BSN-DDC 基础网络

DDC SDK EOS 详细设计

V1.9

中移信息技术有限公司

2021 年 12 月

文档信息

项目名称：BSN-DDC	项目编号：
项目负责人：	所属部门：
编 制 人：鲍宏飞	编制时间：
审 核 人：谢兆颜	审核时间：
批 准 人：	批准时间：
版 本 号：	流 水 号：

修改记录

日期	版本	修改说明	修改者
2021.12.17	V1.0	初版编辑	鲍宏飞
2021.12.20	V1.1	1、管理费充值 api，修改接收方的账户类型为 eos 账户； 2、运营账户充值，增加了运营账户字段； 3、删除计费规则，增加了运营账户字段，业务类型字段，方法名字段。	鲍宏飞
2021.12.22	V1.2	1、补充区块查询功能介绍	鲍宏飞
2021.12.23	V1.3	1、补充签名事件&数据解析	鲍宏飞
2021.12.23	V1.4	1、补充出参字段名和字段	鲍宏飞
2021.12.31	V1.5	1、补充 SDK 使用说明； 2、附录补充区块信息示例和交易信息示例。	鲍宏飞
2021.1.14	V1.6	1、修改了错误的 sdk 接口； 2、修改了调用合约接口的合约接口名和合约入参。	鲍宏飞
2021.1.15	V1.7	1、修改销毁 api 描述	鲍宏飞
2021.1.20	V1.8	1、修改部分接口描述和事件数据解析 2、增加修改 URI 方法 3、增加跨平台授权的方法	鲍宏飞
2021.1.24	V1.9	1、更改 SDK 初始化配置方法名	鲍宏飞

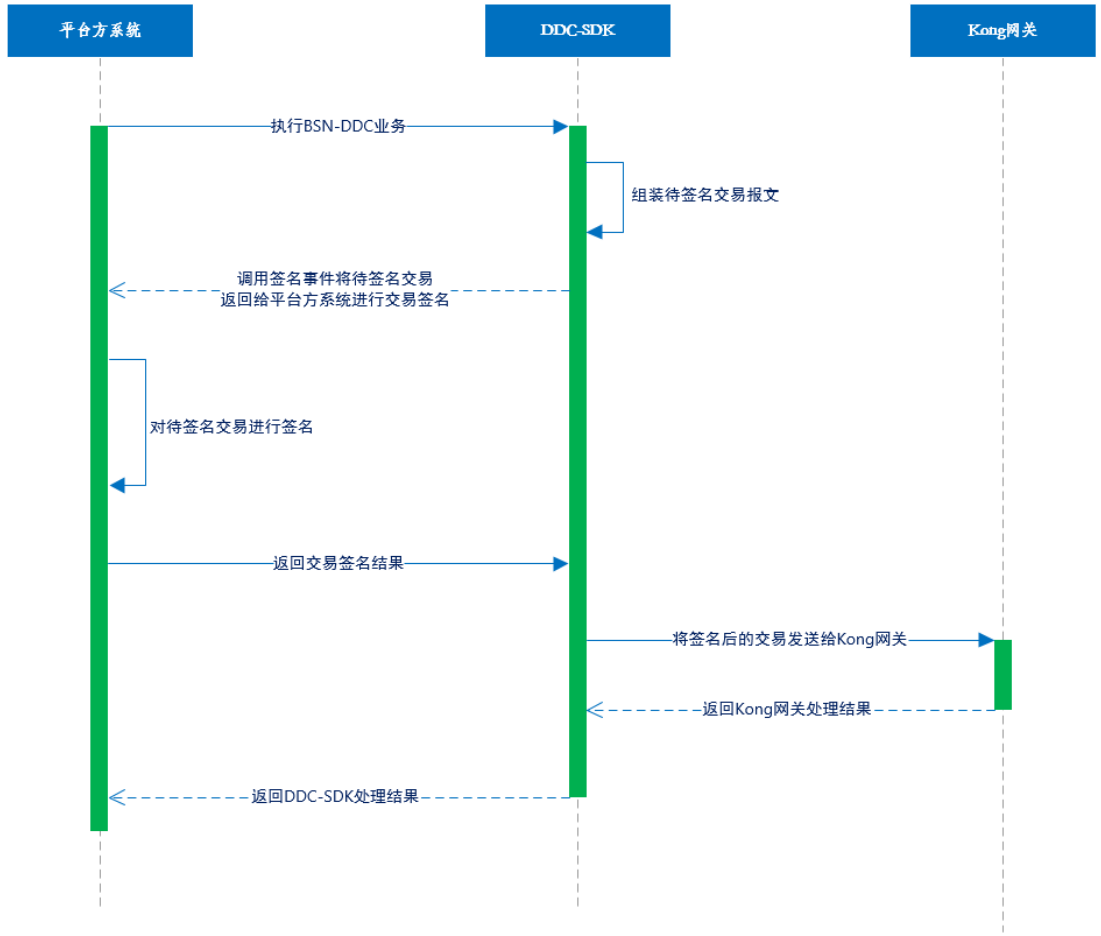
目录

1 编写目的

在《BSN-DDC EOS 合约详细设计》的基础上，为 BSN-DDC EOS 合约提供对应的 Java SDK 接口，方便运营方或平台方通过 Java SDK 实现对合约的远程调用。

2 整体设计

2.1 调用时序图



2.2 开发语言标准

目前使用 Java 语言开发 SDK。

2.3 参数格式标准

- 时间

格式为 yyyy-MM-dd HH:mm:ss 形式的字符串，例如: 2021-05-25 12:30:59 表示 2021 年 5 月 25 日 12 时 30 分 59 秒。

- 返回异常

当 SDK 处理功能逻辑出错时，会抛出相应的运行时异常，包含具体的错误信息。

3 功能设计

3.1 DDC

所有的 BSN-DDC 方法都需要调用签名事件对待签名交易进行签名，签名事件必须有业务调用者服务进行注册并实现交易签名的业务逻辑。

3.1.1 BSN-DDC-权限管理

对账户进行管理，包含了账户的添加、删除、查询及更新账户状态的操作。

3.1.1.1 查询账户

3.1.1.2.1 功能介绍

运营方或平台方通过该方法进行 DDC 账户信息的查询，上级角色可进行下级角色账户的操作。

3.1.1.2.2 API 定义

- 方法定义：AccountInfo getAccount(String account);
- EOS 合约方法：get_table_rows(name contract, name table);
- 调用者：运营方或平台方；
- 核心逻辑：验证 account 为标准 eos name 格式，并且不能为空地址；
- 所在类库：com.bsn.eos.service.DDCPermissionService#getAccount
- 输入参数：

字段名	字段	类型	必传	备注
账户地址	account	String	是	DDC 用户链账户地址

- 输出参数：

字段名	字段	类型	必传	备注
账户 DID	accountDID	String	否	DDC 账户对应的 DID 信息（普通用户可为空）
账户名称	accountName	String	是	DDC 账户对应的账户名称
账户角色	accountRole	enum	是	DDC 账户对应的身份信息。值包含: 1. Operator（运营方）； 2.PlatformManager（平台方）； 3.Consumer（用户方）。
账户上级管理者	leaderDID	String	是	DDC 账户对应的上级管理员，账户角色为 Consumer 时必填。对于普通用户 Consumer 该值为平台管理者 PlatformManager
平台管理账户状态	platformState	enum	是	DDC 账户对应的当前账户状态（仅平台方可操作该状态）。值包含：1.Frozen（冻结状态，无法进行 DDC 相关操作）； 2.Active（活跃状态，可进行 DDC 相关操作）。
运营管理账户状态	operatorState	enum	是	DDC 账户对应的当前账户状态（仅运营方可操作该状态）。值包含: 1.Frozen（冻结状态，无法进行 DDC 相关操作）； 2.Active（活跃状态，可进行 DDC 相关操作）。
冗余字段	field	String	否	冗余字段

- 调用实例：见 4.2.1.2

3.1.1.2 更新账户状态

3.1.1.3.1 功能介绍

运营方或平台方通过该方法进行 DDC 账户信息状态的更改。

3.1.1.3.2 API 定义

- 方法定义：Boolean updateAccState(String sender, String account, AccountStateEnum state, boolean changePlatformState);
- EOS 合约方法：updateaccsta(name sender, name account, PlatformState state, bool change_platform_state);
- 调用者：运营方或平台方;
- 核心逻辑：验证 account 为标准 eos name 格式，并且不能为空地址;
- 所在类库：com.bsn.eos.service.DDCPermissionService#updateAccState
- 输入参数:

字段名	字段	类型	必传	备注
运营方账户	sender	String	是	运营方账户
账户地址	account	String	是	DDC 用户链账户地址
状态	state	enum	是	1.Frozen（冻结状态，无法进行 DDC 相关操作） 2.Active（活跃状态，可进行 DDC 相关操作）
修改平台方状态	changePlatformState	Boolean	是	是否修改平台方状态(仅对运营方生效)

- 输出参数:

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.1.3

3.1.2 BSN-DDC-费用管理

3.1.2.1 充值

3.1.2.1.1 功能介绍

运营方、平台方调用该接口为所属同一方的同一级别账户或者下级账户充值;

3.1.2.1.2 API 定义

- 方法定义：Boolean recharge(String sender,String to,BigInteger amount);
- EOS 合约方法：recharge(name from, name to, asset value);
- 调用者：运营方、平台方;
- 核心逻辑:

1. 验证 from、to 为标准 eos name 格式，并且不是空地址;
2. 验证 amount 必须大于零;

- 所在类库：com.bsn.eos.service.DDCFeeService#recharge
- 输入参数:

字段名	字段	类型	必传	备注
充值账户地址	sender	String	是	充值账户地址
被充值账户地址	to	String	是	被充值账户的地址
金额	amount	BigInteger	是	转移金额

- 输出参数:

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.2.1

3.1.2.2 链账户余额查询

3.1.2.2.1 功能介绍

调用接口查询指定账户的余额。

3.1.2.2.2 API 定义

- 方法定义：String balanceOf(String account);
- EOS 合约方法：get_table_rows(name contract, name table, name account);
- 调用者：所有人;
- 核心逻辑:

1. 验证 account 为标准 eos name 格式，并且不是空地址;
2. 调用 EOS 合约的 contract 字段，是 EOS 合约的合约名;

- 所在类库：com.bsn.eos.service.DDCFeeService#balanceOf
- 输入参数:

字段名	字段	类型	必传	备注
账户地址	account	String	是	查询的账户地址

- 输出参数:

字段名	字段	类型	必传	备注
业务费余额	amount	String	是	账户所对应的业务费余额

- 调用实例：见 4.2.2.2

3.1.2.3 DDC 计费规则查询

3.1.2.3.1 功能介绍

调用接口查询指定的 DDC 业务主逻辑合约的方法所对应的调用业务费用。

3.1.2.3.2 API 定义

- 方法定义：String queryFee(String sender,uint64 business_type,String func_name);
- EOS 合约方法：get_table_rows(name contract, name table);
- 调用者：运营方、平台方，终端用户;
- 核心逻辑：验证 sender 为标准 eos name 格式，并且不是空地址;
- 所在类库：com.bsn.eos.service.DDCFeeService#queryFee
- 输入参数：

字段名	字段	类型	必传	备注
运营方账户	sender	String	是	运营方账户
业务类型	business_type	uint64_t	是	1 表示 ERC-721 2 表示 ERC-1155
方法名	func_name	String	是	EOS 合约方法

- 输出参数：

字段名	字段	类型	必传	备注
业务费	amount	BigInteger	是	查询的 DDC 合约业务费

- 调用实例：见 4.2.2.3

3.1.3 BSN-DDC-721

3.1.3.1 生成

3.1.3.1.1 功能介绍

平台方、终端用户通过调用该方法进行 DDC 的创建。

3.1.3.1.2 API 定义

- 方法定义：Boolean safeMint(String from，String to, String ddcURI);

- EOS 合约方法：mint(name sender, name to, uint64_t amount, std::string ddc_uri, uint64_t business_type, std::string memo);
- 调用者：平台方、终端用户；
- 核心逻辑：

1. 检查发送方、接收者账户地址信息是否为空；
2. 检查发送方、接收者账户地址格式是否正确；

- 所在类库：com.bsn.eos.service.DDC721Service#safeMint
- 输入参数：

字段名	字段	类型	必传	备注
发送方账户	From	String	是	
接收者账户	to	String	是	
DDC 资源标识符	ddcURI	String	否	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.3.1

3.1.3.2 DDC 授权

3.1.3.2.1 功能介绍

DDC 拥有者通过调用该方法进行 DDC 的授权。发起者需要是 ddc 的拥有方。

3.1.3.2.2 API 定义

- 方法定义：Boolean approve(String sender,String to,BigInteger ddcId);
- EOS 合约方法：approve(name sender, name to, uint64_t ddc_id, uint64_t business_type);
- 调用者：DDC 拥有者；
- 核心逻辑：

1. 检查授权者账户地址信息是否为空;
2. 检查授权者账户地址格式是否正确;
3. 检查 ddcId 是否大于 0;

- 所在类库：com.bsn.eos.service.DDC721Service#approve
- 输入参数：

字段名	字段	类型	必传	备注
发送方账户	Sender	String	是	
授权者账户	to	String	是	
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.3.5

3.1.3.3 DDC 授权查询

3.1.3.3.1 功能介绍

DDC 拥有者通过调用该方法进行 DDC 的授权查询。

3.1.3.3.2 API 定义

- 方法定义：String getApproved(String sender,String to,BigInteger ddcId);
- EOS 合约方法：get_table_rows(name contract,uint64_t primary, uint64 ddc_id);
- 调用者：DDC 拥有者;
- 核心逻辑：检查 ddcId 的值是否大于 0;
- 所在类库：com.bsn.eos.service.DDC721Service#getApproved
- 输入参数：

字段名	字段	类型	必传	备注

发送方账户	Sender	String	是	
授权者账户	to	String	是	
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
授权的账户	operator	String	是	

- 调用实例: 见 4.2.3.5

3.1.3.4 账户授权

3.1.3.4.1 功能介绍

DDC 拥有者通过调用该方法进行 DDC 的授权。发起者需要是 ddc 的拥有方。

3.1.3.4.2 API 定义

- 方法定义: Boolean setApprovalForAll(String sender,String to,bool approved);
- EOS 合约方法: approvalall(name sender, name to, bool approved, uint64_t business_type);
- 调用者: DDC 拥有者;
- 核心逻辑:

1. 检查授权者账户地址信息是否为空;
2. 检查授权者账户地址格式是否正确;

- 所在类库: com.bsn.eos.service.DDC721Service#setApprovalForAll
- 输入参数:

字段名	字段	类型	必传	备注
发送方账户	Sender	String	是	
授权者账户	to	String	是	
授权标识	approved	Boolean	是	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.3.6

3.1.3.5 账户授权查询

3.1.3.5.1 功能介绍

通过调用该方法进行 DDC 的授权查询。

3.1.3.5.2 API 定义

- 方法定义：Boolean isApprovedForAll(String sender,String to);
- EOS 合约方法：get_table_rows(name contract, name table, name account)
- 调用者：所有人；
- 核心逻辑：

1. 检查拥有者账户地址信息是否为空；
2. 检查拥有者账户地址格式是否正确；
3. 检查授权者账户地址信息是否为空；
4. 检查授权者账户地址格式是否正确；

- 所在类库：com.bsn.eos.service.DDC721Service#isApprovedForAll
- 输入参数：

字段名	字段	类型	必传	备注
发送方账户	Sender	String	是	
授权者账户	to	String	是	

- 输出参数：

字段名	字段	类型	必传	备注
授权标识	approved	Boolean	是	

- 调用实例：见 4.2.3.6

3.1.3.6 转移

3.1.3.6.1 功能介绍

DDC 拥有者或授权者通过调用该方法进行 DDC 的转移。

3.1.3.6.2 API 定义

- 方法定义：Boolean transferFrom(String from,String to,BigInteger ddcId);
- EOS 合约方法：transfer(name sender, name from, name to, uint64_t ddc_id, uint64_t amount, std::string memo, uint64_t business_type);
- 调用者：DDC 拥有者、DDC 授权者;
- 核心逻辑：

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;
3. 检查接收者账户地址信息是否为空;
4. 检查接收者账户地址格式是否正确;
5. 检查 ddcId 的数值是否大于 0;

- 所在类库：com.bsn.eos.service.DDC721Service#transferFrom
- 输入参数：

字段名	字段	类型	必传	备注
运营方账户	sender	String	是	运营方账户
拥有者账户	from	String	是	
接收者账户	to	String	是	
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注

是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.3.2

3.1.3.7 销毁

3.1.3.7.1 功能介绍

DDC 拥有者和被授权者通过调用该方法进行 DDC 的销毁。

3.1.3.7.2 API 定义

- 方法定义：Boolean burn(String sender，BigInteger ddcId);
- EOS 合约方法：burn(name sender, uint64_t ddc_id, uint64_t business_type);
- 调用者：DDC 拥有者;
- 核心逻辑：检查 ddcId 的数值是否大于 0;
- 所在类库：com.bsn.eos.service.DDC721Service#burn
- 输入参数：

字段名	字段	类型	必传	备注
发送者	sender	String	是	发送者
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.3.3

3.1.3.8 查询数量

3.1.3.8.1 功能介绍

运营方、平台方以及平台对应的终端用户通过调用该方法进行查询当前账户拥有的

DDC 的数量。

3.1.3.8.2 API 定义

- 方法定义：BigInteger balanceOf(String owner);
- EOS 合约方法：get_table_rows(name contract, name table, name account);
- 调用者：所有人;
- 核心逻辑：

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;

- 所在类库：com.bsn.eos.service.DDC721Service#balanceOf
- 输入参数：

字段名	字段	类型	必传	备注
拥有者账户	owner	String	是	

- 输出参数：

字段名	字段	类型	必传	备注
DDC 的数量	balance	BigInteger	是	

- 调用实例：见 4.2.3.4

3.1.3.9 查询拥有者

3.1.3.9.1 功能介绍

运营方、平台方以及平台对应的终端用户通过调用该方法进行查询当前 DDC 的拥有者。

3.1.3.9.2 API 定义

- 方法定义：String ownerOf(BigInteger ddcId);
- EOS 合约方法：get_table_rows(name contract, name table, uint64_t ddc_id);
- 调用者：所有人;

- 核心逻辑：检查 ddcId 的数值是否大于 0;
- 所在类库：com.bsn.eos.service.DDC721Service#ownerOf
- 输入参数：

字段名	字段	类型	必传	备注
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
拥有者账户	owner	String	是	

- 调用实例：见 4.2.3.4

3.1.3.10 获取名称

3.1.3.10.1 功能介绍

运营方、平台方以及平台对应的终端用户通过调用该方法进行查询当前 DDC 的名称。

3.1.3.10.2 API 定义

- 方法定义：String name(BigInteger ddcId);
- EOS 合约方法：get_table_rows(name contract,uint64_t primary, uint64_t ddc_id);
- 调用者：所有人;
- 所在类库：com.bsn.eos.service.DDC721Service#name
- 输入参数：

字段名	字段	类型	必传	备注
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
DDC 名称	ddcName	String	是	

- 调用实例：见 4.2.3.4

3.1.3.11 获取符号

3.1.3.11.1 功能介绍

运营方、平台方以及平台对应的终端用户通过调用该方法进行查询当前 DDC 的符号标识。

3.1.3.11.2 API 定义

- 方法定义：String symbol(BigInteger ddcId);
- EOS 合约方法：get_table_rows(name contract,uint64_t primary, uint64_t ddc_id);
- 调用者：所有人;
- 所在类库：com.bsn.eos.service.DDC721Service#symbol
- 输入参数：

字段名	字段	类型	必传	备注
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
DDC 运营方符号	symbol	String	是	DDC 运营方符号

- 调用实例：见 4.2.3.4

3.1.3.12 获取 DDCURI

3.1.3.12.1 功能介绍

运营方、平台方以及平台对应的终端用户通过调用该方法进行查询当前 DDC 的资源标识符。

3.1.3.12.2 API 定义

- 方法定义：String ddcURI(BigInteger ddcId);
- EOS 合约方法：get_table_rows(name contract,uint64_t primary, uint64_t ddc_id);;
- 调用者：所有人;
- 核心逻辑：检查 ddcId 的数值是否大于 0;
- 所在类库：com.bsn.eos.service.DDC721Service#ddcURI
- 输入参数：

字段名	字段	类型	必传	备注
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
DDC 资源标识符	ddcURI	String	是	

- 调用实例：见 4.2.3.4

3.1.3.13 设置 DDCURI

3.1.3.13.1 功能介绍

DDC 拥有者或授权者通过调用该方法对 DDC 的资源标识符进行设置。

3.1.3.13.2 API 定义

- 方法定义：Boolean setURI(String sender,String owner,BigInteger ddcId,String ddcUri);
- EOS 合约方法：seturi(name sender, name owner, uint64_t ddc_id, std::string ddc_uri, uint64_t business_type);
- 调用者：DDC 拥有者或授权者;
- 核心逻辑：

1. 检查 DDCID 对应的 DDC 资源;
2. 检查调用者身份和信息
3. 触发事件;

- 所在类库：com.bsn.eos.service.DDC721Service#setURI
- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	DDC 拥有者、授权者
拥有者	owner	String	是	拥有者账号
DDC 唯一标识	ddcId	BigInteger	是	DDC 唯一标识
DDC 资源标识符	ddcUri	String	是	DDC 资源标识符

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.3.7

3.1.4 BSN-DDC-1155

3.1.4.1 生成

3.1.4.1.1 功能介绍

平台方、终端用户通过调用该方法进行 DDC 的批量创建。

3.1.4.1.2 API 定义

- 方法定义：Boolean safeMint(String from， String to,BigInteger amount,String ddcURI);
- EOS 合约方法：mint(name sender, name to, uint64_t amount, std::string ddc_uri, uint64_t business_type, std::string memo);

- 调用者：平台方、终端用户；
- 核心逻辑：

1. 检查接收者账户地址信息是否为空；
2. 检查接收者账户地址格式是否正确；
3. 检查需要生成的 DDC 数量是否大于 0；
4. 检查 DDCURI 信息是否为空；

- 所在类库：com.bsn.eos.service.DDC1155Service#safeMint
- 输入参数：

字段名	字段	类型	必传	备注
发送方账户	From	String	是	
接收者账户	to	String	是	
DDC 数量	amount	BigInteger	是	
DDCURI	ddcURI	String	是	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.4.1

3.1.4.2 批量生成

3.1.4.2.1 功能介绍

平台方通过调用该方法进行 DDC 的批量创建。

3.1.4.2.2 API 定义

- 方法定义：Boolean safeMintBatch(String from, String to,List<BigInteger> amounts,List<String> ddcURIs);
- EOS 合约方法：mintbatch(name sender, name to, std::vector<uint64_t>

amounts, std::vector<std::string> ddc_uris, uint64_t business_type, std::string memo);

- 调用者：平台方；
- 核心逻辑：

1. 检查接收者账户地址信息是否为空；
2. 检查接收者账户地址格式是否正确；
3. 检查生成的 DDC 数量集合大小是否大于 0；
4. 检查生成的 DDC 数量集合中每个 DDC 数量是否大于 0；
5. 检查生成的 DDCURI 集合大小是否大于 0；
6. 检查生成的 DDCURI 集合中每个 DDCURI 是否为空；
7. 检查生成的 DDC 数量集合与 DDCURI 集合的大小是否相等；

- 所在类库：com.bsn.eos.service.DDC1155Service#mintBatch
- 输入参数：

字段名	字段	类型	必传	备注
发送方账户	From	String	是	
接收者账户	to	String	是	
DDC 数量集合	amounts	List<BigInteger>	是	
DDCURI 集合	ddcURIs	List<String>	是	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.4.2

3.1.4.3 账户授权

3.1.4.3.1 功能介绍

DDC 拥有者通过调用该方法进行 DDC 的授权，发起者需要是 ddc 的拥有方。

3.1.4.3.2 API 定义

- 方法定义：Boolean setApprovalForAll(String sender,String to,Boolean approved);
- EOS 合约方法：approvalall(name sender, name to, bool approved, uint64 business_type);
- 调用者：DDC 拥有者;
- 核心逻辑：

1. 检查授权者账户地址信息是否为空;
2. 检查授权者账户地址格式是否正确;

- 所在类库：com.bsn.eos.service.DDC1155Service#setApprovalForAll
- 输入参数：

字段名	字段	类型	必传	备注
发送方账户	Sender	String	是	
授权者账户	to	String	是	
授权标识	approved	Boolean	是	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.4.9

3.1.4.4 账户授权查询

3.1.4.4.1 功能介绍

DDC 拥有者通过调用该方法进行 DDC 的授权查询。

3.1.4.4.2 API 定义

- 方法定义：Boolean isApprovedForAll(String sender,String to);
- EOS 合约方法：get_table_rows(name contract, name table, name account)
- 调用者：所有人;
- 核心逻辑：

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;
3. 检查授权者账户地址信息是否为空;
4. 检查授权者账户地址格式是否正确;

- 所在类库：com.bsn.eos.service.DDC1155Service#isApprovedForAll
- 输入参数：

字段名	字段	类型	必传	备注
发送方账户	Sender	String	是	
授权者账户	to	String	是	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.4.9

3.1.4.5 转移

3.1.4.5.1 功能介绍

DDC 拥有者通过调用该方法进行 DDC 的转移。

3.1.4.5.2 API 定义

- 方法定义：Boolean transferFrom(String from,String to,BigInteger ddcId,BigInteger amount);

- EOS 合约方法：transfer(name from, name to, uint64 ddc_id, uint64 amount, string memo, uint64 business_type);
- 调用者：DDC 拥有者;
- 核心逻辑：

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;
3. 检查接收者账户地址信息是否为空;
4. 检查接收者账户地址格式是否正确;
5. 检查 DDCID 数值是否大于 0;
6. 检查 DDC 转移所对应的数量是否大于 0;

- 所在类库：com.bsn.eos.service.DDC1155Service#transferFrom
- 输入参数：

字段名	字段	类型	必传	备注
运营方账户	sender	String	是	运营方账户
拥有者账户	from	String	是	
接收者账户	to	String	是	
DDCID	ddcId	BigInteger	是	
数量	amount	BigInteger	是	DDCID 所对应的数量

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.4.3

3.1.4.6 批量转移

3.1.4.6.1 功能介绍

DDC 拥有者通过调用该方法进行 DDC 的批量转移。

3.1.4.6.2 API 定义

- 方法定义:Boolean safeBatchTransferFrom(String from,String to,List<Integer> ddclds,List<Integer>amounts);
- EOS 合约方法：batchtrans(name sender, name from, name to, std::vector<uint64_t> ddc_ids, std::vector<uint64_t> amounts, std::string memo, uint64_t business_type);
- 调用者：DDC 拥有者;
- 核心逻辑:

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;
3. 检查接收者账户地址信息是否为空;
4. 检查接收者账户地址格式是否正确;
5. 检查转移的 ddclds 集合大小是否大于 0;
6. 检查转移的 ddclds 集合中每个 DDCID 是否大于 0;
7. 检查转移的 amounts 集合中每个 DDC 数量是否大于 0;

- 所在类库：com.bsn.eos.service.DDC1155Service#safeBatchTransferFrom
- 输入参数:

字段名	字段	类型	必传	备注
运营方账户	sender	String	是	运营方账户
拥有者账户	from	String	是	
接收者账户	to	String	是	
拥有者 DDCID 集合	ddclds	List<Integer>	是	
转移数量	amounts	List<Integer>	是	

- 输出参数:

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.4.4

3.1.4.7 销毁

3.1.4.7.1 功能介绍

DDC 拥有者和被授权者通过调用该方法进行 DDC 的销毁。

3.1.4.7.2 API 定义

- 方法定义：Boolean burn(String sender，BigInteger ddcId);
- EOS 合约方法：burn(name from, uint64_t ddc_id, uint64_t business_type);
- 调用者：DDC 拥有者;
- 核心逻辑:

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;
3. 检查需要销毁的 DDCID 集合长度是否大于 0;

- 所在类库：com.bsn.eos.service.DDC1155Service#burn
- 输入参数:

字段名	字段	类型	必传	备注
发送者	sender	String	是	发送者
DDCID	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.4.5

3.1.4.8 批量销毁

3.1.4.8.1 功能介绍

DDC 拥用者通过调用该方法进行 DDC 的批量销毁。

3.1.4.8.2 API 定义

- 方法定义：Boolean burnBatch(String sender，List<BigInteger> ddcIds);
- EOS 合约方法：burnbatch(name sender，std::vector<uint64_t> ddc_ids, uint64_t business_type);
- 调用者：DDC 拥用者；
- 核心逻辑：

1. 检查拥有者账户地址信息是否为空；
2. 检查拥有者账户地址格式是否正确；
3. 检查需要销毁的 DDCID 集合大小是否大于 0；
4. 检查需要销毁的 DDCID 集合中每个 DDCID 数值是否大于 0；

- 所在类库：com.bsn.eos.service.DDC1155Service#burnBatch
- 输入参数：

字段名	字段	类型	必传	备注
发送者	sender	String	是	
DDCID 集合	ddcIds	List<BigInteger>	是	

- 输出参数：

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例：见 4.2.4.6

3.1.4.9 查询数量

3.1.4.9.1 功能介绍

运营方、平台方以及平台对应的终端用户通过调用该方法进行查询当前账户拥有的 DDC 的数量。

3.1.4.9.2 API 定义

- 方法定义：BigInteger balanceOf(String owner,BigInteger ddcId);
- EOS 合约方法：get_table_rows(name contract, name table, name account)
- 调用者：所有人;
- 核心逻辑：

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;
3. 检查 DDCID 集合长度是否大于 0;

- 所在类库：com.bsn.eos.service.DDC1155Service#balanceOf
- 输入参数：

字段名	字段	类型	必传	备注
拥有者账户	owner	String	是	
DDCID	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
数量		BigInteger	是	拥有者账户所对应的 DDCID 所拥用的数量

- 调用实例：见 4.2.4.7

3.1.4.10 批量查询数量

3.1.4.10.1 功能介绍

运营方、平台方以及平台对应的终端用户通过调用该方法进行批量查询账户拥有的 DDC 的数量。

3.1.4.10.2 API 定义

- 方法定义： List<BigInteger> balanceOfBatch(Map<String,BigInteger> ddcs);

- EOS 合约方法：get_table_rows(name contract, name table, name account);
- 调用者：所有人；
- 核心逻辑：

1. 检查 ddcs 集合大小是否大于 0；
2. 检查 ddcs 集合中拥有者账户地址信息是否为空；
3. 检查 ddcs 集合中拥有者账户地址格式是否正确；
4. 检查 ddcs 集合中每个 DDCID 数值是否大于 0；

- com.bsn.eos.service.DDC1155Service#balanceOfBatch
- 输入参数：

字段名	字段	类型	必传	备注
拥有者 DDCID 集合	ddcs	Map<String,BigInteger>	是	

- 输出参数：

字段名	字段	类型	必传	备注
数量集合		List<BigInteger>	是	拥有者账户所对应的每个 DDCID 所拥用的数量

- 调用实例：见 4.2.4.8

3.1.4.11 获取 DDCURI

3.1.4.11.1 功能介绍

运营方、平台方以及平台对应的终端用户通过调用该方法进行查询当前 DDC 的资源标识符。

3.1.4.11.2 API 定义

- 方法定义：String ddcURI(BigInteger ddcId);
- EOS 合约方法：get_table_rows(name contract,uint64_t primary, uint64_t ddc_id);

- 调用者：所有人；
- 核心逻辑：检查 DDCID 数值是否大于 0；
- 所在类库：com.bsn.eos.service.DDC1155Service#ddcURI
- 输入参数：

字段名	字段	类型	必传	备注
DDCID	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
DDC 资源标识符	ddcURI	String	是	

- 调用实例：见 4.2.4.10

3.1.4.12 设置 DDCURI

3.1.4.12.1 功能介绍

DDC 拥有者或授权者通过调用该方法对 DDC 的资源标识符进行设置。

3.1.4.12.2 API 定义

- 方法定义：Boolean setURI(String sender,String owner,BigInteger ddcId,String ddcUri);
- EOS 合约方法：seturi(name sender, name owner, uint64_t ddc_id, std::string ddc_uri, uint64_t business_type);
- 调用者：DDC 拥有者或授权者；
- 核心逻辑：

4. 检查 DDCID 对应的 DDC 资源；
5. 检查调用者身份和信息
6. 触发事件；

- 所在类库：com.bsn.eos.service.DDC1155Service#setURI
- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	DDC 拥有者、授权者
拥有者	owner	String	是	拥有者账号
DDC 唯一标识	ddcId	BigInteger	是	DDC 唯一标识
DDC 资源标识符	ddcUri	String	是	DDC 资源标识符

- 输出参数:

字段名	字段	类型	必传	备注
是否成功	isSuccess	Boolean	是	true/false

- 调用实例: 见 4.2.4.11

3.1.5 BSN-DDC-区块查询

3.1.5.1 获取区块信息

3.1.5.1.1 功能介绍

根据对应区块号，sdk 进行解析，若未提供 action，就将区块中所有 action 信息返回；若提供了 action，将区块中对应 action 中的信息返回，返回数据格式是 Java 数据对象。

3.1.5.1.2 API 定义

- 方法定义: Object getBlockInfo(BigInteger blockNum,List<String> actionList);
- EOS RPC 方法: cleos -u https://api.testnet.eos.io get block 48351112;
- 调用者: 所有人;
- 核心逻辑: 根据对应区块号，sdk 进行解析，若未提供 action，就将区块中所有 action 信息返回；若提供了 action，将区块中对应 action 中的信息返回，返回数据格式是 Java 数据对象。
- 输入参数:

字段名	字段	类型	必传	备注

区块号	blockNum	BigInteger	是	
Action 列表	actionList	List<String>	是	传入空的 List，返回合约中所有 Action 信息

- 输出参数：根据不同的 Action，返回不同的数据对象，具体 action 对应的事件数据，见 3.1.7 节 BSC-DDC-数据解析。

3.1.6 BSN-DDC-签名事件

3.1.6.1 功能介绍

此事件是通用事件，所有的上链待签名交易报文需调用此事件进行签名，业务调用方需要注册此签名事件，并在实现的签名事件中实现签名逻辑，并将最终签名后的结果返回给 DDC-SDK。

3.1.6.2 事件定义

- 输入参数：签名事件类
- 输出参数：签名结果
- String signEvent(SignEvent event)。

3.1.7 BSN-DDC-数据解析

3.1.7.1 权限数据

3.1.7.1.1 添加账户

3.1.7.1.1.1 功能说明

用于对 BSN-DDC-权限合约进行添加账户所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.1.1.2 合约事件

- addaccount(name sender, account_info info)

3.1.7.1.1.3 数据结构

字段名	字段	类型	备注
区块链账户名	account	name	DDC 用户链账户名
账户 DID	account_did	string	DDC 账户对应的 DID 信息（普通用户可为空）
账户名称	account_name	string	DDC 账户对应的账户名
账户上级管理者	leader_did	string	DDC 账户对应的上级管理员，账户角色为 Consumer 时必填。对于普通用户 Consumer 该值为平台管理者 PlatformManager
冗余字段	field	string	冗余字段

3.1.7.1.2 更新账户状态

3.1.7.1.2.1 功能说明

用于对 BSN-DDC-权限合约进行更新账户状态所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.1.2.2 合约事件

- updateacc(name sender,name account, PlatformState state, bool change_platform_state)

3.1.7.1.2.3 数据结构

字段名	字段	类型	备注
调用者	sender	String	是

区块链账户名	account	name	DDC 用户链账户名
平台管理账户状态	state	enum	DDC 账户对应的当前账户状态（仅平台方可操作该状态）。值包含: 1．Frozen（冻结状态，无法进行 DDC 相关操作）； 2．Active（活跃状态，可进行 DDC 相关操作）。
运营管理账户状态	change_platform_state	enum	DDC 账户对应的当前账户状态（仅运营方可操作该状态）。值包含: 1．Frozen（冻结状态，无法进行 DDC 相关操作）； 2．Active（活跃状态，可进行 DDC 相关操作）。

3.1.7.2 充值数据

3.1.7.2.1 充值

3.1.7.2.1.1 功能说明

用于对 BSN-DDC-计费合约进行充值所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.2.1.2 合约事件

- recharge(name from, name to, asset value);

3.1.7.2.1.3 数据结构

字段名	字段	类型	备注
充值账户地址	from	String	充值账户地址

被充值账户地址	to	String	被充值账户的地址
充值额度	value	Asset	充值额度

3.1.7.2.2 DDC 业务费扣除

3.1.7.2.2.1 功能说明

用于对 BSN-DDC-计费合约进行 DDC 业务费扣除所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.2.2.2 合约事件

- receiptpay(name account, BusinessType business_type, name func_name, asset fee, asset balance);

3.1.7.2.2.3 数据结构

字段名	字段	类型	备注
支付人	account	name	支付人账号
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155
调用方法	func_name	name	调用方法
调用花费手续费	fee	asset	调用花费手续费
调用后余额	balance	asset	调用后余额

3.1.7.2.3 设置 DDC 计费规则

3.1.7.2.3.1 功能说明

用于对 BSN-DDC-计费合约进行设置 DDC 计费规则所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.2.3.2 合约事件

- setfee(name sender, BusinessType business_type, name func_name, asset value);

3.1.7.2.3.3 数据结构

字段名	字段	类型	备注
运营方账户	sender	name	运营方账户
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155
方法名	func_name	String	EOS 合约方法
业务费用	value	BigInteger	

3.1.7.2.4 删除 DDC 计费规则

3.1.7.2.4.1 功能说明

用于对 BSN-DDC-计费合约进行删除 DDC 计费规则所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.2.4.2 合约事件

- deletefee(name sender, BusinessType business_type, name func_name);

3.1.7.2.4.3 数据结构

字段名	字段	类型	备注
运营方账户	sender	name	运营方账户
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155
方法名	func_name	String	EOS 合约方法

3.1.7.2.5 按合约删除 DDC 计费规则

3.1.7.2.5.1 功能说明

用于对 BSN-DDC-计费合约进行按合约删除 DDC 计费规则所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.2.5.2 合约事件

- deleteddc(name sender, BusinessType business_type)

3.1.7.2.5.3 数据结构

字段名	字段	类型	备注
运营方账户	sender	name	运营方账户
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.3 BSN-DDC-721 数据

3.1.7.3.1 生成

3.1.7.3.1.1 功能说明

用于对 BSN-DDC-721 业务主逻辑合约进行 DDC 生成所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.3.1.2 合约事件

- mint(name sender, name to, uint64_t amount, std::string ddc_uri, uint64_t business_type, std::string memo);
- receiptmint(name sender, name to, uint64_t ddc_id, std::string ddc_uri, uint8_t allowed, uint64_t amount, uint64_t business_type, std::string ddc_name, std::string ddc_symbol);

3.1.7.3.1.3 数据结构

字段名	字段	类型	备注
发送方账户	Sender	String	
接收者账户	to	String	
发送数量	amount	uint64_t	ERC-721 忽略此数据
DDC 资源标识符	ddcURI	String	可为空
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155
留言	memo	String	

字段名	字段	类型	备注
发送方账户	Sender	String	
接收者账户	to	String	
DDC 唯一标识	ddc_id	uint64_t	
DDC 资源标识符	Ddc_uri	String	
是否进行授权	allowed	uint8_t	
增发数量	amount	uint8_t	
DDC 类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155
DDC 名称	ddc_name	String	
DDC 符号	ddc_symbol	String	

3.1.7.3.2 转移

3.1.7.3.2.1 功能说明

用于对 BSN-DDC-721 业务主逻辑合约进行 DDC 转移/安全转移所产生的区块中的事件

进行解析，并组装成所对应的数据结构。

3.1.7.3.2.2 合约事件

- transfer(name sender, name from, name to, uint64_t ddc_id, uint64_t amount, std::string memo, uint64_t business_type);

3.1.7.3.2.3 数据结构

字段名	字段	类型	备注
调用者	sender	name	运营方账户
转出者账户	from	String	
接收者账户	to	String	
DDC 唯一标识	ddcId	uint64_t	
数量	amount	uint64_t	
留言	memo	String	
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.3.3 冻结

3.1.7.3.3.1 功能说明

用于对 BSN-DDC-721 业务主逻辑合约进行 DDC 解冻所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.3.3.2 合约事件

- freeze(name sender, uint64 ddc_id, uint64 business_type)

3.1.7.3.3.3 数据结构

字段名	字段	类型	备注
发送者	sender	name	发送者
DDC 唯一标识	ddcId	BigInteger	DDC 唯一标识
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.3.4 解冻

3.1.7.3.4.1 功能说明

用于对 BSN-DDC-721 业务主逻辑合约进行 DDC 解冻所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.3.4.2 合约事件

- unfreeze(name sender, uint64 ddc_id, uint64 business_type)

3.1.7.3.4.3 数据结构

字段名	字段	类型	备注
发送者	sender	name	发送者
DDC 唯一标识	ddcId	BigInteger	DDC 唯一标识
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.3.5 销毁

3.1.7.3.5.1 功能说明

用于对 BSN-DDC-721 业务主逻辑合约进行 DDC 销毁所产生的区块中的事件进行解析，

并组装成所对应的数据结构。

3.1.7.3.5.2 合约事件

- burn(name sender, name owner, uint64_t ddc_id, uint64_t business_type);

3.1.7.3.5.3 数据结构

字段名	字段	类型	备注
发送者	sender	name	发送者
拥有者	owner	name	拥有者
DDC 唯一标识	ddcId	uint64_t	DDC 唯一标识
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.4 BSN-DDC-1155 数据

3.1.7.4.1 生成

3.1.7.4.1.1 功能说明

用于对 BSN-DDC-1155 业务主逻辑合约进行 DDC 生成所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.4.1.2 合约事件

- mint(name sender, name to, uint64_t amount, std::string ddc_uri, uint64_t business_type, std::string memo);
- receiptmint(name sender, name to, uint64_t ddc_id, std::string ddc_uri, uint8_t allowed, uint64_t amount, uint64_t business_type, std::string ddc_name, std::string ddc_symbol);

3.1.7.4.1.3 数据结构

字段名	字段	类型	备注
发送方账户	sender	String	
接收者账户	to	String	
发送数量	amount	uint64_t	ERC-721 忽略此数据
DDC 资源标识符	ddcURI	String	可为空
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155
留言	memo	String	

字段名	字段	类型	备注
发送方账户	Sender	String	
接收者账户	to	String	
DDC 唯一标识	ddc_id	uint64_t	
DDC 资源标识符	Ddc_uri	String	
是否进行授权	allowed	uint8_t	
增发数量	amount	uint8_t	
DDC 类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155
DDC 名称	ddc_name	String	
DDC 符号	ddc_symbol	String	

3.1.7.4.2 批量生成

3.1.7.4.2.1 功能说明

用于对 BSN-DDC-1155 业务主逻辑合约进行 DDC 批量生成所产生的区块中的事件进行

解析，并组装成所对应的数据结构。

3.1.7.4.2.2 合约事件

- mintbatch(name sender, name to, std::vector<uint64_t> amounts, std::vector<std::string> ddc_uris, uint64_t business_type, std::string memo);
- receiptmint(name sender, name to, uint64_t ddc_id, std::string ddc_uri, uint8_t allowed, uint64_t amount, uint64_t business_type, std::string ddc_name, std::string ddc_symbol);

3.1.7.4.2.3 数据结构

字段名	字段	类型	备注
发送方账户	Sender	String	
接收者账户	to	String	
DDC 数量集合	amounts	List<uint64_t>	
DDCURI 集合	ddcURIs	List<String>	可为空
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155
留言	memo	String	

字段名	字段	类型	备注
发送方账户	Sender	String	
接收者账户	to	String	
DDC 唯一标识	ddc_id	uint64_t	
DDC 资源标识符	Ddc_uri	String	
是否进行授权	allowed	uint8_t	
增发数量	amount	uint8_t	
DDC 类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

DDC 名称	ddc_name	String	
DDC 符号	ddc_symbol	String	

3.1.7.4.3 转移

3.1.7.4.3.1 功能说明

用于对 BSN-DDC-1155 业务主逻辑合约进行 DDC 安全转移所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.4.3.2 合约事件

- transfer(name sender, name from, name to, uint64_t ddc_id, uint64_t amount, std::string memo, uint64_t business_type);

3.1.7.4.3.3 数据结构

字段名	字段	类型	备注
调用者	sender	name	运营方账户
转出者账户	from	String	
接收者账户	to	String	
DDC 唯一标识	ddcId	uint64_t	
数量	amount	uint64_t	
留言	memo	String	
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.4.4 批量转移

3.1.7.4.4.1 功能说明

用于对 BSN-DDC-1155 业务主逻辑合约进行 DDC 批量安全转移所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.4.4.2 合约事件

- batchtrans(name sender, name from, name to, std::vector<uint64_t> ddc_ids, std::vector<uint64_t> amounts, std::string memo, uint64_t business_type);

3.1.7.4.4.3 数据结构

字段名	字段	类型	备注
调用者	sender	name	运营方账户
转出者账户	from	String	
接收者账户	to	String	
DDC 标识符集合	ddcId	List< uint64_t >	
转移数量集合	amount	List< uint64_t >	
留言	memo	String	
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.4.5 冻结

3.1.7.4.5.1 功能说明

用于对 BSN-DDC-1155 业务主逻辑合约进行 DDC 冻结所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.4.5.2 合约事件

- freeze(name sender, uint64 ddc_id, uint64 business_type)

3.1.7.4.5.3 数据结构

字段名	字段	类型	备注
发送者	sender	name	发送者
DDC 唯一标识	ddcId	BigInteger	DDC 唯一标识
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.4.6 解冻

3.1.7.4.6.1 功能说明

用于对 BSN-DDC-1155 业务主逻辑合约进行 DDC 解冻所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.4.6.2 合约事件

- unfreeze(name sender, uint64 ddc_id, uint64 business_type)

3.1.7.4.6.3 数据结构

字段名	字段	类型	备注
发送者	sender	name	发送者
DDC 唯一标识	ddcId	BigInteger	DDC 唯一标识
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.4.7 销毁

3.1.7.4.7.1 功能说明

3.1.7.4.7.1 功能说明

用于对 BSN-DDC-1155 业务主逻辑合约进行 DDC 销毁所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.4.7.2 合约事件

- burn(name sender, name owner, uint64_t ddc_id, uint64_t business_type);

3.1.7.4.7.3 数据结构

字段名	字段	类型	备注
发送者	sender	name	发送者
拥有者	owner	name	拥有者
DDC 唯一标识	ddcId	uint64_t	DDC 唯一标识
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

3.1.7.4.8 批量销毁

3.1.7.4.8.1 功能说明

用于对 BSN-DDC-1155 业务主逻辑合约进行 DDC 批量销毁所产生的区块中的事件进行解析，并组装成所对应的数据结构。

3.1.7.4.8.2 合约事件

- burnbatch(name sender, name owner, std::vector<uint64_t> ddc_ids, uint64_t business_type);

3.1.7.4.8.3 数据结构

字段名	字段	类型	备注
发送者	sender	name	发送者
拥有者	owner	name	拥有者
DDC 标识集合	ddcIds	List<uint64_t>	DDC 唯一标识集合
业务类型	business_type	uint64_t	1 表示 ERC-721 2 表示 ERC-1155

4 SDK 使用说明

4.1 初始化配置

通过 com.bsn.eos.chain.ChainConfig 类中，完成链地址，合约帐号和密钥的配置；
1. 通过 com.bsn.eos.chain.ChainConfig#setGatewayUrl 接口，配置 EOS 链地址；
2. 通过 com.bsn.eos.chain.ChainConfig#setGatewayApiKey 配置合约账户私钥；
3. 通过 com.bsn.eos.chain.ChainConfig#setDdcContractAndAccount 配置布署合约账户名。

4.2 API 调用

4.2.1 BSN-DDC-权限管理

4.2.1.1 添加运营账户

调用方法实例：

```
/**
 * 添加运营账户
 */
@Test
public void addOperator() {
    DDCPermissionService ddcPermissionService = new DDCPermissionServiceImpl();
    boolean res = ddcPermissionService.addOperator(accountList.get(0), accountList.get(0),
didList.get(0));
```

```
    System.out.println(res);
    getAllAccount();
}
```

4.2.1.2 查询账户

调用方法实例：

```
/**
 * DDC 用户链账户查询
 */
@Test
public void getAccount() {
    DDCPermissionService ddcPermissionService = new DDCPermissionServiceImpl();
    System.out.println(ddcPermissionService.getAccount(accountList.get(0)));
}
```

4.2.1.3 更新账户状态

调用方法实例：

```
/**
 * 更新账户状态
 */
@Test
public void updateAccState() {
    DDCPermissionService ddcPermissionService = new DDCPermissionServiceImpl();
    ddcPermissionService.updateAccState(accountList.get(0), accountList.get(2),
AccountStateEnum.Frozen, true);
    System.out.println(ddcPermissionService.getAccount(accountList.get(2)));
    ddcPermissionService.updateAccState(accountList.get(0), accountList.get(2),
AccountStateEnum.Active, true);
    System.out.println(ddcPermissionService.getAccount(accountList.get(2)));
}
```

4.2.2 BSN-DDC-费用管理

4.2.2.1 充值

调用方法实例：

```
/**
 * 充值与链账户余额查询
 */
```

```
@Test
public void recharge() {
    DDCFeeService ddcFeeService = new DDCFeeServiceImpl();
    ddcFeeService.recharge(accountList.get(0), accountList.get(1), "0.0001 FEE");
    System.out.println(ddcFeeService.balanceOf(accountList.get(1)));
}
```

4.2.2.2 链账户余额查询

调用方法实例:

```
/**
 * 链账户余额查询
 */
@Test
public void balanceOf() {
    DDCFeeService ddcFeeService = new DDCFeeServiceImpl();
    System.out.println(ddcFeeService.balanceOf(accountList.get(0)));
}
```

4.2.2.3 查询 DDC 计费规则

调用方法实例:

```
/**
 * 查询 DDC 计费规则
 */
@Test
public void queryFee(){
    DDCFeeService ddcFeeService=new DDCFeeServiceImpl();
    System.out.println(ddcFeeService.queryFee(accountList.get(0), BusinessTypeEnum.ERC721,
"funcname2"));
}
```

4.2.3 BSN-DDC-721

4.2.3.1 生成

调用方法实例

```
/**
 * 721 生成
 */
@Test
```

```
public void mint721() {
    DDC721Service ddc721Service = new DDC721ServiceImpl();
    ddc721Service.mint(accountList.get(1), accountList.get(1), "https://bitnodes.io/000");
}
```

4.2.3.2 转移

调用方法实例：

```
/**
 * 721 转移
 */
@Test
public void transferFrom721() {
    DDC721Service ddc721Service = new DDC721ServiceImpl();
    ddc721Service.transferFrom(accountList.get(0), accountList.get(1), accountList.get(1),
BigInteger.valueOf(1));
}
```

4.2.3.3 销毁

调用方法实例：

```
/**
 * 721 销毁
 */
@Test
public void burn721() {
    DDC721Service ddc721Service = new DDC721ServiceImpl();
    ddc721Service.burn(accountList.get(0), BigInteger.valueOf(2));
}
```

4.2.3.4 查询数量、拥有者、名称、符号、URI

调用方法实例：

```
/**
 * 721 查询数量、查询拥有者
 */
@Test
public void balanceOf721() {
    DDC721Service ddc721Service = new DDC721ServiceImpl();
    System.out.println(ddc721Service.balanceOf(accountList.get(1)));
    System.out.println(ddc721Service.ownerOf(BigInteger.valueOf(0)));
    System.out.println(ddc721Service.ownerOf(BigInteger.valueOf(1)));
```

```
    System.out.println(ddc721Service.name(BigInteger.valueOf(2)));
    System.out.println(ddc721Service.symbol(BigInteger.valueOf(2)));
    System.out.println(ddc721Service.ddcURI(BigInteger.valueOf(2)));
}
```

4.2.3.5 DDC 授权及查询

方法调用实例：

```
/**
 * 721 DDC 授权及查询
 */
@Test
public void approve721() {
    DDC721Service ddc721Service = new DDC721ServiceImpl();
    ddc721Service.approve(accountList.get(1), accountList.get(2), BigInteger.valueOf(1));
    System.out.println(ddc721Service.getApproved(accountList.get(1), accountList.get(2),
BigInteger.valueOf(1)));
}
```

4.2.3.6 账户授权及查询

调用方法实例：

```
/**
 * 721 账户授权及查询
 */
@Test
public void approveall721() {
    DDC721Service ddc721Service = new DDC721ServiceImpl();
    ddc721Service.setApprovalForAll(accountList.get(0), accountList.get(1), true);

    TableRowsReq tableRowsReq = new TableRowsReq(ChainConfig.ddcContractAndAccount,
ChainConfig.s21userapprTable, ChainConfig.ddcContractAndAccount);
    TableRows rows = ChainUtil.getInstance().getTableRows(tableRowsReq);
    System.out.println(JSON.toJSONString(rows, true));

    System.out.println(ddc721Service.isApprovedForAll(accountList.get(0), accountList.get(2)));
    System.out.println(ddc721Service.isApprovedForAll(accountList.get(0), accountList.get(3)));

}
```

4.2.3.7 设置 DDCURI

调用方法实例:

```
@Test
public void setUri() {
    DDC721Service ddc721Service = new DDC721ServiceImpl();
    ddc721Service.setURI(accountList.get(6),accountList.get(6),
BigInteger.valueOf(9),"https://bitnodes.io/2229");
    TableRowsReq tableRowsReq = new TableRowsReq(ChainConfig.ddcContractAndAccount,
ChainConfig.s21infoTable, ChainConfig.ddcContractAndAccount);
    TableRows rows = ChainUtil.getInstance().getTableRows(tableRowsReq);
    System.out.println(JSON.toJSONString(rows, true));

    TableRowsReq tableRowsReq2 = new TableRowsReq(ChainConfig.ddcContractAndAccount,
ChainConfig.s21balanceTable, ChainConfig.ddcContractAndAccount);
    TableRows rows2 = ChainUtil.getInstance().getTableRows(tableRowsReq2);
    System.out.println(JSON.toJSONString(rows2, true));
}
```

4.2.4 BSN-DDC-1155

4.2.4.1 生成

调用方法实例

```
/**
 * 1155 生成
 */
@Test
public void mint1155() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    ddc1155Service.mint(accountList.get(2), accountList.get(2), BigInteger.valueOf(1),
"https://bitnodes.io/222");
}
```

4.2.4.2 批量生成

调用方法实例

```
@Test
public void mintBatch() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    ddc1155Service.mintBatch(accountList.get(2), accountList.get(2),
```

```
Arrays.asList(BigInteger.valueOf(2), BigInteger.valueOf(2)), Arrays.asList("https://bitnodes.io/222",
"https://bitnodes.io/222"));
}
```

4.2.4.3 转移

调用方法实例：

```
@Test
public void transferFrom1155() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    ddc1155Service.transferFrom(accountList.get(2), accountList.get(2), accountList.get(3),
BigInteger.valueOf(2), BigInteger.valueOf(1));
}
```

4.2.4.4 批量转移

调用方法实例：

```
@Test
public void safeBatchTransferFrom() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    ddc1155Service.safeBatchTransferFrom(accountList.get(2), accountList.get(2), accountList.get(3),
Arrays.asList(2, 3), Arrays.asList(1, 1));
}
```

4.2.4.5 销毁

调用方法实例：

```
@Test
public void burn1155() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    ddc1155Service.burn(accountList.get(0), BigInteger.valueOf(2));
}
```

4.2.4.6 批量销毁

调用方法实例：

```
@Test
public void burnBatch1155() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    ddc1155Service.burnBatch(accountList.get(0), Arrays.asList(BigInteger.ONE, BigInteger.valueOf(2)));
}
```

4.2.4.7 查询数量

调用方法实例：

```
@Test
public void balanceOf1155() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    System.out.println(ddc1155Service.ddcURI(BigInteger.valueOf(0)));
    System.out.println(ddc1155Service.balanceOf(accountList.get(1), BigInteger.valueOf(0)));
}
```

4.2.4.8 批量查询数量

方法调用实例：

```
@Test
public void balanceOfBatch1155() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    Map ddcs = new HashMap();
    ddcs.put(accountList.get(2), BigInteger.ONE);
    ddcs.put(accountList.get(2), BigInteger.valueOf(2));
    System.out.println(ddc1155Service.balanceOfBatch(ddcs));
}
```

4.2.4.9 账户授权及查询

调用方法实例：

```
/**
 * 1155 账户授权及查询
 */
@Test
public void approveall1155() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    ddc1155Service.setApprovalForAll(accountList.get(2), accountList.get(3), true);

    TableRowsReq tableRowsReq = new TableRowsReq(ChainConfig.ddcContractAndAccount,
ChainConfig.t1155userapprTable, ChainConfig.ddcContractAndAccount);
    TableRows rows = ChainUtil.getInstance().getTableRows(tableRowsReq);
    System.out.println(JSON.toJSONString(rows, true));

    System.out.println(ddc1155Service.isApprovedForAll(accountList.get(2), accountList.get(3)));
    System.out.println(ddc1155Service.isApprovedForAll(accountList.get(2), accountList.get(4)));

}
```

4.2.4.10 获取 DDCURI

调用方法实例：

```
@Test
public void balanceOf1155() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    System.out.println(ddc1155Service.ddcURI(BigInteger.valueOf(0)));
    System.out.println(ddc1155Service.balanceOf(accountList.get(1), BigInteger.valueOf(0)));
}
```

4.2.4.11 设置 DDCURI

调用方法实例：

```
@Test
public void setUri() {
    DDC1155Service ddc1155Service = new DDC1155ServiceImpl();
    ddc1155Service.setURI(accountList.get(6),accountList.get(6),
BigInteger.valueOf(9),"https://bitnodes.io/2229");
    TableRowsReq tableRowsReq = new TableRowsReq(ChainConfig.ddcContractAndAccount,
ChainConfig.s21infoTable, ChainConfig.ddcContractAndAccount);
    TableRows rows = ChainUtil.getInstance().getTableRows(tableRowsReq);
    System.out.println(JSON.toJSONString(rows, true));

    TableRowsReq tableRowsReq2 = new TableRowsReq(ChainConfig.ddcContractAndAccount,
ChainConfig.s21balanceTable, ChainConfig.ddcContractAndAccount);
    TableRows rows2 = ChainUtil.getInstance().getTableRows(tableRowsReq2);
    System.out.println(JSON.toJSONString(rows2, true));
}
```

5 附录

5.1 区块信息示例

- **EOS**

参考

https://bloks.io/block/223545017

5.2 交易信息示例

- **EOS**

参考

https://bloks.io/transaction/799a4a9271b67329dd8e49eea882c0bbb1dc704c2fa9686600b934db5bffadb1?tab=raw

```
- {
    "status": "executed",
    "cpu_usage": 1174,
    "net_usage": 144,
    "id": "799a4a9271b67329dd8e49eea882c0bbb1dc704c2fa9686600b934db5bffadb1",
    "block_time": "2021-12-31T07:57:47.500",
    "block_num": 223545017,
    "delay_sec": undefined,
    "expiration": undefined,
  -  "actions": [
   -    {
     -      "account_ram_deltas": [
            ],
     -      "act": {
               "account": "push.sx",
      +        "authorization": [ /* 2 items */ ],
      +        "data": { /* 2 items */ },
               "hex_data": "40c255d3495d51c70500000000000000",
               "name": "mine"
            },
            "action_ordinal": 1,
            "block_num": 223545017,
            "block_time": "2021-12-31T07:57:47.500",
            "closest_unnotified_ancestor_action_ordinal": 0,
            "context_free": false,
            "creator_action_ordinal": 0,
            "elapsed": 257,
            "producer_block_id": "0d5306b91066579c88425d869d5e3380b3b44ba0f3a98fbed6a87ec0c3cc11e6",
     -      "receipt": {
               "abi_sequence": 29,
               "act_digest": "0f5509389f27e7c95c8c805ddd437468d5037c7b6e41033d27a822a50e060cf6",
      +        "auth_sequence": [ /* 2 items */ ],
               "code_sequence": 162,
               "global_sequence": 355140632570,
               "receiver": "push.sx",
               "recv_sequence": 131468092
            },
            "receiver": "push.sx",
            "trx_id": "799a4a9271b67329dd8e49eea882c0bbb1dc704c2fa9686600b934db5bffadb1",
     -      "inline_traces": [
      +        { /* 14 items */ },
      +        { /* 13 items */ },
```