ANASTASIA LABS

Payment Subscription Smart Contract

Design Specification

Contents

Payment Subscription Smart Contract

1. Overview

This Payment Subscription Smart Contract is developed using Aiken it is designed to facilitate automated recurring payments between subscribers and merchants on the Cardano blockchain. This contract empowers users to seamlessly set up, manage, and cancel their subscriptions directly from their wallets. It ensures secure and efficient transactions by automating the process of paying subscription fees, updating service metadata, and handling cancellations, all within a decentralized framework.

2. Architecture

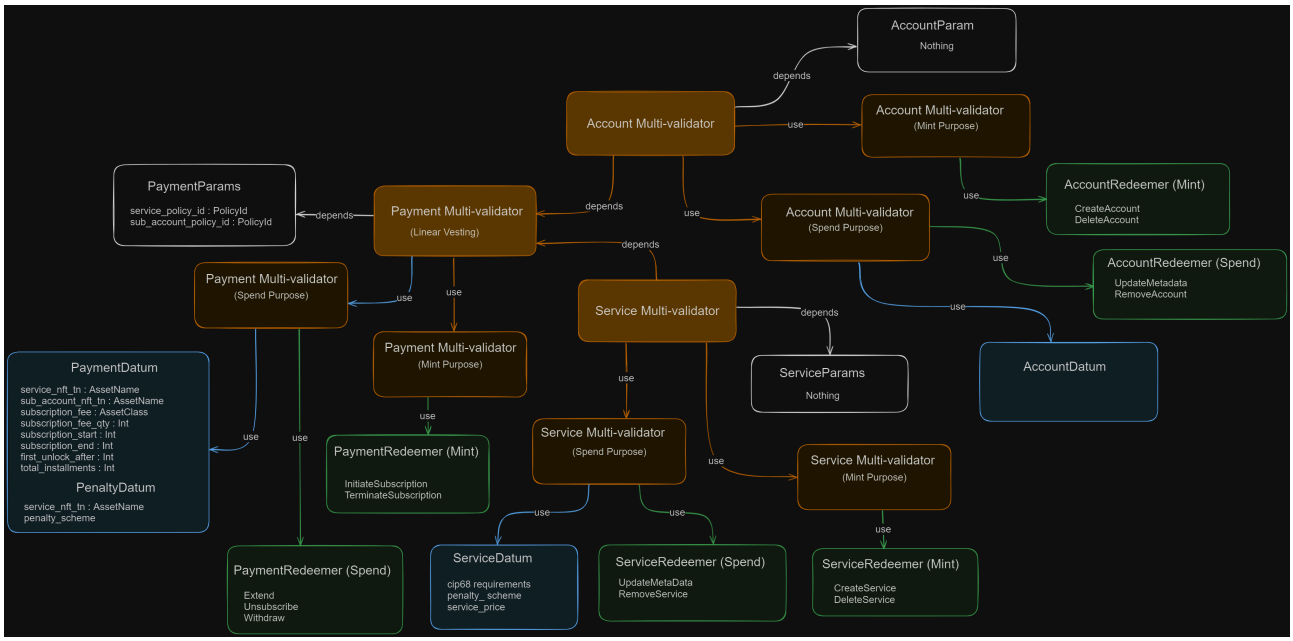

Figure 1: Payment Subscription Architecture

There are three contracts in this subscription system.

1. **Service Contract**

   A multi-validator responsible for creating an initial service by minting a single CIP-68 compliant Service NFT Asset and sending it to the user while sending the reference NFT to the spending endpoint. It also updates the metadata for the user and deletes the service by burning the Service NFT.

2. **Account Contract**

   A multi-validator responsible for creating an account for the user by minting a CIP-68 compliant Account NFT Asset and sending it to the user, while sending the reference NFT to the spending endpoint. It also updates the metadata for the Account and deletes the user account by burning a Account NFT.

3. **Payment Contract**

   This is the core validator. A multi-validator responsible for holding the prepaid subscription fees for a service, renewing a subscription to a service, unsubscribing from a service and withdrawing subscription fees. The contract incorporates a linear vesting mechanism to gradually release subscription fees to the merchant over the subscription period.

3. Specification

3.1. System Actors

1. **Merchant**

   An entity who interacts with the Service Contract to create a service and receive subscription payments for the respective service(s). A user becomes a merchant when they mint a Service NFT.

2. **Subscriber**

   An entity who interacts with the Account Contract to create an account and deposit prepaid subscription fees to the Payment Contract. A user becomes a subscriber when they mint an Account NFT and lock funds to the Payment Contract.

3.2. Tokens

1. **Service NFT**

   Can only be minted by a user when creating a service and burned when the user deletes their service(s) from the system.

   - **TokenName:** Defined in Service Multi-validator parameters with the hash of the Account Policy ID.

2. **Account NFT:**

   Can only be minted by a user when creating an account for the subscription system and burned when the user deletes their account from the system. A check must be included to verify that there are no payments in the Payment Contract before burning.

   - **TokenName:** Defined in Account Multi-validator parameters with the hash of the Account Policy ID

3. **Payment NFT:**

   Can only be minted when a subscription fee is paid to the Payment Contract and burned when a subscriber exits the system.

   - **TokenName:** Defined in Payment Multi-validator parameters with the hash of the Account Policy ID

3.3. Smart Contracts

3.3.1. Payment Multi-validator

The Payment Contract is responsible for managing the prepaid subscription fees, validating subscriptions, and ensuring the proper distribution of these fees over the subscription period given the `total_installments`. It facilitates the creation, extension, and cancellation of subscriptions, allowing both subscribers and merchants to interact with the contract in a secure and automated manner. This contract ensures that subscription payments are correctly handled and that any penalties for early cancellation are appropriately enforced.

3.3.1.1. Parameters

- **`service_policy_id`** : Hash of the PolicyId
- **`account_policy_id`** : Hash of the PolicyId

3.3.1.2. Minting Purpose

3.3.1.2.1. Redeemer

- InitiateSubscription
- TerminateSubscription

3.3.1.2.2. Validation

1. **InitiateSubscription**

   The redeemer allows creating of a new subscription by minting only one unique Payment Token.

   - Validate that out_ref must be present in the Transaction Inputs.
   - Validate that the redeemer only mints a single CIP68 compliant Payment Token.
   - Validate the datum from the Service Contract as a reference input.
   - Ensure that the User NFT doesn't go to the Script
   - Ensure Payment token goes back to the script

2. **TerminateSubscription**

- Validate the datum from the Service Contract as a reference input
- Validate that the redeemer only burns a single Payment Token.

3.3.1.3. Spend Purpose

3.3.1.4. Datum

This is a Sum type datum where one represents the payment datum and the other one represents a penalty datum.

3.3.1.4.1. Payment datum

- **`service_nft_tn`:** Service token name encoding UTxO to be consumed when minting the NFT.
- **`account_nft_tn`:** Account token name encoding UTxO to be consumed when minting the NFT.
- **`subscription_fee`:** AssetClass type for the subscription fee.
- **`subscription_fee_qty`:** Amount of the subscription fee.
- **`subscription_start`:** Start of the subscription.
- **`subscription_end`:** Expiry time of the subscription.
- **`last_claimed`:** The last time the merchant withdrew from the contract.
- **`penalty_fee`:** AssetClass type of the penalty.
- **`penalty_fee_qty`:** Amount the merchant wishes to penalies for early withdrawal. This is optional and can be `0`.

3.3.1.4.2. Penalty datum

- **`service_nft_tn`:** Service token name encoding UTxO to be consumed when minting the NFT.
- **`penalty_fee`:** AssetClass type for the amount of fees to be deducted when subscriber cancels the subscription (optional).
- **`penalty_fee_qty`:** Amount of the penalty fees (optional).

3.3.1.5. Redeemer

- Extend
- Unsubscribe
- MerchantWithdraw
- SubscriberWithdraw

3.3.1.6. Validation

1. **Extend**

   The redeemer will allow anyone to increase the subscription funds by locking the funds in the Payment Contract.

   - Validate that the value of the UTxO is increased as long as the Datum is updated with the Service NFT Token Name.
   - Validate that the extension of the service follows the Service provider rules / datum.

2. **Unsubscribe**

   The redeemer will allow anyone with an Account NFT to spend an Account UTxO to unlock funds back to their address.

   - Validate the user with an Account NFT asset spends a Payment UTxO.
   - Validate that the penalty UTxO is being produced with the merchants Token Name.
   - Validate that the logic follows the Service provider rules / datum as a refernce input to ensure the penalty goes to the merchant when paying the penalty fees.

3. **MerchantWithdraw**

   The redeemer will allow anyone with a Service NFT to withdraw funds from the Payment UTxO or Penalty UTxO.

   - Validate whether the transaction contains a penalty datum or a normal datum.
   - **Payment UTxO**
     - Validate that user with Service NFT spends a Payment UTxO
     - Validate the payment datum against the service datum
     - Implement linear vesting for fund release:
       - Calculate vesting periods based on time elapsed
       - Determine withdrawable amount based on vesting periods
   - Verify any remaining funds are returned to the payment validator

- Update the last_claimed field in the new datum
- **Penalty UTxO**
  - Validate that user with Service NFT spends a Penalty UTxO
  - Ensure the full penalty amount is withdrawn
  - Verify that the PenaltyNFT which is the same as the Payment NFT is burned

4. **SubscriberWithdraw**

   The redeemer will allow anyone with an Account NFT to withdraw funds from the Payment validator.

   - Verify that the service is inactive
   - Validate that user with Account NFT spends a Payment UTxO
   - Verify the subscriber receives all the refund amount
   - Verify that the Account NFT is preserved in the subscriber's output

3.3.2. Service Multi-validator

The Service Multi-validator is responsible for, creating, updating and removing a service.

3.3.2.1. Parameter

Nothing

3.3.2.2. Minting Purpose

3.3.2.2.1. Redeemer

- CreateService

3.3.2.2.2. Validation

1. **CreateService**

   The redeemer allows creating of a new subscription sevice by minting only one unique Service Token.

   - Validate that out_ref must be present in the Transaction Inputs
   - Validate that the redeemer only mints a single CIP68 compliant Service Token
   - Validate that the datum meets the service provider requirements.

3.3.2.3. Spend Purpose

3.3.2.3.1. Datum

- `service_fee`: AssetClass type for the amount to pay for a subscription service
- `service_fee_qty`: Amount of the funds to pay for a service.
- `penalty_fee`: AssetClass type for the amount of fees to be deducted when subscriber cancels the subscription (Optional).
- `penalty_fee_qty`: Amount of the penalty fees (Optional).
- `min_sub_period`: least amount of subscription period.
- `is_active`: Boolean indicating if the service is active

Note: Subscription fees can be based on length of period the subscriber pays for e.g. If they pay for one month, the fees are more than if they pay for 12 months. This introduces the need for a `min_sub_period`.

3.3.2.3.2. Redeemer

- UpdateMetaData
- RemoveService

3.3.2.3.2.1. Validation

1. **UpdateMetaData**

   The redeemer allows for updating the metadata attached to the UTxO sitting at the script address.

   - Validate that Service UTxO with a Service NFT is being spent.
   - Validate that the metadata of the Reference NFT token is updated within acceptable bounds.
   - Validate that the service fee should be limited to a range to prevent extreme flucutation in service fee by the service provider e.g +/-10%
   - Validate that the reference token is spent back to it's own address.
   - Ensure the NFT is still present in the output

2. **RemoveService**

   The redeemer allows the removal of a service by a merchant from the subscription system.

   - Validate that there are two script input and two Script output back to the script address.
   - Validate UTxO with Reference NFT is being spent.
   - Ensure the service is being inactivated.
   - Ensure the NFT is still present in the output.

3.3.3. Account Multi-validator

3.3.3.1. Parameter

Nothing

3.3.3.2. Minting Purpose

3.3.3.2.1. Redeemer

- CreateAccount
- DeleteAccount

3.3.3.2.1.1. Validation

1. **CreateAccount**

   The redeemer allows creating of a new subscription service account by minting only one unique Account Token.

   - Validate that out_ref must be present in the Transaction Inputs
   - Validate that the redeemer only mints a single CIP68 compliant Account Token
   - Validate the Datum has valid account inputs
   - Ensure that the User NFT doesn't go to the Script
   - Ensure ref_nft goes back to the script

2. **DeleteAccount**

   This redeemer allows burning of an Account NFT to remove the subscriber from the system. A Check That there's no payment for the delete account should be done off-chain.

   - Validate that the redeemer only burns a single CIP68 compliant Account Token and Account Reference Token.

3.3.3.3. Spend Purpose

3.3.3.3.1. Datum

- account_details: Arbitrary ByteArray
- cip-68 requirements

3.3.3.3.2. Redeemer

- UpdateMetaData
- RemoveAccount

3.3.3.3.2.1. Validation

1. **UpdateMetaData**

   The redeemer allows for updating the metadata attached to the UTxO sitting at the script address.

   - Validate that the Account UTxO with the Account NFT is being spent.
   - Validate that the metadata of the Reference NFT is updated within acceptable bounds
   - Validate that the UTxO with the Reference NFT of is sent to the spending endpoint.

2. **RemoveAccount**

   The redeemer allows the removal of an account by a subscriber from the subscription system. Must Check That there's no ADA in the payment UTxO in the off-chain code.

   - Validate that Account NFT and Account Reference NFT is spent.
   - Validate that there are two script input and one Script output back to the script address
   - Check that the reference NFT is burned
   - Ensure no output contains the reference NFT

4. Transactions

This section outlines the various transactions involved in the Payment Subscription Smart Contract on the Cardano blockchain.

4.1. Service Multi-validator

4.1.1. Mint :: CreateService

This transaction creates a new service by minting a Merchant NFT. This transaction is performed by the merchant to indicate that a new service is available.

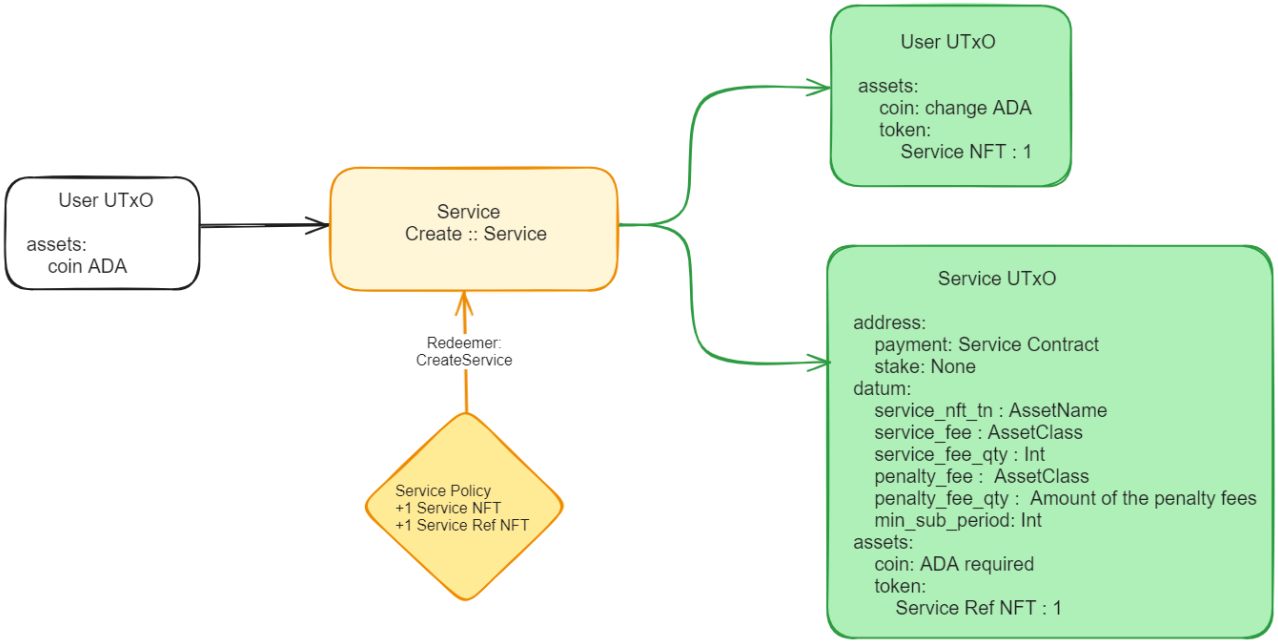

Figure 2: Create Service UTxO diagram

4.1.1.1. Inputs

1. **Merchant Wallet UTxO.**
   - Address: Merchant's wallet address
   - Value:
     - Minimum ADA

- Any ADA required for the transaction.

4.1.1.2. Mints

1. **Service Multi-validator**
   - Redeemer: CreateService
   - Value:
     - +1 Service NFT Asset
     - +1 Reference NFT Asset

4.1.1.3. Outputs

1. **Merchant Wallet UTxO:**
   - Address: Merchant wallet address
     - minimum ADA
     - 1 Service NFT Asset
2. **Service Validator UTxO:**
   - Address: Service Multi-validator Address (Mint)
   - Datum:
     - **`service_nft_tn`:** Service NFT token name encoding UTxO to be consumed when minting the NFT.
     - **`subscription_fee`:** AssetClass type for the subscription fee.
     - **`subscription_fee_qty`:** Amount of the subscription fee.
     - **`penalty_fee`**: AssetClass type for the amount of funds to be deducted when subscriber cancels the subscription.
     - **`penalty_fee_qty`**: Amount of the penalty fees.
   - Value:
     - 1 Service Reference NFT Asset

4.1.2. Spend :: UpdateMetaData

This transaction updates the metadata attached to the UTxO at the script address in accordance with CIP-68 standards. It consumes both the Service NFT and the Service Reference NFT, then sends the updated Service NFT to the user's wallet and the updated Reference NFT to the spending endpoint.

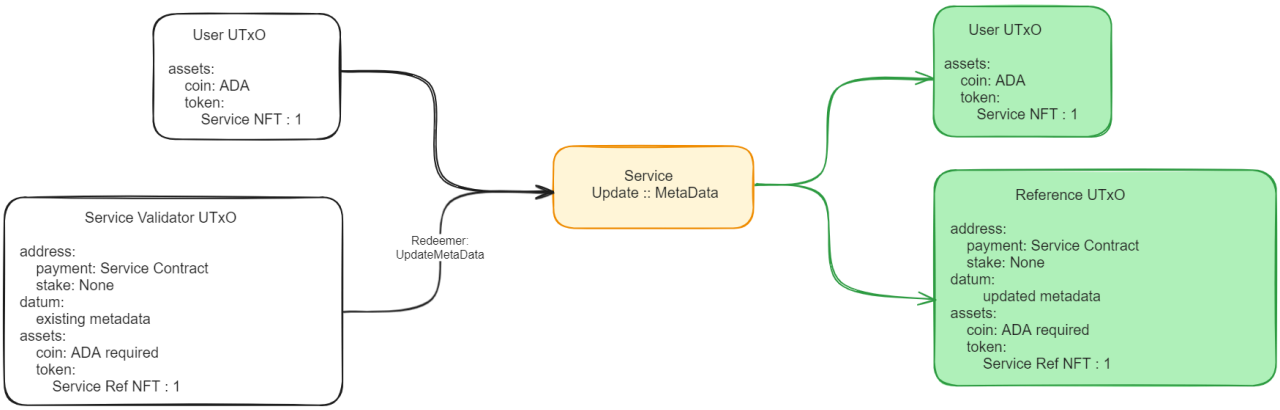

Figure 3: Update Service MetaData UTxO diagram

4.1.2.1. Inputs

1. **Merchant Wallet UTxO**
   - Address: Merchant's wallet address
   - Value:
     - Minimum ADA
     - 1 Service NFT Asset
2. **Service Validator UTxO**
   - Address: Service validator script address
   - Datum:
     - existing_metadata: listed in Section 3.3.2.3.1

.

- Value:
  - Minimum ADA
  - 1 Reference NFT Asset

4.1.2.2. Outputs

1. **Merchant Wallet UTxO**
   - Address: Merchant wallet address
   - Datum:
     - updated_metadata: New metadata for the subscription.
   - Value:
     - Minimum ADA
     - 1 Service NFT Asset
2. **Service Validator UTxO:**
   - Address: Service validator script address
   - Datum:
     - updated_metadata: New metadata for the subscription
   - Value:
     - Minimum ADA
     - 1 Reference NFT Asset

4.1.3. Spend :: RemoveService

This transaction spends the Reference UTxO and the Service NFT to remove a service from the system.

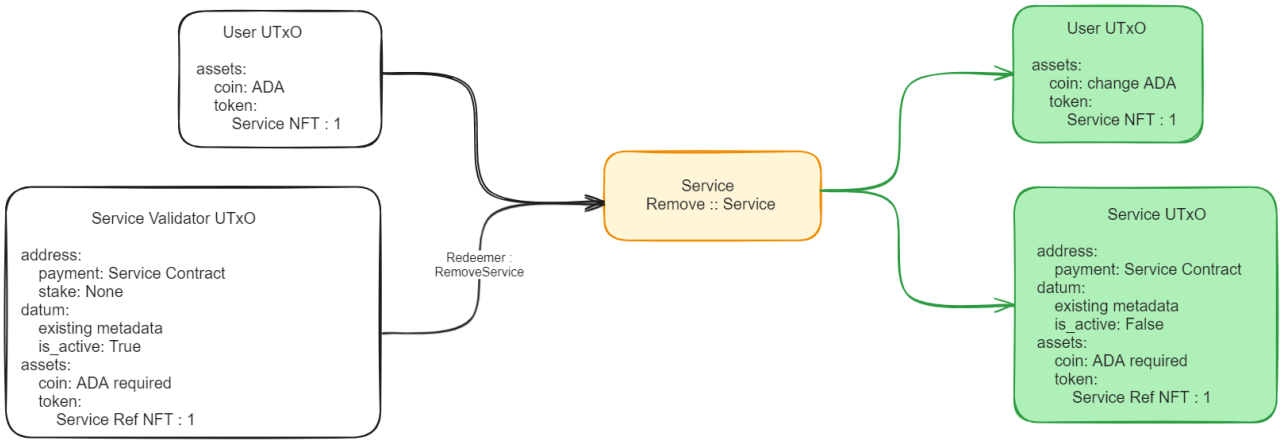

Figure 4: Remove Service UTxO diagram

4.1.3.1. Inputs

1. **Merchant Wallet UTxO**
   - Address: Merchant's wallet address
   - Value:
     - Minimum ADA
     - 1 Service NFT Asset
2. **Service Validator UTxO**
   - Address: Service validator script address
   - Datum:
     - service_metadata: Current metadata listed in Section 3.3.2.3.1.
     - is_active: True
   - Value:
     - Minimum ADA
     - 1 Reference NFT Asset

4.1.3.2. Outputs

1. **Merchant Wallet UTxO**
   - Address: Merchant wallet address
   - Value:
     - Minimum ADA
     - 1 Service NFT Asset
2. **Service Validator UTxO**
   - Address: Service validator script address
   - Datum:
     - service_metadata: Current metadata listed in Section 3.3.2.3.1.
     - is_active: False
   - Value:
     - Minimum ADA
     - 1 Reference NFT Asset

4.2. Account Multi-validator

4.2.1. Mint :: CreateAccount

This endpoint mints a new subscription NFT for a subscriber, establishing a new subscription account.

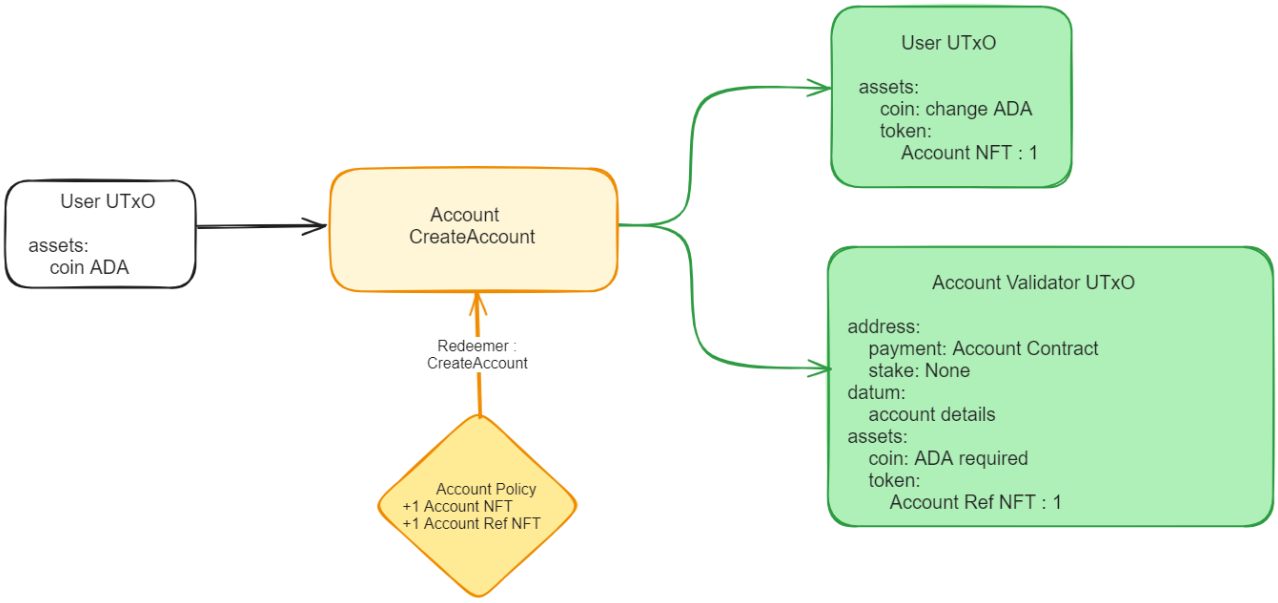

Figure 5: Create Account UTxO diagram

4.2.1.1. Inputs

1. **Subscriber Wallet UTxO.**
   - Address: Subscriber's wallet address
   - Value:
     - Minimum ADA
     - Any additional ADA required for the transaction

4.2.1.2. Mints

1. **Account Multi-validator**
   - Redeemer: CreateAccount

- Value:
  - +1 Account NFT Asset
  - +1 Reference NFT Asset

4.2.1.3. Outputs

1. **Subscriber Wallet UTxO:**
   - Address: Subscriber wallet address
   - Value:
     - minimum ADA
     - 1 Account NFT Asset
2. **Account Validator UTxO:**
   - Address: Account validator script address
   - Datum:
     - account_details: Arbitrary ByteArray
   - Value:
     - 1 Reference NFT Asset

4.2.2. Mint :: DeleteAccount

This transaction allows a subscriber to burn an Account NFT, effectively removing the user from the subscription system. An off-chain check is required to ensure that there are no pending subscription fees in the Payment UTxO.

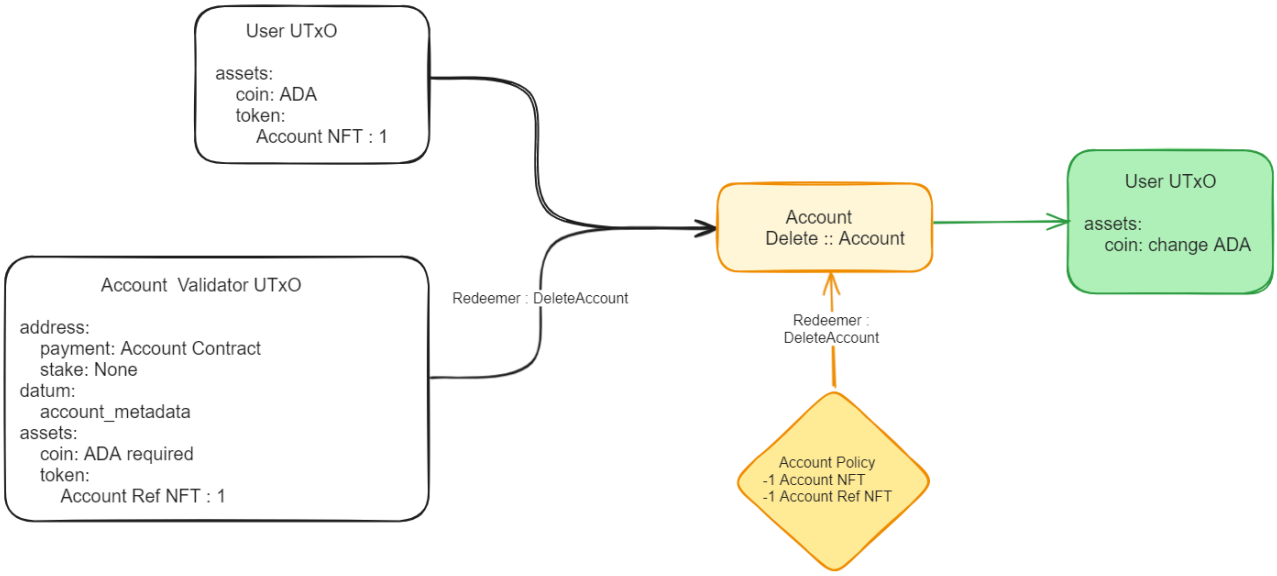

Figure 6: Delete Account UTxO diagram

4.2.2.1. Inputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - Minimum ADA
     - 1 Account NFT Asset
2. **Account Validator UTxO**
   - Address: Account Multi-validator script address
   - Datum:
     - account_details: Arbitrary ByteArray
   - Value:
     - 1 Reference NFT Asset

4.2.2.2. Mints

1. **Account Multi-validator**
   - Redeemer: DeleteAccount
   - Value:
     - −1 Account NFT Asset
     - −1 Reference NFT Asset

4.2.2.3. Outputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - Remaining ADA and other tokens, if any

4.2.3. Spend :: UpdateMetaData

This transaction updates the metadata attached to the Account UTxO at the script address. It consumes both the Account NFT and the Reference NFT, then sends the updated Subscriber NFT to the user's wallet and the updated Reference NFT to the spending endpoint.

Note: The service provider must query UTxOs regularly to facilitate data sync when datum is updated

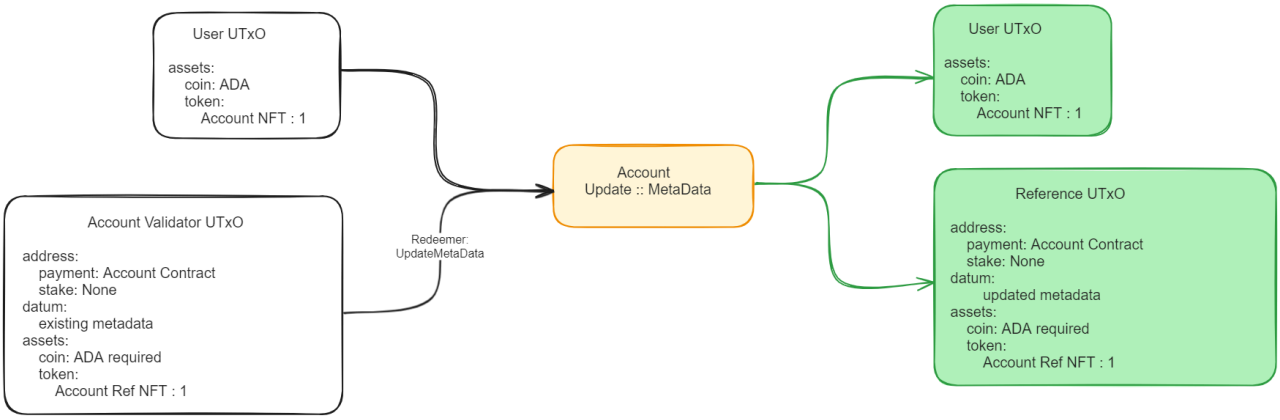

Figure 7: Update Account Metadata UTxO diagram

4.2.3.1. Inputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - Minimum ADA
     - Account NFT Asset
2. **Account Validator UTxO**
   - Address: Account validator script addres
   - Datum:
     - existing_metadata: Current metadata listed in Section 3.3.3.3.1.
   - Value:
     - Minimum ADA
     - 1 Reference NFT Asset

4.2.3.2. Outputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - Minimum ADA
     - 1 Account NFT Asset
2. **Account Validator UTxO**
   - Address: Account validator script address
   - Datum:
     - updated_metadata: updated metadata for the account listed in Section 3.3.3.3.1
   - Value:
     - Minimum ADA
     - 1 Reference NFT Asset

4.2.4. Spend :: RemoveAccount

This transaction effectively terminates the subscription and removes the subscriber's account from the system by consuming the Account NFT and the Reference NFT.

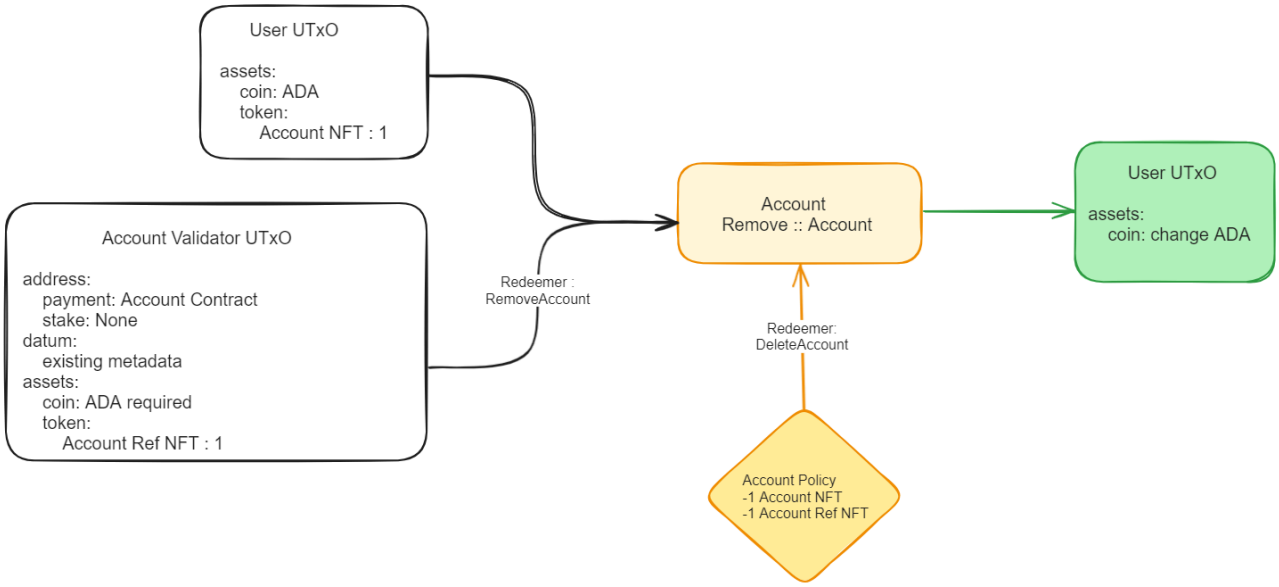

Figure 8: Remove Account Metadata UTxO diagram

4.2.4.1. Inputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - Minimum ADA
     - Account NFT Asset
2. **Account Policy UTxO**

- Address: Account Multi-validator Address (Spend)
- Datum:
  - account_datum: listed in Section 3.3.3.3.1
- Value:
  - Minimum ADA

- 1 Reference NFT Asset

4.2.4.2. Mints

1. **Account Multi-validator**
   - Redeemer: RemoveAccount
   - Value:
     - −1 Account NFT Asset
     - −1 Reference NFT Asset

4.2.4.3. Outputs

1. **Subscriber UTxO**
   - Address: Subscriber wallet address
   - Value:
     - Minimum ADA (remaining after burning the NFT)

4.3. Payment Multi-validator

4.3.1. Mint :: InitiateSubscription

This transaction occurs when a Subscriber locks funds in the Payment validator script address.

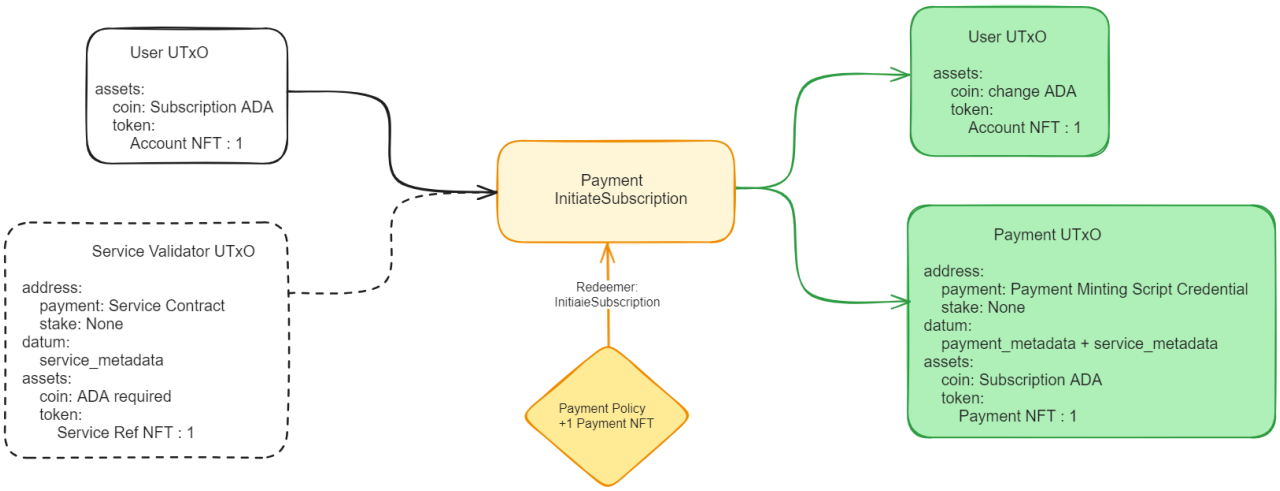

Figure 9: Initiate Subscription UTxO diagram

4.3.1.1. Inputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - 100 ADA: Amount of ADA to Add to the Payment Contract.
     - 1 Account NFT Asset

4.3.1.2. Mints

1. **Service Multi-validator**
   - Redeemer: InitiateSubscription
   - Value:
     - +1 Payment NFT Asset

4.3.1.3. Outputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - Change ADA
     - 1 Account NFT Asset
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Datum:
     - payment datum as listed in Section 3.3.1.4.1
   - Value:
     - 100 ADA: Subscription funds to be withdrawn by merchant
     - 1 Payment NFT Asset

4.3.2. Spend :: Extend

This transaction allows anyone to extend their subscription period by adding more funds to the Payment contract to cover additional time.

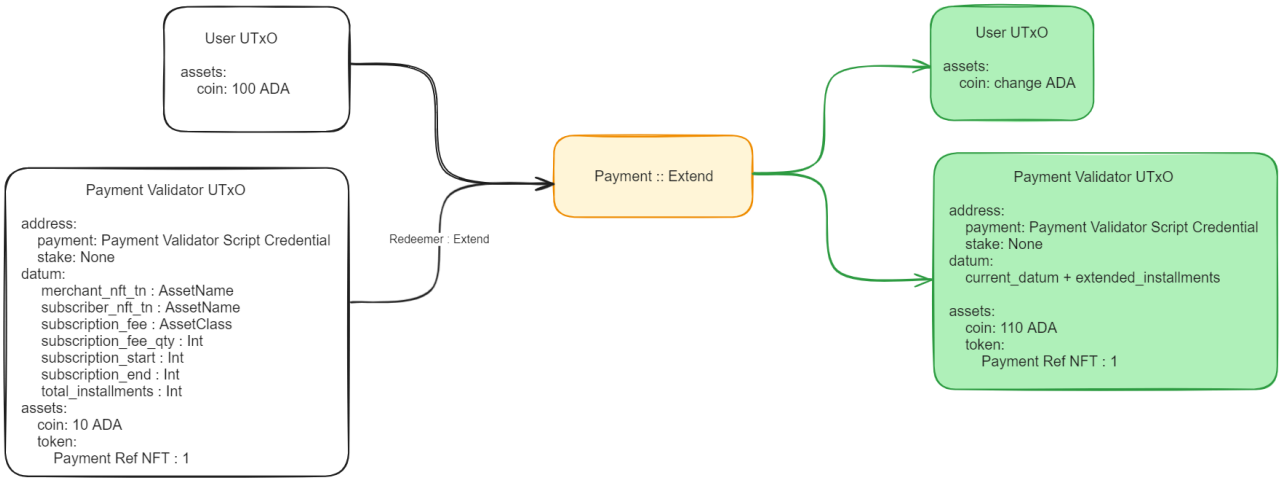

Figure 10: Extend Plan UTxO diagram

4.3.2.1. Inputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - 100 ADA: Amount of ADA to Add to the Contract to extend the payment plan
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Datum:
     - datum listed in Section 3.3.1.4.1
   - Value:
     - 10 ADA: Original amount of ADA before Extending
     - 1 Payment NFT Asset

4.3.2.2. Outputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - –100 ADA
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Datum:
     - datum listed in Section 3.3.1.4.1 + extended installments
   - Value:
     - 110 ADA: Increased ADA to cover the extended subscription period
     - 1 Payment NFT Asset

4.3.3. Spend :: Unsubscribe

This transaction allows the owner of an Account NFT to unsubscribe from a particular service by spending a Payment UTxO, unlocking the remainig pre-paid subscription fee to their own wallet address and creating a Penalty UTxO.

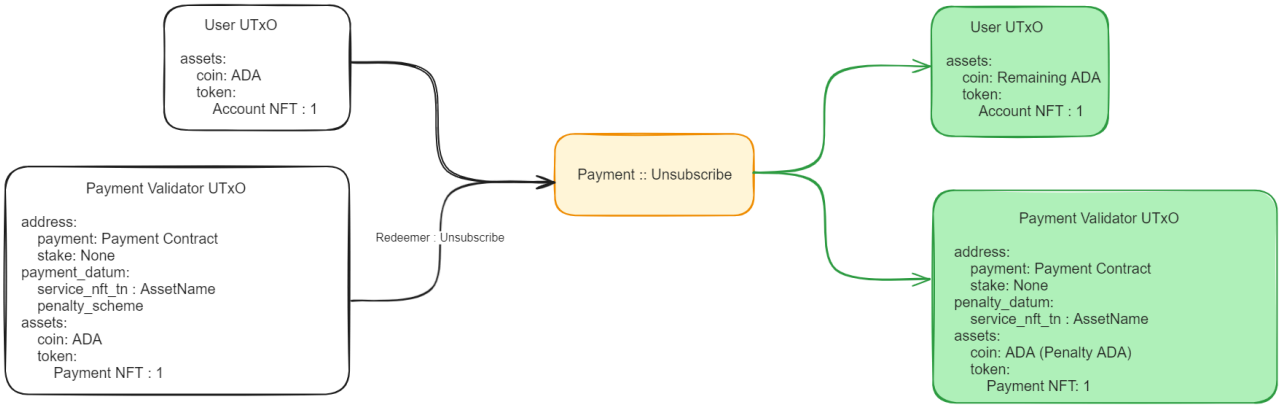

Figure 11: Unsubscribe UTxO diagram

4.3.3.1. Inputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - Minimum ADA
     - 1 Account NFT Asset
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Datum:
     - current_datum: Current payment metadata listed in Section 3.3.1.4.1
   - Value:
     - Minimum ADA
     - Reference NFT Asset

4.3.3.2. Outputs

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - Minimum ADA
     - Unspent portion of the subscription fee (minus any penalties)
     - 1 Account Token Asset
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Datum:
     - penalty_datum: Metadata indicating the penalty for early unsubscription in Section 3.3.1.4.2
   - Value:
     - Penalty ADA
     - Reference NFT Asset

4.3.4. Spend :: Withdraw

This transaction allows anyone with a Service NFT to unlock subscription funds from the Payment UTxO in repect to the specified subscription start and end dates.

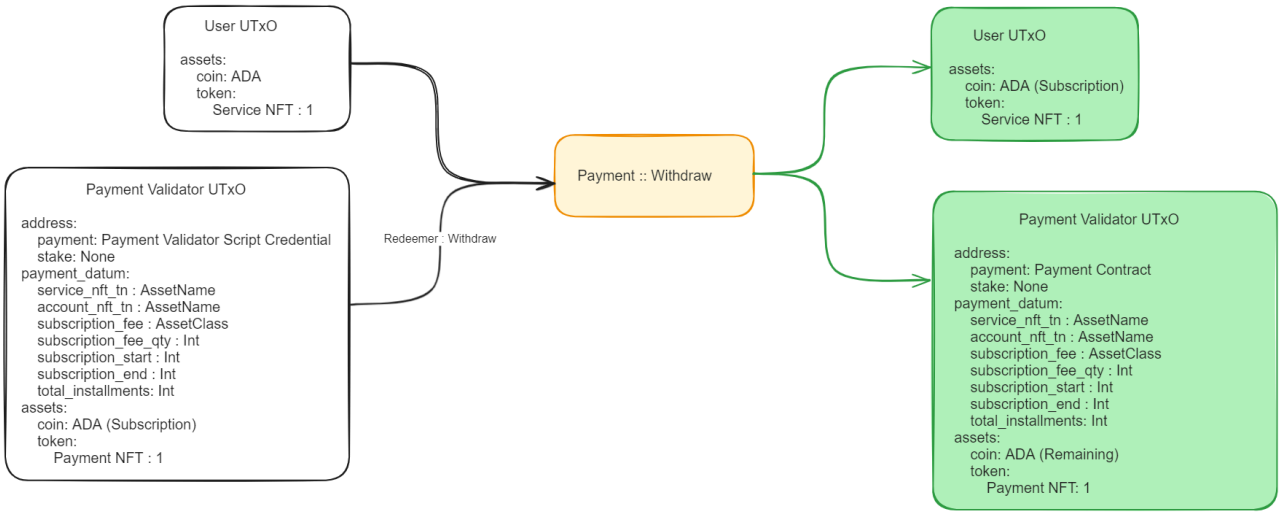

Figure 12: Merchant Withdraw UTxO diagram

4.3.4.1. Inputs:

1. **Merchant Wallet UTxO**
   - Address: Merchant's wallet address
   - Value:
     - Minimum ADA
     - 1 Service NFT Asset
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Datum:
     - payment_datum: listed in Section 3.3.1.4.1
   - Value:
     - Minimum ADA
     - 1 Payment NFT Asset

4.3.4.2. Outputs:

+ **Merchant Wallet UTxO**

- Address: Merchant's wallet address
- Value:
  - Minimum ADA
  - Withdrawn subscription fee for the installment
  - 1 Service NFT Asset

1. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Datum:
     - updated_datum: Metadata reflecting the withdrawal
   - Value:
     - Remaining ADA after withdrawal
     - 1 Payment NFT Asset

4.3.5. Spend :: Penalty Withdraw

This transaction allows anyone with a Service NFT to unlock penalty funds associated to the service from the Penalty UTxO, in turn burning the Payment NFT attached to the UTxO.

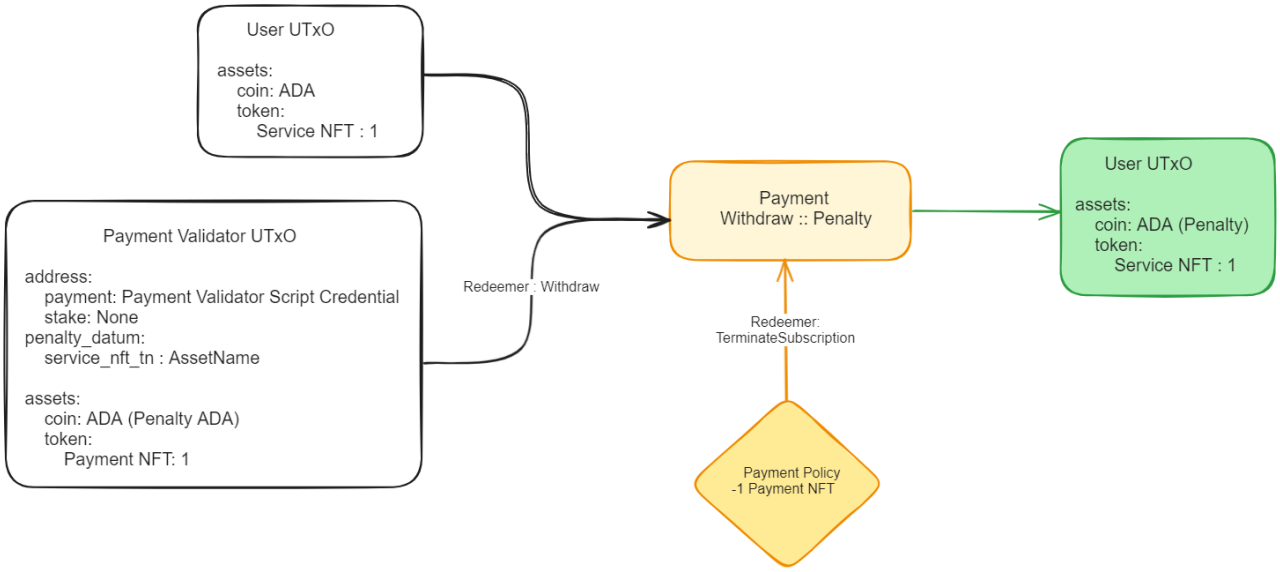

Figure 13: Penalty Withdraw UTxO diagram

4.3.5.1. Inputs:

1. **Merchant Wallet UTxO**
   - Address: Merchant's wallet address
   - Value:
     - Minimum ADA
     - Service NFT Asset
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Penalty Datum:
     - service-nft-tn: Service AssetName
   - Value:
     - Minimum ADA

- Payment NFT Asset

4.3.5.2. Mints

1. **Service Multi-validator**
   - Redeemer: TerminateSubscription
   - Value:
     - −1 Payment NFT Asset

4.3.5.3. Outputs:

1. **Merchant Wallet UTxO**
   - Address: Merchant's wallet address
   - Value:
     - Minimum ADA
     - Withdrawn subscription fee for the installment
     - Service NFT Asset
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Value:
     - Remaining ADA after withdrawal

4.3.6. Spend :: Subscriber Withdraw

This transaction allows anyone with an Accoun NFT to unlock subscription funds from the Payment UTxO only if the status in the ServiceDatum is inactive.

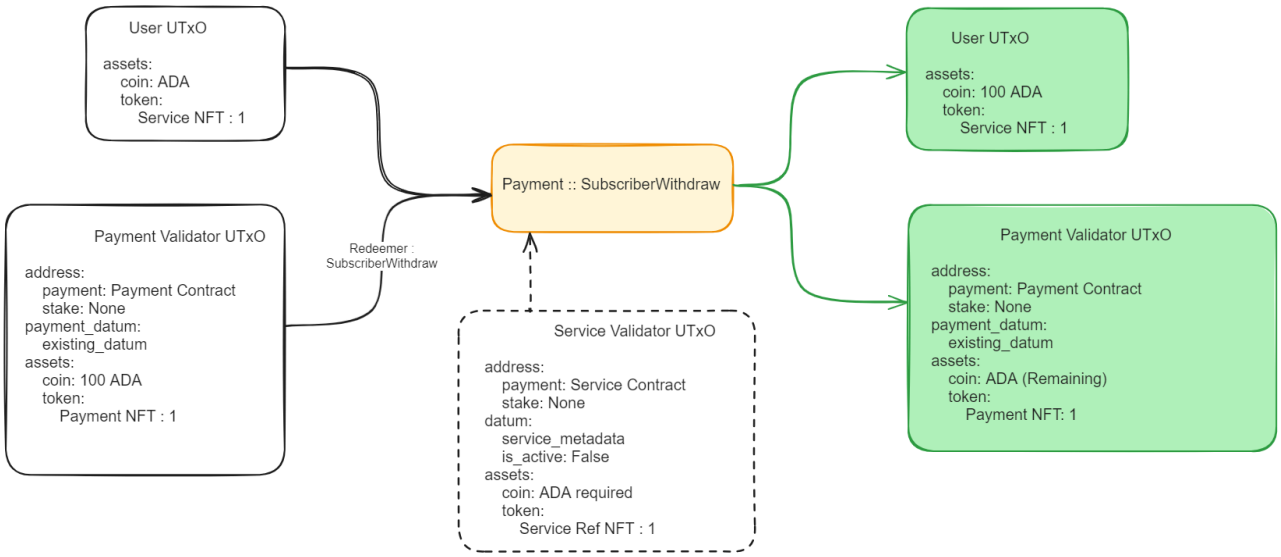

Figure 14: Subscriber Withdraw UTxO diagram

4.3.6.1. Inputs:

1. **Subscriber Wallet UTxO**
   - Address: Subscriber's wallet address
   - Value:
     - ADA
     - Account NFT Asset
2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Payment Datum:
     - existing_datum
   - Value:
     - Subscription ADA
     - Payment NFT Asset

4.3.6.2. Outputs:

1. **Subscriber Wallet UTxO**

- Address: Subscriber's wallet address
- Value:
  - Withdrawn Subscription ADA
  - Account NFT Asset

2. **Payment Validator UTxO**
   - Address: Payment validator script address
   - Payment Datum:
     - existing_datum
   - Value:
     - Remaining ADA after subscriber withdrawal