

基于区块链的可信共享办公及服务租赁平台

14级自动化 丘永春 指导老师 马小峰

CONTENTS

| | | |
|-------|---|-----------|
| 新增内容 | 1 | |
| | | 2 研究背景及意义 |
| 区块链特点 | 3 | |
| | | 4 目标及方法 |
| 实现方案 | 5 | |
| | | 6 实现与测试 |
| 创新展望 | 7 | |

1 新增内容

性能测试

- 使用go语言的并发和同步机制
- 并发机制
 - 函数创建为goroutine时，将被视为一个独立的工作单元
 - goroutine与操作（OS）系统线程之前的映射关系为
m:n
其中 $m > 1$ ， $n \geq 1$ ，可由一个操作系统线程映射到多个goroutine
 - 每隔0.002秒发送一个交易请求，即每秒500次请求
 - 设置发送程序运行时间为100s

```
go func() {  
    for {  
        select {  
            // 每0.002秒发送一次请求  
            case <-time.After(20000000):  
                // Post()为http post类型的请求发送函数，  
                go Post()  
        }  
    }  
}()
```

每隔0.02s发送一次请求

```
// 设置程序运行时间  
time.Sleep(100 * time.Second)
```

设置程序运行时间为100s

性能测试

• 同步机制

- 使用消息通道（channel）同步消息
- 提供了异步进程间同步数据的机制。
- 副goroutine之间的数据通信
- 主goroutine控制运行时间
- 通道send用于同步请求发送次数
 - 变量i统计发送次数
- 通道quit用于同步请求成功执行次数
 - 变量j统计发送次数

```
// 声明两个共享数据通道，用来统计发送请求和请求成功执行次数，  
var quit chan int = make(chan int, 100)  
var send chan int = make(chan int, 100)
```

创建通道同步数据

```
// 执行HTTP_POST()函数时往通道发送消息  
send <- 0
```

发出请求时将消息发送至通道

```
// 将整型数据0作为成功访问的信号塞入通道  
quit <- 0
```

请求执行成功时消息发送至通道

```
go func() {  
    for {  
        <-send  
        j++  
        fmt.Printf("发送交易次数: %d 次\n", j)  
    }  
}()  
go func() {  
    for {  
        <-quit  
        i++  
        fmt.Printf("交易成功执行次数: %d 次\n", i)  
    }  
}()
```

统计发送请求的次数和交易成功执行次数

性能测试结果

- 硬件水平
 - Linux内核 centos 7操作系统，2CPU，4core（每个CPU两个core）

• 右图测试过程->

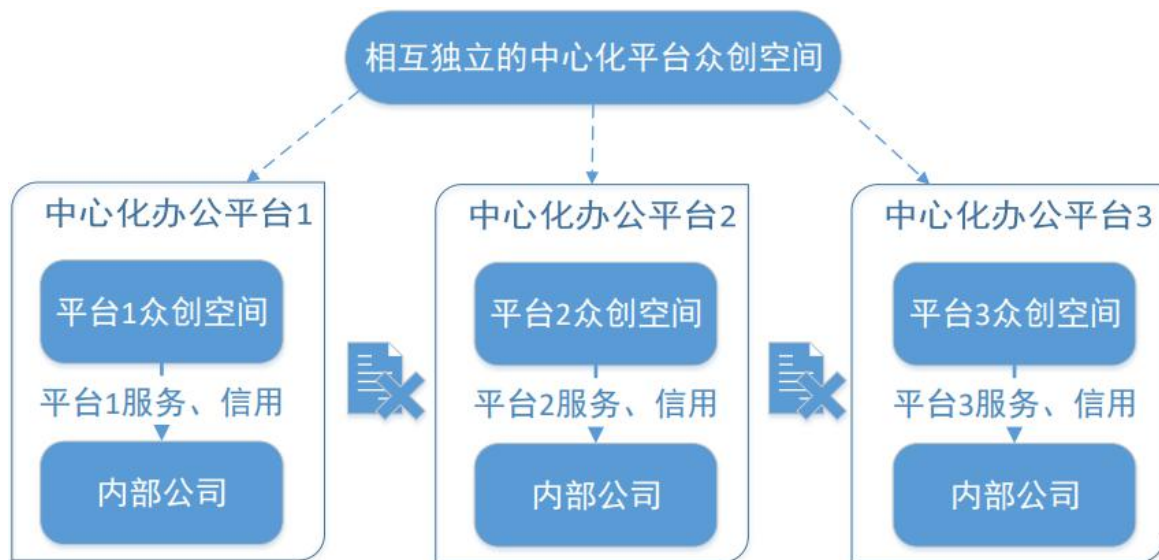
```
交易成功执行次数: 21125 次
发送交易次数: 49994 次
POST请求:创建成功 {"success":false,"message":"Failed to order the transaction. Error code: und
efined"}
POST请求:创建成功 {"success":true,"message":"8ce3b95d2d8e5701c4d821072b766be32db18a6e63a24475c
2470ad3a7014741"}
交易成功执行次数: 21126 次
发送交易次数: 49995 次
发送交易次数: 49996 次
POST请求:创建成功 {"success":true,"message":"70138ccce70f62725932ee816e60d2e385a44595815d1c22c
d8f6710f8647e82"}
交易成功执行次数: 21127 次
发送交易次数: 49997 次
POST请求:创建成功 {"success":true,"message":"c3fd3da3c96a5ef9c76e208e85c4ffdee5e6530e9fc6394e9
963daa37e9a1064"}
交易成功执行次数: 21128 次
POST请求:创建成功 {"success":false,"message":"Failed to order the transaction. Error code: und
efined"}
```

• 右表测试统计->

| 测试函数 | 时长 | 间隔 | 发送数据 | 成功处理数据 | 成功率 | 每秒成功处理交易数 |
|--------|------|--------|--------|--------|--------|-----------|
| regist | 100s | 0.002s | 49997条 | 21128条 | 42.25% | 211.13tps |
| update | 100s | 0.002s | 49997条 | 22367条 | 44.74% | 223.67tps |

2 研究背景意义

现有共享办公平台模型分析



中心化的共享办公平台模型

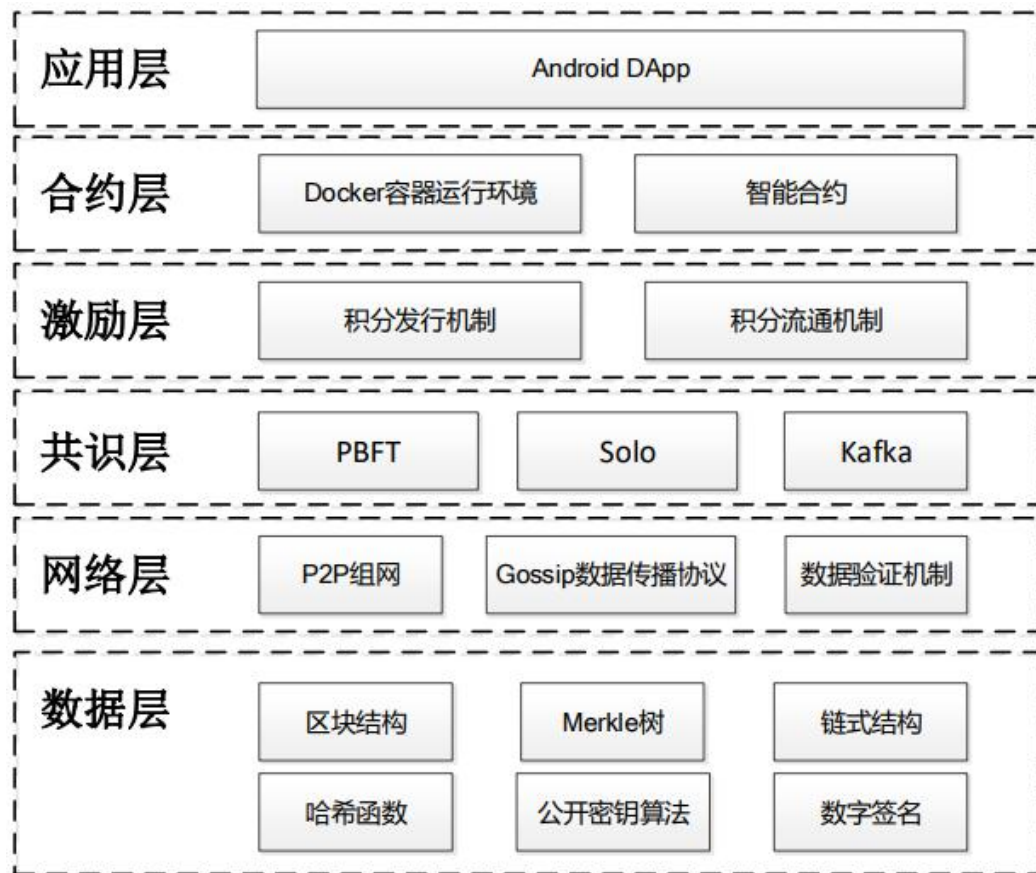
- 数据解释权归中心化机构所有
 - 可篡改数据
- 共享不彻底
 - 没有进一步整合信息资源与服务资源
- 信息不对称、不透明
 - 中心化机构的存在
- 信任与价值没有互相传递
 - 单个平台内部公司或个人积累的信任难以被其他平台认可

中心化平台

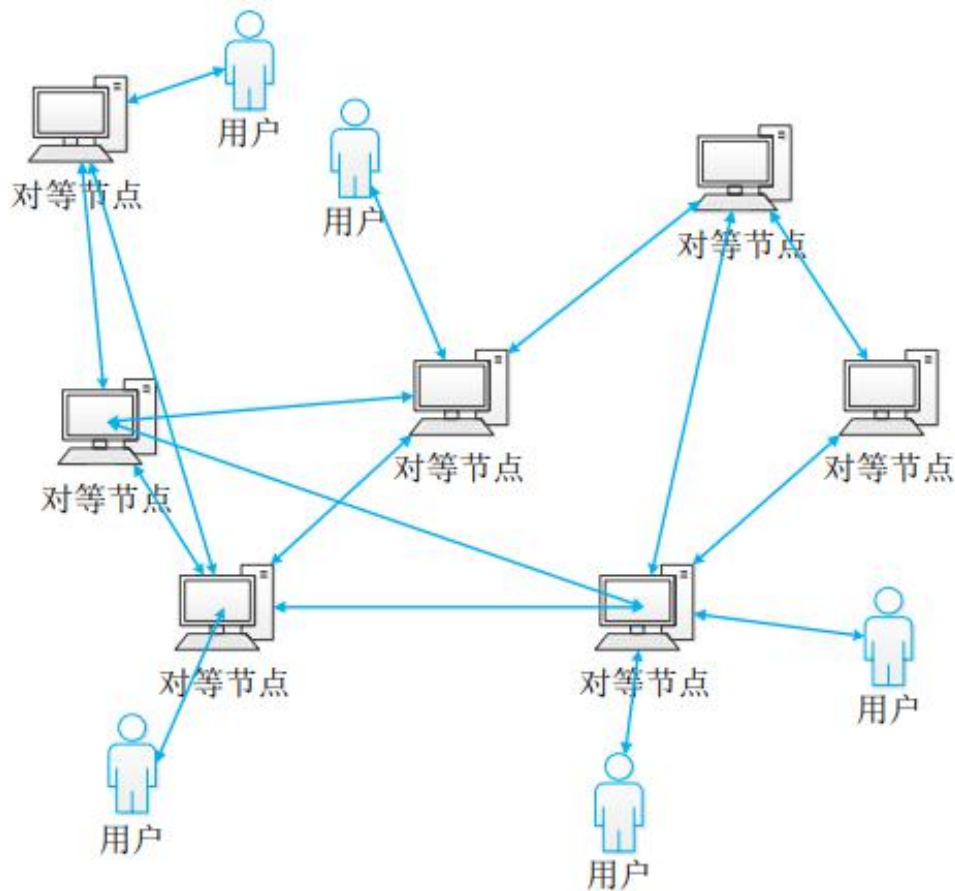
- 数据可篡改、不透明、不信任

3 区块链特点

区块链特点



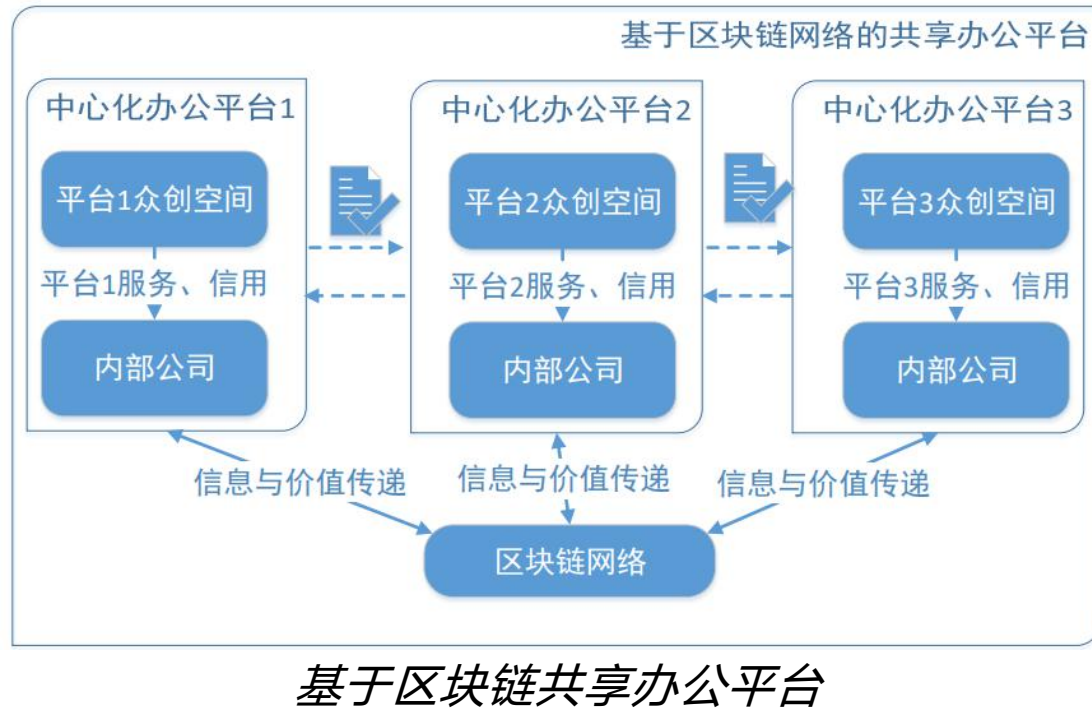
层次结构



网络拓扑结构

- 计算机过去20年多年成果：计算机p2p网络、数据库技术、分布式系统、分布式共识算法、计算机密码学
- 带来的特点：分布式存储、全网共识、不可篡改、数据公开透明的共享账本

区块链共享办公平台模型分析



本项目中区块链用途：

- 去中心化、分布式、不可篡改的可信共享账本

- 根据业务需要建立多方共识、数据共享、不可篡改、可追溯的分布式账本
 - 单个机构记账转为多方共同记账
 - 多方共同维护账本
- 传递信息、信任与价值
 - 数据可信，作为信任衡量的依据
 - 分布式存储、块链式数据结构
 - 资产确权，价值传递
- 无需强大的中央机构进行信用背书

4 目标及方法

研究目标

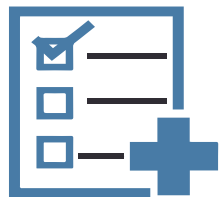
- 目标:

- 消除平台之间、平台与用户间的不信任关系
- 建立一个可信的共享办公及服务租赁平台



去中心化的资源共享的租赁平台

消除或削弱各平台间、众创空间之间的资源与信息壁垒



可信的信用评价体系

基于用户行为的积分作为信用的衡量在各平台间传递



引入多元增值服务

使得信用和积分更具有价值和使用场景



打造闭环生态

积分在共享办公平台内部生态流通，用户服务消费

研究方法

- 分布式账本实现共享数据

分布式存储、分布式共识，使得每个节点记录相同的账本数据，从而达成节点之间的互信。

- 通道技术隔离数据

使用身份验证机制，确定各节点的真实身份之后便可以通过共享的通信通道。一组有商业合作关系而建立的实体可加入通道共享数据。其他参与方无权访问

- 智能合约实现业务功能

区块链网络中约定了参与方的协议内容。参与方都对合约的内容达成一致，并遵守合约执行的结果。合约独立运行，自动化执行。

- 并发机制实现性能测试

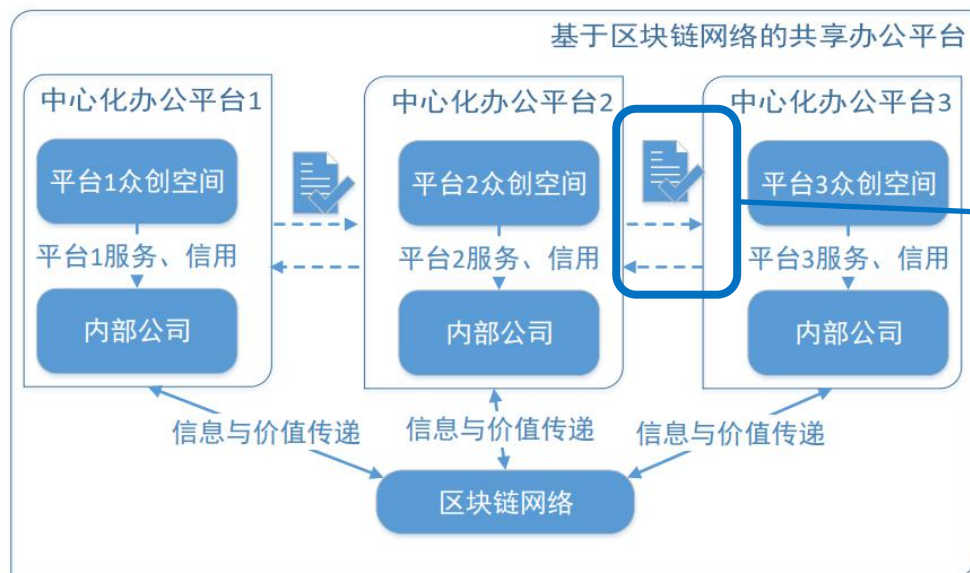
区块链的交易性能是区块链项目最大的局限于瓶颈，本设计将通过并发访问控制机制和异步进程之间的数据同步机制实现对区块链网络的性能测试。

5 实现方案

p2p网络部署

ec1d23dc9129
example.com
7bb46831fc72
example.com
a0ff9114886a
example.com
f19f52b070c8
example.com
73b021ad2612
mple.com
6b63b418422e
36225ddb138e

hyperledger/fabric-peer:x86_64-1.0.0
hyperledger/fabric-peer:x86_64-1.0.0
hyperledger/fabric-peer:x86_64-1.0.0
hyperledger/fabric-peer:x86_64-1.0.0
hyperledger/fabric-orderer:x86_64-1.0.0
hyperledger/fabric-ca:x86_64-1.0.0
hyperledger/fabric-ca:x86_64-1.0.0



- 构建去中心化的资源共享与租赁平台
- 工具与方法
 - 数据存储
 - levelDB与CouchDB数据库技术
 - 分布式网络
 - docker容器模拟p2p对等网络
 - 分布式共识技术
 - Solo或Kafka算法排序, 由orderer排序服务节点统一全网唯一的交易内容及数据, 分发给Peer对等节点
 - 证书生成技术
 - 通过证书文件、秘钥确认网络成员身份
 - 通道(Channel)技术
 - 设置具有记录同一组数据权限的节点
- 要点:
 - 通过docker构建了去中心化的分布式网络
 - 生成证书确定网络成员关系
 - 使用通道技术规定了成员间数据访问权限

智能合约技术建立信用评价体系

```
type User struct {
    ID          string `json:"id"`
    Password    string `json:"password"`
    Role        string `json:"role"`
    Balance     float64 `json:"balance"`
    CrdPoint    int    `json:"crePoint"`
    FunPoint    int    `json:"funcPoint"`
}
```

```
user.FunPoint = user.FunPoint + service.FunPoint
user.FunPoint += 5
bytes, _ = json.Marshal(user)
balance = strconv.FormatFloat(user.Balance, 'E', -1, 64)
crdPoint = strconv.Itoa(user.CrdPoint)
funPoint = strconv.Itoa(user.FunPoint)
```

```
8de14f4dd8fe    dev-peer1.org1.example.com-publicservice-v0
                dev-peer1.org1.example.com-publicservice-v0
62adf56ba04b    dev-peer0.org1.example.com-publicservice-v0
                dev-peer0.org1.example.com-publicservice-v0
b5eab8f6134e    dev-peer1.org1.example.com-register-v0
                dev-peer1.org1.example.com-register-v0
67cfb09fdef5    dev-peer0.org1.example.com-register-v0
                dev-peer0.org1.example.com-register-v0
```

- 两种积分：信用积分、功能积分
 - 信用积分代表信用
 - 功能积分代表劳动付出
- 积分发行方式：
 - 信用积分
 - 注册原始积分、完成约定、工位/服务好评
 - 扣除途径：未履行约定、服务差评、破坏物品
- 智能合约一旦执行便是独立的沙箱环境，与宿主机隔离，安全隔离。

使用积分建立信用评价体系

- 两种积分：信用积分、功能积分
 - 信用积分代表信用
 - 功能积分代表劳动付出

积分的发行



信用积分

- 发行途径：注册积分、完成约定、工位/服务好评
- 扣除途径：未履行约定、服务差评、破坏物品

功能积分

- 发行途径：注册积分、发布工位、使用工位



积分的流通

信用积分

- 不可以转移

功能积分

- 可以转移

积分的使用

信用积分

- 高级服务优先权：平台运营培训、律法培训、融资机会、接触优质公司机会

功能积分

- 发布任务、租赁工位以及各种生活服务，如外卖、打车、电影票、其他任务

智能合约互相调用实现多元增值服务与评价功能

| CONTAINER ID | IMAGE |
|----------------------------------|---|
| f4a00d63d34e | dev-peer0.org1.example.com-comment-v0 |
| rg1.example.com-comment-v0 | |
| 665e13b6fcfa | dev-peer1.org1.example.com-comment-v0 |
| rg1.example.com-comment-v0 | |
| 8de14f4dd8fe | dev-peer1.org1.example.com-publicservice-v0 |
| rg1.example.com-publicservice-v0 | |
| 62adf56ba04b | dev-peer0.org1.example.com-publicservice-v0 |
| rg1.example.com-publicservice-v0 | |
| b5eab8f6134e | dev-peer1.org1.example.com-register-v0 |
| rg1.example.com-register-v0 | |
| 67cfb09fdef5 | dev-peer0.org1.example.com-register-v0 |

```
Query Response:{\"id\":\"a1001\",\"password\":\"123456\",\"role\":\"1\",\"balance\":1500,\"crePoint\":1000,
2018-06-19 13:47:13.713 UTC [register] Info -> INFO 012 ##### Register Invoke #####
update
[a1001 1.5E+03 1005 1005]
1500
```

```
strGrdPoint = strconv.Itoa(user.CrdPoint)
strFunPoint = strconv.Itoa(user.FunPoint)
fmt.Printf("+++++grade:%s;funPoint:%s;balance:%s\n",user.CrdPoint,user.FunPoint,user.CrdPoint)
trans = [][]byte{[]byte("update"), []byte(userID), []byte(string(
response = stub.InvokeChaincode("register", trans, "mychannel")
```

- 项目设计了三个合约完成不同类型功能：



register:提供与用户注册、修改用户相关信息的功能"



publicService:与发布工位/多元服务、申请服务相关（同时去调register合约以改变积分）"



comment:与评论服务相关（同时去调register与publicSrvice合约改变积分、添加评价记录）"

- 使用ChaincodeStubInterface接口实现互相调用

- 实现了智能合约间的相互调用，完成复杂业务

使用Android APP构建项目DAPP

- 页面activity主要生命周期：
 - 创建、启动、启动完成、停止和销毁
- 数据绑定
 - XML属性id设置唯一标识符
 - Java对象findViewById()方法获取属性值
- c++后端使用socket与前端Java通信
- 使用有限状态机分发请求
- 模块化、业务逻辑清晰

```
<activity
    android:name=".MainActivity"
    android:label="主页" >
    <intent-filter>...
</intent-filter>
</activity>
<activity
    android:name=".LoginActivity"
    android:label="登录"
    android:theme="@style/myTheme" >
</activity>

<EditText
    android:id="@+id/ETRentTime"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1.3"
    android:textColor="#000000"
    android:background="@layout/shape_edit"
    android:ems="10" />
```

```
servaddr.sin_port = htons(50000); //绑定TCP端口

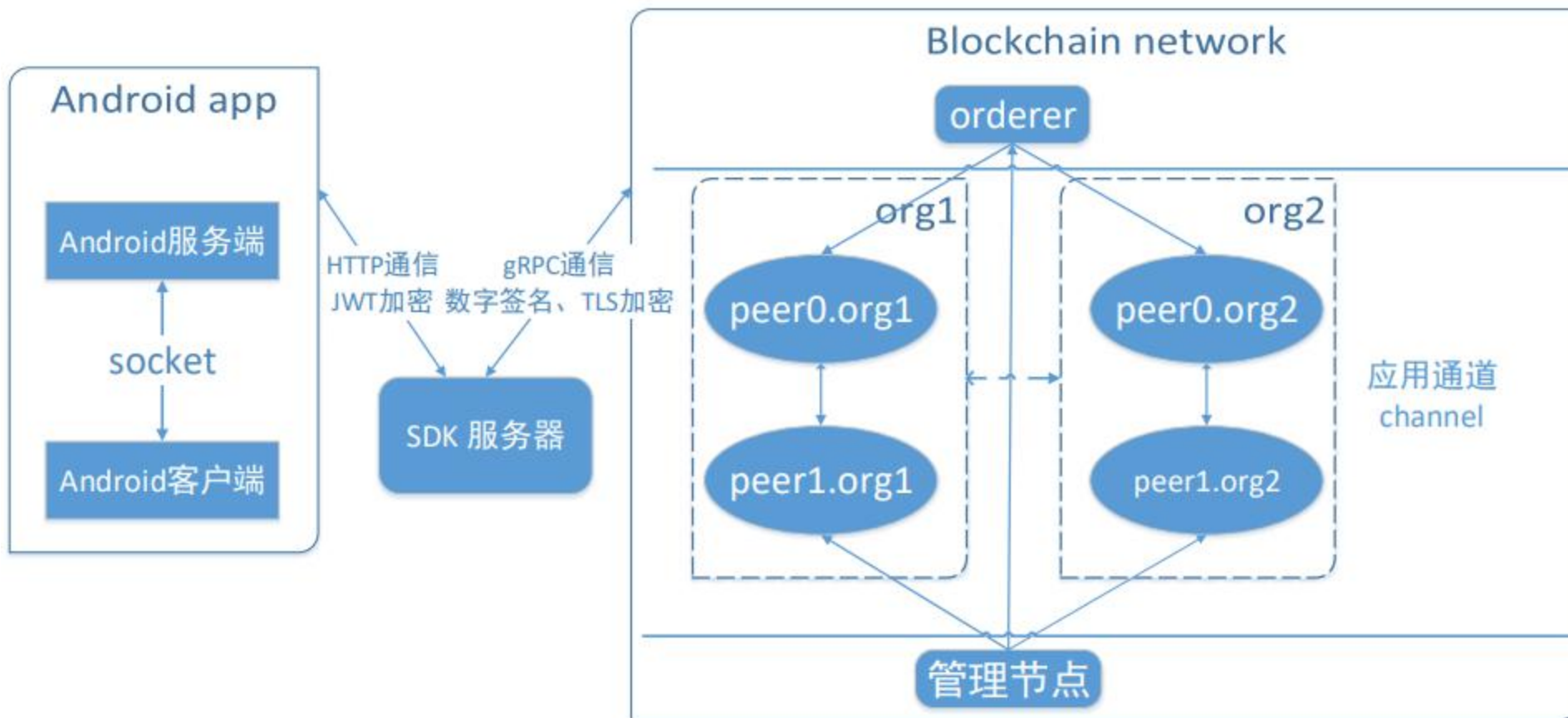
// 置端口重用
ret = 1;
setsockopt(listenfd, SOL_SOCKET, SO_REUSEADDR, (void *)&ret, sizeof(ret));

sock = new Socket();
sock.connect(new InetSocketAddress(serverIP, serverPort), timeOut);
```

```
switch(input["operate"].asInt())
{
    /* 用户操作 */
    case CLIENT_USER_SIGNUP: //注册
        bc_user_signup(input, output);
        break;
    case CLIENT_USER_SIGNIN:
        bc_user_signin(input, output);
        break;
    case CLIENT_USER_CHGPWD:
        bc_user_chgpwd(input, output);
        break;
    case CLIENT_USER_REVISE:
        bc_user_revise(input, output);
        break;
    case CLIENT_USER_QUERY:
        bc_user_query(input, output);
        break;
}
```

6 实现与测试

项目整体架构



合约逻辑测试

```
注册用户a1002 //////////////////////////////////
传入的6个参数的含义与注册a1001相同，其中第三个参数为2，表示公司或个人
注册用户a1002返回结果为操作状态和交易ID
{"success":true,"message":"e1d60dce9b772cb51da5cbbcbdba69b4d6bf754249ac842def81eeb35c9bee01"}

查询用户a1002 //////////////////////////////////
args传入的1个参数为：用户ID |
返回结果为查询结果为用户a1002的注册信息
{"success":true,"message":{"id":"a1002","password":"123456","role":"2","balance":1000,"crePoint":1000,"funcPoint":1000}}

用户a1001发布服务b1001(owner is a1001) //////////////////////////////////
args传入的8个参数为：服务/工位ID | 发布者ID | 类型（1为工位，2为其他服务） | 类型2（服务名称）
| 价格（float64） | 描述字段 | 地点 | 工位容量
{"success":true,"message":"e41f3a0bf45f86220f57e5c6d6e04402fe12392b870409690a972a76de8044e1"}

查询服务b1001 //////////////////////////////////
args传入的1个参数为：服务/工位ID
{"success":true,"message":{"id":"b1001","ownerId":"a1001","type":"1","name":"日租办公室","function":"","money":500,"funPoint":0,"description":"面积大，位置好，设备齐全。","palce":"兆润领域商务广场11楼","capacity":8,"state":"0","top":false,"topPoint":0,"userId":"","comment":null}}

用户a1002申请服务b1001 //////////////////////////////////
args传入的2个参数为：服务/工位ID | 申请者ID
{"success":true,"message":"e945e1c5fe2ad1c6d486e67cfd65f40cdc4e5c014c0f280081eaf2c63b28b31a"}

查询服务b1001 //////////////////////////////////
args传入的1个参数为：服务/工位ID
{"success":true,"message":{"id":"b1001","ownerId":"a1001","type":"1","name":"日租办公室","function":"","money":500,"funPoint":0,"description":"面积大，位置好，设备齐全。","palce":"兆润领域商务广场11楼","capacity":8,"state":"1","top":false,"topPoint":0,"userId":"","comment":null}}

用户a1002评价服务b001（属于a1001） 评论使得功能积分+5////////////////////////////////
args传入的4个参数为：使用者ID | 服务ID | 评级（整型1~5） | 评论
{"success":true,"message":"45fcc6f43fa869295627a8e4dc17ab9a8fffb443587ea9ab071efc7469692f1"}
```

Android DApp逻辑测试



首页界面

12:48 15:37

用户名* 测试用户

设置密码*

确认密码*

公司信息 (*为必填)

公司名称* 上海同济

联系电话* 15221584563

Email 15221584563@163.com

公司地址 曹安公路4800

公司法人* 测试

收入* 万元

注册

填写注册信息



用户中心



生活服务

Android DApp逻辑测试



租用工位



使用点评



点评成功



评价记录

[SDK测试视频](#)


[Android测试视频](#)

性能测试

- 开头所示.....

性能测试分析

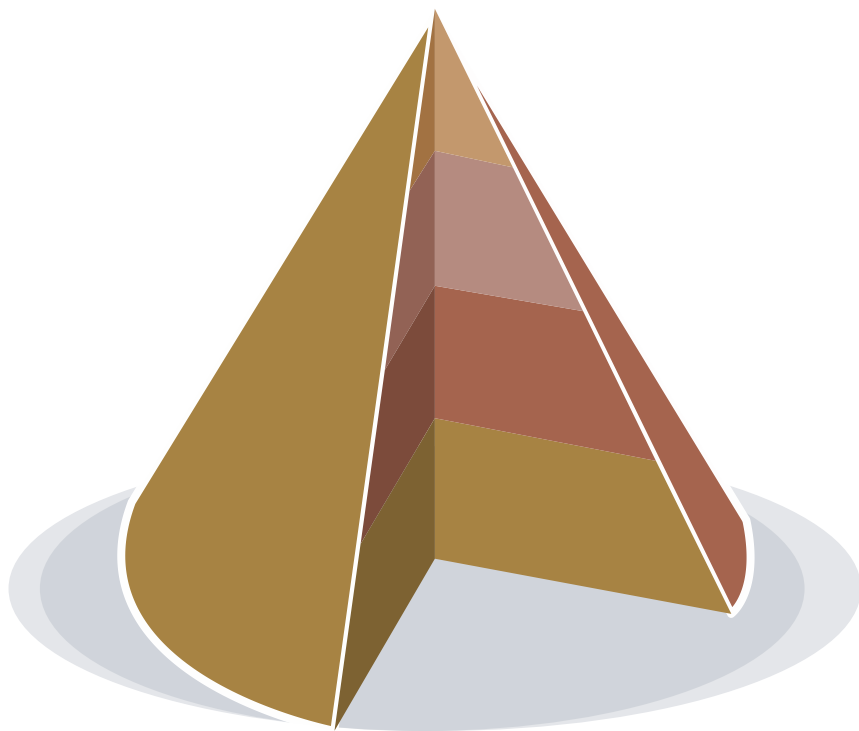
- (1) 背书节点检查节点身份进行交易提案背书（ECC签名），模拟测试结果。
- (2) 背书节点将经过背书的交易提案转发给排序节点，排序节点将模拟的交易统一排序打包成区块。
- (3) 对等节点将从排序节点获得同步的数据区块，记录在本地的账本中。
- 其中对获得足够多数背书节点的背书、排序节点统一将交易排序和各对等节点同步数据区块是相比较于中心化系统需要额外消耗的时间。



7 创新与展望

创新归纳

基于区块链的共享办公平台的创新点



重组业务关系

- 打通信息孤岛与资源壁垒，增强信息与资源流通效率



打造生态闭环

- 日常生活、工位租用、商业合作、投融资、广告等，



创新盈收模式

- 改工位出租为主的盈利模式，变为一个提供多元化增值服务的平台



建立评价体系

- 建立一个完整、不可篡改的评价体系，鼓励积极诚信行为

基于区块链的共享办公平台不足与展望

进一步去中心化

- 封装gRPC协议接口、获取身份认证证书，与区块链网络交互的SDK可除去

性能不足

- 研究更高效的签名、共识算法
- 硬件设施提高

IPFS存储

- 加入IPFS存储协议。实现可寻址、不可篡改、对等网络的内容分发与存储。

央行数字货币

- 央行数字货币推出后可尝试使用央行数字货币进行支付

物联网、人脸识别

- 嵌入智能芯片，工位自动完成计时、收费等功能。硬件或DApp导入人脸识别接口，人脸的特征值可存在区块链存证。

去中心化后的监管问题

- 数据加密传输难以监控广告投放的监管的思考



THANKS

请各位老师批评指正！

多元增值服务



共识过程

