NFT Marketplace架构分享

背景介绍

2020年是DeFi元年

2021年是NFT元年

...

1. **数字资产的兴起与需求**
2. **去中心化的市场需求**
3. **版权保护和二级市场**
4. **全球化市场的潜力**
5. **技术与金融的融合**

项目意义

当前的NFT交易市场不仅是一个**基于区块链的应用**，也是**链上技术与链下服务高度结合**的典型范例。通过项目的设计和开发，可以探索如何将区块链的去中心化、透明性、不可篡改等特点与传统的链下业务流程进行有机融合，创建一个灵活、可扩展的系统架构，不仅服务于NFT交易市场，**还能支持未来其他潜在的链上应用，如Bitcoin上的铭文、符文等新兴数字资产**。以下从几个关键角度说明项目的深远意义：

1. **技术架构的通用性和可扩展性**
2. **链上技术原理与链下服务的结合**
3. **去中心化应用的场景扩展**

NFT基本概念

特性	ERC20	ERC721	ERC1155
类型	同质化代币	非同质化代币	可同时支持同质化和非同质化代币
用途	货币、奖励代币、稳定币等	数字收藏品、NFT、游戏资产等	混合资产模型，支持多种类型代币

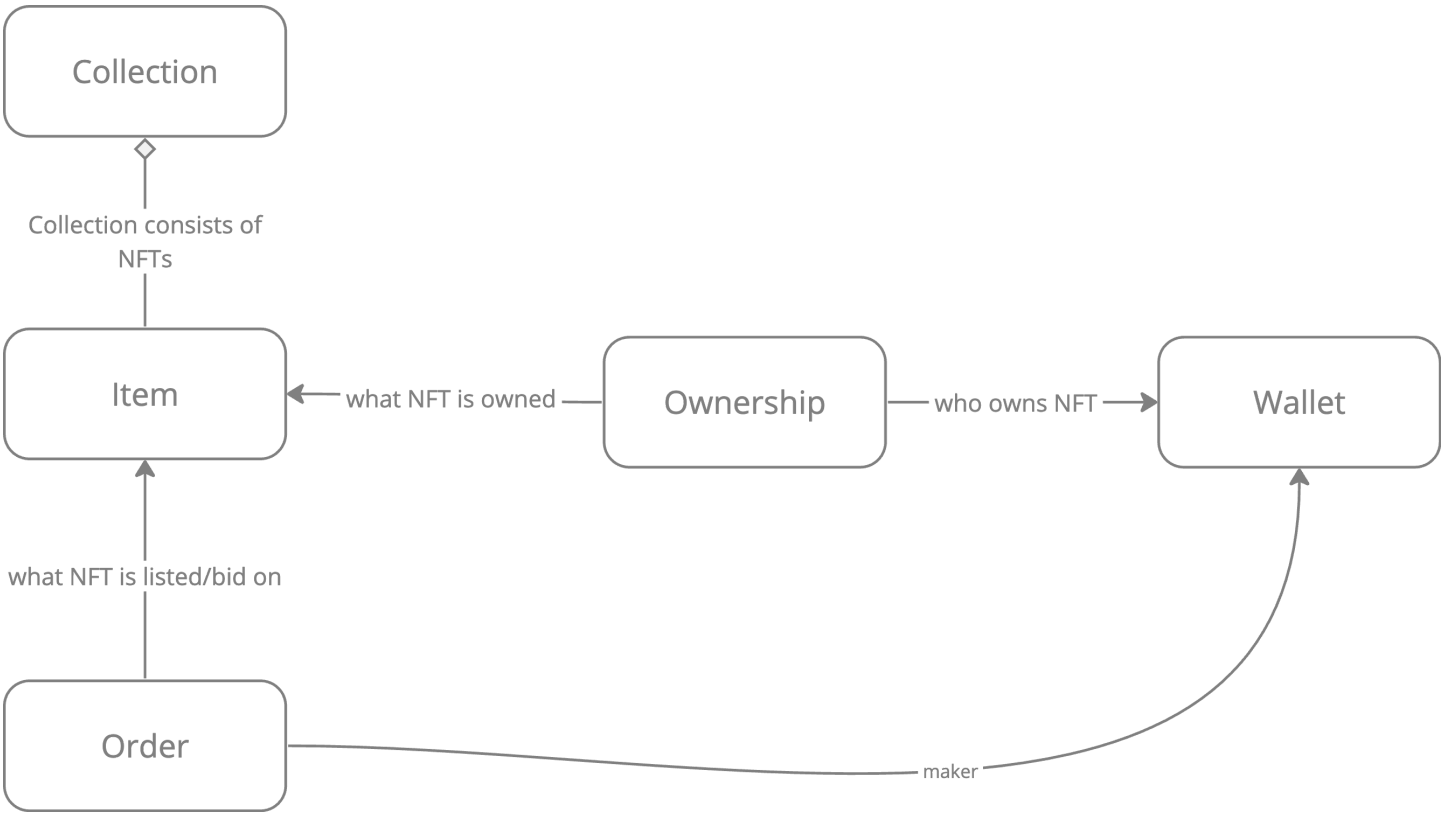
代币的唯一性	代币之间没有区别，每个代币都是相同的	每个代币都是独特的、不可替代的	同时支持同质化和非同质化代币，灵活性高
代币ID	没有代币ID，每个代币相同	**每个代币都有唯一的ID**	不同ID的代币可以代表不同的资产
批量转移	不支持批量转移，需逐个发送	**不支持批量转移，需逐个发送**	支持批量转移，减少Gas费用
适用场景	同质化资产，如货币、投票权	数字艺术品、NFT、唯一资产	游戏资产、数字商品、虚拟货币等
数据结构复杂性	简单，存储每个地址的代币余额	**复杂，每个代币有独立的元数据**	复杂，支持多种类型资产，灵活性强
转移成本	低	较高，因为每个代币都是唯一的	低，可以同时转移多个代币
标准接口函数	`balanceOf`，`transfer`，`approve`等	`ownerOf`，`transferFrom`，`approve`等	`balanceOf`，`safeTransferFrom`，`batchTransfer`等
智能合约交互	简单，同质化	复杂，需要处理每个代币的唯一性	相对复杂，但提供更高的灵活性

NFT的核心操作

详见：https://eips.ethereum.org/EIPS/eip-721

1. **transfer**
   a. `safeTransferFrom(address _from, address _to, uint256 _tokenId)`
   b. `transferFrom(address _from, address _to, uint256 _tokenId)`
2. **approve**
   a. `approve(address _approved, uint256 _tokenId)`
   b. `setApprovalForAll(address _operator, bool _approved)`

NFT数据模型



Collection ——— NFT 集合的实体

Item ——— 代表交易系统中代表NFT的实体

Ownership ——— 代表NFT的所有权，也就是Item的Owner， 即Item和Wallet的关联关系

Order ——— 代表出售或购买NFT意愿的实体。

Activity ——— 代表 NFT 状态下发生的事件：mint, transfer, list, buy等

NFT交易模式

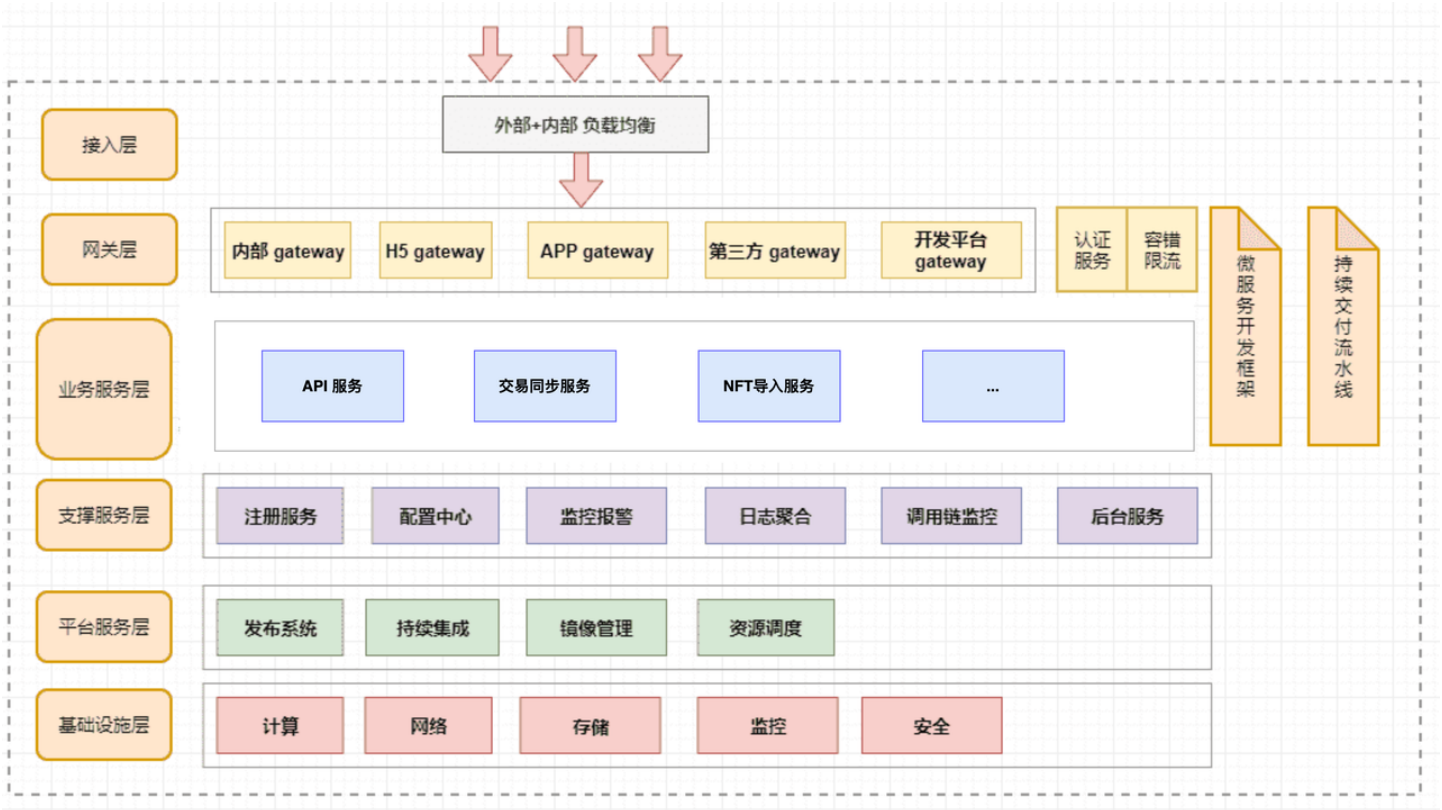
1. **NFT订单在链下**: 非dex
2. **NFT订单在链上**:dex

订单簿 OrderBook: Maker, Taker: 用户; 价格确定于订单

做市商 AMM: ERC721——AMM: Maker, Taker: 一方是池子, 一方是用户; 价格是随池子变化的;

项目描述

架构图



核心交易模块 （链上）： EasySwapContract

API服务：EasySwapBackend

交易同步服务：EasySwapSync

NFT导入服务

订单中继服务

定时任务模块

微服务功能描述

API服务

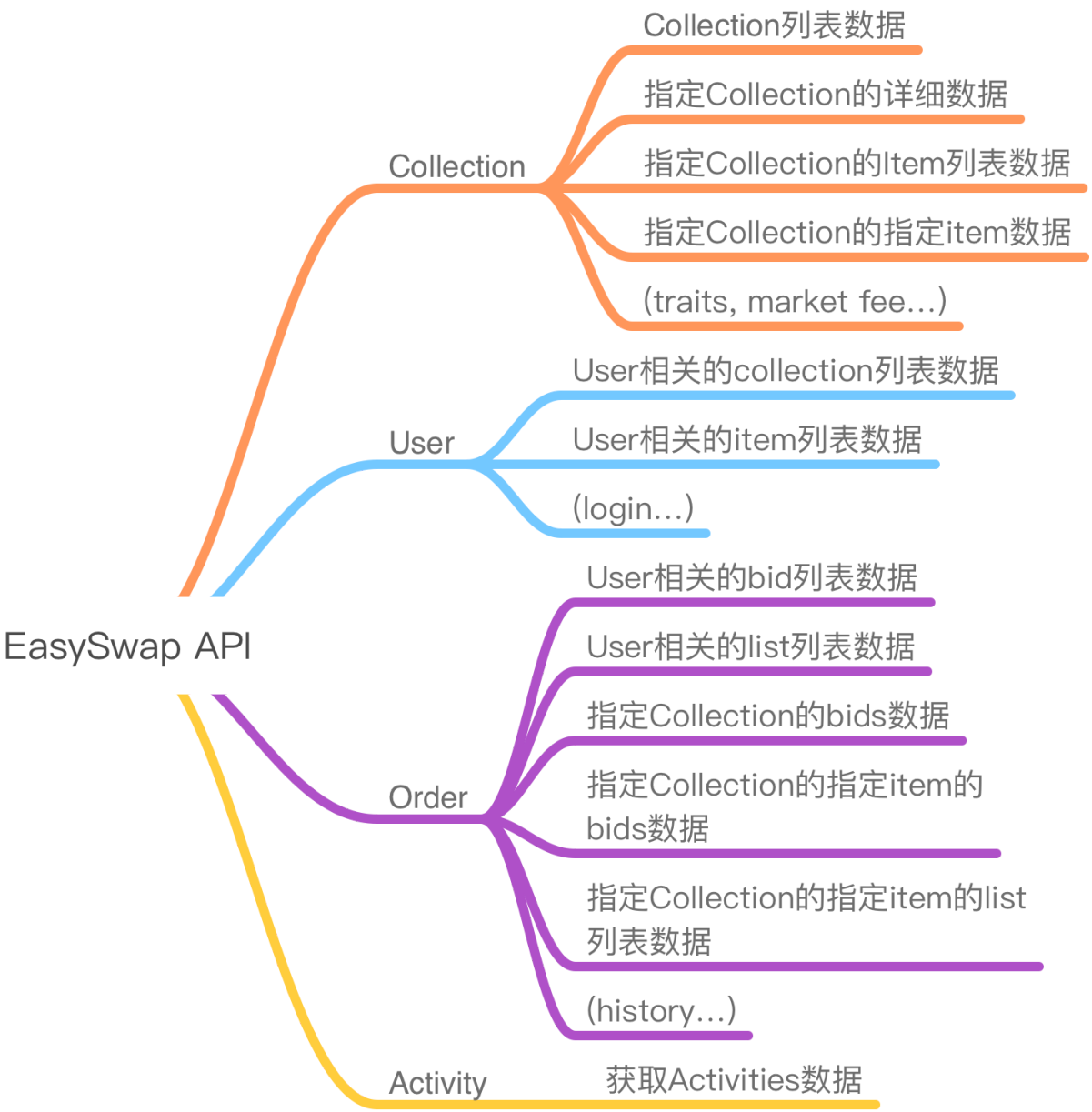
涉及Collection，Item， Order， Activity等实体的相关接口请求

```
1   create table ob_collection_sepolia
2   (
3       id                bigint auto_increment comment '主键'
```

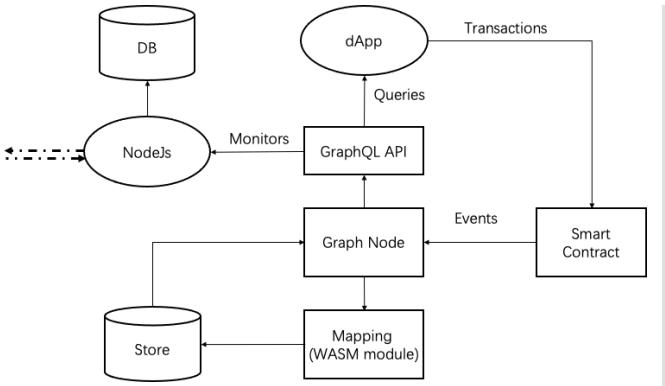
```
        primary key,
    chain_id           tinyint    default 1 not null comment '链类型(1:以太坊)',
    symbol             varchar(128)         not null comment '项目标识',
    name               varchar(128)         not null comment '项目名称',
    creator            varchar(42)          not null comment '创建者',
    address            varchar(42)          not null comment '链上合约地址',
    owner_amount       bigint     default 0 not null comment '拥有item人数',
    item_amount        bigint     default 0 not null comment '该项目NFT的发行总量',
    floor_price        decimal(30)          null comment '整个collection中item的最低的listing价格',
    sale_price         decimal(30)          null comment '整个collection中bid的最高的价格',
    description        varchar(2048)        null comment '项目描述',
    website            varchar(512)         null comment '项目官网地址',
          /twitter...
    volume_total       decimal(30)          null comment '总交易量',
    image_uri          varchar(512)         null comment '项目封面图的链接',
    // is_need_refresh    tinyint    default 0 not null comment '是否需要刷新',
    create_time        bigint               null comment '创建时间',
    update_time        bigint               null comment '更新时间',
    constraint index_unique_address
        unique (address)
);

create table ob_item_sepolia
(
    id                 bigint auto_increment comment '主键'
        primary key,
    chain_id           tinyint      default 1                    not null comment '链类型',
    token_id           varchar(128)                              not null comment 'token_id',
    name               varchar(128)                              not null comment 'nft名称',
    owner              varchar(42)                               null comment '拥有者',
    collection_address varchar(42)                               null comment '合约地址',
    creator            varchar(42)                               not null comment '创建者',
    supply             bigint                                    not null comment 'item供应量',
    list_price         decimal(30)                               null comment '上架价格',
```

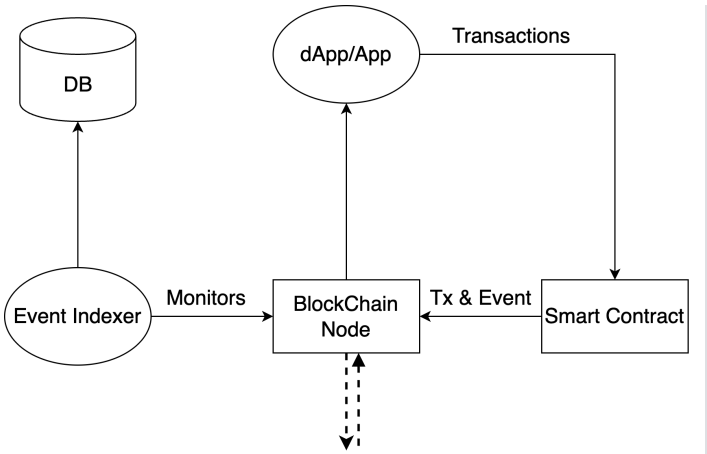
```
    list_time          bigint                                  null comment '上架时间',
    sale_price         decimal(30)                             null comment '上一次成交价格',
    create_time        bigint                                  null comment '创建时间',
    update_time        bigint                                  null comment '更新时间',
    constraint index_collection_token
        unique (collection_address, token_id)
);

create table ob_order_sepolia
(
    id                 bigint auto_increment comment '主键'
        primary key,
    marketplace_id     tinyint     default 0     not null comment '0.local',
    order_id           varchar(66)               not null comment '订单hash',
    order_status       tinyint     default 0     not null comment '标记订单状态',
    order_type         tinyint                   not null comment '1: listing 2:offer 3:collection bid 4:item bid',
    event_time         bigint                    null comment '订单时间',

    collection_address varchar(42)               null,
    token_id           varchar(128)              null,
    expire_time        bigint                    null,
    price              decimal(30) default 0     not null,
    maker              varchar(42)               null,
    taker              varchar(42)               null,
    quantity_remaining bigint      default 1     not null comment 'erc721: 1, erc1155: n',
    size               bigint      default 1     not null,
    salt               bigint      default 0     null,
    currency_address   varchar(42) default '0x0' not null ,
    create_time        bigint                    null comment '创建时间',
    update_time        bigint                    null comment '更新时间',
    constraint index_hash
        unique (order_id)
);

create table ob_activity_sepolia
(
    id                 bigint auto_increment comment '主键'
        primary key,
    activity_type      tinyint                 not null comment '(1:Buy,2:Mint,3:List,4:Cancel Listing,5:Cancel Offer,6.Make
```

```
    Offer,7.Sell,8.Transfer,9.Collection-bid,10.Item-bid)',
77      maker               varchar(42)             null comment '对于
    buy,sell,listing,transfer类型指的是nft流转的起始方，即卖方address。对于其他类型可以
    理解为发起方，如make offer谁发起的from就是谁的地址',
78      taker               varchar(42)             null comment '目标方,和maker相
    对',
79      marketplace_id      tinyint     default 0   not null,
80      collection_address varchar(42)              null,
81      token_id            varchar(128)            null,
82      currency_address    varchar(42) default '1' not null comment '货币类型(1表示
    eth)',
83      price               decimal(30) default 0   not null comment 'nft 价格',
84      block_number        bigint      default 0   not null comment '区块号',
85      tx_hash             varchar(66)             null comment '交易事务hash',
86      event_time          bigint                  null comment '链上事件发生的时
    间',
87      create_time         bigint                  null comment '创建时间',
88      update_time         bigint                  null comment '更新时间',
89      constraint index_tx_collection_token_type
90          unique (tx_hash, collection_address, token_id, activity_type)
91  );
```

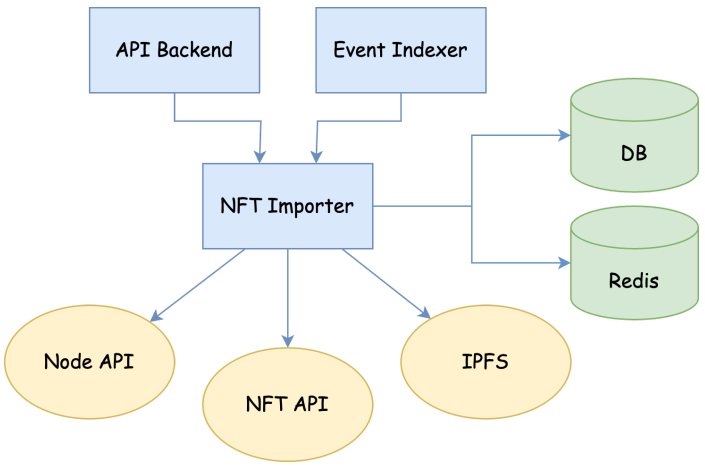


基于TheGraph的同步架构	独立实现的同步架构





Importer服务架构



交易合约核心功能

交易合约核心功能

基于智能合约实现基于订单簿模型的NFT交易系统, 即能够支持以下写入操作和查询操作;

写入操作:

链上订单簿(OrderBook DEX)支持 create limit sell/buy, market sell/buy order, edit(cancel&create)/cancel order 功能;

1. **limit sell order:**
2. **limit buy order:**
3. **market sell order:**
4. **market buy order:**
5. **edit/cancel order:**

查询操作:

1. 支持从链上查询订单（包括已经过期订单）;

合约构成及组件

1. OrderBookExchange: 实现完整的订单簿交易逻辑
    a. OrderStorage: 用于存储订单信息的模块
    b. OrderValidator: 用于处理订单逻辑验证的模块
    c. ProtocolManager: 用于管理协议费的模块
    d. ...
2. OrderVault: 独立存储订单相关资产的模块;