Community Management for Tezos

What does the community need
Dinkar Ganti.

Introduction

- Tezos is an advanced blockchain ledger technology based on OCaml that has been on mainnet for more than a couple of months without any known issues and excellent mechanism for governance as well as low-energy baking.
- I would like to present what i consider is needed to help the community grow.

What Can we do next?

- Now that tezos is deployed, what do we need to do next as a community. We will try to address that in the coming slides.

Next slides

- Goals of Michelson
- Ease of use of Liquidity
- How do we structure the community, covering the various roles that need to be filled to create a vibrant community.

Brief note about Michelson

- Quoting from the page on Michelson-lang
  - “Our current implementation of Michelson is based around an OCaml GADT, which we have used to verify the type-soundness of the language. Additionally, the implementation of a stack based language maps directly to the semantics. The same is not true for any efficient implementation of the lambda-calculus. There have also been two formally verified implementations of Michelson, one in Coq and one in F*. One day, we hope to replace our current implementation with a verified one.”
- As a community, we would like to see the above intention to fruition as both Coq and F* guarantee termination as well as put an upper cap on the number elements in a list that needs to be reversed as in this example:
  - July 27, 2017: Reverse
  - Here's a contract that takes a list of strings and reverses it. This contract gets into the list instructions and the `LOOP` instruction.

Brief note about Liquidity

- As is evident from the code samples on Michelson, the language is a bit hard to program in. Enter Liquidity.
  - ```
    type game = {
      number : nat;
      bet : tez;
      player : UnitContract.instance;
    }

    type storage = {
      game : game option;
      oracle_id : address;
    }
    ```
  - The above is a sample from the documentation where the developer can implement smart contracts using language constructs that are similar to OCaml therefore a bit more approachable as well as maintainable.
    ```

How do we structure the community

- In a centralized world, everyone would run to a venture capitalist to fund the next “big” idea. For the Tezos community, do we really need that?
- Can we instead, structure our community into streams that we think will provide for the ecosystem?
  - Provers - a group of individuals who have built up a level of skill to prove Coq/F* code that eventually can get converted into Michelson (as outlined in one of the goals of Michelson).
  - Contract library developers to write them.
  - Application developers
  - Protocol developers
  - Advanced research and testing
  - Client libraries
  - Sample code
  - Documentation
  - Hackathons/Mobprogramming
  - Tez-Support
  - TzCIP - Tezos Community Improvement Proposals

Provers

- This community comprises of highly skilled researchers who can review/prove contracts as needed.
  - The community will be subsidizing their time as the value is in ensuring that the block chain is not compromised
- This community can act as the gate keepers to deploying contracts, the benign kind.

Contract library developers

- Much as in developing C, Java, OCaml libraries a core set of contracts that have been proven is needed to help speed up the deployment of contracts and applications. For example,
  - Holiday calendar that is global using algorithms such as the ones documented in “Calendrical computations”. The reason this might work is that there is a way to actually compensate the owners of the algorithm through a royalty based smart contract. This might encourage developers to reuse algorithms and not re-invent them to be tested/proved again.
  - Shopping carts.
  - Chain of custody based contracts.
  - Legal contracts
  - TezSupport contract (tbd)

App developers

- The community needs application developers to help grow the network. Application developers can of course raise funds through traditional means such as an ICO or they could perhaps chose to release their application and the code to the public to build on the open sourced contract libraries that they used. As a community we need to provide for a platform that might convince the application developer to choose any approach and still have an expectation to be rewarded for the initiative.

Protocol developers

- Self amending ledgers presents interesting possibilities that can help develop dedicated communities with a specialized requirements. I am still not sure if there is a market here, and would probably love to explore that with the core developers at some point.

Anticipated improvements to the Tezos Nodes

- Light nodes
- Faster transaction commits
- Sharding

There is more on this in Tezos' foundations articles. As a community we need to manage these announcements and help the Tezos foundation with their work in a way we can, perhaps in testing, documenting and maybe even in actual code commits.

Client Libraries

- Each developer has their own favorite programming language. As a community we need to fund initiatives to create SDKs or libraries to help developers implement web or mobile applications using Tezos. The libraries that interest me are
    - Haxe as it targets many platforms including the iphone and the android to name a few.

Create more samples

- The best way to learn a language is to code in it.
- The second best way, perhaps, is to read examples and sample code. We need to develop more of them in
  - Haxe
  - Javascript
  - Haskell
  - Ocaml

Documentation

- Documentation that reduces the barrier to entry has been known to creating a successful open source project. Tezos’s documentation is excellent, though the needs of the community can be supplemented by community driven videos, notes and readme for various projects.

Hackathons/mob programming

Instead of hackathons, i would like to influence the community to accept the community building practices of #mobprogramming to help build robust software.

- The oracle is a great resource for learning to write smart contracts. We need to make it accessible in a variety of programming languages.
- Rewrite the oracle in haxe/haskell/ocaml (your fave middleware here) : link
  - The current sample is illustrative. We need to build further on that with the help of the liquidity team for example so we can show how liquidity can be used inside a web application. This session could be divided into 2 parts
    - Session on liquidity and contracts.
    - Session on the details of injection of operations, the libraries that we can use etc.

Mobprogramming (contd.)

- Specifically, these parts:
  - Build transaction manually, sign and inject. This can be done with:
    - /chains/main/blocks/head/helpers/forge/operations
    - Signing must be done locally:
      - prepend bytes 03
      - hash with Blake2B
      - Sign with either ed25519, secp256k1, p256 depending on your private key
    - /chains/main/blocks/head/helpers/scripts/run_operation
    - /injection/operation
- As a community we need to setup an environment where one can freely develop and deploy applications that members of the community think will be useful to the community.

Tez Support

- At some point support activity needs to be paid for, therefore this group needs to be formed by the community much as we fund a local fire-station or a library.
  - Establish roles and expectations.
  - Create a dedicated bug report system to help various levels of support communicate bug reports in appropriate level of detail.
- This organization will most likely replicate most of the follow the sun model for technical support and even though they can and most likely not be the only ones on the matrix, the user community can expect that at least a dedicated number is always on call and that they have the necessary tools to help the users as well as the channels to escalate to the core dev team if needed.

Tezos Community Improvement Proposals

- Borrowing from the spirit of PEP (Python Enhancement Proposals), these proposals would most likely focus on improvement in the usability of the ledger and will cover topics that don’t overlap with any of the core development proposals.

Summary

Tezos with its self amending ledger and support for delegated-proof-of-stake among many many other features, is way ahead of the rest of the public block chain based solutions. This is an exciting time to be part of a community that is going to be developing ecosystems that will help put together solutions that the wider community of end users needs.