# Atomic Swap Open Protocol

June 24th, 2023

Author: Lawrence Ley, B.Sc.

CONTEXT SOLUTIONS

*“Implementing the right solution”*

Version 1.0

# Table of Contents

# Introduction

In September 2021, the Alzonzo hard fork enabled smart contract functionality on Cardano. Smart contract development back then was very difficult and redefined the term bleeding edge. Regardless of those initial challenges, a number of DeFi protocols and applications were launched and Cardano DeFi was born. A year later with the Vasil hard fork and alternative smart contract languages such as Helios, writing smart contracts on Cardano became increasingly easier and more powerful.

On March 4th 2023, @zhekree posted on twitter about the open source repository cardano-swaps. It was written in haskell so I decided to write my own atomic swap using Helios which I have become very familiar with. I also decided to expand and modify some of the original concepts by cardano-swaps. Therefore, I am excited to share the Atomic Swap Open Protocol for feedback and comments.

# What is it

The Atomic Swap Open Protocol (ASOP) is a protocol that defines how an atomic swap can be used in a fully decentralized, scalable and secure manner.  The protocol also has additional features such as escrow contracts for settlement and user tokens for fraud prevention and regulatory compliance if required. Finally, each swap can be denominated in any native Cardano asset including Ada.

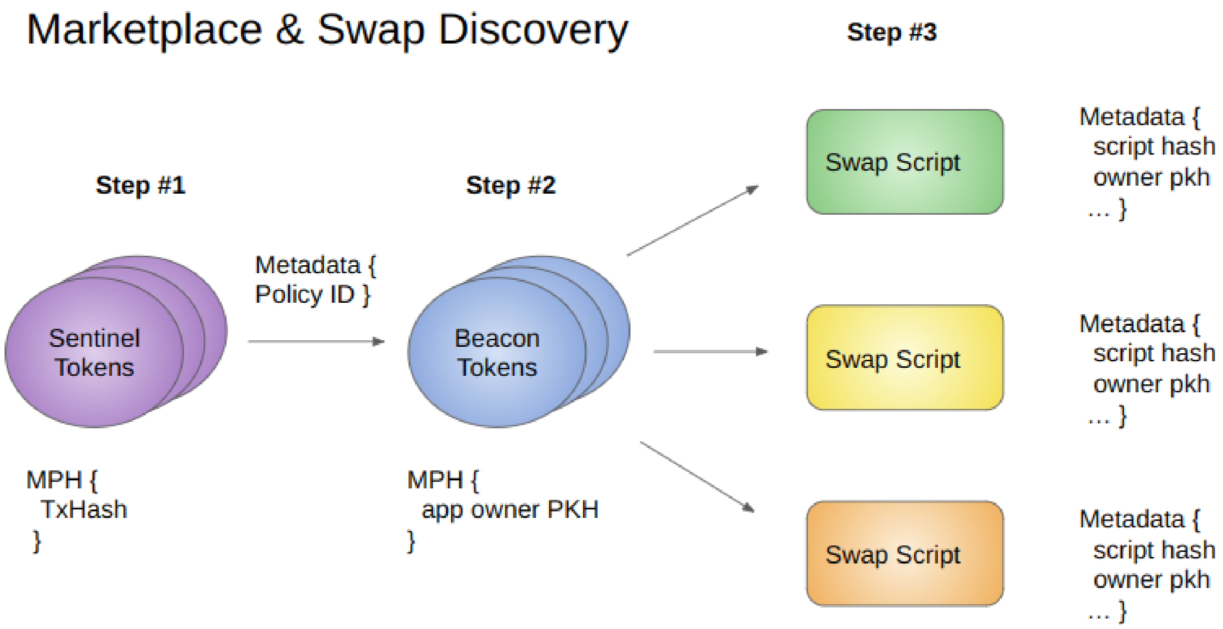
# Why use it

Here are the following benefits in using an atomic swap architecture:

- Retain self custody (including staking capability) while the asset is being offered for sale
- You can update or close the swap at any time without any restrictions
- No impermanent loss because you are not providing liquidity to a pool
- Highly secure because you and only you have the keys to update or close the swap position
- Fast finality because there are no other dependencies for the execution of the swap
- No batchers are used which eliminates front-running transaction risk
- Fees are known in advance and no fees are typically charged if the transaction fails

## How does it work

In order to execute a swap, there needs to be a discovery of the marketplaces offering swaps and the swap details. The following diagram shows this process.



**Step #1**

A Sentinel token can be minted by anyone but must include the Beacon policy ID of the marketplace in the Sentinel minting transaction metadata. The Sentinel token can be located anywhere, but should be located at the marketplace owner address should they want to burn the token and de-list their marketplace.

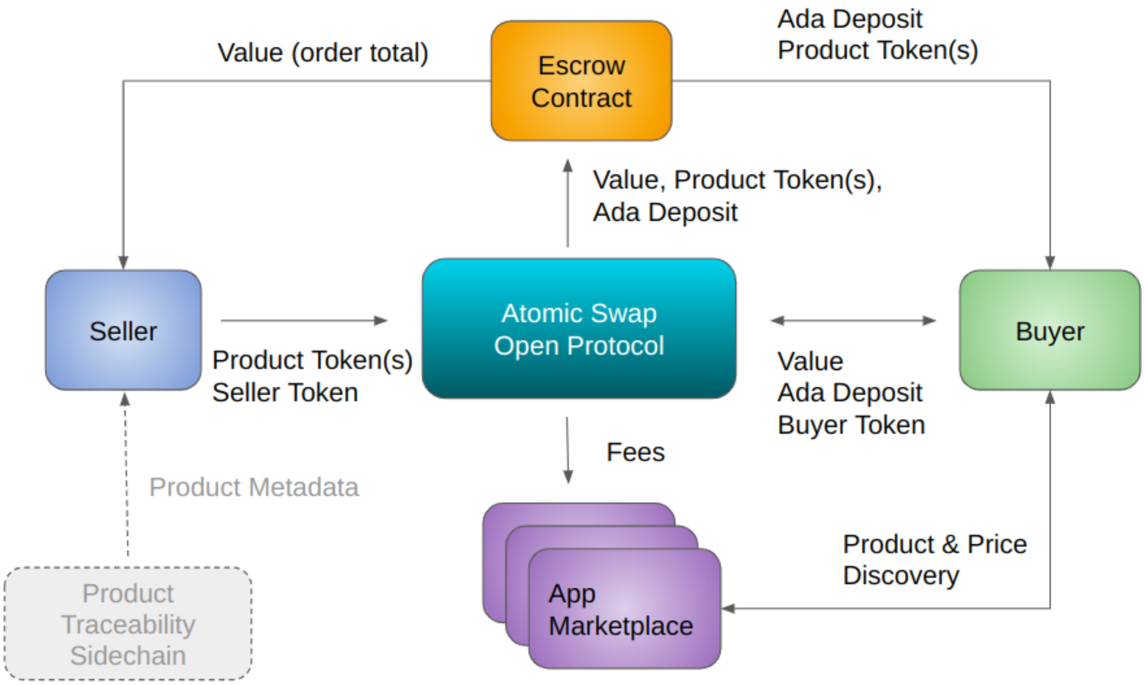
**Step #2**

Beacon tokens are minted every time a swap position is opened.  A Beacon token is burned when a swap position is closed.

**Step #3**

A Beacon token must always exist at the swap address.  The swap smart contract enforces that any transaction involving the swap must always include a beacon token.

# Atomic Swap High Level Design

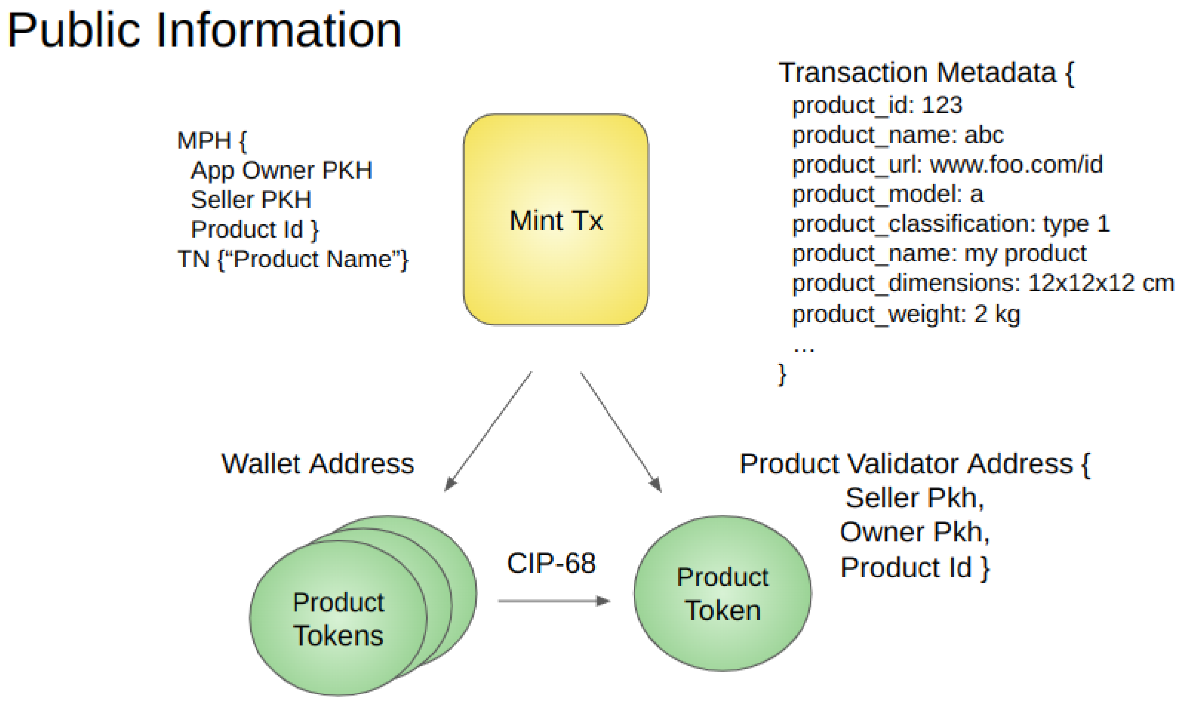


| Item | Description |
|---|---|
| Ada Deposit | A deposit in Ada when a swap is using an escrow contract |
| App Marketplace | A marketplace app with a user interface for buyers and sellers |
| Buyer | The wallet that is executing an asset swap |
| Buyer Token | A user token that can only be spent by the buyer’s wallet |
| Escrow Contract | An optional multisig settlement contract |
| Fees | The fees that go to the marketplace owner when a swap is executed |
| Product Token | A digital asset that may represent a physical product |
| Product & Price Discovery | The ability to discover what swaps are open |
| Seller | The wallet that can open/update/close a swap position |
| Seller Token | A user token that can only be spent by the seller’s wallet |
| Value | A Cardano native asset (including Ada) that is used as a medium of exchange |

# Product Tokens

Product Tokens are used to represent something being bought or sold. This can however be any digital asset on the Cardano blockchain such as existing NFTs or tokens. This allows for creation of secondary digital asset marketplaces if needed.

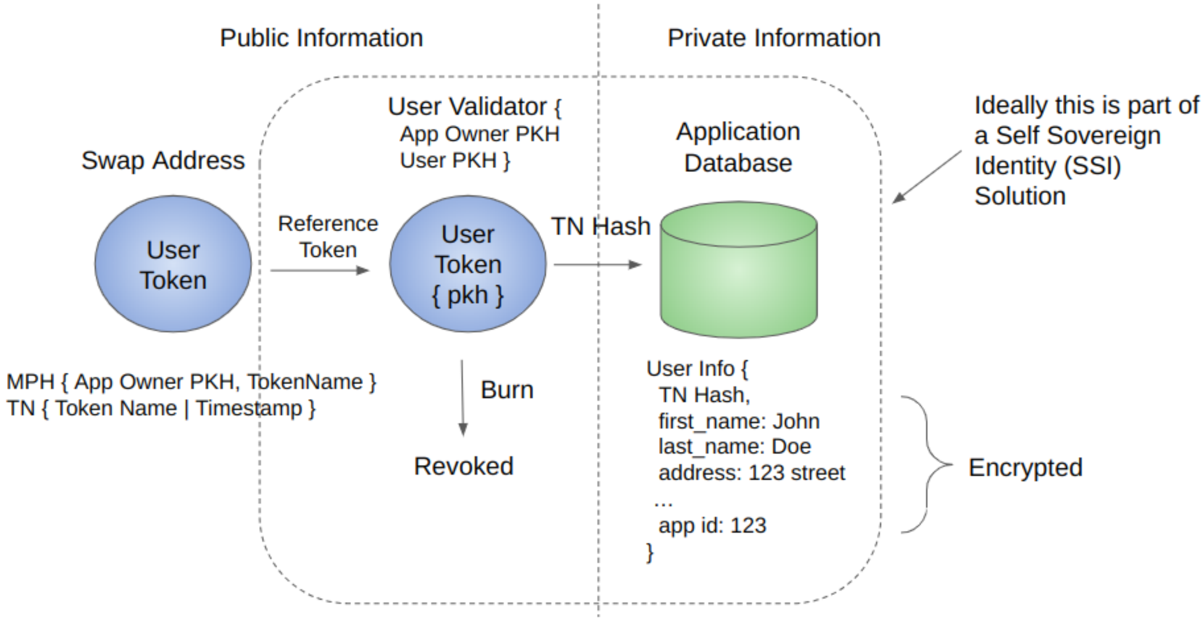
The following diagram shows how a product token is minted to represent a real physical product. In particular, by adopting the CIP-68 standard to use a soul-bound token locked at address, the minting transaction metadata is easily available and contains all product information.



```
Transaction Metadata {
  product_id: 123
  product_name: abc
  product_url: www.foo.com/id
  product_model: a
  product_classification: type 1
  product_name: my product
  product_dimensions: 12x12x12 cm
  product_weight: 2 kg
  …
}
```

# User Tokens

The presence of both a buyer and a seller user token is required to execute a swap transaction. The buyer and seller token are created the same way by minting a user token, signed by the marketplace owner. A token becomes a seller (or buyer) token in the context of its use. For example, if a seller opens a swap, they must include a user token. This user token will be included in the open swap and will be considered the “seller” token. Conversely, when a buyer executes a swap, they will also have to provide a user token and in this context, the token will be considered a “buyer” token. For tokenized real-world products, user information is required for shipment.



When a user token is minted, there is a soul-bound token also created and located at a user validator address. When a swap is executed, both the user token and the reference user token (as a reference input) must be included. Therefore, there must always be the 2 user tokens for each user in a given swap transaction for it to be successful. Additionally, a user token can only be spent from the public key that was used to mint it.

The decision to have user tokens at the protocol level was not taken lightly. User token verification can be done at the application level and the current design does not limit the marketplace ability to capture additional information. That said, to help facilitate adoption beyond the typical “crypto degen”, some level of consumer protection against fraud and avoiding sanctioned entities is required. The amount of the verification will reside with the marketplace owner who ultimately signs (authorizes) the creation of a user token for their marketplace.

# Challenges, Risks and Opportunities

The biggest challenge initially will be liquidity. If there are no buyers, then it is hard to attract sellers and vice versa. But by leveraging transaction metadata and native minting policies, decentralized independent marketplaces can be discoverable and create a viable swap ecosystem.  SPOs (Stake Pool Operators) or social media influencers who already have established brand awareness and trust may be in an ideal situation to set up their own marketplace. Marketplace owners can also connect with other known marketplaces and create a network of “trusted marketplaces” which can increase the swap offering to their customers.

Each marketplace owner must be aware of the laws and regulations that they must be in compliance with. This includes, but not limited to, what type of products are legal to buy and sell as well as what type of customers they can interact with. Please consult a lawyer and an accountant prior to setting up a marketplace.

# Summary

The goal of the Atomic Swap Open Protocol is to lay a foundation for real world DeFi use cases leveraging blockchain technology. I would encourage anyone interested in contributing or provide feedback to raise an issue or pull request on GitHub. None of this would have been possible without others contributing and sharing open source code such as cardano-swaps, strica and Helios.

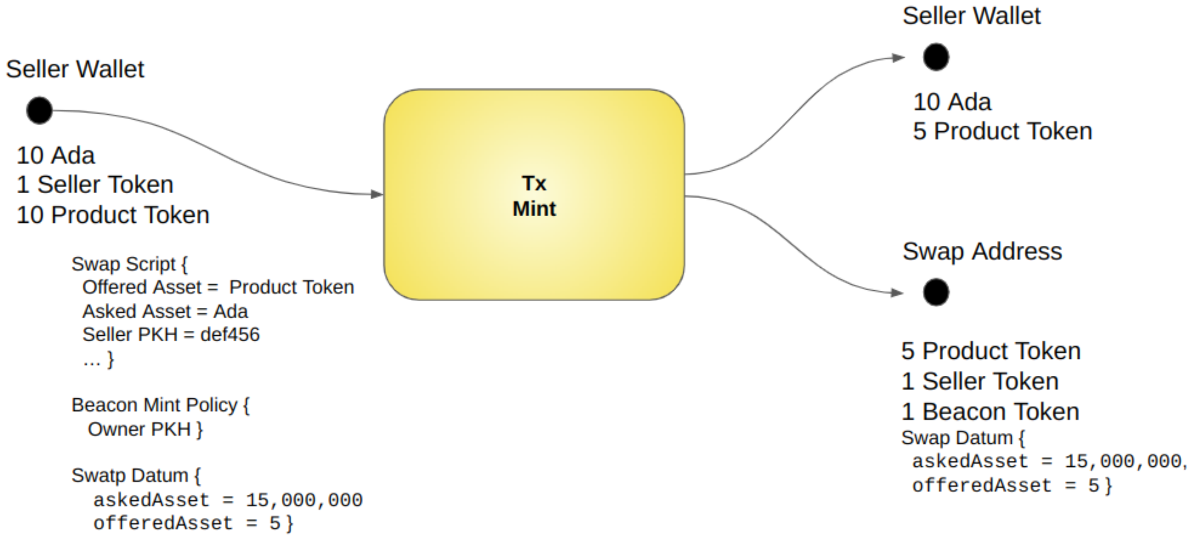
In closing, Cardano has made some great design choices such as using a UTXO architecture. This creates a local state transaction and makes the atomic swap possible.

Cheers!

Lawrence

# Appendix - High Level Transaction Design

# Open Swap



Seller Wallet

10 Ada
1 Seller Token
10 Product Token

Swap Script {
 Offered Asset =  Product Token
 Asked Asset = Ada
 Seller PKH = def456
 … }

Beacon Mint Policy {
  Owner PKH }

Swatp Datum {
  askedAsset = 15,000,000
  offeredAsset = 5 }

Tx
Mint

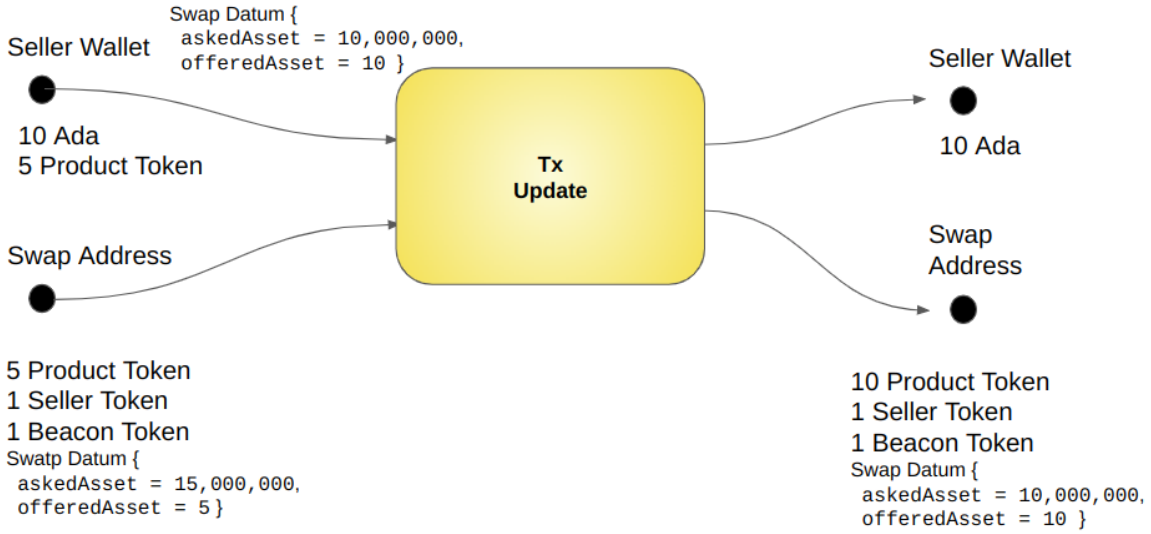
Seller Wallet

10 Ada
5 Product Token

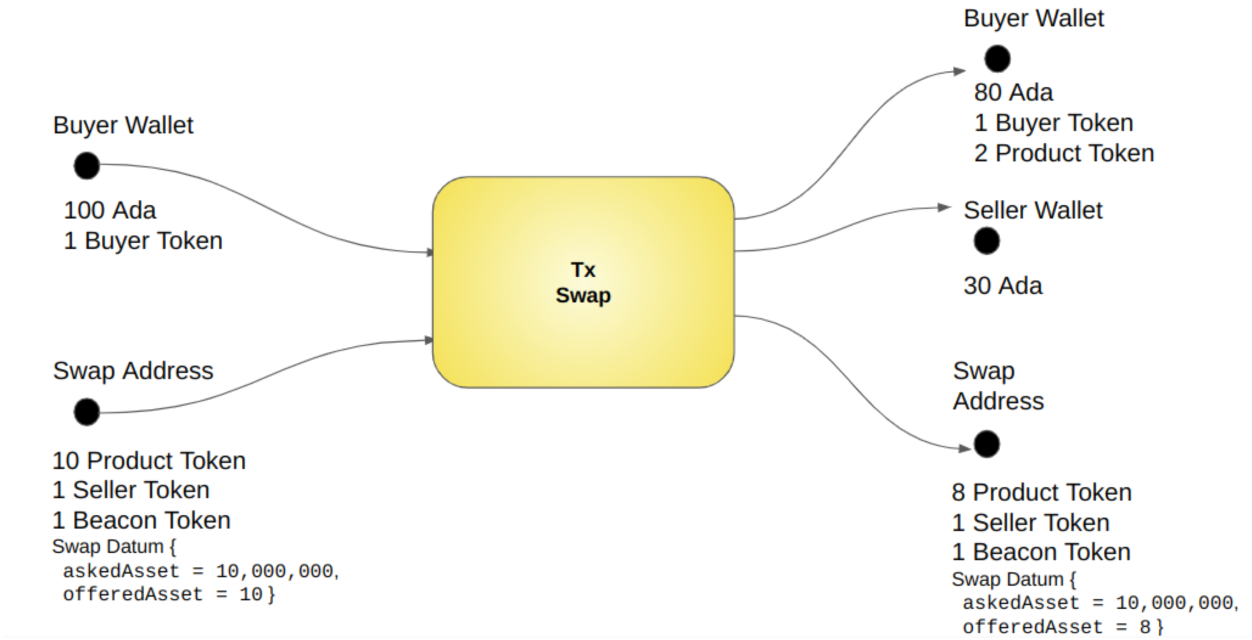
Swap Address

5 Product Token
1 Seller Token
1 Beacon Token
Swap Datum {
 askedAsset = 15,000,000,
 offeredAsset = 5 }

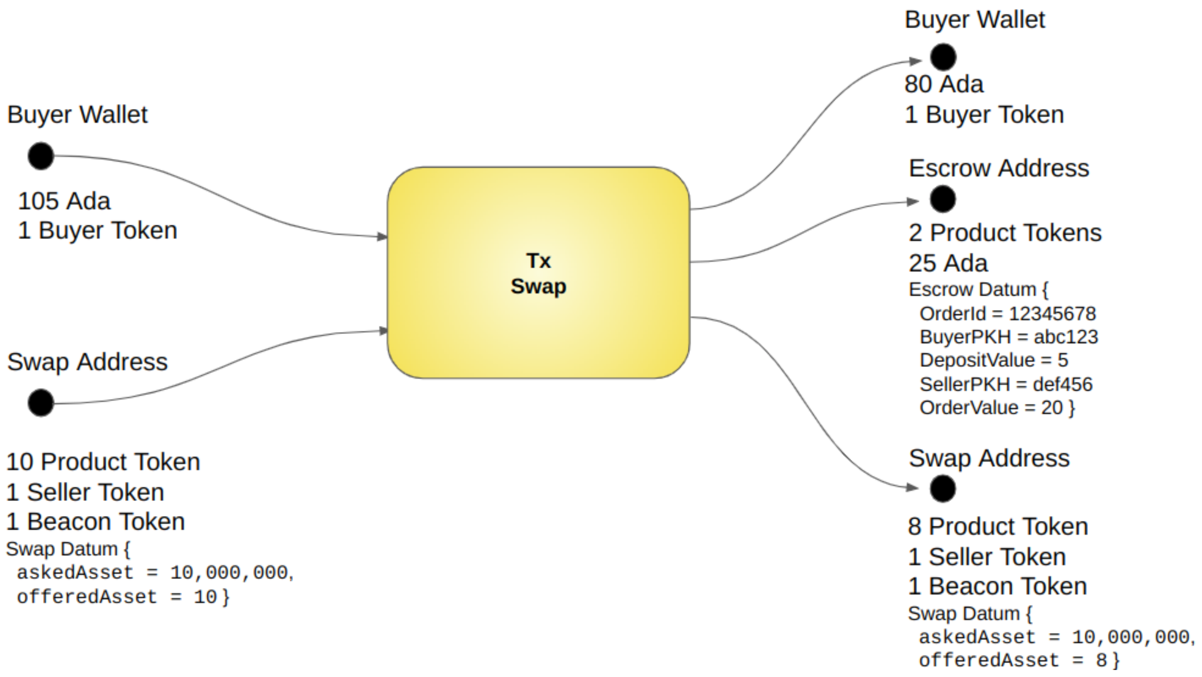
# Update Swap (Asked Price and/or Offered Qty)



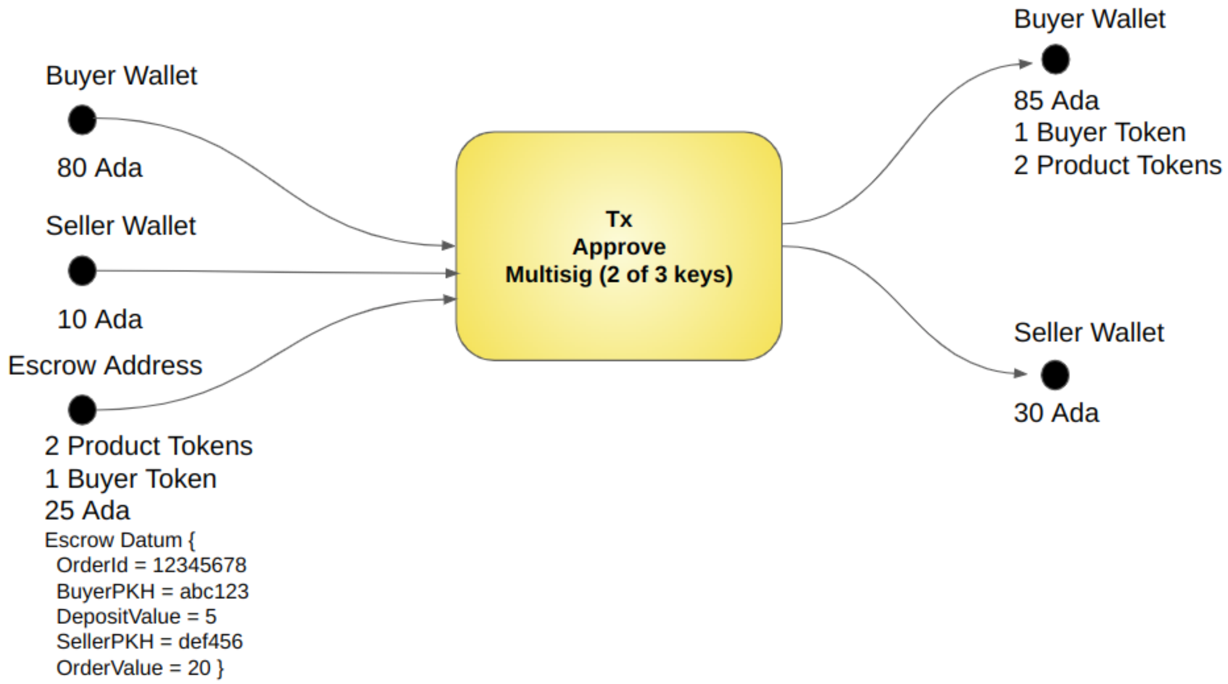
# Asset Swap


Buyer Wallet
100 Ada
1 Buyer Token
Swap Address
10 Product Token
1 Seller Token
1 Beacon Token
Swap Datum {
askedAsset = 10,000,000,
offeredAsset = 10 }
Tx
Swap
Buyer Wallet
80 Ada
1 Buyer Token
2 Product Token
Seller Wallet
30 Ada
Swap
Address
8 Product Token
1 Seller Token
1 Beacon Token
Swap Datum {
askedAsset = 10,000,000,
offeredAsset = 8 }

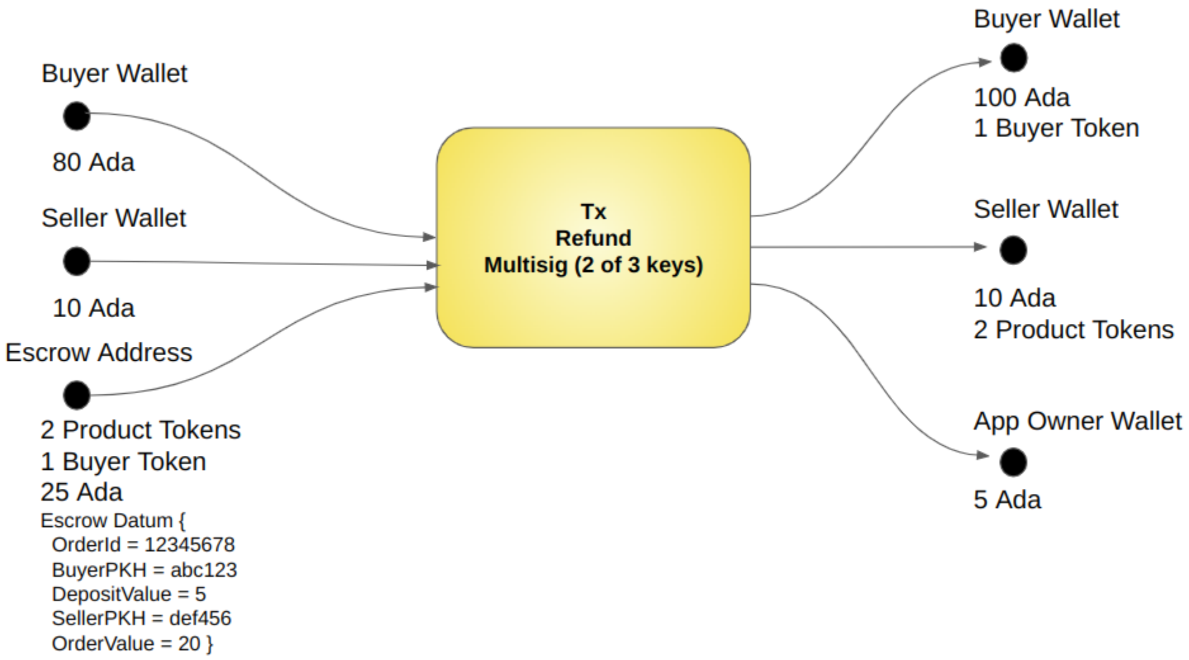
# Asset Swap Escrow



## Approve Escrow (Seller and Buyer approve)



# Refund Escrow (Dispute with Seller)



Buyer Wallet
80 Ada

Seller Wallet
10 Ada

Escrow Address
2 Product Tokens
1 Buyer Token
25 Ada
Escrow Datum {
  OrderId = 12345678
  BuyerPKH = abc123
  DepositValue = 5
  SellerPKH = def456
  OrderValue = 20 }

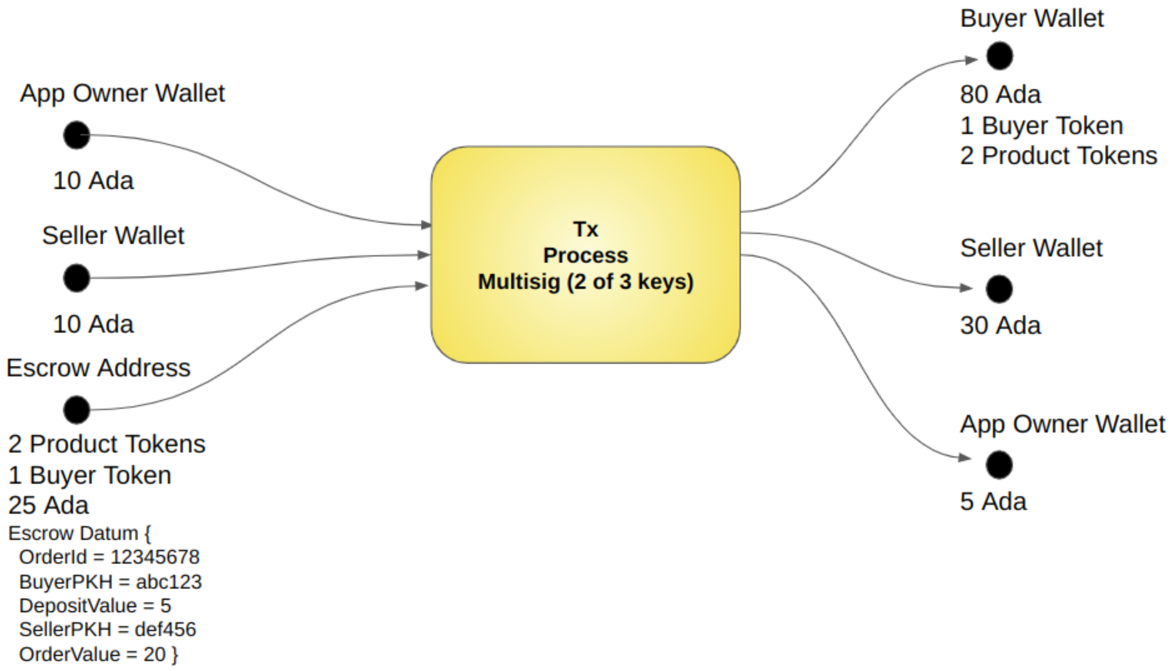
**Tx**
**Refund**
**Multisig (2 of 3 keys)**

Buyer Wallet
100 Ada
1 Buyer Token

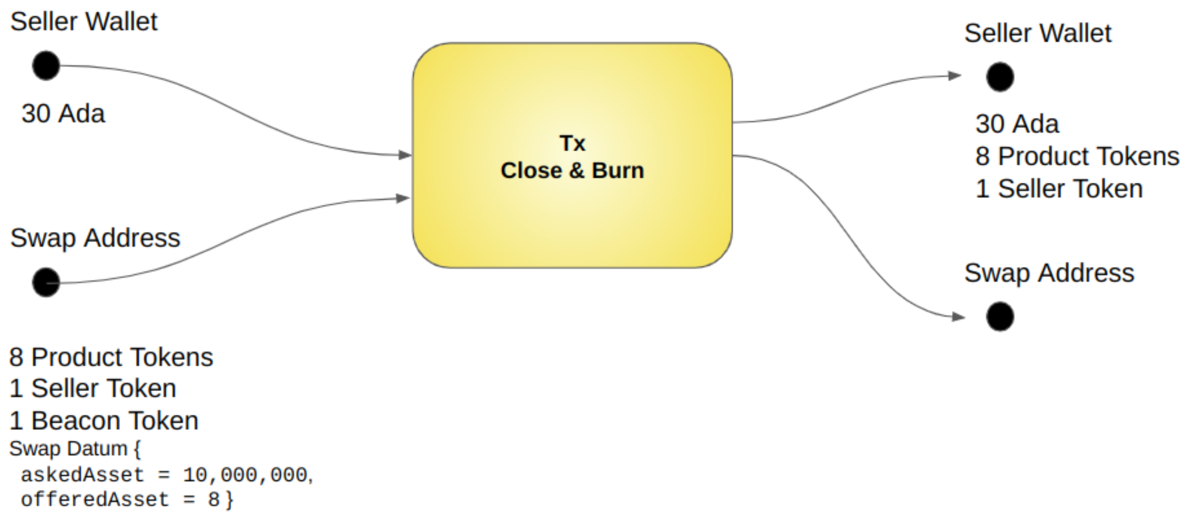
Seller Wallet
10 Ada
2 Product Tokens

App Owner Wallet
5 Ada

# Process Escrow (Dispute with Buyer)



# Close Swap


Seller Wallet
30 Ada
Swap Address
8 Product Tokens
1 Seller Token
1 Beacon Token
Swap Datum {
askedAsset = 10,000,000,
offeredAsset = 8 }
Tx
Close & Burn
Seller Wallet
30 Ada
8 Product Tokens
1 Seller Token
Swap Address