易包含了上述的比特币充值验证机制，当合约验证了该笔交易后，就会将锁定的 X 资产发送给 A。

上述机制我们使用了比特币的服务，同时比特币的网络不需要进行任何修改。这种机制即便在比特币网络出现了问题不能对外服务甚至被主链索引移除时，历史记录都能够被有效验证——因为所有的验证是基于消息的输入以及主链的 Merkle Tree Root，而非链与链之间的交互。

11 开发计划

Grid 基金会将首先完成 Grid RFC 文档的撰写。该文档为技术文档，主要包括系统的结构与子系统接口定义等。

Grid 基金会将会支持各版本客户端的实现，初期将会实现 Go 及 C#两种客户端的开发。开发目标致力于增进开发人员的开发效率，降低软件的 BUG 率，增加团队间的技术讨论，以及系统架构各组件微服务化的程度。在整个开发过程中，需要做到微服务间的兼容。

12 开源基金会运作

基金会内部设置理事长、副理事长、理事及成员。基金会章程由第一届理事会起草审议及执行。第一届理事长由项目发起人担任，副理事长由承诺锁定代币的持有人指定。会费通过代币支付。软件使用 GPL 授权，商用部分的授权费用通过 Grid 代币支付给基金会，代币费用由基金会年度更新发布。

这样的开放机制设计旨在解决核心代码升级等问题，同时使得生态中各方利益保持一致。

参考文献

[1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[2] Vitalik Buterin. Ethereum White Paper: A Next Generation Smart Contract and Decentralized Application Platform. 2013

[3] Melanie Swan. Blockchain: Blueprint for a new economy. ” O’Reilly Media, Inc.”,2015.

[4] Frederick P. Brooks. The Design of Design: Essays from a Computer Scientist. “Addison-Wesley”, 2010.

[5] Andrew S. Tanenbaum. Modern Operating Systems “Pearson”, 2007

[6] Joseph Poon and Thaddeus Dryja, The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments. 2016

[7] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. 2014.

[8] Hyperledger Whitepaper. 2016

[9] Muhammad Saqib Niaz and Gunter Saake. Merkle Hash Tree based Techniques for Data Integrity of Outsourced Data. 2015

[10] Robert McMillan. The inside story of mt. gox, bitcoin’s 460 dollar million disaster. 2014.

[11] Sunny King, Scott Nadal. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. 2012

[12] David Schwartz, Noah Youngs, and Arthur Britto. The ripple protocol consensus algorithm. Ripple Labs Inc White Paper, 5, 2014.

[13] Leslie Lamport. The Part-Time Parliament. *ACM Transactions on Computer Systems*, 21(2):133–169, May 1998.

[14] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

[15] Leslie Lamport. Time, Clocks, and the Ordering of Events in a Distributed System. *Communications of the ACM*, 21(7):558–565, Jul 1978.

[16] Paul Tak Shing Liu. Medical record system using blockchain, big data and tokenization. *Information and Communications Security*, pages 254–261. Springer, 2016.

[17] Robert Love. Linux Kernel Development. “Addison-Wesley”, 2010

[18] Shawn Wilkinson and Tome Boshevski, Storj: A Peer-to-Peer Cloud Storage Network. 2016.

[19] Contract. URL *https://en.bitcoin.it/wiki/Contract,* 2014.

[20] Mandatory activation of segwit deployment, UASF, BIP 0148. URL *https://github.com/bitcoin/bips/blob/master/bip-0148.mediawiki,* 2017

[21] Smart Property. URL *https://en.bitcoin.it/wiki/Smart_Property*, 2016.