Celes Chain

技术白皮书

版本号:Beta 0.7
二零一八年一月

Celes链

白皮书摘要

本篇白皮书描述了Celes Chain（本文有时候称之为“CC技术平台”或“C链”或“CC”）的顶层设计。CC 是一个创新的区块链平台，为在这个平台上运行的各类金融服务应用提供底层信息服务与技术支持，并为相关监管和合规机构留有接口，提升监管效率。本白皮书提供了我们在设计Celes Chain 背后的逻辑，宏观理念，系统设计和其他非技术类相关信息。此版本为beta版，具体内容今后仍会有修正，敬请读者留意。

Celes Chain 使用去中心化底层共识和区块链技术支持各类金融应用，包括但不限于：各类信息，服务和跨市场的应用。CC允许各类型的参与者，包括了相关监管和合规机构。CC技术平台不但可以支持大规模的应用，并能为这些应用提供安全的服务。

我们使用时分多重证明共识协议[1]，这是一种在“去中心化”与“效率”之间取得较好平衡的共识。这套共识使我们的CC技术平台可以做到：（1）底层矿机在挖矿竞争达到真正的去中心化，且（2）底层矿机有足够的动力维持一定水平的硬件资源稳定的支持的CC平台上的各项应用。

我们将会使用和开发CC特有的的脚本语言，Celes Chain Services Lanaguage (CC服务语言，CCSL), 一个强大的，直观，简洁且针对各项金融服务而开发的脚本语言。CC平台上的各类参与者可以使用强大的CCSL组合自己的金融和商业逻辑，撰写智能合约(smart contracts)[2]以及开发相关应用。我们同时将一些行业共通的逻辑部分抽取出来，类似于主协议（master agreements)的性质，开发成脚本语言的模版，方便用户可以直接引用。同时，监管机构可以根据法律规定，对各类应用，数据和用户行为进行监管。同时，我们还会把部分协议，应用，代码和数据编译成为符合先行法律规范的文本，方便进一步合规和监管。此外，这些应用的数据，合约以及相关信息都将会以加密后的形式作为“记录”放入底层共识“公链”内，以保证这些“记录”不会被篡改或非法窃取。

我们在白皮书的开始部分，将会对目前的金融行业状况以及现有的各类区块链项目做一个简单的现状描述和回顾，并讨论我们是如何思考和解决这些现状带来的问题：包括”如何解决效率与去中心化的矛盾，如何达到大规模的应用，如何保持客户信息和安全性以及监管与合规问题等“，从而完成了Celes Chain的整体构思和设计，打造一个区块链上的“华尔街”，并为各大金融机构和用户服务。然后我们会讨论CC的层级设计之间的关系，以及代币产品在各个产品之间如何流动和保障CC的运行的稳定性。

目 录

Contents

01 项目介绍

1.1 目前的金融系统

千禧年以来，以华尔街为首的全球金融市场曾进入了一段金融产品创新和爆发的时期。但由于高杠杆的使用以及产品的复杂性，2008年爆发了一次全球性的金融危机，大众对银行和金融机构都不同程度的失去了信心 [3]。为了挽救金融系统，提振大众的信心以及加速生产力的发展，各国政府都不约而同的走上了增发货币，注入流动性[4]以及收紧监管的道路 [5]。增发货币和注入流动性的政策，一方面确实扭转了全球经济下滑的趋势开始走向好转，另一方面也增加了大众对货币贬值和通货膨胀的隐忧。

此外，在监管的方面，欧洲抓紧了MIFID2的监管实施 [6]，而美国也实行了Volcker Rules以及Dodd Franks法案 [7]，进一步监管金融机构，尤其是OTC的交易。这也就导致了近年来的金融机构的监管成本逐渐增高，缺乏创新和活力，使中小金融机构更难生存，增加了金融行业的垄断程度，有更多的银行和金融机构变成为了“大到不能倒”，但又难以改变自身[8]。

1.2 现有的区块链技术

随着密码学的兴起，许多先驱都开始了对密码学货币的探索，并试图尝试针对目前金融系统的缺憾做出一些改进：泛滥的流动性以及成本高昂的监管。

1.2.1 比特币

比特币[9]是世界上第一个有实际广泛应用的去中心化密码货币。比特币使用的工作证明(Proof of Work，PoW)，在接近10年的实践过程中，证明了这种共识可以有效的解决去中心化账本所可能带来的“拜占庭将军问题”[10]。其基本原理是各个全节点都存储和验证全部的历史账本，然后通过PoW来达成全网络对下一部账本的唯一共识。

由于计算工作证明需要提前投入计算机算力，因此可以有效的避免黑客的攻击，使之成为一个无需要信任和中介清算的，点对点的电子现金系统。由于矿工付出的算力努力维护了比特币网络的安全，比特币网络全体节点达成共识，一致为该名矿工增发一定数量的比特币作为固定奖励。该固定奖励每四年减半，所以比特币总数收敛于两千一百万枚。由于比特币的铸币过程是由挖矿产生的，所以比特币的铸币过程也是有章可循且提前规定好的，且整个网络的比特币数量最终恒定，解决了超发货币产生的通胀问题。目前比特币是世界上去中心化程度最高的密码货币。但是，比特币同时也存

在清算速度慢[11]，浪费电力[12]以及不能支持应用[13]等遗留问题。

1.2.2 以太坊

以太坊是目前世界上第二大的区块链网络[14]。以太坊的发明主要是为了解决比特币所遗留下来的两个问题。以太坊增加了“图灵完备”的脚本语言以支持“智能合约”[15]。这些智能合约可以在以太坊网络上自我运行，且永久储存，不会受到篡改。通过这些“智能合约”，用户可以在以太坊上开发去中心化应用（DApps，Decentralized Apps）[15]。以太坊同样使用PoW共识“选举”出节点来运行这些智能合约。我们通过比特币已经了解了PoW可以使网络达到真正的去中心化，所以与比特币网络类似，以太坊网络也是去中心化的。

然而，由于以太坊是一条“公链”，交易对手之间的智能合约都会被其他不想干的三方看见，这就导致以太坊并不适合运行一些需要保护隐私和用户信息安全的应用。所以为了运行这一类应用，还需要做出一些相应的修改。巴克莱，瑞士银行和瑞士信贷银行都尝试使用“修改过的”以太坊网络来推进他们内部的MIFID2合规进度[16]。根据ibtimes的报道，该修改过的“以太坊网络”有可能牺牲了“去中心化”使用一条私链或者联盟链以解决用户信息安全问题[17]。

此外以太坊也同时存在节点之间竞争过于激烈，网络节点没有足够的动力去投资能运行大规模应用的硬件资源。2017年底的备受欢迎的“密码猫”出现在了以太坊网络内，导致以太坊系统拥堵，其他正常的交易都遇到了不同程度的延迟问题。

1.2.3 其他技术

其他先驱者也都在相应的作出尝试，试图通过修改现有的共识体系来创建一个可以解决比特币或者以太坊问题的，可以实用的区块链公链。Daniel 提出的EOS.io项目使用了Delegated Proof of Statek (DPoS)共识[18]，利用一种选举人代议制度而不是挖矿的方法达成共识。这种方法的明显好处是选举人团选出的代议人群是一个相对稳定和合作的群体，会比以太坊更有动力去投资硬件资源。然而，有议论表示DPoS并不能组织一个寡头通过多个id控制了整个代议团队而控制了整个EOS网络，成为了一个真正的中心化网络[19]。

还有其他组织，如R3或者Digital Asset Holdings，都尝试使用联盟链或者其他妥协了的方法。虽然能高效的执行了智能合约，且也能对监管和法规给出了解决方案，但这都是或多或少牺牲了“中心化”的前提才达成的。由此可见，如果需要解决“去中心化”与“效率”和“隐私”这一对矛盾，还是需要从底层共识协议开始思考。

1.3 Celes Chain

Celes Chain，CC 是一个为各类金融应用提供底层IT服务的去中心化区块链平台。我们要打造一个区块链上的“华尔街”为广大金融机构和用户提供服务。我们的使用了创新的共识算法，可以做到真正的去中心化，兼顾效率。此外我们还将要设计专门的脚本语言，支持智能合约以及各类应用。我们在设计上全方位的支持了高效监管，并留有专门的监管合规编译层以求达到最大的限度的当地的法规合规。该监管合规编译层负责将智能合约编译成符合语义的法律文档。

我们在系统设计上，采取了类似TCP/IP[ref]的协议层次（参考图1）分为应用层，监管合规编译层以及数据层以及底层共识层。确保系统在各层级之间能互相通信。

参与者

图1 Celes Chain 系统构造图

02 Celes Chain 的体系综合叙述

Celes Chain的综合描述分为使用者群体描述和若干模块层的逻辑结构描述两部分，且如图2所见，各使用者群体所对应的系统架构是一样的，但是具体来说权限和解释有不同，这会在下一章作出阐述。

CCCeles链的体系图：

图2 Celes Chain的使用群体以及系统架构

2.1 使用者群体描述

Celes Chain的使用者群体包括：参与者，监管者，用户。

参与者(Participator)主要是指机构使用者，在CC上编写商业逻辑，开发相关应用与提供服务供用户使用。和常见的金融机构类似，参与者拥有其商业逻辑和应用的所有权，并对用户的数据安全负责。

监管者(Regulators and Administrators)，包括通常意义的监管合规机构以及合资格的CC网络监管者，主要负责审查CC上的各项服务是合规，拥有比参与者更大的权限

。简单来说，监管者是CC里面的“超级管理员”，从各个维度和方向上对CC及其应用进行监管。

用户(Users)是指（1）使用CC上应用服务的单位或者个人，（2）维护底层共识和CC运行的网络节点。

2.2 监管者的角色描述与CC的合规努力

我们专门拿出这个小节来具体描述CC是怎么配合各国监管和法律，并为各参与者提供底层服务的。首先，我们设定了监管者这个特殊使用者组别，赋予这个监管者最大的权限可以接触整个CC里面的所有个层级，所有数据和使用者群体信息。监管者这个权限交给合资格的金融从业监管者以及CC网络管理员。

监管者同时也对不同类别的应用和用户群体进行规范，保证各类应用以及服务不会交给不适合的用户群体去使用而产生合规的问题。

此外Celes链在设计智能合约CC脚本语言的时候，将会采用能配合不同法律体系的脚本语言版本，力求参与者群体可以自由选择所需要符合法律合规的方式，最终达到代码即法律合规文本的目标。然而各国法律合规会有相当的不同，也存在时效性的问题，所以CC还有一个的法律与合规编译层负责将这些代码编译成现实法律可读的文本。

由于CC采用“去中心化”的区块链方式记录关键数据和见证，监管者都可以获得任何数据和记录，所以目前金融界应用各类合规报备的成本，如MIFID2等，在CC上做应用的话，就会有较大的节省。同时因为区块链的存在，CC上的服务理论上是不存在隐瞒或者曲解交易数据的情况。Ceres链从“使用者”，“代码”和“文本”三个方向努力，配合监管，力求在这个区块链上的开发服务都能合规。

2.3 各模块层的设定

Celes链分为应用模块层，法律合规编译层，数据层和底层共识层四部份，且其中：

应用模块层主要解决5个问题：（1）与使用者群体的交流，（2）为参与者和监管者提供商业逻辑的应用平台，包括CC脚本语言以及CC虚拟机。（3）参与者生成的的应用经过监管者，存入CC应用“商店”，供其他使用者群体使用。（4）为了提高效率，相同功能类别下的参与者和使用者对模版的共同理解与共识可以做成模版，并在今后的应用中调用该模版以节省资源和成本。（5）CC脚本语言在设计的时候以及采用与各国法律体系兼容的方式，并且使用CC脚本语言编写产生的代码，模版与应用都由专门的法律合规翻译模块产生法律文本并记录在数据层。

法律合规编译层是CC为了迎合监管要求所特别设定的一个层级。所有用CC脚本语言生产的应用，代码，模版都要经过法律合规编译层产生能与相关法律匹配的法律合规文本，并由相关对手方签发有效，符合法律效力。使用群体在使用这个层的时候可以选择使用那个法律体系或者国家的文本样式。

数据层是CC存储和通信各类数据与见证的地方。数据大致分为三类（1）需要见证的数据，清算或者签名等，（2）商业逻辑，用户信息，合同，智能合约和数据文本 （3）其他数据等。CC使用者群体根据成本，保密性和”去中心化“的需要，选择（1）加密数据写入公链（2）摘要见证写入公链（3）数据写入应用自己的数据库（4）数据写入公用数据库。

底层共识模块层使用了时分多重证明协议。时分多重证明协议使用了PoW挖“燃木”，并异步得使用“燃木”进行PoB方式产生代币和产生区块的方式达成共识。这使CC可以在去中心化与效率之间取得一个较好的平衡，给予底层矿工足够的动力投资稳定的硬件资源的支持，使基于这种公链共识协议的CC的效率可以媲美于一般的联盟链或者私链[1]。

03 应用模块层 (Application Layer)

从逻辑上来看，应用模块层分为使用者界面(UI Sub Layer)子层，商业逻辑子层（Business Logic Sub Layer), 以及CC虚拟机（CC Virtrual Machine)四部分。

使用者界面子层(UI Sub Layer) 主要是解决使用者与CC各类应用的交互问题，包括但不限于：提供界面，表单，各类输入，输出/显示，身份验证入口，跨平台以及API服务等等。

CC 商业逻辑子层(CC Business Logic Sub Layer) 主要是为使用者提供编写商业逻辑：这包括代码，应用，智能合约，以及智能清算等服务。CC商业逻辑的编写是基于CC专门开发的一套CC脚本语言。这套脚本语言除了能提供“图灵完备”（Turing Complete)的逻辑[ref]，还留有与显示法律合规兼容的功能。开发好的商业逻辑文档放入下一层级即数据库层储存，并由监管者检查放行，放入CC应用商店内供其他使用者或者用户使用。某一项逻辑以内的参与者共同同意签署对某一项模版的共识，这就可以极大的加快商业逻辑的开发速度。

CC虚拟机（CC Virtual Machine) 是指来自底层公链节点以及相应节点的计算机资源的抽象描述[ref]。虚拟机通过占用其节点资源来支持和运行CC商业逻辑，这就包括代码，应用，清算和数据处理等等。虚拟机也会相应获得一定的回报。与虚拟机相应的节点的分配机制，回报与公链共识问题会在后面底层共识以及代币循环的部分章节有进一步阐述。

3.1 使用者界面子层 (UI Sub Layer)

用户界面子层面向使用者，可使用其他合适的用户界面语言编写，并与下一个子层商业逻辑子层进行对接。在这一层级，对外可适用相应的界面脚本语言，针对不同的平台/跨平台的进行编写。CC平台同时会提供相应的API，为专业使用者在不同的平台上实施程序化策略，下载各类数据，甚至养殖API经济(API Economics)，对数据或者内容进行批发和零售。此外，也可以通过VR/AR等体感交互，以增强使用者的用户体验。API经济与AR/VR不是本白皮书的讨论重点，在这里就不做进一步展开描述。

3.2 CC商业逻辑子层 (CC Business Logic Sub Layer)

这一层级主要是为使用者提供编写**商业逻辑**。在我们Celes链里，商业逻辑被定义为：代码，应用，智能合约，以及智能清算等服务。

我们CC将会专门开发的一套CC脚本语言。这套脚本语言除了能提供“图灵完备”（Turing Complete)的逻辑，这样使开发者可以开发完整的逻辑。此外，针对所编写的逻辑，CC会进行运行该逻辑所消耗的计算机资源进行评估。关于计算机资源的定义与CC的生态环境在后面的章节有所描述。开发好的商业逻辑交给虚拟机在去中心化的网络内自动运行。商业逻辑一旦生成，就不能被篡改。

美国证监会SEC认为，计算机代码可以是具有法律效力的合同[20]。我们在开发CC脚本语言的时候，也会把如何将代码与法律做好对接的情况考虑进去。我们会尝试针对不同法系（大陆法系或者欧美法系），甚至更细致的针对不同国家，不同监管机构开发相应的脚本语言。我们力求能开发出同时具备计算机脚本语言基本功能的，但在代码描述上又尽可能与法系合规描述近似的新一代脚本语言。

开发好的商业逻辑文档放入下一层级即数据库层储存，并由监管者检查放行，放入CC应用商店内供其他使用者或者用户使用。CC的商业逻辑与应用商店将于苹果的应用商店类似，应用开发者拥有该应用一定程度的权益。显而易见，应用是否能上架或者下架，除了有监管者负责监督，底层矿工节点也会根据“经济性”原理选择是否执行该应用。换句话说，在CC里面，任何应用都应该产生价值，且该价值要大于其的运行成本。

3.2.1 CC智能脚本语言（CC Smart Script Languages, CSSL）

CC智能脚本语言，CSSL用于开发CC的商业逻辑，包括代码，应用，智能合约以及只能清算等服务。CSSL将会具备：图灵完备，消耗评估，自动执行，法律合规兼容性，简单易用以及隐私保护等特点。

◆ 图灵完备

首先CSSL会选择与以太坊类似的图灵完备的脚本语言，即包括了循环的逻辑。然而为了保持网络稳定性，CC会对最大的循环次数做出一定限制。根据通用图灵机的概念，图灵完备就是现代编程语言所能拥有的，可以达到的冯诺伊曼体系结构所能达到的最高计算力。一般来说，除了一部分脚本语言（如比特币自带的脚本语言）以外，大部分的计算机语言都是图灵完备的。使用图灵完备的脚本语言，可以使CSSL在逻辑上做到和其他编程语言兼容，并在理论上能够实现任何其他语言所能实现的逻辑，以及最大限度的复制现实的商业逻辑。

◆ 消耗评估与分析

由于CC是一个公链系统，节点会选择那些收益与占用的计算资源比值比较高的应用优先运行。所以任何一段代码在执行以前，都会对其所可能占用的CPU，内存，存储资源以及带宽做一个评估。评估的其中一个标准，可以是但不限于：（1）估算对基本运算符号的使用次数即运算复杂度（2）所申明的变量个数（3）代码长度。

值得注意的是，这里并没有包含由外部条件触发的逻辑。以太坊为了使其消耗评估变得可行，将以太坊的脚本语言鼓励起来。CSSL保留了合约的外部输入，但会要求这个外部输入必须是CC内的一个参与者，并要求该参与者能对这些“外部输入”的内容和范围作出一个大致的评估，以帮助底层矿工评估消耗。底层矿工需要对消耗的基本单位进行定价，然后通过全网算力和燃木的分布情况，估算全网内基本单位的平均价格。这样开发者就可以根据平均价格和消耗评估，分配奖励以激励矿工稳定的支持该应用。

消耗评估的另一个用处就是可以防止CC内部的使用者，因为“无心”之失编写出错误的程序导致死循环，耗费CC网络的基本资源。开发者需要先支付，商业逻辑才能获得执行。

◆自动执行

商业逻辑一旦写入网络，除非逻辑各方同意终止逻辑，否则就会根据自动执行，这是去中心化公链系统的优势。举例来说，一份信托契约，受益人是信托人的子女，将信托之财产赋予十八岁后的子女。这条逻辑在CC上写入与执行就非常容易了，主体部分只要一个条件语句就可以实现。

◆法律合规的兼容性

尼克萨博在1990年首先提出智能合约的概念，但是由于缺少可信的执行环境，并未有实际应用[21]。CC网络是可信的去中心化平台，天然就是适合使用智能合约的平台。Mary Juetten [22] 在一份采访报道中，提出了Contract as Code的概念，也就是“合约即代码”。我们在设计和开发代码的时候将会考虑到如何使代码变得在法律意义上更加可读，增加 “Intention to be legally bound”这一个申明语句。

由于法律合规条纹非常复杂，在具体设计代码的时候，我们会考虑考虑到当地具体的法律法规，以设计不同的表达形式，力求代码本身就具备一定的法律可读性。我们知道这是一个非常有挑战性的设想，但作为区块链世界与真实法律世界的桥梁，法律合规的兼容性是很有意义的一个方向。

我们在设计兼容性的同时，也预想到了可能会发生的困难以及当地法规可能出现的变动。CC特别增加了一个法律合规编译层，对代码做独立的解读，并生成合规的文档存储在链上。

◆简单易用

我们会开发相应的工具包，方便开发者做调试和除虫等工作。同时，开发者也可以在函数库与类的商店里面找到快速开发模版，加速开发进度。

◆隐私保护

与一般的公链不同，CC除了可以把商业逻辑加密过后写入公链，还可以只将通过哈希函数得到的摘要“见证”写入公链。使用后者在隐私保护和资源占用方面会更占优势。

3.2.2 CC应用“商店”

CC与苹果的应用商店类似，开发出来的应用首先要经过监管者应用审查，然后才能上架。监管者同时会评估该产品的质量，效率与用户评价，决定是否继续保留上架。同样与应用商店类似，CC的商店也会有排行榜的机制。

值得一提的是，法律合规的监管者，能够以CC的监管者身份审查任意应用是否能上架，并检查该应用与目标用户是否符合当前监管的要求。

3.3 CC虚拟机 (CC Virtual Machine)

虚拟机的概念是指来自底层公链节点以及相应节点的计算机资源的抽象描述。虚拟机通过占用其节点资源来支持和运行CC商业逻辑，这就包括代码，应用，清算和数据处理等等。虚拟机也会相应获得一定的回报。

进一步理解什么是虚拟机，因为CC网络上每个区块时间内，各个网络节点竞争上岗。所以哪个节点作为那个时间段的主机，是不确定的，称之为虚拟机。

虚拟机根据应用所需要消耗的计算机资源和回报来分配虚拟机本身拥有的资源。在这里我们借用EOS.io[18]对计算机资源的定义：CPU，硬盘空间，内存空间以及网络带宽。

3.4 商业逻辑应用举例

我们来举几个如何使用CC开发金融应用的例子。以下编程语言采用以太坊的solidity作为格式模版，仅仅作为逻辑上的示范，不代表以后CSSL会使用这种表达形式。

3.4.1 发债的代码解析 (Debt Originations)

发债过程可以比较复杂，但核心内容就是定义债券的数量，债券凭证的转移，以及到期以后债券凭证的销毁。如果在CC上发一个债，如果只考虑最简单的一些核心逻辑的话，可以编写以下的代码：

```
1 pragma cssl ^0.0.1; //第一句话申明语言版本号

//2行到22行，定义了一张债券的合约，命名为MyBond
//3行到7行，定义了一些公用变量如债券名字，总产量，债券转移的函数以及当合约结束的时候，债券销毁函数
```

```
2 contract MyBond {

3     string public standard = 'Bond 0.1';
4     uint256 public totalSupply;
5     mapping (address => uint256) public balanceOf;

6     event Transfer(address indexed from, address indexed to, uint256 value);
7     event Burn(address indexed from, uint256 value);

// 8行到11行，定义了初始债券初始数量

8     function MyBond(
9         uint256 initialSupply
      ) {
10          balanceOf[msg.sender] = initialSupply;
11        totalSupply = initialSupply;
    }

// 12行到17行，定义了初始债券转移函数的具体内容，如何从一个地址减去，如何在另一个地址上加上一定的数量。

12     function transfer(address _to, uint256 _value) {
13         if (balanceOf[msg.sender] < _value) throw;
14         if (balanceOf[_to] + _value < balanceOf[_to]) throw;
15         balanceOf[msg.sender] -= _value;
16         balanceOf[_to] += _value;
17         Transfer(msg.sender, _to, _value);
    }

// 18行到22行定义了当债券到期了以后，如何销毁债券。
18     function burnFrom(address _from, uint256 _value) returns ( bool success ) {
19         if (balanceOf[_from] < _value) throw;
20          balanceOf[_from] -= _value;
21         totalSupply -= _value;
22          return true;
    }
}
```

3.4.2 信用证与贸易融资 (Letter of Credit and Trade Finance)

这里举一个做信用证的代码编写思路做例子。我们可以了解信用证的代码编写方式和发债的部分近似。首先：（1）信用证有一个银行或者其他金融机构的三方保证，这就需要一个数字签名。我们可以通过调用一个数字签名的模版或者函数，让该金融机构在上面做一次数字签名。（2）然后我们需要给这个信用证做一个事件驱动，将该信用证与某一个具体贸易关联起来，也就有了资产关联。（3）用户在拿这张信用证做贴现的时候，其他金融机构可以很简单的验证这个数字签名是否来自做三方保证的金融机构。

04 法律合规编译层 (Legal Compliance and Regulation Compiler)

法律合规编译层是我们CCCC的一个特色层级，专门为CC商业逻辑编译成为附和法律合规条例的合同。事实上，在金融业里面，类似的应用和思路并不鲜见。譬如在行业内进行衍生品交易的时候，前线业务人员通常先和客户确认一个简单条款（termsheets），内容通常都是列举一系列变量和简单的计算逻辑。确认方式则有多种，可以使用电话，email甚至传真。前线条款确认和，后台工作人员会根据前线条款生成语义明确的法律合同，也叫后台合同(backoffice confirmations)。

我们借鉴这种由简单前线条款生成后台合同的方式来编译CC商业逻辑，使其成为法律合规。我们目前的想法是使用两套系统来完成这个编译工作，这两种系统都来自于人工智能的相对来说较成熟的技术。

4.1 人工智能专家系统 (A.I. Experts System)

第一种方式就是俗称的专家系统[23]，我们借助传统衍生品交易方式的，产生大量的代码模版和法律文书模版。这两类模版通过一个映射矩阵，将各自的变量映射到相对应的模版中去。开发人员在编写代码的时候，如果选择了使用法律合规编译的专家系统所带模版的时候，在定义变量的时候就会受限于模版的定义，包括但不限于大小的范围，获取的方式等等。这样做的好处是编译的质量会较高，也具有很好的法律合规的可读性。缺点是需要积累大量的代码模版，而且模版的灵活性也不够。

4.2 深度学习与人工神经网络 （Deep Learning and Neural Network）

第二种方式使用深度学习和人工神经网络来学习大量样本，自动从代码生成具有法律语义意义的合同文本文档。这样做的好处是很明显，可以自动适应各种商业逻辑和代码。缺点也是比较明显，存在不能生成语义清晰的法律文档的可能性。

我们有可能使用以上两种方式的一种或者两种混合的方式来生编译法律合规文档。我们意识到目前人工智能关于语义识别和生成的挑战，但如果结合了我们特别设计的CSSL语言，有可能可以生成语义变化“连续”且“平滑”的文档。我们认为这将会是一个非常值得尝试的方向。

05 数据存储层 (Data Storage Layer)

在CCCC里面，数据存储层所处理的数据有几种不同的类型：包括使用者群体数据，各类代码类数据，LCR编译后的数据，各类清算数据，以及Celes Chain的代币账本数据。

使用者群体数据是指所有使用者的KYC数据以及各类身份验证以及签名。这类关键性的隐私数据一般会由应用开发者通过加密本地或者使用CC的公用数据库存储，然后放一个哈希“见证”在公链上。

各类代码类数据包括代码本身，程序，合约，函数，应用，模版以及商业逻辑等。这部分数据既可以加密储存在本地或者公用数据库内，也可以直接存放加密数据在公链上，视代码本身的价值而定。譬如说，像模版之类的公开信息，就可以放在公用数据库内，就在公链上留下“见证”就可以了。

LCR编译后的数据一般都是附和法律语义的合同文件以及相关数字签名。这一部分数据与代码类数据类似，也是根据其价值选择在公用数据库内存储或者在公链上留有“见证”。

各类清算数据和Celes Chain的代币账本数据，除非有特别的安排，一般来说是公开存储在公链上，以符合去中心化的要求。

什么是“见证”？一个“见证”就是对一段信息使用哈希函数做个哈希“摘要”。由于哈希函数是单向函数，且不可逆[24]。这段“摘要”就可以作为一个见证放在公开的公链上，任何人都可以验证手上的原文的真伪，但无法在没有原文的情况下通过“见证”还原原文。

06 底层共识模块层

本协议层使用了时分多重证明协议[1]。时分多重证明协议使用了PoW挖燃木，并异步得使用燃木进行PoB方式产生代币和产生区块的方式达成共识。这使CC可以在去中心化与效率之间取得一个较好的平衡，给予底层矿工足够的动力投资稳定的硬件资源的支持，使基于这种公链共识协议的CC的效率会比较高。

PoW，即Proof of Work又叫做工作证明[25]，是一种通过计算机算力投票来表决网络领导地位的一种共识算法。其投票方法是通过计算哈希函数求解，来证明自己的计算机算力比网络内的其他的节点来的快。

PoB，即Proof of Burn又叫做焚烧证明[26]，是一种通过焚烧自己手中的代币来表决谁对网络领导地位的承诺。焚烧代币的数量越多，能获得网络领导地位的概率就越大。

6.1 时分多重证明协议的效率

这里我们引用TDMPC的论文[1]，简要的描述一下时分多重证明协议的效率比PoW高的证明过程。我们首先定义任意一个节点的算力如下：

M=m_1+m_2 ~(1)

m_1~Node's total resources invested in mining
m_2~Node's total resources invested in system to run applications
M~Node's total resources invested in mining and system to run applications

然后我们通过证明代换，我们可以得到任意一个节点的挖矿收益率如下：

Return = −b/(P·C)·m_1^2+((M·b)/(P·C)+a/P)·m_1

P~total mining resources within the system
C~total applications
〖m_1〗_ /P~expected mining hit ratio
a~fixed rewards from mining
b~ rewards from running applications

由于收益率是一个以m1为自变量的负二次曲线，所以一定有一个m1使这个收益率最大化。我们可以得出一个结论是当任意一个节点将所有的资源掉配去挖矿而不是去增加计算机硬件资源，可以使收益最大化。因此在一个普通PoW系统里面，由于矿机没有动力去投资计算机硬件资源，是不能够稳定的运行大规模应用的。
相反如果去掉了挖矿的固定收益，那么任意矿工必须配置1:1的挖矿/计算机硬件资源比例以达到最大的挖矿收益。

6.2 CC底层共识模块

CC底层共识使用时分多重证明协议。如图下所示，使用速度较慢的工作证明(PoW)，通过算力挖掘燃木wood。然后使用速度较快的焚烧证明(PoB)，通过焚烧燃木wood来挖掘CC代币作为奖励以及产生区块。

从章节6.1以及TDMPC的理论我们可以知道，当PoW+焚烧证明获得的代币奖励全部都来自其运行的应用时候，区块链网络可以稳定的运行大规模应用。这也就是在我们CC里面，我们设定矿工不能获得固定收益的原因。

如果CC价值变高，外部算力可以通过PoW来间接获得代币作为奖励和分享价值，体现了真正的去中心化。

由于挖燃木的速度和挖代币的速度允许不一样，所以可以使用PoB来高速挖掘代币并产生区块，也就是说基于这个PoB的平台可以承接大规模应用且有效率。

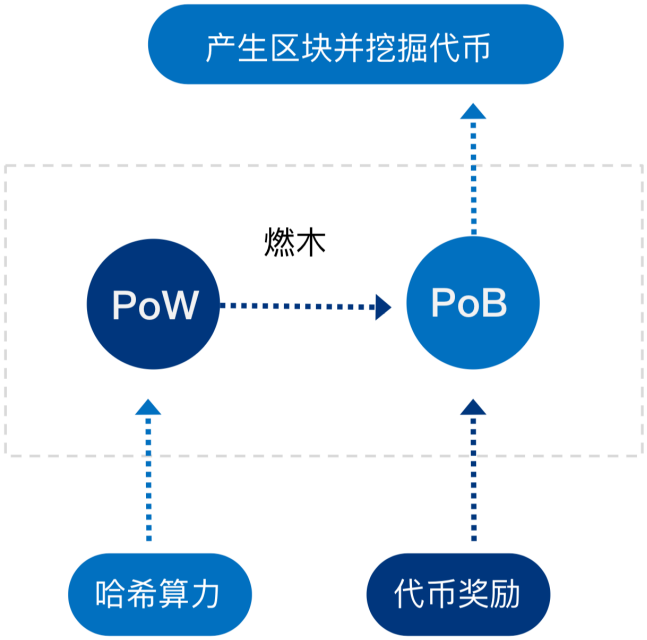

图3 – CC底层共识模块

07 代币产品的使用，循环和生态系统

7.1 代币产品的使用

首先，我们使用的代币，是基于以太坊ERC20合约标准建立的一种产品。用户拥有这种产品的时候可以选择以下三种策略（如图所示）

使用策略1：购买CC的服务和应用
使用策略2：通过代币购买外部哈希算力（如有）再间接获得代币
使用策略3：通过购买CC的服务获得盈利，再通过法币购买外部哈希算力（如有），再间接获得代币。

用户通过自行选择对自己最有利策略使用代币，允许通过套利来获利，并通过这种套利方法来增值

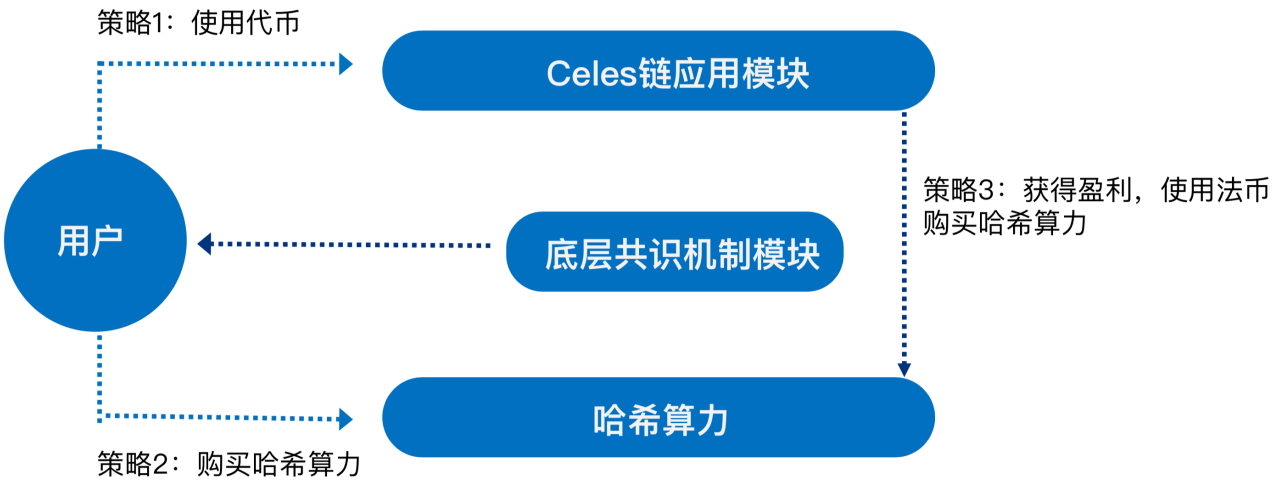

图4–代币产品使用路径和策略

7.2 代币产品的循环和生态系统

首先，哈希算力输入底层共识机制并出块支持CC的多种应用。如图所示，出块奖励代币投入应用模块中使用并进入代币收集。CC系统分配收集来的代币作为奖励注入底层共识机制，维护CC网络的安全。早期私募众筹的代币作为起始代币最先进入应用系统，节点使用CC服务后，并归入代币收集。

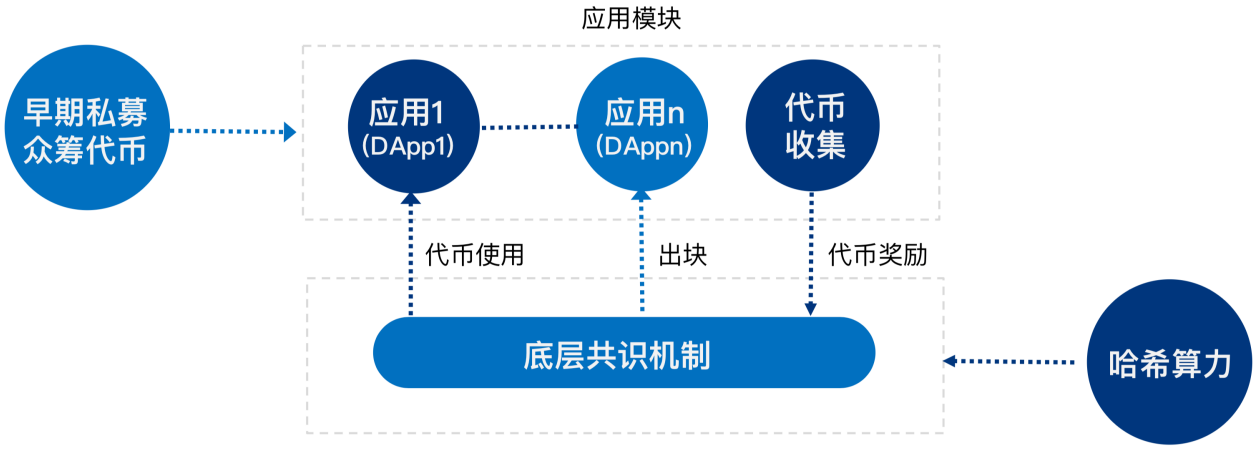

图5 – 代币产品的循环和生态系统

08 结论

Celes Chain 使用去中心化底层共识和区块链技术支持各类金融应用，以打造一个区块链上的“华尔街”作为目标，解决了目前金融市场存在的监管成本高和“信心”问题。同时我们使用时分多重证明共识协议[1]，这是一种在“去中心化”与“效率”之间取得较好平衡的共识。这套共识使我们的CC技术平台可以使我们的底层矿机有足够的动力维持一定水平的硬件资源稳定的支持的CC平台上的各项应用。

将会新开发CC特有的的脚本语言，是一个强大的，直观，简洁且针对各项金融服务而开发的脚本语言。CC平台上的各类参与者可以使用强大的CCSL组合自己的金融和商业逻辑以及开发相关应用。同时，监管机构可以根据法律规定，对各类应用，数据和用户行为进行监管。此外，我们还会把部分协议，应用，代码和数据编译成为符合先行法律规范的文本，方便进一步合规和监管。最后，这些应用的数据，合约以及相关信息都将会以加密后的形式作为“记录”放入底层共识“公链”内，以保证这些“记录”不会被篡改或非法窃取。

10 引文列表

【1】Yeung (2018), Time Division Multiple Proofs Consensus Protocols, working paper
【2】Nick Szabo, Smart Contracts, 1994, http://www.virtualschool.edu/mon/Economics/SmartContracts.html.
【3】John C. Bogle (2008), Commentary: Why we lost faith in Wall Street -- and what to do, CNN, http://www.cnn.com/2008/POLITICS/12/01/bogle.investors/index.html
【4】Ivana Kottasova (2016), $9 trillion and counting: How central banks are still flooding the world with money, CNN, http://money.cnn.com/2016/09/08/news/economy/central-banks-printed-nine-trillion/index.html
【5】Governor Daniel K. Tarullo (2016), Financial Regulation Since the Crisis, https://www.federalreserve.gov/newsevents/speech/tarullo20161202a.htm
【6】Philip Stafford (2018), Mifid II and dark pools: what are regulators up to?, https://www.ft.com/content/491bbfa8-f3ba-11e7-8715-e94187b3017e
【7】Volcker Rule, https://en.wikipedia.org/wiki/Volcker_Rule, wikipedia as of Jan 17,2018
【8】Peter Pham (2018), Why are banks too big to fail?, https://www.forbes.com/sites/peterpham/2018/01/15/why-are-banks-too-big-to-fail/
【9】Satoshi (2008) , Bitcoin: A Peer-to-Peer Electronic Cash System
【10】Leslie Lamport (1982), The Byzantine Generals Problem, ACM Transactions on Programming Languages and Systems
【11】Evelyn Cheng (2018), Second-largest cryptocurrency ripple may have run ahead of itself, CNBC, https://www.cnbc.com/2018/01/05/second-largest-cryptocurrency-ripple-may-have-run-ahead-of-itself.html
【12】Alex Hern (2018), Bitcoin's energy usage is huge — we can't afford to ignore it, theguardian, https://www.theguardian.com/technology/2018/jan/17/bitcoin-electricity-usage-huge-climate-cryptocurrency
【13】https://www.reddit.com/r/ethtrader/comments/65nc7d/what_dappssoftware_is_being_built_on_bitcoin/, taken at Jan 17, 2018
【14】http://www.businessinsider.com/the-worlds-2nd-largest-crypto-currency-ethe

reum-had-an-even-bigger-price-surge-than-bitcoin-2017-5, taken at Jan 17, 2018
【15】https://www.ethereum.org/
【16】Banks tap Ethereum smart contracts for MiFID II compliance, https://www.finextra.com/newsarticle/31465/banks-tap-ethereum-smart-contracts-for-mifid-ii-compliance/blockchain
【17】
http://www.ibtimes.co.uk/ubs-barclays-credit-suisse-thomson-reuters-explore-ethereum-based-mifid-ii-solution-1651014, taken at Jan 17, 2018
【18】
https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md, taken at Jan 17, 2018
【19】
https://www.reddit.com/r/ethereum/comments/6s6eh8/delegated_proofofstake_vs_proofofstake_also_im/, taken at Jan 17, 2018
【20】Securities and Exchange Commission, July 25, 2017, Report of Investigation Pursuant to Section 21(a) of the Securities Exchange Act of 1934: The DAO
【21】Josh Stark (2016),https://www.coindesk.com/blockchain-smarts-contracts-real-world-law/
【22】Mary Juetten (2017), https://www.forbes.com/sites/maryjuetten/2017/08/16/legal-technology-and-smart-contracts-contract-as-code-part-i/#15f7dd6a8b24
【23】赵南元 (2002), 认知科学揭秘（第二版）
【24】http://www.aspencrypt.com/crypto101_hash.html, taken at Jan 17, 2018
【25】https://en.wikipedia.org/wiki/Proof-of-work_system, take at Jan 17, 2018
【26】https://en.bitcoin.it/wiki/Proof_of_burn, taken at Jan 17, 2018