ChainsAtlas: A Multi-Chain Interoperability Network

Ayoub Ben Chaliah, Jan Hanken, Marcos Medeiros

ChainsAtlas Labs
info@chainsatlas.com

August 15, 2022

Abstract

The concept of decentralized applications has revolutionized the crypto-world and has created the basic grounds for Web3. It is not that the current number of nodes and their combined computational power across different smart contract implementations wouldn't be sufficient to provide the foundation to lift web3 into a next phase of adoption. It is rather that power isn't provided in a way that would enable it to be used in a globally efficient approach. The main issue is that each of the largely adopted implementations is still, to a large extent, isolated, thus, impeding the possibility of combining the processing power behind each blockchain network. ChainsAtlas is a new blockchain technology that leverages virtualization to create a decentralized homogeneous environment across different smart contract implementations.

Note: ChainsAtlas is a work in progress. New updates will be posted on the website: https://chainsatlas.com

Introduction

ChainsAtlas is a new blockchain technology that leverages virtualization and multiple distributed ledger technologies with a smart contract feature (for the sake of simplicity in this document often referred to as blockchains) to create a cross-chain decentralized homogeneous network for distributed software and data processing. The combined infrastructure that powers existing blockchains allows for the possibility of running more complicated software in a decentralized manner, and with it a more reliable *Web3* infrastructure. ChainsAtlas is a combination of decentralized components, which are categorized as either nodes or units. The latter, in essence, are smart contracts running on several blockchains that, among other features, have a virtualization component. The virtualization component simulates a specific instruction set, allowing the same software to run on different chains regardless of the underlying smart contract implementation. Further, the virtualization approach allows for much of existing software that wasn't initially developed under a blockchain context, to run on a decentralized network just like a smart contract. Thus, bringing a massive amount of existing software to the blockchain ecosystem.

ChainsAtlas use a dual consensus protocol, *Proof-of-Stake* (PoS) and a novel class of useful *Proof-of-Work* (PoW), called *Proof-of-Strategy* (PoSg). In the context of PoSg, strategists provide the optimal execution strategy for a given bytecode under a variable set of constraints, and they operate nodes called Pythians that validate those strategies. With PoS, Pythians enable cross-chains communication and provide constant updates on the supported blockchains state.

Motivation: We, as the authors, strongly believe that Bitcoin and, by extension, blockchain technology have shown an unprecedented potential to make real positive change in society by reducing inequalities and opening opportunities to people around the world. Growing up in different parts of the world with different experiences, we all could agree on the importance of furthering this potential.

Contents

1 Network components overview

- **Athena**: The governance token that gives voting rights and staking rewards in Hermes (utility token).
- **Hermes**: The utility token generated by the staking of Athena and used to pay for smart contract execution on the network.
- **Client**: The user who wants to execute software under a set of preferences and conditions in exchange for an amount of Hermes.
- **Strategist**: The entity that supplies the management unit with an execution strategy that matches the clients' settings while minimizing the processing cost in exchange for Hermes.
- **Pythia**: A node operated by a strategist. It enables cross-chain communication between various units and evaluates and oversees strategy execution. To run a Pythia node, one must stake Athena.

  Note: The term "Pythians" refers to a collection of Pythia.
- **Management unit**: The smart contract that manages Hermes and Athena balances, verifies consensus over the optimal strategy, oversees elections and controls the virtualization units.
- **Virtualization unit**: Smart contracts that handle the execution of the client's bytecode. They are deployed on various blockchains.
- **Proof-of-Strategy** : A novel useful *Proof-of-Work* where the strategist (the Proof-of-Strategy equivalent of the miner) earns Hermes by providing the optimal strategy for executing the client's software under a set of constraints.
- **Chain profile**: A set of characteristics that define each of the supported blockchains in the network in a standardized manner.
- **Election cycle**: A time-limited event for Pythians to vote and update the chain profiles of supported blockchains. All non-measurable characteristics of a given chain profile can be updated (such as the decentralization score). Pythians can also vote to exclude existing blockchains or to include new ones in the network.
- **Agora DAO**: A decentralized autonomous organization that regulates the ChainsAtlas ecosystem based on the voting of its members. Every Athena staker, e.g., Pythia, is a member of the Agora DAO and has a right to vote.
- **OLYMPUS treasury**: A reward pool that collects part of the Hermes paid by the client to distribute among Pythians and the grant fund at the end of a 28 days period. Each Pythia's reliability score dictates its share of the OLYMPUS treasury. The use of the grant fund is decided through the Agora DAO, in which all Pythians have a vote.
- **Gerousia**: An assembly of developers that decides over final approval of changes to the code base under strategic goals set by the Agora DAO. The Gerousia is elected in regular intervals.

2 Chains characteristics

Each blockchain used in the network must have a smart contract feature and a "chain profile" and be Turing complete. The chain profile is a set of characteristics that define it from the point of view of the management units. These profiles serve as standardizing structure that allows for a unified approach to comparing the different blockchains of the network and validating the compatibility of a given strategy with a given underlying blockchain.

Further, these profiles allow clients to minimize the execution cost or the execution time by trusting the network's recommendations. For instance, the consensus of the network might recommend that blockchain A have the same privacy level as blockchain B at a lower cost. While the client can always choose the exact blockchains to be used to execute the transaction, this would allow the client the choice of exploring different blockchains based on the trust in the community consensus.
The chains' characteristics belong to one of the following two groups, (1) measurable features such as the number of transactions per second and (2) elected features such as a privacy level. The Election protocol (explained in the following section) will set and continuously update the latter's value.

Chains profiles			
Name	Blockchain A	Blockchain B	Blockchain C
Consensus Protocols	a_1	b_1	c_1
Number of validators	a_2	b_2	c_2
Transactions Per Second	a_3	b_3	c_3
Transaction size limit	a_4	b_4	c_4
Average transaction fee	a_5	b_5	c_5
...			

2.1 Characteristics: Measurable and Elected

There are two categories of chain characteristics: measurable and elected. The management units verify the consensus of Pythians before validating the recommendation of both types. The difference is that measurable ones are defined autonomously while elected ones are defined actively through voting.

Measurable characteristics are characteristics that can be represented with an ordinal variable and are measured by each Pythia. Examples of those include:

- Transaction size limit
- Number of validators
- Block time
- Market cap
- Network age
- Average transaction fee

Elected characteristics are the characteristics that are either subjective or those whose values can't be represented with an ordinal variable, and these are defined by the consensus of the stakeholders. Examples of those include:

- Decentralization score
- Privacy score
- Consensus protocol

3 Distributed Decentralized Processing Network

The Distributed Decentralized Processing Network (**DDPN**) leverages the best features of each blockchain implementation to execute software through management and virtualization units. These units are smart contracts deployed on multiple blockchains. The management units are responsible for managing transactions, wallets, and communication with virtualization units to guarantee the proper execution of the client software with an optimal strategy. The virtualization units create a distributed Runtime for software, including those not designed to run on a blockchain, by deploying smart contracts that simulate a specific machine instruction set. Further, as explained in more detail in the following sections, the **DDPN** allows users to (1) run code that surpasses the transaction size limit and (2) asynchronously execute bytecode via segmentation and execution transfer.

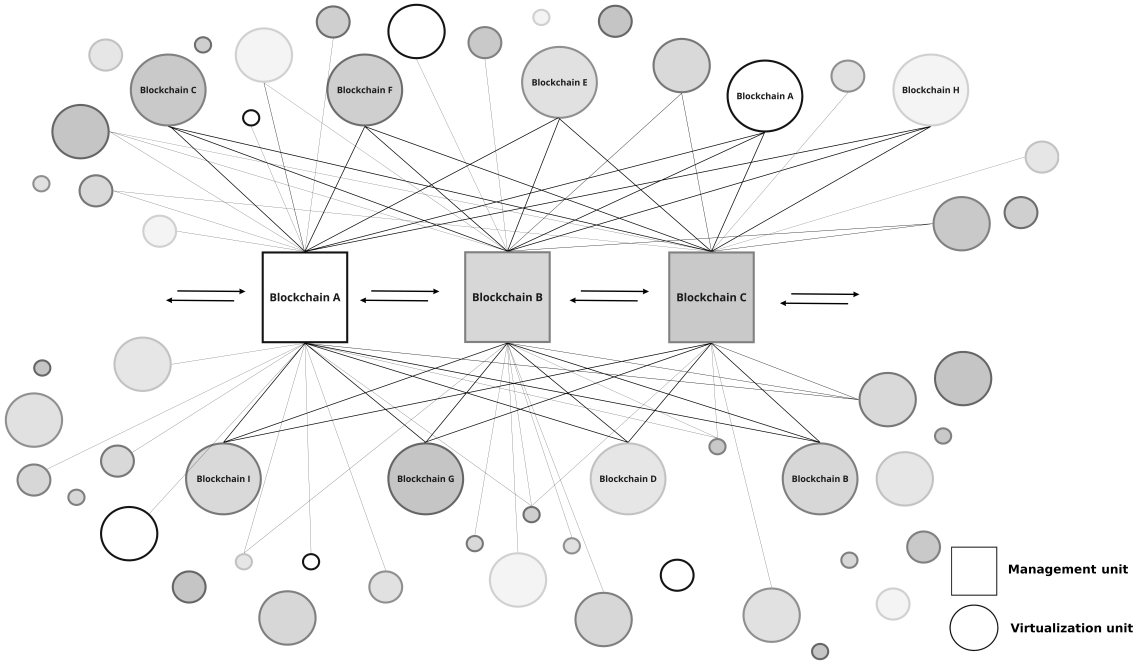


Figure 1: ChainsAtlas Network - Management and Virtualization units

3.1 Bytecode segmentation

The transaction size limit is one of the possible diverging characteristics among smart contract implementations. Considering virtualization units, this translates to a limitation of the size of the client's bytecode to be run on the blockchain. The **DDPN** addresses this limitation by segmenting the bytecode to run it within the limits of each virtualization unit's underlying blockchain.

Formulation of the segmentation:

Let $\mathcal{B}$ a bytecode, and $\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_d$ be an ordered segmentation of $\mathcal{B}$, such as:

$$\bigcup_{i=1}^{d} \mathcal{S}_i = \mathcal{B} \tag{1}$$

3.2 Execution transfer

The execution transfer process handles the bytecode segmentation. Each bytecode is considered a transaction, and splitting occurs when a given transaction size exceeds the underlying Blockchain limit. The management units send the resulting smaller transactions to the virtualization units that, in turn, dump the virtualization state (memory and registers) at the end of each bytecode segment $\mathcal{S}_i$ to be loaded in the next transaction.

- For applications with a high volatile memory footprint, the strategy must set the proper choices of blockchains — more on this in the Proof-of-Strategy section.

3.3 Synchronous and asynchronous execution

The execution of a bytecode that uses multiple transactions can occur in two manners, Synchronous or Asynchronous. The advantage of the latter, when possible, is that it requires less time for completion.

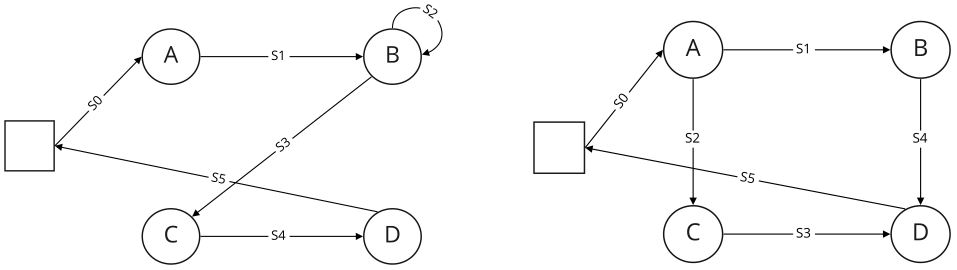
When the execution is synchronous, the bytecode segmentation is subject to:

$$\forall i, j \in \{1..d\}, i \neq j\colon\ \mathcal{S}_i \bigcap \mathcal{S}_j = \varnothing \tag{2}$$

Asynchronous execution occurs when a subset of the segmentation group contains segments that are not dependent on other segments. The bytecode and its segmentation method define the possibility of asynchronous execution. For instance, the strategist might define a segmentation where:

$$\exists i, j \in \{1..d\}, i \neq j\colon\ \mathcal{S}_i \bigcap \mathcal{S}_j \neq \varnothing \tag{3}$$

However, to ensure that a given segmentation is not allowing attacks based on vulnerabilities such as race conditions, all strategies will be validated by the Pythians (defined in the section below) without the proposed segmentation.



(a) Synchronous execution

(b) Asynchronous execution

Figure 2: Examples of sync v. async execution patterns

3.4 Pythians

Pythians are responsible for communications between the virtualization and management units. Before strategy acceptance by the network, a set of Pythians validate the strategy. Three criteria regulate the selection of these Pythians:

Factors	Weight
Stake	40%
Randomised selection	20%
Validation reliability score	40%

Management units set the validation reliability score by accepting or ignoring the Pythians' evaluations of the supplied strategies. Each Pythia's initial score is 0. Management units update the Pythia score when the Pythia's evaluation of a strategy (positive or negative) is the same as the one provided by at least 51% of the selected Pythians for the same strategy.

3.5 Strategists

Strategists are those who run Pythia nodes and supply the management units with valid execution strategies. They are the counterpart of the client. To win a client's offer, the strategist must submit its strategy for validation. The validation takes place in other Pythian nodes. Once approved and chosen as the optimal one among the competing strategies, the management unit executes the strategy. After the execution, the management unit rewards the strategist with Hermes.

4 Proof of Strategy

The network enables transactions between entities that include clients and strategists. The client provides the bytecode and a set of constraints, such as cost range and privacy. The strategist provides the optimal execution strategy that minimizes the execution time and cost while respecting the constraints set by the client.

4.1 Offer:

The client provides an offer that combines bytecode, a set of constraints and preferences, and the acceptable cost range. The last two we will be referred to as offer metadata. The constraints allow selecting the underlying blockchain on which the provided bytecode must run. This selection can be done directly by providing the IDs of the authorized blockchains or indirectly by setting conditions to filter the underlying blockchains, including, but not limited to: privacy score, number of validators, and number of transactions per second. In order to match the blockchains that meet these constraints, the management units use the results of the latest elections (see section 7).

Clients can also provide preferences (or soft constraints), such as the priority level given to latency versus cost, providing a value $\alpha \in [0, 1]$. If $\alpha \geq 0.5$, the network will prioritize a strategy that minimizes the execution time rather than cost as long as the total cost is within the cost range indicated in the offer.

Chained submission

If a given offer size surpasses the transaction size limit, the bytecode can then be supplied in chunks as long as the size of each chunk, except for the last one, is equal to the transaction size limit. This approach minimizes the number of necessary submissions to the management units.

In the case of a chained submission, the first submission holds the metadata and the first bytecode segment. Each subsequent submission must have the same size as the transaction size limit and contain (1) a bytecode segment and (2) a hash (SHA256) of the following submission. The last submission is an exception as it only holds the last segment, and its size can be less than the transaction size limit, as mentioned earlier.

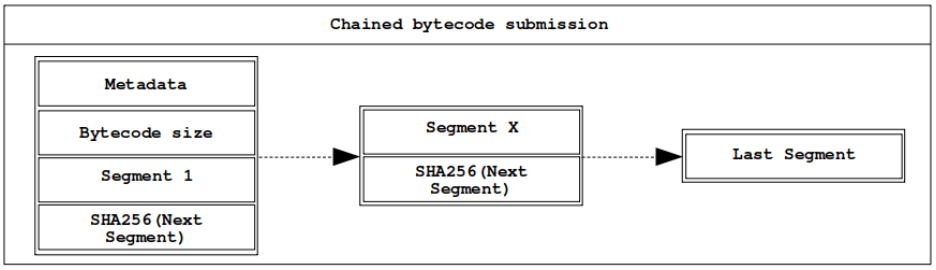


Figure 3: Chained offer submission base design

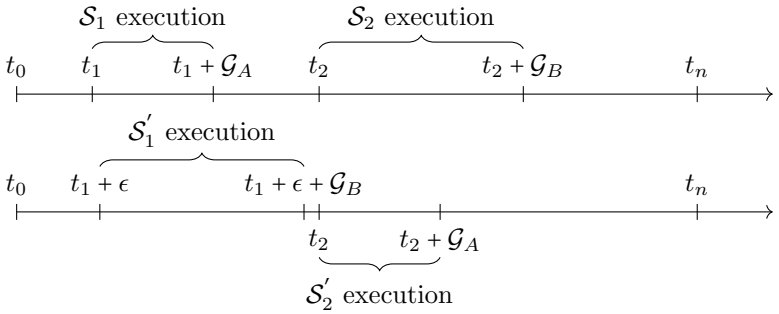
4.2 Strategy

The *Proof-of-Strategy* is a novel class of useful *Proof-of-Work* in which the reward (see section 11) goes to the first strategist to provide the optimal valid execution strategy that meets the constraints set by the client. This protocol incentivizes the strategists to provide the most cost-efficient strategy as the level of cost optimization directly impacts their reward while also incentivizing them to optimize in terms of execution time as it is the primary factor for selecting the optimal strategy. Often the strategist will use a stochastic optimization algorithm as this implies that the more iterations the strategist runs, the more likely this would result in a strategy with a higher profit margin. However, this also increases the risk of another strategist submitting their proposal earlier and effectively winning the bid.

4.2.1 Valid strategy

At epoch t of network $\mathcal{N}$, a strategy is valid if the requested resources (Virtualization and Management units) are within the intersection of resources that respect the client's constraints and the network's available resources. $\mathcal{R}_\mathcal{C} \bigcap \mathcal{R}_\mathcal{M}$.

Consider (1) the synchronous execution strategy $\mu_\mathcal{S}$ and (2) the bytecode $\mathcal{B}$ split into two ordered segments $(\mathcal{S}_1, \mathcal{S}_2)$ that run on virtualization units deployed on Blockchain A and B, respectively, and (3) the bytecode $\mathcal{B}^{'} = \mathcal{S}_1^{'} \bigcup \mathcal{S}_2^{'}$. Let us assume bytecode $\mathcal{B}$ execution starts at t_1, and its strategy is approved.



Let us define $\mu_\mathcal{B}$ as the strategy for bytecode $\mathcal{B}$, $\mu_{\mathcal{B}'}$ as the strategy for bytecode $\mathcal{B}'$, and $\mathcal{G}_X$ as the estimated time for the execution of a given segment or bytecode on a virtualization unit running on top of blockchain X.

$$\mu_\mathcal{B} \bigcap \mu_{\mathcal{B}'} = [t_1 + \epsilon, t_1 + \mathcal{G}_A] \bigcup [t_2, t_2 + \mathcal{G}_B]$$

strategy $\mu_{\mathcal{B}'}$ is valid if it verifies:

$$\forall t \in [t_1 + \epsilon, t_1 + \mathcal{G}_A] \bigcup [t_2, t_2 + \mathcal{G}_B]; \exists R_t^X \in \mathcal{R}_\mathcal{C} \bigcap \mathcal{R}_\mathcal{M} \text{ and } R_t^X \neq \varnothing$$

In other words, for each epoch t, R_t^X, the ensemble of virtualization units that run on blockchain X during t is not a null set.

Estimated execution time

The definition of an estimated execution time $\mathcal{G}_X$ for executing a given segment (or entire bytecode) $\mathcal{S}$ on a unit running on blockchain X is:

$$\mathcal{G}_X = \begin{cases} \alpha_X X_t^S + (1 - \alpha_X) X_{t-1}^S & : t > t_0 \\ X_{t_0}^S & : t = t_0 \end{cases} \tag{4}$$

X_t^S is the number of transactions per second in blockchain X at t, t_0 is the first epoch in the network $\mathcal{N}$, and α_X is the smoothing factor for blockchain X. α_X is part of the voted chain's characteristics and is subject to adjustments via elections.

Execution cost

The execution cost is the sum of the cost for the segment (or entire) bytecode $\mathcal{S}$ execution on a given blockchain and the variable transfer cost that depends on the cross-chain communication required to execute a given strategy.

Assuming (1) a strategy that uses three virtualization units running on top of blockchains (A, B, C) to run $\mathcal{B} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$, (2) a transfer cost matrix defined as $\mathcal{W} = (c_{ij})_{(i,j) \in M^2}$ with c_{ij} as the cost of execution transfer between a unit on the blockchain i to a unit on blockchain j, (3) $h_\mathcal{B}^X$ as the cost of executing bytecode $\mathcal{B}$ on a unit running on blockchain X and (4) c_ϵ as the management unit cost. The total execution cost definition is:

$$c_{AB} + c_{BC} + c_{CD} + h_{\mathcal{S}_1}^A + h_{\mathcal{S}_2}^B + h_{\mathcal{S}_3}^C + c_\epsilon$$

4.2.2 Optimal strategy

The optimal strategy is a valid strategy that first minimizes the execution time and second minimizes the execution cost respecting the client settings.

Assuming (1) an epoch t of Ledger $\mathcal{L}$ and (2) n valid strategy $(\mu_i)_{i\in 1..n}$, the optimal strategy $\mu_{\mathcal{O}}$ is determined as follows:

$$\alpha\frac{f_{\mathcal{L}_t}(\mu_{\sigma_{n-1}}) - f_{\mathcal{L}_t}(\mu_{\sigma_n})}{f_{\mathcal{L}_t}(\mu_{\sigma_n})} \geq (1-\alpha).\frac{g_{\mathcal{L}_t}(\mu_{\sigma_n}) - g_{\mathcal{L}_t}(\mu_{\sigma_n})}{g_{\mathcal{L}_t}(\mu_{\sigma_{n-1}})} \implies \mu_{\mathcal{O}} = \mu_{\sigma_n} \quad (5)$$
$$else\ \mu_{\mathcal{O}} = \mu_{\sigma_{n-1}}$$

Where:

- $(f_{\mathcal{L}_t}(\mu_i))_{i\in 1..n}$ is the estimated time for executing each valid strategy.
- $(g_{\mathcal{L}_t}(\mu_i))_{i\in 1..n}$ is the cost for executing each valid strategy.
- σ_i is a permutation that verifies $\forall i \in \mathbf{N} \cap [1, n-1]$: $f_{\mathcal{L}_t}(\mu_{\sigma_{i+1}}) \leq f_{\mathcal{L}_t}(\mu_{\sigma_i})$
- $\alpha \in [0, 1]$ is the time-cost trade-off coefficient and part of the client's offer.

Asynchronous execution optimization

Bytecode redundancy, described in equation (3), is not a requirement for asynchronous execution and is permitted only if the increased cost is within the parameters defined by the client. An execution with such segmentation must result in a time-cost trade-off.

Assuming $\mathcal{S}_{\sigma_s}$ to be a segmentation that verifies (1) and (2), $\mathcal{S}_{\sigma_a}$ to be a segmentation that verifies (1) and (3), $\mathcal{T}(\mathcal{S})$ and $\mathcal{C}(\mathcal{S})$ to be the time and the cost for the complete execution of $\mathcal{S}$ respectively, then asynchronous based execution is valid if:

$$\mathcal{T}_{\sigma_a} \leq \mathcal{T}_{\sigma_s} \quad (6)$$
$$\mathcal{C}_{\sigma_a} \leq \mathcal{O}_{max} \quad (7)$$

5 Management units

The management units are smart contracts deployed on all the supported blockchains, each holding the same state information. In order to maintain synchronization of state and the strategy order book, management units verify consensus among Pythians before validating requests to update its state or submit a valid strategy.

As the core management component of the network, management units hold multiple responsibilities:

- Provision and maintenance of wallets and balances for Hermes and Athena;
- Verification of bytecode segments' integrity through hash computation;
- Bytecode's strategy triggering;
- Management of swaps between Hermes and the required tokens for executing a strategy;
- Delivery of execution output or reference to the execution output location (depending on the offer type) to the client;
- Distribution of rewards (see section 11);
- Maintenance of the grant fund (see section 10).

6 Virtualization units

The virtualization units are smart contracts composed of three components:

- **Virtualization manager**: manages the execution flow of the unit and receives/transmits data as indicated in the joined metadata supplied by the management units;
- **Logger**: documents the transaction by computing hashes of the variable parts of the transaction and storing the results in the immutable memory of the underlying blockchain;
- **Simulator**: interprets and executes the bytecode.

The simulator is a 32bit machine simulator based on a modified version of the DLX instruction set, a RISC processor architecture. It has an ephemeral memory that includes a general-purpose memory and registers. Registers are usually a specialized data set with memory that may hold an instruction or a storage address. Additionally, some instructions in the bytecode specify registers as part of the instruction.

The simulator populates the general-purpose memory with the bytecode and data the virtualization manager provides. In sequence, it starts fetching instructions from the bytecode. Once the execution of the bytecode is complete, the virtualization manager resumes the smart contract's execution flow and forwards the harvested output from the simulation to the logger. After receiving the output, the logger documents the transaction by storing the computed hashes immutably in the blockchain.

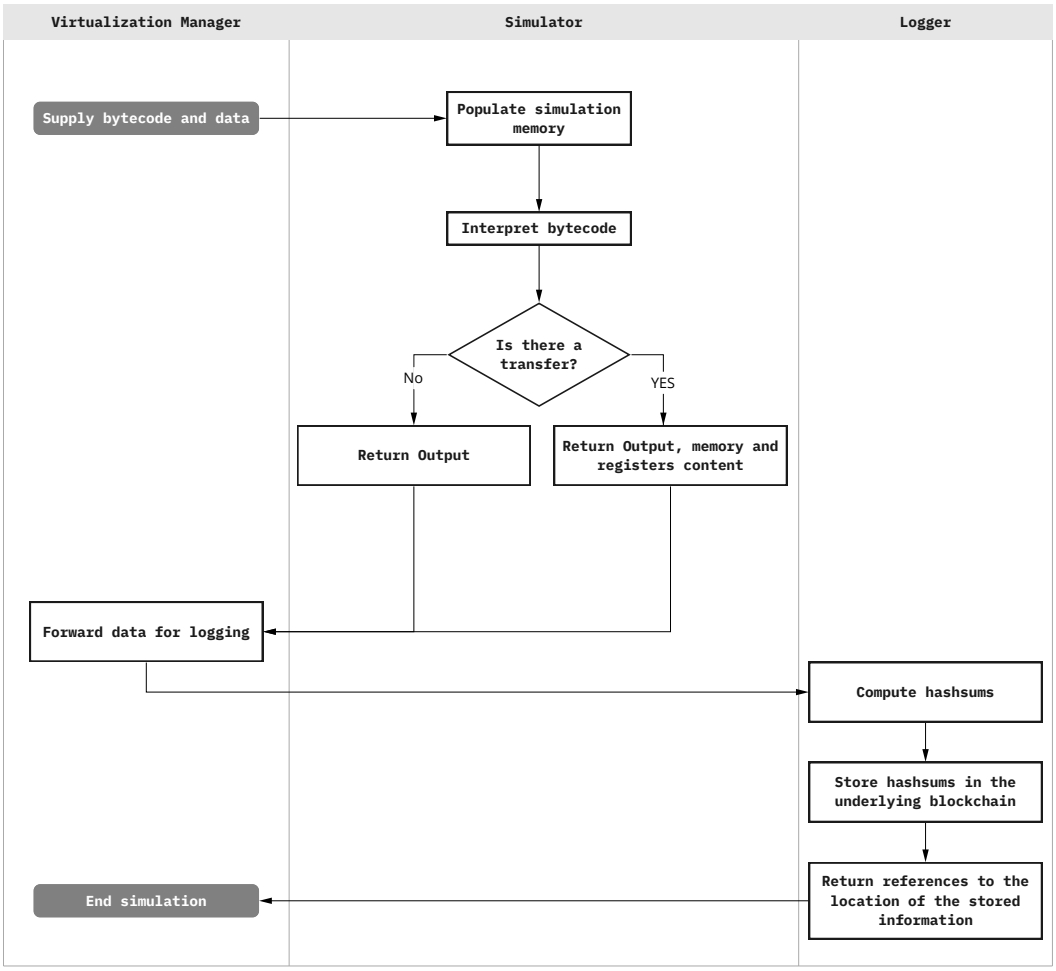


Figure 4: Base virtualization unit design

7 Operational protocol

7.1 Chains characteristics' election protocol

ChainsAtlas provides the necessary technology and procedures to constantly expand the number of blockchains supported or update the characteristics of the existing ones. In order to submit a new profile or update an existing blockchain profile, one must send the management unit a request for an election in which eligible Pythians will vote to validate the proposed profile, determining how much and under which constraints it will contribute to the network.

To approve chains profiles, ChainsAtlas uses a three-step election process:

1. Only strategists can trigger an election by 10 % of staked Athena (with delegated Athena added to the total amount of the Pythia node it is staked with) having to call for an election within a 12 hours window which starts an election process (3 days);
2. Once the election process starts, each Pythia is assigned a new weight based on stake, randomized voting, reliability score, and previous nomination. The voting takes place in the 3 days window of the election process;
3. At the end of the election process, management units update the chain profile with the new consensus and update the Pythians' reliability score.

Note: The voting does not update measurable characteristics since the Pythians in the network will continuously measure these features, and the management units will adopt the updates based on a consensus protocol.

Voting eligibility

To be eligible to participate in the election protocol, one must have Athena staked on a Pythia node, and the stake cannot be in the On-Ramp or Off-Ramp (see section 8.2) phase to be valid. All valid Athena managed by the Pythia, including those delegated to it, count as voting units for the Pythia's address.

Elections consensus protocol

With each new smart contract implementation brought into the network, an election based on *Proof-of-Stake* will determine if the contract gets deployed or rejected. The network computes the hash sums (SHA256) for each implementation bytecode and, if approved, immutably stores it in the underlying blockchains, adding it to the ChainAtlas processing network.

Factors	Weight
Stake	45%
Uptime	15%
Reliability Score	30%
Randomised Voting	10%

Voting factors

Each new Pythia has a default reliability score value that updates at the end of each election cycle $\mathcal{E}$ based on the following criteria:

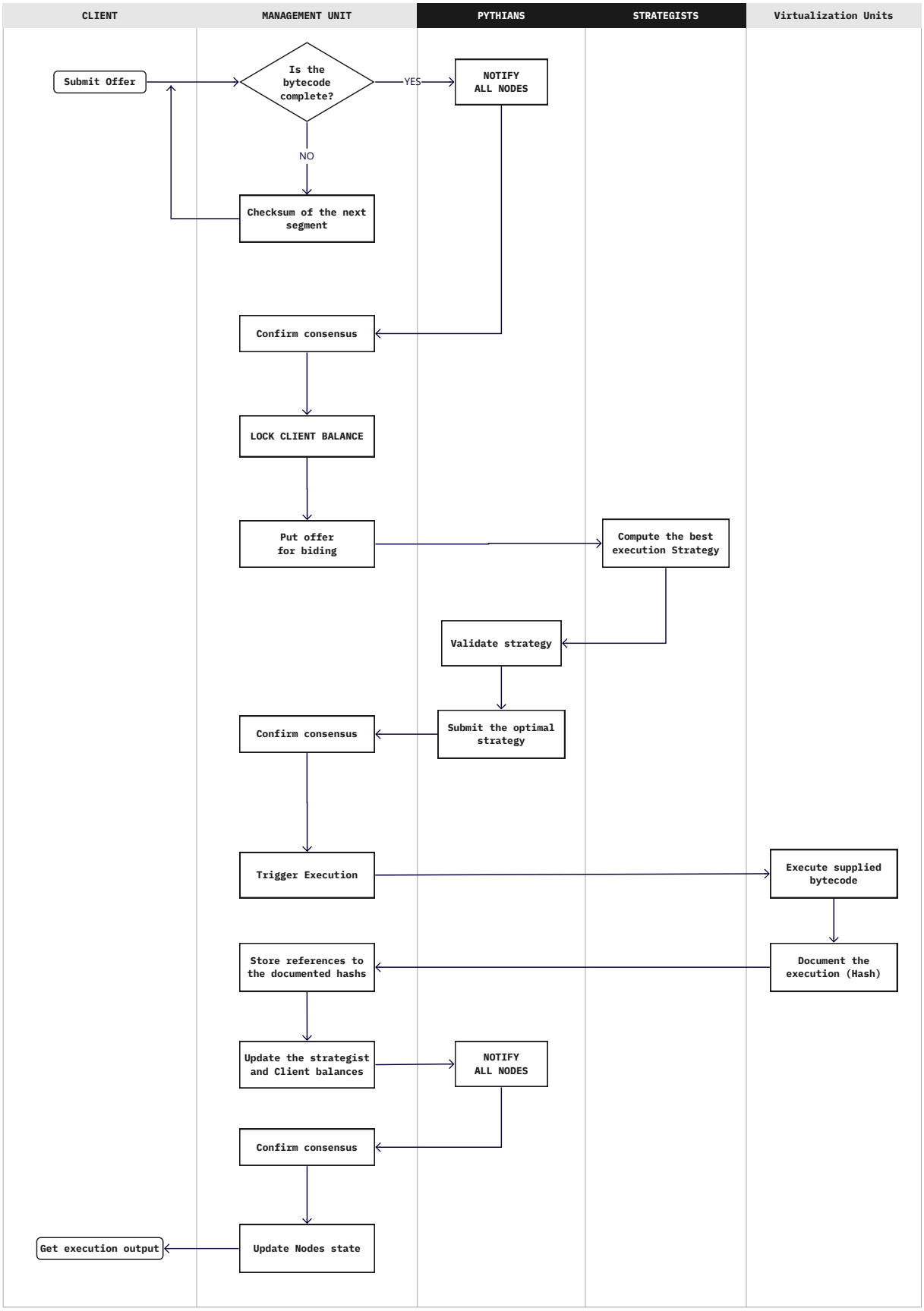
- **Stake**: determined by the amount of Athena staked on a specific Pythia node.
- **Uptime**: determined by the duration of time a Pythia node has been online during the last 7 days.
- **Reliability score**: determined by the euclidean distance between the Pythia's vote and the winning vote of the last election cycle. The closer the distance is, the higher the score is for the next election cycle.
- **Randomized voting**: impedes the Pythians' ability to collude and create biased voting, such as favoring or disfavoring a specific blockchain regardless of whether the blockchain profile reflects reality. For that reason, in randomized voting, some of the Pythians are selected through a pseudo-random algorithm.

Voting weights

The ChainsAtlas network is a work in progress. As such, these weights are subject to change based on the balance of the Pythia selection algorithm.

Pythians with a higher stake should get the higher priority, but stake should not be the only selection criteria. Such a scenario discourages the participation of Pythians with smaller stakes or who cannot coerce others for self-nomination. As a result, the network should also reward the loyalty of Pythians that are reliable and active for extended periods.

7.2 Offer Protocol sketch



8 Tokenomics

8.1 Dual Token System

ChainsAtlas, in essence, represents a coordinating entity for the operation of computation on independent external blockchains, which creates several implications for the interaction of its coin structure with other coins native to those chains where the computation takes place. This interaction settles cost of execution on the external blockchain, among other utilities. It is not practical in terms of rapid adaption, usability, predictability, and efficient operations of ChainsAtlas to use the same token for staking and governance as for day-to-day operational needs.

The network tokenomics is separated into two intrinsically connected coins to protect its governance structure from influence induced by interaction with tokenomics of external blockchains:

- **Athena:** A governance token that, when staked, grants voting power and rewards in Hermes;
- **Hermes:** A utility token used to pay for cost of execution and receive rewards through operating Pythians, staking, or delegating Athena.

The dual token system differentiates between a token solely used for governance and another one fulfilling the purpose of providing operational utility on the chain and interacting with underlying blockchains.

8.1.1 The Athena governance token

ChainsAtlas' governance token is called Athena. It is required to have Athena fully staked in a Pythia node to acquire voting rights for the staking address (more about staking under 8.2). Athena has a fixed supply of 100M, and it cannot be minted or burned outside of a governance decision that requires a 67 percent majority of staked Athena in a vote with a strategists quorum of 67 percent (more about governance votes on ChainsAtlas under 9).

8.1.2 The Hermes utility token

ChainsAtlas' utility token is named Hermes and mainly serves to settle cost of execution on ChainsAtlas and connected chains. It is also a reward to strategists and other staking addresses for their participation in securing ChainsAtlas' protocol. Hermes will be minted by staking Athena (as described under 11.1) and burned when used on other blockchains (as described under 11.2). The initial supply is 1B.

8.2 Staking and Stake Delegation

There are two main consensus mechanisms deployed to secure the ChainsAtlas network. For one, a novel class of useful *Proof-of-Work* called *Proof-of-Strategy* fulfills a fundamental task in the operational protocol of ChainsAtlas. Furthermore, a delegated *Proof-of-Stake* is used, which serves a critical role in the governance of the network and securing the built-in Oracle protocol that is essential to keep ChainsAtlas' network fundamentally decentralized.

- **Directly**: the strategist stakes a minimum of 100 Athena tokens and collects all the Hermes rewards.
- **Delegation**: the Athena owner delegates its tokens to an existing Pythia node. In this case, the Pythia can charge a fee of up to 10% of the rewards associated with the delegated Athena, and the remaining reward goes to the Athena owner's address.

All stakes are locked on-chain in smart contracts distributed throughout the ChainsAtlas network of on-chain management units, and the Pythia has no direct control over them at any moment. There are two mechanisms involved in the staking process:

- **The On-Ramp:** The locking process is called on-ramp and takes 28 days in which access to tokens is impossible, and they do not grant voting rights. The token owner can stop the on-ramp process at any time, and the token will be returned to the owner's address, in which case there will be no reward distribution and no voting rights. Should the token owner decide to start the on-ramp process again, the 28 days period will have to be started from 0 again.

- **The Off-Ramp:** Another 28 days period applies when removing tokens from staking. This process cannot be interrupted; the token remains locked for the whole period but does not yield any rewards. The token is available to its owner once the off-ramp process is complete.

9 Governance Protocol

9.1 Agora DAO

Agora DAO is the public space for the ChainsAtlas community to engage in relevant decisions of the ecosystem, including but not limited to transaction fees calculation, voting factors weights, codebase updates, and grant fund disbursement.

To participate in the Agora DAO, one must have Athena fully staked with a Pythia node either as a strategist or through delegation (see section 8.2).

The Agora DAO holds regular voting sessions for matters such as Grant fund disbursement (see section 10) and extraordinary voting sessions.

Regular voting session protocol

Regular voting sessions are held every 28 days. Here, new proposals regarding the following matters will be decided upon as long as they have been approved by at least 10% of staked Athena within a 14 days period:

- Petitions to enhance the list of decisions for this voting round.
- Development grant proposals.
- Petitions to add or remove strategic development areas.

Regular voting sessions last for a period of 28 days, and every petition requires a majority vote of a quorum of 20% of staked Athena and 20% of strategists.

Extraordinary voting session protocol

An extraordinary voting session can be triggered at any time by either

- a 60% majority vote of the Gerousia
- a majority vote of strategists
- a 20% vote of all Athena staked

with a voting period of 24 hours each.

Note: This session is meant for extraordinary situations and shall not be used to discuss items for a regular voting session without due cause.

Extraordinary voting sessions last for a period of 3 days, and every petition requires a majority vote of a quorum of 20% of staked Athena and 20% of strategists.

9.2 The Gerousia

The Gerousia is the development governing committee, and its members are selected based on the results of an election. These elections are held every six months. The process comprises two stages and lasts for two weeks, in which stakeholders can vote for a committee consisting of 5 stakeholders. In the first week, the 20 stakeholders who received the most votes are selected (each vote is multiplied by the number of Athena the voter holds). In the second week, 5 of the 20 named stakeholders will be selected to form the new Development Committee for the next 6 months. The committee oversees new development proposals and may approve or reject pull requests. For the election to be valid, at least 20% of staked Athena with at least 20% of strategists must participate. If the minimum turnout is not met, the committee's mandate will be extended for an additional 3 months, during which time the

Assembly will reconvene. The committee needs a majority to approve new updates. Every decision by the Gerousia has to be announced 1 day prior to its execution and can be vetoed through a majority decision with a quorum of 30% of staked Athena and at least 50% of strategists. The Gerousia only has the power to decide on proposals within the realm of strategic areas of development decided upon by an Agora vote as well as those targeting operational matters within the existing functionality of the protocol such as bug fixes or minor improvements.

10 OLYMPUS treasury

The OLYMPUS treasury comes to strengthen the ecosystem in multiple ways. It is a path to obtaining and maintaining decentralization, transparency, and scalability. The treasury distributes its collected Hermes among the Pythians fund (75%), the Grant fund (15%), and the Agora fund (10%).

Pythian fund (75%)

The Pythian fund intends to reward reliable and active Pythians for their loyalty, promoting:

- **Decentralization**: Pythians gain more upside for staying active and reliable besides Athena staking alone.
- **Transparency**: Pythians are oracles that ensure transparency and state synchronization among management units. Increasing the number of Pythians also increases transparency.
- **Scalability**: The distributed nature of the entire network allow for high parallel scalability

Grant fund (15%)

The Grant fund's objective is to foster decentralization and growth of the ecosystem with new and exciting projects chosen by the ChainsAtlas community through voting sessions in the Agora DAO.

Agora fund (10%)

As with any network infrastructure, ChainsAtlas has operational costs and is subject to contingencies. The Agora fund's objective is to financially secure the network governance so it can prosper and be resilient even in adverse situations.

11 Reward distribution

The network can reward both Pythians and Strategists. Pythians receive Hermes rewards by staking Athena and remaining reliable and active on the network, and strategists receive Hermes rewards by providing optimal execution strategies for clients' bytecode.

11.1 Pythia reward system

To run a Pythia node, one must stake Athena to ensure commitment to the ecosystem. The Pythia node is an oracle that oversees the balances between each blockchain transaction to check if they are correct and that management units hold the correct state. They are also the only entities with voting rights. To participate in chain profile and other Agora DAO elections, one must operate a Pythia.

Pythia nodes mint Hermes at a ratio equivalent to a variable percentage of the staked Athena price. This percentage follows a real-time calculation based on average transactions per second (more under 11.2). Since Pythians are one of the network decentralization and transparency foundations, they have a share of the OLYMPUS treasury Pythian Fund that redistributes the fund's Hermes based on the Pythians' reliability score and stake.

11.2 Strategist reward system

Strategists are foundational to network scalability. More strategists supply more strategies, resulting in more bytecode execution potential.

Given the stochastic nature of the optimization algorithms used for strategy selection, every Pythia comes with a default optimization algorithm (DOA) that has a chance to provide valid strategies even if the node is not running on top-tier hardware. The DOA reduces the hardware performance impact on the Proof-of-Strategy protocol, increasing decentralization and opportunity.

To run a Pythian node, one must be a Strategist. Therefore, to further increase the likelihood of Strategist participation in the network, the OLYMPUS treasury distributes 75% of its funds to Pythians. This measure aims to incentivize Strategists without relying on the number of strategies submitted, as it would centralize rewards on those with top-tier computational power. Instead, the Pythians fund uses the Pythians uptime and reliability score as a weight factor for Hermes distribution, fostering decentralization and scalability simultaneously.

To run a smart contract on the network, the client must submit a bytecode with its constraints and preferences. The client pays the transaction fee with Hermes, and strategists compete with each other to submit a valid execution strategy first. Once a strategy is approved, Hermes is locked and transferred to the executing node to cover the cost of execution in the underlying blockchains. After execution, the Management unit awards the remaining amount of Hermes to the following list:

- burn and compensate for the selling pressure against other blockchain tokens.
- reward the Strategist of the contract.
- OLYMPUS treasury.

The exact distribution percentage to each recipient depends on the average amount of transactions at any given time and will be calculated and adjusted accordingly, along with the percentage of staking rewards.

11.3 Slashing

Slashing occurs when the strategist presents a strategy with an incorrect estimated execution time or incorrect execution cost. The allowable margin of error for the estimated execution time is 5%, and for the execution cost is 0.1%. If a strategy violates either of these conditions, it is considered an inconsistent strategy. With each inconsistent strategy within a window of 50 consecutive supplied strategies, the penalty acceleration factor s_i increases. The purpose of this process is to penalize the intentional submission of incorrect strategies and to accelerate the exclusion of those who might try to cheat the network to win the most bids.

$$\frac{\mathcal{A}}{2}\left(\frac{1}{1+e^{-\frac{|T_v - T_s|s}{min(T_s,T_v)}}} + \frac{1}{1+e^{-\frac{|O_v - O_s|s}{min(O_s,O_v)}}}\right) \tag{8}$$

Where:

- $\mathcal{A}$ is the median amount of Athena held by strategists
- s_i is the number of inconsistent strategies out of the last 50 strategies
- T_v, O_v are the Execution time and cost determined by the validators respectively
- T_s, O_s are the Execution time and cost claimed by a given strategy respectively