Automated Issuance Architecture for Tokenised Negotiable Certificates of Deposit (NCDs)

Technical Design Document
Target Network: Hedera (HSCS & HTS)

November 20, 2025

Abstract

This document outlines a "Factory-based" architecture for deploying Tokenised Negotiable Certificates of Deposit (NCDs) on the Hedera network. The design leverages the Hedera Token Service (HTS) for asset safety and the Smart Contract Service (HSCS) for logic enforcement. We propose a segregated architecture where an **ERC-4626 Vault** manages custody and maturity enforcement, while separate **Entry/Exit Adapters** manage the pricing logic required for Australian-style Discount Securities. This approach maximizes composability, auditability, and risk isolation.

Contents

1 Introduction

The digitization of Real World Assets (RWA) is shifting from manual, bespoke deployments to automated, API-driven factories. This architecture addresses the specific requirements of the Australian Money Market, where NCDs are treated as **Discount Securities** (purchased below par, redeemed at par).

1.1 The Problem

Current DeFi standards (like ERC-4626) assume a "Deposit Asset X $\rightarrow$ Get Asset X" workflow. However, a Discount NCD requires a "Pay Currency Y $\rightarrow$ Get Debt Token X" workflow. Furthermore, institutional issuance requires strict isolation of risk per deal.

1.2 The Solution: The "Deal Factory"

We introduce a design where a single JSON Request for Quote (RFQ) triggers the atomic deployment of a dedicated environment for that specific deal. This isolates the asset, the maturity logic, and the liquidity pool from all other deals on the platform.

2 Why Hedera?

While this architecture could exist on Ethereum, Hedera provides three distinct advantages for institutional RWA:

1. **HTS (Native Assets):** On Ethereum, a token is a smart contract ($ERC20.sol$). If the contract has a bug, the asset is lost. On Hedera, the token is a native ledger entity. The smart contract *wraps* the token but does not *contain* it. This significantly reduces "Smart Contract Risk."
2. **Token Association (Anti-Spam/Compliance):** Hedera requires accounts (and contracts) to explicitly "Associate" with a token before receiving it. This prevents "Dusting attacks" and ensures that the Vault contract explicitly opts-in to manage the NCD asset.
3. **Finality & Cost:** Fixed fees ($0.05 USD for associations/transfers) allow for precise business modeling, unlike gas-auction markets.

3 Architectural Design

The system follows a **Segregated Responsibility** pattern. We separate the *Storage of Value* from the *Transaction Logic*.

3.1 Core Components

- **The Deal Factory:** A singleton contract. It receives the RFQ parameters and spawns the deal-specific contracts.
- **The NCD Vault (ERC-4626):** The "Time Lock." It holds the NCD Token and issues transferable Shares. It is immutable and standard.
- **The Redemption Pool:** The "Exit." It holds the USDC required to pay back investors at maturity.
- **The Swap Router:** A singleton entry point. It calculates the discount price and executes the initial purchase.

3.2 Design Diagram: The Factory Spawn Process

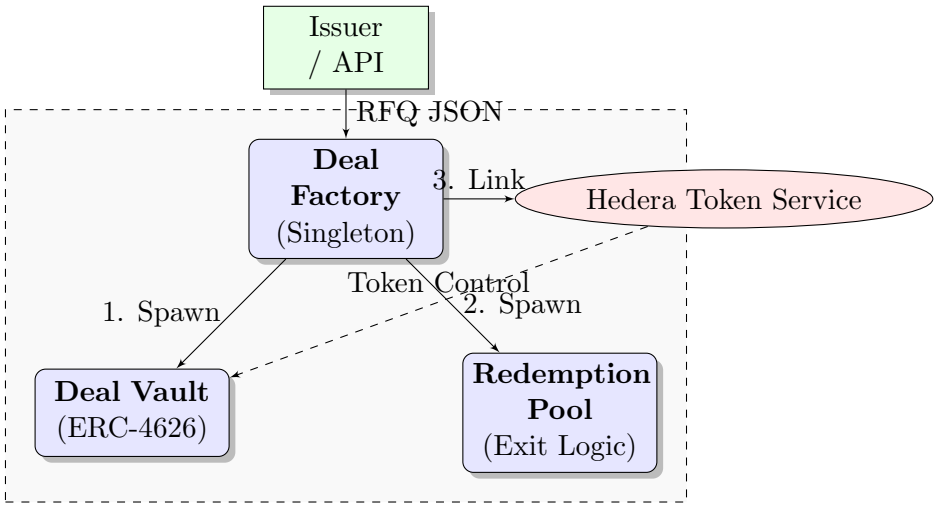

Figure 1: Atomic Deployment of a Deal Ecosystem

4 Design Rationale

4.1 Why separate the Vault from the Payment Logic?

We utilize the **Adapter Pattern**.

- **The Vault** is "dumb." It only knows how to hold Asset X and release it after Date Y. This makes it formally verifiable and chemically pure ERC-4626.
- **The Adapters** (Router/Pool) handle the "business." They handle the discount math ($Price = FaceValue/Yield$).
- **Benefit:** If we want to change from Unsecured NCDs to *Secured Collateralised Lending*, we only change the Asset passed to the Vault. The Vault code remains identical.

4.2 Why a new Vault for every Deal?

- **Asset Immutability:** ERC-4626 requires a fixed 'asset()' address. Since every NCD is a new HTS Token, every NCD needs a new Vault.
- **Maturity Immutability:** The maturity date is hardcoded into the Vault's constructor. This provides a trustless guarantee to the investor that the date cannot be changed by an admin key.
- **Risk Segregation:** If Borrower A defaults on Deal A, the liquidity for Deal B is in a completely different smart contract address.

5 Operational Run-Sheet

5.1 Phase 1: RFQ & Deployment

1. **Quote:** Investor accepts 5.00% Yield, $1M Face Value, Dec 31 Maturity.
2. **Mint:** Backend mints HTS Token "NCD-DEC31".
3. **Spawn:** Backend calls 'DealFactory.deployDeal(NCD-DEC31, USDC, Dec31)'.

4. **Result:** A new Vault and Redemption Pool are deployed on-chain.

5.2 Phase 2: Investment (The Discount Entry)

The Investor wants to buy $1M Face Value, but only pays $950k (Discount).

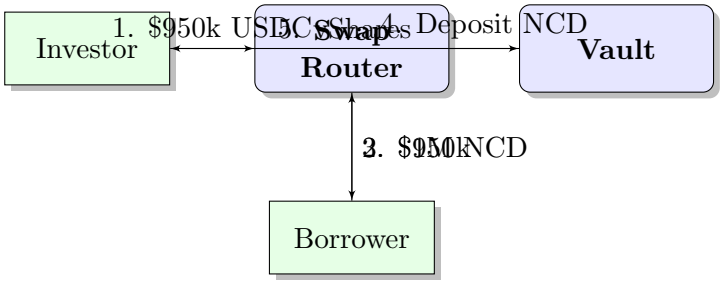

Figure 2: Cash Flow Diagram: The Discounted Entry

5.3 Phase 3: Redemption (The Par Exit)

At maturity, the Investor redeems Shares to get Face Value (in USDC).

1. **Pre-Funding:** Borrower sends $1M USDC to the **Redemption Pool**.
2. **Action:** Investor calls ‘redeem‘ on the Pool.
3. **Logic:**
   - Pool takes vShares → Withdraws NCD from Vault.
   - Pool burns NCD.
   - Pool sends $1M USDC to Investor.

6 Related Work & Industry Comparison

6.1 Abrdn / Archax (Hedera)

In 2022, Abrdn tokenized money market funds on Hedera.

- **Architecture:** They utilize a ”Permissioned Token” model. The logic is not in an open Vault but enforced via HTS Key configurations (KYC/Freeze).
- **Comparison:** Our proposal adds an open DeFi layer (ERC-4626) on top of the permissioned asset, allowing for greater composability with other applications while retaining HTS security.

6.2 Ondo Finance (Ethereum)

Ondo uses a similar ”Factory” pattern for creating tranches of US Treasuries (OUSG).

- **Architecture:** They separate the *Fund Management* contract from the *Token*.
- **Comparison:** Our design mirrors Ondo’s separation of concerns but is optimized for the specific ”Discount Security” math of the Australian market, whereas Ondo typically uses a Rebase or NAV-update model.

7 Conclusion

This architecture provides a robust, scalable foundation for institutional lending on Hedera. By leveraging the Factory Pattern, we ensure that every deal is chemically pure in its execution—segregated, immutable, and standard. The use of ERC-4626 ensures future compatibility with the broader DeFi ecosystem, while the Adapter pattern solves the specific complexity of settling Discount NCDs.