[1引言 2](#_Toc518003344)

[1.1编写目的 2](#_Toc518003345)

[1.2背景 2](#_Toc518003346)

[1.3定义 2](#_Toc518003347)

[1.4参考资料 2](#_Toc518003348)

[2总体设计 2](#_Toc518003349)

[2.1需求规定 2](#_Toc518003350)

[2.2运行环境 3](#_Toc518003351)

[2.3基本设计概念和处理流程 3](#_Toc518003352)

[2.4结构 4](#_Toc518003353)

[2.5功能需求与程序的关系 7](#_Toc518003354)

[2.6人工处理过程 7](#_Toc518003355)

[2.7尚未问决的问题 8](#_Toc518003356)

[3接口设计 8](#_Toc518003357)

[3.1用户接口 8](#_Toc518003358)

[3.2外部接口 8](#_Toc518003359)

[3.3内部接口 8](#_Toc518003360)

[4运行设计 9](#_Toc518003361)

[4.1运行模块组合 9](#_Toc518003362)

[4.2运行控制 9](#_Toc518003363)

[4.3运行时间 9](#_Toc518003364)

[5系统数据结构设计 10](#_Toc518003365)

[5.1逻辑结构设计要点 10](#_Toc518003366)

[5.2物理结构设计要点 12](#_Toc518003367)

[5.3数据结构与程序的关系 12](#_Toc518003368)

[6系统出错处理设计 12](#_Toc518003369)

[6.1出错信息 12](#_Toc518003370)

[6.2补救措施 13](#_Toc518003371)

[6.3系统维护设计 13](#_Toc518003372)

**概要设计说明书**

# 1引言

## 1.1编写目的

本设计文档是可信数字资产存证应用的研发概要设计，将应用开发进程中或者项目结束后提供给双方人员使用，同时也可作为实施后期的维护人员使用。

## 1.2背景

近年来大家对数字资产的关注呈现上升趋势，所谓的数字资产，就是以电子数据形式存在的可编程控制的资产。而区块链技术也随着比特币的大热得到了前所未有的发展，其去中心化，去中介，免信任的特点非常适用于在货币的交易流程中。随着技术的不断更新，区块链也有了更多的应用场景。利用区块链去中心化的特点，实现对数字资产的存储，鉴权等操作。通过区块链技术手段，对传统的存证应用进行技术升级，应用于更多场景下的存证服务。

鉴于此，我们将开发出以区块链网络为底层的资产存证软件，名为可信数字资产存证应用。

**委托单位：** 新华三技术有限公司 **开发单位：**Genesis-Block 团队 **负责人：**曾凯

## 1.3定义

Fabric：IBM开源的区块链项目，本应用后台区块链采用的架构

SADA：可信数字资产存证应用简称

## 1.4参考资料

《国家标准软件开发文档规范》

《软件开发流程》，清华大学出版社，2005年1月版

# 2总体设计

## 2.1需求规定

软件项目的大致功能要求如下：

1. 底层区块链网络采用至少4个节点
2. 编写对资产进行保存，查询，修改等操作的链码
3. 编写与链码进行交互的接口
4. 普通用户填写注册信息后注册成功并进入系统
5. 采用图片识别对资产中的关键信息进行提取
6. 普通用户上传资产到存储服务器和区块链网络
7. 普通用户对自己的资产进行鉴权操作
8. 普通用户根据资产类别进行资产查询浏览
9. 普通用户根据资产标签对资产进行查询浏览
10. 普通用户加入组织
11. 组织管理者对组织内用户的资产进行鉴权
12. 超级管理员对区块链网络进行可视化查看
13. 超级管理员对系统的用户，角色，权限进行管理

详细说明参加《需求分析说明书》

## 2.2运行环境

主机：PC端 内存 1G 以上

操作系统：Window 10/7 等，Linux

浏览器：Chrome/Firefox

## 2.3基本设计概念和处理流程

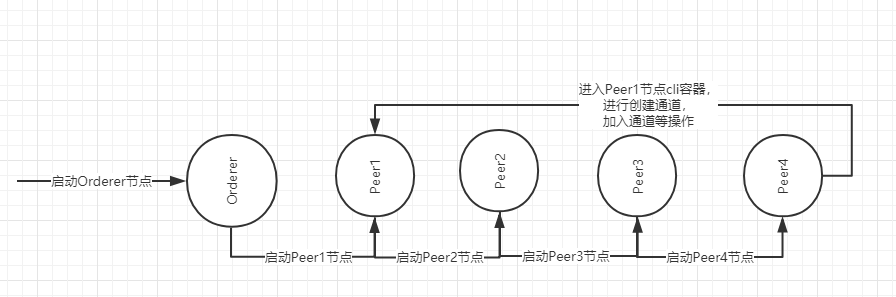
**基本概念**

Hyperledger Fabric是 The Linux Foundation 主办的 Hyperledger 项目之一。Hyperledger Fabric 旨在作为开发模块化体系结构的区块链应用程序的基础，以便诸如共识和会员服务等组件可以即插即用。它使用容器技术来托管构成系统应用逻辑的智能合约（也称为链代码）。 简而言之，Hyperledger Fabric是为企业构建的领先的开源、通用区块链结构。

**基本流程**

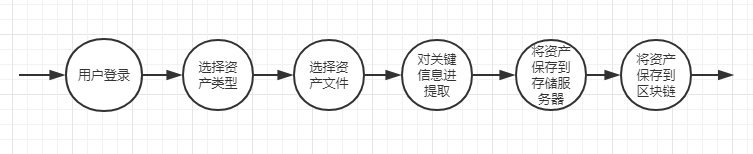
（1）底层区块链网络的创建与启动流程：

底层区块链网络使用的是 Fabric 1.0 ，采用多机部署的方式，有4 个 Peer 节点加一个Orderer 节点，以及2个CA 节点，网络的启动流程如下：



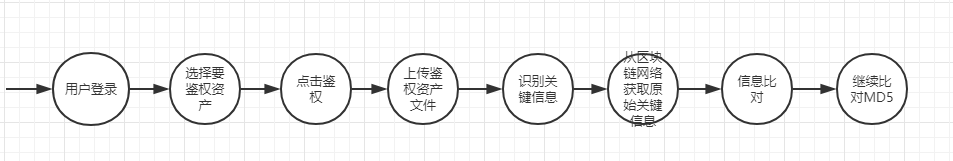
（2）用户资产上传流程

用户选择资产类型，上传资产图片或文件后，先识别出关键信息，然后将关键信息以及资产的MD5值保存到区块链网络。上传流程如下:



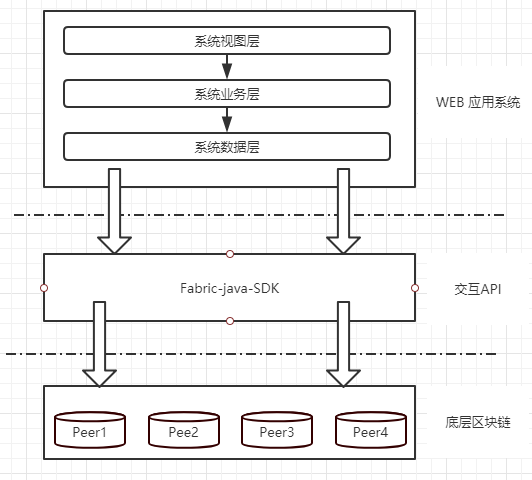
（3）用户鉴权流程

用户选择一张需要鉴权的照片，上传并提取关键信息，从后台区块链网络中读取原资产关键信息，进行比对。鉴权具体流程如下：



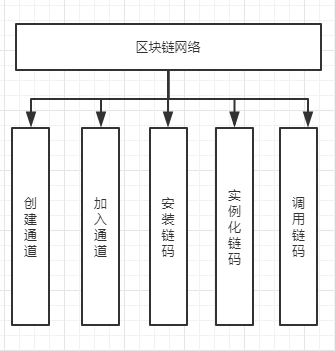
## 2.4结构

根据项目小组进行项目需求分析以及多次讨论，将整个项目分为3个部分，分别是底层区块链网络，与区块链网络进行交互的程序以及供用户使用的web系统，整体架构图如下：

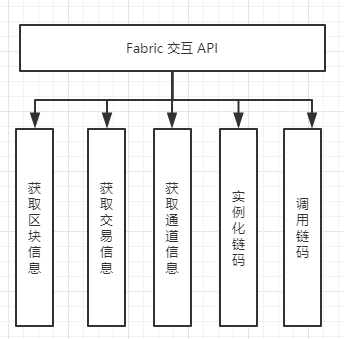


虽然说这三个部分是分开的，可以相互独立运行，但是他们是密不可分的。每一层都为上一层提供了非常重要的数据接口。

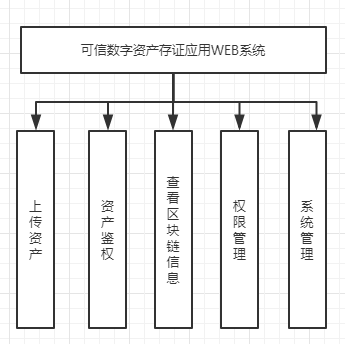
底层区块链网络的主要操作体现在链码的方法上，调用链码对资产信息的保存，查询，删除，更新等操作。



交互API主要是通过fabric 官方提供的 java SDK 来与区块链网络进行交互，除了能安装，实例化，调用链码，还能对获取区块信息，交易信息，通道信息等



WEB 应用的主要功能有资产的上传与修改，区块链信息的查看以及对系统用户，角色，权限等的管理。



## 2.5功能需求与程序的关系

|  |  |  |  |
| --- | --- | --- | --- |
| 功能需求 | 区块链网络 | 交互API | WEB 应用 |
| 底层区块链网络采用至少4个节点 | √ |  |  |
| 编写对资产进行保存，查询，修改等操作的链码 | √ |  |  |
| 编写与链码进行交互的接口 |  | √ |  |
| 普通用户填写注册信息后注册成并进入系统 |  | √ |  |
| 采用图片识别对资产中的关键信息进行提取 |  |  | √ |
| 普通用户上传资产到存储服务器和区块链网络 |  |  | √ |
| 普通用户对自己的资产进行鉴权操作 |  |  | √ |
| 普通用户根据资产类别进行资产查询浏览 |  |  | √ |
| 普通用户根据资产标签对资产进行查询浏览 |  |  | √ |
| 普通用户加入组织 |  |  | √ |
| 组织管理者对组织内用户的资产进行鉴权 |  |  | √ |
| 超级管理员对区块链网络进行可视化查看 |  |  | √ |
| 超级管理员对系统的用户，角色，权限进行管理 |  |  | √ |

## 2.6人工处理过程

底层区块链人工处理有：

1. 多机部署下区块链网络的启动
2. 创建通道，加入通道，更新锚节点

交互API人工处理有：

1. 启动程序
2. 安装并实例化链码

WEB 应用人工处理有：

1. 资产的上传选择及名称等输入
2. 用户查询资产的输入
3. 鉴权资产时的资产选择及名称等输入
4. 不同权限角色用户的各种功能输入
5. 不同权限角色用户的各种服务时请求的提交

说明在本软件系统的工作过程中不得不包含的人工处理过程（如果有的话）。

## 2.7尚未问决的问题

（1）关键信息的识别准确度

（2）交互API的设计与实现

# 3接口设计

## 3.1用户接口

用户的登录注册界面

资产上传界面

资产鉴权界面

资产查询界面

## 3.2外部接口

与区块链网络进行交互的API接口

识别关键信息调用外部的API接口

## 3.3内部接口

内部接口方面，各个模块采用函数调用，参数传递、返回值的方式进行信息传递。具体参数的结构将会在系统数据结构设计的内容中说明。接口传递的信息将以Bean对象或者JSON格式封装了数据，以参数传递或返回值的形式在各模块间传递。

# 4运行设计

## 4.1运行模块组合

WEB系统

依据不同的角色，系统决定用户可以访问的页面以及可以使用的功能

交互API

与后台区块链网络进行交互的时候需要Token秘钥，如果没有秘钥或者秘钥失效，将不能进行交互操作

底层区块链网络

区块链网络启动过程必须依照固定的启动顺序，否则将会出现错误

## 4.2运行控制

WEB 系统：

1. 普通用户登录，可以进行资产的上传与鉴权，个人信息的修改，加入组织等操作
2. 组织管理者登录，可以对组织内用户的资产进行鉴权操作，对个人资产的上传与鉴权操作，修改个人信息等
3. 超级管理员登录，可以通过控制台对区块链网络的运行情况进行查看，可以对系统的用户，权限，角色进行管理等

交互API：

1. 使用区块链网络的管理员身份生成一串Token秘钥，使用该秘钥可以利用SDK进行创建通道，加入通道，安装链码，实例化链码，调用链码等操作。

底层区块链网络：

（1） 按照Orderer——Peer1——Peer2——Peer3——Peer4 的顺序进行启动

## 4.3运行时间

（1）底层区块链网络出块时间间隔为两秒

（2）交互API的响应时间不超过10秒，具体由机器的配置决定

（3）WEB 系统各个页面的响应时间不超过5秒，具体由机器的配置决定

# 5系统数据结构设计

## 5.1逻辑结构设计要点

**用户数据：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 是否为空 | 主/外键 | 数据描述 |
| id | varchar | 否 | 主键 | 用户Id |
| name | varchar | 否 |  | 用户姓名 |
| nickname | varchar | 是 |  | 用户昵称 |
| password | varchar | 否 |  | 密码 |
| salt | varchar | 否 |  | 盐值 |
| orgId | varchar | 是 |  | 组织Id |
| sexId | varchar | 是 |  | 性别Id |
| birthday | varchar | 是 |  | 出生日期 |
| email | varchar | 是 |  | 邮箱 |
| mobile | varchar | 是 |  | 手机号 |
| userPic | varchar | 是 |  | 用户头像 |
| remark | varchar | 是 |  | 备注 |
| adminIs | varchar | 否 |  | 是否为管理员 |
| deleted | varchar | 否 |  | 是否启用 |
| createtime | varchar | 是 |  | 创建时间 |
| updatetime | varchar | 是 |  | 修改时间 |

**资产数据：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 是否为空 | 主/外键 | 数据描述 |
| id | varchar | 否 | 主键 | 资产Id |
| fileName | varchar | 否 |  | 资产文件名称 |
| filePath | varchar | 是 |  | 资产文件路径 |
| tag | varchar | 否 |  | 资产标签 |
| assetType | varchar | 否 |  | 资产类型 |
| commonType | varchar | 是 |  | 资产文件类型 |
| keyInfo | varchar | 是 |  | 关键信息 |
| name | varchar | 是 |  | 资产名称 |
| isDelete | int | 是 |  | 是否删除 |
| userId | varchar | 是 | 外键 | 资产拥有者id |
| assetMd5 | varchar | 是 |  | 资产Md5值 |
| createDate | varchar | 是 |  | 创建时间 |
| updateDate | varchar | 是 |  | 修改时间 |

**标签数据：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 是否为空 | 主/外键 | 数据描述 |
| id | varchar | 否 | 主键 | 资产标签Id |
| num | int | 是 |  | 标签数量 |
| userId | varchar | 是 | 外键 | 标签拥有者id |
| createDate | varchar | 是 |  | 创建时间 |
| updateDate | varchar | 是 |  | 修改时间 |

**角色数据：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 是否为空 | 主/外键 | 数据描述 |
| id | varchar | 否 | 主键 | 角色Id |
| name | varchar | 是 |  | 角色名称 |
| code | varchar | 否 |  | 角色编码 |
| remark | varchar | 是 |  | 角色备注 |
| createtime | varchar | 是 |  | 创建时间 |
| updatetime | varchar | 是 |  | 更新时间 |
| sort | varchar | 是 |  | 排序 |
| deleted | varchar | 是 |  | 是否删除 |

**功能数据：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 是否为空 | 主/外键 | 数据描述 |
| id | varchar | 否 | 主键 | 功能Id |
| name | varchar | 是 |  | 功能名称 |
| code | varchar | 否 |  | 功能编码 |
| parentId | varchar | 是 |  | 功能父Id |
| url | varchar | 是 |  | 功能访问路径 |
| icon | varchar | 是 |  | 功能图标 |
| functype | varchar | 是 |  | 功能类型 |
| queryId | varchar | 是 |  | 数据查询Id |
| py | varchar | 是 |  | 功能拼音简称 |
| pingyin | varchar | 是 |  | 功能拼音全称 |
| remark | varchar | 是 |  | 备注 |
| createtime | varchar | 是 |  | 创建时间 |
| updatetime | varchar | 是 |  | 更新时间 |

**数据字典数据：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 是否为空 | 主/外键 | 数据描述 |
| id | varchar | 否 | 主键 | 字典Id |
| name | varchar | 是 |  | 字典名称 |
| levelcode | varchar | 否 |  | 字典等级编码 |
| code | varchar | 是 |  | 字典编码 |
| parentId | varchar | 是 |  | 父id |
| value | varchar | 是 |  | 值 |
| deleted | varchar | 是 |  | 是否禁用 |
| remark | varchar | 是 |  | 备注 |
| createtime | varchar | 是 |  | 创建时间 |
| updatetime | varchar | 是 |  | 更新时间 |

## 5.2物理结构设计要点

在本应用中，主要对用户的信息以及资产信息进行建立索引操作，以加快检索的效率

## 5.3数据结构与程序的关系

|  |  |  |  |
| --- | --- | --- | --- |
| 数据结构 | 资产上传 | 资产鉴权 | 权限管理 |
| 用户数据 |  |  | √ |
| 资产数据 | √ | √ |  |
| 标签数据 | √ |  |  |
| 角色数据 |  |  | √ |
| 功能数据 |  |  | √ |
| 数据字典数据 | √ | √ | √ |

# 6系统出错处理设计

## 6.1出错信息

区块链网络：

1. 区块链网络启动失败，打印出错日志信息

交互API

1. 执行方法出错了，将会输出详细的错误日志信息

WEB 应用

1. 用户登录，如果输入的用户或者密码与系统内的用户不匹配，那么系统将调到登录页面，并在控制台输出详细的错误日志信息
2. 上传资产，如果关键信息获取失败或者上传资产信息到区块链网络失败，将会在控制台输出详细的错误信息
3. 资产鉴权，如果获取关键信息失败或者从区块链网络的获取关键信息失败，将会在控制台输出详细的错误信息
4. 表单数据填写格式不正确，将会在页面显示错误信息
5. 操作数据库异常，将会在控制台打印错误日志

## 6.2补救措施

（1）后备技术：定期备份数据库文件，一旦数据库发生异常，立即用备份的数据库覆盖原来的数据库文件。

（2）安全技术：对安全要求较高的数据进行加密传输，使用Shiro 框架来对系统的权限做细颗粒控制

（3）恢复及再启动技术：定期的对数据库和服务器进行更新检查，并定期备份重要文件

## 6.3系统维护设计

（1） 可以使用JUnit工具在编码过程中设计单元测试案例

（2） 系统各层之间采用接口设计，方便对系统进行扩展

（3） 固定时间对系统进行维护检测

（4） 若系统出现瘫痪，可启动备用系统维持运转