摘要:

目前的区块链技术缺乏使终端用户与开发者连接起来的能力以及缺乏建立大规模业务的技术。因此，我们提出了EOS，一种高性能以及自治的区块链，一个大规模的面向消费者的分布式应用操作系统。

本文概述了EOS基础的背景，愿景和EOS的底层软件架构，我们正在做的目标就是使得EOS能为广泛和多样化的用户群提供智能商业业务。

关键字--- EOS，区块链，智能合约

1. 介绍

数字货币和智能合约的概念已经被讨论了很久，但只有最近几年才取得进步。

本文介绍了基于EOS的EOS.IO软件，作为新的通用智能合约平台。 在这里将会把EOS与三个领先的区块链产品进行比较，因为（a）它们是应用分布式账本技术（DLT）中广泛被认同的产品（b）足够重要，（c）本文作者熟悉。

Bitcoin（Nakamoto 2008），在区块链领域似乎是一个数字现金和智能合约的代名词虽然它吸引了cypherpunks（加密爱好者），媒体和基金持有人的注意，但它没有在商业应用上有所突破。Ethereum (Woods 2014)试图用“永不停止的世界电脑”来实现智能合同平台，Bitshares (Larimer et al 2014) 努力开拓基于区块链的资产交易市场。 数百种山寨币/二代币努力满足各种差异化的需求。 Corda (Brown et al 2016) 则完全放弃了区块链技术，探索出一个了多方工作流程解决方案。

我们真的很接近目标，但从最终用户的角度来看我们仍然没有达到目标。 我们应该从最终用户的角度来看待问题，重新审视用户需求，进而建立一个实用而又有效的区块链基础架构。首先，我们先来总结一下时下分布式账本技术的市场趋势。然后，我们来看看用户需求，以及如何满足他们的需求。然后，我们看一下我们的架构是否足以满足市场需求。

最后，我们会与已有系统做一个对比。有关EOS.IO软件的更多技术细节，请参阅“EOS.IOTechnicalWhitePaper” (Larimer 2017).

Ian Grigg 是一位财务密码学家，同时也是block.one的合伙人，可参看他的blog：http://iang.org/。

说明：

- 本文主要论述EOS.IO软件，它准许一个社区支持EOS区块链。由于该软件是开源的，并且社区只受宪法的约束，所以本文只作陈述，而不代表任何EOS区块链。

- 我尽可能地中立，但难免会存在偏见。为准确起见，已把一些信息排除，而且把一些批评作了修改。

- 本版本还只是草稿，希望各位不吝赐教。如有错误，定当修改。

2. 背景

市场 市场是充满竞争的，无论是对于分布式账本或者是区块链或者是其他任何产品。那么现在市场供应着什么样的产品呢？ 比特币可能可以被视为一条以安全为卖点的区块链，但一条足够强大的链的价值应该由附加在其之上的业务来显现出来。 也许市场也认识到了这一点，Ethereum宣称是一台不可回滚的图灵完备的全球计算机，这个目标可能吸引计算机科学家，但对其他学科来说似乎是难以触及的。 R3建立了Corda，以满足金融机构的需求，这是一个很大的市场，但它同时对于用户来说却相当昂贵以及单一。本部分主要审视从架构来分析这些系统的特征以及其必要性，这些特征透视出行业对于区块链系统的要求以及基准。

共识 使用区块链，我们就一系列事务达成共识，使得任何事务都不会与任何其他事务冲突，无论是在本块中还是在之前的块之中。 也被称为“两军问题“，这是一个有着很丰富历史的问题，使在不同地方的执行者能达成协议，“我知道你看到的就是我看到的”。见图1。

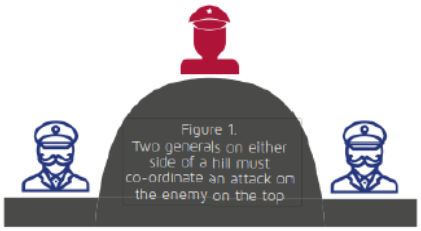


Figure 1.
Two generals on either side of a hill must co-ordinate an attack on the enemy on the top

Bitcoin建立了工作量证明方法，通过共享或分布式账本作为将整个开放式社区结合在一起的方式，其中所有各方都拥有完整的副本。 这个机制就像是在许多矿工之间运行乐透彩票，来确定谁能产生每个块。彩票中的门票是由SHA2碰撞产生的，由于这需要电力来计算，彩票的获奖者将获得一定数量的比特币作为奖励。 实际上，任何人都可以是将军，赢得彩票的人就是那个设定这一刻战争计划的人。接着将军可以选择接受或者以无效为由拒绝接受这个计划/块。

在本白皮书写作时候，完全共享的共享式账本和工作量证明成本耗费巨大，比特币占用了4%，Ethereum占用了11%的资源。Permissioned ledgers（Swanson 2015）不仅阻止了我们想要排除的共享账本的问题，而且还将我们重新带回传统计算机科学认为高效的共识方法，即数据库科学中的方法 - 实用而中心化的设计。同时提出了权益证明算法，exotic crypto graphy 和 secure

enclaves。 Corda (Brown 2016)创建了一种新的共识机制，使用户在一个承载交易的智能合约中选择特点位置。 通过一个内部可变更的名为notaries的服务器并利用上述任何一种方式达成识，Corda将运营成本降低到与当今传统IT基础架构相当的水平。

价值 类似地，有多种方式可以对价值--比如现金--进行交换。20世纪80年代至90年代的智能卡现金（Smartcard money）通过每张卡的内部的数据交换协商来实现原子性卡对卡交易。在同一时期，David Chaum的eCash（Chaum 1983）普及了数字货币的概念，这是一个随机数字，通过盲签名技术可将数字货币从一个用户转移到另一个用户。 三重记账法（Grigg 2005）创立了一个新的方法：每一方都可以看到相同的收据，其中每一个收据都记录了一个人到另一个人的交易。支出和收入的和则为余额。

比特币使用UTXO(未花费交易输出 unspent transaction output)概念，这是一种状态驱动的结构。每个交易记录都会记录以前未使用的金额，并创建一个新的未来可花费的金额。 相比之下，Ethereum虚拟机提供了一个数据库机制，使得可以从数据表中构建货币，灵活性显着提高，但是也造就了攻击的可能。

以下五个各异的机制表明，解决价值传送的方法并不是不可改变的。

基于状态的转账模式 比特币的区块作为一个UTXO列表，在上层，保留了一个状态声明包括这些币/块/链的本质和时间信息。UTXO设计的双重性源于轻量级或“SPV”客户端以在共享式分类帐本中证明其进入的币的需要：只有有限访问能力的客户端只需要从特定区块位置跟踪每个货币的原点，就可以确定新的交易是否正确。接收者不需要证明新增交易之外的任何东西，例如发送者的余额等去确保交易正确。

区块链就是一幅状态的图，这一概念被广泛接受。甚至Ethereum将UTXO替换为更为强大的虚拟机的时候也将状态纳入到共识算法内。在新的区块到达时，每个验证节点从所有的合约中找到状态信息，并经过计算和确认精确的退出状态。

智能合约 比特币通过将验证“脚本”附加到其交易中来增加更多业务逻辑，以提供有限形式的合约，这种合约通常被称为智能合约（Szabo 1994，1997）。 Ethereum的不可回滚的全球图灵完备计算机的概念提供了更全面强大的编程能力，消息传递和数据存储。Corda反对这些设计，以通过命令驱动的更改来验证和确认UTXO状态，但也限制了只有直接访问方拥有访问权限。 Ethereum和Corda都推出了更强大的高级语言来编写智能合约。

性能 比特币受到每秒3笔交易（TPS）的限制，此后的交易可能会被严重拖延。Ethereum似乎有15 TPS的速度限制，最近的一个标志性拥堵事件是有一笔交易支付了2000美金的交易费来试图跳过排队队列。 限制区块链的吞吐量的原因有很多：比如需要验证之前的块，处理新的块和挖矿。 Corda很大程度地避免了这些限制，通过可选择的，以其独立的和本地的公证节点，而这些都不需要取得更广泛的共识即可达成。 每个系统受到的限制只是网络速度。

用例 尽管围绕区块链有很多的炒作，但是目前也只有相对较少的成功使用案例。比特币建立了一个单一的货币，竞争币的爆发，彩色币的失败，缺少全功能的智能合约。Ethereum试图打破这些限制，但迄今为止仍还不清楚结果如何，或者有人会认为在未来有越来越多的人利用来筹集资金，比如基于ERC-20的智能合约能产生稳定的用户量。另外，也许令人惊讶的是，EOS的两个前辈是已经达到了相当的使用量和规模，去中心化交易所（Bitshares）和社交媒体 (Steem)。然而，基于智能合同的愿景其实仍未实现。

治理 对于这个题目来说，比特币的关键特性不在于我们可以使用加密技术来解决什么问题，也不其设计是在足够开放和去中心化的情况下保持相当高的稳定性，而是它必须保持这些特性才能生存下去。所有人都能输入数据这一行为不仅是分布式账本上哈希挖矿的共识模型的键，也是系统生存的关键。以前的数字现金系统失败了，因为有一个中心机构受到各种各样的攻击，显示了其治理模式的失败。就像是比特币时代的中心化交易所往往遭到盗窃，违约，拒绝服务攻击，破产，扣押等问题。

随后，世界分为两部分：一种是以区块链为代表的完全去中心化的开放系统，另一种是完全相反的中心化的带授权的账本，两者之间的关系是不确定的。 在这两种情况下，开放的社区里的分歧也同样引发了用户治理，管理以及治理工作的问题。

在开放的社区里一般做法是买者自负原则，它通过仔细地设置了一个能够满足大部分需求的环境，但能做的事情却被代码限制了。在这个情况下技术团队在技术和链之前划下了一条不可逾越的线。我们认为这更加危险。随着时间推移，比如一些改进的提案会被中心化的权力机构如基金会或者团队提出，而这可能对用户是一种伤害的。典型就是 Bitcoin 和 Ethereum.

相反，在授权的网络或者带围墙的花园模式，限制了只有特定实体才能进入和操作。在这种情况。在这种情况下，各方开立账户，由代理人操作，并以良好的安全的规范进行交易。 无论是明示还是暗示，良好行为通常被认为超出技术定义的层面。不利的是，可能需要高昂的架设和维护成本，每年维护者将收取更多的费用。 这种方法通常在诸如银行等严格监管的市场中被采用，另外Corda正在使用这种方法。这两种状态都不是用户友好的 - 用户通过在买者自负模式上花了太多的钱。而无论是在竞争层面还是社会层面，从“许可”开始发展系统会逐步发展成为成为一个人为鉴别的系统。用户也时常怀疑这两者这两种系统是否可行。

3. 远景

终极目标 我们的用户需要什么？ 在理论上，她想要：

- 了解她的朋友，商业伙伴和客户。
- 与他们沟通
- 能够与他们定下合约：

    微观层面，做点对点的协议，和

    宏观层面，建立一个能够为市场服务的复杂业务。
- 作为业务的必要组成部分，能够保留和提升她的价值（支付账单等）。

    然后，所有必须被安全地完成
- 能够投资于可预测的业务。 这是一个复杂的问题，而这似乎需要三个条件。

    了解行业的生态系统正在发展，而不是正在倒退。

    为前景努力付出，在未来得到合理的回报。

    因为她知道 合同，资产，交易，意图 如果发生问题，她想要能否解决她的困难。

    包括与她的朋友，她的生意和她的资产，快速，便宜，没有不适当的升级。

    我们假设她想要的就是她需要的。

大想法 众所周知，由于某种原因，点对点智能合约和数货币被排除在更广泛的互联网之外。比特币太不安全了，而且它的智能合约是不透明的；而以太坊则太可怕了，太难用了。 Corda是“大公司”。其他系统都有自己的弱点，所有这些都只限于精英开发者，而每个人都有不同的看法。

我们需要的是能为日常每一个人提供智能的业务。 日常使用的分布式应用程序需要建立在在一个全球的区块链上，有足够的性能能让大企业的业务能在其之上运行，足够互相连接到一起，并且安全可靠。即使是华尔街的Gordon Gecko都能和非洲的Mama Biashara进行交易。

目标 我们面前的愿景是一个单一的全球性的基于区块链的智能合约平台，可以随时随地扩大规模，以便在安全和安全的环境中处理长期的商用智能合同。

更实际来说，互联网上有很多具有价值的事物，我们专注于网络上的内容，而暂时先不关注移动和应用程序。 Web应用程序的开发者需要什么？ 我们假设目标用户是网络企业家，因此让我们将从他们的角度去看问题。

核心功能 我们的设想这个系统将可以，每秒处理成千上万个交易的商用智能合约，并且易于使用和安全的语言进行开发。 主要功能包括：

- 使用基于事件架构的高性能消息传递系统
- 委任权益证明共识算法
- 核心是消息，合约主要用来表达意图
- 无论对于开发者还是企业主都易于使用
- 拥有治理能力

以下部分将深入探讨。

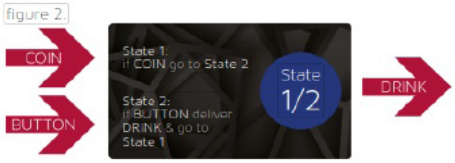
4. 架构

设计哲学 在很大程度上，EOS的软件的实际方法是基于Bitshares和Steem上迭代升级开发过来的大规模高性能区块链系统，以支持最终用户业务。 大多数功能已经被证明行之有效，然后我们将其重组为一个系统并赋予其新的目的 - 用于构建分布式应用程序。

本节介绍了EOS针对以前的实践提出的一些重要的架构差异。 有关更多技术细节，读者可参考EOS.IO技术白皮书（Larimer 2017）。

消息即中介 EOS.IO软件设计理念从更流行的基于状态的共识算法转换为不那么流行的基于事件的共识算法（Grigg，2017-1）。这种方法将事件来源模式（Fowler，2005）与基于事件的区块链结合在一起。

在计算机科学中，确定性状态机被构建为代码，状态（内存）和事件，都被看做是输入输出。每当发生什么事情导致更改时，一个实用的计算机将中间代码保存到内存中，重新启动后，通过读回这些中间代码来恢复自身。 在建立一个实用的状态机时，我们可以选择保存事件或者保存状态，这个选择主要取决于我们要优化的内容。



在图2中，我们要保存显示为红色的信息还是显示为蓝色的状态？ 如果机器选择保存状态的话，我们可能更专注于状态是什么（例如数据库）。 如果选择保存消息，有可能会在询问状态的情况下会更加有用。（例如协议或合法重要日志，如三重记账法（Grigg2005）） 在选择保存状态下重新启动的速度更快，而选择保存消息的话吞吐速度更快。因为用户需要高性能，所以我们的设计会选择保存消息。 重新启动消息或事件会类似于从头开始恢复，因此会非常缓慢，因此我们将会优化一下，将会使用保存检查点技术来重新恢复状态。但是，这里是一个关键的问题，在保存这个状态时，一个行为者仍然受到保存的信息而不是状态的束缚，所以我们可以进行优化，甚至如果需要重新计算检查点。 如何优化这个话题太大了，但我们预测在理论上可以处理每秒300万次交易。

共识 对于构建在消息上的共识模式，EOS.IO使用了委托权益证明算法作为共识算法（DPOS），这是Steem和Bitshares（Larimer 2014）被证明为行之有效的两层的治理结构。 在第一层，出块的方式为每

轮21个块，每个见证人每轮获得一个块，并且被奖励将会发放给那些参与验证传入的消息和生成该消息块的见证人。一个见证人产生的一个块由下一个见证人验证。

类似于比特币的最长链机制，并且在很短的顺序内，见证人们将会聚集在最长的链条上。 被一定个数的见证人接受的区块被声明为不可变的，且这些不可变区块就变成了一个检查点。像工作量证明一样，见证人可以审查（忽略）信息，也可以通过从自己对未来的知识中引入自己的方式进行前瞻性的审视。为了对见证人的不良行为提供治理方法，每一轮的见证人都不断地由社区通过权益证明算法（PoS）选出。 由于这种二级的区块链管理架构选举是见证人而不是区块，因此所谓的“nothing at stake”的弱点不存在。

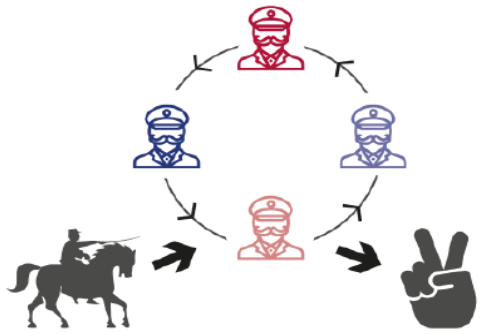


figure 3. Delegation allows replacement of Generals after a bad campaign

实际应用上，一系列将军由选举产生。 在这场选举之后，利益相关者仍有权力取代任何一个有不良意图的将军。

DPOS避免了采矿税，为利益相关者节省了成本。 来自区块奖励能最大程度分配给见证人。由于是由社区选举产生，所以见证人有动机促进和推广社区的发展。

根据宪法，见证人的奖励被限制在每年5%（Larimer 2017-2）。 根据习惯，我们建议见证人将大部分回馈给社区，以实现共同的进步 - 软件改进，争议解决等等。 本着“吃我们自己生产的狗粮”的精神，根据该设计设想，社区投票选出一系列的开放式智能合约来扮演一个基金会的角色。 被称为社区福利合同，该机制强调了DPOS在社区实现直接的在线治理的重要性。

智能合约 该架构更接近合同的性质。将合同视为谈判，承诺和事件的动态表现。我们认为信息是智能合约的最根本的元素，因为它们更好地捕捉了智能合约所有阶段：谈判，意图，履行和违约义务都是比状态更好的信息。

用户将合同视作为的消息互锁处理程序的虚拟构造。 用户可以通过添加消息处理程序并使用其帐户内置的数据库存储其所在的合同的内部位置来将她的帐户转换为合同代理。几个消息处理程序一起工作可以处理消息流，以便通过其生命周期执行完整的合同或合法的协议。

从合同的角度来看，消息的到达，接受和处理是比状态更简单的抽象。 比如在交易所上看到的订单簿：接受买入和出售的报价。 当时机到来，它必须交叉计算价格，然后给双方发出订单。

基于消息传递的系统中的订单簿就是一个其传入的消息和传出的消息集合，这是一个相对容易处理的任务。 相比之下，在完全以状态为基础的系统中，所有交易者必须在将最终状态提交给区块链之前，向所有交易对手（包括数量和价格）谈判可接受的价格等。 这意味着交易者在同意之前

会先看看解决方案，然后再进行下去。 在实践中，唯一已知的解决这个问题的方法是使用代理和消息传递。 代理接收已提交的消息，决定输出结果，然后发送确认结果的消息。

可用性 一个区块链的直接用户是为最终用户创建 web 应用程序的开发者。要支撑一个最终用户, 软件必须首先支持开发人员, 并且必须以帮助开发人员支持其用户。对开发人员的支持包括 (a) 工具、(b) 语言和 (c) 环境。

在大的范围里, 开发人员将得到一个基于 web 的工具包, 它提供了一个完全的服务框架, 用于构建基于区块链的分布式 web 应用程序。帐户、命名、权限、恢复、数据库存储、调度、身份验证和跨程序异步通信都包含在内。该体系结构的目标是为应用程序的开发人员提供一个操作系统, 并将其集中在 web 上, 因为这是大批用户的所在的地方。

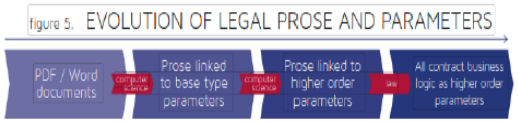
figure 4. Tensions between stakeholders in a blockchain

语言 在我们的行业规模的分布式应用程序的环境中, 编写智能合约的语言在影响列表中是很高的。 EOS.IO 中的大部分的结构已经被证明为有着坚实的基础, 并被应用在Bitshares 和Steem上, 而智能合约部分则为未知领域。

这使得我们有必要仔细分析采用哪种语言。从智能合约技术的角度来看, 成功关键是: 各方参与者、开发者和操作者。

- 参与方需要的合约，首先是一个真正的合约。参与方需要的合约也要是可协商的(negotiable)，可读的，清晰明确的。它需要忠实地理解人类的意图。如果能够解决争端以及具有强制执行能力的话，那就更好了。
- 开发者需要语言和系统是容易学习的，有表现力的(expressive)，安全的。
- 区块链的操作者—区块的生产者以及完全节点app—需要合约是可扩展的，同时能够赚取一定的利润，利息，而不需要关心别人的目的或开发者的意图。

我们把参与方的需求放在首位，这就需要我们把纯法律文本与计算机代码进行融合，用一些参数把他们粘合起来，以“驱动这笔交易“，并且能在多个合约中重用文本和代码。许多研究都致力于把这法律文本与代码融合起来，要么融合为更高序列的参数，要么融合为一个合法地特定域语言，但是目前为止都没有成功。这是一个悬而未决的研究领域。



沿着这些线，我们的第一个诱惑是给开发者的：一个基于wren的源解析的脚本语言，同时能

定制一个管理合约消息处理器的设计的功能。示例代码片段如下：

```
apply:
// assuming all prior steps pass,
// perform the state transition
// that updates balances and/or
// creates a new account for receiver
                var from =
          Balance[message.from]
                var to = Balance.find( action.to )
                from.bal = from.bal -
          action.amount
                to.bal = to.bal + action.amount
```

这种混合代码——wren代码是很容易学习和阅读的，这就使得它是作为自动合约化(automated contracting)的理想语言。但是，它有点慢：一个普通的交易就需要1000tps，这使得我们没法满足操作者—我们的生产者以及应用事务—的需求。

由于我们想要把性能提升100倍，我们团队就转向了wasm(webassembly)，这是一种新的中间层语言。EOS框架使用wasm后的第一次试验，交易性能就达到了50000tps。

然而，wasm把困难从操作者抛给了参与方：现在对于一个合约来说有三个观察角度：法律文本，初始的c代码以及wasm代码。

因此我们就有理由问一个问题：参与方同意的合约是什么？我想直面这个问题。过去差不多二十年，我在网上看到了不少的合约，其中成百上千的合约都出现了纠纷，很多争论或疑惑的关键都在于：合约的意思到底是什么。即使是DAO事件，这件事情的主因还是安全性，不能为了把钱拿回来就随意的想改什么就改什么。却不去解决关于事实的理解，事实的涵义，以及权力方面的争端。这涉及到在法庭上，有多少争端是涉及理解与疑惑的，有多少是由强权操控的。我对此并不乐观。

经历了DAO事件以及其它的一些事情，我觉得一个合约的规则过于教条了。然而，对于DLT空间的未受管理的部分而言，还有机会释放合约的组件以达到更佳的性能。同时，我们要关注治理问题，要能处理争端，使参与方能解决问题。

在写这篇文章的时候，能为合约开发者所用的语言仍然在开放中。不论是wasm，或wren或是其它，我们仍然需要搭建架构，以获得更好的性能与可用性。每一个命名的消息处理器都需要为每个静态的、只读的、可读写的代码提供标记，而这些部分都还可以优化。为了消除再次生成的问题，正在执行的消息必须标记直到完成，或者在失败的时候扔掉。对于熟悉数据库的人，我们提供了类sql的表结构，使得他们能够轻松上手。加密功能就不需要开发者关心，而且也是不可见的。

对于DLT的空间来说，竞争条件都是在内部。wren是一门小巧紧凑的语言。而wasm还没有标准化。wasm的早期工具是以c和c++为目标，c和c++虽然流行，但是比较

复杂，相较于高级语言来说的话，比如wren。从长期来看，这些挑战不应是不可逾越的，wasm项目已经能与大多数语言协同工作了，而DApp里的代码是在网站而不是在处理器上的。能接受多种流行语言的特性是很有吸引力的，也是Corda jvm的一项优势，但是，比特币或以太坊却无法轻易做到这点。结论就是，供开发者选择的语言和工具有很多。我们希望选择一种易于阅读和理解的脚本语言，同时也是安全的，可扩展的语言。但是，鉴于目前的情况，必须做点妥协。

治理 让我们转换到运行环境上。现实中，事情常常出错。我们的希望是既能减少出错的频率，也能减少出错的代价，但是出错是无法彻底根除的，我们只能在错误发生的时候提供补救的方法。

EOS.IO软件假设所以使用这个区块链网络的成员都是基于这个宪法，并一致同意所有社区成员都服从于此宪法。

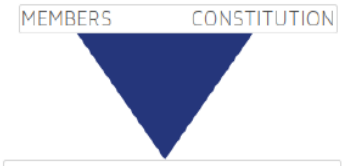


figure 6. Members forge a Community with a Constitution

为了社区的利益，宪法定下了若干规则。宪法的治理包含三个方面：解决争端的仲裁权，选择区块的区块生产者，以及社区公投。这三者互相支持也互相制衡。公投由社区发起，用于投票选举生产者和仲裁员，以及投票决定对代码和宪法的修改。仲裁员可以颁布规则以解决争端，以及决定重大修改比如硬分叉。区块生产者则负责技术部分，禁止错误的交易或者生成新的交易作为补救。仲裁员颁布规则，由生产者执行，或者用户会寻求外部的强制执行。

这样的制衡机制保证了，没有人能一家独大，包揽权力。即使是创始人或开发者，也没有多少能力去干涉社区成员的权力。硬分叉或其它的升级都有一个定义好的路径，而成员间的争端则会安排到适当的地方解决，然后回到正途。上面的治理机制的一个更大的好处是，整个过程都是公开透明的，只需要在代码中写好合约处理器必须接受并管理争端，处理公投以及类似的事情。

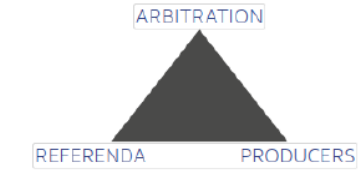


figure 7. Community can appoint governors to manage responsibilities

为了让这样的机构运转起来，用户就必须同意宪法，该宪法授权生产者选择区块，把争端交由仲裁解决。同时，宪法也以区块链的形式规定了法律权力，声明每位成员获取相应的权利，同时支持其它成员的相应权利。这种把你的权利与其它成员的权利进行交易的方式，构成了整个社区的基石，社区既包含了对平台的使用，也意味着对宪法的认可。

因此，即使社区是在内部进行治理的，我们仍然保留了公共的入口。作为一项治理机制，宪法更应该建立一个围栏而不是一堵墙，看门人应当是一个交易或者签名。

5.对比

比特币 作为第一个发布的也是最成功的加密货币，比特币就是一个条基线。当时，作为“第一个”，它的缺点也一样明显：UTXO的验证模式意味着，需要额外的代码才能处理复杂的智能商务(smart business)。状态被很好地锁在链上，但是协调的工作却由应用来完成。而且它对于资产没有很好的框架，尤其是包含btc的交易。比特币还缺乏一个经过深思熟虑的治理层，其结果就是升级变得非常困难，社区内部乱成一团。比如，正是由于缺乏治理，人为的3tps的限制使得比特币无法扩展。

以太坊 为了改正比特币的缺点，以太坊建立了图灵完备的虚拟机。但它仍然有几个明显的不足。首先，它需要耗费大量资源，执行数千的程序才能找到共识，这使得资源阻塞到了15tps的水平。其次，由于只使用一种语言，vm，工具包等等都限制了开发者的能力。第三，它还有创始委员会的问题，由少数大股东把持，而拒绝合理的治理机制。现在对以太坊的使用，主要是那些想终结以太坊或与之竞争的项目，用来筹集资金。极少的用例取得成功，这表明以太坊的智能合约概念在成功之前还有很长的路要走。

Corda Corda最主要的区别就是它不是区块链，而是一个参与方(party) 对参与方(party)工作流的框架。参与方不是把合约和行为提交到区块链中，而是交易消息，通过公证人达成共识。它的特点在于机密性，高性能，以及让参与方控制合约的能力(无论成功还是失败)。然而，工作流只适合参与方数量很小的情况，因此，在发行资产方面，尤其是现金以及现金支配的交易的情况下，它的缺点就十分明显。它的另一个缺点是，它为监管行业设计的墙保护方式，对于小公司来说没有吸引力。

6.结论

用户体验 对于像EOS这样的区块链来说，它的直接用户就是企业以及开发去中心应用或DApp的开发者。他们的用户就是零售，金融，媒体等常规用户。后一种消费者不需要知道区块链是什么。因此，我们的目标是提供给开发者一个平台，使得

他们可以开发广泛的商业逻辑，而通信机制却是隐藏的。

DApp的开发者将会得到一个强大的账号，许可平台和消息发送平台。用户界面是用户所熟悉的—用来搭建网站的webkit工具包，以及访问区块链的能力。这个方式称之为“区块链的操作系统”。

事实上，区块链被用以隐藏起来，对于使用者来说事实上可以看成没有必要使用区块链的，比如steem，一个开看作架设在区块链的上的博客而已。

用例 一个EOS区块链是一个高性能的消息处理平台。常见的用例将包括供应链，资源管理，用户消息比如社交媒体，资产发布，交易，汇款账户以及博弈。

典型的用例可能就是Uber。乘车共享是基于为司机与乘客都设定了行为标准。如果司机和乘客是同一个社区的成员，那么就会有一个好处：责任的基础和行为的标准将会由社区宪法以及争端处理机制涵盖，那么他们的合约将是双边的，这就把监管难度降到最低。由于合约可以是双边的，业务流程可以如此拆分：在市场上追踪乘客，追踪可用的车辆，找到有匹配的话，就协商合约，搭客，完成，付钱，而且社会化追踪可以用DApp来完成。

社区 为了支持商业，我们就必须解决问题。为了拓展解决问题的能力，这就必须由社区自己来完成，这就意味着这种能力必须包含在架构之中。为了让社区进步，我们必须保留公共入口，公共入口提供了一些工具，用户能用于治理。用户希望能自己决定他们该冒的风险和义务。

当捆绑成为一个宪法下的社区时，用户将明白权利，义务都是基础标准，正如宪法中表诉的一样。此外，可靠的名称以及一个信任网络可以减少互联网的匿名性，同时让人们觉得他们的重要性。

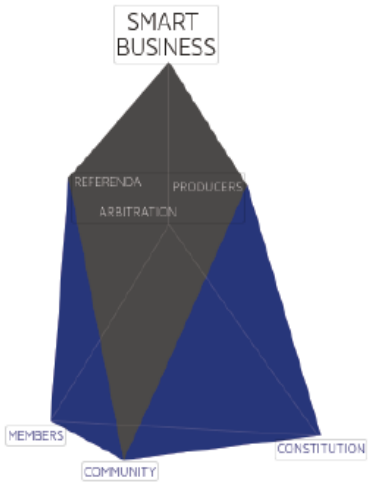


figure 8. The point is Smart Business

致谢

本文得到了Brendan Blumer, Arthur Doohan, Dan Larimer, Wendy Lee, Aaron Leibling, Konstantinos Sgantzos, Joseph VaughnPerling, Kokuei Yuan等人的诸多有用反馈。

参考

[1] Richard Brown, James Carlyle, Ian Grigg, Mike Hearn, “Corda: an Introduction” 2016

[2] David Chaum, “Blind Signatures for Untraceable Payments”, 1982 UC Santa Barbara http://blog.koehntopp.de/ uploads/ Chaum.BlindSigForPayment.1982.PDF

[3] Christopher D. Clack (1), Vikram A. Bakshi, Lee Braine “Smart Contract Templates: foundations, design landscape and research directions”, 2016

[4] Christopher D. Clack (2), Vikram A. Bakshi, Lee Braine “Smart Contract Templates: essential requirements and design options”, 2016

[5] Martin Fowler, “Event Sourcing”, 2005 https:// martinfowler.com/eaaDev/ EventSourcing.html

[6] Ian Grigg, “The Ricardian Contract,” 2004

[7] Ian Grigg, “Triple Entry Accounting,” 2005

[8] Ian Grigg, “The Sum of All Chains - Let’s Converge,”

2015

[9] Ian Grigg, blog post “The Message is the Medium,” 2017-1
[10] Ian Grigg, blog post “Seeking Consensus on Consensus,” 2017-2

[11] Ian Grigg, blog post “A Principled Approach to Blockchain Governance” 2017-3
[12] Vinay Gupta, interview “Bitcoin Cannot be divorced from pre-existing political theory,” 2014

[13] Daniel Larimer, “Delegated Proof-of-Stake (DPOS)” 2014.
[14] Daniel Larimer, Charles Hoskinson, Stan Larimer, “A Peer-to-Peer Polymorphic Digital Asset Exchange” 2014. [15] Dan Larimer, “EOS.IO Technical White Paper” block. one 2017 https://github.com/EOSIO/ Documentation/blob/ master/ TechnicalWhitePaper.md
[16] Dan Larimer, block post “Implementing a Hypothetical Currency Application on EOS,” 2017-1 https:// steemit.com/eos/ @eosio/implementing-a-hypothetical-currency- application-on-eos

[17] Dan Larimer, blog post “What could a blockchain Constitution look like?” 2017-2

[18] Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System ” 2008

[19] Tim Swanson, “Consensus-as-a-Service” 2015 http://www.ofnumbers.com/wp-content/ uploads/2015/04/ Permissioned- distributed-ledgers.pdf
[20] Nick Szabo, “Smart Contracts”, 1994

[21] Nick Szabo, “Formalizing and Securing Relationships

on Public Networks”, 1997
[22] Gavin Woods, “Ethereum: A Secure Decentralised Generalised Transaction Ledger”, 2014