软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.000000]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

CoderChain: 一个基于区块链的开发者社区*

林 毓植[1], 林 毅[1], 吴 垠熠[1], 黄 家承[1], 梁 华超[1], 郭 旭[1], 郭 丽敏[1], 戚 宗城[1]

[1](广东工业大学 计算机学院,广东 广州 510000)

通讯作者: 林毓植, E-mail: leslieme668@outlook.com

摘 要: 本文提出一个基于区块链的开发者社区，称为 CoderChain (码农链) ，供广大软件开发人员共享，评估和学习代码以及其他代码或者软件相关的知识，旨在为程序员，开发人员等计算机从事者建立一个公开透明，互相促进的社区。通过应用区块链的思想和技术，CoderChain 从如下三个主要方面构建开发者社区：基于点对点的区块链网络，完全去中心化存储，多维度的价值模型评估。CoderChain 继承传统开源平台的优势和思想，在此基础上为社区的代码建立精准的多维度价值评估模型，建立公认可信的开发者综合素质标识。在本文中，我们将仔细阐述我们的研究工作（包括 CoderChain 的底层架构和上层应用的设计），最后，我们对 CoderChain 未来功能机制的完善进行了展望.

关键词: 开源社区;区块链;去中心化存储;代码评审;软件开发

CoderChain: a developers' community based on the blockchain

LIN Yu-Zhi[1], LIN Yi[1], WU Yin-Yi[1], HUANG Jia-Cheng[1], LIANG Hua-Chao[1], GUO Xu[1], Guo Li-Min[1], QI Zong-Cheng[1]

[1](School of Computer Sci. & Tech., Guangdong Univ. of Technology, Guangzhou, China)

Abstract: This paper figures out a developers' commuity based on the blockchain, Which called CoderChain. CoderChain provide a way for the developers especially for the comgrammer to sharp, evaluate amd learn code or some software skill, Which establish a open and transparent, mutually reinforcing autonomous community. With the help of blockchain thought and techonology, CoderChain establish a developers' commuity from three main aspects as followed, Point-to-point blockchain network, fully decentralized storage, multi-dimensional value model evaluation. CoderChain inherits the advantages and ideas of the traditional open source platform. On this basis, it establishes a precise multi-dimensional value evaluation model for the community code, and establishes a recognized and credible developer's comprehensive quality identification.The paper will clearly explain the reasearch including the underlying architecture and the design of the upper application of CoderChain. Finally, we look forward to the improvement of the future functional mechanism of CoderChain.

Key words: Open source community; Blockchain; Decentralized storage; Code review; Software development

开源模式是一种分布式的软件，代码开发模式，鼓励参与的开发者使用或者修改某个开源程序的原始设计，达到开放协作，共同维护的目的。随着开源模式的普及和发展，世界上成立了许多基于开源模式的源代码社区组织，吸引了全球范围内的开发者的参与，是分布在世界各地的开发者交流合作的重要途径，有助于开发者贡献和分享自身知识，并且可以在社区中学习他人分享的知识，为开发者提供了自由学习交流的空间。开源社区极大的推动了软件和程序的技术创新，在开源软件或项目的发展过程中起了至关重要的作用，重塑了开发者的开发模式。

自从开源社区的大力推广和应用之后，随之而来的问题隐患也逐渐浮现，当今的开源社区主要面临三个问题：开源社区以中心化模式建设，开发者劳动成果保障，开源社区中代码价值评估机制缺失。第一：开源社区一直以中心化服务商为广大开发者提供开源服务，但是从建设效率性和数据安全性的层面上看，中心化的服务存在许多隐患，比如访问效率低下，网络阻塞，单点故障隐患，数据可能会被篡改等重要威胁。第二：长期以来，开发者在持续贡献和创新的同时，开发者的劳动成果难以得到有效保障，鉴于价值载体（源代码）和开源模式的特殊性，如何保障开源社区中开发者的劳动成果和知识产权成为一个悬而未决的难题。开源社区出现大量代码滥用，不加引用等现象，但基于传统中心化架构却无法从根源上解决此类问题。第三：开源社区的发展由开发者个人兴趣推动，如果参与的开发者活跃度或者积极性较低，那么会让整个开源社区的环境氛围陷入困境。社区的发展和进取是靠社区中每一个开发者的不断努力和贡献，开发者个人需要有自主的进取心和动力，反之，社区也应给开发者提供开展空间，激励和促进开发者提高自身水平，发挥潜在价值，并且目前的开源社区缺乏有效的代码价值评估机制。为了解决当今开源社区中存在的问题，我们彻底改变开发者社区的底层基础架构，对上层应用做出重大变革。

CoderChain 是一个基于区块链的开发者社区，利用区块链作为社区的底层架构，以去中心化的思想建设开发者社区，使开发者社区真正实现开发者自治，社区中的决策事务都与全体开发者的决策相关联。CoderChain 采取混合链双链共行机制的底层基础架构，相对于传统开源社区的中心化存储方案，CoderChain 采用基于 IPFS 的去中心化分布式数据存储，将所有数据加密并且分块存储在世界各地的空闲硬盘上，从空间上充分有效地利用空闲剩余的存储空间，从时间上提高了访问数据的效率和降低网络阻塞的概率。为了保障每一位开发者的劳动成果，社区会对每一份原创代码进行版权标识，生成原创作者和代码特征信息捆绑性证书，有效地支持开发者维护个人知识产权。相对于当今开源社区中激励模式的缺乏，本社区实施价值贡献激励机制，奖励社区中实际价值高的代码或者项目的原创开发者。社区从代码静态分析和代码测试准确评估代码或项目的实际价值和潜在价值。在代码价值评估的基础上，为开发者建立多维度价值度量模型，进而建立社区公认，可信，真实的开发者综合素质标识。

本文第 1 节简要叙述区块链的主要技术概念和思想。第 2 节详细论述开发者社区的底层架构建设，并介绍主要算法的实现。第 3 节对社区去中心化存储方案的实现进行总结。第 4 节详细讨论社区中代码项目价值评估，开发者多维度价值度量的实现。最后总结全文，并且提出对 CoderChain 未来技术的展望。

1 区块链技术和思想

区块链[1][4][5]是由一系列加密数据（简称区块）链接而成，每一个数据区块都存储着时间戳，上一个区块的哈希值和一些交易数据等。区块链可以比喻为一本记满所有交易记录的账本，每个参与点对点区块链网络的节点都拥有此账本，通过共识机制的保障，所有网络中的节点都可以认定这一本账本是真实可靠，不受篡改的。

区块链作为交易双方的信任基础，使得交易可以直接由双方发起而不需要第三方中介的信任担保。区块链不

仅充分解决了交易场景中存在的信任问题，其功能特性也广泛适用于其他应用场景，比如不可篡改性，可溯源性，去中心化等。

区块链的基础是点对点式的去中心化网络，因此网络中所有节点都拥有完整记录的区块链加密账本。共识机制保证每个节点的区块链账本是否一致和真实，比如经典的比特币[2]中的共识算法 PoW（工作量证明机制）通过计算机的算力保证共同区块链的真实性，并且通过激励机制维持节点的正常运营。接下来将介绍区块链三个关键技术概念和思想：点对点网络，共识机制，Token 激励机制。

1.1 点对点网络

点对点网络（Peer-to-Peer），也称对等式网络，是去中心化的，依靠全体网络节点交互的互联网网络模型，网络中不存在任何中心服务端、中心化的服务。点对点网络的节点之间交互连接、协同，每个节点即作为客户端（Client）使用网络中其他节点所提供的信息和服务的同时也充当服务端（Server），对其他节点提供服务。点对点网络不仅仅去除了中心化服务带来的潜在风险和隐患，而且还进一步提高信息传输的效率，同时在安全层面上，点对点网络极大增加了攻击者的攻击成本，攻击者需要攻陷网络的大部分正常节点才能够发动一次成功的攻击。

1.2 共识机制

共识机制[3]是区块链技术的重要组成部分，旨在使网络中所有诚实节点拥有一致的区块链视图，是解决分布式点对点网络中数据一致性的有效方法，同时满足两个性质：一致性与有效性。设计良好的共识机制可以提高系统的交易效率，节省系统资源，维持并促进区块链网络的良好发展。现有较为知名的区块链共识机制有工作量证明（PoW），权益证明（PoS）和股权授权证明（DPoS）等。

1.3 Token激励

比特币使用的共识机制是工作量证明机制（PoS），通过利用全球大量的计算机算力保证区块链网络的正常运行，其中算力的来源便是比特币代币的激励和驱动。正是因为价值高昂的比特币代币激励，使得区块链网络中不断有新节点的产生，并且有着越来越多的算力去运营维持自身的稳定发展。

并不是所有区块链共识机制都需要代币激励，也不是所有区块链应用中都需要代币的存在，代币激励是一种激励手段，目的是为了使网络中的节点从利益的角度上更加趋向于做诚实节点，让节点在为网络做贡献的同时给予其应得的奖励。在 CoderChain 中，我们借鉴了代币激励的思想，通过社区中的代币来激励广大开发者创造价值，贡献价值。

2 社区底层架构

为了减少开发者的开发任务和重复造轮子的现象(re-invention of the wheel)，CoderChain 提出了基于公有链和联盟链双链共行机制。

2.1 系统整体架构

CoderChain 主要分为三部分，分别为 CoderChain 社区、CoderChain 社区基础以及 CoderChain 网络和货币基础，整体架构如下图所示：

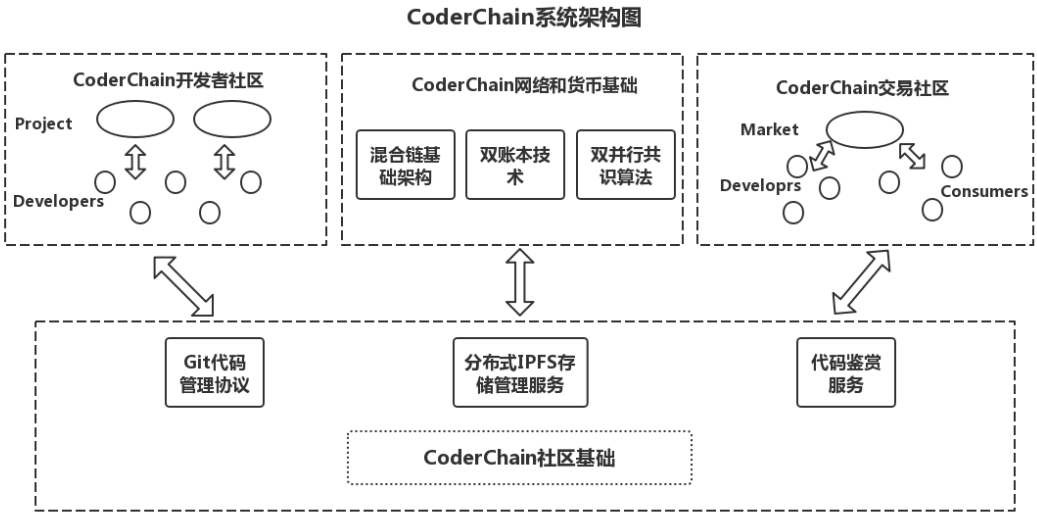

Fig 1. CoderChain 系统整体架构概念图

(1) CoderChain 社区：包括 CoderChain 开发者社区和 CoderChain 交易社区；其中 CoderChain 开发者社区由开发者组成，借助 Gti 系统进行代码上传，并且使用 IPFS 系统进行代码信息和开发者信息的分布式存储；而交易社区由开发者和交易者组成，使用来自开发者社区的代码和开发者信息，通过代码鉴别系统进行服务。

(2) CoderChain 社区基础：主要为 CoderChain 社区提供关键服务，包括 Git 代码管理协议、分布式 IPFS 存储管理服务、代码鉴赏服务等。

(3) CoderChain 网络和货币基础：主要基于区块链技术，为 CoderChain 系统提供运行基础的混合链基础架构、双账本技术、双并行共识算法。

2.2 CoderChain 区块链基础网络

2.2.1 混合链基础架构

CoderChain 采用分片链机制，将公有链和联盟链结合起来。由于 CoderChain 面对的代码的不确定性和逐渐偏向实体性，需要将使用混合式分片链机制，并通过哈希索引技术实现链之间的通信，保证同一数据的准确性和平台的权威性。

CoderChain 平台的开发者社区是由多个子社区所构成的，从应用场景的角度来讲，单个子社区既可以对应着某一个公司的内部开发者团体，也可以对应某一个班级的学生团体。每一个子社区都可视为一个联盟网络，该网络上发生的所有交易或者代码提交都将被记录在其所对应的联盟链上，所有联盟链上的交易和代码提交集合将会全部整合到公有链上，区块的结构如下图所示：

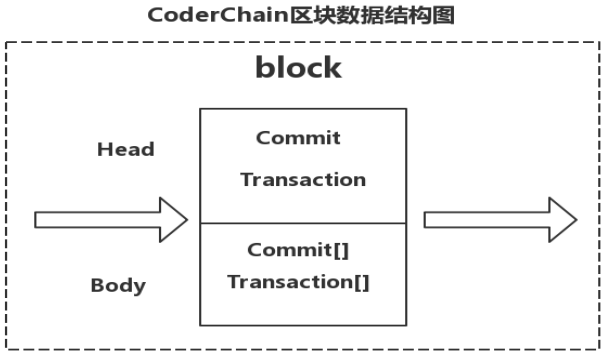

图 2. CoderChain 数据结构概念图

在 CoderChain 中，无论是被用于交易的代码还是被用于提交的代码，公有链和联盟链只记录其附属信息。对于每份代码而言，它的附属信息都记录于一个对应的结构体中，而这一结构体将作为交易对象和代码提交对象的成员保存于区块中，代码的附属信息主要包括了版本管理和账户管理两个维度：

1. 账户管理：每一个账户，包括个人账户和团体账户都会有一个 CoderID，它被用于标识代码的唯一所有者。对于团体账户，CoderChain 还设立了权限管理（比如只读权限、可写权限等）。

2. 版本管理：

(1) 代码版本：记录着代码的当前版本信息。

(2) 代码哈希摘要：指向一个 CodeID，在 CodeID 中保存着代码的哈希值。

(3) 代码来源：代码所属的子社区。

(4) 代码创建时间：当前版本的代码提交时的时间戳。

(5) 代码使用的合同：标注代码采用了何种开源协议。

介于代码所占存储空间的不可确定性，CoderChain 公有网络上的节点是不存储代码本体的，代码本体将只存储于其所属联盟网络的节点上，并且用户可以通过某一代码的哈希摘要定位到该代码，这样做既有效地缓解了公网节点的存储压力，也不影响代码的搜寻效率。

2.2.2 CoderChain 双账本技术

CoderChain 的每个区块头部中都记录着两个默克尔根，分别对应交易数据和代码提交数据所形成的默克尔树的根节点的值，一笔交易由一个 Transaction 结构体表示，而一笔代码提交则由一个 Commit 结构体表示。CoderChain 将这两种类型数据的存储和管理对应两个不同的账本，分别是交易账本和代码提交账本。

(1) 交易账本主要用于记录代码交易或者代码引用过程的信息，包括交易的发起者信息和被交易代码的附属信息。

(2) 代码提交账本主要用于记录代码提交过程的相关信息，主要包括当前代码的附属信息

2.2.3 默克尔树在区块链中的应用

默克尔树是一种树形数据结构，一般为二叉树，在默克尔树中，每个叶节点均以数据块的哈希值作为标签，除叶子节点以外的节点以其子节点标签的哈希值作为标签。

在 CoderChain 中，默克尔树被用于组织和管理相关的数据结构。一个区块中所有 Transaction 和所有 Commit 会分别被构造成对应的默克尔树，并且这两个默克尔树的根节点的值也将会记录在区块头中。之所以采用二叉树类型的默克尔树对上述内容进行组织和管理，是因为其在数据处理上具有一定的高效性和安全性。

对于一个有着 N 个叶子节点的二叉树类型的默克尔树而言，若希望查找一个区块中是否存在着某一笔交

易，整个过程只需要 O(log N)的时间，而传统的哈希树算法的时间复杂度为 O(N)。同时，默克尔树的根节点具有唯一性，因此两组不同的数据块所形成的默克尔树理论上是不会相同的，任意一个叶节点的变动也会导致当前的默克尔树完全发生变动，这也意味着由一组特定的 Transaction 或 Commit 所归纳出的默克尔树是可以被唯一确定的。

CoderChain 是一个有着大量轻节点的区块链网络，在存储资源方面，轻节点有着一定的劣势，而默克尔树的应用使得轻节点在验证某笔 Transaction 或 Commit 是否已被记录的过程中节省了同步所有的区块资源的行为操作，只需要同步全网中最长链的所有区块头和该笔 Transaction 或 Commit 所处区块的内容，在很大程度上降低了 CoderChain 对轻节点的硬件各方面的需求，由于区块头的存储空间占有量很小，所以验证过程中所涉及的哈希计算操作即使目前的底端硬件设备也是可以支持和稳定运行的。

2.2.4 混合链的基础算法

DAG（有向无环图）算法由集合的顶点和有向边构成，每条边连接不同的节点，顶点之间不存在循环返回的可能。DAG 算法能够最快找到节点间的最短路径，大幅度提高交易确认效率和并发性能。从数据结构体系来看，DAG 模式是一种典型的谣言传播算法，其核心机制即在于异步通讯。DAG 区块链技术不但可以支持高并发、结合双层共识机制，并且在安全层面上防止双花攻击的发生。

2.2.5 双共识机制

由于 CoderChain 的基础架构采用了混合链技术，所以在 CoderChain 中，公有链与联盟链也会采用不同的共识机制。

(1) 在联盟链中，CoderChain 平台使用实用拜占庭容错算法 PBFT（Practical Byzantine Fault Tolerance）来维持 Transaction 和 Commit 的合法性。PBFT 算法不仅支持容错故障节点，并且支持容错作恶节点，充分解决联盟链的集群环境需要考虑故障节点和作恶节点的存在问题。

(2) 在公有链中，CoderChain 采用了 DPOS（Delegated Proof of Stake）共识算法。在应用 DPOS 算法的区块链网络中，网络的正常运转依赖于受托人（Delegates），共识算法保证受托人之间关系的完全平等性，受托人由全体平台开发者选举产生，同时每个受托人都需要提供一台服务器加入区块链网络，并且保证该节点的正常运行，经过一轮选举产生的受托人将轮流参与区块的打包工作，维持当前区块链网络整体的高效运转和上层应用的稳定运营。DPOS 算法不仅在一定程度上防止作恶行为的发生，还在整个区块链网络有效运行的情况下不多大地消耗算力资源。

3 去中心化的分布式存储

集中式存储模式将所有数据存储在单个大型主机系统或者多主机集群系统，由数据管理系统集中进行管理，所有的数据行为操作都由中央系统完成。集中式存储模式存在单点故障，访问性能瓶颈、存储可靠性和安全性低等问题。如何在保障开发者社区中数据的安全的同时改进并提高各类数据访问和传输操作的效率是一个富有挑战性的问题。随着计算机和网络技术的高速发展，点对点式网络文件存储模式逐渐显现优势，近年来[6]，一种基于区块链技术的去中心化分布式存储网络[7][8]被提出-IPFS。

IPFS 星际文件系统是一个面向全球，旨在创建持久且分布式存储和共享文件的网络传输协议，通过内容寻址改变 HTTP 协议的服务端客户端的信息传输模式，是一种内容可寻址的对等超媒体分发协议。IPFS 从根本上改变了资源查找方式，相对于传统的模式中浏览器查找文件必须遵守 HTTP 协议，比如首先查找目标服务器的 IP 地址，随后向目标服务器索要文件，这种体系下文件资源的位置取决于服务器管理者，用户只能寄希望于文件没有被移动或者服务器没有被关闭，现实场景存在潜在故障隐患和被恶意攻击的风险。IPFS 根据文件内容查找资源，与中心服务器的位置或者文件的名字和路径无关。IPFS 网络是不固定的、细粒度的网络，

并由全体节点参与构成分布式网络，适应现代内容分发网络（CDM）的要求，提高共享各类数据，包括图像、视频流、分布式数据库、操作系统方面上的效率。

鉴于 IPFS 的技术优势和思路，CoderChain 的存储模式已 IPFS 为基础架构，并在其基础上构建一个完全去中心化分布式存储系统。

该系统的基础框架主要由下列三部分组成：

(1) 文件版本管理

CoderChain 提供文件的历史版本控制器，让多节点使用和保存处于不同版本的文件项目，基于版本管理模式下，开发者可以选择回溯历史版本的文件数据，使整个项目活动的生命周期是可见和可控制的。

(2) 存储空间的利用

随着计算机存储技术的进步和发展，硬件设备的存储空间成增长趋势，但是存储空间闲置的现象普遍存在，如果可以通过某种途径充分有效地利用闲置的存储空间的同时，不影响存储硬件设备本身的正常运行，那么将极大地节省数据存储资源的成本。CoderChain 通过分布式存储网络串联所有参与网络的硬件设备，每个运行 CoderChian 的硬件设备都将被视为一个独立的节点，并且通过 CoderChain 贡献硬件设备闲置的存储空间。

(3) 劳动成果保障机制

CoderChain 会对开发者的原创代码等劳动成果进行有效的保障。基于区块链的技术特点和思想，CoderChain 将会提取每个开发者的代码等劳动成果特定的特征信息，并且将开发者的身份信息与代码的特征信息绑定，保证代码权属的真实性和唯一性，并且受保护的代码的后续交易将会被实时记录。

1. 开发者身份信息

在 CoderChain 开发者社区中，每个参与社区的开发者将被赋予 IPFS 系统中一个唯一的身份，也可称为节点，每个身份都持有唯一的公私钥对，身份的标识可以是身份公钥的哈希散列，也可以用自定义的别名来替代复杂难记的散列值（开发者的各个属性，用户空闲存储空间共享）。该身份用于标识开发者在社区内的各项活动，如社区讨论、交易、代码创作等。同时开发者身份信息也是用来保护开发者劳动成果原创性、侵权追责的重要依据。

2. 版权信息记录

为开发者提交的每一份原创代码使用散列函数生成唯一的哈希值标记，该标记作为证明代码原创性的重要依据，能够有效地支持代码鉴权、授权、维权等服务。CoderChain 将版权保护的信息全部记录上区块链，包含代码特征信息、开发者身份信息、创建的时间等，基于区块链技术保证信息的公开性、不可篡改性和可追溯性，确保开发者可以无阻碍的查询所有相关信息。

4 价值评估与度量

相对于当今开源社区中激励模式的缺乏，本社区不仅蕴含了开源思想，还对项目的每次代码提交和版本迭代都进行价值评估，并以此实施价值贡献激励机制，奖励在社区中实际价值高的代码或者项目原创开发者。社区从代码静态分析和代码测试准确评估和鉴赏代码或项目的实际价值和潜在价值。在代码评估的基础上，为开发者建立多维度价值度量模型，进而建立社区公认，可信，真实的开发者综合素质标识。

4.1 代码静态分析

代码评审是指通过阅读代码来检查源代码与编码标准符合程度以及代码质量的活动，通过代码评审可以量化开发者的编码水平以进行分析和比较。但目前国内外并无通用的自动化代码评审工具[9]，且代码语言种类和设计模式种类繁多，因此人工评审是目前代码评审最主要的方式。本小节将探讨静态代码分析在人工代码价值评审方面的应用。

静态代码分析是利用手工或工具，通过词法分析、语法分析、控制流和数据分析流等技术，在不运行代

码的情况下，对软件源码进行规范性、安全性、可靠性等方面进行评判的一种代码分析技术。代码静态分析相比起其他分析方式效率较高，且已有成熟的代码静态分析工具可供使用，但是目前常用的代码静态分析工具并不能覆盖所有的检测环节，只能检测现有的部分问题，且存在较高的误报率[10]。因此，CoderChain 中不仅会使用多种静态代码分析工具交叉对源代码进行扫描，综合不同工具的结果，同时也会使用机器学习和人工审计的方式来对代码进行多维度的分析，以更加精准地辨识开发者的编码水平。在人工审计和机器学习建模中，对代码中以下特征进行提取和处理：

(1) 编程语言：类型安全, 执行效率, 开发难度, 代码规模

(2) 代码编写与设计: 语法错误, 类/函数设计模式[11], 控制结构，冗余代码

(3) 安全审计：危险 API 调用，内存管理，竞态问题，溢出检测

代码静态分析的得分由三个部分组成：工具评分、机器学习模型评分以及人工审计评分。评分权重配比依项目的发展分为三个阶段：

(1) 项目初期，由于样本数量不足，机器学习模型尚需优化，评审人员亦需要根据工具评分和其他参考资料建立评分标准，因此工具评分所占比重最大，权重配比为工具评分>人工评分>机器学习模型评分。

(2) 项目发展期，随着评审人员经验的积累和人才的引入，人工评审将比代码静态分析工具评审的结果更具参考价值，同时代码样本数量的增长将不断地优化模型，令机器学习模型评分更加接近人工评审的评分。在这个阶段，权重配比为人工评分>工具评分>机器学习评分

(3) 项目成熟期，随着模型的不断扩充和优化，机器学习模型评分将可以自动化对项目中大多数用户的代码进行静态分析且得到相对准确的结果。此时人工评审的作用为维护优化模型，如对结果进行抽查，为对模型得出的极端结果进行分析处理，依照编程语言、编程范式和用户编码水平的发展动态地对模型进行修改和优化。在该阶段，权重配比为机器学习模型评分>人工评分>工具评分。

4.2 代码测试

为了准确评判代码价值，除了对代码做静态分析之外，社区还会对代码在平台上进行代码测试。因为在应用开发过程中，代码的设计和抽象逻辑性都很难在单独的某行代码中体现出来。因此，社区将从代码测试的三个方面来完善代码静态分析在项目设计等方向的不足，对代码做较为全面评审，分别为软件测试，功能测试和故障测试

4.2.1 软件测试

软件测试旨在发现软件或者程序潜在的错误而执行的测试操作，测试结果与代码的质量成反比，即软件测试用以评审代码的错误。此外，失效模式及后果分析(FMEA)随着软件测试阶段先于制造开发过程，开始于产品设计或需求任务发起，伴随代码版本迭代指导贯穿实施于整个产品周期。在评判代码价值激励开发者活动的同时，提高代码质量和产品开发效率。

4.2.2 功能测试

功能测试阶段需要验证系统的各个模块是否能准确有效的完成功能。测试算例的自动生成是成功进行软件测试的关键技术之一[12][13][14]。整个测试阶段用例自动生成，社区支持根据项目发起者的需求生成测试准则；并根据测试准则和源代码生成测试用例；自动完成源代码的编译和链接；自动执行测试用例，并自动采集测试结果；将测试结果中的故障信息移交给下一步的故障测试，并计算故障评分。

(1) 测试准则生成：对源代码和项目需求进行分析[15]，获得相关信息和用户输入准则生成测试准则，以 xml 等语言的形式生成准则文件。

(2) 测试用例生成：根据需求和代码对输入数据作等价类划分，同时涵盖无效输入的等价类和有效输入的等价类。选取具代表性，最可能发现程序逻辑的错误的样例。并通过分析输入和输出条件的边界值，补充测试用例。除此之外，在程序的适当位置安插覆盖监视器，确保控制语句的执行次数，以便增加相应的测试数

据。

4.2.3 故障测试

常见的几种故障分类方法有：出现错误的开发阶段，对应失效的后果，解决故障的困难程度，异常发生的错误频率。根据不同的故障分类方法了解根据软件故障对整体系统的影响和严重程度，检测难易度，发生频度，解决故障难度，建立风险系数模型(即多因素模型或指数模型)，可以较为全面进行故障评分，将代码评审结果数字化，比如风险系数越高表示代码质量越差。

4.3 社区影响

从社区影响上的角度上，目前代码的社区价值基于社区开发者的评价机制，主流的开源代码托管平台（如 GitHub、码云等）所采用的代码点评机制都是基于群众性的“star”机制，而针对这一机制，目前网络上出现了相对应的刷取 star 等造假行为，严重损坏了平台评价机制的公平性和权威性。

4.3.1 代码价值度量

CodeChain 不再依赖于传统代码托管平台的评价机制，而是采用使用代码静态分析结合代码测试的方式，从源代码角度与软件产品角度对代码的安全性、单元及模块测试等性能进行一系列的评测，对所评测每个指标给予相应的权重，从而建立一个基于代码本身属性的可量化的代码价值度量。随着 CodeChain 项目发展，各方面的评分与权重比会进行相应的调整，以保证评分的准确性与发展性。

通过代码价值度量的建立，首先将代码的评价机制由主观性的“star”机制转变为客观、权威以及可量化的评价机制，增强了对代码质量评审结果的说服力。其次代码上传者与平台的其他开发者可以通过该代码的评测结果了解其各方面的优点与不足，以便确定后续版本的更新内容以及对代码提出相应的“issue”，促进开发者之间的交流。

相对于传统的代码托管平台所提供的功能，CoderChain 不仅具有多维度的代码评测机制，而且根据社区开发人员各方面能力与行为建立的开发者价值评测机制。

4.3.2 开发者价值度量

传统的代码托管平台只是为个人或组织提供软件仓库管理服务的在线分布式版本控制系统，通过开发者与项目、项目与项目、开发者与开发者三种关系为开发者构建了一个基于代码托管平台的社交环境。而如何在这个环境中对开发者进行个人价值的度量，CodeChain 在基于代码价值度量的前提下，进一步提出了一个基于开发者所提交代码的质量以及其自身行为的开发者价值度量。

其相关的参数名称及相关描述见表 1.

Table 1 开发者价值度量

表 1 开发者价值度量相关参数

名称	描述
活跃度	开发者的代码提交、commit 频率
积极性	开发者对 issue 的响应速度
声望值	开发者所获得的 star、follower 的数量、代码的引用量等
贡献度	开发者对于软件项目社区提交贡献的成功率
Coding 能力	开发者的代码价值
其他	有待补充

代码托管平台上的资源，除了所被托管的代码，还有活跃在平台上形形色色的开发者。在如何充分利用到开发者资源的问题上，CodeChain 通过对表 1 中的各项指标的量化采集，能够对开发者的综合素质建立一个多元的判定度量。

通过构建多维度的价值度量，来对开发者进行客观、公平的评价，对其各个方面的特点进行展现。基于此度量，社区组织者或项目发起人能够对开发者进行客观的评估，更好地筛选与选择与社区（或项目）契合度高的开发人员加入到核心开发中，进一步促进社区（或项目）的发展。

5 总结

传统的开源社区不再适应当今时代开发者的切实需求，本文提出基于区块链的开源代码社区 CoderChain 为广大开发者提供开源交流服务。CoderChain 基于公有链和联盟链双链共行机制，应用分布式代码管理存储，并且建立多维度代码鉴赏机制，准确评估价值量高的代码和综合素质高的开发者，进而建立社区公认的开发者综合素质标识，为 CoderChain 社区的所有开发者提供互相促进与进步的代码开发平台。

在未来的社区建设任务中，我们将仔细研究劳动成果量化机制与社区贡献激励机制与社区代码多维度价值评估机制的结合，改变开源社区中开发者的收益模式的同时激励开发者不断为社区，其他开发者做出有效贡献。

References:

[1] YUAN Yong, WANG Fei-Yue. Blockchain: The State of the Art and Future Trends[J]. ACTA AUTOMATICA SINICA, 2016, 42(4): 481-494. 袁勇, 王飞跃. 区块链技术发展现状与展望[J]. 自动化学报, 2016, 42(4): 481-494.

[2] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system[J]. Consulted, 2008.

[3] 韩璇, 刘亚敏. 区块链技术中的共识机制研究[J]. 信息网络安全, 2017(9):147-152.

[4] Davidson, Sinclair and De Filippi, Primavera and Potts, Jason, Economics of Blockchain (March 8, 2016). Available at SSRN: https://ssrn.com/abstract=2744751 or http://dx.doi.org/10.2139/ssrn.2744751

[5] D Shrier, W Wu, A Pentland - Massachusetts Institute of Technology , 2016 - getsmarter.com

[6] 侯孟书. 基于 P2P 的分布式存储及其相关技术研究[D]. 电子科技大学, 2005.

[7] Benet J. IPFS - Content Addressed, Versioned, P2P File System[J]. Eprint Arxiv, 2014.

[8] Sicilia M A, Sánchez-Alonso S, García-Barriocanal E. Sharing Linked Open Data over Peer-to-Peer Distributed File Systems: The Case of IPFS[M]// Metadata and Semantics Research. Springer International Publishing, 2016:3-14.

[9] 罗琴灵. 基于静态检测的代码审计技术研究[D].贵州大学,2015

[10] 高远. 基于模型的代码自动化评审[D].南京大学,2012.

[11] 何剑涛. 基于设计的代码自动评审方法[D].南京大学,2014.

[12] 郑人杰，实用软件工程，北京：清华大学出版社，1997

[13] Mark Fewster , Dorthy Grahmaz 著，舒智勇等译，软件测试自动化技术与实例详解，电子工业出版社，2000

[14] 刘胜，软件测试及其自动化，信息技术，no: 4, 2001:39-40

[15] 毛颖, 测试用例自动生成系统研究与实现, 2007，电子科技大学