Skunks Future Timelines

During Todays Architecture meeting the Anoma Node engineering team made the following outline of work we wish to do for the next 6

months of Anoma Node development.

Note that this might change quite dramatically once we see V2 specs, however this is a good overview of the work we wish to work on.

Item

Time Frame

Comment

Scry + Supervision

2 Weeks

Text

Snapshotting/Event Replaying

1-2 Weeks

Simplifying the System

Nock VM Changes

2 Weeks

Making it Stateful

VM Standard Library Changes

??? Scope

Supporting Juvix

Software Architecture For Consensus

1 Week

Before Consensus from specs

External API Considerations

2 Weeks

Analysis on our subsystems

Proper Storage Separation

1 Month

Model Sharded Storage

Operators Web Dashboard

< 1 Week

User Facing Documentation

3 Weeks

Docs for actual users

Anoma Node V2

??? Scope

Depends On Specs

Live Views to Livebook

In perpetuity

View for our Code

FEMA analysis

We should try

Proper OTPfying the codebase

On Going

We are doing this in Scry

Items

The strategy for all these Items are the same

1. We do analysis
2. We figure out a map (note Y axis is meaningful) of the issues
3. We break out work and assign it.

Vm Standard Library Changes

- We need to collect the list of functions we need to implement

Software Architecture For Consensus

- Halt if all nodes don’t agree
- Just something basic that setups up our architecture
- We can get multi-node communication (Proper)
- We can see where it fits into our architecture
- This Prepares us for V2 Specs. We know what’s included roughly so we ideally don’t need to change our design much.

Proper Storage Separation

We need to properly think about this problem on it’s own

This encompasses making transactions run in parallel since we’ll have the proper sharded space

This implies that our backend’s API has to note what keys they wish to read etc.

External API Considerations

Our Transport, TCP Connections, etc. are a bit of a mess, we should do analysis on the codebase and aggressively simplify the code.

Further we need to think about how other people call into our application and fixup any issues therein.

Operators Web Dashboard

We should take Phenoxi’s live dashboard at first

Then add Anoma specific operations to the Node

Web dashboards are nice and there for later extension.

Live Views to Livebook

I have commissioned D to work on our Livebook Architecture.

But we can in the meantime make graphical views for our codebase as is.

I think we really should consider this before any work from D comes in, as it’s high leverage.

User Facing Documentation

Application documentation From Juvix + Application user group

However we should try to document our codebase and our application for user operation.

Greater internal documentation would likely be useful here even to users who don’t care as it shows how our codebase works and what they might be able to do.

FEMA analysis

At the very least we should analyize what each Engine/Genserver assumes and how the failure modes relate.

We should document what each server assumes etc etc etc

Proper OTPfying the codebase

The Scry changes are already starting this process however we have a long way to go, and I believe FEMA analysis will start to aid this process as it makes what each engine requires obvious, meaning that when we want to add proper restart behaviors it will aid those efforts.