

DAO 治理

- 什么是DAO
 - Decentralized Autonomous Organization 去中心化自治组织
 - 全新的组织形态: 去中心化, 自治, 可治理, 公开透明, Token激励
 - 围绕一个共同愿景 (纲领) 而组成的团体
 - DAO 的公开透明, 更容易吸纳贡献者, 形成大规模协作

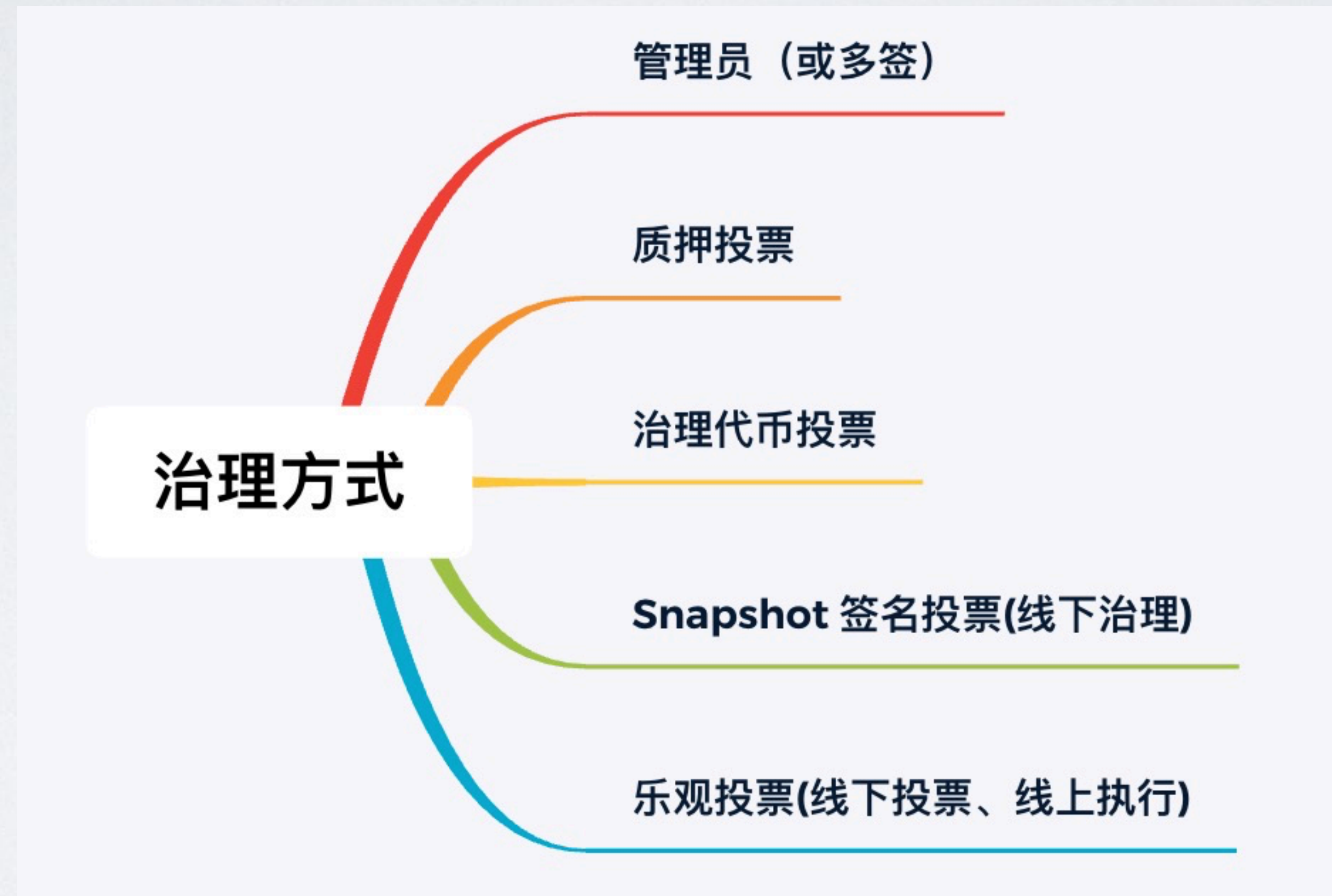
DAO 治理

- DAO治理： 怎样的方式来管理DAO
 - 没有权利中心， 决策由集体做出
 - 信奉“Code is law”，通过合约实现管理的代码化（根据代币投票）
 - Defi 带动了 DAO 的实践

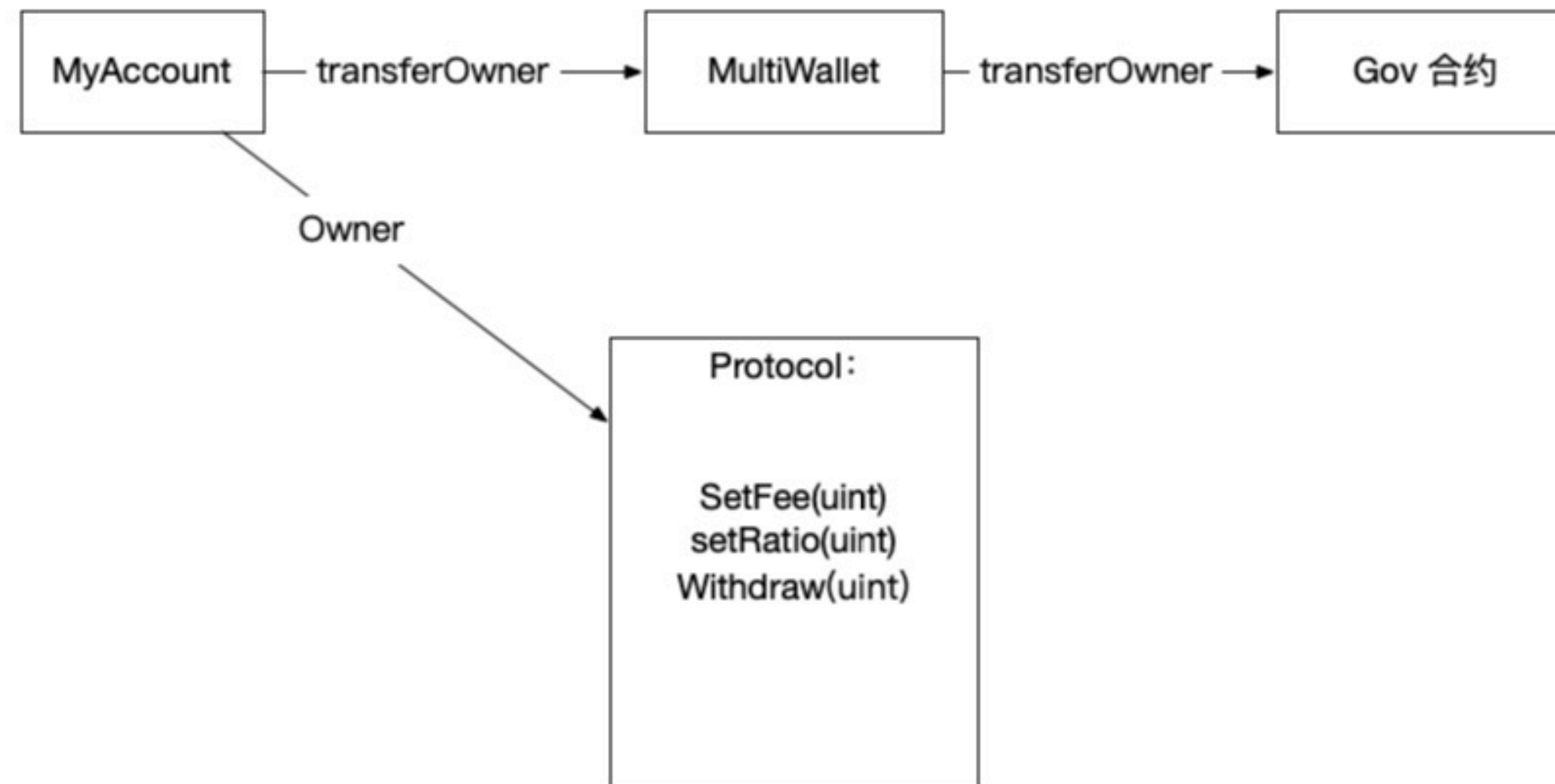
DAO 治理



DAO 治理



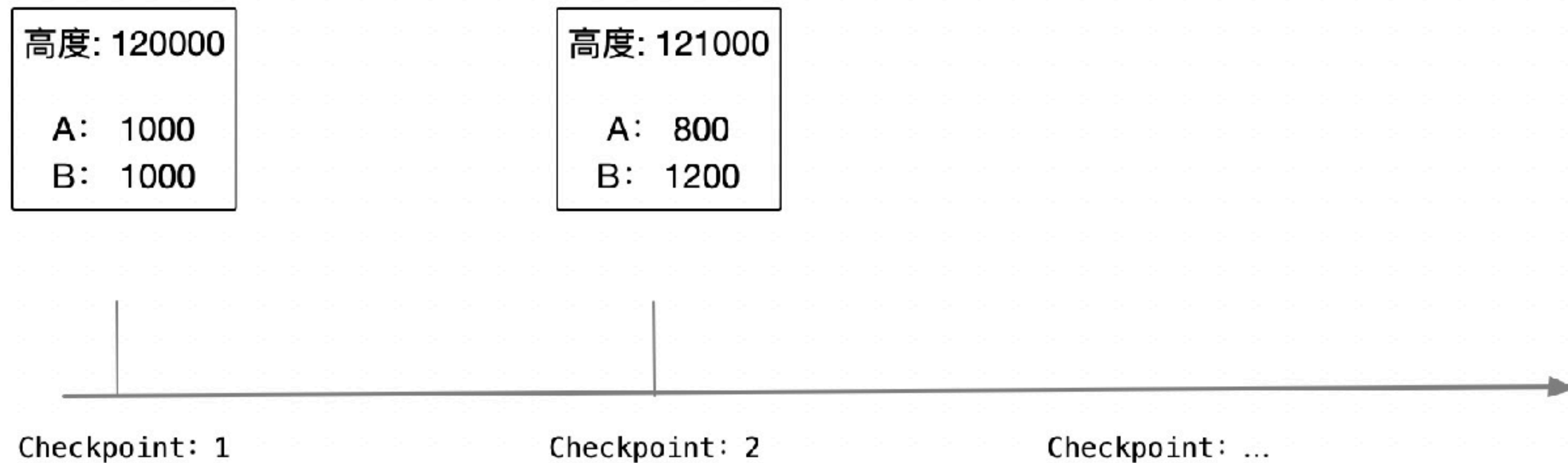
DAO 治理



如何计票?

问题 I: 用户会频繁转账, 如果防止转账后, 重复投票?

在每次转账的时候, 记录一个检查点, 检查点上写入区块高度, 及对应的票数。



投票时, 定位的提案区块高度在哪个检查点上, 从对应的检查点上, 获取其对应的票数。

如何计票?

```
struct Checkpoint {
    uint32 fromBlock;
    uint96 votes;
}

/// 每个检查点对应的投票数
mapping (address => mapping (uint32 => Checkpoint)) public checkpoints;

///每个账号有多少个检查点
mapping (address => uint32) public numCheckpoints;

function getPriorVotes(address account, uint blockNumber) public view returns
(uint96) {
    // 采用二分法
}
```


投票治理

- 提案：提交执行的动作（调用哪个合约的哪个方法）
- 统计投票数：怎样的方式来管理系统（协议）
- 执行提案

问题2：治理合约不知道未来有什么提案，如何执行所有可能的函数调用。

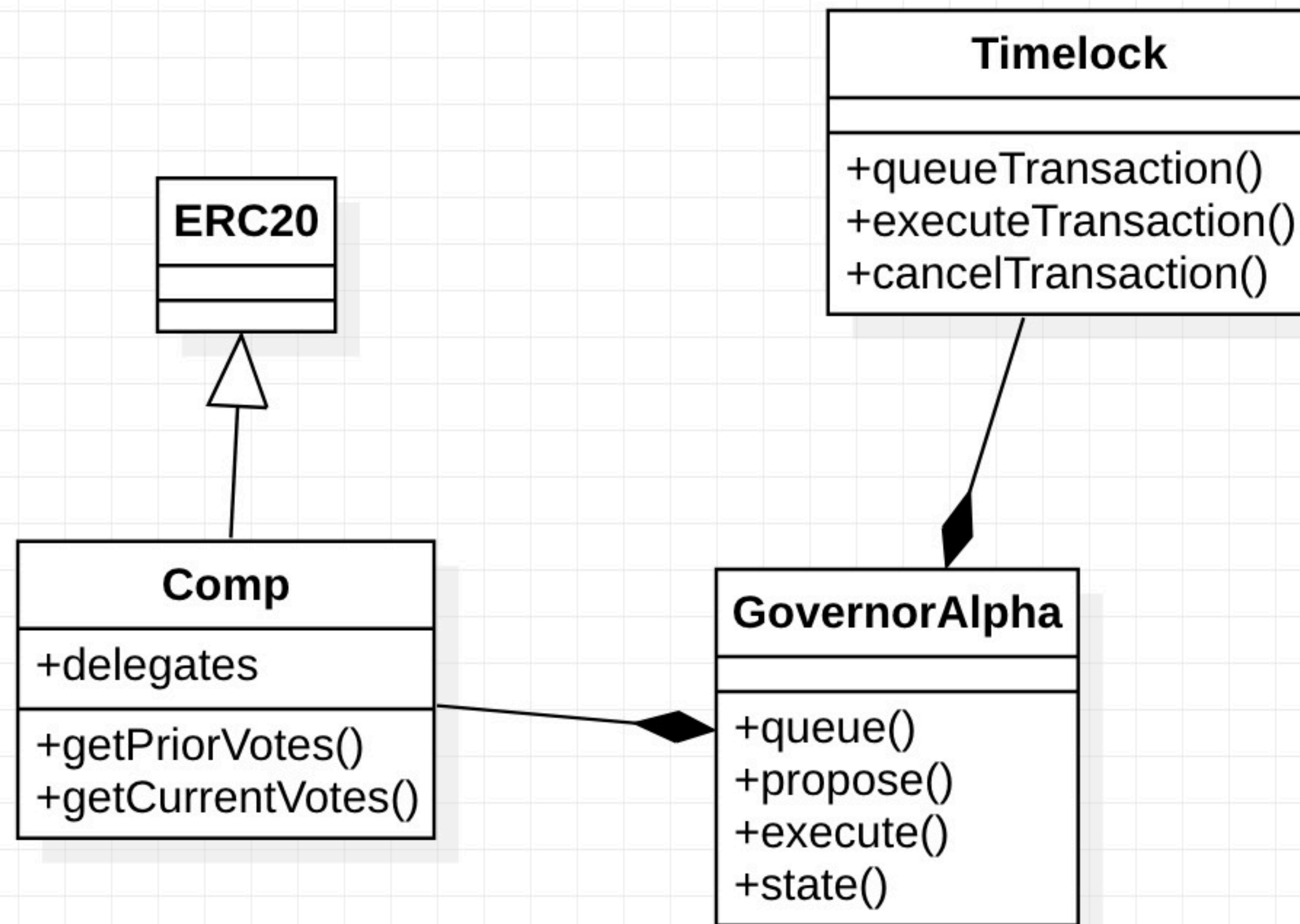
利用底层 call 调用，如调用合约的setFee(uint256) 方法：

```
bytes memory payload = abi.encodeWithSignature("setFee(uint256)", 10);
(bool success, bytes memory returnData) = address(target).call(payload);
require(success);
```


投票治理

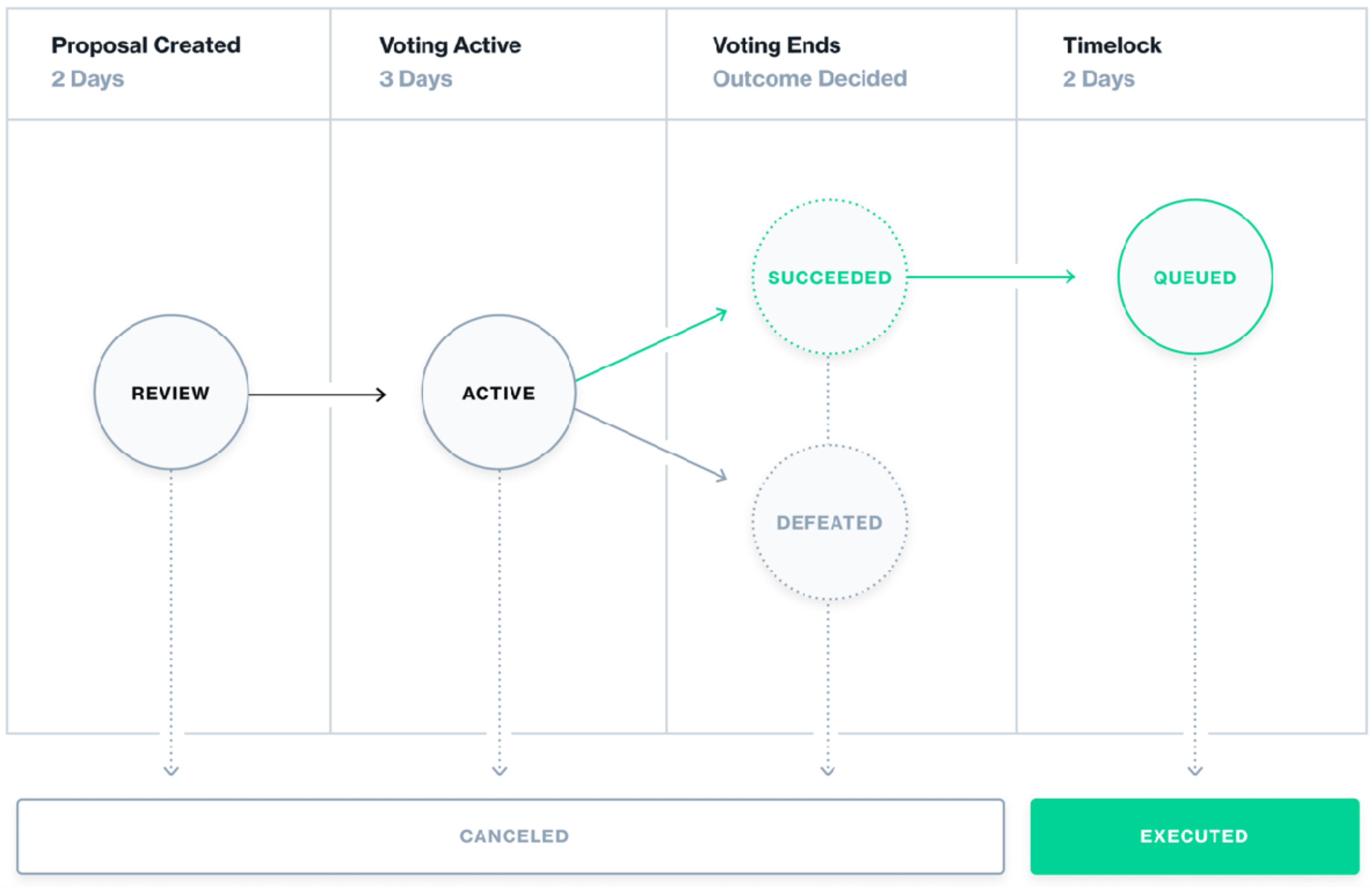
```
struct Proposal {  
    uint id;  
    address proposer;  
    uint eta;  
  
    address[] targets;  
    uint[] values;  
    string[] signatures;  
    bytes[] calldatas;  
  
    uint startBlock;  
    uint endBlock;  
  
    uint forVotes;  
    uint againstVotes;  
    bool canceled;  
    bool executed;  
  
    mapping (address => Receipt) receipts;  
}
```


投票治理



投票治理

- 治理过程



练习题

- 实现一个可以可计票的 Token
- 实现一个通过 DAO 管理资金的 Bank:
 - 管理员可以从Bank合约中提取资金withdraw()
 - 治理 Gov 合约作为管理员, Gov 合约使用 Token 投票进行治理
 - 通过发起提案从Bank合约资金

<https://decert.me/challenge/4cbe2544-6848-4881-b2f5-c4f291241621>