**第一部分：架构设计逻辑**

区块链环境

(Fabric 1.0.2)

访问界面

Block-listen-tls

**区块链事件监听**

前端后台

(nodejs angularjs)

数据更新(30S一次)

**BaaS平台**

数据库（mysql）

**区块链后台**

Fabric 1.0.2 nodeJs SDK

客户端

直接调用后台接口

读取/写入数据

Grpc通信

调用接口

读取事件端口数据

更新区块链最新数据

写入日志数据

**前端后台处理流程：(**通过BaaS平台界面访问)

1. 客户端访问BaaS平台界面，从数据库读取数据进行展示。
2. 客户端在界面发起区块链相关操作，则调用区块链后台相关接口访问区块链环境，操作

成功后,写入相关记录到数据库。

注：客户端可以不通过BaaS平台界面，直接调用区块链后台相关接口访问区块链环境，并返回结果信息。系统每隔30S进行数据更新，发现区块链环境数据有变化后，会更新最新数据到数据库。

1. **数据更新处理流程：**

1.通过js的setTimeInterval,间隔30S调用区块链后台相关接口获取最新数据，获取成功后，更新相关记录到数据库。

1. **区块链事件监听处理流程：**

1.在连接区块链环境的peer节点的事件监听端口，当发生创建管道，初始化合约，调用合

约和升级合约时，获取相关数据，并写如新记录到数据库的log表。

1. **分：Channel[channelName][orgName]和client[orgName]两个全局对象的变化过程**

**一、正常流程**

（1）初始化：

Balance-transfer后台启动时，会重新初始化所有组织的client[orgName]对象

（2）创建管道过程：

2.1.构造request请求，包括：

let request = {  
 **config**: channelConfig,  
 **signatures**: [signature],  
 **name**: channelName,  
 *//orderer: channel.getOrderers()[0], //origin code* **orderer**: helper.*newOrderer*(client), *//edit by lmh 20170904* **txId**: client.newTransactionID()  
};

2.2 client发送createChannel(request)请求到区块链环境的orderer节点。

2.3设置全局变量Channel[channelName][orgName]对象。每次创建一个新管道。会产生属于这个管道的所有组织的Channel[channelName][orgName]对象。

1. 加入管道

1.获取要加入管道的peer所属的组织的client[orgName]对象，然后产生事务ID

tx\_id = client.newTransactionID();

1. 获取要加入管道的peer所属的组织的channel[channelName][orgName]对象,构造request请求，发送给orderer节点，获取创世区块信息。

let request = {  
 **txId** : tx\_id  
 };  
 channel.getGenesisBlock(request);

3.同时产生两个请求，一个是发送加入管道的请求，一个是连接eventHub获取请求的处理结果。

4.获取全局变量Channel[channelName][orgName]对象，执行addPeer（）操作。

let peer = client.newPeer(  
 ORGS[org].**peers**[peerName].**requests**,  
 {  
 **pem**: Buffer.**from**(data).toString(),  
 **'ssl-target-name-override'**: ORGS[org].**peers**[peerName][**'server-hostname'**]  
 }  
 );

peer.setName(peerName);

channel.addPeer(peer);

1. **停止balance-transfer后台重启时重新初始化流程**

更新线程要更新的数据：（已知区块链环境所有peer节点）

1. 管道数据，遍历所有peer节点查询加入的管道，汇总得到所有的管道数据。

得到两个视图数据：

Peer节点 加入管道数据

和 管道名 加入peer节点

1. 安装的智能合约和初始化的智能合约

主要逻辑：

从区块链环境查询出相关信息，然后和数据库的记录进行对比，同步最新数据记录到数据库。

实现步骤：

1.遍历所有peer节点，每个节点都调用三个查询接口。

（1）查询peer加入的管道信息

（2）查询peer安装的智能合约

（3）查询peer初始化的智能合约

1. 相当于做一个小型的区块链汇总数据库

先把所有的管道名

拿到一条数据：

Peer1:channelName

得到peer1对应节点的pk\_Id,

写入日志数据