

**项目计划书修订版（B）**

2020.11.17

王芳琴

**基于区块链的投票方案**

目录

[基于区块链投票系统 1](#_Toc5993)

[1.1 背景和现状 1](#_Toc22555)

[1.2 匹配度分析 4](#_Toc4961)

[投票流程分为四个阶段： 4](#_Toc13547)

# 基于区块链投票系统

**一** 需求分析

### 背景和现状

#### 背景

现代民主已为许多国家所采用，并以此为基础建立了一个国家的法律体系。但是，选举是一项使公民能够参与民主，从而投票决定其优先选择来管理国家事务的计划。选举系统可以用于需要人们在某些问题上达成一致而没有任何偏见的组织。在民主的政府体制中，人们对主题有不同的看法和观点，因此，决策是通过投票完成的，这可以通过传统的选票或电子投票（电子投票）系统来实现。

当前，非洲大多数发展中国家的现有投票方法是传统的选票投票系统，因此，选票是进行投票的唯一可用平台。此要求将选民限制为手动拇指打印，因此会在投票期间和之后导致违规行为。它还使投票系统容易受到攻击，从而为篡改和修改投票结果提供了机会。以下是这种投票方法面临的一些挑战；1由于手动计票，该系统速度慢，效率低下且不准确。选民无法及时获得选举结果。2该系统需要巨大的人力成本和时间，即操作成本很高，因为您必须将大量人员部署到投票站，这肯定会需要成本。它为欺诈，操纵选票和其他选举违规行为提供了空间，即不可靠。这需要大量的文书工作，可以破坏选票和选民的登记册。由于这些缺点，有必要引入自动化系统来解决这种手动投票方法。在开发解决现有选票系统所面临问题的平台时，引入了电子投票系统以满足需求并启用了无压力投票平台。虽然，近来选民的冷漠感有所增加，特别是在精通技术的年轻一代中，鼓励投票系统为吸引年轻的选民参加选举活动提供一种途径。因此，系统应该允许一些健壮的安全性和投票功能要求，例如投票者的身份验证，完整性。正确性，隐私权，可用性，可审核性，尤其是投票系统的安全性

与传统的选票系统相比，电子投票系统更不容易出错，对环境友好，并且可以实时统计和计票，从而提高选民在选举期间的投票率

#### 1.1.2 国内外现状

数字签名是数据鉴权和身份认证的主要手段，是一项重要的信息安全技术，它的基本作用是保证传送信息的安全性和可用性，并确认签名者的身份7，盲签名技术是一种特殊的数字签名，它可以保证在签名者不知道所签消息真实内容的前提下得到签名者的有效签名。D.Chaum于1983年首先提出盲签名），并设计了基于RSA签名体制的盲签名方案。1993年，Okamoto首先提出了基于离散对数难题的盲签名方案1，之后，对盲签名的研究不断深入，盲签名中也增加了群盲签名、部分盲签名等特殊算法，盲签名是数字签名技术的重要组成部分，有着重要的地位。现今，已经提出的盲签名应用主要涉及电子支付、电子现金、电子合同签署及公平电子投票协议等安全场景。

电子投票协议是现代民主活动的象征，用于满足互联网环境下的选举和投票活动的各种需求，电子投票的应用已经深入到民主生活和政治选举中，比较著名的应用案例：2000年，美国的佛罗里达州选举试验、2002年，巴西总统选举、2004年，美国总统大选、2007年，法国总统大选初选、2009年，中国杭州市基层选举、2014年，法国国家教育部投票。

许多研究者都致力于设计一种安全高效的电子投票协议，目前主要有两个发

展方向：一是基于同态加密技术，掩盖选票的票面信息，并通过公开信道传输1；二是基于匿名信道技术，掩盖投票者的身份信息，其中，基于同态加密技术的投票协议1l采用高次剩余加密方式，需要进行大规模的计算和传输，效率低下，实用性弱，基于匿名信道的投票协议所采用的底层密码学技术可以分为盲签名、环签名和代理签名三种。其中盲签名算法可以有效地保护所签署消息的具体内容，在电子投票这种需要匿名的应用场景中能够发挥重要作用。基于百签名算法的电子投票协议可分为有可信第三方（Trusted Third Party，TP）的投票协议和无TTP的投票协议两种.

无TTP的投票协议最初是由Michael Merrit等1）提出，该协议每次投票需要运行的投票次数与投票者数成正比，每轮的计算量大。苏云学3等提出了一种匿名投票的方案，一次完整的投票过程需要执行固定的六轮计算.Patrick Mcorryl1）

等提出了一种利用区块链平台--以太场--实现的投票协议，但是由于以太坊不支持加密计算，导致程序冗余复杂：总体来说，无TTP的电子投票协议鱼有一定发展，但是协议通常较为复杂，因为效率问题而无法进入实用阶段（347在有TTP的电子投票协议中，D.Chauml1"1提出的方案首先引进盲签名思想来保护选民的身份隐私。1992年，Okamoto和Ohta提出FO0协议18，它是最早实现大量商用的投票协议，该协议结合了盲签名和比特承诺技术，是投票协议发展过程中一个重要的进步，FO0协议简单高效，它使电子投票技术正式进入实用阶段。

基于FO0协议出现了大量的商用投票软件，例如EVOX和SENSUS，但后来FOO协议被证明存在选票碰撞缺陷，无法满足不可伪造性要求，而且存在共谋攻击的威协，此后有许多在FO0协议基础上的改进研究。1996年，Juang和Leil0提出了FO0的改进方案，方案要求所有投票者全部参加投票.丛清日等21提出新的方案，解决了F00协议的弊端，但是在结果公布阶段，公开了投票者的身份信息签名和公钥，渔成隐私数据泄露，破坏了匿名性，上述协议均存在TTP，投票者被动信任TTP，且必须设立TTP.在这种情况下用户无法发现和防止TTP的共谋攻击，上述存在的问题增加了电子投票协议的使用门槛和信任成本，限制了电子投票协议的普及.

目前，区块链技术因其不可幕改、安全可信等特点，被称为是具有改变传统业务乃至机构运作方式的潜力的重大煎覆性技术1，随着区块链技术的快速发展，些研究人员提出了使用区块链技术的电子投票协议.2015年，Czepluchl22提出区块链技术可以应用于电子投票领域，同年ZHAO12）等人提出一种使用比特币和zk-SNARKs进行投票的方法，具有隐私性、有效性和不可撤销性.2016年，一种使用零币的投票协议也被提出24，但是由于零币的软件开发比较困难，所以不具备实用性，同年，CJason9等提出一个使用百签名和比特币实现的协议，但是系统管理员能够通过比特币地址和消息得知投票者的身份，即存在管理员共谋攻击的可能性。

除此之外，利用区块链技术的电子投票应用也不断涌现，2018年11月，泰国民主党利用区块链技术选举党内领导人，比较典型的应用还有Bicongress，Pollow MyVote 和TIVI，这些应用使用区块链网络提供投票底层服务，但是仍然依赖第三方机构来保护用户隐私数据，同时这些应用大多缺乏适当的文档，关于其内部工作的问题仍然存在不透明的问题。

#### 1.1.3 存在的问题及面对的风险

投票活动是社会生活中制定决策的重要依据，投票形式从传统的举手表决、匿名投票到电子投票演化，投票成本不断降低，投票过程愈发便捷。但是不管在投票的哪种形式下都有各种问题。

1. 暗箱操作

记录中最早得雅典投票之一是多数决的陶片放逐，由每个人放陶片选择放逐的人，这种方法简单直接，但是只能在小范围使用。随着时代的发展这种方法逐渐被取代。在早期投票方式多为人工投票，由政府或机构组织，给人投票权，并由这些组织和机构计票这避免不了暗箱操作。

发展至今，网络投票层出不穷，许多选秀节目多用观众投票选出人气最高的演员或歌手。但是还是会出现刷票情况。例如之前炒作的沸沸扬扬的蔡徐坤刷票，宣传新歌的微博中获得上亿次转发。微博的使用人数也就三亿人左右，上亿次的转发量是什么概念？相信里面的真实数据并没有表面上那么恐怖。蔡徐坤出身自选秀节目，为了吸引观众，提高选手出道的曝光率，采用了“全民投票机制”，而有些选手为了提高票数只能想方设法的圈粉拉票，但是有些人却喜欢走捷径，电子刷票就是其中最快的方法。在这种需求下，虚假流量应运而生。日前，《歌手2019》便遭遇“刷票造假事件”。其实，追溯根源还是“全民投票机制”惹的祸，不管是微博投票还是微信等等社交软件的投票，这种方法早在十多年前湖南卫视就玩得相当熟悉了。很多80、90后肯定对“短信投票”不陌生，当年的电视选秀节目《超级女声》、《快乐男声》想必都耳熟能详，在那个手机都还没普及的年代，有多少还在读书的孩子省吃俭用的用小灵通为艺人投票？一块钱一条的短信投票，哪怕放在现在都觉得昂贵。刷票行为破坏了投票公平公正的规则，而又很难避免暗箱操作的情况，最主要的原因是计票记录可被后台篡改，票数也能被篡改，并没有所谓的公开透明。

1. 信息泄露

如今只要手机上安装着微信社交软件的人相信基本上都参加过微信投票活动，并且是很积极的参加，在我们的朋友圈里每天都可以看到各式各样的微信投票活动，有的是朋友转发的活动，自己都根本不知道这个微信投票活动是属于哪里的，因此就会有很多人在不知不觉中被骗或者个人信息被泄露。如今朋友圈微信投票活动，早已蜕变成“传销式”商业活动，参与者的个人信息很可能会被泄露，父母和孩子也早已成为“黑色利益链”牟利的工具。这样的微信投票活动，你还会让孩子参加吗？如今很多人一打开微信，求点赞、求投票的消息就扑面而来，朋友圈好像已经成了“投票圈”。这些“投票比赛”大多不需要拼才艺，只需要一张照片和相关信息即可报名参加，比赛规则也很统一，只需要投票数名次靠前即可得奖，但奖品却十分诱人。很多父母给孩子报名参加，大多数人也不在乎投票的“举手之劳”，有时候也会帮忙转发。可很多人并不知道：朋友圈微信投票活动，早已蜕变成“传销式”商业活动，参与者的个人信息很可能会被泄露，父母和孩子也早已成为“黑色利益链”牟利的工具。打开朋友圈，到处都是言辞恳切的“拉票”链接。打开这些链接，不难发现，主办方多是私利机构，公信力并不强。可很多家长却选择性忽视这些，觉得孩子的事情就应该重视。父母为孩子报名参赛，本意是想鼓励孩子，或者是想为孩子赢取奖品，可根本没有意识到：父母把孩子的身份、姓名、照片，甚至就读的学校，住宿的地址（声称以便邮寄奖品）都提供给后台，孩子的个人信息毫无疑问存在泄露的风险。警方表示，他们在打击电信诈骗的时候，就遇到过这类案例：后台获取参赛者信息后，将信息非法贩卖，骗子可以通过孩子身份、照片或就读学校的信息，制造孩子重病，绑架等骗局，对父母进行诈骗。

央视曝光微信投票骗局，即使后台不贩卖信息，孩子参加此类比赛就安全吗？参赛孩子大多都以真实姓名和照片参赛，有的参赛页面还具体到年龄、学校、班级和居住的小区。孩子一旦参赛，参赛页面全网用户都能看见，甚至比赛结束后，参赛页面也会一直存在，不法分子随时可以获取孩子信息。很多人贩子和孩子搭讪，能准确叫出孩子名字，并对孩子的基本信息了如指掌。这个投票页面，把孩子的真实姓名、照片、幼儿园信息全部暴露了出来。朋友圈微信投票的泛滥，让很多人心生怨言，可碍于情面，大家还是会帮忙投上一票。但在帮别人投票的过程中，我们自己的个人信息也会被泄露。有时候，微信投票并不是对选手号码直接点击就可以，而是要先关注公众号，或者让公众号获取你微信的权限（昵称、头像等个人信息），有的甚至还需要填写手机号验证码，而这一过程中，我们的信息就会被泄露。有时候我们明明新办的手机卡，没有在网上购物或者商店消费的时候用过，可还是会收到各种商家的“骚扰”电话或短信。这很可能就是在参与了这类活动后，导致我们的个人信息被泄露了。现在人们微信大多都绑定了银行卡，信息一旦泄露，就会危及财产安全。有媒体报道，王先生给朋友家孩子投票，手机莫名其妙被扣了6570块钱（银行卡全部余额），经过他多方努力，最终只追回来2000多块钱。这些比赛大多都是通过公众号或者网址进行发布的，投诉最多把公众号和网址链接封掉，而他们的真实身份和公司根本不会对外公布，联系方式更是会随时更改，他们换个公众号和大赛名字，就可以开始新一轮骗局。不排除一些正规比赛通过这种方式进行，可是很多骗子也看到这个机会，收集人们信息，利用收集到的信息做不法勾当，给我们带来经济和名誉上的损失。如下图，在一，二，三，步骤都是不安全的，步骤一中，不能保证后台不会收集发布投票人的信息，该方法中步骤二，首先链接就不能保证其安全性，该系统还会容易被黑客攻击，再通过链接攻击接收链接的投票人。该步骤中也会泄露投票人信息。步骤三中由后台计票，计票过程中就不能保证没有内定，后台管理人员不会对数据进行更改，且一般投票的票数都能通过购买礼物来增加票数，可以说，很多私人机构投票是以盈利为目的，没什么公平而言。

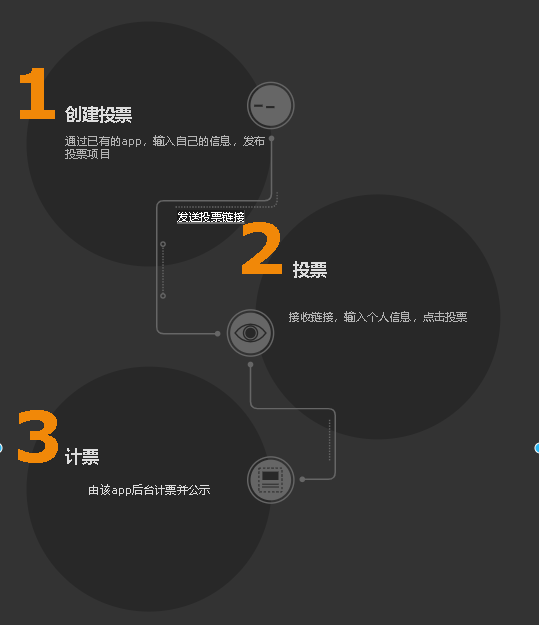


图1

而以太坊给投票带来了希望，其不可篡改性，可以很好的应用在投票系统中，实现投票人一人一票，记录公开，投票透明。

## 匹配度分析

在网络投票应用中引入区块链技术，可以借助区块链的去中心化同步记账、身份认证、数据加密和数据不可篡改等特征，确保投票信息可信任且可追溯，使各社会主体共同建造、共同维护、共同监督，从而满足公众的知情权、监督权，增强网络投票的客观性与可信度。

（1）**保证投票匿名性：**用户只需注册，登录，投票，数据记录在区块链中，过程由智能合约完成，投票中途不受第三方平台控制，防止用户投票信息泄露。同时，点对点查询投票信息，实现投票信息只由投票者查询。

（2）**可信的投票数据：**区块链将加密技术的功能与透明性相结合，这使其成为在线投票的便捷且安全的选择。 记录在区块链上的选票将具有防篡改功能。 人们现在可以在家中方便地进行投票。 他们可以在不损害安全性或隐私的情况下使用计算机或移动设备进行投票。 此外，他们将能够核实自己的选票并确保被计票。

（3）**投票的可验证性：**投票者进行投票操作后，选票信息将公开 存储在区块链上，并且无法窜改,无论是投票者本人或是其他 人都可以对选票的真实性和有效性进行验证。在计票阶段，候 选者的私钥公开后,所有人都可以验证和统计有效的选票数。

（4)**投票资格：**只有符合条件的人才能被赋予投票权，拥有投票权才能进行投票。

（5）**投票的唯一性：**选民数组存储已投票的选民列表。它确保没有人可以第二次投票。一旦选民投票，他的状态将变为“已投票”，并且投票合同将进行检查以确保他不再投票。

1.3基于区块链的投票系统方案设计

1.3.1 业务设计

基于区块链的投票系统是一个去中心化的投票应用。利用这个投票应用，用户可以在不可信的分布环境中对特定的候选人投票，每次投票都会被记录在区块链上，所谓去中心化应用（DApp：Dcentralized Application），就是一个不存在中心服务器 的应用。在网络中成百上千的电脑上，都可以运行该应用的副本，这使得它几乎不可能 出现宕机的情况。

基于区块链的投票是完全去中心化的，因此无须任何中心化机构的存在。

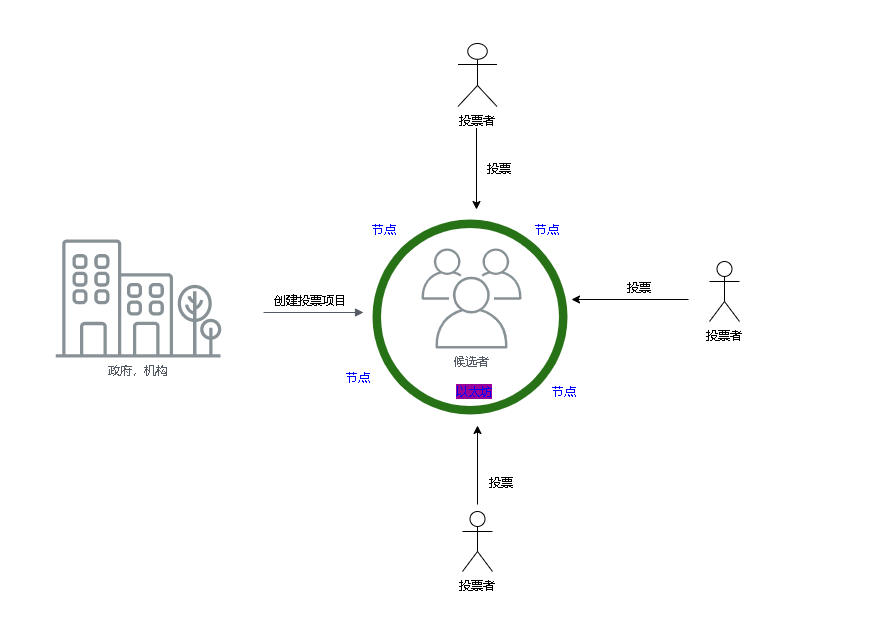


图2

1. 创建投票

由政府，机构或个人根据需求创建投票项目，在程序中命名投票名称，给定发布人名字，发布该项目。

2.添加投票人

发布人部署后在addVoter方法添加投票人，输入将被添加人的地址和姓名

3.查看投票合同的公共变量

查看发布人是谁，投票内容的内容，合约的当前状态以及投票人员的名称和她们是否投票。此操作可以确保透明度。

1. 发布人设置投票开始

发布人触发开始投票，在这开始后，不能添加任何新的投票者，该投票就是一成不变的。（只有发布人地址才能触发开始投票方法）

1. 投票

只有之前被添加的投票和才有权力投票，且合约会确认该投票者是否已经投票，如果她有投票选票，该合约将拒绝接受她的投票。如果没有，则接受她的投票，并且状态会更新为表示已投票。一旦该投票者投票投票，合同将不再允许该投票者再次投票。

1. 审查

在任何时候，任何人都可以检查一下voitRegister，以查看注册表中是否有人没有投票。

1. 结束投票

结束投票是重要的一步，因为它停止了允许尚未投票的任何人投票。除非投票结束，否则无法点票。现在，选票合同会计算已收到的所有票数，并记录“是”的总数。完成后，每个人都可以查看结果。投票过程已经结束，没有人，甚至没有主席被允许更改投票合同的任何性质。

1.3.2 架构设计

一．中心化应用构架

一个典型web应用的服务端通常由 Java，Ruby，Python 等等语言实现。前端代码由 HTML/CSS/JavaScript 实现。 然后将整个应用托管在云端，比如 AWS、Google Cloud Platform、Heroku....，或者放在你租用的一个VPS 主机上。

用户通过客户端（Client）与 web 应用（Server）进行交互。典型的客户端包括浏览器、命令行工具（curl、wget等）、 或者是API访问代码。注意在这种架构中，总是存在一个（或一组）中心化的 web 服务器，所有的客户端都需要 与这一（组）服务器进行交互。当一个客户端向服务器发出请求时，服务器处理该请求，与数据库/缓存进行交互， 读/写/更新数据库，然后向客户端返回响应。

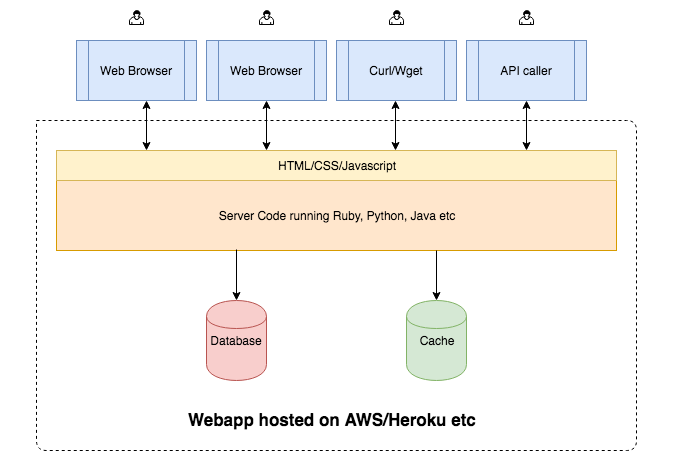


图3

1. **基于以太坊的去中心化应用构架**

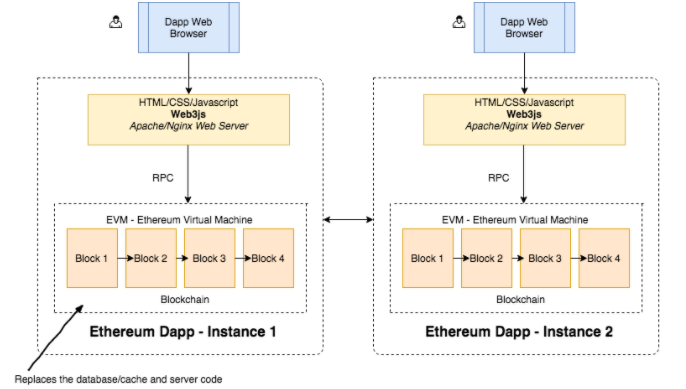


图4

基于以太坊的去中心化应用构架，每个客户端（浏览器）都是与各自的节点应用实例进行交互，而不是向 一个中心化的服务器请求服务。

### 基于区块链投票系统构架

该方案电子投票系统是在以太坊私有区块链环境上进行开发运行的，其中投票的协议由本文提出投票方案通过智能合约控制实现，所以投票系统具有去中心化的效果。投票的每一个阶段由投票发起者监管，负责其中每一个投票环节的开始和结束，一旦开始执行，智能合约自动执行投票流程。投票系统的架构如图4.1所示。投票系统采用分布式架构，每一个投票者通过操作投票系统Web前端页面实现投票每个环节并与智能合约交互。而以太坊智能合约通过编写调试完成后，在本地区块链节点通过以太坊虚拟机将智能合约编译成字节代码部署到区块链上，并拥有一个特定合约地址。当需要调用这个智能合约的方法时，需要向这个智能合约的地址发送一笔交易即可，通过JavaScript库web3.js的API调用，从而实现与之交互的去中心化投票系统DApp应用。以太坊虚拟机是建立在以太坊区央链上的一个代码运行环境，但虚拟机本没有存储在区块链内，而是和区块链一样同时存储于各个节点计算机上。每个节点都会对合约的部署和调用进行相同的计算，并存储相同的数据，以确保将最真实的结果记录在区块链上，从而实现系统的去中心化。

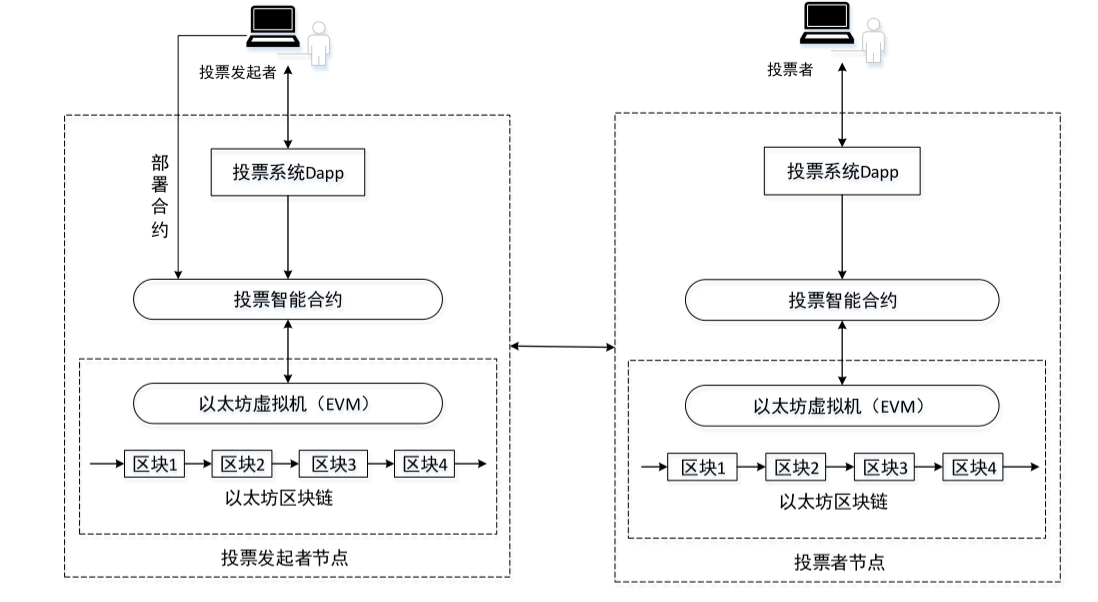
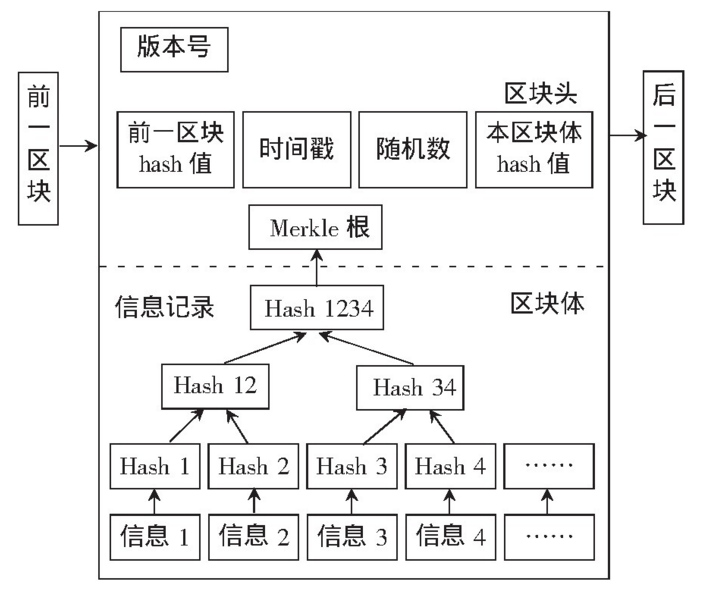


图5

如图6，对数据进行区块化封装，每个区块细分为区块头和 区块体两部分。其中，区块当前的版本号、上一个区块的地址、时间 戳、当前区块哈希值与随机数封装在区块头内；当前区块的交易数量以及经过验证的、区块创建过程中生成的所有记录封装在区块体中。区块 体由Merkle树组成，在Merkle树的叶子节点上，保存着每一个投票者的投票的记录，这些数据通过两两Hash计算向上形成Merkle树 的根记入区块头。

图6

1.3.3 交互设计

（一）整体业务流程设计

投票流程分为四个阶段：创建投票、获得投票权，投票、计票、审查。

1. 创建投票：项目发起者登录，创建项目，按要求输入投票创建单位提议名称。
2. 投票：投票者通过令牌进行投票，若投票成功则会返回交易哈希。
3. 计票：每一次成功的投票都会被记录在区块上，并会显示，每一次成功的投票都能看到票数的变化，事先制定好的时间截止后，程序自动停止，计算出总数，并排出得票最高者公示。
4. 审查：投票者可以通过自己的地址查询自己的投票信息，每个人只能看到他人是否投票，不能查询他人的投票详情。

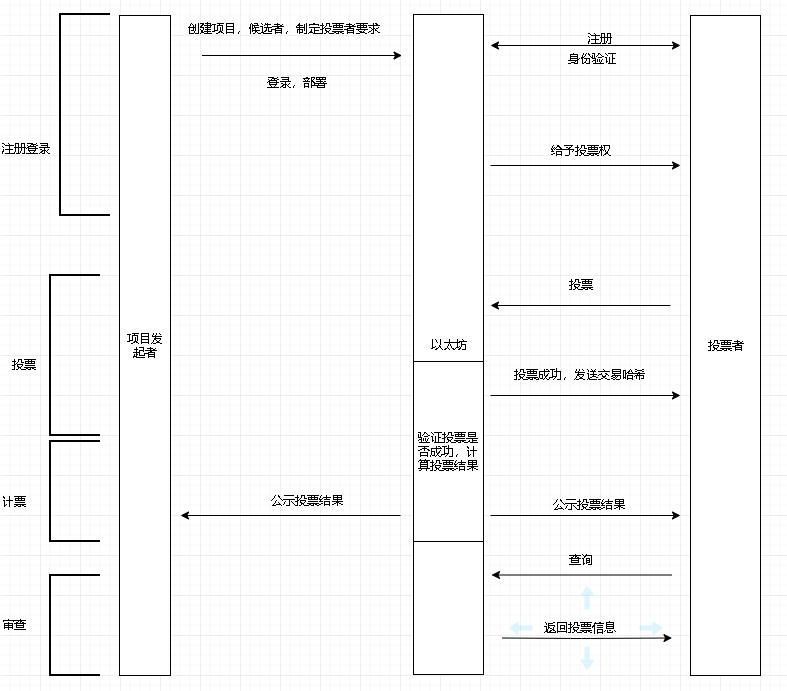


图7

1. **创建项目流程**

如图8所示，创建项目为整个系统第一个步骤，只有项目创建成功才能由主席添加投票者，继而有投票者参与投票，所以投票的创建为后面投票和审计环节奠定基础，创造条件。

创建项目客户端主要由主席来操作。第一，提交项目信息，主要填写项目的信息，和添加投票者，由主席提交项目信息后，发送到SDK。第二，SDK向CA发送注册项目信息，CA审核前端发送的项目信息。第三，若审核通过则由CA记录下项目信息，未通过则返回信息到SDK，前端返回到项目信息界面，提示审核信息未通过。需重新提交项目信息。第四，CA审核通过会发送投票权给投票者，再然后把用户投票权通过发到客户端用户界面。然后创建项目流程返回到客户端结束。

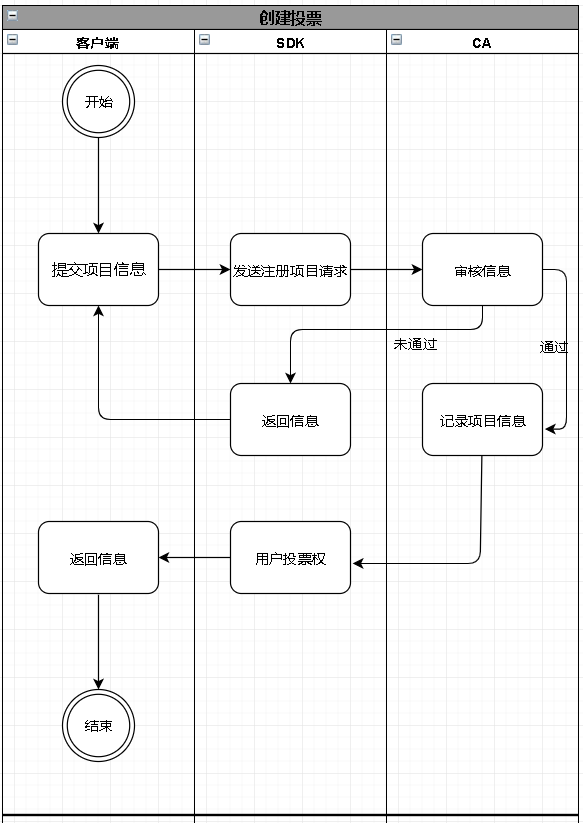


图8

1. **投票流程设计**

如图9，该客户端页面主要由用户也就是投票者来使用。第一，投票者使用自己的地址再客户端登录。第二，登录后生成正式选票，也就是投票，提交是否同意提案。第三，在客户端提交后由智能合约确认该投票者是否具有投票权，也就是创建项目时主席是否给予该用户投票权。若该投票者拥有投票权，继而可以投票。若通过智能合约审核该用户不具有投票权，则会返回登录，登录正确的用户。第四，由智能合约判断该用户提交的投票是否有效，若通过智能合约审核该用户提交的投票无效，则会返回客户端的生成正式选票阶段，再次提交投票。第五，若通过智能合约审核该用户提交的投票有效，该投票会被记录在区块链上了。第六，返回结果到客户端，再在前端提示成功。

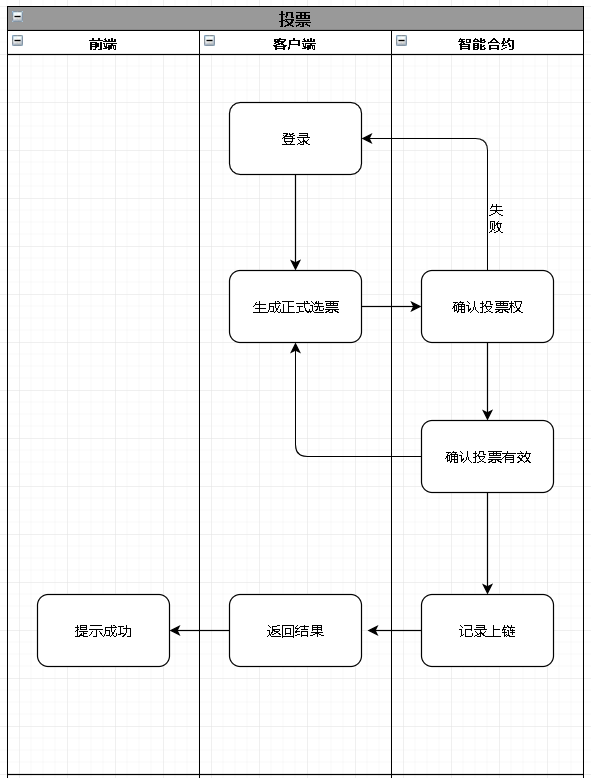


图9

2关键技术及方法

（一）区块链技术

在区块链的模型架构中，共识机制和安全机制是区块链的核心技术组件，现根据电子投票系统的应用，分别对非对称数字密码、分布式结构、共识机制等几个关键技术问题简要描述。

1.非对称数字加密

区块链的模型设计中，一般采用非对称加密技术对数据进行加密验证。该算法使用的是一对密钥：公钥和私钥。这对密钥在非对称加密应用方式有两种。一种是公钥加密私钥解密，主要接收数据信息。另一种是私钥加密（签名）公钥解密（验证签名），主要判断是否是该节点发送的消息。因此，利用非对称数字加密技术在安全性能方面大大提高。

RSA是常用的非对称加密算法之一，该算法既能用于数据加密又能用于数字签名，因此本系统采用了RSA算法。该算法RSA算法是将两个约数较多且容易相乘的大素数进行乘积运算，若想要将乘积因式分解则困难7，因此可公开乘积并可将其作为加密密钥。RSA算法具有抗攻击性，因此在数据加密方面采用RSA算法安全可靠。

2.分布式结构

区块链是一个具有共享状态的密码性安全交易的单机，分布式网络结构保证了数据记录存储在非中央计算机上，计算任务运行在多台计算机上，全网中的所有节点都将记录并存储计算任务中的数据交易信息。为此，区块链技术采用了开源的、去中心化的协议18，构建了一个分布式的结构体系，保证数据的完整记录和存储。

1. 分布式传播。区块链中产生新的交易数据均由单个节点向全网节点发送，若信息拦截者对部分节点传播路径进行恶意摧毁，不会影响整体信息的传输，全网中的所有节点均收到该数据信息。（2）分布式记账。区块链通过构建分布式机构体系和共识协议，允许所有节点在参与数据记录和验证其他节点结果的正确性。只有当所有参与记录的节点（或者大部分节点）比对结果一致通过，记录数据才允许被写入区块并最终将交易结果写入区块链。

（3）分布式存储。区块链构建一个分布式结构的网络系统，数据库中所有数据实时更新并存放于所有参与记录的节点。即使部分节点损坏或被攻击，也不会影响整个数据库的数据记录。

3.共识机制

分布式系统中，最重要问题是全网所有节点如何达成共识。共识机制就是为了保证底层区块链数据的一致性，并有效的抵抗恶意节点的攻击和破坏。在节点网络中，所有节点在是否攻击外部节点问题上达成一致，但存在部分恶意节点，故意传递错误消息给其余正确节点，干扰判断，这就是拜占庭容错问题。传统的Paxos，Raft没有考虑这个问题；而POW算力过高，POS，DPOS过于集中化，三者广泛应用于公有链的虚资产。PBFT算法是联盟链中最常用的共识算法，通过投票来达成共识的机制，并解决数据分叉以及投票数据丢失和篡改等问题。

PBFT容错算法是一种基于消息传递一致性的算法，节点集合P中，每个节点都有对应唯一的编号，该系统中能容忍恶意或坏死节点的数最多为f，当节点数P13f+1的情况下可以达成共识。PFBT共识过程如下：（1）客户向服务器中的主节点发送调用服务操作的请求命令。（2）主节点向全网的节点广播请求。（3）所有的节点处理该请求，并将结果返回客户。（4）客户将返回的处理结果进行整理，当返回结果中至少有f+1个结果相同，则可停止该操作。

PFBT验证过程三阶段：预准备、准备、确认阶段。

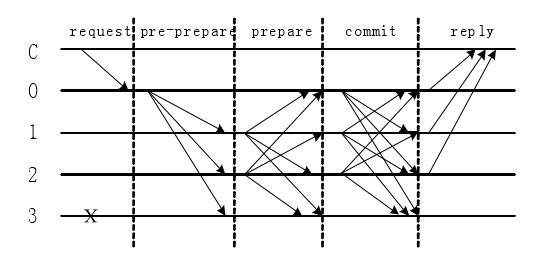
预准备阶段，主节点分配一个系列号n，将PRE-PREPARE消息发送给其余节点。之后主节点将PRE-PREPARE消息广播到其他各节点。

准备阶段，各节点在收到PRE-PREPARE消息后，需验证消息的内容没有被篡改，验证结束后，该主节点会向各节点发送PREPARE消息。

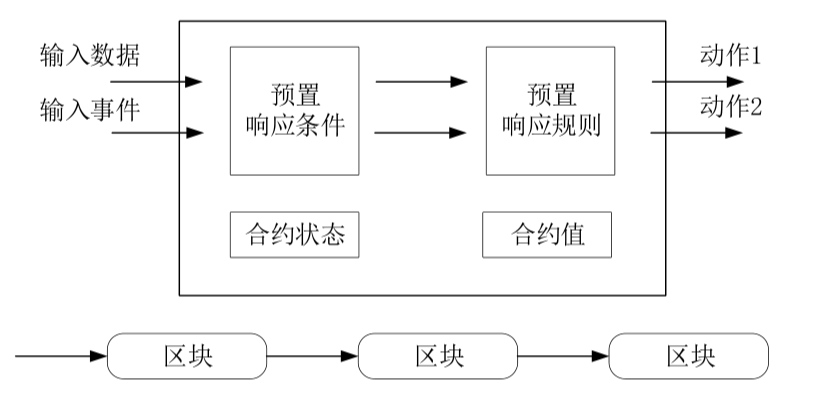
确认阶段，当节点验证PREPARE消息为真后，会在全网中广播PREPARE消息。

验证完COMMIT消息后，节点可以将结果返回给客户端。

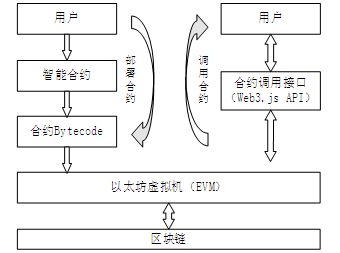
该算法性能比较高，消耗资源少。因此比较适合现代的电子投票系统。执行流程如图1所示。



1. 智能合约（Smart Contract）



是一个由计算机处理的、可执行合约条款的交易协议。以太坊的智能合约是一段可被以太坊虚拟机执行的代码。以以太坊特有二进制形式存储在区块链上，并由以太坊虚拟机解释，被称为以太坊虚拟机码（Bytecode）。智能合约像一个可以被信任的人，可以临时保管资产，总是按照事先的规则执行操作。它对接收的信息进行回应，可以接收和存储价值，也可以向外发送信息价值。目前，主要使用一种类似Java语言的Solidity语言进行智能合约开发。智能合约部署和调用流程如图所示。



1. 智能合约的开发和实现
2. 智能合约的开发

开发语言：solidity

编辑器： Visual Studio Code

在线编辑器：Remix

交易桥梁： Matemask

1. 智能合约的实现
2. Ballot合约概述

|  |  |
| --- | --- |
| 合约名称 | 合约内容 |
| Ballot.sol | 实现创建投票，投票，审查 |

|  |  |
| --- | --- |
| 方法及参数 | 功能 |
| constructor( string memory \_ballotOfficialName,string memory \_proposal) | 初始化项目 |
| function addVoter(address \_voterAddress, string memory \_voterName) | 添加投票人，给予投票人投票权 |
| function doVote(bool \_choice) | 投票 |
| function endVote() | 结束投票 |
| function startVote() | 开始投票 |
| ballotOfficialAddress = msg.sender; | 查看主席地址 |
| ballotOfficialName = \_ballotOfficialName; | 查看主席名称 |
| finalResult = countResult; emit voteEnded(finalResult); | 最终结果 |
| proposal = \_proposal; | 提案 |
| modifier inState(State \_state) {require(state == \_state);\_;} | 投票状态 |
| votes[totalVote] = v; totalVote++; found = true; | 总票数 |
| totalVoter++; emit voterAdded(\_voterAddress); | 总投票人 |
| voterRegister[\_voterAddress] = v; | 个人投票情况 |

1. 页面实现
2. 创建投票

合约部署时就需要填写主席名称和提案内容也就是提案名称，部署后，主席将添加投票人，给予他们投票权。输入他们地址和名称再提交即可。而后可以启动该投票项目。在一切准备就绪后即可点击开始投票提交按钮，项目正式启动，在投票项目时间截止时则需要主席点击结束投票提交按钮。项目立即停止。且这一切操作只有该合约部署者可操作。



1. 投票

此页面由投票人来操作，功能主要是提交投票，是否同意提案。要求提交投票人的地址和投票人名称。再可提交你是否同意。



1. 审计

  
pragma solidity ^0.5.0;

contract Ballot {

struct vote{

address voterAddress;

bool choice;

}

struct voter{

string voterName;

bool voted;

}

uint private countResult = 0;

uint public finalResult = 0;

uint public totalVoter = 0;

uint public totalVote = 0;

address public ballotOfficialAddress;

string public ballotOfficialName;

string public proposal;

mapping(uint => vote) private votes;

mapping(address => voter) public voterRegister;

enum State { Created, Voting, Ended }

State public state;

//creates a new ballot contract

constructor(

string memory \_ballotOfficialName,

string memory \_proposal) public {

ballotOfficialAddress = msg.sender;

ballotOfficialName = \_ballotOfficialName;

proposal = \_proposal;

state = State.Created;

}

modifier condition(bool \_condition) {

require(\_condition);

\_;

}

modifier onlyOfficial() {

require(msg.sender ==ballotOfficialAddress);

\_;

}

modifier inState(State \_state) {

require(state == \_state);

\_;

}

event voterAdded(address voter);

event voteStarted();

event voteEnded(uint finalResult);

event voteDone(address voter);

//add voter

function addVoter(address \_voterAddress, string memory \_voterName)

public

inState(State.Created)

onlyOfficial

{

voter memory v;

v.voterName = \_voterName;

v.voted = false;

voterRegister[\_voterAddress] = v;

totalVoter++;

emit voterAdded(\_voterAddress);

}

//declare voting starts now

function startVote()

public

inState(State.Created)

onlyOfficial

{

state = State.Voting;

emit voteStarted();

}

//voters vote by indicating their choice (true/false)

function doVote(bool \_choice)

public

inState(State.Voting)

returns (bool voted)

{

bool found = false;

if (bytes(voterRegister[msg.sender].voterName).length != 0

&& !voterRegister[msg.sender].voted){

voterRegister[msg.sender].voted = true;

vote memory v;

v.voterAddress = msg.sender;

v.choice = \_choice;

if (\_choice){

countResult++; //counting on the go

}

votes[totalVote] = v;

totalVote++;

found = true;

}

emit voteDone(msg.sender);

return found;

}

//end votes

function endVote()

public

inState(State.Voting)

onlyOfficial

{

state = State.Ended;

finalResult = countResult; //move result from private countResult to public finalResult

emit voteEnded(finalResult);

}

}

# 3结束语

随着互联网技术以及现代密码学技术的发展，网络投票成为一种新的投票方式，解决了传统的投票方式成本高、效率低、失误多等众多缺点。虽然网络投票系统使得整个投票过程在活动的组织、选票的收集和结果的统计等方面都节省了大量的成本，但是目前的电子投票系统也存在弊端：（1）、数据传输的安全性不能保障，攻击者很容易入侵系统，篡改甚至破坏投票结果；（2）、选民的个人隐私信息遭到泄露风险；（3）、电子投票系统数据丢失、文件损坏、官员受贿、供应商结党营私等各种丑闻；（4）、电子投票的结果完全由中心化控制，选民无法验证自己的投票结果是否正确。这些弊端也是制约了构建一个更加安全高效投票环境，所以这些问题也是急需解决的。

区块链技术具有的去中心化、数据公开可验证、防篡改和准匿名性的特性，而智能合约是运行在区块链上能实现特定交易功能的一段程序，将投票协议的智能合约运行在区块链上能实现特定的身份审核、选票审核、自我计票等逻辑功能，可以有效取代传统的第三方计票机构，为解决目前投票系统中遇到的安全问题提供了一个有效的解决方案。

本文以安全的电子投票方案为研究背景，以以太坊为代表的区块链技术为平台，采用智能合约和去中心化应用的方式，创新地实现了基于区块链的投票系统，本投票系统可以应用于现实的投票场景中，满足人们对安全的电子投票系统的需求。本文的主要工作内容分为如下几个部分：

（1）提出一种安全的多候选人电子投票方案

本文研究了国内外安全高效的电子投票协议，详细介绍了以以太坊为代表的区块链技术。结合当前安全的电子投票协议创新地提出了一种安全的多候选人电子投票方案。该方案是采用分布式EIGamal加密体制和两个零知识证明协议设计整个投票方案的执行流程，包括：离散对数知识证明协议和合法性知识证明协议。通过方案的安全性分析表明本文提出的投票方案的优点是满足电子投票基本的要求，对投票者的隐私保密性做到最大化的保护，同时还具备自我计票功能，可以取代第三方计票机构，并且在区块链技术的支持下具备投票结果公开可验证、防篡改等特性。

（2）设计实现投票协议的智能合约

在本文提出安全的投票方案基础上，使用Solidity语言编程实现了由一个合约文件组成的投票方案的智能合约。该合约控制整个投票过程，有创建提案，添加投票人，给予投票人投票权，投票，结束投票，开始投票，查看主席地址 ，查看主席名称，查看最终结果，查看提案 ，查看投票状态，查看总票数，查看总投票人，查看个人投票情况方法。

（3）