DAO stack

Matan Field, Yaron Velner, others

Abstract. DAO stack is a framework for...

1 Introduction

2 System of Value

Agents. An *agent* is an entity with an address that can interact with blockchain contracts. Formally, an agent is either an account whose private key is known to some individual or a contract. We denote the set of all agents by $\mathcal{A}$. An *agent operation* is a valid transaction (technically for Ethereum either an external or internal transaction). We denote the set of finite sequences of agent operations by $\mathcal{O}$. Let f be a function from $\mathcal{O}$ to some domain D. We say that a sequence $O' \in \mathcal{O}$ is the *f-successor* of $O \in \mathcal{O}$, if O' is the shortest extension of O such that $f(O) \neq f(O')$ (if two shortest extensions exist, then one is chosen arbitrarily, e.g., according to lexicographical order).
System of value. A system of value is a tuple $\mathcal{S} = \langle T, T_u, R, R_u, C, C_u, E_s, E_o, E_u, D, D_u, U \rangle$, where intuitively

- T is the *native token* (or value) of the system and T_u is a mechanism to update the token distribution.
- R is a *reputation* metric, controlled by the system and R_u is a mechanism to update the reputation distribution.
- C is a mechanism to submit *contributions* for the system and C_u is a mechanism to change C.
- E_s and E_o are mechanisms for the evaluation of a contribution. The role of E_s is to evaluate *subjective* contributions and the role of E_o is to evaluate *objective* ones. E_u is a mechanism to update E_s and E_o.
- D is a delegation mechanism to perform agent actions in other systems and D_u is a mechanism to change D.
- U is a mechanism to update all the *update mechanisms*, namely, T_u, R_u, C_u, E_u and U itself.

A system of value is an agent.

We now formally define each of the components.
Tokens. A token distribution is a function $T : \mathcal{A} \to \mathbb{N}$ (where $\mathbb{N}$ is the set of non-negative integers). The distribution of the tokens, defined by T, can be changed by the mechanism T_u in two ways, namely, *transferring tokens* and *minting tokens*.

Token transfer changes the value of T for at most two agents [1]. In a transfer the sum of all tokens in the system is not changed. Formally, let T be the token distribution before a transfer and T' be the result of one transfer, then it always hold that $\sum_{a \in \mathcal{A}} T(a) = \sum_{a \in \mathcal{A}} T'(a)$.

Token minting changes only the value of $T(\mathcal{S})$. It may increase or decrease it. **[Matan: I'm not sure decreasing make sense since one can transfer his tokens to somewhere else, unless it's about frozen or constraint tokens - Yaron: you mentioned burning tokens. For example burning 5 tokens in exchange to one ether]**

The update function T_u dictates the rules on how T is changed. For example it can limit token transfer volume or prevent the transfer of tokens before some maturity threshold is reached. It can also dictate minting policy such as constant inflation (or deflation) rate, backing of the token with other tokens, etc.

Formally, the update function T_u is a function $T_u : \mathcal{O} \to (\mathcal{A} \to \mathbb{N})$ (i.e., it assigns a distribution function to every sequence of operations), that satisfy the next restriction: If a sequence of agent operations O' is the T_u-successor of O, then either:

[1] Technically, transfer between more parties are braked down into a sequence of two party transfers.

- $\sum_{a\in\mathcal{A}}(T_u(O))(a) = \sum_{a\in\mathcal{A}}(T_u(O'))(a)$ and there exists exactly two agents a_1, a_2 such that $(T_u(O))(a_i) \neq (T_u(O'))(a_i)$ for $i = 1, 2$.
- For every agent $a \neq \mathcal{S}$: $(T_u(O))(a) = (T_u(O'))(a)$.

[Matan: I didn't understand the "next restriction"; also, maybe there's a typo in there?]
Reputation. A reputation is a function $R : \mathcal{A} \to \mathbb{N}$ (maybe also negative reputation is allowed?). **[Matan: Negative reputation has only meaning when entity are not disposable (which they usually are). Basically it means that entities are also tied up to some external data, a social-media profile, a physical identity, a company, external (physical or digital) ownership of assets etc.]** The reputation update mechanism may increase or decrease the reputation of a single agent. Therefor, reputation is not explicitly transferable [2]. Hence, the reputation update mechanism is a function $R_u : \mathcal{A} \to (\mathcal{A} \to \mathbb{N})$, such that if O' is the R_u-successor of O, then there exists at most one agent a with $(R_u(O))(a) \neq (R_u(O'))(a)$.

If the system wishes to update the reputation of more than one agent, then it has to break it down to a sequence of single reputation updates. **[Matan: Why? not clearly possible; definitely not for BF-like reputation flow. Yaron: in any case, in solidity you can only do bounded number of operations. It is not possible to update the reputation of all members. In reputation flow, each agent would have to initiate the update of its own reputation.]**
Contribution submission. A contribution is a bulk of data that is submitted to the system by one of the agents. A contribution can be immediately rejected (e.g., only agents from certain list are eligible to submit contribution) or can be stored in the contract for evaluation. For the sake of abstraction we consider data bulks to be integers (e.g, the according to their byte encoding). A contribution submission mechanism is a function $C : \mathbb{N} \times \mathcal{A} \to \{0, 1, 2\}$. Let d be a contribution that is submitted by agent a. Then:

- If $C(d, a) = 0$, then the contribution is not stored and immediately rejected.
- If $C(d, a) = 1$, then the contribution is a candidate for *objective evaluation*.
- If $C(d, a) = 2$, then the contribution is a candidate for *subjective evaluation*.

The update mechanism of C is a function $C_u : \mathcal{O} \to (\mathbb{N} \times \mathcal{A} \to \{0, 1, 2\})$. There are no restrictions on C_u.
Evaluation of contributions. Contributors are rewarded (might also be penalized???) with a multi-value reward, which consists of: (i) reputation; (ii) native (system) token; and (iii) non-native tokens that the system control. For simplicity we describe the case where the system holds at most one non-native token, namely the token of a system $\mathcal{S}'$, and the extension to multiple non-native tokens is straight forward. We say that an agent a is rewarded by (x, y, z) if its reputation $R(a)$ is increased by x, its native token balance $T(a)$ is increased by y and its token balance in $\mathcal{S}'$ is increased by z via a transfer update. Hence, a reward is a triplet in $\mathbb{N} \times \mathbb{N} \times \mathbb{N}$ ($\mathbb{N}^3$).

The *objective evaluation function* E_o decides how to reward a contribution by considering only the identity of the submitter and the content of the submission. I.e., the objective evaluation is a function $E_o : \mathbb{N} \times \mathcal{A} \to \mathbb{N}^3$. For example, E_o may reward $(0, 1, 0)$ for an agent who submitted a proof of paying 1 Ether to the system's contract. Another example is where E_o rewards $(0, 0, 1)$ for an agent who submitted a proof of transferring the native token back to the system. A contribution d from agent a is eligible for a reward only if $C(d, a) = 1$.

The *subjective evaluation function* E_s decides how to reward a contribution based on the actions of agents. E.g., agents may evaluate the contribution by taking a vote. Hence, the subjective evaluation mechanism is a function $E_o : \mathcal{O} \times \mathbb{N} \times \mathcal{A} \to \mathbb{N}^3$ A contribution d from agent a is eligible for a reward only if $C(d, a) = 2$.

TODO - should put restrictions on subjective evaluation? E.g., only reputation counts? Must be blind to the actual identity of the agent, etc.

[Matan: Can unify objective and subjective distributions; can say generally agents vote on "proposals" which are forms of suggested distributions, e.g. 1. propose to distribute 100 tokens to this agent for this contribution; 2. propose to distribute x tokens to objective activity of this type; 3. propose to distribute tokens for subjective contributions according to median

[2] An agent may chose to delegate its reputation by handing partial control over the agent's operations. **[Matan: This may have consequences]**

(or other rule); 4. propose to distribute tokens for subjective contributions according to ... mixture of objective measures and agents inputs of various degrees...]

The function E_u updates the evaluation mechanisms. Formally it is a function $E_u : \mathcal{O} \to ((\mathbb{N} \times \mathcal{A} \to \mathbb{N}^3) \times (\mathcal{O} \times \mathbb{N} \times \mathcal{A} \to \mathbb{N}^3))$.

Inter-system extension For a system of value to be an agent in a different system of value, it must have the ability to perform agent operations. In particular it should be able to submit contributions and to evaluate other contributions. In the most general setting, the system should be able to delegate its token holdings and reputation**s** (i.e., reputation in other systems) to perform actions in other systems. Formally, a *delegation mechanism* is a function $D : \mathcal{O} \to \mathcal{O}$, which decides on $\mathcal{S}$ next action or actions based the actions of its agents. For example, agents can take a vote on the next action of the system. We note that D can only decide on actions that $\mathcal{S}$ does in other systems. The function D_u updates the delegation mechanism. Formally it is a function $D_u : \mathcal{O} \to (\mathcal{O} \to \mathcal{O})$.

General update mechanism. Each of the token, reputation, contribution submission and contribution evaluation update mechanism and even the general update mechanism may be changed. For this purpose we extend the evaluation mechanism, such that in addition to suggested reward, it would also output *opinions* on whether the submitted contribution should replace T_u, R_u, C_u, E_u, D_u or U. The general update mechanism U is a threshold for every mechanism, such that certain mechanism is replaced if and only if the alternative is evaluated with an higher value.

3 Recommended Mechanisms

3.1 Recommended Token Mechanism

3.2 Recommended Reputation Mechanism

3.3 Recommended Contribution Mechanism

3.4 Recommended Upgrading Mechanism

4 Analyzing Circle of Systems

5 Implementation