Free TON DEX Architecture proposal

Dates: 1 November – 15 November 2020

General description

The goal is to describe possible architecture, design approaches, and technological stack for implementation of decentralized exchange (DEX) on top of Free TON blockchain and infrastructure.

Submission by Dmitry @dnugget and Yaroslav @yanmsk, Radiance Team.

Backlink on forum:
https://forum.freeton.org/t/contest-proposal-freeton-dex-architecture-design-proposal/3067/7

Our team approach involves the following sections:

- elucidating the requirements,
- going through existing AMM and legacy Decentralised exchange approaches and their limitations,
- solutions to limitations and proposed Architecture from a smart contract and Application perspective.

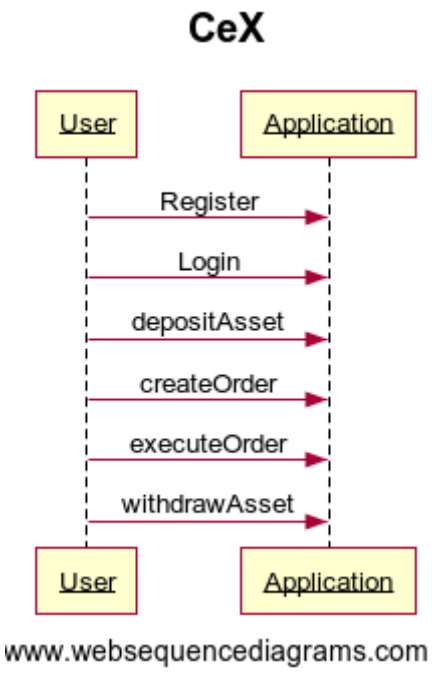
1. Requirements

The original contest overview and requirements is well thought out and covers the range of requirements of the submission.

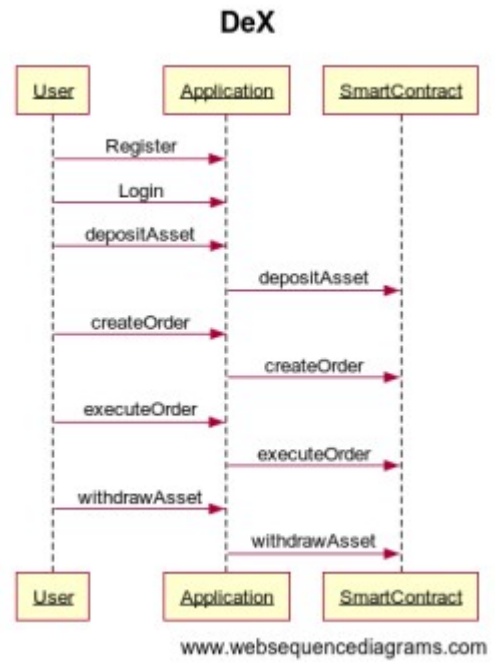
2. Crypto Exchanges

For sake of completion we would start with centralised exchanges, then describe the aspects of decentralized exchanges and then cover automated market makers and their specific attributes, advantages and limitations.

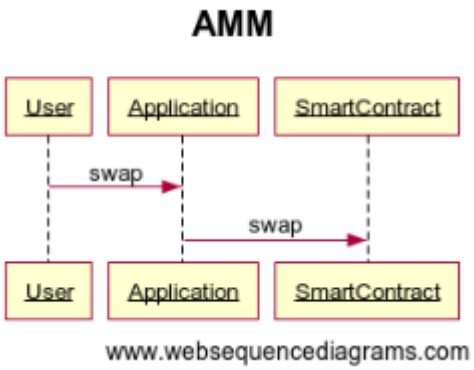
Centralised Exchanges:
https://www.binance.com/en



Decentralised Exchanges
https://idex.io/



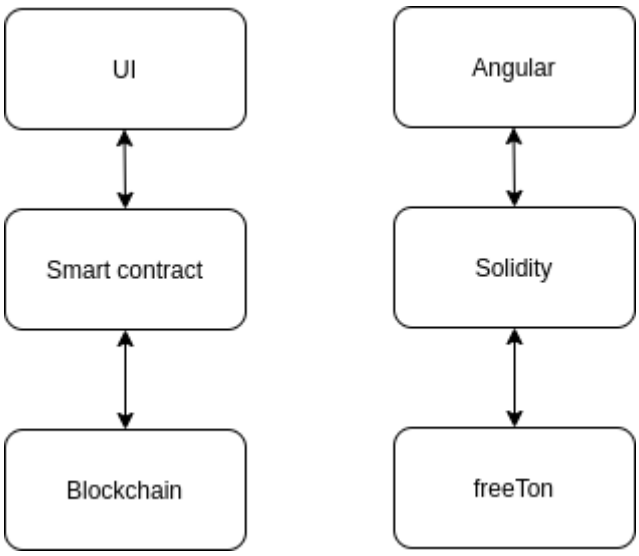
Automaticed Market Makers



3. Proposed Architecture

Free TON supports solidity as a smart contract language, paired with asynchronous behaviour and great performance perspectives which makes a solid base for DEX.

A simpleton illustration of the proposed architecture is illustrated below;



The proposed solution involves the following layers:

a. UI : a simple, intuitive user interface encompassing required exchange features covering one page. Think uniswap or google.

b. Smart contract : The evolution of crypto exchanges and increased decentralisation has to do with the development of solidity as a programming language and corresponding tools. Solidity is an obvious choice of smart contract programming language.

c. Blockchain: Free TON.

4.1 Automated Market Makers and their features

Unlike traditional centralised and decentralised exchanges, DEX with swaps are automated market makers with the following features:

1. No registration or login required for users. DeFi based DEX's follow an user experience where, as long as the user is logged in with a supported wallet (for example metamask or any web3 wallet in ethereum), the user is capable of utilizing all the features of the platform.

2. Price of an asset in the pool (asset/asset pair) is calculated based on predefined rules. Different strategies are used as predefined rules. Constant product market maker model is an example used in uniswap, wherein the arithmetic product of quantities of each asset in the pool is kept at a constant. Taking uniswap's model in to account, assume USDT-ETH pool, if the quantity of USDT tokens are represented as 'a' and the quantity of ETH coins are represented as 'b', a * b = k, will remain a constant throughout the lifetime of the pool. The value a or b can move inversely proportional each other with any change in the liquidity of the pool. Some of the advantages and disadvantages of the constant product market will be discussed below in section existing challenges of the DEX. Different protocols employ different strategies to achieve automated market maker mechanisms.

3. Liquidity providers need to satisfy both the demand and the supply side in accordance with the existing pool ratio, thereby ensuring price remains the same irrespective of liquidity provider actions.

4. Traders can't manipulate the price with orderbooks, as there are no order books. In traditional centralised exchanges and decentralised exchanges, order book and candlestick charts and bid/ask curve are an essential part of the exchange along with order creation. An exchange by way of bots, or traders by way of placing trades to their advantage can mislead the market on both bid and ask sides. A common strategy being placing a lot of sell orders to make the asset in a dump cycle, or a perceived dump cycle. With no order books, and instant trades, any and all limitations of placing orders and order books are entirely removed in DeFi DEX's.

4.2 User Flow

Stakeholders of the DeFi DEX can be categorized into the following four broad categories:

Investors/Developers/Team:
The protocol or the product is usually associated with a known set of team members who are either investors and backers or the builders and developers of the product and protocol.
Uniswap started as a learning exercise of the founder and has grown to be the gold standard of AMM based DEX. As with conventional startup's, uniswap raised capital from venture capital with multiple investment rounds. Sushiswap has anonymous developers with no apparent investors. Multiple food meme'd DEX are anonymous products.

Traders:
Traders are the users of the product and the protocol. Any user who has a need to trade/swap any available asset to an asset available in the pool for a fixed and predetermined target asset in the pool are classified as traders. Traders are the primary revenue stream of the protocol and product.

Liquidity Providers:
Users who provide liquidity to the available pools and also create their own pools are liquidity providers. Liquidity providers enable liquidity and stability to the protocol. Without sufficient liquidity, a pool may suffer consequences like one sided liquidity. Liquidity providers are benefited by a share of the transaction fee accrued for all trades of the protocol for their contribution to the market.

Token Holders:
A native governance token is provided to the team and also liquidity providers and users who stake their liquidity provider tokens. A token economics of the native token and different approaches are discussed in the Economic model of the DEX. As a rule of thumb, these token holders are provided with a share of the profit of the product, thus giving value proposition the underlying token. The token holders are also given the ability to create proposals and vote for change proposals of the protocol, thereby enabling decentralised governance with full respect to Free TON.

Users	Functions	Features
Team	Develop the product	Part of the profit as tokens Governance activity
Traders	Trade/Swap assets	
Liquidity Providers	Add Pair/pool Add liquidity Remove liquidity	Proportionate Transaction feed Yield farming
Token Holders	Governance activity	Entitled for a part of the profit Create proposal Vote on a proposal

Liquidity providers actions:

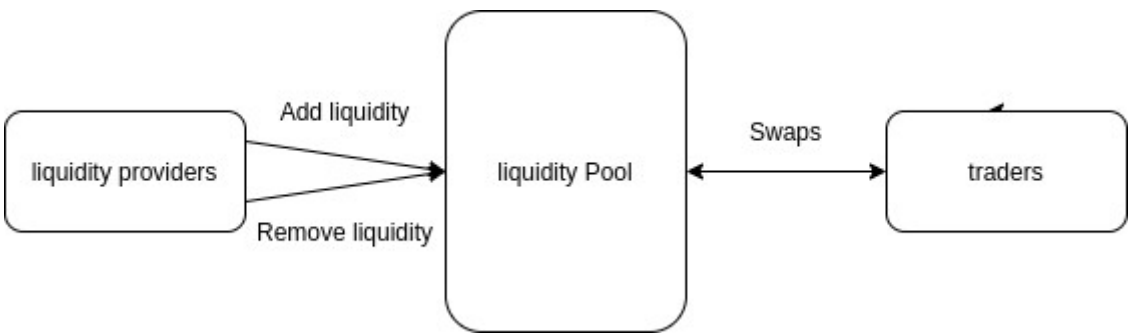
a. Add Pair/Pool
Any user can create a pool or pair by providing both supply and demand. Assuming ETH is valued at 400$ and USDT is valued at 1$; any user willing to create the USDT-ETH must provide 1ETH and 400USDT tokens, setting the initial value of USDT at $1. However, the DEX is oblivious to any price action or price discovery. If the user creates a pool with say 1ETH and 800USDT tokens, then the initial value of USDT is set at $0.50. This will enable arbitrageurs to buy USDT at a lower rate in the pool and continue to do so till the price of USDT reaches $1. If on the other hand, the pool is created with 1ETH and 200USDT tokens, the initial value of USDT is set at $2, this would enable arbitrageurs to sell USDT at a higher rate in the pool and continue to do so till the price of USDT reaches $1. Hence price parity is reached by the market participants and protocol or the team doesn't have any say in the market action, price action and reaching price parity.

b. Add liquidity
Any user can add additional liquidity to an existing liquidity pool at the current pool ratio and is entitled to be rewarded based on the user's share of provided liquidity for all the trading activity of the pool.

c. Remove liquidity
An user can remove their liquidity from the pool any point in time and will be removing the share of the transaction fees along with the provided liquidity.

All the above actions are performed without any registration and are handled in a decentralised manner by way of direct interaction with the smart contract. It is pertinent to note, these smart contracts are publicly verified and are audited to ensure the safety of user assets and the intentions of the developers.

Trader actions
A trader can swap any asset in any available pool as long as the user has the necessary funds in terms of the required asset and gas fees. As with the liquidity providers, the interaction is with the smart contract directly and is done in a decentralised manner.



Token holder actions
A native governance token is the part of the economic and governance model of the most Decentralised Finance system.

a. Entitled for a part of the profit
Profit in the liquidity pools are collected from the traders at a set fixed rate; 0.3% is the standard with Uniswap and the other clones. The 0.3% is provided to the liquidity providers in full in the current version of Uniswap and there is a proposal to split the 0.3% in to 0.25% for liquidity providers and 0.05% for the developers. This will be achieved by community voting. In Sushiswap, 10% of all minted SUSHI tokens are set aside for developers.

b. Create proposal
MakerDAO's decentralised governance is adopted across the spectrum in DeFi use cases. The holders of the native token are enabled to create a proposal for voting. This proposal can be accepted or rejected by the community with a stipulated time frame.

c. Vote on a proposal
Token holders are entrusted with the capability to vote on existing proposals. The voting is on-chain and can be verified publicly in a trust-less manner. This behavour is fully compliant with Free TON practice.

4.3 UI, Interface and Smart contracts

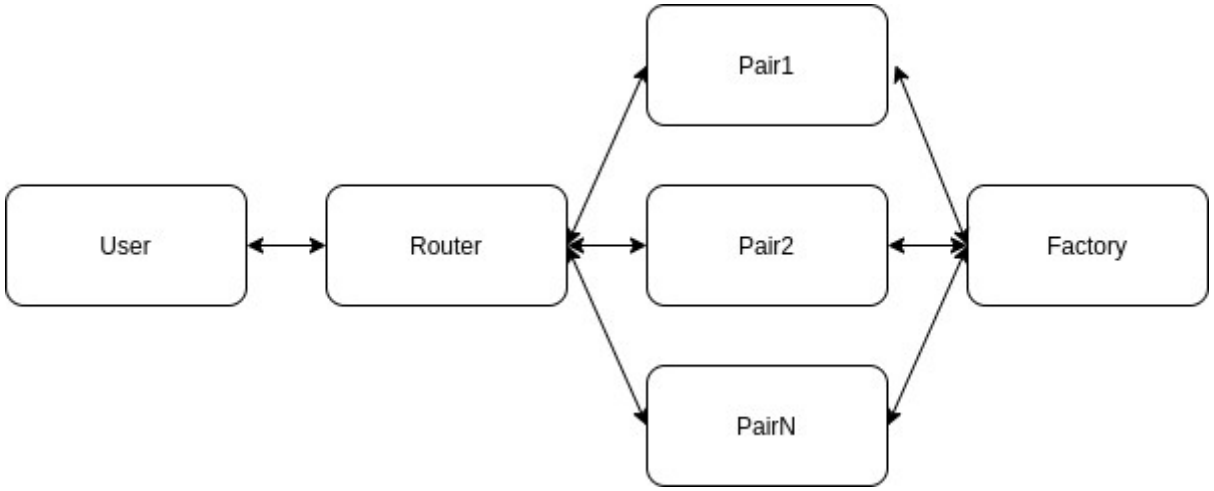
4.3.1 Choice of User Interface:
User interfaces of decentralised exchanges are sleek and intuitive. A simple one page user experience is provided for swap and pool related functions. React is the choice of framework for Uniswap and it augurs well with the modern UI/UX requirements of simple and intuitive characteristics. All clones of Uniswap on ethereum and side chains use React. Some of the adoptions on Binance smart chain also use the original Uniswap design for consistency and familiarity among Decentralized Finance based DEX users.

4.3.2 Interfaces
Web3 enables interfacing web applications with blockchain nodes and dApp Server via TON SDK Client library. There are many wrappers for popular programming languages exist after Free TON SDK Bindings contest.

4.3.3 Smart contracts
Solidity is the choice of all ethereum, binance smart chain and tron based Decentralised exchanges. Solidity offers a battle tested smart contract language with a future smart contracts debugger solution for the developers on Free TON. Solidity has the advantage of

being the choice of smart contract development language of a lot of successful decentralised applications. Given, solidity syntax and language semantics are similar to javascript, it gives greater ability for programmers to learn and adapt.

4.3.3.1 Smart contract architecture
Similar to Uniswap, any decentralised exchange will have a set of smart contracts for all functions, like swap, add pool, add and remove liquidity and any token incentive associated with the product. Uniswap differentiates its smart contracts into core smart contracts and periphery smart contracts, which enables the design to be modular and scalable.



4.3.3.2 Core Smart Contracts
Factory and pair contracts form the core components of the Dex architecture. Factory smart contract creates and indexes all the pairs in the automated market maker. The pair contracts enable swap and liquidity features of a particular pair and enables price discovery for decentralised oracles.

UniswapV2Factory

https://etherscan.io/address/0x5c69bee701ef814a2b6a3edd4b1652cb9cc5aa6f#code

Diagram (see Annexure I): uniswapFactory.svg

UniswapV2Router02

https://etherscan.io/address/0x7a250d5630b4cf539739df2c5dacb4c659f2488d#code

Diagram (see Annexure II): uniswapRouter.svg

UniswapV2Pair

https://etherscan.io/address/0x0d4a11d5eeaac28ec3f61d100daf4d40471f1852#code

Diagram (see Annexure III): uniswapPair.svg

4.3.3.3 Periphery Smart Contracts
These smart contracts are responsible for feature specific functions of a swap product.

4.4 Economic Model of DeX

The economic benefit and incentives of the stakeholders of the platform are explained in this section.

As discussed in the previous section, there are four broad categories of the users:

a. Team
b. Traders
c. Liquidity Providers
d. Token Holders

The economic motivation and incentive of the above categories are discussed in section 4.2. In this section, we will through how transaction fees, yield farming and broader token economics of a decentralised exchange can be desgined.

A native token represents the value proposition of the protocol. The token is provided with additional governance features; which will be discussed in section 4.6 Governance Model of DeX.

4.4.1 Transaction fees

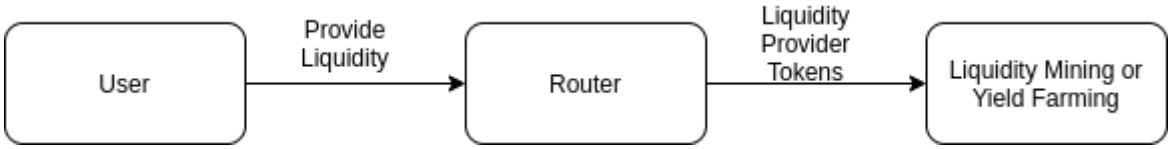
Uniswap charges 0.3% transaction fees for all trades and are divided proportionately to the liquidity providers of the respective pools. There is a proposal to split the 0.3% in to 0.25% for the liquidity providers and 0.05% for the team. The code to split the transaction fees is available in the smart contracts and can be enabled by a flag. The proposal has to get community approval.

In Sushiswap, there is an identical 0.3% transaction fees; and 0.25% is provided as incentive to the liquidity providers and 0.05% is shared among SUSHI (native token of sushiswap) holders. Sushiswap also rewards the developers by minting 10% tokens for the developers.

A Transaction fee of 0.03% and a split of 0.25% to the liquidity providers and 0.05% to the team or the wider token holders is reasonable and in line with the industry practices.

4.4.2 Yield Farming

Yield farming or liquidity mining provides rewards and incentives to the liquidity providers of the protocol besides the transaction fees. Annual percentage yield provides a measure of the yield per pool for all the platforms and is a good measure to market and build up interest in the platform.



The following link provides a list of pools and protocols and their yield farming numbers, https://www.coingecko.com/en/yield-farming

4.4.3 Token Economics

A native token provides a number of advantages to the protocol in terms of benchmarking market capitalization, community building, decentralised governance and raising investments. Design aspects to consider:

a. Total supply of the token
b. Initial circulating supply of the token
c. Share of premine
d. Emission of token as rewards for yield farming and other activities.

4.5 Non custody of Assets and liquidity

Decentralised Finance implies inherently there is no custody of either the assets or their liquidity or the rewards. Custodial services and models are prone to be centralised and are vulnerable to security threats. On the other hand, non custodial approaches with smart contracts enable decentralised and fair mechanism of handing assets and position itself as a credible alternative.

4.6 Governance Model of DeX

A portal page should be enabled to allow the token holders to create proposals and vote for and against the proposal. Creation of proposal and voting should be on-chain. Design considerations are the following:

a. A minimum amount of tokens for the token holders to create a proposal
b. A timeline for each proposal
c. SMV based voting

5. Requirement Tracability

The following section provides the overall requirements and the sections covering these requirements.

5.1 General requirements

a.The basic economic model and description of money flow in the system
[Comment] Refer Section 4.2 User Flow and 4.4 Economic Model of DeX

b. The general technical architecture of the solution including all of the features listed below in the hard evaluation criteria section along with the proposed customer journeys
[Comment] Refer Section 4.1 Automated Market Makers and their featuers and 4.2 User Flow

c. Detailed technical specification of the proposed implementation with the justification of the selected approach: smart contracts, integration layer, interfaces
[Comment] Refer Section 4.3 UI, Interface and Smart contracts

5.2 Hard requirements

a. Support for non-custody exchange of any tokens within the FreeTON network
[Comment] Refer Section 4.5 Non custody of Assets and liquidity

b. Support for adding liquidity to the exchange from external blockchains at least through smart contract system or using atomic swap bridges described in atomic swap contest
[Comment] Considering atomic swaps between Free TON and Ethereum virtual machine based chains are achieved through bridge contracts; wherein assets are locked in the parent chain and are deposited in the child or target chain and swaps and other trades are executed in the target chain and corresponding asset are withdrawn back to the root chain and unlocked in the smart contract. The same approach could be made available with a set of smart contracts for bridge architecture.

c. Non-custody approach to both exchange and liquidity provision
[Comment] Refer Section 4.5 Non custody of Assets and liquidity

d. Scalability potential for adding new tokens
[Comment] Refer Section 4.3 UI, Interface and Smart contracts. Given the Uniswap approach of a factory desing pattern to create new pairs and index them seemlessly, adding new tokens and pairs wont be a challenge.

e. Maximum utilization of the Free TON network advantages such as speed and sharding
[Comment] One of the disadvantages of ethereum in its current form is the use of energy intensive proof of work consensus algorithms which pose a major scaling challenge. Free TON overcomes the scalability challenges using by byzantine fault tolerance based proof of stake. Adding sharding enables increase in scalability by orders of magnitude. Ethereum also suffers from front running because of low network throughput; which is overcome in Free TON by use of proof of stake consensus.
Reference: https://docs.ton.dev/86757ecb2/p/07ddda-walk-through-the-catchain

f. Optional governance mechanism or a possibility of adding such a mechanism in the future to govern the parameters of the exchange, including but not limited to the liquidity provision and the fees
[Comment] Refer Section 4.6 Governance Model of DeX

g. A good economic model for exchange, liquidity makers (providers), and liquidity takers
[Comment] Refer Sections 4.1 Automated Market Makers and their featuers, 4.2 User Flow and 4.4 Economic Model of DeX

5.3 Soft requirements

a. Your position on existing DEX problems (see “Existing DEX problems to be aware of” section below) and how they are tackled in the proposed architecture
[Comment] Discussed in Section 5.4 Existing DEX problems

b. Detailed and easily understandable charts explaining the architecture and business processes
[Comment] Section 4. Proposed Architectue gives a holistic view of the proposed architecture from business, user and a technical stand point

c. Brevity

d. Readiness to participate in the implementation of the solution in the next stage

5.4 Existing DEX problems to be aware of

1. Frontrunning (flash boys 2.0 and Mooniswap vs Uniswap value proposition). If it is not possible in Free TON (as it is), should be clearly stated why.

[Comment] Front running is the strategy of using either private data or publicly available data to take undue advantage of existing trades. Assume, there is a buy order of a token at $x and if there are sell orders of the same token at value greater than $x, then a frontrunner could execute the buy order at $x and sell the same tokens for the existing sell orders at value greater than $x. This could be made possible by sequencing the calls in such a way that front running is made possible. Given, the gas costs and pending transactions of ethereum, all orders placed are publicly available. And these publicly available orders can be manipulated by increasing gas prices of front running orders.

Free TON's use of Proof of stake as a consensus enables much greater scalability and avoids problems of excessive gas costs and pending transactions. As long as, the transactions are executed instantly and not a long list of pending transactions, front running could be avoided.

2. Shortcomings of standard curves and its inflexible nature in liquidity based approach. These result in the issue - dilution of liquidity into several AMMs (general and specific ones with a more specific curve like Uniswap vs. Curve) which leads to non-optimal price execution.
[Comment] Uniswap uses an invariant for constant product maker. The invariant suffers from price fluctuations when the liquidity is not large enough and during the initial days of

the pool. Curve on the other hand proposes a model to ensure the price fluctuations are kept at a minimum for stable tokens. It is pertinent to note, stable tokens are dependent on market actions also for their value and such fluctuations cause underlying issues for the stability aspect of the coin/token. An approach could be to use a hybrid model, where, uniswap based invariants are used for eth and tokens.

3. One-sided liquidity. Impairment loss problems. The market risk of two assets in the pool is widely discussed. These problems should be somehow covered or at least mentioned for further research.
[Comment] Impermanent loss in providing liquidity can be defined as the loss of assets while providing liquidity against holding the same asset. Impermanent loss occurs when there is a role for arbitrageurs on account of price fluctuation of one of the assets in the pool. Impermanent loss is temporary at any given price point, and becomes permanent once the liquidity is removed.

Reference: https://uniswap.org/docs/v2/advanced-topics/understanding-returns/

Curve.fi and balancer.exchange provides approaches to minimize impermanent loss by way of stable invariants in constant product market maker and weighted average market maker respectively. The issue of impermanent loss can be overcome by using a hybrid approach.

4. Another liquidity pool-based approach problem is the customized proportion of assets, number of assets in the pool, and metapools. How to regulate the number of assets in the pool, what is the price optimal routing, will it be available to create metapools.
[Comment] Metapools are liquidity pools where a single token is pooled in with all the other tokens of the standard pool without diluting the underlying pool liquidity. Metapools enable exposure to assets in the underlying pools and without diluting the liqudity. Design aspects to consider in metapools are the number of pools to be associated when creating a metapool.

5. Orderbook-based exchanges should state clearly how the on-chain order book is maintained and how its scalability issue is resolved. It is naive to state that transactional expenditures would stay at an effective zero level, so it has to be covered.
[Comment] The nature of automated market makers is to execute orders at market price and to go away of order book entirely. However, stop and limit orders are integral to market dynamics of crypto exchanges and industry at large. To design decentralised exchanges with orderbook and to leverage automated marker makers involves on-chain storage of orders. The approach should be to make the users incur the cost of creating orders and cancelling orders and on-chain execution of the orders at the prerequisite price points.

6. Economic Hacks

Economic hack involves finding arbitrage opportunities or creating one by way of a combination of flash loans and price fluctuations in pools across multiple protocols. The following links provide insights in to the three most recent attacks.

https://cointelegraph.com/news/hacker-steals-15m-after-degens-pile-into-unreleased-yearn-finance-project

https://blog.scorechain.com/5-days-after-the-harvest-finance-hack-what-we-know-so-far/

https://medium.com/the-capital/defi-pulse-is-offline-after-a-24-million-hack-on-harvest-finance-420fe536460a

https://twitter.com/valentinmihov/status/1327750899423440896

7. Vampire Attacks

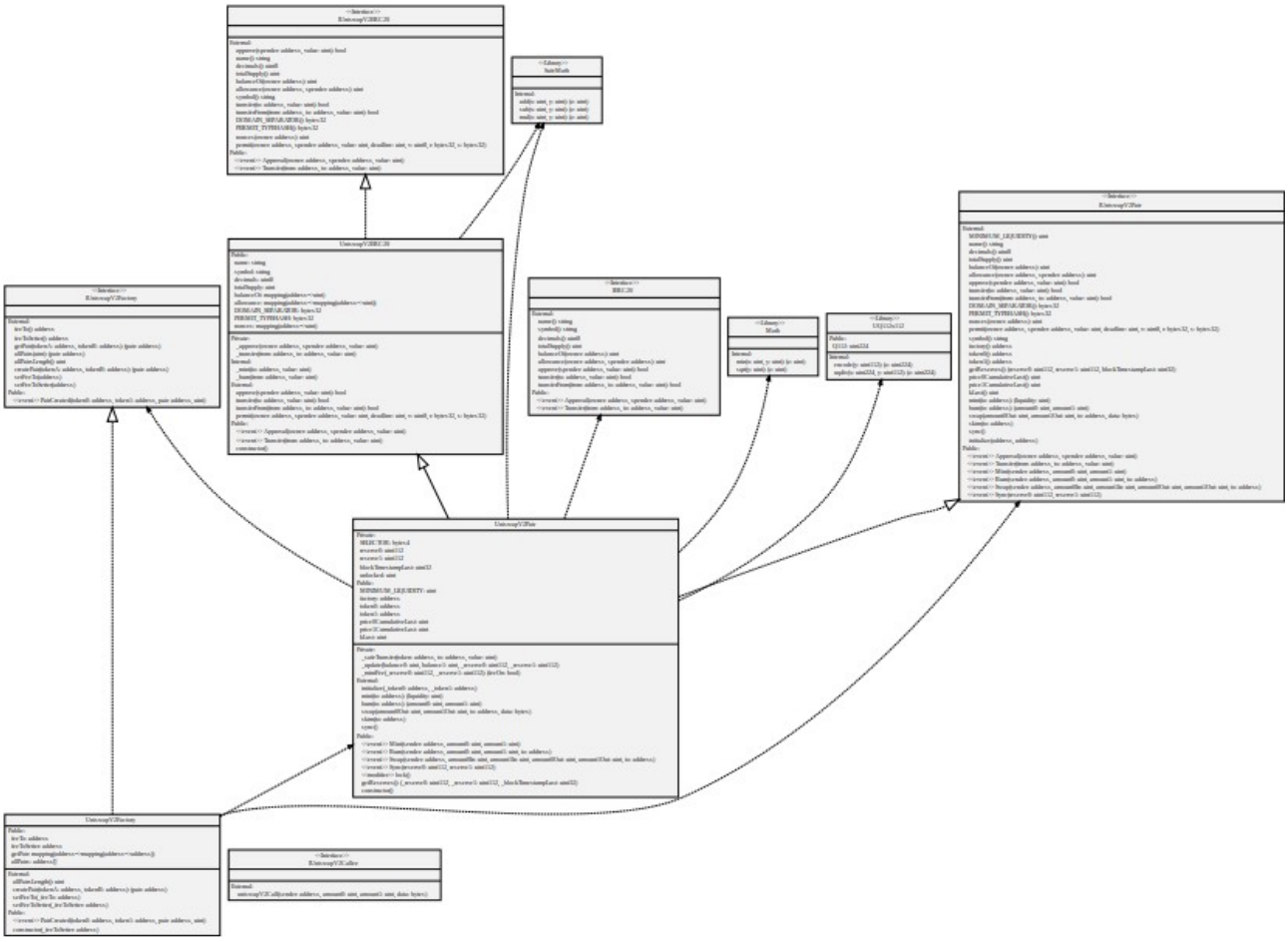
Sushiswap
Sushiswap is a clone of Uniswap with an attractive token economics. Sushiswap proposed an Uniswap like AMM with a native token for governance and as rewards for developers, liquidity providers and token holders. Sushiswap also used an approach, later termed as vampire attack. The approach is to use liquidity provider tokens of Uniswap for liquidity mining on sushiswap. And then to later migrate those liquidity to Sushiswap natively.

The approach was launched before Uniswap unveiled its UNI token. And provided Uniswap had a simple incentive scheme for liquidity providers and no way for Uniswap to lock its liquidity provider tokens, sushi took advantage, and users staked their liquidity provider tokens on to sushi and started yield farming on sushi. The migration of the Uniswap liquidity provider tokens to sushi happened after 14 days of the launch of the platform. And most value locked were later migrated to Uniswap. Sushi was launched on September 4th and on September 11th it had a maximum total value locked of $1.4Billion. And on September 18th, the value was depleted to $500Million and as of writing this article Sushiswap has a value of $300Million.

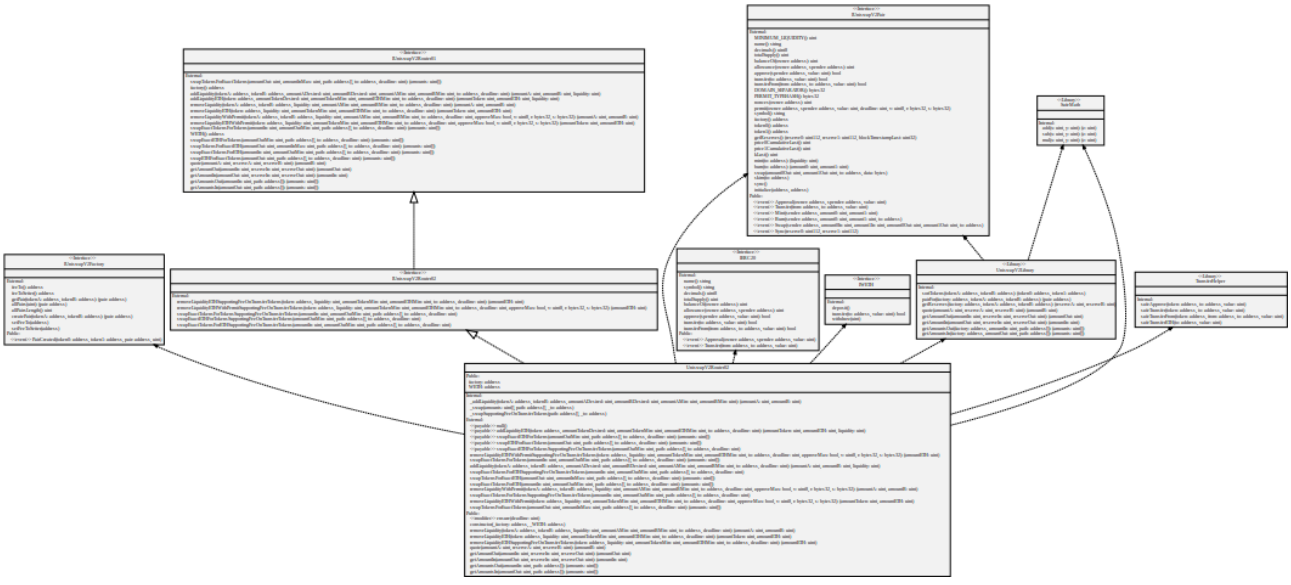
8. Yam Governance Bug

A rebase bug in YAM protocol leads to a lot more tokens being minted than desired. Possible causes of the same is because of hasty development, not adequate testing and lack of credible audit process.

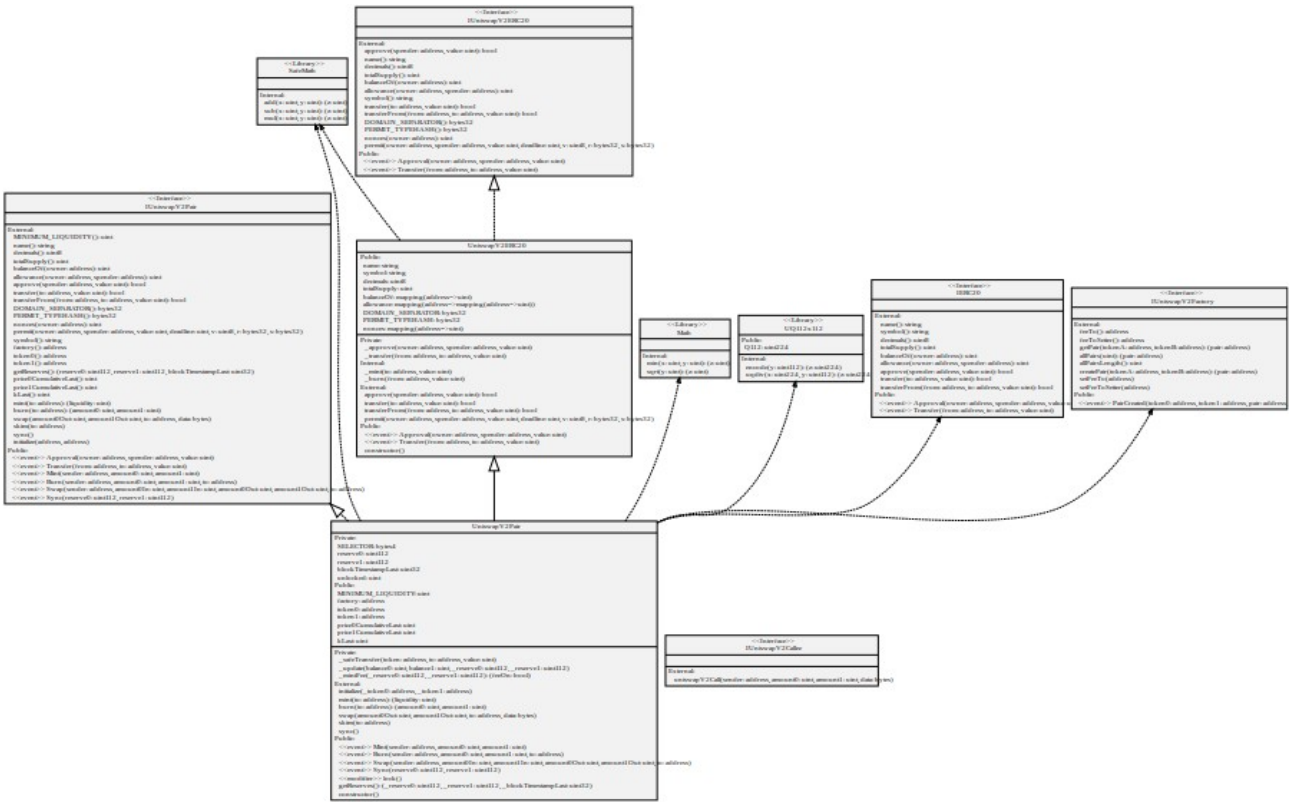
Annexure I: uniswapFactory.svg



Annexure II: uniswapRouter.svg



Annexure III: uniswapPair.svg



EOF.