量子链的分布式自治协议

何华明

Defining the Blockchain Economy

QTUM.ORG

区块链协议与分叉

- 使用自治协议对升级、分叉进行管理的背景
- 影响分叉的因素
- 分布式自治实现的无分叉目标的可能性

区块链协议与分叉

- 使用自治协议对升级、分叉进行管理的背景

协议升级的分叉问题。对区块链应用的部署带来巨大的影响，目前主流的数字货币都曾经经历过软分叉和硬分叉对社区带来的巨大的伤害和影响，包括比特币、以太坊在内无一幸免。分叉作为一种网络和软件升级的手段，无可厚非，尤其在出现需要修复的关键漏洞时侯。

如何达成分叉的共识，并降低分叉对生态系统的影响是整个区块链行业都需要思考的问题。比如比特币网络扩容的问题，如何在比特币不同的扩容方案中做出大多数都认可的选择，以及如何达成这种共识，从而避免1MB 区块还是2MB区块长达2-3年的争论？

区块链协议与分叉

- 影响分叉的因素，区块链协议升级、安全修复需求，催生了各种分叉、Altcoin等。

算法、功能:
共识算法
加密算法
交易脚本
虚拟机
...

策略、参数类:
区块大小
出块时间
交易数量
Gas策略
...

关键漏洞
DAO
Parity多重签名钱包

区块链协议与分叉

- 分布式自治实现的无分叉目标的可能性：

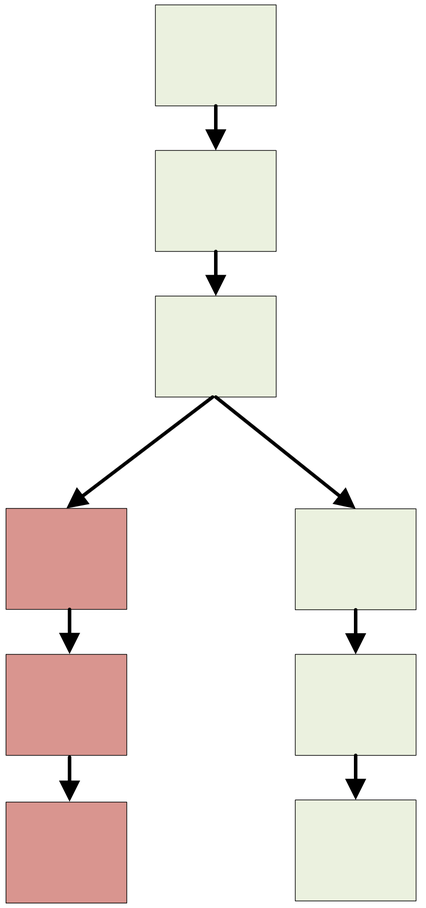
策略类是比较容易实现的：简单的状态变化容易掌控

更复杂的实现实现区块链网络的自动演化、无分叉：

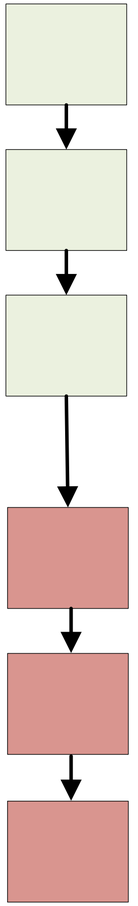
理论上可能性：{ APPLY(S,TX) --> S' or ERROR } vs 安全：
关键漏洞无法避免，修复最有效的还是硬分叉。

量子链采用分布式自治协议（ Decentralized Governance Protocol, DGP ），基本实现了对参数类的分布式自治

难度： 关键漏洞 >算法、功能 > 策略、参数



分叉升级



自治、无分叉

量子链分布式自治协议简介

- 量子链分布式自治协议设计目标
- 量子链分布式自治协议技术选择
- 量子链分布式自治协议实现
- 量子链分布式自治协议运行流程

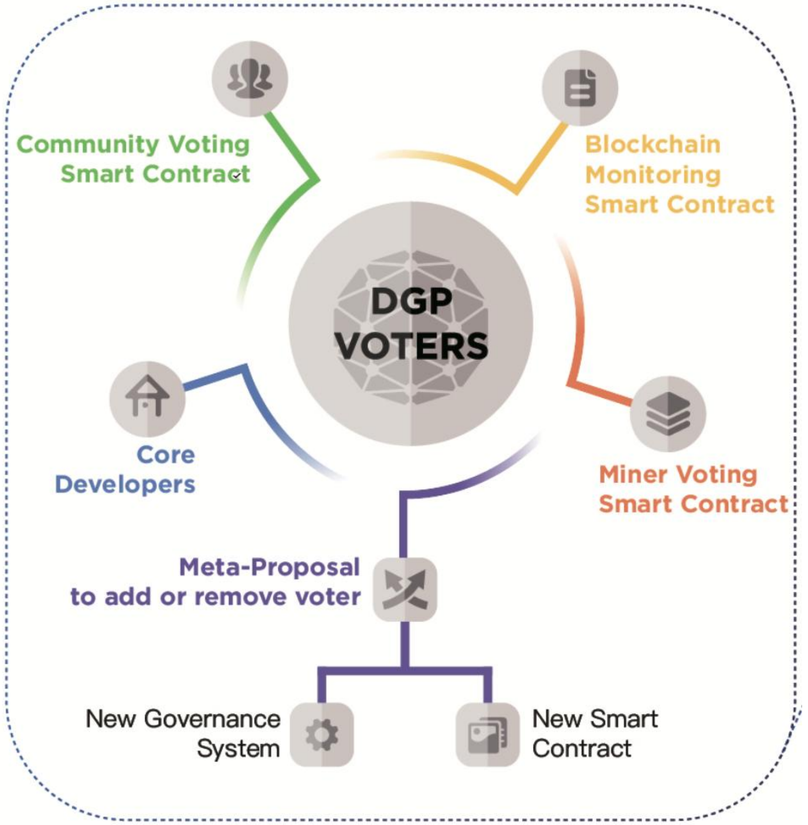
量子链分布式自治协议简介

- 量子链分布式自治协议设计目标：

  # 分布式自治：社区、管理者以及核心开发者等多方参与

  # 从提案到授权的治理机制：voting

  # 实现策略类参数治理：Self-regulating, self-modifying, and self-aware blockchain，无需更改软件版本实现区块链网络的自动升级和快速迭代



量子链分布式自治协议简介

- 量子链分布式自治协议技术选择：

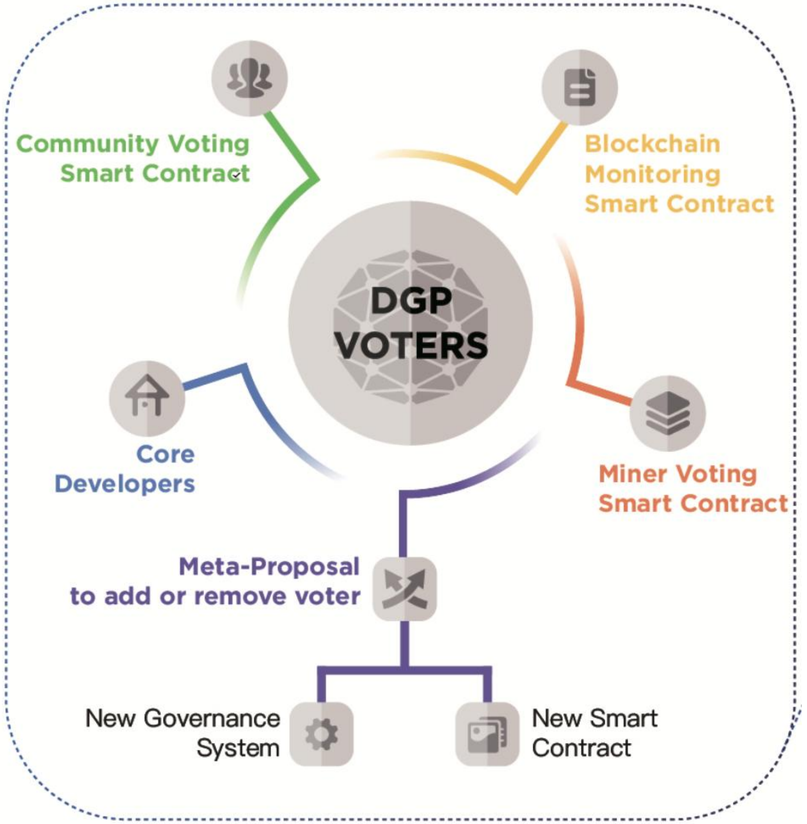
自治协议的实现需要某种可编程技术，UTXO和EVM提供这种特性。

（1）基于交易脚本

通过在交易脚本上实现协议逻辑，非图灵完备，实现比较复杂。

（2）基于智能合约

图灵完备的可编程能力，可以灵活实现复杂的逻辑。

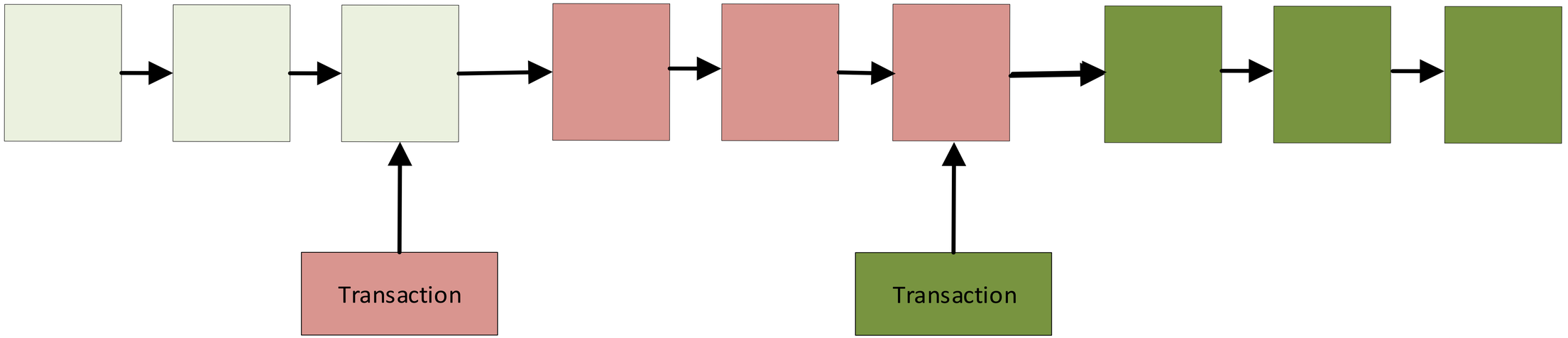


量子链分布式自治协议简介

- 量子链分布式自治协议设计实现：

自治协议核心逻辑的实现，是由一系列的智能合约组成，区块链核心代码在共识过程中执行协议的智能合约，获得当前的共识状态。

通过Transaction完成区块链网络的状态转换，升级无需区块链网络软件更新



APPLY(S,TX) --> S' or ERROR

量子链分布式自治协议简介

- 量子链分布式自治协议设计实现：

安全性考虑，智能合约本质是一段代码。理论上，采用了图灵完备的智能合约可以实现任意复杂度的协议设计，甚至是区块链的核心协议，如共识部分的代码等；权衡效率、安全等；

当前协议仅适用于在安全范围内对特定参数进行更改，同时对参数生效时间采取一定的时间限制。

支持如下参数：

Block size,

Min GasPrice,

Block GasLimit

Gas Schedule

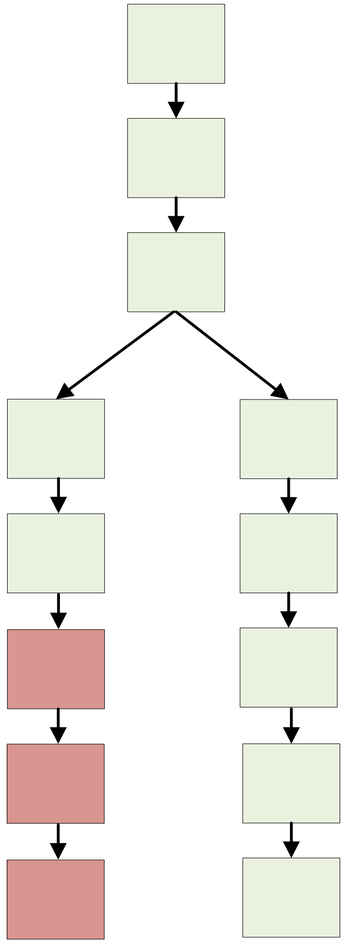
参数的更改提案通过后，在一定数量的区块后生效，避免产生可能分叉

量子链分布式自治协议简介

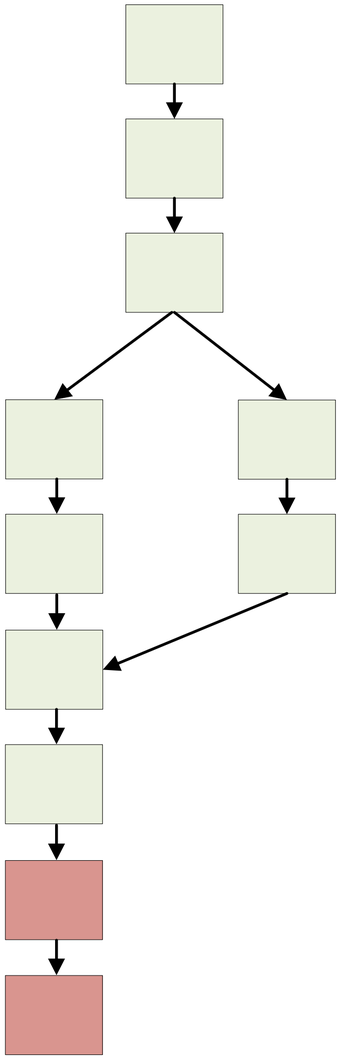
- 实现的安全性考虑：

仅适用于在安全范围内对特定参数

提案在一定数量的区块后生效，
避免可能的分叉



可能的分叉



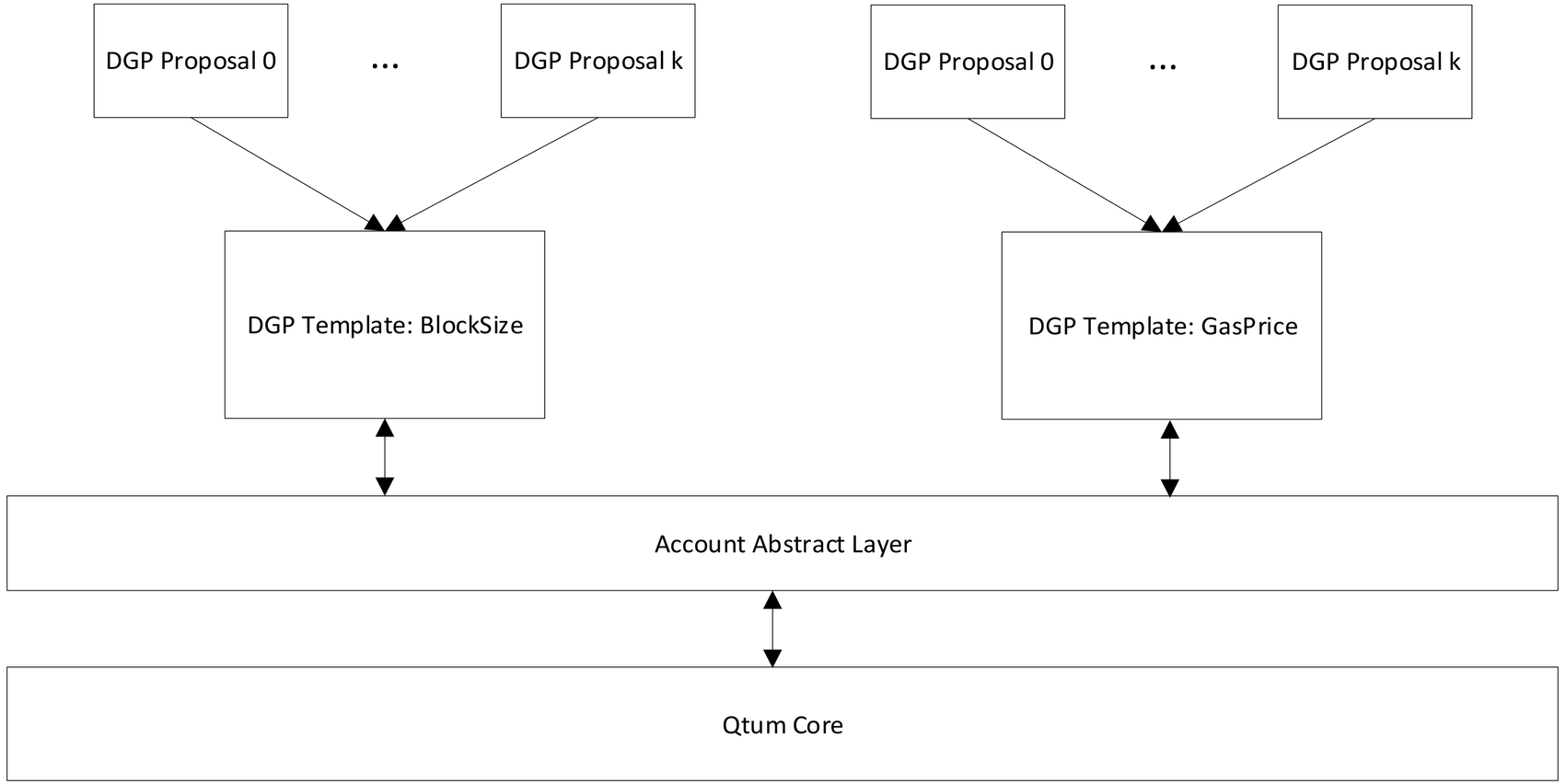
避免分叉

量子链分布式自治协议简介

- 量子链分布式自治协议的设计实现

分布式自治协议核心是由一系列的智能合约实现

Qtum core通过和合约交互执行共识结果



量子链分布式自治协议简介

- 量子链分布式自治协议的设计实现

  合约部分：

  # 设计了三大功能合约，包括管理参与者、提案投票、提案议题

  (1) meta-proposal智能合约管理多方参与：社区、管理者以及核心开发者等

  (2)治理提案的智能合约：对提案进行投票，完成决策

  (3)承载具体议题的提案采用智能合约编写，并公开部署在Qtum区块链

量子链分布式自治协议简介

- 量子链分布式自治协议的设计实现

合约部分：

创世块嵌入了常见的区块链参数治理的智能合约

每个治理的主题都由独立的智能合约控制(模板)，这意味着每个功能有独立的治理、授权机制以及内置限制条件

Block size，Min GasPrice，Block GasLimit，Gas Schedule

DGP合约具备自毁功能

在提案治理上发生意外时启动，治理参数退回到默认状态

量子链分布式自治协议简介

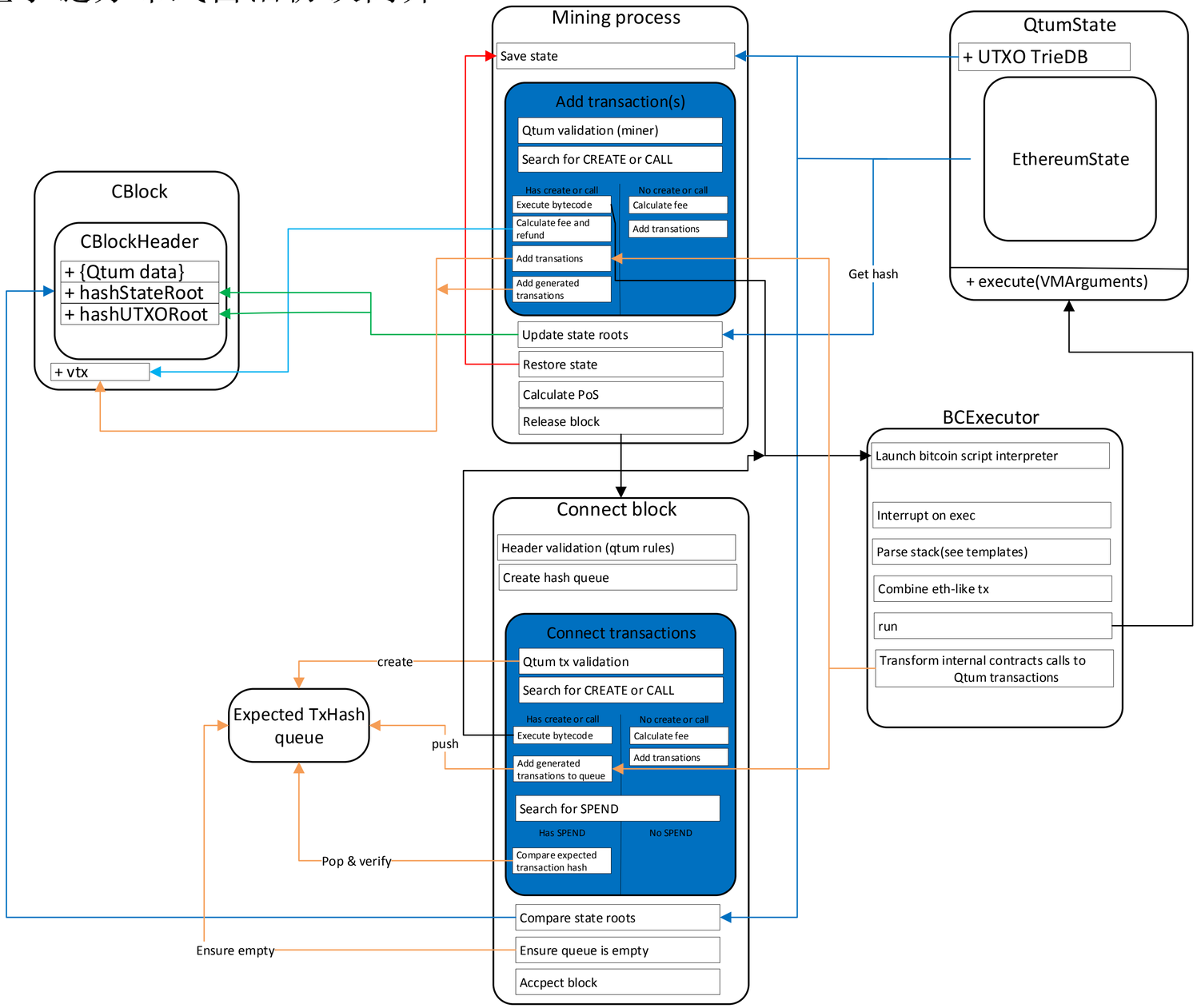
- 量子链分布式自治协议的实现：

Qtum core：

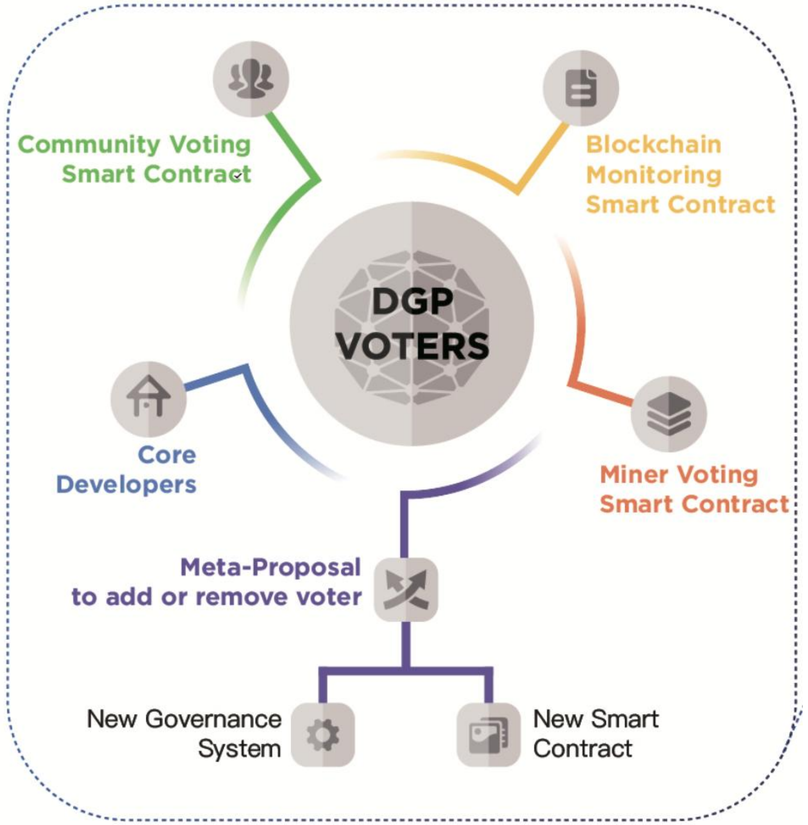
QtumState执行DGP合约，获取当前区块的DGP达成共识的参数值。

区块生成过程检查共识

区块验证过程添加到区块链结构



量子链分布式自治协议流程



1. 设计DGP参数更改提案。提案中需要明确新的参数值、实施变更的起始区块以及投票过程所对应的区块数量。
2. 向社区公布更改提案，并收集相关反馈；
3. 根据社区反馈，对更改提案进行相应调整；
4. 将最终版提案发送至DGP智能合约；
5. 投票立即启动；
6. 控制节点可以发送一笔交易给DGP智能合约，表达自己的赞成或者不赞成
7. 在投票过程中，若提案未获得足够投票或收到过多反对票，则该提案被否决，不执行任何修改；
8. 若提案得到足够同意票，则DGP分布式自治合约将提案中的相关数据存储在持久RLP储存器；
9. 钱包和区块链上的所有节点可以通过EVM定时检查合约执行结果从而判断是否有新的更改发生；
10. 更改提案禁止在500个区块内生效，以避免出现不必要的孤儿块或分叉；500个区块后钱包和节点执行参数更改。

量子链分布式自治协议简介

- DGP-template智能合约分析
- DGP-BlockSize智能合约分析

DGP-template智能合约分析

- DGP-template智能合约分析

该合约的主要通过下面的几个变量进行提案管理：

```
paramsInstance[] paramsHistory; // 保存所有提案参数历史
address[] adminKeys; // 管理人员密钥
address[] govKeys; // 治理人员密钥
proposals currentProposals; // 当前正在投票的提案状态
votesRequired activeVotesRequired; // 提案通过所需条件
```

合约使用变量currentProposals保存当前提案的投票统计，变量paramsHistory保存了所有通过提案的记录信息，保存的最近一个参数为当前区块链网络使用的参数。

- Meta-propsal管理功能

```
if(_type==0 || _type==1){
   if(tallyAdminVotes(currentProposals.keys[_type].votes)>=activeVotesRequired.adminVotesForManagement){
       if(isAdminKey(currentProposals.keys[_type].proposal) || isGovKey(currentProposals.keys[_type].proposal)) throw;

       if(_type==0) adminKeys.push(currentProposals.keys[_type].proposal); // elected

       if(_type==1) govKeys.push(currentProposals.keys[_type].proposal); // elected

       clearAddressProposal(_type);
      }
}
```

https://github.com/qtumproject/qtum-dgp/blob/master/dgp-template.sol.js

DGP-template智能合约分析

- 普通议题提案管理功能

```
if(_type==2){

  if(tallyAdminVotes(currentProposals.keys[_type].votes)>=activeVotesRequired.adminVotesForParams &&
     tallyGovVotes(currentProposals.keys[_type].votes)>=activeVotesRequired.govVotesForParams){
          if(paramsHistory.length>0 && paramsHistory[paramsHistory.length-1].blockHeight==block.number+1) throw;

         paramsHistory.push(paramsInstance(block.number+1,currentProposals.keys[_type].proposal;

          clearAddressProposal(_type);
   }
}
```

提案用于普通的区块链网络参数，当提案投票数大于在变量activeVotesRequired.adminVotesForParams设置的值时，提案的投票通过，并清空当前投票设置。通过的具体提案保存在paramsHistory。

https://github.com/qtumproject/qtum-dgp/blob/master/dgp-template.sol.js

DGP-BlockSize智能合约分析

- DGP-BlockSize智能合约分析

```
pragma solidity ^0.4.8;

contract blockSize{

uint32[1] _blockSize=[
2000000 //block size in bytes
];
function getBlockSize() constant returns(uint32[1] _size){
                return _blockSize;
}

}
```

用于产生普通具体的区块链网络参数议题，提案投票之前，部署到区块链上。通过把该智能合约的地址增加到提案管理部分，进行投票处理。

https://github.com/qtumproject/qtum-dgp/blob/master/blockSize-dgp-template.sol.js

Defining the Blockchain Economy

QTUM.ORG