# 《中间件技术课程报告》

姓名：郝俊生

学号：2018104185

## 基于区块链的学分鉴定系统

### 背景

比特币网络已经稳定运行了将近九年的时间，从而证明了作为比特币底层支撑技术，即区块链技术具有十分广阔的应用远景。2014年前后，区块链技术逐渐引起了人们广泛的关注。基于区块链技术还将可能制做一个相互信任的基础网络设施。

在传统的高等教育领域，存在着以下几点问题。第一：学生的学分和成绩记录系统不完善，造成用人单位难以获取到所需的全部的有效信息；第二：在就业时，又可能存在学历造假等问题，用人单位验证需要花费巨大的精力。随着区块链技术渐渐进入人们的视线，不断有各种应用在多种的场景落地。结合区块链技术的安全性，不可篡改性，区块链技术在教育领域的落地是必然的。所以利用区块链技术可以建立方便、可信的证书体系，可以有效的降低求学、求职和人才雇佣的成本。因此可以利用区块链技术构造安全可靠的数字证书体系，从而可以明显的降低学生的求职成本和用人单位的雇佣成本。

学分鉴定系统是基于开源项目Hyperleder Fabric来实现的，在APP上收集到数据后，使用加密算法加密后，通过网络上传到服务器，服务器把信息广播到区块链网络中等待共识，达成共识后把结果存在区块链网络中，用人单位可以根据assetID查询成绩的真实性。Hyperleder Fabric项目是首个面向企业联盟链的开源分布式账本平台。超级账本项目引入了权限控制和安全保障，从而开创了区块链技术的全新领域。事实上宣布了区块链技术已经不完全局限在完全开放的公有链模式下，更重要的是标志着区块链技术已经被主流企业认可并在实践中采用。超级账本项目中提出和实现了许多创新的设计和理念，包括完备的权限和和审查管理，可信的隐私保护，以及可插拔，可扩展的实现框架。本项目提出一种基于区块链的运行在开放的、缺乏信任和无中心节点的互联网环境下的服务交易平台。

将Hyperleder Fabric项目应用在高校学生的学分存储和验证中，可以很大程度的提高教育应有的公信力，并且降低用人单位的成本，一定程度上减少了简历造假的风险，这就是区块链给高等教育领域带来的意义。

### 二、方案设计

本设计利用手机APP获取学生的基本信息，使用加密算法生成数字摘要，通过手机网络把数据发送到服务器进行处理，确认是合法的用户发送的信息后，把数据广播到区块链网络等待共识，区块链网络达成共识后，即可把共识的结果存放在区块链网络中，同时返回给用户唯一的ID。用户可以根据该ID和其他的一些信息在区块链网络中查询到真实的没有被篡改的成绩等信息。



我使用了下图所示的Hyperledger Fabric 1.3的应用开发模型来实现学分存链和鉴定的应用场景，基于区块链的学分鉴定进行分层设计，包括Hyperledger Fabric 1.3底层平台，智能合约，业务层和应用层。

每个层的主要功能如下所示：

（1）区块链底层平台：提供分布式账本的维护和状态数据库维护等区块链功能，实现数据的不可篡改和智能合约的业务逻辑。

1. 智能合约：智能合约通过链码来实现，包括存链和鉴定，链码查询都用智能合约来实现。
2. 业务层：业务层是应用程序的后端功能，处理APP的各种具体的业务请求。
3. 应用层：一般由第三方企业进行开发，为用户提供丰富的功能。



### 三、代码分析

一）环境配置（SCSP/artifacts）

## 1、生成公私钥和证书

Fabric中有两种类型的公私钥和证书，一种是给节点之前通讯安全而准备的TLS证书，另一种是用户登录和权限控制的用户证书。这些证书本来应该是由CA来颁发，但是我们这里是测试环境，并没有启用CA节点，所以Fabric帮我们提供了一个工具：cryptogen。

1.1 编译生成cryptogen（SCSP/artifacts/bin/）

直接执行命令make cryptogen即可。

1.2 配置cryptogen.yaml文件（SCSP/artifacts/channel/）

该文件提供了一个Orderer Org和两个Peer Org的配置。其中Name和Domain就是关于这个组织的名字和域名，这主要是用于生成证书的时候，证书内会包含该信息。而Template Count=2是说要生成2套公私钥和证书，一套是peer0.org2的，还有一套是peer1.org2的。最后Users. Count=1是说每个Template下面会有几个普通User（Admin不包含在这个计数中），这里配置了1，也就是说只需要一个普通用户[User1@org2.example.com](mailto:User1@org2.example.com)，可以根据实际需要调整这个配置文件，增删Org、Users等。

2、生成创世区块和Channel配置区块

2.1 编译生成configtxgen（SCSP/artifacts/bin/）

## 与1.1类似，我们可以通过make configtxgen命令生成configtxgen

### 2.2 配置configtx.yaml文件（SCSP/artifacts/channel/）

该文件里面配置了由2个Org参与的Orderer共识配置TwoOrgsOrdererGenesis，以及由2个Org参与的Channel配置：TwoOrgsChannel。Orderer可以设置共识的算法是Solo还是Kafka，以及共识时区块大小，超时时间等。而Peer节点的配置包含了MSP的配置，锚节点的配置。

### 2.3 生成创世区块

配置修改好后，就用configtxgen生成创世区块。

### 2.4生成Channel配置区块

最终，我们在channel文件夹中，应该是能够看到：

channel/   
├── softwarechannel.tx   
├── genesis.block   
├── crypto-config

├── ordererOrganzitions  
└── peerOrganzations

## 3、配置Fabric环境的docker-compose文件

### 3.1 配置Orderer（SCSP/artifacts/base.yaml）

其中ORDERER\_GENERAL\_GENESISFILE=/etc/hyperledger/configtx/genesis.block，这个创世区块就是我们之前创建的创世区块，这里就是Host到Docker的映射：../channel/genesis.block:/etc/hyperledger/configtx/genesis.block

另外的配置主要是TL，Log等，最后暴露出服务端口7050。

### 3.2 配置Peer（SCSP/artifacts/base.yaml和SCSP/artifacts/docker-compose.yaml）

在Peer的配置中，主要是给Peer分配好各种服务的地址，以及TLS和MSP信息。

### 3.3 配置CLI（SCSP/artifacts/docker-compose.yaml）

CLI在整个Fabric网络中扮演客户端的角色，我们在开发测试的时候可以用CLI来代替SDK，执行各种SDK能执行的操作。CLI会和Peer相连，把指令发送给对应的Peer执行。

## 二）运行

### 1、启动Fabric网络

我们通过执行脚本（SCSP/runApp.sh）即可启动fabric网络。在该文件中，首先通过以下命令将正在启动的网络关闭：

docker-compose -f ./artifacts/docker-compose.yaml down

接着，通过以下命令将正在运行的docker容器删除：

CONTAINER\_IDS=$(docker ps -aq)

docker rm -f $CONTAINER\_IDS

同样，通过以下命令将生成的镜像删除：

DOCKER\_IMAGE\_IDS=$(docker images | grep "dev\|none\|test-vp\|peer[0-9]-" | awk '{print $3}')

docker rmi -f $DOCKER\_IMAGE\_IDS

最后，重新启动fabric网络：

docker-compose -f ./artifacts/docker-compose.yaml up -d

### 2、创建Channel（SCSP/scripts/install.sh）

curl -s -X POST \

http://localhost:4000/channels \

-H "authorization: Bearer $ADMIN\_GFE\_TOKEN" \

-H "content-type: application/json" \

-d '{

"channelName":"softwarechannel",

"channelConfigPath":"../artifacts/channel/softwarechannel.tx"

}'

3、将Peer加入到Channel中（SCSP/scripts/install.sh）

curl -s -X POST \

http://localhost:4000/channels/softwarechannel/peers \

-H "authorization: Bearer $ADMIN\_DEKE\_TOKEN" \

-H "content-type: application/json" \

-d '{

"peers": ["peer0.fabric.deke.com","peer1.fabric.deke.com"]

}'

4、安装Chaincode（SCSP/scripts/install.sh）

curl -s -X POST \

http://localhost:4000/chaincodes \

-H "authorization: Bearer $ADMIN\_GFE\_TOKEN" \

-H "content-type: application/json" \

-d "{

\"peers\": [\"peer0.fabric.gfe.com\",\"peer1.fabric.gfe.com\"],

\"chaincodeName\":\"$CC\_NAME\",

\"chaincodePath\":\"$CC\_SRC\_PATH\",

\"chaincodeType\": \"$LANGUAGE\",

\"chaincodeVersion\":\"v0\"

}"

5、Chaincode实例化（SCSP/scripts/install.sh）

curl -s -X POST \

http://localhost:4000/channels/softwarechannel/chaincodes \

-H "authorization: Bearer $ADMIN\_GFE\_TOKEN" \

-H "content-type: application/json" \

-d "{

\"peers\": [\"peer0.fabric.gfe.com\"],

\"chaincodeName\":\"currency\",

\"chaincodeVersion\":\"v0\",

\"chaincodeType\": \"$LANGUAGE\",

\"args\":[]

}"

6、用户和服务提供方进行注册（SCSP/scripts/regist.sh）

7、服务提供方加入候选队列（SCSP/scripts/listener.sh）

8、服务协商过程（SCSP/scripts/request.sh）

## 三)Node.js

选择使用Node.js来实现相应的业务代码。服务端监听客户端的请求。app文件夹存放操作区块链网络的相关代码，bishe文件夹存放的Fabric需要的配置文件，node\_modules存放的是Fabric的SDK，app.js是业务逻辑逻辑代码，负责和客户端进行交互，config用于配置，run.sh用于启动Fabric网络。

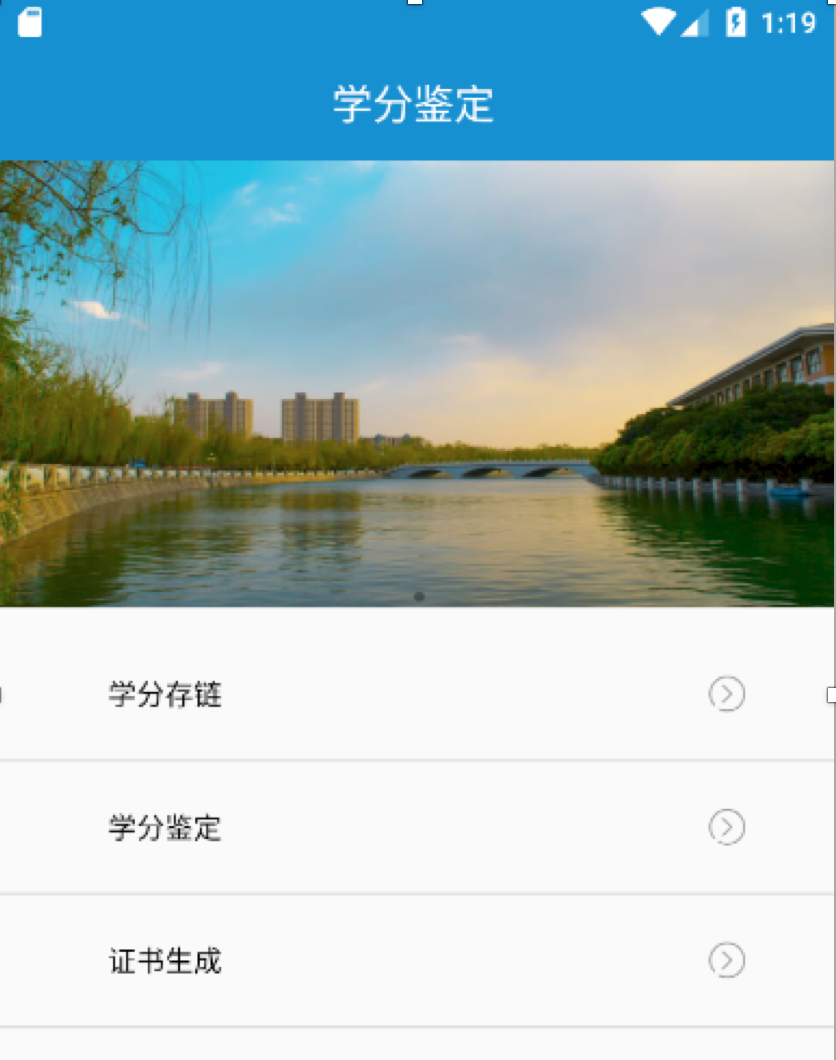
详细的分析如下所示：首先Org1请求注册，然后创建channel通道，使用curl命令调用本地的已经配置好的文件，传入请求头，json格式的文件，以及文件的路径。之后加入channel通道，再然后安装资产链码到Org1上，依然需要指定请求体的类型，链码的名字，链码的路径。在然后为Org1的peer1节点进行实例化资产链码。

当上面的代码成功执行完毕后，Fabric网络即可成功启动，通过调用app.js开始监听客户端发送的请求。

## 四）Android客户端的实现

Activity目录下存放具体的业务逻辑代码；adapter目录下存放适配器的代码，蛀牙用在ViewPager和ListView上；base目录下存放一些基类；bean目录下存放一些实体类；constants目录下存放一些常量；Fragement目录下存放具体的fragement；receiver目录下存放广播接收者；util目录下存放一些实用的工具。





### 四、总结

通过本次课程设计，我熟练掌握了基于Fabric的区块链平台的搭建;基本掌握了基于区块链Fabric平台再次开发应用的能力；学会了基于Fabric CA的证书的申请，签发和挂失功能。基本实现了在区块链中存取数字证书的功能。掌握了利用Android平台开发APP来实现和用户进行交互，并且把用户的请求发送到服务端的功能。基本实现了采用Node.js高级编程语言为应用提供后台服务API的功能。