Distributed Town

1. Org Template:

Token Design & Credit System

Community-Template: Reputation

Token-Type: (1) Governance Token. (2) Transferable

Token-Properties:

(1) Voting + Reputation (2) Reward/Payment for service

Community-Creation: Users can easily create new communities. Each community is created identical to the other, following the *DiTo-Community* Template

DiTo-Community Template:

- **Community Funds (initVault):** 96 000 DiTo
- **Members Cap (memCount):** 6-24 members / community*
- **New-User Membership (lowCount):**
  Each **DistributedTown's User** will receive **2000 DiTo (initVault/48)** when they join their first Community.
- **Members TOT. Credit (diCount):**
  in the range [2000 (**lowCount**), 3840 (**upCount**)]
  lowCount: initVault/48
  upCount: initVault/25
  => diCount: 2000 ≥ diCount ≤ 3840

Distributed Town

2. Org Template: Skills (default)

Skills (Types & Values):

(1) Community Life (each skill) ==> 12 DiTo (defSkill)
(2) At Home (each skill) ==> 6 DiTo (defSkill)
(3) Professional (each skill) ==> 24 DiTo (defSkill)

defSkill * XP = skillValue

Initial diCount / Member:

lowCount + sum skillValue

“Gig” / Tasks / Rewards:

Members of a community can post tasks/gigs/helps that they need from others. They work as “open bounties” - where publishers specify *(1) label* (skill needed), *(2+3) Title & Description* of help needed, *(4) reward amount* in DiTo. For basic wallet administration and transactions:
1. **lowCount** = 2000 DiTo = min DiTo ownable per member
2. **upCount** = 3840 DiTo = max DiTo ownable per member 3. **initCount** = the amount of DiTo a user owns at the time of initiating a transaction
4. **newCount** = initCount - the cost for that action / interaction ==> **newCount = initCount - Cost**

For a Gig to be published, a transaction (diTx) needs to be valid ==>
diTx = True if newCount > lowCount

Distributed Town

3. Org Template:

Voting Power & Approval Rate

- Voting Power (**repCount**) = **diCount**
- **Approval Rate (can be implemented later):**
  - To be approved, a proposal needs to have 50% + 1 of the votes ==> majQuota = (votCount / 2) + 1
  - Each member can vote only once, each vote's value = voter **diCount ==> voteSum = Σ diCount(1→votCount)**
  - At least 50% + 1 of the community (memCount) must vote ==>
    ==> **votCount > (memCount/2) + 1**
  - The total amount of the voters need to hold at least the 25% of the total repCount assigned (diRep) ==>
    ==> **voteSum ≥ diRep / 4**