去中心化交易所开发文档

1.引言

1.1目的

本文档提供了去中心化交易所(DEX)开发所需的详细信息，包括功能模块、用户界面设计、后端服务架构以及安全性设计。

1.2范围

本文档涵盖以下主要部分:

功能模块描述

用户界面(UI)设计

后端服务架构

安全性设计

2.功能模块

2.1首页

轮播图:用于广告宣传。

币种涨幅榜:展示币种波动情况，如AU、UTO、BTC、USDT、ETH。

快捷按钮:提供帮助中心、自选、充值、提现等快速入口。

公告栏:展示最新公告。

2.2领空投

项目方发布空投:项目方可以发布空投活动，并填写合约地址。

用户领取空投:用户可以领取空投，需要填写钱包地址并通过去中心化身份验证。

2.3币种涨跌幅：我们将实时监控并更新币种的价格变动和涨跌幅数据，为用户提供即时的市场动态信息。

热门榜：代币通过将Uto代币转入黑洞的方式获得上榜资格，转入金额的大小直接影响其在热门榜上的排名，金额越大，排名越靠前。

审查榜(人工审查)：代币需支付一定的审查费用，提交至DAO组织委员会进行审查，以确定其是否符合平台标准，避免割韭菜等不正当行为。

主流币榜：(自动评估)

审查榜(自动审查)(一般模板开发或者是上架的都会自动通过)(检测是否有老鼠仓、公共储备资金和流动性池有没有上锁、上永久的是自动上榜)

上架要求：上架前，必须燃烧至少1枚Uto代币并转入黑洞，作为上架的基本条件。

排行机制：

燃烧数量：代币燃烧的Uto代币数量是决定其在热门榜、审查榜。行中位置的主要因素，燃烧数量越多，排名越高。

时间顺序：若代币燃烧的Uto代币数量相同，则按照燃烧时间的先后顺序进行排名，新近燃烧的代币将获得更高排名。

审查流程：所有代币在上架前需经过DAO组织委员会的严格审查，确保其合法性和安全性，防止不正当行为。

费用支付：代币上架前需支付审查费用，以支持DAO组织委员会的运营和审查工作。

透明度：所有上架代币的燃烧记录和审查结果将在平台上公开，保证整个过程的透明度和公正性。

持续监管：代币上架后，平台将持续对其进行监管，确保其持续符合平台的标准和要求，任何违规行为都可能导致代币被下架。

3.用户界面设计

3.1首页设计

首页应包含所有快捷操作入口，以及实时更新的市场信息。

应有清晰的导航栏，方便用户快速找到所需功能。

3.2领空投界面

界面应提供简单的表单填写，以便用户领取空投。

需要有明确的提示信息，指导用户完成领取流程。

3.3行情界面

行情界面应提供实时的数据更新和图表展示。

应有筛选和排序功能，使用户能够根据需要查看不同币种的信息。

3.4交易界面

交易界面应简洁明了，方便用户快速下单。

应提供详细的交易对信息和交易历史记录。

3.5行情走势

曲线图（分时）:展示市场深度和实时成交情况。

K线图:提供市场深度和成交信息，以及币种的发行时间、发行总量、流通总量等简介。

3.6 UtoSwapv1去中心化代币价值交换系统

UtoSwapv1的去中心化代币价值交换系统。该系统借鉴了Uniswap v4的架构，并引入了若干创新机制，旨在优化流动性和定价策略。以下是对该系统运行原理的详尽解析。

初步理解

UtoSwapv1采用了一种修改自Uniswap v4的流动性池模型。Uniswap v4以其自动做市商（AMM）而著称，通过恒定乘积公式\[ x \times y = k \]来管理代币交换，确保流动性的高效利用。UtoSwapv1在此基础上进行了扩展，引入了更复杂的池子结构和动态调整机制，以优化流动性和定价策略。

关键组件解析

1. 流动性池（LP）：作为系统的核心，LP池负责管理代币交换所需的流动性。LP池中的代币遵循恒定乘积公式，以维持价格的稳定性和流动性。

2. 代币池：代币池持有代币的供应量，用户可以向其中添加代币以获得流动性池的收益，或在需要时从池中移除代币。

3.对付储备金：该池管理用户在交换代币时存入的资金。这些资金根据预设的比例分配至LP池和代币池，以平衡流动性和代币价值。

4. 动态容差调整系统：该系统根据代币的流通量动态调整容差值，以控制代币价值的增长，确保其与市场条件相匹配。

1. 混合定价机制：结合了LP池和支付准备金池的输入，通过动态调整比例来确定代币的买入和卖出价格，从而优化定价策略。

https://github.com/Uniswap/v4-core

https://docs.uniswap.org/concepts/protocol/oracle

运行原理:

1.对付储备金与流通量:

代币的最低价值由对付储备金除以流通数量决定

2.代币池子输入:(默认权限直接锁黑洞)

用户可以向代币池子输入代币数量，同时添加新发行的代币。

对付储备金池子输入:(默认权限直接锁黑洞)

储备金池(最低值建议1枚USDT)支持任何代币

公式

对付储备金池÷全网用户持有流通数量包含LP池子里面代币、不含主代币池、不含黑洞地址=初始价值

用户可以向对付储备金池子输入代币数量(建议添加99.9%)(场外没有币)

添加LP流动性(初始价格相同)(默认权限直接锁黑洞)

公式

恒定乘积公式\[ x \times y = k \]=初始价值

设置(买入和卖出回流百分比 )范围是从最低值精确到18位小数，到最高值5%。

买入滑点( ) % U回流对付储备金池(最小值为零)

卖出滑点( ) % U和币同时回流LP流动池(最小值为0.01%)

允许用户添加锁仓币、可以设置线性释放到代币池、可设置多久释放完成

允许用户设定动态容差（）

示例

初始条件：

中文名称：乌托邦币

英文简称：Uto

总供应量：1亿+1万+1枚

锁仓1亿枚线性释放代币池（300年释放完成）

线性释放公式

假设：

• \(T\)是总释放时间（年）。

• \(R\_0\)是初始每小时释放量。

• \(t\)是已经过的时间（年）。

• \(R(t)\)是在时间\(t\)时的每小时释放量。

• \(Q\)是累计释放量。

• \(\text{hours\\_per\\_year}\)是每年的小时数，即\(365\times 24\)。

1.每小时释放量随时间线性减少

每小时释放量\(R(t)\)随时间线性减少，可以表示为：

\[R(t)=R\_0\times\left(1-\frac{t}{T}\right)\]

2.累计释放量

累计释放量\(Q\)是从\(t=0\)到\(t=T\)的积分：

\[Q=\int\_0^T R(t)\times\text{hours\\_per\\_year}\,dt\]

代入\(R(t)\)：

\[Q=\int\_0^T R\_0\times\left(1-\frac{t}{T}\right)\times\text{hours\\_per\\_year}\,dt\]

3.计算积分

\[Q=R\_0\times\text{hours\\_per\\_year}\times\int\_0^T\left(1-\frac{t}{T}\right)\,dt\]

计算积分：

\[\int\_0^T\left(1-\frac{t}{T}\right)\,dt=\left[t-\frac{t^2}{2T}\right]0^T=T-\frac{T^2}{2T}=T-\frac{T}{2}=\frac{T}{2}\]

因此：

\[Q=R\_0\times\text{hours\\_per\\_year}\times\frac{T}{2}\]

4.求解初始释放量\(R\_0\)

假设目标是释放\(\text{total\\_tokens}\)个代币：

\[\text{total\\_tokens}=R\_0\times\text{hours\\_per\\_year}\times\frac{T}{2}\]

解得：

\[R\_0=\frac{\text{total\\_tokens}}{\text{hours\\_per\\_year}\times\frac{T}{2}}\]

\[R\_0=\frac{2\times\text{total\\_tokens}}{\text{hours\\_per\\_year}\times T}\]

公式总结

• 初始每小时释放量\(R\_0\)：

\[R\_0=\frac{2\times\text{total\\_tokens}}{\text{hours\\_per\\_year}\times T}\]

• 每小时释放量随时间\(t\)的变化：

\[R(t)=R\_0\times\left(1-\frac{t}{T}\right)\]

• 累计释放量\(Q\)：

\[Q=R\_0\times\text{hours\\_per\\_year}\times\frac{T}{2}\]

示例

假设：

• 总代币数量\(\text{total\\_tokens}=100,000,000\)枚。

• 总释放时间\(T=300\)年。

• 每年的小时数\(\text{hours\\_per\\_year}=365\times 24=8760\)小时。

计算初始每小时释放量\(R\_0\)：

\[R\_0=\frac{2\times 100,000,000}{8760\times 300}\approx 76.102739726027397260\text{枚/小时}\]

每小时释放量随时间\(t\)的变化：

\[R(t)=76.102739726027397260\times\left(1-\frac{t}{300}\right)\]

累计释放量\(Q\)：

\[Q=76.102739726027397260\times 8760\times\frac{300}{2}=100,000,000\text{枚}\]

以下是实现线性释放的代码：

from decimal import Decimal, getcontext

import time # 导入time模块

# 设置精度为18位小数

getcontext().prec = 18

# 初始化参数

total\_tokens = Decimal('100000000') # 总代币数量

total\_years = Decimal('300') # 总释放时间（年）

hours\_per\_year = Decimal('365') \* Decimal('24') # 每年的小时数

hours\_per\_day = Decimal('24') # 每天的小时数

days\_per\_week = Decimal('7') # 每周的天数

days\_per\_month = Decimal('30') # 每月的天数

# 计算初始每小时释放量

# 使用公式：total\_tokens = R0 \* hours\_per\_year \* total\_years / 2

initial\_release\_rate\_per\_hour = total\_tokens / (hours\_per\_year \* total\_years / Decimal('2'))

# 初始化变量

current\_release\_rate\_per\_hour = initial\_release\_rate\_per\_hour # 当前每小时释放量

total\_released = Decimal('0') # 已释放的代币总数

time\_elapsed = Decimal('0') # 已经过的时间（年）

# 进度条设置

progress\_bar\_length = 50 # 进度条长度（字符数）

# 模拟释放过程

while time\_elapsed < total\_years:

# 计算当前周期内的释放量

current\_release\_rate\_per\_hour = initial\_release\_rate\_per\_hour \* (Decimal('1') - time\_elapsed / total\_years)

release\_in\_current\_period = current\_release\_rate\_per\_hour \* hours\_per\_year

if total\_released + release\_in\_current\_period > total\_tokens:

# 如果当前周期释放后超过总代币量，则调整释放量

release\_in\_current\_period = total\_tokens - total\_released

# 更新已释放的代币总数

total\_released += release\_in\_current\_period

# 更新经过的时间

time\_elapsed += Decimal('1')

# 计算进度

progress = total\_released / total\_tokens

progress\_bar = int(progress \* progress\_bar\_length)

# 计算每小时、每天、每周和每月的释放量

release\_per\_day = current\_release\_rate\_per\_hour \* hours\_per\_day

release\_per\_week = release\_per\_day \* days\_per\_week

release\_per\_month = release\_per\_day \* days\_per\_month

# 打印当前周期的释放信息

print(f"年份 {int(time\_elapsed)}:")

print(f" 当前释放速率: {current\_release\_rate\_per\_hour:.18f} 枚/小时")

print(f" 当前周期释放量: {release\_in\_current\_period:.18f} 枚")

print(f" 每天释放量: {release\_per\_day:.18f} 枚")

print(f" 每周释放量: {release\_per\_week:.18f} 枚")

print(f" 每月释放量: {release\_per\_month:.18f} 枚")

print(f" 累计释放量: {total\_released:.18f} 枚")

print(f" 进度: [{'#' \* progress\_bar}{'-' \* (progress\_bar\_length - progress\_bar)}] {progress:.18f}")

print(f" 已经过时间: {time\_elapsed:.18f} 年")

print("-" \* 60)

# 暂停1秒钟

time.sleep(1)

# 检查最终释放量是否达到100%

final\_release\_percentage = (total\_released / total\_tokens) \* Decimal('100')

if final\_release\_percentage == Decimal('100'):

print(f"释放量达到目标范围（100%）: {final\_release\_percentage:.18f}%")

else:

print(f"释放量未达到目标范围（100%）: {final\_release\_percentage:.18f}%")

print(f"释放100%的代币需要大约{total\_years:.18f}年。")

以下是考虑动态容差后的初始价及定价公式和相关计算的优化版本：

初始价及定价公式

权限黑洞代币池子

• 数量：1万枚 Uto

• 兑付储备金：1枚 USDT

公式

\[\text{初始价值}=\frac{\text{兑付储备金池}}{\text{全网用户持有流通数量（包含LP池子里面代币，不含主代币池、不含黑洞地址）}}\]

• 初始价值：1 U

权限黑洞 LP 流动池

• 数量：1枚 Uto

• 兑付储备金：1枚 USDT

公式

• 恒定乘积公式：\[x\times y=k\]

• 初始价值：1 U

买入滑点

• 滑点比例：0.5%U 流回兑付储备金池

卖出滑点

• 滑点比例：0.5%U 和币同时回流 LP 流动池

考虑动态容差后的定价公式

动态容差设置

• 起始容差：1%

• 最终容差：5%

• 容差代币区间：2～1000万枚

• 超过1000万枚时容差保持为：5%

动态容差计算公式

1. 每枚代币容差增加量：

\[\text{每枚代币容差增加量}=\frac{\text{最终容差}-\text{起始容差}}{\text{最多代币容差}-1}\]

2. 任意代币\(x\)的容差计算：

\[T(x)=\text{起始容差}+\left(\frac{\text{最终容差}-\text{起始容差}}{\text{最多代币容差}-1}\right)\times(x-1)\]

3. 对于流通量减少的情况：

\[T(x)=\text{最终容差}-\left(\frac{\text{最终容差}-\text{起始容差}}{\text{最多代币容差}-1}\right)\times(\text{最多代币容差}-x)\]

买入价值计算公式

\[\text{买入价值}=\left(\frac{\text{兑付储备金池}}{\text{全网用户持有流通数量（包含LP池子里面代币，不含主代币池、不含黑洞地址）}}+\text{动态容差}+\frac{\text{LP流动池}[x\times y=k]}{2}\right)\]

卖出价值计算公式

\[\text{卖出价值}=\left(\frac{\text{兑付储备金池}}{\text{全网用户持有流通数量（包含LP池子里面代币，不含主代币池、不含黑洞地址）}}+\frac{\text{LP流动池}[x\times y=k]}{2}\right)\]

买入流动权重计算

权重计算公式

\[\text{权重}=\frac{\text{某部分的数量}}{\text{总数量}}\times 100\%\]

计算过程

1. 黑洞代币池子：1万枚 Uto

2. 权限黑洞 LP 流动池：1枚 Uto

3. 总数量：1万枚+1枚=10001枚

储备金池资金流入权重

\[\text{储备金池权重}=\frac{10000}{10001}\times 100\%\approx 99.99\%\]

LP 流动池流入权重

\[\text{LP 流动池权重}=\frac{1}{10001}\times 100\%\approx 0.01\%\]

解释

• 储备金池：由于其数量占绝大部分（10000枚），因此其权重非常高，约为99.99%。

• LP 流动池：数量极少（1枚），因此其权重非常低，约为0.01%。

买入操作及更新

买入操作

假设用户买入\(x\)枚 Uto，实际消耗的 USDT 数量为\(y\)。

1. 计算动态容差：

• 根据当前流通量\(x\)，使用动态容差公式计算当前容差\(T(x)\)。

• 如果流通量\(x\)在2～1000万枚之间，则根据公式计算动态容差。

• 如果流通量\(x\)超过1000万枚，则容差保持为5%。

2. 计算买入价值：

买入价值计算公式

\[\text{买入价值}=\left(\frac{\text{兑付储备金池}}{\text{全网用户持有流通数量（包含LP池子里面代币，不含主代币池、不含黑洞地址）}}+\text{动态容差}+\frac{\text{LP流动池}[x\times y=k]}{2}\right)\]

解释

(对付储备金池÷全网用户持有流通数量包含LP池子里面代币、不含主代币池、不含黑洞地址+动态容差+LP流动池\[ x \times y = k \])÷2=买入价值

3. 滑点处理：

• 0.5%的\(y\)流回兑付储备金池。

• 实际消耗的 USDT 数量为\(y\times(1-0.005)\)。

4. 资金分配：

• \(y\times 99.99\%\)转入兑付储备金池。

• \(y\times 0.01\%\)转入 LP 流动池。

5. 更新数据：

• LP 流动池和兑付储备金池的金额更新。

• 代币池中增加\(x\)枚 Uto。

• 重新计算权重。

卖出操作及更新

卖出操作

假设用户卖出\(x\)枚 Uto，实际获得的 USDT 数量为\(y\)。

1. 计算动态容差：

• 根据当前流通量\(x\)，使用动态容差公式计算当前容差\(T(x)\)。

• 如果流通量\(x\)在2～1000万枚之间，则根据公式计算动态容差。

• 如果流通量\(x\)超过1000万枚，则容差保持为5%。

2. 计算卖出价值：

卖出价值计算公式

\[\text{卖出价值}=\left(\frac{\text{兑付储备金池}}{\text{全网用户持有流通数量（包含LP池子里面代币，不含主代币池、不含黑洞地址）}}+\frac{\text{LP流动池}[x\times y=k]}{2}\right)\]

解释

(对付储备金池÷全网用户持有流通数量包含LP池子里面代币、不含主代币池、不含黑洞地址+LP流动池\[ x \times y = k \])÷2=卖出价值

3. 滑点处理：

• 0.5%的\(x\)和对应的 USDT 数量转入 LP 流动池。

• 实际卖出的 Uto 数量为\(x\times(1-0.005)\)。

4. 资金分配：

• \(y\times 99.99\%\)转入兑付储备金池。

• \(y\times 0.01\%\)转入 LP 流动池。

5. 更新数据：

• LP 流动池和兑付储备金池的金额更新。

• 代币池中减少\(x\)枚 Uto。

权重更新逻辑

1. 计算新的权重：

• 在每次交易后，根据最新的 LP 流动池、兑付储备金池和代币池的数量重新计算权重。

• 权重计算公式为：

\[

\text{权重}=\frac{\text{某部分的数量}}{\text{总数量}}\times 100\%

\]

• 例如：

• 黑洞代币池子：1万枚 Uto

• 权限黑洞 LP 流动池：1枚 Uto

• 总数量：1万枚+1枚=10001枚

• 储备金池权重：

\[

\text{储备金池权重}=\frac{10000}{10001}\times 100\%\approx 99.99\%

\]

• LP 流动池权重：

\[

\text{LP 流动池权重}=\frac{1}{10001}\times 100\%\approx 0.01\%

\]

2. 更新权重的具体步骤：

• 在每次交易后，根据交易后的最新数据重新计算权重。

• 更新权重后，需要将新的权重值存储到系统中，以便后续的交易和计算使用。

• 例如，如果用户买入了\(x\)枚 Uto，那么代币池的数量会增加\(x\)枚，LP 流动池和兑付储备金池的金额也会相应更新。此时，需要重新计算权重：

• 新的总数量=原总数量+\(x\)

• 新的储备金池权重=\(\frac{\text{新的储备金池数量}}{\text{新的总数量}}\times 100\%\)

• 新的 LP 流动池权重=\(\frac{\text{新的 LP 流动池数量}}{\text{新的总数量}}\times 100\%\)

3. 权重更新的频率：

• 权重更新应在每次交易完成后立即进行，以确保系统的实时性和准确性。

• 权重更新的频率应与交易的频率保持一致，以保证系统能够及时反映最新的市场情况。

补充说明

虽然文档中提到了“每次交易后更新权重”，但为了更清晰地指导开发和维护，建议补充上述详细的权重更新逻辑和步骤。这样可以确保开发人员和维护人员能够更好地理解和实现这一功能，避免因理解不一致而出现错误。

注意事项

1. 动态容差的计算：动态容差的计算需要根据当前流通量实时更新，确保代币价值的稳定性和市场灵活性。

2. 滑点处理：买入和卖出的滑点处理需要严格按照公式执行，确保资金的合理分配。

3. 数据更新：每次买入或卖出操作后，都需要更新 LP 流动池、兑付储备金池和代币池的数据，并重新计算权重，以保证系统的正常运行。

初始价及定价公式

例如(未考虑动态容差+LP流动池容差)

买入0.995枚Uto 预计需要消耗≈1枚USDT、0.995枚USDT 99.99%转入储备金池、0.01%转入LP流动池、滑点0.5%、0.005枚转入储备金池

每次购买更新LP 流动池和储备资金池、代币池、重新计算权重

0.995枚Uto:99.99%代币池和0.01%从流动池转入用户钱包

预计到账≈0.995枚Uto

例如卖出0.995枚Uto 0.99枚Uto 99.99%转入代币池、0.01%转入LP流动池、滑点0.5%，0.005枚Uto和USDT转入LP流动池

0.99枚USDT99.99%储备金池和0.01%从流动池转入用户钱包

预计到账≈0.99枚USDT

每次卖出更新LP 流动池和储备资金池、代币池、重新计算权重

4.2API设计

API应提供完整的端点，以支持前端的所有功能。

应有安全措施，如认证和授权，保护API端点。

4.3服务器和网络

服务器应有足够的处理能力和存储空间。

网络应设计为高可用性和安全性。

5.安全性设计

5.1加密和认证

所有敏感数据应进行加密处理。

用户认证应采用多因素认证方式。

5.2防御机制

系统应具备防止DDoS攻击的能力。

应有定期的安全审计和漏洞扫描。

5.3合规性

平台应遵守相关的法律法规和行业标准。

应有透明的隐私政策和用户协议。

6.7安全性考虑

认证授权:确保只有订单的所有者可以查看、修改或取消订单。

数据加密:订单数据在传输和存储过程中应进行加密，以防止数据泄露。

6.8用户界面设计

直观展示:订单簿应直观展示所有挂单，包括价格、数量、订单状态等信息。

操作便捷:用户应能轻松地进行挂单、取消和修改操作。