Character Token

Whitepaper

-

Notice

Character Token was built using Tendermint, Cosmos SDK and Starport technologies which are open source. As such, the whitepapers for the consensus, validator, minting and governance protocols are outside the scope of this whitepaper; links to which may be found in the References section at the end of the paper.

Definitions

NFC - Non-Fungible Contract - a record in the database that describes an Application Programming Interface using a serialization protocol, by default Character Token supports Protocol Buffers.

NFT - Non-Fungible Token - in Character Token, an NFT is any record in the database that holds an instantiation of the stateful portions of the NFC serialization. The NFT is created through a signed transaction tied to the burning of a token. NFTs in Character Token are also referred to as assets or characters.

UUID - Universally Unique Identifier - A string representing a transaction/database entry in Character Token

Walled Garden - The description of a world that has assets that are not transferable to the user for permanent ownership. The rules or physical limitations of the world prohibit the transfer of assets between other worlds.

World - Any system which enables human interaction through digital assets, rules and API.

Worldbuilder - A person or company that constructs all or part of an asset.

Worldbuilding - The process of creating digital representations in a virtual world.

Problem

World and asset creation are at the heart of most digital experiences. Worldbuilding refers to the process of creating artistic digital representations of environments in which the user is expected to interact; complete with rules that govern user actions within those environments. Assets refer to the objects that are created, owned and controlled by the users within the virtual world.

Assets are created by the worldbuilders, for the consumption of the users. Worldbuilders in many cases may be a centralized entity such as a game developer or art group. Sometimes the users may be responsible for some worldbuilding tasks through some sort of democratized capability or tool (level editor, character designer, or game engine).

Traditionally, worlds are walled gardens; they contain assets that are only valid within the world and those assets may not be directly owned by the user or transferable to other experiences. This poses a problem because there is a loss of potential where a character, item, or world asset may be viable in other experiences and, therefore, have universal value rather than just value in the walled garden ecosystem.

Ownership of an asset cannot easily be democratized. Recently, NFTs (Non-Fungible Tokens) have been innovated to empower asset owners with the ability to claim creative or at least ownership rights through smart-contract. Unfortunately, smart-contracts are commonly complex and sometimes built with security flaws.

Currently, there exists no standard to ensure that the NFT can be incorporated into alternate worlds because the NFT is not also paired with an NFC (Non-Fungible Contract). The NFC is any Application Programming Interface (API) that enables a common interpretation of an NFT in such a way that all or part of the metadata contained within the NFT may be interpreted in predictable ways, ensuring that at least a basic contract of the NFT and the asset's configuration are honored.

Solution Statement

Character Token is a layer 1 blockchain designed to overcome the following challenges:

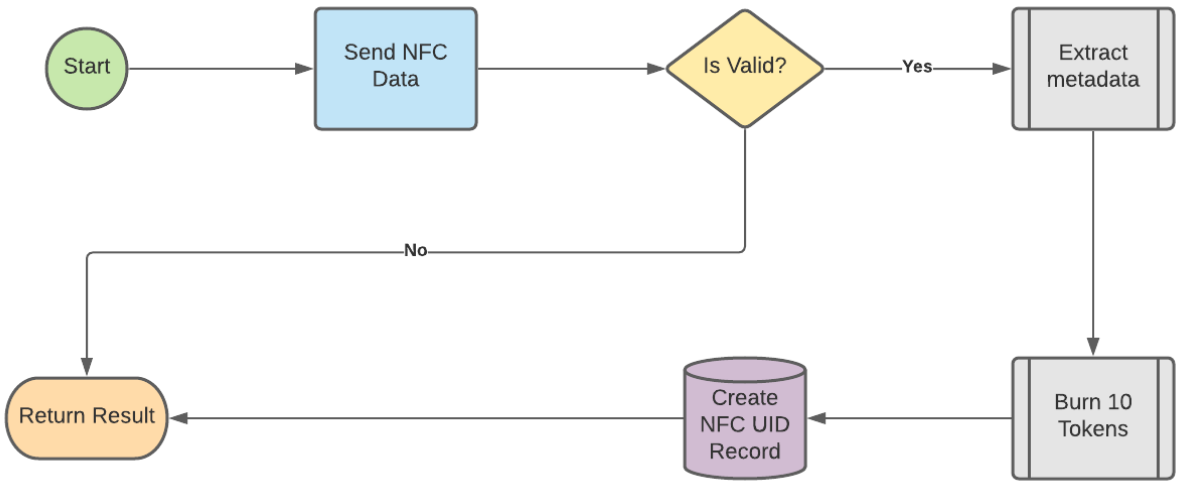
1. Overcome some restrictions of walled garden worlds
2. Assure the ownership rights of users to assets
3. Secure smart-contracts
4. Support metadata and enable universal API through NFC

This paper will explore the above technical challenges and provide a description of how Character Token solves those challenges. It does not go into depth about the Cosmos SDK, Tendermint or Starport which were used as the underlying blockchain technology. Whitepapers

and other documentation exists for each of those technologies. Links to references are in the appendix.

NFC (Types/Protocols)

Character Token provides the ability to store application interaction contracts as NFC. These contracts may be queried by any system in order to validate and recall any asset. The Character Token system allows a user to generate a “buy-type” transaction, which creates the NFC. The following describes the API workflow that Character Token validators use to generate an NFC:



NFC Protocol Buffer Wire Format

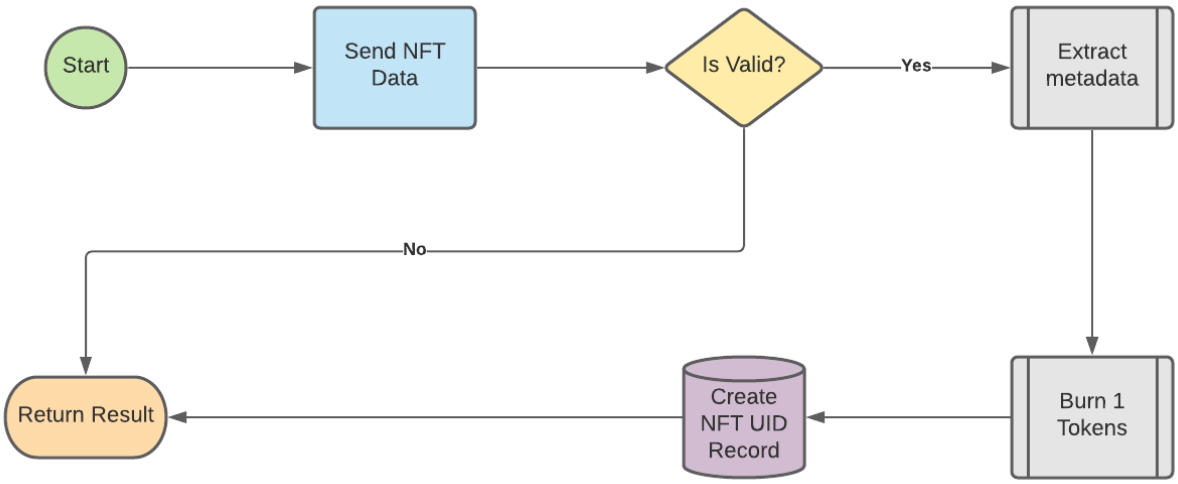
```
message Ctype {
  string creator = 1;
  string index = 2;
  string name = 3;
  string language = 4;
  bytes schemaFile = 5;
  string messageName = 6;
  string group = 7;
}
```

NFT (Assets/Characters)

Character Token also supports the creation of assets, which are the instantiation of one or more parts of an NFC. Assets have a hard dependency upon the NFC. All assets must have an embedded CType ID text string (see previous wire format) which refers to the transaction id of the transaction that created the NFC. Transaction IDs in Character Token are UUID.

The asset contains the binary representation of an NFT; including images, lines and other metadata. Assets may be queried in Character Token by ID, name, creator, or other conditions, such as all characters that are on sale on the open market.

The Character Token system allows for a user to generate a “buy-char” transaction:



NFT Protocol Buffer Wire Format

```
message Character {
  bool tradeRestricted = 1;
  int64 saleTime = 2;
  string creator = 3;
  string index = 4;
  string name = 5;
  string owner = 6;
  string ctype = 7;
  string cost = 8;
  string license = 9;
  bytes value = 10;
}
```

NFT Transactions Supported:

1. SetSale
    a. Must supply unique index of an existing NFT
    b. Sets the NFT for sale in the blockchain
    c. User signing the transaction must be Owner of the NFT
    d. Must supply a cost greater than current value

    e. Must supply a length of time in minutes for the sale to be valid, after that time, the sale is no longer valid. (Sol-Earth standard minute)
2. Unlist
    a. Must supply unique index of an existing NFT
    b. Unsets a listed sale for the NFT
    c. User signing the transaction must be Owner of the NFT
3. BuyChar
    a. Must supply unique index of an existing NFT
    b. Creates or Purchases the NFT; the Owner is set to the signer of the transaction; if it is a new NFT, the Creator is set to the signer of the transaction also
    c. On Purchases, signer must be anyone other than the current owner of the NFT
    d. On Purchases, New Owner must have enough ctok currency in their wallet to purchase the NFT plus gas used up to that point.
    e. On Purchases New Owner may only update the following fields:
        i. tradeRestricted - if true, the NFT may no longer be traded in any capacity
        ii. saleTimeMinutes - if greater than 0, sets the new NFT for sale at the given cost
        iii. cost - the sale price of the NFT on the open market
        iv. license - a use license to help define legal copyrights
        v. name - the new name of the NFT after purchase

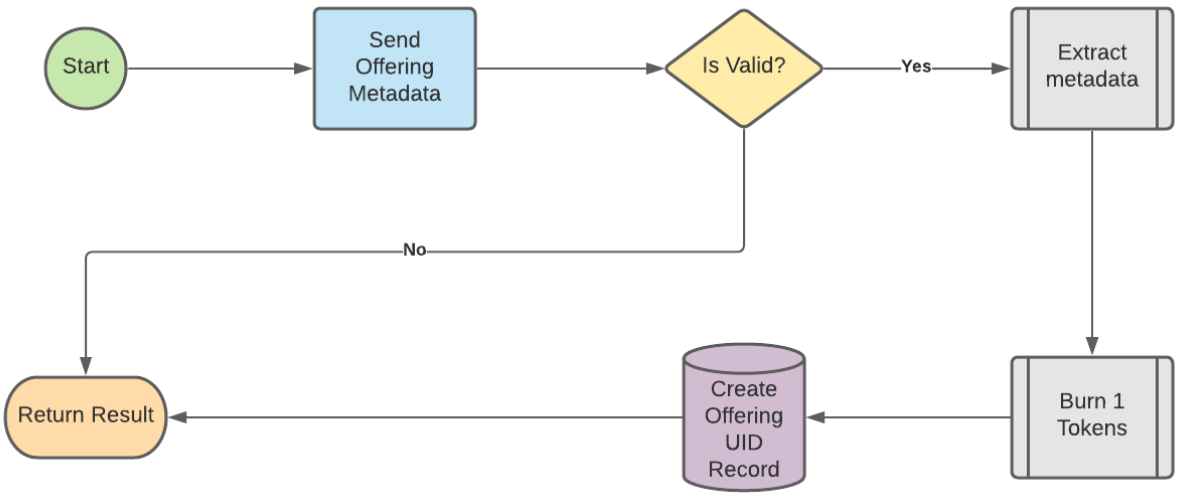
The following queries are possible:

1. Get all by Name
2. Get all of a certain Creator by public wallet id
3. Get all for Sale
4. Get all of a particular NFC by CType index
5. Get all of a current Owner by public wallet id
6. Get one by NFT index
7. Get all assets in the Database

Offerings

Because Assets may be set into a state which allows them to be freely transferred by any wallet, a system is needed when there needs to be a secure, anonymous transfer of ownership controlled by the owner. Offerings are the way to ensure secure transfer between wallets. Offerings are an asynchronous transfer mechanism, which allows the owner of an asset to accept bids for an asset at will and for any amount.

When creating an offering, the “create-offer” transaction type is used:



Offering Protocol Buffer Wire Format

```
message OfferContract {
  int64 expiresAt = 1;
  string creator = 2;
  string index = 3;
  string charId = 4;
  string value = 5;
}
```

Offering Transactions Supported:

1. CreateOffer
    a. Must supply unique index of an existing NFT to create an offering for it
    b. Creator of the Offering may not be the owner of the NFT
    c. Must supply a bid greater than zero, which will be transferred to the Owner upon acceptance
    d. May supply a length of time in minutes for the Offer to be valid, after that time, the Offer is no longer valid. (Sol-Earth standard minute)
2. AcceptOffer
    a. Must supply unique index of an existing Offer to accept an offer for an NFT
    b. Owner of the NFT must be the signer of the transaction to accept the Offer
    c. Owner of the offer must have enough funds to complete the transaction; Owner of the NFT must have enough funds to pay the gas.
    d. New cost of the NFT will be set to the accepted Offer cost. All offers for the NFT will be removed once an offer is accepted.

The queries for the offering type are as follows:

1. Get all Offers for an NFT
2. Get an Offer by offer Index
3. Get all Offers in the database

Advanced Use Cases

GRPC

The NFC, NFT and Offering describe a system governing stateful transactions on the surface. Advanced use cases may exist where the NFC may also describe expected behaviors and state transitions. The GRPC protocol is one example of a way to express state transitions within an NFC. Any NFC described in terms of GRPC may store the entire API in such a way that the contract describes both behavior and state expectations. Additionally, multiple, groupled NFC may describe transitions in state or chained behaviors.

Open API

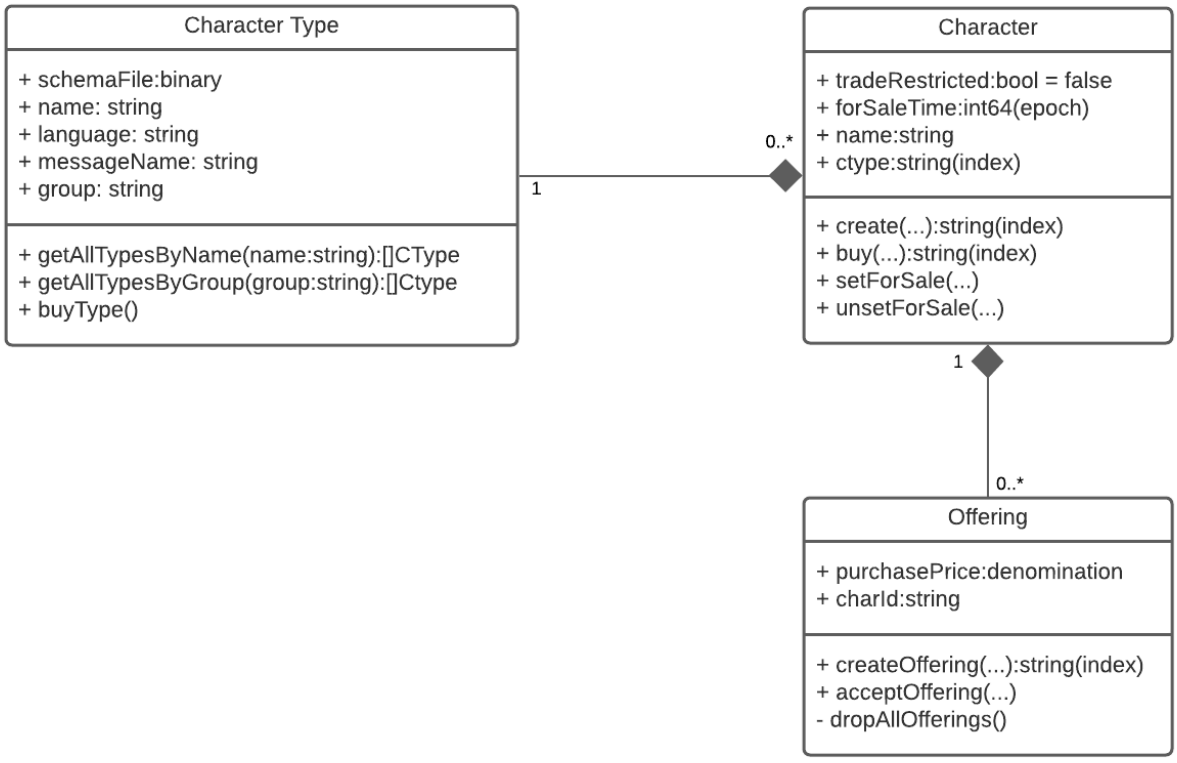
As an extension, the OpenAPI specification may also serve the same purpose as the GRPC use case described above. In this case, one may also incorporate JSON binary as the state transitions.

Encrypted API (Encrypted NFC and NFT)

The NFC may also be used to express the expectation of encrypted NFC and/or NFT as the metadata in an NFC may be expected to be interpreted using a pre-shared key of some kind. The implementations of which are left as an exercise for the reader.

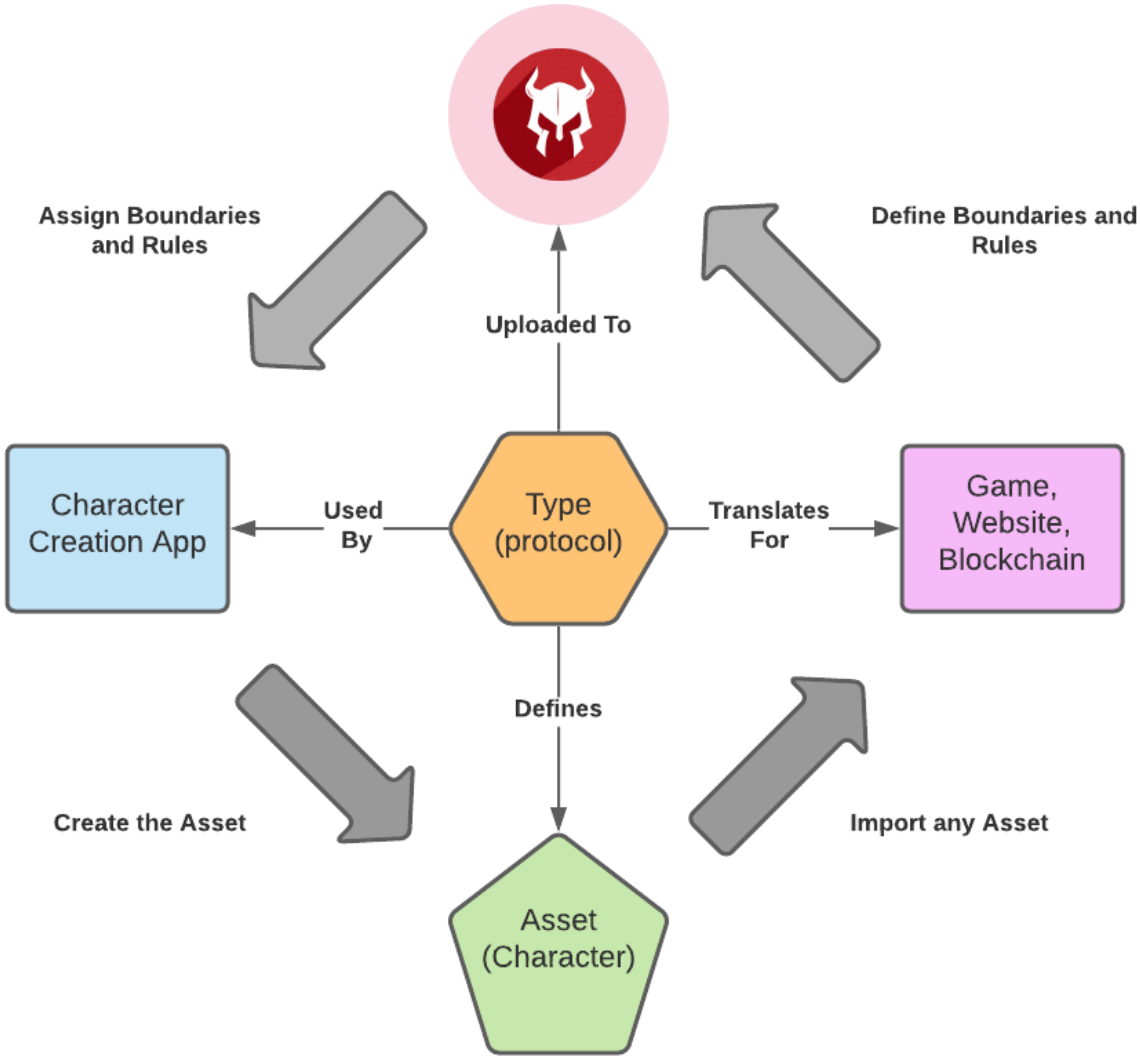
UML Design and High-Level Relationships

The following is a UML representation of the relationships between the Character Token stateful entities:



These classes are brought together to support the primary stateful use case described in the problem statement, the federation of NFT using the NFC as a contract. Below is a sample use-case involving a traditional walled garden game world and how Character Token may be

used to federate the NFT/asset:



REFERENCES

The flexibility of this system is possible due to simplicity of design and concentration on the domain of interoperability. THe goal of the project is to retain its flexibility, provide a backend for other systems, and remain open to change through its governance protocol.

Character Token was built using Tendermint, Cosmos SDK and Starport technologies which are open source. The whitepapers for the consensus, validator, minting and governance protocols are all contained in the following web addresses:

Consensus without Mining

Starport CLI

Whitepaper - Resources

Character Token Open Source Project links:

All Projects

Project Wiki

Testnet Genesis Files

Project Source