Atmosphere design

Components

Validator

The network layer is responsible for communication between the validators in the Smart Mesh network, including communication initialization, discovery of other nodes and message processing. When instances of validator begin, they start listening to a specific interface and port for incoming connections. After connection and peering, validators exchange messages with each other based on the protocol rules for gossip or epidemic.

The main purpose of the component is to simplify the network layer as much as possible. For example, network layer doesn't need to know the payload information of the application messages, also doesn't need to provide application-level data to connect peers or to establish network connectivity and conversely to send and receive messages. Application shouldn't understand details of the network implementation.
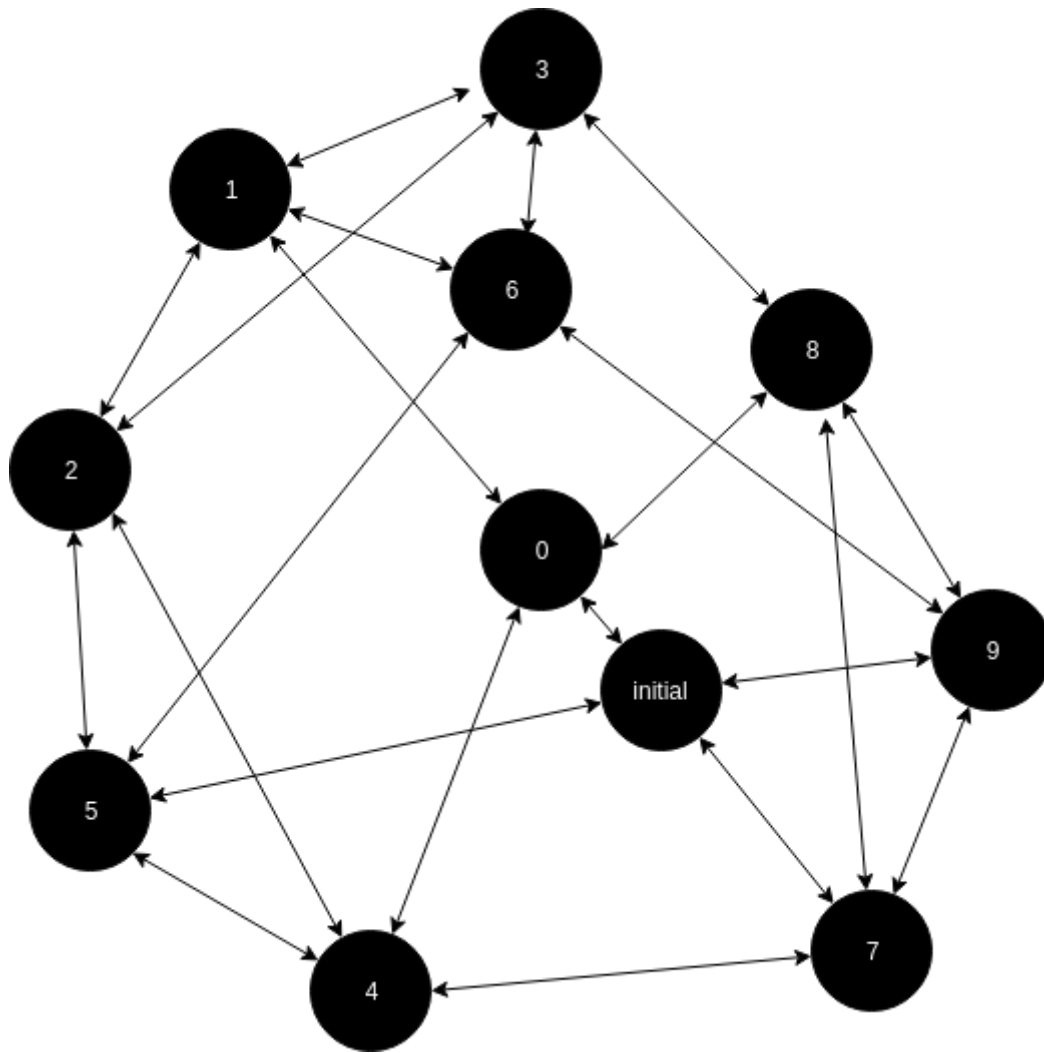
Network

Bi-directional peering through interrogation of neighbor neighbors provides reliable communication (messages delivered to all nodes> 99% of the time, based on a random build of the network).

Peering is performed by collecting information about suitable nodes through the Connect / Get peers sent to the network.

Then the Validator selects a random node and tries to connect to it.
If this is possible, the validator has reached of the minimum connection state and search process stops. I.e. if we are already connected to the peer-to-peer network, we are not looking for new connections ourselves, without necessary.

If the minimum connection has not yet been reached, then the validator continues to search candidates for connection and will try to connect to them.

Mesh network example where target peering is 4.

Network component continues a peer-to-peer search if number of connected nodes is less than specified minimum. Network component rejects peer attempts if number of connections is equal to or greater than maximum number of possible connections. Even if the maximum peer-to-peer connections are achieved, the network service should still accept and respond to a reasonable number of connections (for the purpose of assembling other node topologies, etc.).

Selecting the ZeroMQ protocol provides significant flexibility in both the available connection templates and the underlying transport layer capabilities (IPv4, IPv6, etc.)

Connection states:
- Not connected
- Connected - Connection is a prerequisite for peering.
- Peering - A bi-directional relationship that forms the base form for transmitting an application-level message (Gossip).

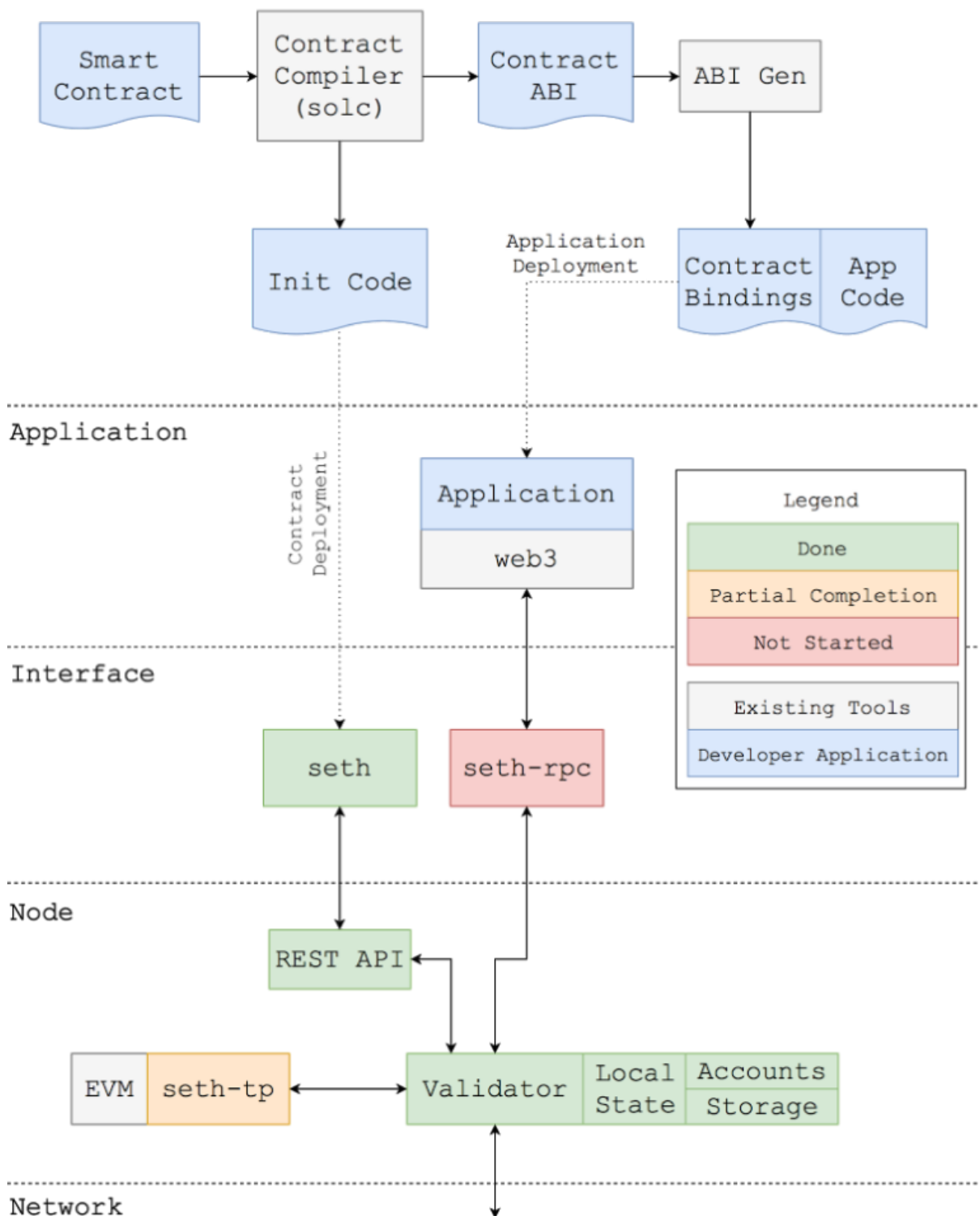All messages on network are serialized in protobuf

Smart contracts

The Smart Mesh node will work with EVM smart contracts thanks to hyperledger Seth. Seth family of transactions - Allows you to create and execute intellectual contracts. This transaction family integrates the Hyperledger Burrow implementation of the Ethereum virtual machine (EVM) into the Hyperledger Sawtooth platform using the Sawtooth Go SDK.

Contract and Application Development

```
Smart          Contract         Contract        ABI Gen
Contract  →    Compiler    →    ABI        →
               (solc)
```

Init Code

Application Deployment

Contract App
Bindings Code

Application

Application
web3

Legend

Done	(green)
Partial Completion	(orange)
Not Started	(red)
Existing Tools	(white/gray)
Developer Application	(blue)

Contract Deployment

Interface

seth   seth-rpc

Node

REST API

EVM | seth-tp ↔ Validator | Local State | Accounts Storage

Network

- Validator - Validator Sawtooth Service
- Seth-tp - The seth-tp transaction processor is a component that implements the functionality of "Ethereum-like" on the Sawtooth platform. Running Seth on the Sawtooth network is equivalent to connecting the seth-tp process to all the nodes of the validator.

- Seth - The client seth is a custom CLI tool for interacting with the Sawtooth network on which Seth is deployed.
- Seth-rpc - The seth-rpc server is an HTTP server that acts as an adapter between the Ethereum JSON RPC API and the client interface provided by Sawtooth.

Gas - in such a network, is free.

Consensus

The consensus protocol in Atmosphere is PoET (The Proof of Elapsed Time) is a lottery protocol that is created on a trusted execution environment to meet the need for large populations of participants.

At a high-level, PoET stochastically elects individual peers to execute requests at a given target rate. Individual peers sample an exponentially distributed random variable and wait for an amount of time. The peer with the smallest sample wins the election. Cheating is prevented through the use of a trusted execution environment, identity verification and blacklisting based on asymmetric key cryptography, and an additional set of election policies.

PoET essentially works as follows:
1. Each Validator requests a timeout from the enclave (trusted function).
2. The chosen leader is the Validator with the shortest waiting time for a particular transaction block.
3. One function, for example "CreateTimer", creates a timer for the transaction block, which is guaranteed to have been created by the enclave.
4. Another function, such as CheckTimer, checks that the timer was created by the enclave. If the timer has expired, this function creates an attestation that can be used to verify that the validator has actually waited for the allotted time before claiming a leadership role

The PoET leader selection algorithm meets the criteria of the lottery algorithm. He randomly distributes the election of leaders across the population of validators with a distribution similar to that provided by other lottery algorithms. The likelihood of an election is proportional to the resources provided (in this case, resources are general purpose processors with a reliable execution environment). The performance appraisal provides information to verify that the certificate was created in the enclave (and that the validator was waiting for the allotted time). In addition, low participation costs increase the likelihood that the population of validators will be large, increasing the reliability of the consensus algorithm.

Atmosphere blockchain

I researched the code base of several blockchain projects, which can be used for Atmosphere development and I settled on the Hyperledger Sawtooth framework.
Out of the box, we get:
- EVM (Ethereum Virtual Machine) to support Ethereum smart contracts.
- JSON-RPC Ethereum-like API through which the smartmesh application will interact with the Atmosphere.
- The ability to create customized transaction families relatively quickly.
- The ability to change the consensus in the already launched blockchain, if we want to add new opportunities after the launch of the blockchain.
- The ability to implement a customized p2p mechanism which will provide communication between nodes in the mesh network (relatively easily, in comparison with Ethereum and Bitcoin).

What we need for Atmosphere blockchain implementation:
1. Adapt the Atmosphere blockchain based on Hyperledger Sawtooth with the Android operating system (version of Android 5 and higher).
2. Add master nodes functionality to the consensus protocol to encourage honest master nodes or punish those who want to deceive the network.
3. Create an AAR-library for the Hyperledger Seth transaction family, using gomobile.
4. Develop a small JSON-RPC API for the Hyperledger Seth using Go and integrate it to the corresponding AAR-library.
5. Develop a new Hyperledger Seth transaction family for communication between Atmosphere nodes using the mesh protocol.
6. Develop a Plasma smart contracts for EVM.

Estimation to develop Atmosphere:
~ 650 - 800 hours

Plasma non hackable Exchanges

Facts:
- In Atmosphere direct calculations between network participants are impossible.
- All mutual settlements between network members are made in Plasma like smart contracts.
- Atmosphere does not pay for gas, all transactions are free.

The mechanism for synchronizing blocks between two block of chains is possible only by the master nodes (participants of two blockchains at once)

To become a master node in a directly smart contract you need to deposit on him and become an authorized operator of this contract, i.e. This user can confirm the blocks created by the participants outside the contract.

Any network member can present evidence of master node fraud with the help of a certain endpoint in the contract, then the master of the node is deprived of his deposit in the contract of plasma. it is frozen forever.

To be able to exchange values in the Atmosphere, these values must be present in the Mesh blockchain system and must be deposited into a specific plasma contract.

We have not decided yet - how we will reward the master node for their labors. I think this will be a small fee described in a smart contract.

We will have to significantly expand functionality of Plasma to achieve our goals.