L70

Q

上海九方云智能科技有限公司

企业标准

Q/ JFY 086-2023

金融区块链技术用户协议规范

Specification for financial block-chain technology user protocol

2023-03-31 发布　　2023-04-01 实施

上海九方云智能科技有限公司 发布

目录

前　　言

本文件按照GB/T 1.1—2020《标准化工作导则 第1部分：标准化文件的结构和起草规则》的规定起草。

本文件由上海九方云智能科技有限公司提出。

本文件由上海九方云智能科技有限公司归口。

本文件起草单位：上海九方云智能科技有限公司。

本文件主要起草人：张信之、崔桂彬、程超、凌忻洋。

引　　言

随着科技的不断进步和金融领域的不断创新，金融区块链技术正逐渐成为金融行业的重要组成部分。作为一种基于区块链技术的金融服务和交易系统，金融区块链技术为用户提供了更安全、透明和高效的金融交易环境。然而，为了确保金融区块链技术的正常运作和用户权益的保护，制定一份明确规范的用户协议变得至关重要。

金融区块链技术用户协议规范旨在明确用户在使用金融区块链技术过程中的权利和义务，以及平台提供商的权利和义务。该协议涵盖了诸多方面，包括用户的隐私保护、知识产权、风险承担等内容。通过遵守该协议，用户可以更好地了解自己在金融区块链技术中的地位和权益，并与平台建立起互信和合作的关系。

用户在金融区块链技术的使用过程中，应遵守相关的法律法规和协议规定，并自行承担使用金融区块链技术所带来的风险。同时，用户也享有一定的权利，包括真实、准确、完整地提供个人或实体信息，以及对个人信息的隐私保护等。

平台作为金融区块链技术的提供商，有责任保障系统的稳定运行，并采取合理的安全措施保护用户的个人信息和交易数据的安全性。平台对用户提交的信息进行审核，并根据用户的行为采取相应的措施，以维护金融区块链技术的正常秩序。

通过制定和遵守金融区块链技术用户协议规范，我们可以为用户和平台之间的合作提供明确的指导和保障，进一步促进金融区块链技术的发展和应用。用户在使用金融区块链技术之前，应仔细阅读并理解相关的用户协议和条款，以确保自身权益得到充分保护。

1 范围

本文件给出了金融服务中制定用户协议符合监管部门相关规范，将用户协议进行链上管理。
本文件适用于在金融服务市场中进行地方业务链系统建设或服务运营的机构。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本 (包括所有的修改单) 适用于本文件。

GB/T 25069 信息安全技术术语
GB/T 32905-2016 信息安全技术SM3密码杂凑算法
GB/T 32918—2016 SM2椭圆曲线公钥密码算法
GM/T 0015—2012 基于SM2密码算法的数字证书格式规范
GM/T 0004-2012 SM3密码杂凑算法
JR/T 0184—2020 金融分布式账本技术安全规范
JR/T 0193—2020 区块链技术金融应用评估规则

3 术语和定义

GB/T 25069 界定的以及下列术语和定义适用于本文件。

3.1 区块链 blockchain

区块链是分布式数据存储、点对点传输、共识机制、加密算法等计算机技术的新型应用模式。是一个共享的分布式账本技术，其中用户协议通过附加块永久记录

3.2 区块 block

数据以文件形式被永久记录，这些协议文件称为区块。一个区块是一些或所有最新的用户协议的记录集，且未被其他先前的区块记录。

3.3 区块头 hotsCoin

区块头里面存储区块的头信息，包含上一个区块的哈希值（PreHash）、当前区块的哈希值（Hash）、以及时间戳（TimeStamp）等等。

3.4 去中心化 decentralization

去中心化是一种现象或结构，必须在拥有众多节点的系统中或在拥有众多个体的群中才能出现或存在。节点与节点之间的影响，会通过网络形成非线性因果关系。

3.5 分布式账本 DistributeDledger

数据通过分布式节点网络进行存储。分布式账本不是必须具有自己的货币，他可能会被许可和私有。

3.6 分布式网络 DistributeNetwork

处理能力和数据分布在节点上而不是拥有集中式数据中心的一种网络。

3.7 全节点 FullNode

全节点拥有完整区块链账本的节点，全节点需要同步所有区块链数据么，能够独立校验区块链上的所有交易并实时更新数据，主要负责区块链的交易的广播和验证

3.8 节点 Node

由区块链网络的参与者操作的分类帐的副本。

3.9 记账节点

维护账本的网络节点，一个或多个peer节点组成peer组织。

3.10 共识节点

区块链网络中参与交易的。

3.11 智能合约 smart contract

智能合约是一种旨在以信息化方式传播、验证或执行合同的计算机协议。智能合约允许在没有第三方的情况下进行可信交易，这些交易可追踪且不可逆转。

3.12 共识协议 consensus protocol

分布式账本系统中各节点为达成一致采用的计算方法。

3.13 拜占庭容错　byzantine fault tolerance

存在消息丢失的不可靠信道上试图通过消息传递的方式达到一致性是不可能的。因此对一致性的研究一般假设信道是可靠的，或不存在这个问题。

3.14 跨链技术　cross chain

各区块链连接的桥梁，主要实现各区块链的原子交易、内部信息互通。

3.15 哈希值 Hash

一种将任意长度的消息压缩到某个固定长的的消息的方式。

3.16 哈希树　HashTree

一种树形数据结构，每个叶子节点均以数据块的哈希作为标签，而非叶子节点则以其子节点标签的加密哈希作为标签。

3.17 闪电网络

是实现进行链下交易，其本质上是用哈希时间锁定智能合约来安全地进行0确认交易的一种机制，通过设置巧妙的“智能合约”，使得用户在闪电网络上进行未确认的交易和黄金一样安全。

3.18 对等网络　P2P

对等计算机网络，是一种在对等者（Peer）之间分配任务和工作负载的分布式应用架构，是对等计算机模型在应用层形成的一种组网或网络形式。

3.19 SHA256

SHA-256是比特币一些列数字货币使用的加密。

3.20 高级加密标准 AES

密码学中的高级加密标准（Advanced Encryption Standard），又称非对称加密法，是美国联邦政府采用的一种区块加密标准。

3.21 共享账本　Ledger

每一个节点都保存相同的账本雷数据。

3.22 Hyperledger Fabric网络组成

是Hyperledger中的一个区块链项目，包含一个账本，使用智能合约并且是一个通过所有参与者管理交易的系统。

4 应用环境

金融用户协议中的应用环境，是金融用户协议上链应用环境的参考示例。在该参考示例中，区块链服务为用户协议构建由各金融服务公司、第三方机构和监管部门作为参与方组成的协作联盟，通过不可篡改、隐私保护的用户协议来提高双方信任。

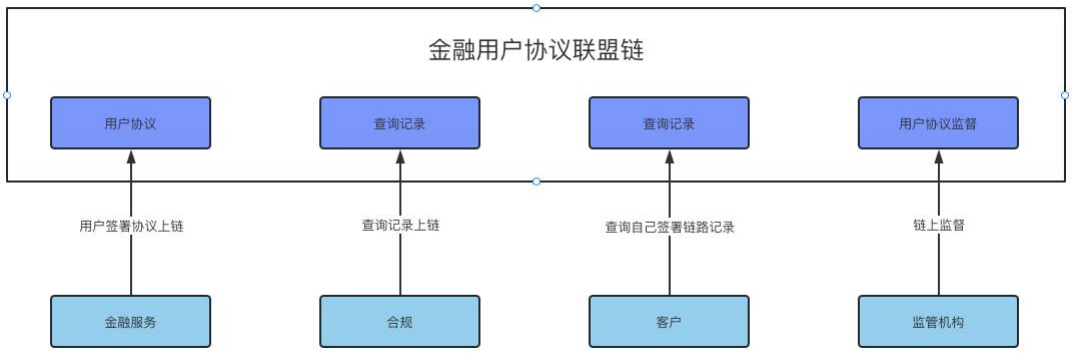
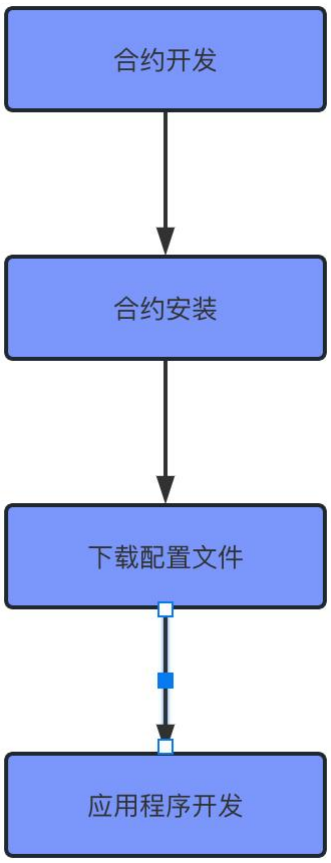


图 A.1 应用环境参考

5　用户协议上链

5.1　使用简介

本指导文章针对Java开发经验的人员进行开发指导，其中合约与应用程序需自行开发，整体流程如下：



5.2 合约开发

合约主要用于操作账本上的数据。作为运行在区块链上的、特定条件下自动执行的代码逻辑，合约是用户利用区块链实现业务逻辑的重要途径，基于区块链特点，合约的运行结果是可信的，其结果是无法被伪造和篡改的。

1. 下载合约SDK文件，基于SDK提供的类库进行合约开发

2. 合约SDK接口

接口	说明
FuncName() string	获取智能合约请求中指定的智能合约函数名
Parameters() [][]byte	获取请求参数。
ChainID() string	获取智能合约所在链ID。
ContractName() string	获取智能合约名称。
TxTimestamp() time.Time	获取本次交易的时间戳。

表1 stub接口

接口	说明
Version() (uint64, int32)	获取当前迭代位置（某笔交易）的
TxHash() []byte	获取当前迭代位置（某笔交易）的hash。
IsDeleted() bool	被查询的key，当前是否已经在状态数据库中
Timestamp() uint64	返回当前迭代位置（某笔交易）的时间戳。

表2 HistoryIterator接口

接口	说明
GetKV(key string) ([]byte, error)s	功能：获取状态数据库中某个key对应的value。 入参：某个键值对的key信息，不可为空。 返回值：返回[]byte类型的value值；当key不存在时，value为nil。 error：当网络出错，状态数据库出错，返回error信息。
PutKVCommon(key string, value interface{}) error	功能：写状态数据库操作，与PutKV功能相同；与PutKV接口的不同之处在于 value不是[]byte类型，而是一个实现了Marshal(vinterface{}) ([]byte, error)接口的数据，接口内部，会将value通过Marshal接口序列化，然后再形成写集。

	入参：要写入的键值对，要求key != ""，并且value实现了Marshal接口，可以序列化为[]byte。 error：入参错误。
DelKV(key string) error	功能：删除状态数据库中的key及其对应的value，此接口只是将待删除的key放入写集，打包到交易中，当交易排序、出块、并校验通过之后，将key删除。 入参：要删除的key要求key != ""。 error：入参错误。
GetIterator(startKey, endKey string) (Iterator, error)	功能：查询状态数据库中，按字典序，以startKey开头，以endKey结尾的所有状态数据，结果以迭代器的形式呈现；查询范围是左闭右开的，[startKey, endKey)。 入参：startKey是待查询状态数据的按字典序的起始key，startKey != ""，endKey是待查询的状态数据的按字典序的结束key，endKey!= ""。 返回值：Iterator是查询结果的迭代器，可以通过此迭代器，按顺序读取查询结果。 error：入参或网络错误。
GetKeyHistoryIterator(key string) (HistoryIterator, error)	功能：查询一个key对应的所有历史的value 入参：key是待查询历史value值的key信息，key != ""。 返回值：HistoryIterator是按顺序返回包含历史value结果的迭代器结构体变量。 error：入参或网络错误。
SaveComIndex(indexName string, attributes []string, objectKey string) error	功能：为objectKey保存索引信息，indexName_attributes_objectKey构成索引信息，注意，此处只是形成索引信息的写集，只有当含有此写集的交易经过排序、出块，并校验通过后，才会写入状态数据库。 入参：indexName 索引标记，indexName != ""，attributes需要当做索引的属性，至少包含一个属性信息，objectKey 待索引的key值，objectKey != ""。 error：入参错误。
GetKVByComIndex(indexName string, attributes []string) (Iterator, error)	功能：通过索引信息，查找满足某种查询条件的key/value，key/value以迭代器的形式输出。 入参：indexName 索引标记，indexName != ""，attributes需要当做索引的属性，至少包含一个属性信息 返回值：满足索引条件的key/value的迭代器变量。 error：入参或网络错误。
DelComIndexOneRow(indexName string, attributes []string, objectKey string) error	功能：删除objectKey的某个索引，indexName_attributes_objectKey构成索引信息，注意，此处只是形成索引信息的写集，只有当含有此写集的交易经过排序、出块，并校验通过后，才会写入状态数据库。 入参：indexName 索引标记，indexName != ""，attributes需要当做索引的属性，至少包

	含一个属性信息，objectKey 待索引的key值，objectKey != ""。 error：入参错误。
SplitComKey(comKey string) (string, []string, error)	功能：将查询到的复合键分离为objectkey和对应的attributes。 返回值：objectkey，和attributes字符串数组。
GetKVByPartialComKey(objectType string, attributes []string) (Iterator, error)	功能：部分复合键查询。 返回值：Iterator包含查询返回信息，支持迭代获取。

表3 contractStub接口

3. 合约结构

Java语言合约由合约文件及依赖包构成，包含包声明、依赖包导入、智能合约的方法定义。

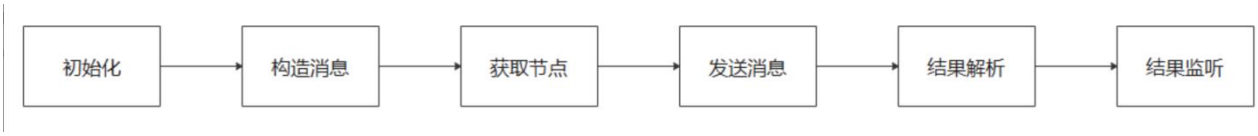
```
public class ExampleContract implements Contract {

  // 功能：合约的初始化（Init）接口，需要合约开发者在智能合约中实现此接口，供合约使用者在启动合约之后调用。注意，一般将合约启动时，首先需要执行且只需要执行一次的逻辑放到此方法中

  // 入参：stub是智能合约SDK为本次合约执行交易准备的上下文对象，可以通过stub提供的API函数，获取交易请求相关信息、读写状态数据库、写日志等

  // 返回值：需要返回给合约调用者（区块链客户端）的信息，没有信息需要返回时，返回值可以为null

  // 抛出异常：初始化过程的异常信息，可由合约编写者自行设定异常逻辑

  @Override

  public byte[] init(ContractStub stub) throws ContractException {

  }

  // 功能：合约被调用（invoke）接口，需要合约开发者在智能合约中实现此接口，将主要的合约执行逻辑，放到此接口内，供合约使用者调用。

  // 入参：stub是智能合约合约SDK为本次合约执行交易准备的上下文对象，可以通过stub提供的API函数，获取交易请求相关信息、读写状态数据库、写日志等

  // 返回值：需要返回给合约调用者（区块链客户端）的信息，没有信息需要返回时，返回值可以为null

  // 抛出异常：初始化过程的异常信息，可由合约编写者自行设定异常逻辑

  @Override

  public byte[] invoke(ContractStub stub) throws ContractException {

  }

}
```

5.3 应用程序开发

区块链服务启动时会启动一系列grpc接口，监听客户端发送的消息，与客户端交互完成各种请求。在开发客户端时，如果从底层grpc接口开始，进行各种消息封装、消息发送、返回值解析等工作，不仅会导致开发量过大，并且造成重复劳动。

SDK则是将区块链服务提供的各种grpc接口进行封装，同时封装各接口所需类型的消息。在开发客户端时，只需要关注自己的业务逻辑，调用相应接口封装并发送消息即可，不需要关注底层消息发送接收的具体过程。

基于SDK开发流程



5.3.1 利用合约发送交易

步骤1 合约调用信息构建

接口方法：

```
public RawMessage buildInvokeRawMsg(String chainId, String name, String function, String[] args)
```

参数说明

参数	类型	说明
chainId	String	链ID
name	String	合约名称
function	String	调用合约中的方法名
args	String[]	合约方法参数

返回值

类型	说明
RawMessage	消息体，用于合约调用

步骤2 背书请求消息构建

接口方法：

```
public RawMessage getRawMessageBuilder(ByteString payload) throws CryptoException
```

参数说明

参数	类型	说明
payload	ByteString	合约调用信息，由invocation.toByteString()得到

返回值

类型	说明
RawMessage	消息体，用于背书请求

步骤3 背书请求消息发送

接口方法:

```
public ListenableFuture<RawMessage> invoke(RawMessage rawMessage) throws InvalidParameterException
```

参数说明

参数	类型	说明
rawMessage	RawMessage	背书请求的消息体，步骤2的返回值

返回值

类型	说明
ListenableFuture	future对象，用于获取背书请求结果。

步骤4 落盘消息构建

接口方法:

```
public TxRawMsg buildTxRawMsg(RawMessage[] rawMessages) throws InvalidProtocolBufferException,
TransactionException, CryptoException
```

参数说明

参数	类型	说明
rawMessage	RawMessage	消息体，步骤3的背书请求结果。

返回值

类型	说明
TxRawMsg	交易消息体，包含原始信息和哈希。

步骤5 落盘消息发送。

接口方法:

```
public ListenableFuture<RawMessage> transaction(RawMessage rawMessage) throws InvalidParameterException
```

参数说明

参数	类型	说明
rawMessage	RawMessage	消息体，步骤4返回的交易消息体中的原始信息。

返回值

类型	说明
ByteString	落盘结果。

5.3.2 利用合约查询数据

步骤1 查询请求消息构建

接口方法

```
public RawMessage buildInvokeRawMsg(String chainId, String name, String function, String[] args)
```

参数说明

参数	类型	说明
chainId	String	链ID
name	String	合约名称
function	String	调用合约中的方法名
args	String[]	合约方法参数

返回值

类型	说明
RawMessage	查询请求需发送的消息

步骤2 查询请求消息发送

接口方法

```
public ListenableFuture<RawMessage> invoke(RawMessage rawMessage) throws InvalidParameterException
```

参数说明

参数	类型	说明
rawMessage	RawMessage	消息体，用于查询请求

返回值

类型	说明
ListenableFuture	future对象，用于获取查询结果。

5.3.3 文件上链

接口方法

```
func (bc *BsClient) UploadFile(filePath, fileName string) (*UploadFileResponse, error)
```

参数说明

参数	类型	说明
filePath	String	待上链文件在本地路径。当前支持不大于100MB的任意格式文件
fileName	String	文件在链上的名称，不允许包含“/”。

返回值

类型	说明
UploadFileResponse	文件上链返回信息

5.3.4 文件下载

接口方法

```
func (bc *BsClient) UploadFile(filePath, fileName string) (*UploadFileResponse, error)
```

参数说明

参数	类型	说明

filePath	String	待上链文件在本地路径。当前支持不大于100MB的任意格式文件
fileName	String	文件在链上的名称，不允许包含“/”。
versionId	int	待下载文件的版本号。版本号要求大于1，可通过查询文件历史版本获取文件的版本号信息

返回值

类型	说明
error	下载成功返回为nil，反之返回error

5.3.5 组织加密

接口方法

```
func (client *GatewayClient) EncryptDataWithE2EE(consensusOrgID, encOrgID string, decOrgIDs
[]string, data string, options ...interface{}) (txID string, err error)
```

参数说明

参数	类型	说明
consensusOrgID	String	共识组织ID
encOrgID	String	执行加密操作的组织ID
decOrgIDs	[]String	除了执行加密操作的组织ID以外，可以解密该消息的组织ID列表。
data	String	需要加密的数据明文。
options	interface{}	他选项，目前支持输入一个bool类型，用于指定加密后，是否更新群组密钥。

返回值

类型	说明
txID	加密后返回密文对应的交易ID，解密时输入交易ID可以获得对应明文。

5.3.6 组织解密

接口方法

```
func (client *GatewayClient) DecryptDataWithE2EE(consensusOrgID, decOrgID string, txID string)(data string, err
error)
```

参数说明

参数	类型	说明
consensusOrgID	String	共识组织ID
decOrgIDs	[]String	除了执行加密操作的组织ID以外，可以解密该

		消息的组织ID列表。
txId	String	密文对应的交易ID

返回值

类型	说明
data	解密后的明文信息