Architecture overview for the Random Swap dApp

Summary

In this document we describe the overall architecture of the Random Swap dApp web functionality.

The dApp is the second part of the NFT Guild Catalyst proposal to deliver an open source portfolio of smart contracts and dApp templates that make swaps of NFTs possible on Cardano. The project has already delivered a specific swap solution which allows a collector to select both the NFT they want to receive and also the NFT from their wallet they want to “give” (swap in) as part of a swap interaction.

The random swap is designed to be more exciting as the user will be unable to see which NFT they will receive from a swap interaction. Think of this as a lottery where you give away something of value but can get something of larger value back. In this case, what you receive can be a very rare, or in other way, very valuable NFT. Different from a lottery however, you always get something in return, but it might be an NFT of lower or similar value than what you swapped in. We describe the architecture of the random swap by first describing how transactions and smart contracts on Cardano work. Then we will present a historical and functional description of how we implemented the specific swap solution, as it follows much of the same concept. In some places they are even the same solution with a common code base.

The structure of a dApp on Cardano

A decentralized application, or dApp for short, is a web application like any other website, but with the functionality that it interacts with a blockchain. A very simple example of a dApp is a web page that lets you connect to your blockchain wallet, using a wallet browser extension, and send coins or other tokens from your wallet to the wallet of one of your contacts. When you do this, the transaction is saved on the blockchain, and all of the details, like creating the transaction and submitting this to the blockchain is handled by the dApp. You only need to input your password.

More complex examples can be large explorer dApps that let you navigate and explore the blockchain data and DeFi (Decentralized Finance) dApps that let you buy, borrow or lend tokens.

The simple example mentioned above makes use of regular wallets and you control the spending of funds because you are the only one with access to the secret spending keys of that particular wallet address. The DeFi dApps on the other hand usually make use of another kind of wallet, called a smart contract, where spending of its tokens is controlled by

a script. There are multiple types of smart contracts, but the most used type is a so-called validator, that runs a script to validate if some particular spending of the tokens it holds is allowed or not.

There is a crucial difference between these two wallet types and this is the fact that you can trust the behavior of a smart contract validator, because it operates according to the script that controls its funds. If you are able to read and understand the script that is used by a validator, you can trust that it will behave according to those rules when you interact with it. On the other hand, you cannot trust how the owner of a regular wallet will behave. If you send funds to a random wallet on a blockchain, you can not expect anything. In some cases, you get the funds returned because the receiver is trustworthy and will not want anything that is not earned, but just as likely you might not get anything back and your funds will be lost. Because smart contract validators are objective and operate according to scripted rules, these kinds of wallets are used for trustless exchange of tokens on a blockchain.

The specific swap architecture

When we designed the specific swap templates, we developed a collection of smart contracts that operate under different rules to support various ways of swapping NFTs. They guard the NFTs that can be swapped for other NFTs by the users and also validate that no one can empty the smart contract or in other ways cheat or steal from the contract by doing some illegal interaction. In the same way, the users can trust that they will receive something back when they interact with the smart contracts because they can inspect the code to see that it behaves according to the agreed rules.

The smart contracts are designed to be owned by a specific wallet. This ownership only gives the special authority to be able to remove NFTs from the smart contract; typically to close a swap pool smart contract after it has fulfilled its purpose. Other than this, the ownership does not give any other advantages. All swaps are validated strictly by inspecting which and how many NFTs are given to the swap pool and which and how many NFTs are asked for in return. No one, not even the owner is allowed to swap NFTs in a way that violates the swap rules.

To manage or create new swap pool contracts, an owner connects to their wallet using a special admin page in the dApp and specifies the rules for the swap pool contract. When the type of swap pool has been selected, the owner configures what policy ID the NFTs must be part of to be allowed to be swapped (change ownership) using this swap pool. In addition, there are different ways to select exactly which NFTs of the policy can be swapped. When the configuration is complete, the smart contracts are built and the information, like address and script are given to the owner. The owner then adds this information to the specific swap dApp and collectors (users) are then able to swap their NFTs from this policy ID with the swap pool.

The overall structure of the specific swap pool solution is described with the following two figures:

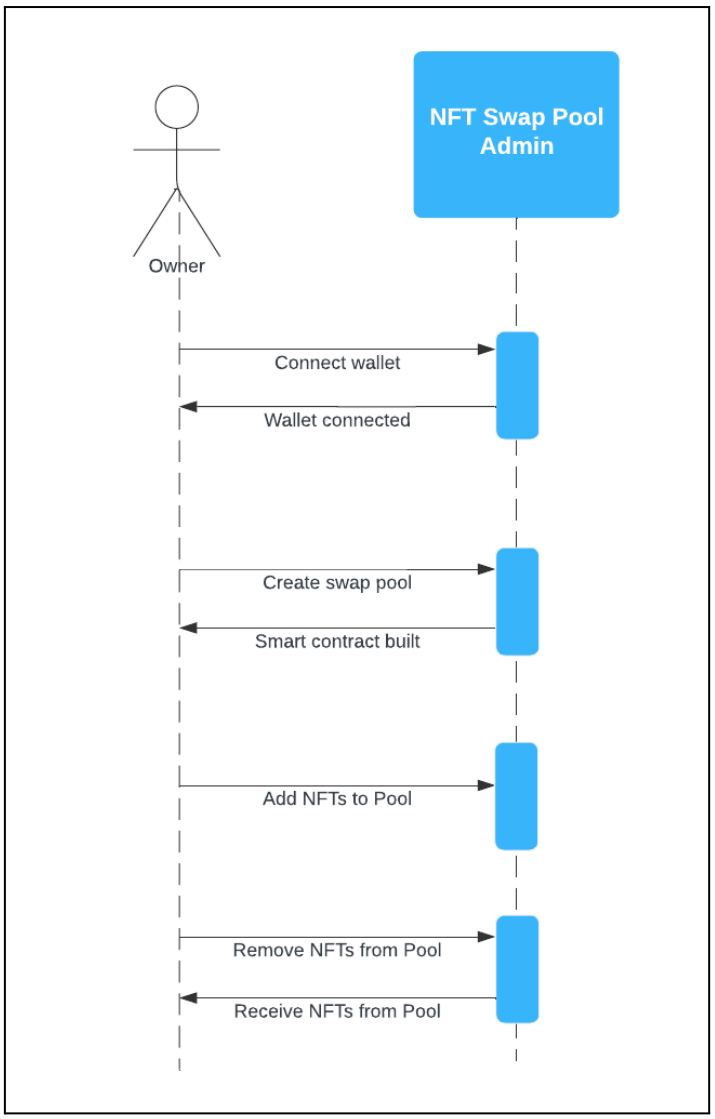

Figure 1: Specific Swap Pool Admin module

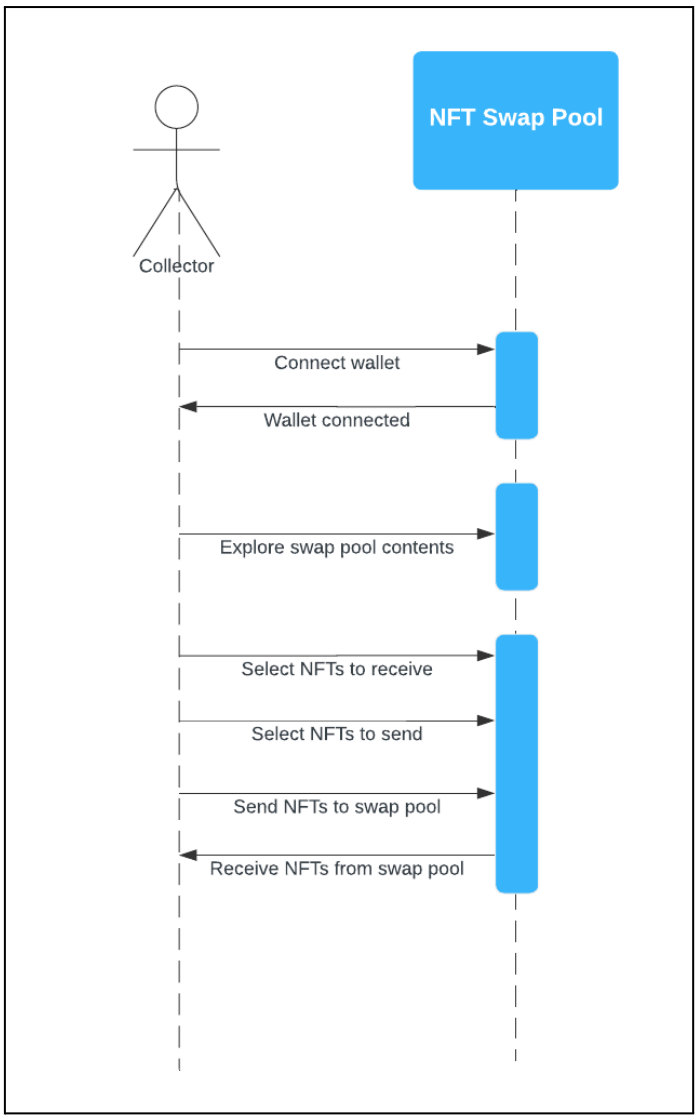

Figure 2: Specific Swap Pool module

The random swap architecture

Our random swap dApp is going to follow the same overall user experience as the specific swap. The Admin page is extended to include new actions for building the random swap pool contract and adding / removing NFTs to this type of pool. The difference is that when you create a random swap pool, two contracts are built for you; The *RandomSwap* and *RandomSwapQueue* smart contracts. The two solutions will feel very similar to work with. The main extension of the Admin page will be the swap queue contract which is marked with a blue arrow in Figure 3. After the creation of the contracts, the dApp is configured to make use of these contracts. When the dApp is configured correctly, the owner can add NFTs to the swap pool using the “Add NFTs” button.

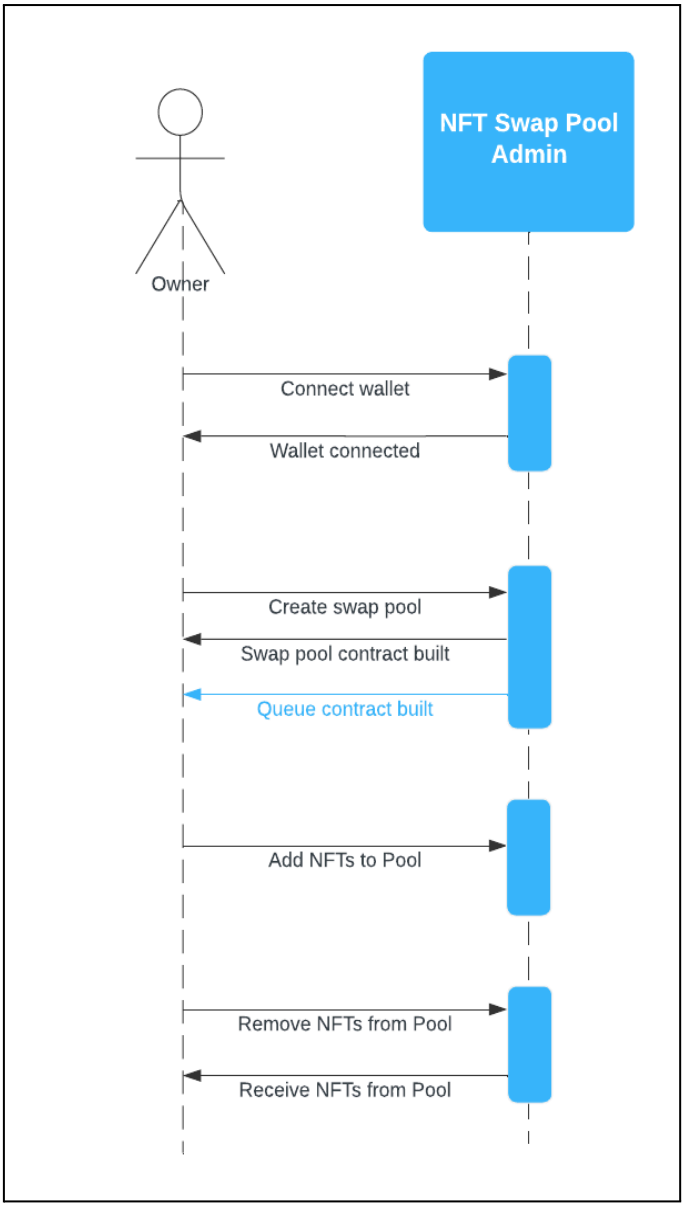

Figure 3: Random Swap Pool Admin module

The main differences between specific and random swap pool architecture will be in the swap pool functionality. Collectors will be able to explore the pool NFTs, but will naturally not be able to select which NFTs to receive. In addition, the returned NFTs are received in a subsequent transaction. As was described in the Adding Randomness Final Spec Report, we do this to inhibit users from rejecting swaps that are not to their liking. The whole idea of this concept is that the swap shall be random. The NFT selection is done randomly by supplying the swap smart contract with an oracle / reference UTxO that is used during the swap transaction. The random swap contract then validates the selection to ensure that it was selected at random according to the random oracle data.

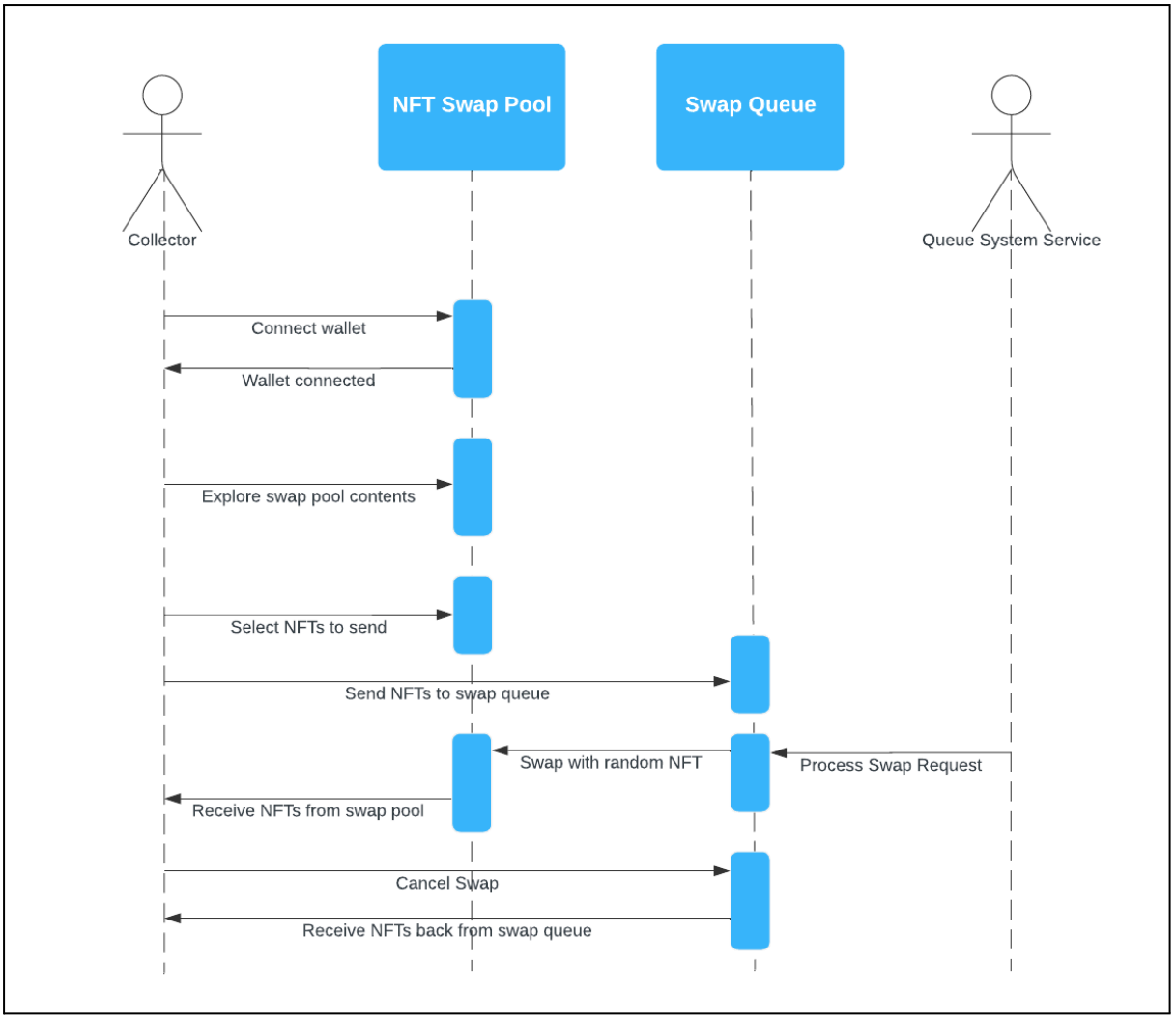

Figure 4: Random Swap Pool module

As can be seen in Figure 4, the swap architecture is more complex than for the specific swap in Figure 2. We have introduced two new components to the design, namely the Swap Queue and accompanying Queue System Service. The queue contract acts as a temporary storage for swap requests waiting to be processed, and the Queue System Service will process these requests in a FIFO (First In First Out) manner. In the random swap, the NFT to be swapped is sent from the collector's wallet to a queue smart contract. This queue is then processed by the Queue System Service which then places the NFT into the swap pool, exchanges it with a random NFT from the swap pool and then sends this back to the collector.

The Cancel Swap operation is also a new addition to the design. This operation is there to make sure Collectors can get their NFTs back in the unlikely event something fails or inhibits the Queue System Service to perform the swap.