中山大学数据科学与计算机学院本科生实验报告

（2019年秋季学期）

课程名称：区块链原理与技术　　任课教师：郑子彬

年级	17级	专业（方向）	软件工程
学号	17343136	姓名	颜治文
电话		Email	
开始日期	2019-12-8	完成日期	2019-12-12

一. 项目背景

在第二次试验中开发出来的智能合约（Asset.sol）的基础上完成前端+后端+链端的微型应用程序。

具体实现功能：账户创建，账户查询，账户间借贷，账户间还款，账户向银行融资这五个操作。

二. 方案设计

本次试验采用的工具包为：web3SDK，因此是采用java语言编写的。

存储设计：

本次试验采取的的是：将每次交易情况上链（Emit），同时将交易数据存入数据库来进行存储的方法。

其中交易上链采用的是事件提交的方法：

```
public List<RegisterEventEventResponse> getRegisterEventEvents(TransactionReceipt transactionReceipt) {
                List<Contract.EventValuesWithLog> valueList = extractEventParametersWithLog(REGISTEREVENT_EVENT, transactionReceipt);
```

```
        ArrayList<RegisterEventEventResponse> responses = new
ArrayList<RegisterEventEventResponse>(valueList.size());

    for (Contract.EventValuesWithLog eventValues : valueList) {

        RegisterEventEventResponse typedResponse = new RegisterEventEventResponse();

        typedResponse.log = eventValues.getLog();

        typedResponse.ret = (BigInteger) eventValues.getNonIndexedValues().get(0).getValue();

        typedResponse.account = (String) eventValues.getNonIndexedValues().get(1).getValue();

                    typedResponse.status = (BigInteger)
eventValues.getNonIndexedValues().get(2).getValue();

        responses.add(typedResponse);

    }

    return responses;

  }
```

其中register的事件对应的java类型为：

```
public static class RegisterEventEventResponse {

  public Log log;

  public BigInteger ret;

  public String account;

  public BigInteger status;

}
```

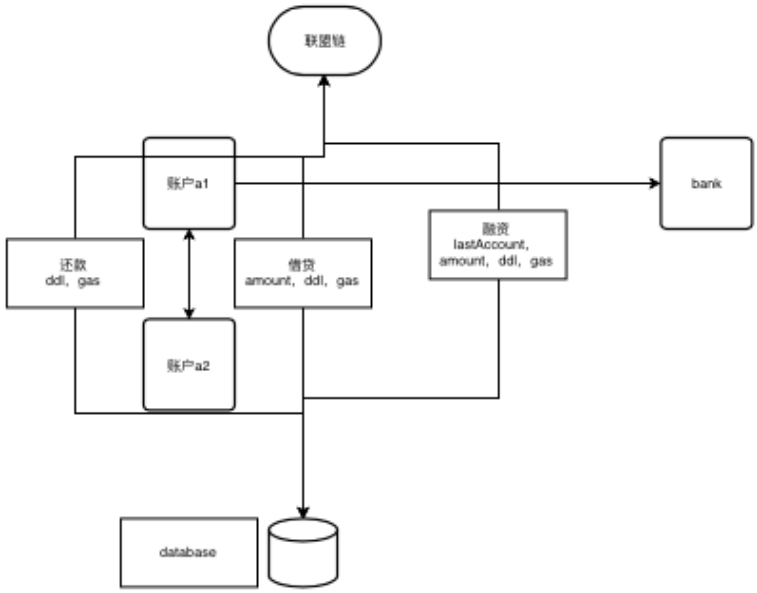
其中具体的代码逻辑如下：

```
public void register(String account, BigInteger status, TransactionSucCallback callback) {

  final Function function = new Function(

      FUNC_REGISTER,
      Arrays.<Type>asList(new org.fisco.bcos.web3j.abi.datatypes.Utf8String(account),
      new org.fisco.bcos.web3j.abi.datatypes.generated.Int256(status)),
      Collections.<TypeReference<?>>emptyList());
  asyncExecuteTransaction(function, callback);
}
```

数据流程图：



三. 功能测试

本次试验采用的前端交互为命令行模式。即通过运行脚本，并在后面添加参数的方法运行每个模块。

1，节点启动

这里使用的是在第一次试验中创建的4节点联盟链。因此节点启动操作需要前往对应文件夹运行启动脚本。

```
root@fiscobcos-VirtualBox:/home/fisco-bcos/fisco/nodes/127.0.0.1# ./start_all.sh

try to start node0
try to start node1
try to start node2
try to start node3
 node2 start successfully
 node3 start successfully
 node1 start successfully
 node0 start successfully
```

2，环境配置与代码编译

本次试验采用了gradle软件进行快速打包编译。具体需要的第三方jar包依赖项将会自动添加到工程项目中。

```
root@fiscobcos-VirtualBox:/home/fisco-bcos/asset-app# ./gradlew build

BUILD SUCCESSFUL in 3s
4 actionable tasks: 2 executed, 2 up-to-date
```

3，deploy（合约部署）

```
root@fiscobcos-VirtualBox:/home/fisco-bcos/asset-app/dist# ./asset_run.sh deploy
 deploy Asset success, contract address is 0xc4e15dced85bcd8b838082189a7a216ab61
08efb
```

```
root@fiscobcos-VirtualBox:/home/fisco-bcos/asset-app/dist# ./asset_run.sh regist
er a1 0
root@fiscobcos-VirtualBox:/home/fisco-bcos/asset-app/dist# ./asset_run.sh owe a1
 a2 100 123
 Usage :
root@fiscobcos-VirtualBox:/home/fisco-bcos/asset-app/dist# ./asset_run.sh retu a
1 a2 123
root@fiscobcos-VirtualBox:/home/fisco-bcos/asset-app/dist# ./asset_run.sh requ a
1 a2 a3 100 123
```

四. 界面展示

五. 心得体会

本次试验由于划分了若干个层次进行，因此还算成功地完成了所有任务，最终实现了一个能够使用的区块链应用。

在此过程中，我感觉到的最大的难点在于：1，环境配置繁杂。尽管相关教程案例非常详细，但仍旧会出现在预料之外的结果；2，对该项目的工作原理的理解。这是一个循序渐进的过程，但也因此在试验初期对智能合约的编写造成了不小的困扰。

依照自己的理解，这次就是一个运行在联盟链上的一个用于解决信任传递问题的应用程序。多个节点竞争同一条链的大包权，而打包的数据则来自于智能合约调用的结果。使用智能合约的用户无需关心交易的可靠性，因为这一点是通过将交易情况上链来完成的。具体实现便是一个emit。用户将各自的账户与账单情况存入数据库以进行持久化。通过查询，修改数据库来实现借贷，还款，融资等操作。