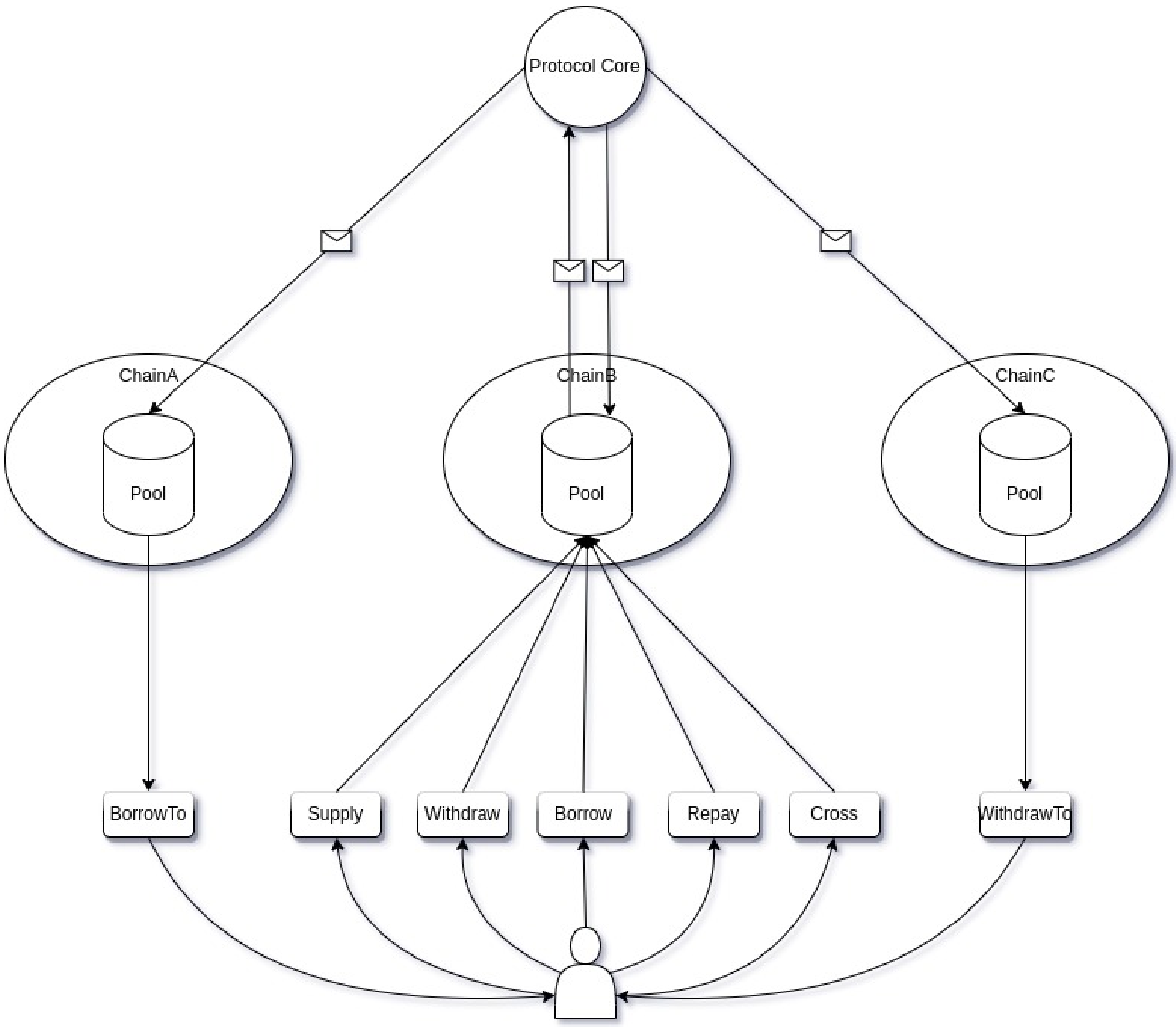
OMNI Protocol

1 Introduction

As the number of blockchain public chains continues to grow, there is a growing demand for full-chain operations from users. Cross-chain messaging protocols have become the new hotspot, and the gradual saturation of single-chain DeFi in the past, all of which mark the arrival of the all-chain era. In order to meet the development of blockchain in the all-chain era as well as the needs of users, the protocol has been designed by drawing on the Aave lending protocol and the cross-chain messaging protocol. The full-chain lending protocol is a protocol designed specifically for the full-chain. Compared with the Aave protocol, which provides liquidity in a single chain for lending and borrowing from a third-party cross-chain bridge for cross-chain lending, the full-chain lending protocol allows all lending operations to be performed in any chain. The full-chain lending protocol no longer relies on single-chain liquidity and supports cross-chain exchanges using protocol liquidity. This significantly increases the lending protocol's capital utilization and enables users to perform lending and borrowing operations without any perception of the chain. The full-chain lending protocol consists of a cross-chain messaging protocol, which is responsible for passing messages between different chains, and a borrowing protocol, which is responsible for handling user operation messages. In the full-chain lending protocol, users can make deposits and debits in any chain. On the one hand, when a user makes a deposit on a chain, liquidity is provided for that chain, which can be used not only for borrowing and lending but also for cross-chain and other operations; on the other hand, users can use the protocol's liquidity to borrow and cross-chain on any chain, and the interest and cross-chain fees they pay are distributed to the liquidity provider.

1.1 Basic Concepts

The full-chain lending protocol consists of a protocol core as well as multiple liquidity pools, with dedicated liquidity pools on each chain, and the protocol core is responsible for managing the liquidity of all chains. Through a bridge, the protocol core is able to communicate with the liquidity pools on each chain using a cross-chain messaging protocol. Users can use the protocol to deposit, withdraw, lend, repay, and cross-chain by interacting with the liquidity pools on each chain.



1.2 Symbol

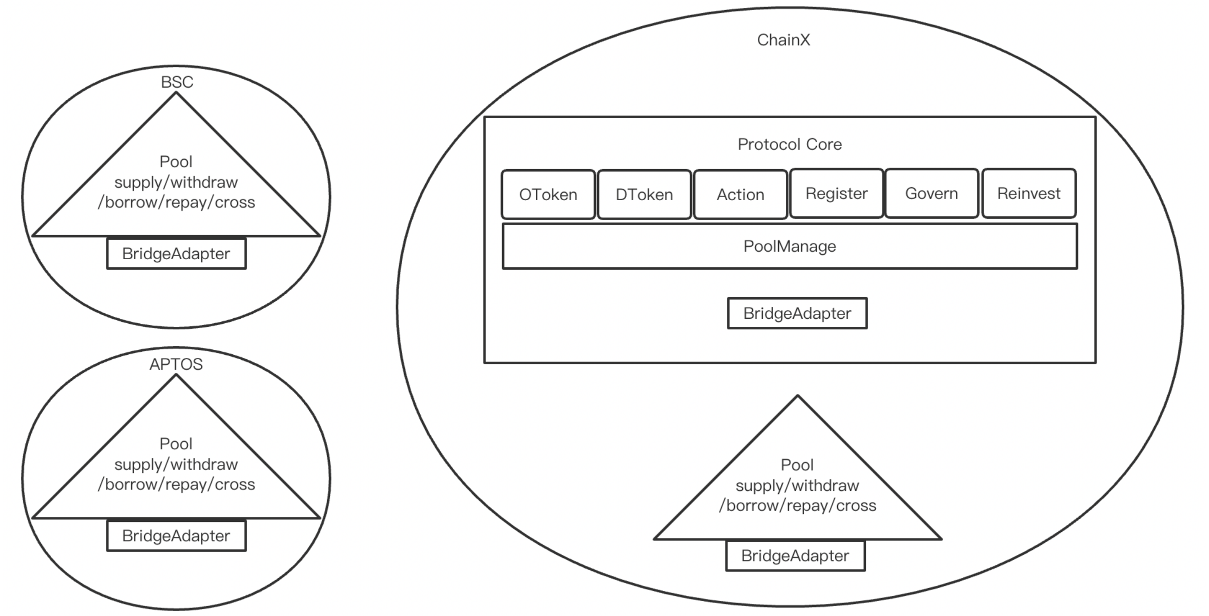
Variable	Formula	Description
T	-	Current number of seconds
T_l	-	Timestamp of the last update of the reserve data
ΔT	$\Delta T = T - T_l$	Delta time
T_{year}	$T_{year} = 365 * 24 * 60 * 60$	Number of seconds in a year
L_c	-	Current amount of liquidity available in the reserve

Variable	Formula	Description
D_c	-	Current amount of debt in the reserve
U_c	$U_c = \frac{D_c}{D_c+L_c}$	Representing the utilization of the deposited funds
$U_{optimal}$	-	The utilization rate targeted by the model, beyond the variable interest rate rises sharply
BR_b	-	base variable borrow rate, Constant for $D_t = 0$
BR_{slope1}	-	Constant representing the scaling of the interest rate versus the utilization, when $U_c < U_{optimal}$
BR_{slope2}	-	Constant representing the scaling of the interest rate versus the utilization, when $U_c >= U_{optimal}$
RF	-	Treasury Interest Factor
BR_c	$BR_c = \begin{cases} BR_b + U_c * BR_{slope1} & U_c < U_{optimal} \\ BR_b + BR_{slope1} + \frac{U_c - U_{optimal}}{1-U_{optimal}} * BR_{slope2} & U_c >= U_{optimal} \end{cases}$	Current borrow interest rate
LR_c	$LR_c = BR_c * U_c * (1 - RF)$	Current liquidity interest rate
BL_t	$BL_t = (\frac{BR_c}{T_{year}} + 1)^{\Delta T} * BL_{t-1},\ BL_0 = 1$	Cumulated borrow index

Variable	Formula	Description
CL_t	$CL_t = (\frac{LR_c * \Delta T}{T_{year}} + 1) * CL_{t-1},\ CL_0 = 1$	Cumulated liquidity index
CF	-	Collateral coefficient $[0, 1]$. A coefficient used to prevent the downside risk of collateral prices
BF	-	Borrowing coefficient $[0, 1]$. A coefficient used to prevent upside risk of loan prices
CS_c^t	-	Represents the liquidity of the token t on the c chain at the current moment
EDR	-	Expected distribution ratio of token liquidity on different chains
TR	-	The threshold of the EDR
SoT_t	-	Used to represent the number of scaled oTokens owned by the user at time t. $SoT_0 = 0$
oT_t	-	Used to represent the number of oTokens of owned by the user at time t

Variable	Formula	Description
SdT_t	-	Used to represent the number of scaled dTokens owned by the user at time t. $SdT_0 = 0$
dT_t	-	Used to represent the number of dTokens of owned by the user at time t

2 Protocol Architecture



2.1 Pool

The token pool is the pool of funds for the protocol and is responsible for hosting various assets on each chain. The token pool is stateless, and the state of users is unified and managed by Protocol Core. This approach makes the token pool very simple, and new public chains can quickly join the protocol, which is conducive to achieving the goal of chain-wide ecology. At the same time, the token pool is stateless, ensuring that users' top-ups, withdrawals, debits and repayments can be performed separately on different chains, greatly increasing the flexibility of users' operations. The main features of token pools:

- Token ID: The tokens in each token pool use $(chainId, tokenId)$ as a unique identifier in the protocol to facilitate the unified management of different tokens
- User Lending: Users interact with the protocol through the token pool's top-up, withdrawal, borrowing and repayment interfaces to achieve chain-wide Lending
- User Swap: Users implement chain-wide Swap through the cross-link port of the token pool and DeFi protocols on different chains

2.2 BridgeAdapter

Bridge adapters are adapters for cross-chain messaging protocols. By adapting cross-chain messaging protocols (e.g. LayerZero), it enables cross-chain transmission of protocol messages. In order to achieve the commonality of messages from different public chains, the protocol develops a unified message specification. Among them, the types of message attributes meet the unified conversion of numeric values into 64-bit integers, addresses into byte types, and other data are also unified with byte types.

The bridge adapter also acts as a bridge between the protocol and the cross-chain messaging protocol. The bridge adapter receives cross-chain messages from the token pool or token pool manager, generates events and message encoding, and sends the encoded data out through the cross-chain messaging protocol. It also receives cross-chain messages from the cross-chain messaging protocol, decodes the messages and generates events, and delivers the cross-chain messages to the token pool or token pool manager. When the destination chain of a cross-chain message is the current chain, the cross-chain message is passed directly to the token pool manager.

2.3 PoolManage

Token Pool Manager is a unified manager of token pools on different chains, including token classification of token pools, liquidity management, etc. It also receives cross-chain messages from token pools on different chains via bridge adapters and updates the protocol status of users and assets using the interaction module. As a token pool manager, the main functions implemented by the Token Pool Manager are.

- Managing token types: classifying tokens on different chains. Classification is the mapping of the same tokens on different chains to the same class (e.g. USDT). Each category corresponds to an ERC20 shadow token in the Token Pool Manager. The shadow tokens help manage the global state of the token (e.g. USDT) across the protocol. As an example, ideally the total supply of shadow tokens represents the sum of the number of tokens held in custody in different token pools on the chain at the current moment. At the same time the shadow tokens would serve as the actual escrow assets at the core of the protocol, helping to implement Lending related operations.
- Expected Distribution Ratio (EDR): used to manage the token pool liquidity. The token pool manager not only maintains the global state of tokens through shadow tokens, but also records the liquidity of the same type of tokens on different chains. The expected

distribution ratio refers to the expected distribution of tokens of the same class across different chains. It can be set manually by governance, and in the future it can be automatically adapted by a suitable algorithm. CS_c^t is used to denote the liquidity that tokens t on c chains have at the current moment. TR is used to denote the threshold value (e.g., 0.2). By setting different thresholds and the liquidity of c chain tokens t satisfies the following equation: the protocol inhibits or even prohibits the depletion of c chain tokens t liquidity. At the same time, it encourages the act of increasing the liquidity of c chain tokens t. Encouragement is provided by waiving fees, accrued interest rewards, treasury rewards, governance token rewards, etc.

$$\frac{CS_c^t}{\sum_c CS_c^t} <= EDR * TR$$

- Managing user addresses: In order to meet the chain-wide ecological goals of the protocol, address binding is required before any operation can be performed on non EVM chains (e.g. APTOS). Address binding is the binding of a user's EVM address to a non EVM address.
- Manage behavior status: The user's interactions through the token pool are received and processed by the token pool manager through the bridge adapter for cross-chain messages. The processing results are recorded by the token pool manager through storage or events to record the behavior state. The recording of behavior state facilitates the tracking of user interactions and thus ensures that user interactions are processed by the protocol in full.

2.3.1 oToken

The entitlement token oToken is a derivative token received by the depositor upon deposit, 1:1 mapping shadow tokens in the Token Pool Manager. The entitlement token oToken is automatically accumulated as CL_t increases, and the increase represents the interest received by the depositing user. SoT_t is used to indicate the number of scaled oToken the user has at moment t. m is used to represent the number of tokens with m deposited/withdrawn by the user. oT_t is used to denote the number of oTokens the user has at moment t. The number of entitled tokens oToken is determined by both SoT_t and CL_t, as follows.

User deposit:

$$SoT_t = SoT_{t-1} + \frac{m}{CL_t}$$

$$oT_t = SoT_t * CL_t$$

User withdrawal：

$$SoT_t = SoT_{t-1} - \frac{m}{CL_t}$$

$$oT_t = SoT_t * CL_t$$

2.3.2 dToken

The debtification token dToken is the debt incurred by the borrower to borrow money. The debtified token dToken is automatically accumulated as BL_t increases, and the increase represents the amount of interest to be paid by the borrower user. SdT_t is used to indicate the number of scaled dTokens a user has at time t. m is used to represent the number of tokens that the user borrowed/repaid in m. dT_t is used to denote the number of oToken the user has at moment t. The number of debtified tokens dToken is determined by both SdT_t and BL_t, as follows.

User Borrow：

$$SdT_t = SdT_{t-1} + \frac{m}{BL_t}$$

$$dT_t = SdT_t * BL_t$$

User Repay：

$$SdT_t = SdT_{t-1} - \frac{m}{BL_t}$$

$$dT_t = SdT_t * BL_t$$

2.3.3 Action

The interaction module is used to receive protocol messages from the token pool manager and update the protocol status based on user behavior using the protocol module. The interaction module checks the validity of the behavior based on the user behavior. For example, when a user borrows, the interaction module obtains the collateral price through the prediction machine and checks whether the collateral value meets the borrowing requirements. By checking the validity, the interaction module updates the protocol status of the equitized token, the debt token, including updating the borrowing rate BR_c, the liquidity rate LR_c, the borrowing index BL_t, and the liquidity index CL_t.

2.3.4 Register

The registration module is mainly used to introduce new chains and new tokens. The registration module requires super privileges through governance to complete the related operations. When adding a new chain through the registration module, you only need to deploy the same Pool contract if it is an EVM chain, but if it is a non-EVM chain, you need to redevelop the corresponding token pool contract and deploy it according to the protocol. After the deployment is completed, the token ID of the new chain will be written to the corresponding category in the Token Pool Manager in the Registration module to complete the registration of the new token. Subsequent additions of new tokens to the chain only need to repeat the same operation. The registration module also implements other actions that require super privileges, such as the management of prophecy machine contract addresses and interest rate strategy contract addresses.

2.3.5 Govern

Governance is one of the most important modules for the sustainability of the protocol. Through decentralized on-chain governance, the protocol can be continuously upgraded and optimized, and bring more benefits to the protocol participants. The governance module adopts the idea of DAOStack, which is co-governed on-chain through governance tokens and reputation system, and the governance scope includes the whole protocol. In the early stage, the development team will act as a proxy for governance, and after the rules for issuing governance tokens are determined, it will gradually transition to decentralized governance, and eventually be fully governed by the community.

2.3.6 Reinvest

The reinvestment module is used to improve the utilization of user funds and avoid excess idle assets by earning secondary returns for users by depositing idle funds in external protocols (e.g. Aave). The reinvestment module will update and maintain the supported external protocol yields using an off-chain prophecy machine. When a user invests in an external agreement through the agreement, the agreement will help the user to take the risk of M1 (e.g. 20%) and the acquired return agreement receives M2 (e.g. 10%). M1, M2 can be set through governance. $M1 > M2$, can be used for widely recognized agreements such as Aave, thus attracting Aave users; $M1 < M2$, can be used for certain high yield but not highly recognized agreements. When users recharge in this agreement, if the yield of external agreement is higher than this agreement and the liquidity is sufficient, the user funds and a certain percentage of the agreement funds will be deposited in the external agreement to record the user's investment status. When the user makes a withdrawal or lending in this agreement, if the liquidity is insufficient or the yield of the external agreement is lower than the agreement, the funds are retrieved from the external agreement. In order to ensure the safety of the agreement and the user's funds, each type of external agreement has a limit on the investment of the user's funds and the agreement funds to avoid excessive risk exposure.

The reinvestment module has a high degree of fit with the chain-wide ecological objectives of the agreement. The aggregation of chain-wide ecological funds is achieved through the protocol, while the reinvestment module further channels idle funds into excellent external protocols. The simplicity of the protocol to introduce new chains and new tokens can help complete the aggregation of funds quickly and efficiently. And the reinvestment module can be done by external application in addition to active discovery of quality projects (such as Aave). Therefore, the reinvestment module can be a bridge for communication between chain-wide funds and chain-wide ecological projects. For the reinvestment module, what the agreement needs to do is to reasonably balance the risks and returns of external agreements and ensure the safety of the overall operation of the agreement.

3 Risk

Protocols are risky. In the DeFi project, sufficient caution must be exercised regarding the risks present in the protocol. Compared to the first generation of DeFi lending protocols, Compound and Aave, this protocol introduces a sufficient number of functional features and flexibility. Enjoying these conveniences inevitably brings with it additional risks. The main risks of this protocol are the risk of bankruptcy due to fluctuations in asset values and the risk of messaging introduced by external cross-chain messaging protocols.

3.1 Lending Risks

3.1.1 Liquidation Permit

The agreement uses overcollateralization to hedge against bankruptcy risk from fluctuations in asset values. Like other DeFi agreements, the insolvency risk considers bad debts resulting from the decline in the value of the borrower's collateral that would render it insolvent. The difference is that the agreement also considers insolvency risk due to an increase in the value of the borrower's liabilities. To reduce the systematic insolvency risk, the protocol introduces a collateral decline risk factor CF and a liability escalation risk factor BF. The value of the user's assets must meet the following conditions or they will be liquidated.

$$TotalCollateral * CF * BF >= TotalDebt$$

3.1.2 Asset Layer

In the first generation of DeFi lending protocols (e.g. Aave), users are provided with the ability to lend on a handful of the most liquid ERC20 tokens. These protocols relied on a licensed listing system to protect users from the risk of volatile assets. At the same time resulting in a large unmet demand for lending and borrowing of volatile assets remains. To address the unlicensed problem of listing new assets, current solutions are isolated liquidity pairs and isolated liquidity pools.

Isolated liquidity pairs are, as the name suggests, isolated super-collateral pairs: users can only receive one specific collateral to another asset at a time, while users need to provide liquidity for different isolated pairs separately, creating extreme liquidity segregation.

Isolated liquidity pools are the solution used in this protocol. By layering assets, segregation of liquidity is achieved while isolating risk. This is a solution that trades off between isolated liquidity pairs and a single liquidity pool. Asset tiers range from 2 to 4, and different tiers can have unique designs to meet risk control needs. Currently, assets are divided into two tiers: ordinary assets and high-risk assets. The common asset liquidity pool is the cornerstone of the protocol and targets the tokens with the best traditional liquidity. Ensuring the liquidity of the common asset pool can effectively improve the protocol's ability to resist risk. The high-risk asset pool targets assets with high volatility, and controls the maximum risk of the high-risk asset pool through debt caps. Key characteristics of tiered assets.

Ordinary assets: Ordinary assets can be used for both collateral and lending without any restrictions. When bad debts occur due to the downside of collateral prices and the upside of liability prices, the reserves kept in the treasury will be used to pay out. Ordinary assets perform more consistently for the most part, with fewer instances of bad debts. The accumulation of reserves far outpaces the accumulation of bad debts, keeping the agreement up and running and continuously improving its resilience to risk.

High Risk Assets: High risk assets with unpredictable volatility. Such assets are prone to spikes and drops as collateral or liabilities, leading to bad debts. To meet the availability of high-risk assets while reducing systemic risk.

- The risk of high-risk assets, whether as collateral or liabilities, is borne by both the borrower and the lender. In case of bad debts, no payout will be made using the treasury reserve. This is determined by the weak risk tolerance at the beginning of the agreement.
- High risk assets as collateral or liabilities must be segregated from risk with separate sub-accounts.
- Higher risk assets have higher borrowing and lending rates, offsetting the risk taken by the user.
- The high-risk assets are used as collateral and the liabilities must be in a stable currency. The liability stablecoin has a lending cap to control the total downside risk of the high-risk assets from exceeding the lending cap.
- High-risk assets are treated as liabilities, and the collateral must be stablecoin. The collateral stablecoin has a deposit cap, and the total upside risk of the controlled high-risk assets will not exceed the deposit cap.

3.2 Cross-Chain Messaging Protocol

The protocol makes use of external cross-chain messaging protocols for message delivery. Cross-chain messaging protocols are currently the hardest hit by money theft. However, the protocol has high requirements for the security of cross-chain messaging protocols. In order to meet the requirements of the protocol, the cross-chain messaging protocols that are currently well recognized (e.g. LayerZero) are first selected. Also, in order to achieve risk diversification, multiple cross-chain messaging protocols can be relied on. The security of the message transmission is ensured by confirming the correctness of the message several times.

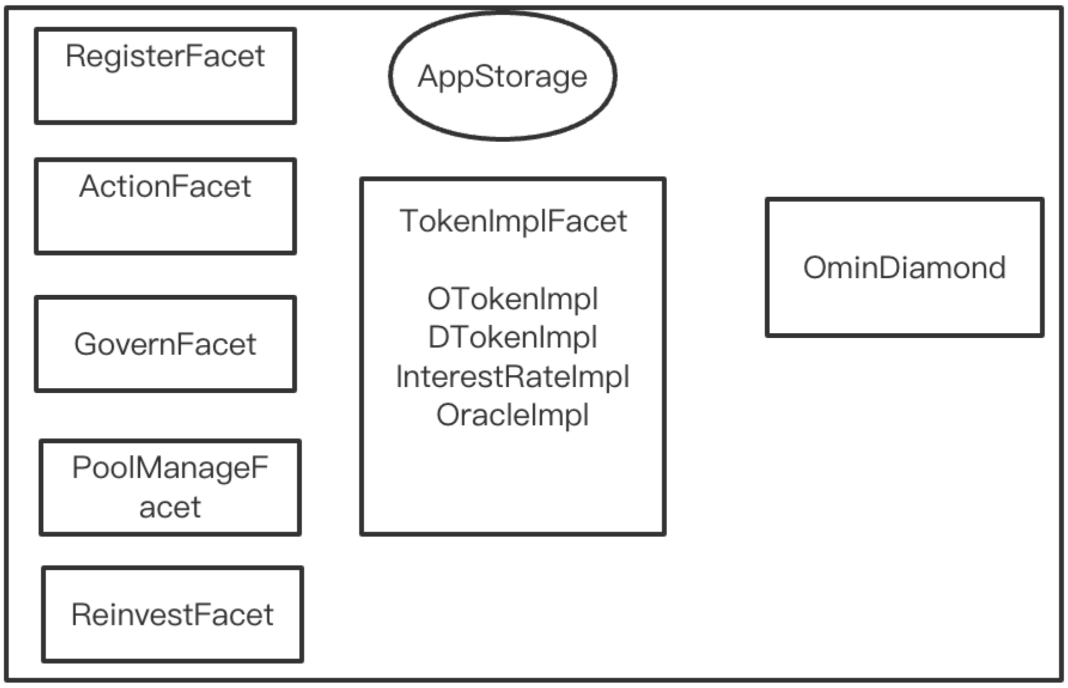
3.3 Liquidity risk

Compared to the usual risk of liquidity pool depletion, full chain liquidity depletion is less likely, but single chain liquidity depletion is highly likely. Single-chain liquidity depletion will result in users being unable to withdraw and borrow assets on that chain, although users can still withdraw and borrow assets from other chains, but in general not in line with user expectations. The protocol adjusts the liquidity between different chains through the expectation distribution. When the liquidity of that chain is less than the protocol's desired distribution, it discourages users from further liquidity depletion and encourages users to increase their liquidity, thus eliminating single-chain liquidity risk.

4 Protocol Contract

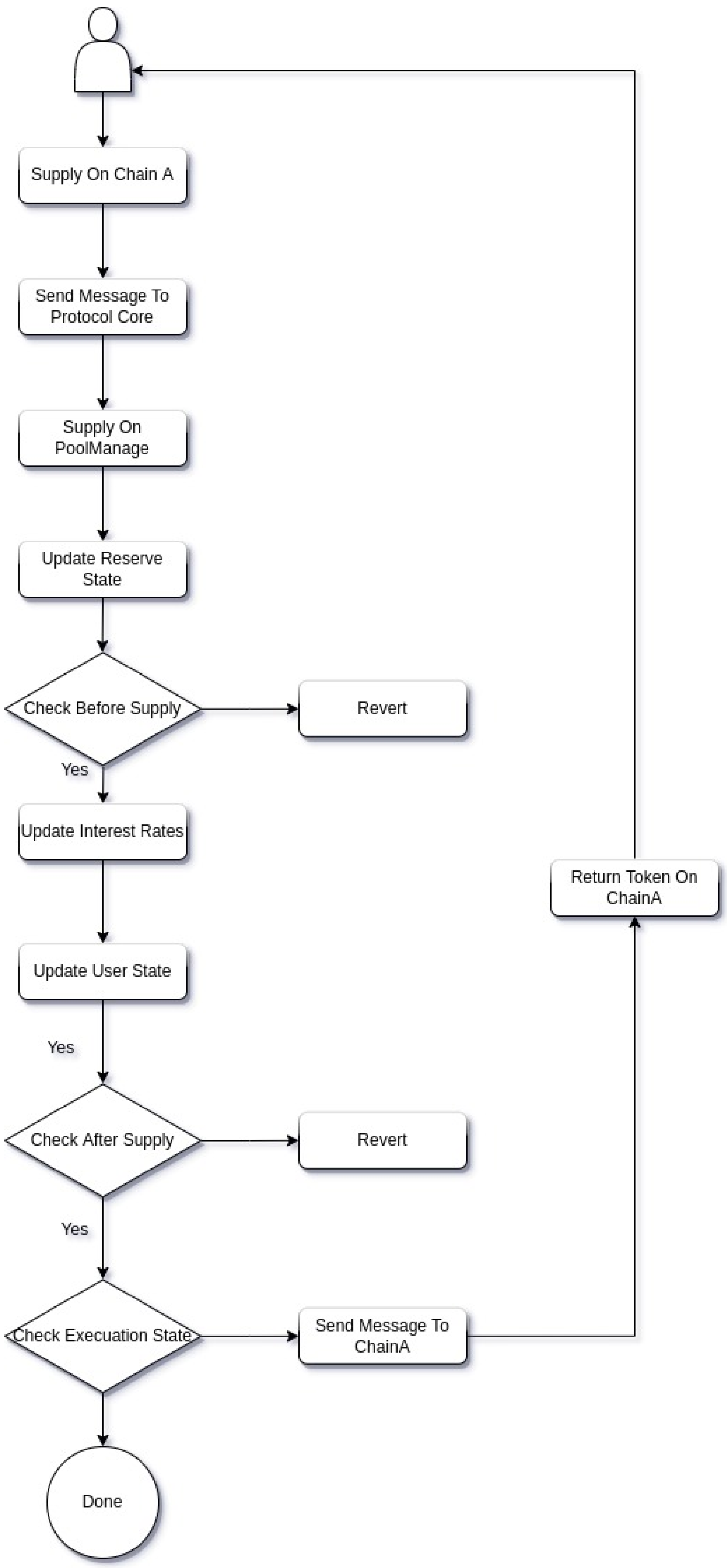
The protocol core uses EIP2535 protocol to build the contract code, all data is stored in AppStorage, and all interactions are realized through Facet. User operations are performed through the OmniDiamond proxy contract to make delegateCall calls to the protocol core. The benefits of adopting the following architecture for the protocol contract are: 1) storage and interaction behavior are completely separated; 2) the interaction behavior contract is very easy to upgrade; 3) the contract upgrade will not have any impact on the front-end application through the OmniDiamond proxy contract unified call.

Code Architecture

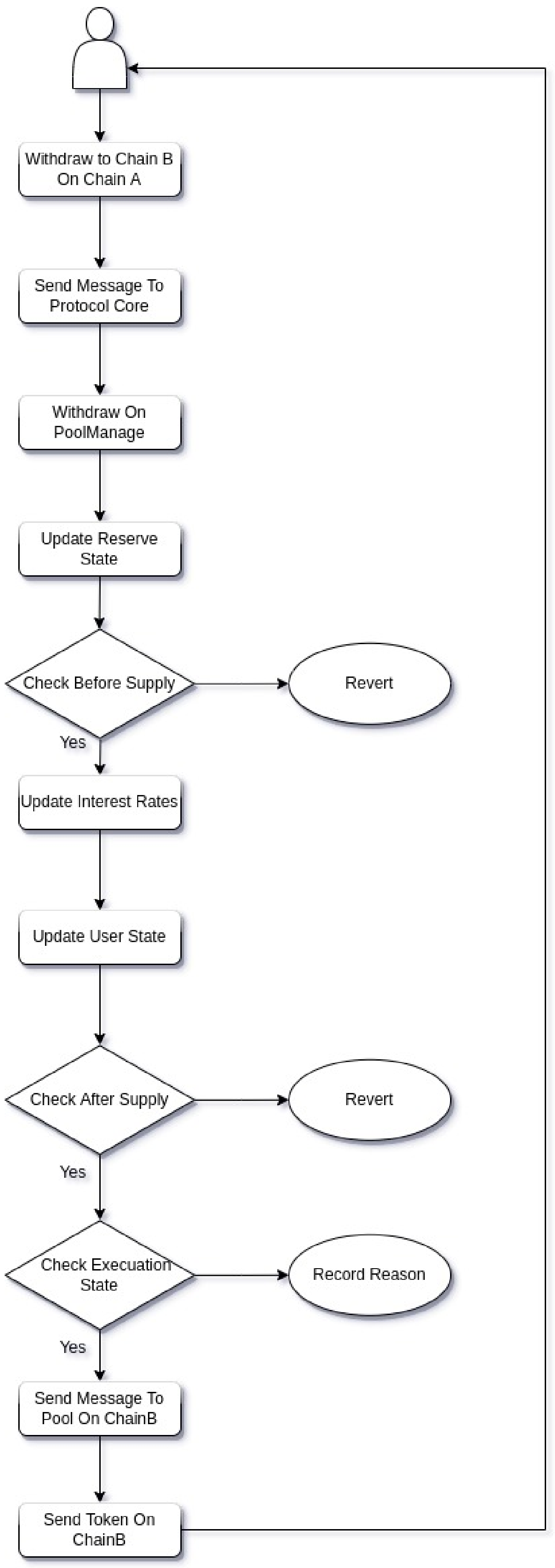


5 Interaction

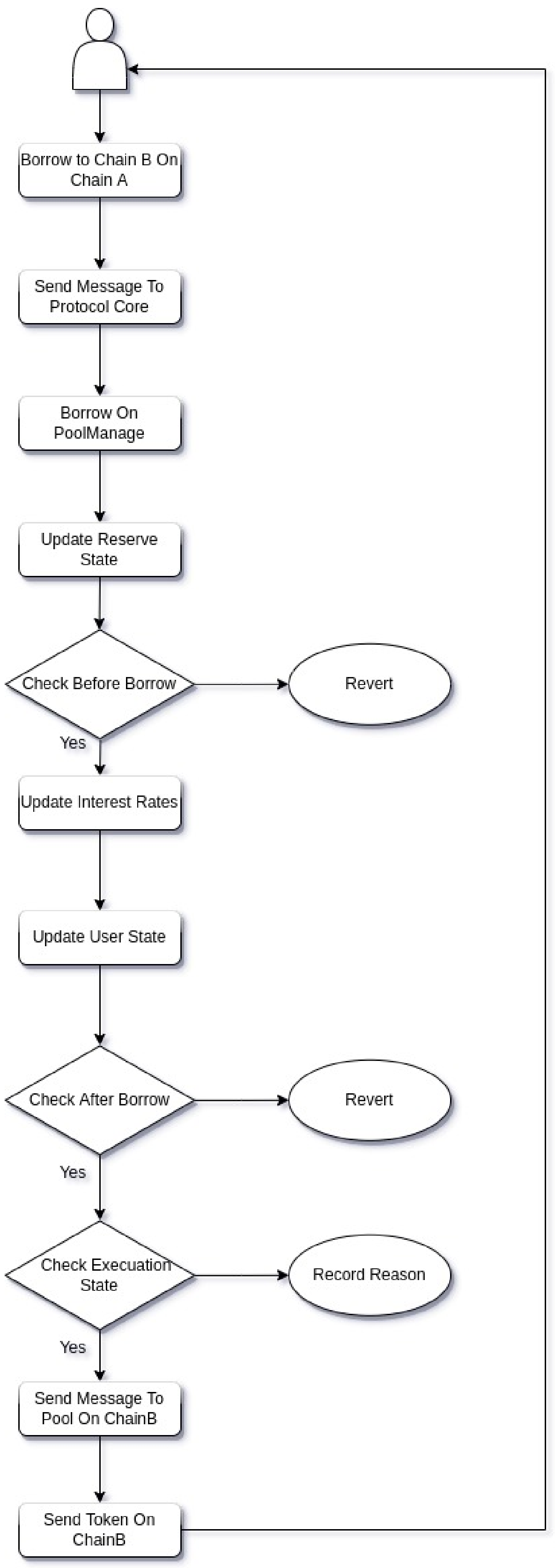
5.1 Deposit


Supply On Chain A
Send Message To Protocol Core
Supply On PoolManage
Update Reserve State
Check Before Supply
Revert
Yes
Update Interest Rates
Return Token On ChainA
Update User State
Yes
Check After Supply
Revert
Yes
Check Execuation State
Send Message To ChainA
Done

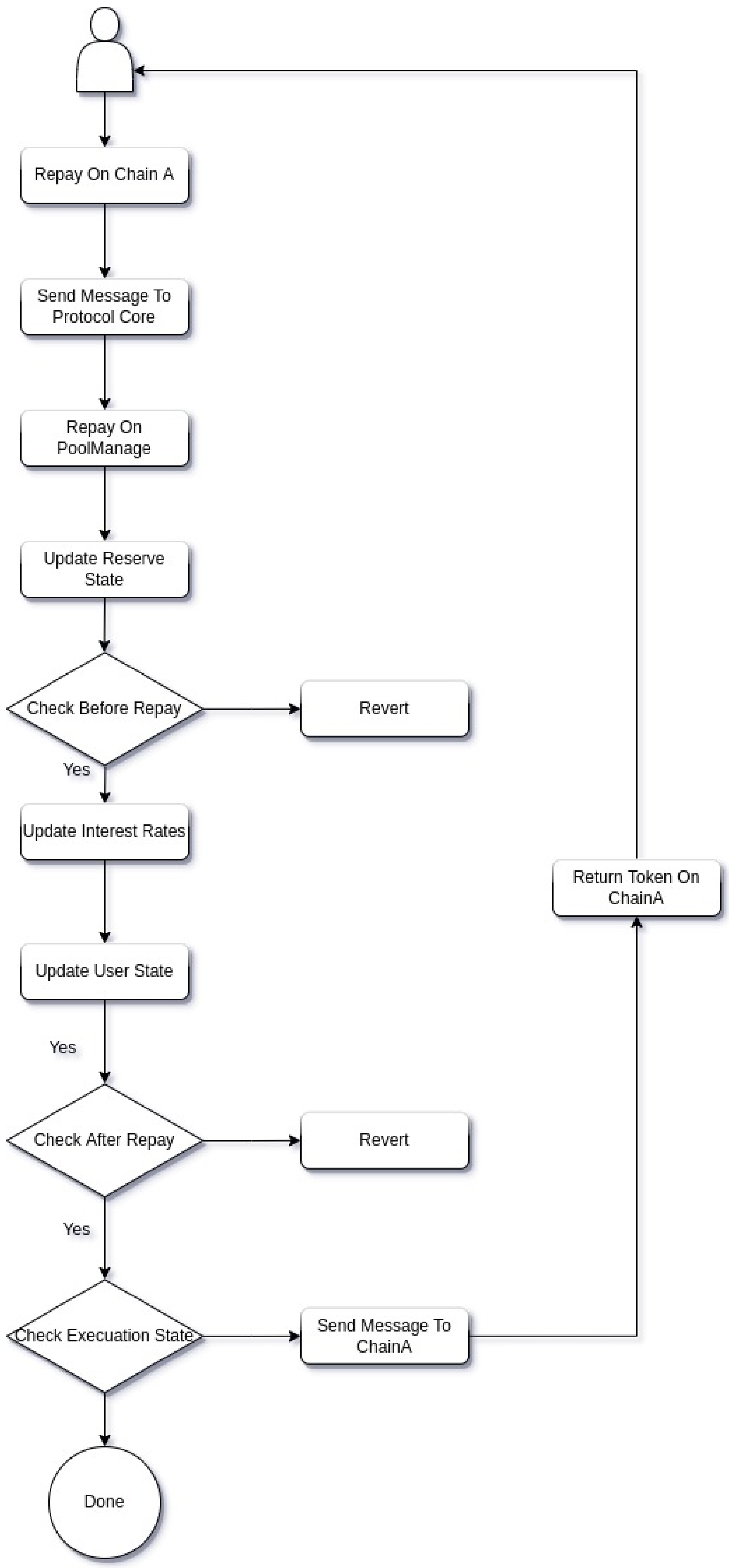
5.2 Withdraw


Withdraw to Chain B On Chain A
Send Message To Protocol Core
Withdraw On PoolManage
Update Reserve State
Check Before Supply
Revert
Yes
Update Interest Rates
Update User State
Check After Supply
Revert
Yes
Check Execuation State
Record Reason
Yes
Send Message To Pool On ChainB
Send Token On ChainB

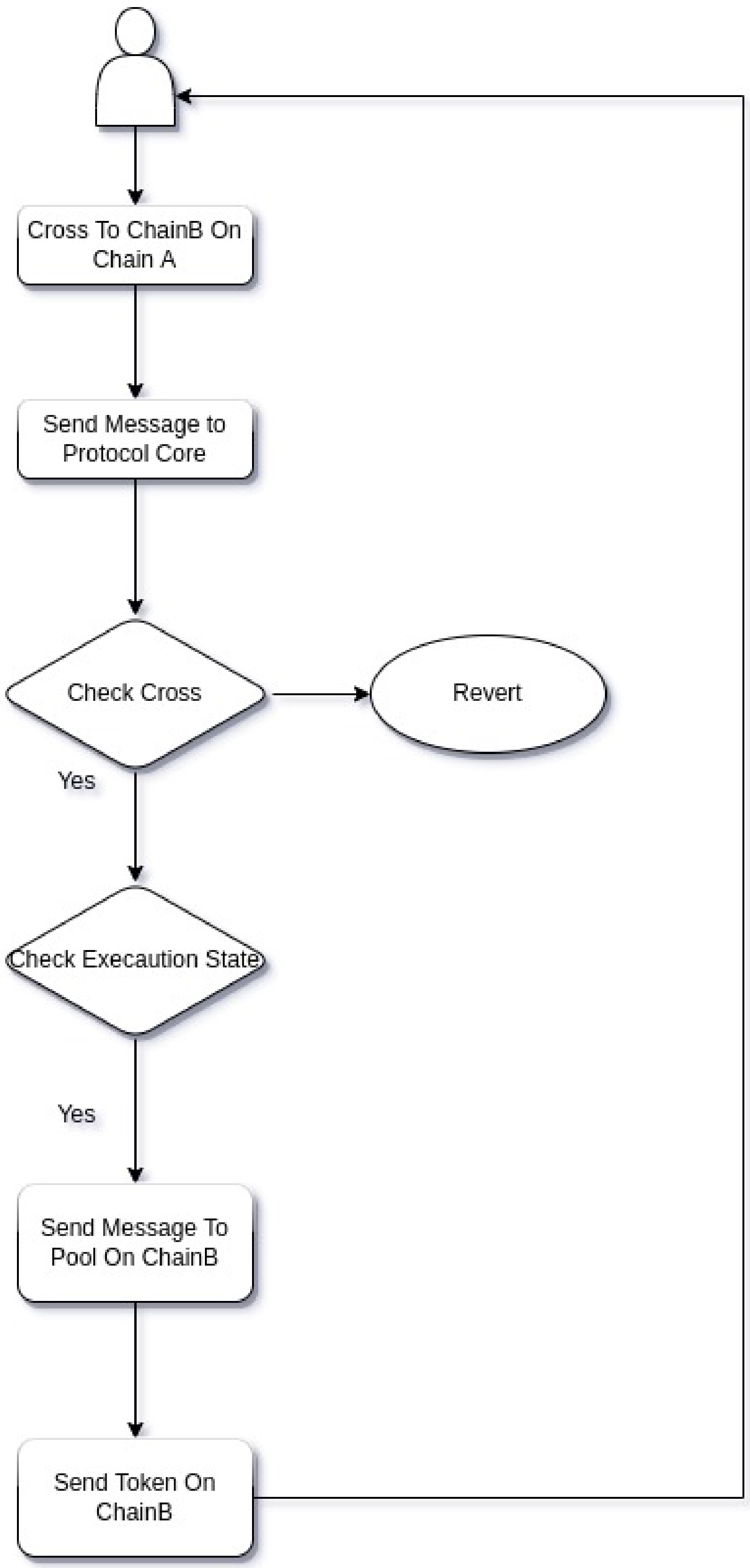
5.3 Borrow


Borrow to Chain B On Chain A
Send Message To Protocol Core
Borrow On PoolManage
Update Reserve State
Check Before Borrow
Revert
Yes
Update Interest Rates
Update User State
Check After Borrow
Revert
Yes
Check Execuation State
Record Reason
Yes
Send Message To Pool On ChainB
Send Token On ChainB

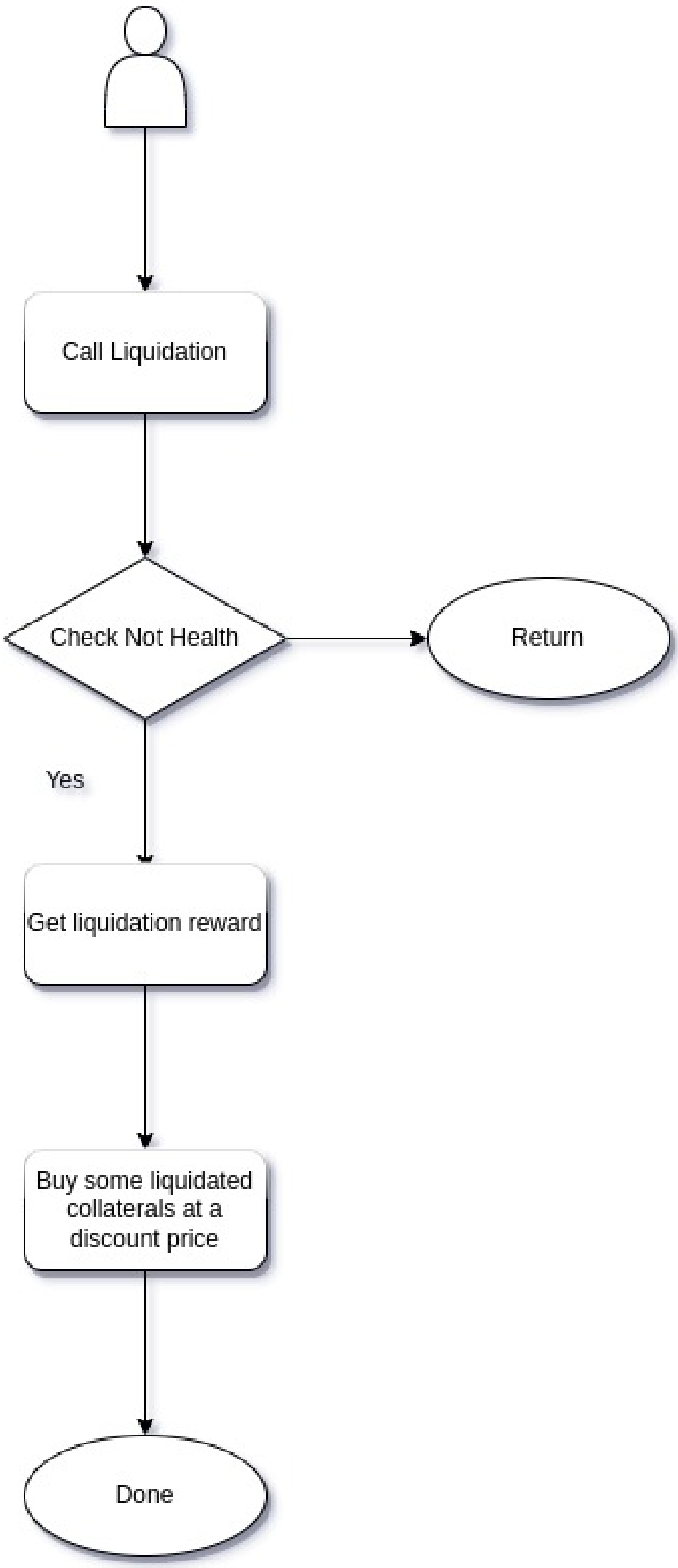
5.4 Repay


Repay On Chain A
Send Message To Protocol Core
Repay On PoolManage
Update Reserve State
Check Before Repay
Revert
Yes
Update Interest Rates
Return Token On ChainA
Update User State
Yes
Check After Repay
Revert
Yes
Check Execuation State
Send Message To ChainA
Done

5.5 Cross


Cross To ChainB On Chain A
Send Message to Protocol Core
Check Cross
Revert
Yes
Check Execaution State
Yes
Send Message To Pool On ChainB
Send Token On ChainB

5.6 Liquidate


Call Liquidation
Check Not Health
Return
Yes
Get liquidation reward
Buy some liquidated collaterals at a discount price
Done

6 Outlook

The protocol is designed to enable chain-wide lending and exchange of tokens, while there are inevitably some areas where improvements can be made. Risk control is a part of the development process and the future that will always need attention and improvement. The protocol should be as risk-averse as possible, controlling the maximum risk size and pre-defined risk remediation options. Protocols are full-chain ecological and can do some further interesting cutting-edge work. The horizontal expansion of token's chain-wide lending and exchange to NFT market; the vertical mining of further collateralization and application of the equityized tokens are the directions that can be explored in the future.

7 Conclusion

The protocol borrows some of the best designs from Aave, while incorporating token pools and cross-chain messaging protocols to form a new full-chain lending protocol. The full-chain lending protocol is a cutting-edge path compared to the current DeFi application by allowing lending and cross-chain operations through a full-chain liquidity pool. This is the first step in the full-chain era and lays a good foundation for the development of the full-chain era. The full-chain lending protocol unifies the liquidity pool through just the right design to provide chain-wide operational convenience for lending and borrowing DeFi applications, solving the current dilemma of silos between chains.