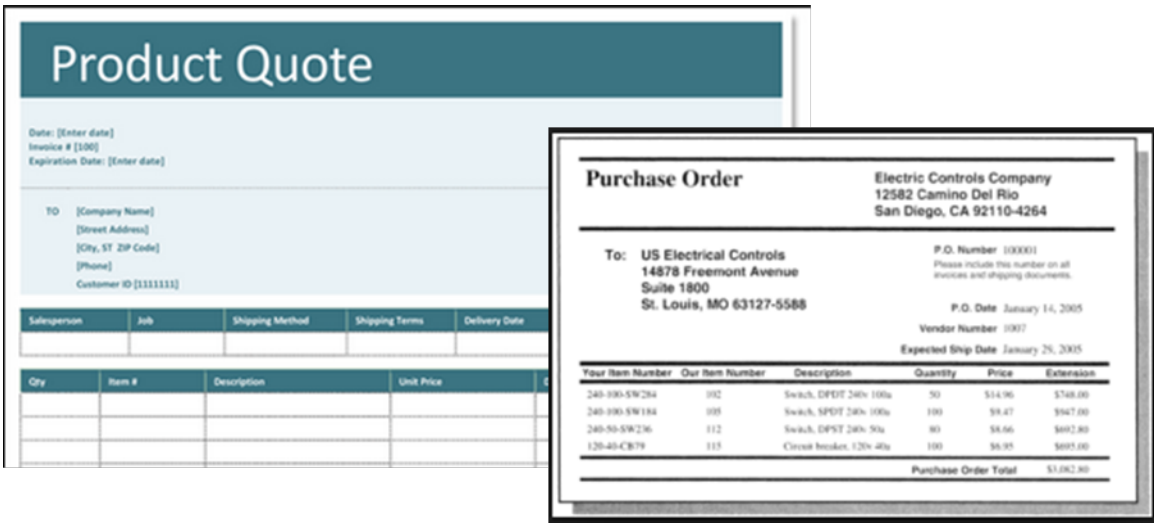


Anatomy of Asset and Value Transfers with the Extensible Blockchain Object Model and Extensible Smart Object Assets on the DataGrid Blockchain

- What if there is a better, more intuitive, more scalable way to manage assets on the blockchain?

Overview

The following describes at a conceptual level a simple model for the transfer of assets and value between two accounts on the DataGrid Blockchain ("DGB". This makes use of the Extensible Blockchain Object Model ("XBOM")'s unique flexibility for representing objects directly in user accounts. The descriptions given are deliberately simplified to highlight main features. This paper follows from the "**Rethinking Sharding and Smart Contracts For Maximizing Blockchain Throughput**"[1] and "**Extensible Smart Object Assets, Smart Object Asset Ownership and Fractional Smart Object Asset Ownership With the DataGrid Blockchain Extensible Blockchain Object Model**"[2] papers. We start off with some definitions, present simplified transfer transaction flows, and conclude with further comments.

Object Reference Definition

All accounts on the DGB consist of object instances. Each object is identified by an object reference. An object reference consists of the account address and the object index within the account. Object references are defined as the following:

- System global wellknown
- Account wellknown
- Account local

[1] https://medium.com/@dbeberman/rethinking-sharding-and-smart-contracts-for-maximizing-blockchain-throughput-5eaf91bb5781

[2] https://medium.com/@dbeberman/extensible-smart-object-assets-smart-object-asset-ownership-and-fractional-smart-object-asset-995c259a8508

A system global wellknown object reference is to a unique object that is always available on the blockchain. Any object may send a message to a global wellknown object.

An account wellknown object reference is to an object that every account contains an instance of. As will be described below, every account contains a root object, and a DGT balance object.

An account local object reference is to an object that is dynamically created by an account and is specific to a particular account's state. For example, an account may contain multiple asset reference objects.

As stated, an object reference consists of two parts:

Account Address : Object Index

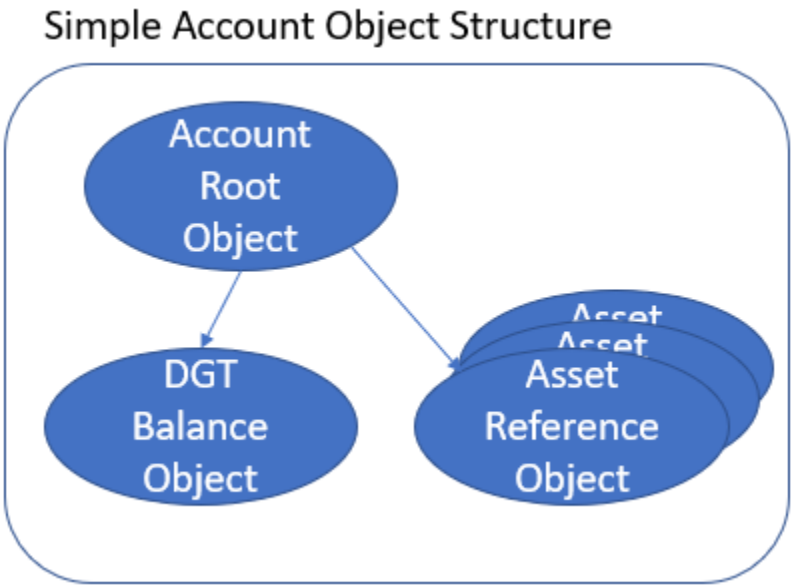
Where the account address is the account ledger entry address. If the account address is blank, then all messages referenced by the object index are defined to be to an object that is within the current account.

Object Accessibility

Object accessibility has the following possible scopes:

- Global accessibility: anyone can send a message to the object
- Local accessibility: only the account containing the object may send a message to the object
- Account friends: only accounts that have been authorized by the account containing the object may send a message to an object within that account.

Simple Account Object Structure



A basic logical account structure consists of a wellknown root object, which contains by reference a DataGrid Token ("DGT") balance object and a list of asset reference objects owned by the account.

Account Root Object

Every account has a wellknown root object which receives transactions that are sent to an account when not explicitly referencing an object within the account. That is, the object reference is the following:

Account Address : <blank>

Equivalently a transaction sent to an object reference as the following:

Account Address : <wellknown root object index>

Also references the root object in the account.

DGT Balance Object

The DGT balance object maintains the balance of token owned by the account. All additions and subtractions of DGT balance are accomplished through messages to this object. A DGT Balance object shall reject a subtraction that would result in a negative balance.

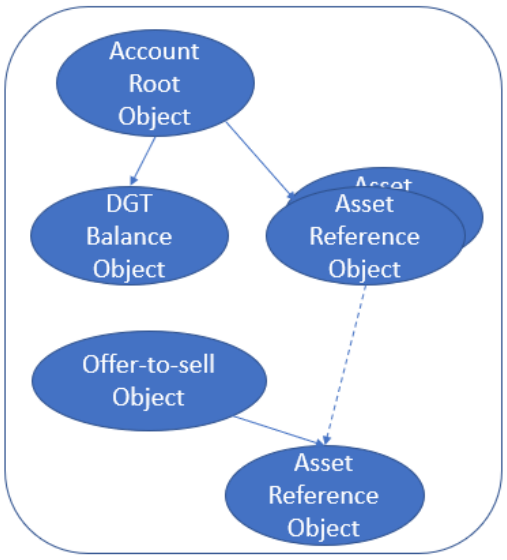
Asset Reference Object

The asset reference objects define the ownership of assets for the account. An account may have an unlimited number of asset reference objects. The reference field values contained within these objects are transferred between accounts for asset ownership transfer transactions.

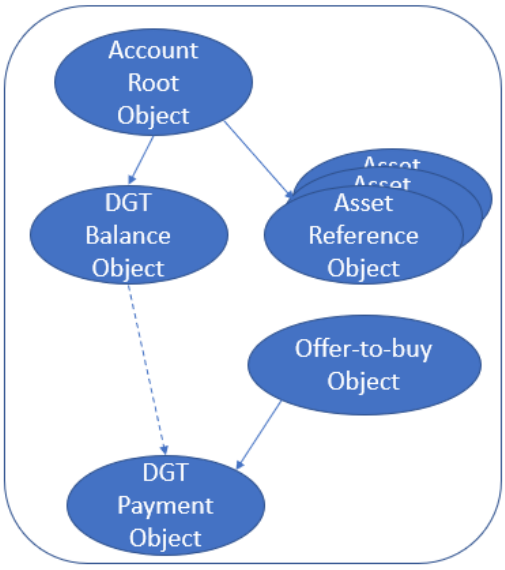
Simple Purchase Model

Direct Account-to-Account Asset Transfer Transactions

Selling Account Object Structure



Buying Account Object Structure



The simple purchase model enables the direct sale of an asset from one person or entity to another. As an example, this can be thought of as a private sale. There two transaction flows to accomplish such an exchange between accounts. The transactions are initiated by the buyer or the seller. The transaction flows are as follows:

A seller initiated transaction:

- The seller creates an offer-to-sell object with the following parameters:
  - An asset reference object that the seller owns. The object is removed from the asset list and inserted in the containment tree of the offer-to-sell object.
  - An amount to pay for the asset in DGT
  - The address of the buying account
- The seller gives the buyer, the reference to the offer-to-sell object

- The buyer creates an offer-to-buy object with the following parameters:
  - The offer-to-sell object reference
  - The amount of DGT
  - The offer-to-buy object sends a message to the offer-to-sell object with its self object reference
    - The offer-to-sell object verifies the amount of DGT in the offer-to-buy object
  - The offer-to-buy object creation finishes successfully returning its object reference to the buyer

- The seller sends a message to the offer-to-sell object, accepting the offer-to-buy:
  - The offer-to-sell object sends the asset object reference value to the offer-to-buy object.
  - The offer-to-sell object increments the seller's DGT balance retrieved from the offer-to-buy's DGT.
  - The offer-to-sell object sends a delete message to the offer-to-buy.
  - The offer-to-sell deletes itself

Notes:

This transaction flow is written as a 3-way protocol. The seller explicitly accepts the buyer's offer. It is also possible to have an automatic offer acceptance. In that use case, the offer-to-sell object would perform the last transaction directly on receipt of the offer-to-buy message. It is then a 2-way transaction and the object-to-buy object is deleted immediately, returning a null object reference on its creation transaction.

Alternatively, there may be multiple additional related transactions before the seller accepts the offer-to-buy and completes the asset transfer.

A buyer initiated transaction:

- The buyer creates an offer-to-buy object with the following parameters:
  - The address of the seller account
  - The type of the asset to buy
    - Asset specific parameters
  - The amount of DGT of the offer
- The buyer gives the seller the offer-to-buy object reference

The seller then creates an offer-to-sell as above, passing it the offer-to-buy object reference, and performs the offer acceptance transfer.

Simple Payment Model

A simple payment model does not include an asset transfer. It consists solely of a DGT balance transfer between accounts. The transaction flow is from the sender to the receiver. The transaction flow is as follows:

- The sender creates a DGT transfer object with the following parameters:
  - The amount of DGT
  - The address of the destination account
- The DGT transfer object sends a message to the destination account with its self object reference
- The destination account retrieves the DGT and adds it to its DGT balance
- On return the DGT transfer object deletes itself

Notes:

In this example, the DGT transfer object exists only during the transaction as a temporary object. It is not added to the state of the sending account.

Further Comments

In the above examples we used the generic terms "**offer-to-sell**" and "**offer-to-buy**" as the names of the objects involved. Depending on the types of asset and value transfers, the offer-to-sell can be thought of as an analogous to a price **quote**, and the offer-to-buy as a **purchase order**.

It should be understood that the above are intentionally simplistic examples. The flexibility of the XBOM enables near infinite variations of value and asset transfers on the DGB. The XBOM is an open architecture. We encourage and expect users globally to independently develop and use their own variations of value and asset transfers.

If should also be understood that there are no smart contracts involved in these examples, or needed in even more complex variations. All transactions occur directly with user accounts. The code is directly associated with the object instances alleviating the need for separate smart contract accounts, and any associated bottlenecks.