ClinTex

CLINICAL TRIALS INTELLIGENCE

CTi Platform Technical Paper

Contents

General Information
Solution Description
CTi Platform Applications: Blockchain Environment Integration
CTi Platform: Application Operations (Blockchain Environment)
Classes, Roles and Characteristics of Users
Decentralised Components of CTi (By Timeline of Integration)
Stage 1 - Off-Chain CTi Platform Powered by Clx Payments with Consortium Chain Storage Verification of Data
Stage 2 - Off-Chain CTi Platform with On-Chain Data Storage with Storj File System
Stage 3 - On-Chain CTi Platform with Storj
Contracts
CLX Contract
Consortium Proof of Authority Contract
Smart Contract Permissions and Rules
CTi Platform Internal Token
Security
Storage
CTi File System with Root Encryption
Oracles + Nodes

General Information High Level Overview

ClinTex CTi is a distributed ledger technology platform for the clinical trials industry. It uniquely provides an end to end decentralised solution that incorporates tools to drive significant quality and operational improvements in running clinical trials through advanced clinical data review, insightful operational KPIs, predictive data analytics and machine learning (AI). The CTi platform links clinical trial sponsors together in a common ecosystem, enabling real-time access to information and the ability to share knowledge more easily. Improved efficiencies in clinical trial management are achieved by:

- Incorporating Machine Learning into clinical trials management and oversight. This is enabled by capitalising on the immutable nature of a blockchain ledger where the full history of clinical trial key metrics are stored and will be used to power bespoke predictive analytics algorithms.
- Being a fully integrated platform that allows auditable workflow management/oversight in clinical trials. This is enabled by having preventative and corrective actions (CAPA) recorded immutably on the blockchain.
- The creation of a clinical eco-system that enables interoperability and safe storage of all clinical trial data views. Here blockchain technology is employed as the optimum solution to address the privacy and security concerns of all stakeholders.

Through the CTi platform ecosystem, seven separate decentralised applications will provide valuable clinical insight that leverages the use of predictive analytical tools and machine learning (AI) for the benefit of Clinical Project Managers, Clinical Trial Physicians, Clinical Data Managers, Bio-statisticians, Pharmacovigilance, and Data/Clinical Monitors.

The CTi platform distinguishes itself from other tools used in clinical trials by:

- Providing the first ever collaboration platform for clinical trials
- Exploiting the immutability and interoperability of distributed ledger technology to create an ecosystem that fosters collaboration across the entire pharmaceutical industry. This will be achieved through the creation of a perpetually increasing library of data analytics, facilitating the sharing of “lessons learned” across corporate boundaries without any compromise of sensitive data
- Bringing Machine Learning to clinical trial management
- Eliminating the need for hardware costs to be borne by the client
- Introducing an attractive pay-per-use model for clients
- Applying powerful and insightful data analytics functionality across administrative, operational and clinical functions in clinical trials
- Allowing for workflow management, “closing the loop,” and full audit-trail functionality to identify, action and resolve issues detected by the tool.

Solution Description

Clinical Trials Intelligence Platform (CTi) contains a number of applications that focus on the key areas of waste and bottlenecks typically experienced during clinical trials, such as:

- Clinical trial monitoring
- Clinical data accuracy and availability
- Patient identification, recruitment and retention
- Investigator/physician recruitment including contracts and payments
- Supporting vendor data, including contracts and payments
- Clinical trial operations and workload/resource
- Medical review/interpretation of clinical data "as it happens"

Furthermore, a unique Clinical Data Predictive Analytics application further leverages blockchain technology to deliver unique insights that use machine learning to predict clinical and operational events. This can save pharmaceutical companies time, effort and cost by driving earlier and more accurate decision making.

Key tools of CTi platform:

- CTi-OEM Application
- The CTi-CDV Application
- The CTi-PDA Application
- The CTi-RBM Application
- The CTi-PRR Application
- The CTi-SIM Application
- The CTi-VMM Application

CTi Platform Applications: Blockchain Environment Integration

CTi runs a permissioned consortium blockchain based on the Ethereum network. This will allow CTi to avoid high fees for transactions of tokenised data.

Key features of using an Ethereum consortium blockchain:

1. Infrastructure of nodes controlled by Proof of Authority consensus, with extensions like Proof of Work, Proof of Stake/Reputation, & Delegated Voting.
2. Blockchain managed under the rules of Greedy Heaviest Observed Subtree (GHOST) protocol but without the strict requirements of Proof of Work mining.
3. Faster performance.
4. Operability with any Ethereum based DApp.
5. Dedicated Ethereum Virtual Machine with the option to amend the Gas[1] amount calculation.
6. No requirement to store the entire history from the public Ethereum genesis block, only from the genesis block of the consortium.

[1] Gas is a critical element to execute smart contracts and for prevention of DDoS or infinite loop attacks.

Function	Integration in blockchain	Tools for integration
CTi Data Model - collection of data	Smart contract and DApp to upload data to CTi DB(Storj)	RESTful JSON, Metamask, Infura, custom wallets
CTi Data Model - transfer of data to permissioned blockchain	Smart contract, Oracle node with Dapp	RESTful JSON, Metamask, Infura, payment gateway, Apache Kafka at Oracle level
CTi-OEM CTi-CDV CTi-PDA CTi-RBM CTi-PRR CTi-SIM CTi-VMM	Smart contracts, DApp, EVM(p)[2] ,Storj	RESTful JSON, Metamask, Infura, web3.js for front-end
CTi Processing Engine	Smart contract for data upload, DApp, Storj, Oracle	RESTful JSON, Metamask, Infura, gateway, Apache Kafka, Docker, web3.js

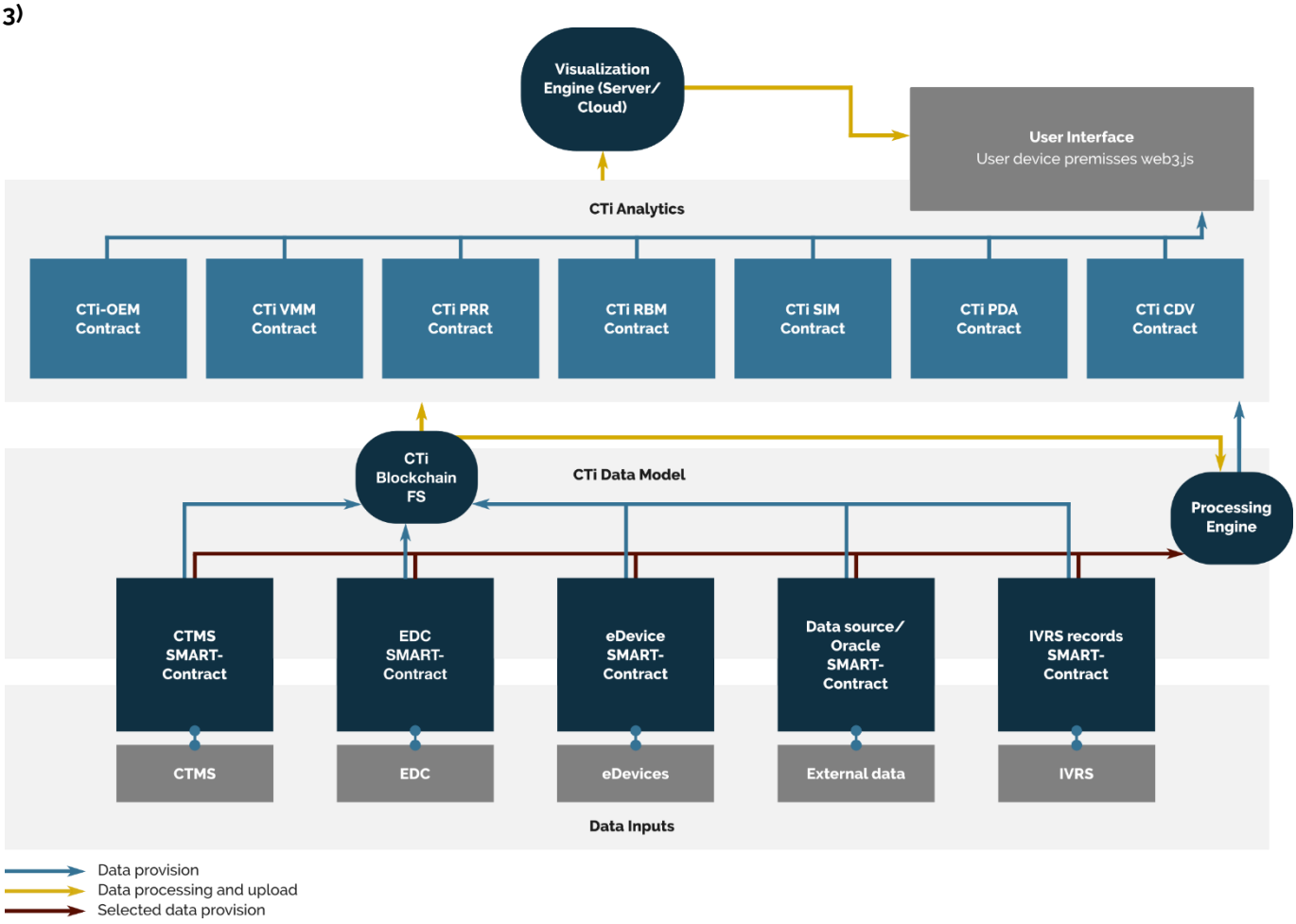
As the EVM(p) source code runs into some limitations when running fast computations in parallel, the CTi permissioned network will utilise specialised nodes that will take the clinical data from its decentralised database (Storj), use apache Kafka to map the specialised smart contract, and finally arrange the data for execution with Docker.

CTi Platform Applications: Blockchain Environment Integration

In the blockchain environment, all elements of the CTi platform are represented by their own smart contract with a dedicated DApp.

[2] EVM(p) - Ethereum Virtual Machine on permissioned nodes infrastructure.

Structure of Smart Contract for CTi Platform DApps: On-chain CTi Platform with Blockchain Automation (Phase 3)



The CTi GUI will use a web3.js based frontend to present the visualisations from each DApp, with Metamask/Infura connecting to each contract and addressing user commands to it directly.

A native data carrier token, CTX (ERC721), will allow external vendors to input data directly to the CTi file system in mutable form. The reward contract will provide remuneration for this data provision.

The connection contracts will have the additional function of automating the process of data uploads or smart-contract fabric arrangements. Criteria for the required data is sent with the carrier token to the platform via smart contract. A connection contract will listen to the transmissions by the data collection and platform contracts. If the data matches, the contract algorithms will trigger a smart contract to include the data in the analytic processing, or just set a connection between the requester and the provider directly.

To provide security of access and pseudo-anonymous identification of the input data, the data carrier token must have special fields of verification (token as a "container"). Platform tokens have another type of identification (with proof of ownership) that results in the need for 1 native data token (ERC721, as data containers) and 1 utility token (the ERC20 for public use: access and in-platform payments).

CTi will use tokens as a container to send information to the CTi (Storj) database under the distribution rules of the corresponding smart contract. Part of this information is provisioned directly to the processing engine, and the outputs will be delivered to each CTi module's smart contract.

The tokens that are used for data-input contracts and tokens that are used for CTi modules are different and require an additional accounting and data provision mechanism in their SMART contracts.
To send data directly to the CTi database (Storj), the container tokens are arranged through an integration with Docker, while the hash of that data (with roots) is written into the metadata of the token transaction.

Classes, Roles and Characteristics of Users

The user-facing DApps' GUI will interact with smart contracts that arrange the visualisation of the data from the blockchain based backend.

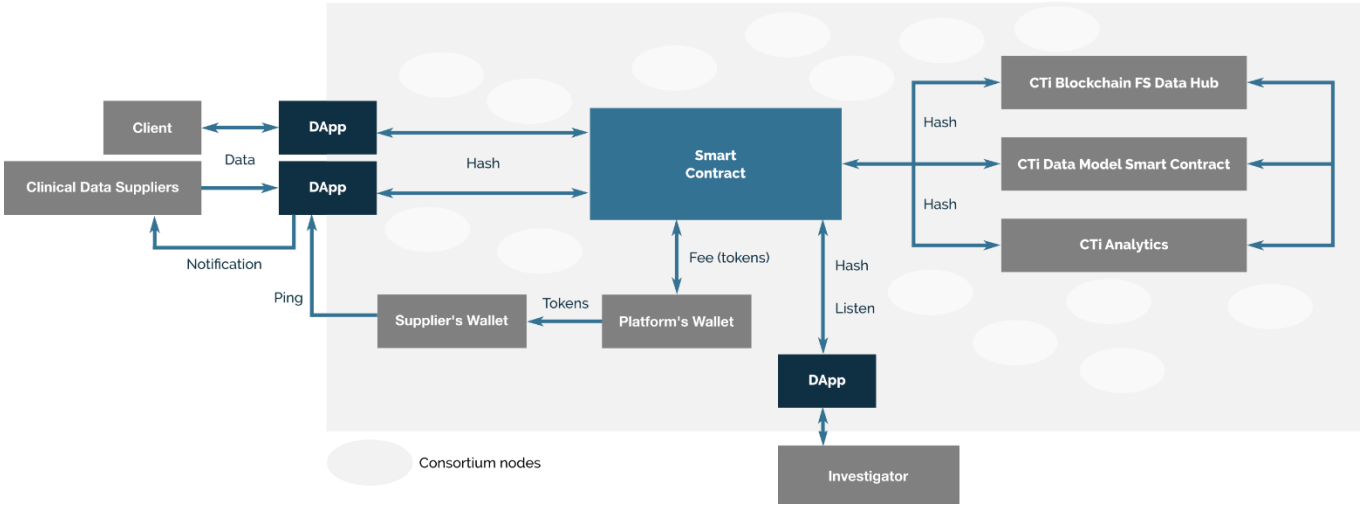
Key users and their characteristics:

1. Client - key user of data that can:
    a. Purchase an access license and establish a node in consortium
    b. View CTi analytics, visualisations, etc.
    c. Initiate a payment for data providers with a new smart contract

2. Third Party Vendors – clinical data owners that can:
    a. Provide data under agreement reflected in the smart contract
    b. Obtain reward payments

3. Investigator- clinical research physicians that can:
    a. Obtain reward payments for CTi data provided via source systems

Blockchain (Permissioned) Environment



Decentralised Components of CTi (By Timeline of Integration)

Stage 1 - CTi Platform (Off-chain) Powered by CLX Payments with Consortium Chain Storage Verification of Data

Structure data flow:

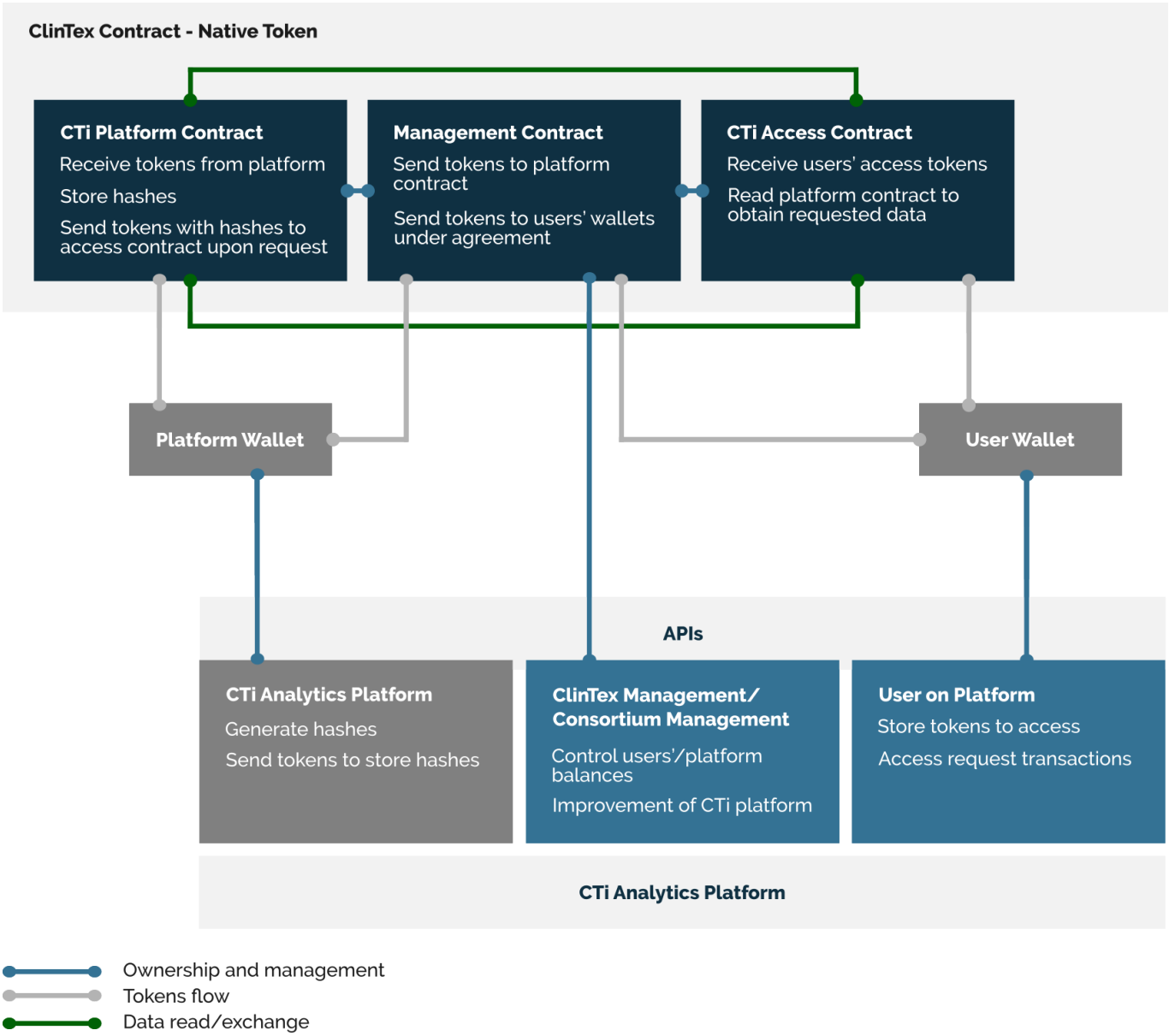
- Data is collected on CTi data model server/cloud and converted into a predefined standard, including a timestamp of all changes.
- The structured data is then sent to the CTi processing engine for algorithm processing.
- The output of processing engine will be indexed with hashes of the data timestamp. Mutable data output of the results are also hashed. Both hashes are concatenated into a single string and stored on the Ethereum blockchain. Indexing hashes will then be used for data-history searches.
- The CTi data output is sent to the CTi applications for representation to users.
- The user interface (GUI front-end) will have an integrated CLX (ERC20) wallet and will enable interaction between the user and the blockchain through a sequence of queries and system responses:
    - A smart contract will provide access to the applications by verifying proof of ownership of a predefined amount of CLX tokens in the user's platform wallet.
    - When the user sends queries (a request to the system for data), the CTi application will replicate the query to the blockchain (this is performed in the backend of the platform).
    - The CTi application verifies the cloud data's immutability through hashing. When a user performs a search or requests a view of any clinical data or visualisation, an internal search is performed for the corresponding hash on the blockchain. When the hash is found, the data is valid, and can be displayed to the user.

To obtain access to CTi applications, users will stake a CLX token balance (loaded into their integrated CTi wallet). Once verified, access will be provided under a service agreement license.

CTi cloud stored data will be accessed using a RESTful JSON API via encrypted channels.

Key Decentralised components at this stage:

1. The CLX (ERC20) public token will be the utilised for the purchase of the CTi access licence, with personal X.509 SSL digital certificate to encrypt the connection.
2. The CTi permissioned Ethereum consortium chain. This is the CTi blockchain under consortium agreement for PoA, utilising an internal (ERC721) token.
3. The management smart contract. This includes permissions to make changes in the consortium chain through predefined signatures.
4. The platform smart contract. This includes the rules of internal token flow, data hash storage, and distribution.
5. Access smart contact. This is the rule-set to match user signatures with permissions. This is used for data access and data hash verification.



ClinTex Smart Contract Structure (Phase 1)

Interaction of CTi's Decentralised Components:

1. Most components will run on the CTi permissioned Ethereum blockchain, with the exception of the CLX (ERC20) token which runs separately on Ethereum's permissionless blockchain.
2. The management contract will own the other contracts and include permissions to revoke (kill) them and/or deploy new versions of them.
3. The CTi Platform contract will actively listen to the access smart contract in order to obtain data hashes and provide them in the metadata of transactions from the user wallet to the platform wallet that is connected with the CTi processing engine.
4. The access contract will listen to Ethereum Mainnet to track user payments of CLX for the CTi license, and to ensure the validity of the users' certificate.

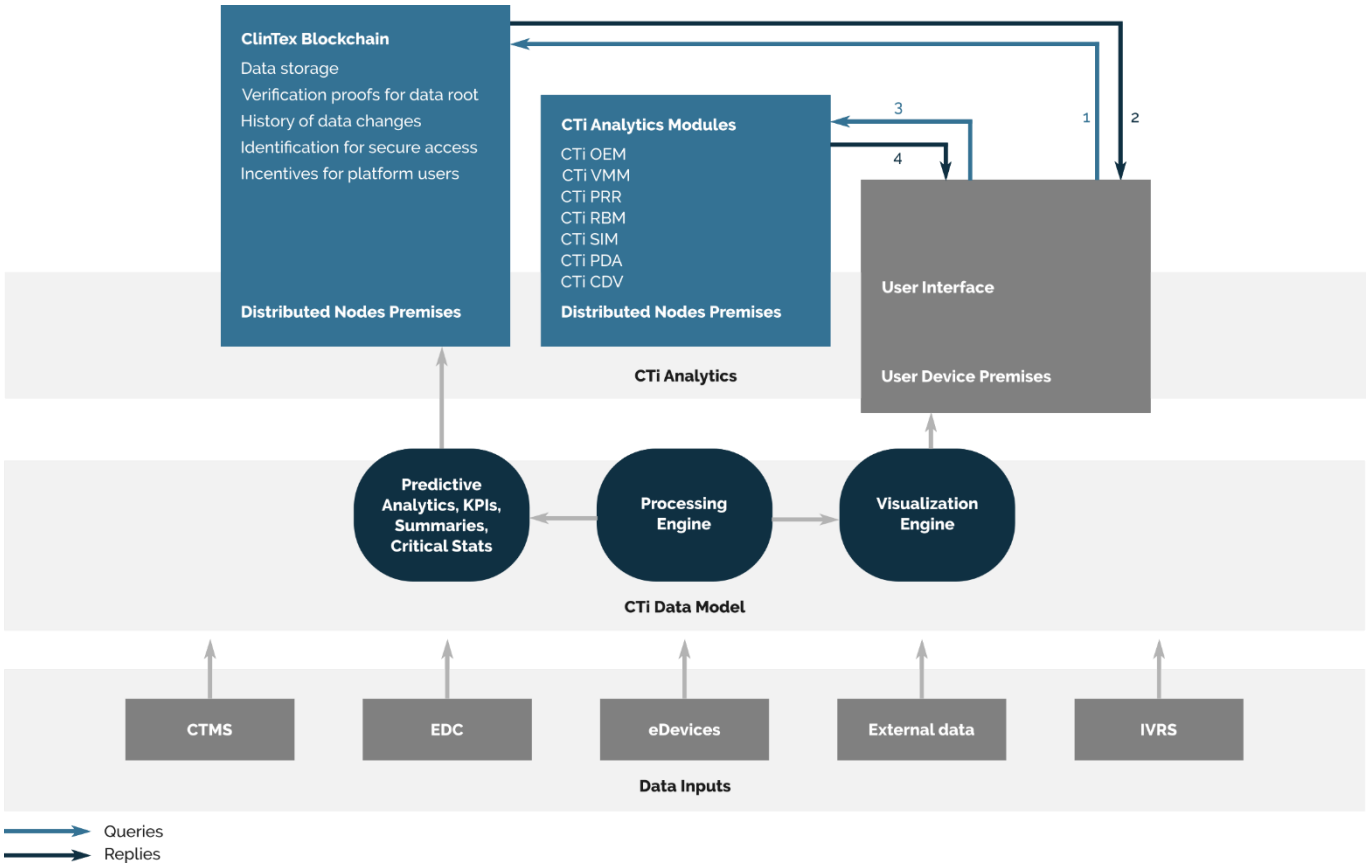
Stage 2 - CTi Platform (Off-Chain) with On-Chain Data Storage Using Storj File System

CTi will use Storj as its decentralised storage provider. Storj is an open source decentralised cloud storage platform which keeps data spread across a decentralised network, eliminating the problem of having a single point of failure. Storj also encrypts all data, making it impossible for anyone to gain access to users' files without possession of the corresponding private encryption key.

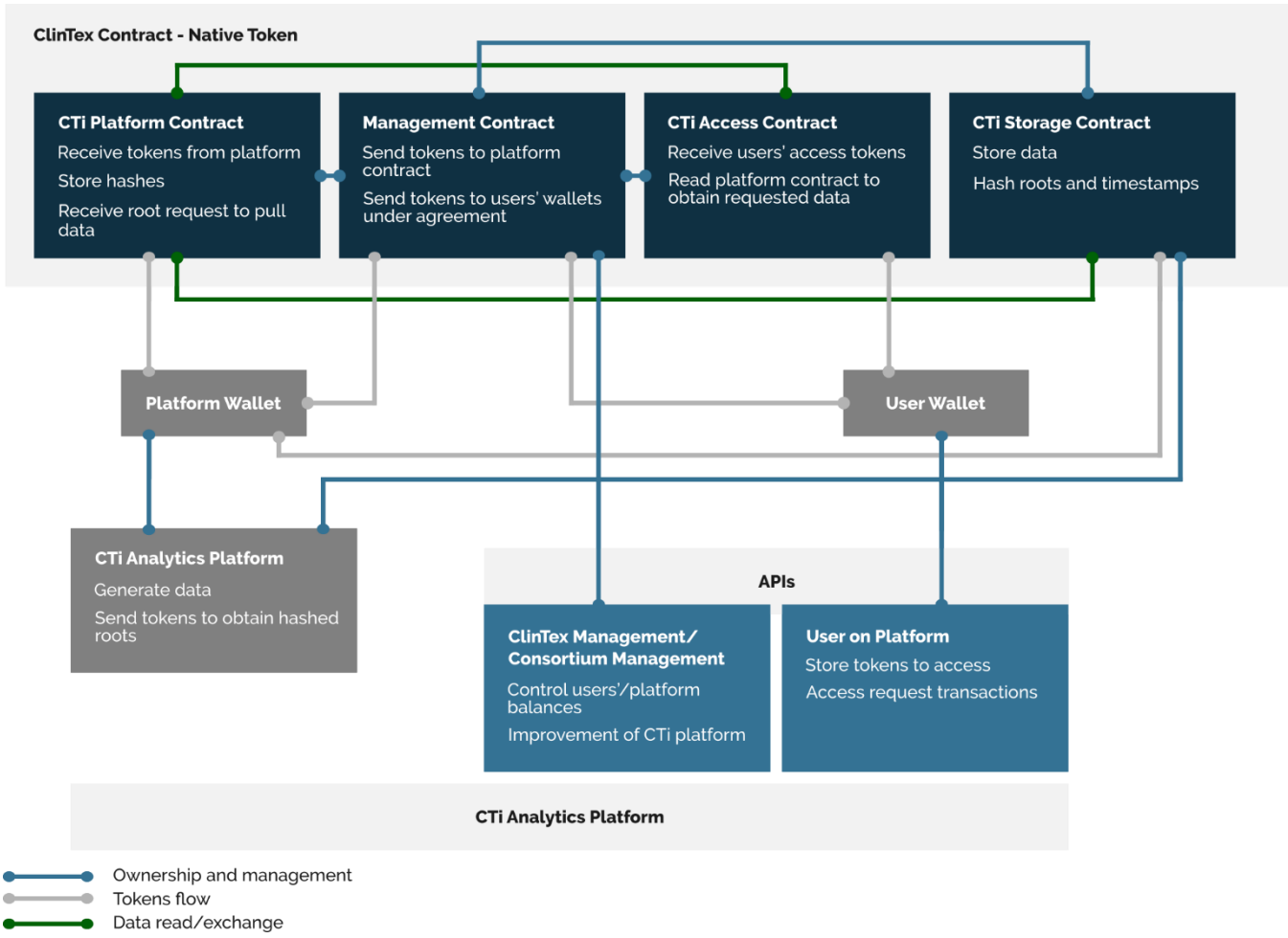
The introduction of the blockchain-based Storj file system will change the approach of how the blockchain and CTi Modules operate together.

Apart from the CLX token access wallet, users now have no need for direct blockchain interaction as every query can go through each CTi application's GUI front end.

CTi Platform Extension with Blockchain File System (Phase 2)



ClinTex Smart-Contract Structure (Phase 2)



The CTi consortium chain will now include an additional dedicated storage smart contract, to facilitate the integration of Storj within its permissioned blockchain.

The storage smart contract will be deployed with the management contract and will contain the rules for data upload and download to and from Storj via the CTi consortium chain.

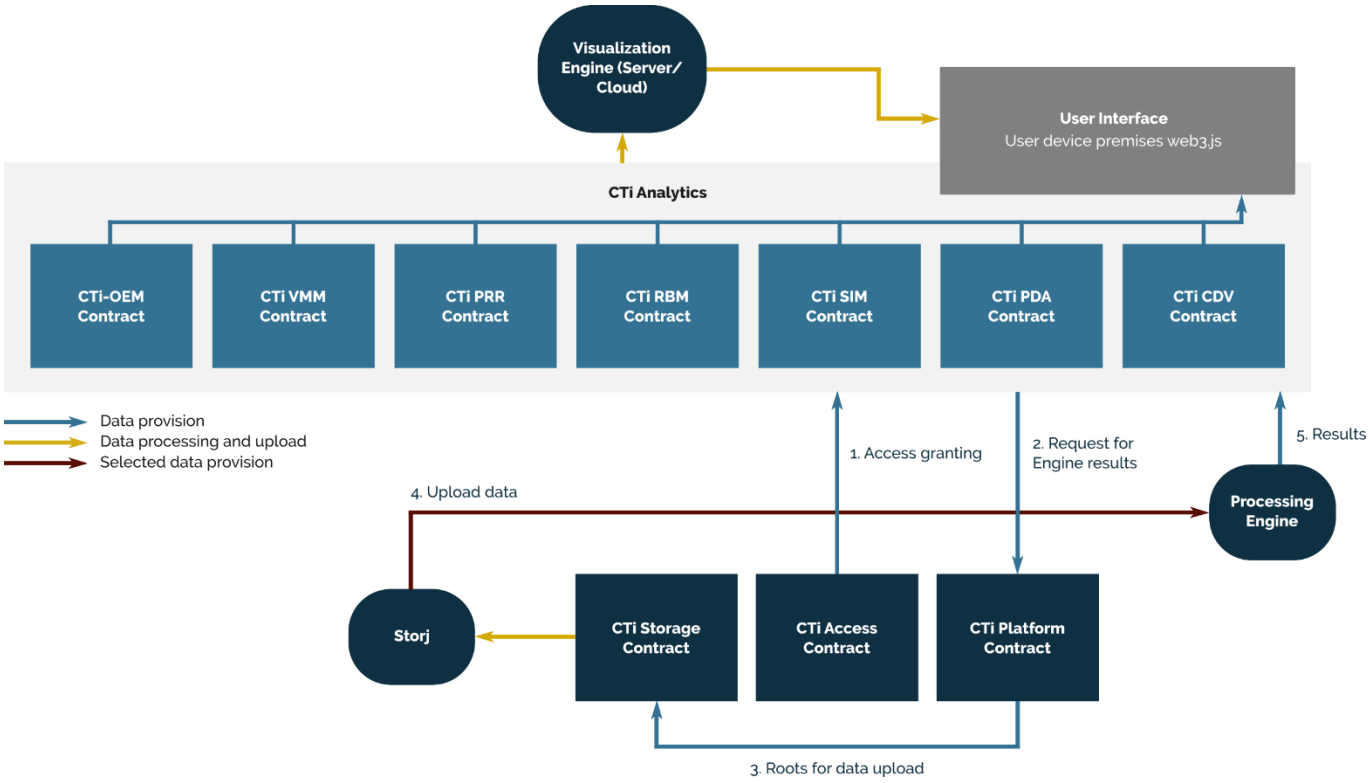
Stage 3 - CTi Platform (On-Chain) with Decentralised Clinical Database (Storj)

This stage consists from two sub-processes:

1. Migration of the CTi platforms' computation powers to the blockchain.

This requires the implementation of new smart contracts and new decentralised components, including a dedicated smart contract for the CTi Processing engine.

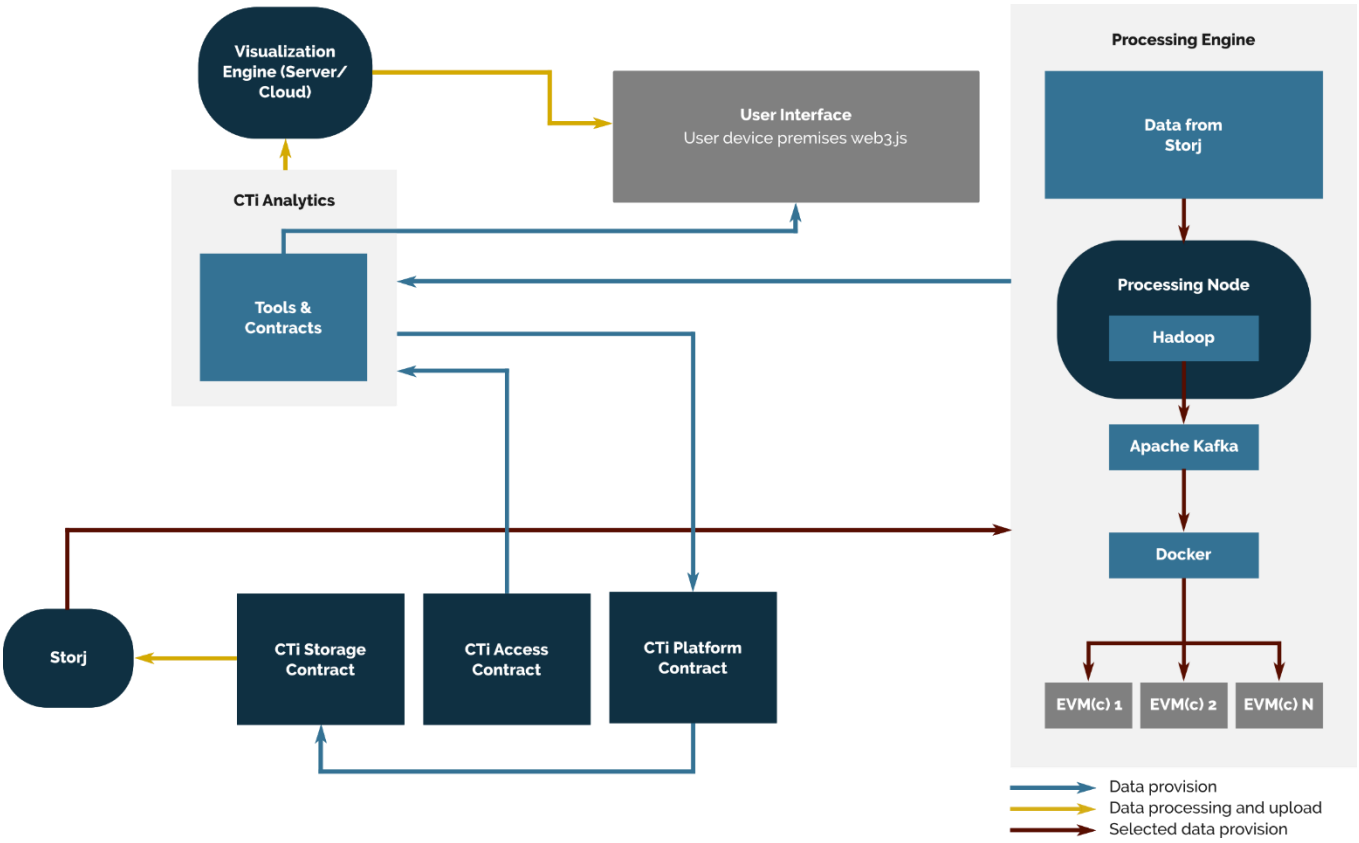
Structure of Smart Contract for CTi Platform DApps (Phase 3):



When a user engages the CTi front end:

- The access smart contract will verify that the user licence is valid by checking the staked CLX balance.
- On user request, the CTi applications will send requests for data processing to the platform contract.
- The platform contract will then send roots of the required data and initiate and upload of it from Storj.
- Data is uploaded from Storj to the processing engine.
- The processing engine will then provide the results directly to the applications environment for visualisation to the user.

CTi Processing Engine in Blockchain Environment



An intermediary process will be required to facilitate efficient decentralised computations when data goes to the processing engine smart contract. A smart contract will upload data from Storj directly to one of the processing nodes[3], where the following steps will be executed:

- Data is split into packages defined by its purpose with Hadoop.
- The packages are arranged into the correct sequence with Apache Kafka and sent via Docker for final delivery to EVM(c).
- EVM(c) will perform the decentralised computations and pack results via Docker to Apache Kafka.

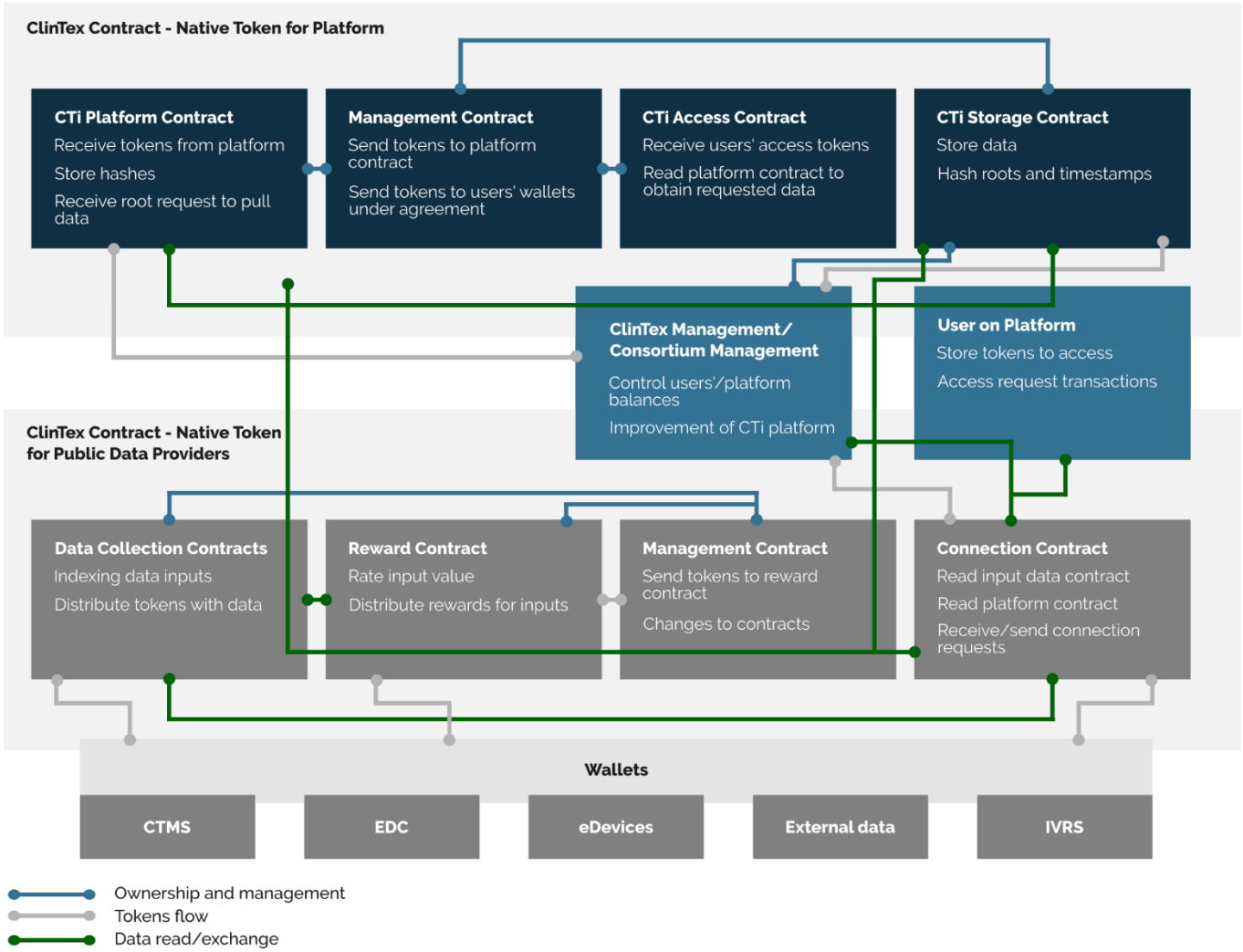
This process could be thought of as a sharding of EVM, instead of entire blockchain sharding. It is assumed that with such an approach, performance of computations inside a blockchain-based VM will increase significantly.

2. Data provision through CTi permissionless blockchain.

This extension is deployed in parallel with sub-process 1 above and will introduce new smart-contracts in the CTi permissionless environment.

[3] As it is not feasible to have all nodes processing, it will be arranged by a process of delegate selection.

ClinTex Smart-Contract Structure (Phase 3)



The additional smart contract introduced to the CTi public Ethereum blockchain will roll out the following features:

- The data collection smart contract. This contract processes direct hashing of data and loading to Storj through Metamask/ Infura and CTi's web3.js user interface. This contract will also have a fabric function to integrate contracts per each source of data.
- The reward contract. This contract enables the introduction of CLX token rewards. This contract will include the rules based on predetermined criteria.
- The connection contract. This contract is for back-end coordination of the data flow to Storj and setting remuneration for its provision through oracalisation of the consortium chain gateway.
- The management contract. This creates the ability to re-establish new types of contracts and kill old versions of them.

Contracts

CLX Contract

The CLX contract is dedicated to the Token Generation Event (TGE) of the CLX ERC20 token in Ethereum's permissionless blockchain.

Specifications of the TGE and parameters of the generation and distribution process will be available in the smart contract code on GitHub.

As the platform for CLX token distribution is Ethereum Mainnet, participants will be able to receive and store the CLX (ERC20) token on any wallet that supports the ERC20 protocol.

Consortium Proof of Authority Contract

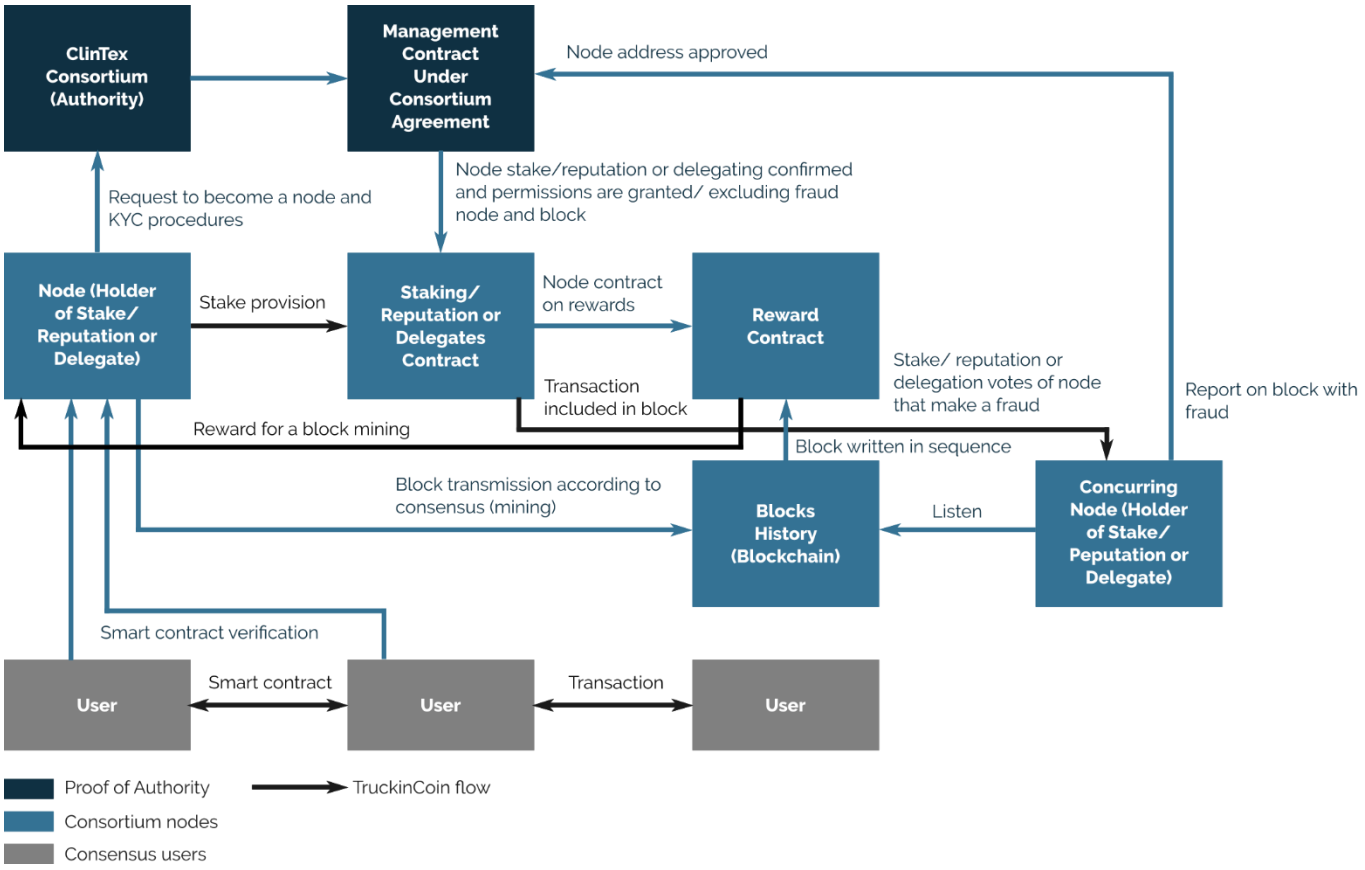
Although anyone can set up a node and step in to support the network in its production, business logic and data access will be granted upon Proof of Authority (PoA) implementation, both on node consensus and smart-contract layers.

Although there is general control at management level over consortium nodes, to allow for improved consensus stability over PoA (GHOST Proof of Work) nodes, the Ethereum consortium chain source code will have several amendments (in the form of smart contracts) to allow for 3 new forms of node participation: staking model, reputation model and delegation.

With this mechanism the system will introduce an economic motivation for nodes to follow the technical consensus consortium and verify transactions and blocks.

The PoA protocol will allow predefinition of nodes that are responsible for smart-contract verification.

Proof of Authority Consensus



Establishing mechanism for consensus:

1. Anyone who wants to become a node must follow a procedure of confirmation, complete KYC/AML procedures, and sign off a Service Level Agreement.
2. ClinTex (as a leader of the consortium) with its Proof of Authority will verify nodes.

From this point, the CTi platform can be deployed under different independent consortiums:

Option 1 - Staking model

- The node operator purchases CLX from public exchanges and uses them as a stake in the staking contract. This stake is frozen in the contract and the reward contract uses it to calculate an amount of reward for mined blocks. The amount of each reward is calculated as a percentage of stake in the model multiplied by the amount of transaction fees/smart-contract gas (complex fees for smart-contract) in each block.

Option 2 - Reputation model

- Nodes are allowed to verify transactions with minimal reputation points. The nodes reputation is increased with each transaction verified. Amount of the reward is calculated as a percentage of reputation points in the reputation model multiplied by the amount of transaction fees/smart-contract gas (complex fees for smart-contract) in a block.
    - For both of the above options there are other nodes that are competitors and listen to the ledger to verify that blocks were written according to the technical consensus. If the technical consensus was not respected, the concurring node will report to the management contract. This block will be excluded from the ledger, the transaction with fraud will be blocked and only trusted transactions will fall into the mempool of nodes to be written in the next block. Any node running at the same time is a verification and concurring node for other ones.

Option 3 – Delegation

- Delegated Byzantine Fault Tolerance (DBFT) consensus could be run on the consortium nodes. A smart-contract could sit between 10 delegates that will use BFT protocol to vote on written blocks secondly. This would allow for the protection of interests of all consortium members of the permissioned network.

With this hybrid model, ClinTex ensures that nodes are rewarded according to their stake/reputation or as delegates in a trusted and transparent manner. At the same time, ClinTex has the authority to upgrade the network and assure users that nodes follow the rules while the consortium is responsible for system maintenance.

Smart Contract Permissions and Rules

The smart contracts that are responsible for key tasks in CTi interact using platform internal tokens (ERC721) in the permissioned environment, or with CLX (ERC20) in the permissionless environment.

All communication between users and smart contracts is arranged with DApps that will include all permissions inside them.

CLX TGE Smart Contract	
Tasks	Issue of CLX token, control over distribution process
Permissions	Permissionless smart contract that requires ETH on user balance to interact with it
Rules	Rules of smart contract defined by ERC20 standard (no customisation required)
Interact with	User wallet (DApp), CTi permissioned blockchain gateway (Oracle)

CTi Permissioned Blockchain Management Smart Contract	
Tasks	▪ Control of internal permissioned blockchain environment, smart contracts that handle back-end activity and implementation of new rules of consensus for nodes (if required by consortium) ▪ Send tokens to platform contract to ensure token flow ▪ Send tokens to users' wallets under agreement as incentives
Permissions	Accessible only for CTi platform administration and consortium members. In the case of pure consortium deployment (upon request to ClinTex), permissions belong to consortium holders as a multisignature smart contract
Rules	On multi-signature, the smart contract allows a "kill" function to existing contacts and/or allows deployment of new smart contracts as a library inside the platform

Interact with	Interacts with users DApps, node software and all smart contracts inside CTi's permissioned blockchain

CTi Permissioned Blockchain Access Smart Contract	
Tasks	▪ Receive users' access tokens from platform subscription ▪ Read platform contract to obtain requested data
Permissions	Permissioned: accessible only for user with minimal required amount of CLX for subscription and ETH their transaction
Rules	Verify user address with a type of subscription to match user uploaded/downloaded data with appropriate CTi tool or processing engine result
Interact with	User wallet, management and platform smart contracts

CTi permissioned blockchain Platform Smart contract	
Tasks	▪ Receive tokens from platform to supply data storage and token flow ▪ Store hashes of data in off-chain/on-chain storage ▪ Send tokens with hashes to access contract upon request ▪ Indicate reward for data provision
Permissions	Only for data providers
Rules	With access to platform, users that indicate data provision will be able to send transactions to smart contract addresses with data hash in metadata. For used information, data providers receive a reward according to the management contract's pre-defined amount.
Interact with	User wallet, management smart contract, access smart contract, storage smart contract with Storj deployment

CTi Permissioned Blockchain Storage Smart Contract	
Tasks	▪ Store data in Storj ▪ Hash roots and timestamps for data indexation

Permissions	Accessible only for management smart contract, listen to platform smart contract, have access to Storj.
Rules	Storage contract listens to platform contract, obtains hash of data, then timestamps and roots in Storj in hashed and concatenate form
Interact with	Management smart contract, Storj, platform smart contract

CTi Permissioned Blockchain Smart Contract per Each Tool	
Tasks	▪ Communication with tool DApp ▪ Obtain data requests and their upload ▪ Support of processing engine data transportation (with Docker plugged)
Permissions	Permissionless for any user with subscription for dedicated tool
Rules	Platform contract obtains secondary data from relevant contract, storage contract pulls required data, user interacts with tool frontend that is connected to DApp in back-end
Interact with	Tool DApps, platform UI, platform and storage smart contract, Storj

CTi Permissioned Blockchain: Processing Engine Smart Contract	
Tasks	Ensure acceptance of data for computation in EVM(c)
Permissions	Strictly permissioned; only nodes have ability to provide data in it
Rules	Allow deployment with smart contract fabric function rules for distributed computations of data for CTi Platform
Interact with	Processing nodes, nodes, EVM(c)

CTi Permissionless Blockchain: Management Smart Contract	
Tasks	▪ Control of internal permissionless blockchain contracts that define rules of external parties' involvement to CTi platform

	▪ Send CLX to reward contract to ensure token flow ▪ Obtain CLX from CTi platform subscription wallet
Permissions	Permissioned with multisignature
Rules	Ability to deploy or kill CTi related smart contract when multisignature in Ethereum environment has been obtained
Interact with	CTi platform Ethereum wallet, all CTi-related smart contracts inside public Ethereum environment

CTi Permissionless Blockchain: Connection Smart Contract	
Tasks	▪ Read the data collection contract in public Ethereum blockchain ▪ Read platform contract through gateway ▪ Receive/send connection request messages between platform to reduce gas fees on continuous data provision
Permissions	Permissioned to access only the management smart contract, to send CLX to reward smart contract in case of accidental sending of funds to its address
Rules	Listen to data uploads to activate data transfer to Storj, listen to platform gateway to ensure that data at upload is not repeated
Interact with	Management smart contract, data collection smart contract and permissioned blockchain gateway

CTi Permissionless Blockchain: Data Collection Smart Contract	
Tasks	▪ Indexing data inputs by provision of hash in metadata ▪ Distribute CLX with data hashes in metadata through gateway
Permissions	Permissionless, accessible only with the dedicated DApp (limit wallets with CLX, but not related to CTi activity)
Rules	Users' dedicated DApps that enable them to provide data inputs directly to the CTi platform. DApps to have integrated hash functions to provide the hash for metadata and for indexing of information piece.

	Gateway activated by connection of the contract scan transaction and mirror hash from CLX transaction metadata to the internal token in the permissioned blockchain
Interact with	Dedicated DApps, gateway, connection smart contract

CTi Permissionless Blockchain Reward Smart Contract	
Tasks	▪ Rate input value through oraclisation of gateway to permissioned blockchain ▪ Distribute CLX as a reward for data inputs
Permissions	Permissionless, increase reserves with subscription, and decrease with reward payments
Rules	After deployment, instead of ClinTex Ethereum wallet, users will send CLX for subscription to the reward smart contract address. The reward contract will then distribute according to the amount of collected revenue inside the permissioned blockchain, calculated in the back-end.
Interact with	Management smart contracts, users' wallets, gateway

CTi Permissionless Blockchain Smart Contracts per Each Data Source	
Tasks	Ease process of fees calculation for Storj usage, integrate IoT or other automated data source
Permissions	Permissionless
Rules	Users on the CTi platform front-end choose the type of data source they will use to provide data. This will activate deployment of the appropriate smart-contract. Each data source will get a unique private signature to provide an identity feature for them in future operations. It automatically includes additional CLX/ETH for covering Storj fees. This will be compensated after each data purchase[4]
Interact with	Dedicated DApp, gateway, users' wallets

CLX (ERC20) will be used for all CTi activity on the permissionless Ethereum blockchain.
CTX (ERC721) will be used for all CTi activity on the permissioned blockchain, to enable the transfer of data.

[4] This is done to prevent deployment of a contract without any further data provision.

CTi Platform Internal Token (ERC721)

Key Specifics of CTX Internal Token:

Feature	Internal token
Value	Zero value without metadata. With metadata value will increase by 1 each transaction to provide an input to the gateway to the reward smart contract
Amount	Unlimited (basically limited only by amount of natural numbers)
Transaction fee	Zero, all fees are prepaid with subscription
Metadata	Included inside token
Minting	All tokens are pre-mined and minted
Burning	After token disposal, the metadata is deleted from it and the token unique ID is equal to the last minted token
Access	Owner ID is equal to user ID that is obtained during registration process
Utility	Full utility token
Market availability	Not available on the market. It could be reached with wallets and DApps, but transactions occur only inside the permissioned blockchain

Security

Security is a major critical aspect of the CTi platform, and as such ClinTex will be vigilant in its use of the latest security measures including unique blockchain security features.

Storage

CTi's client-side encryption to Storj nodes in combination with Storj's own encryption of all data will make it impossible to gain access to users' files without possession of the corresponding private encryption key.

Storj's decentralised nature also means its designed to protect from traditional attacks on cloud storage, including:

1. Spartacus
2. Sybil
3. Eclipse
4. Hostage bytes
5. Cheating owner
6. Faithless farmer
7. Defeated audit attacks

More on the Storj structure: *https://storj.io/storj.pdf*

CTi File System with Root Encryption

Despite the permissioned nature of the CTi blockchain, consortium members will have different levels of access to it. Hashes of data provided through the public blockchain are visible publicly, which could lead to the compromising of a Storj node by attacking a particular node that stores the required hash.

To avoid such vulnerabilities, a hash of the data that is used for the search within the Storj nodes will be combined with a unique identifier, known only to specific authorised parties.

This will enable a combination of on-/off-chain communication, as during data provision the user will obtain Storj's root to shards SALT as well as the timestamp of the data upload. The user end will hash their data with the DApp, and then hash root to its storage and concatenate both values. The result of this is hashed once more and then concatenated with the timestamp hash.

In off-chain encrypted channels, arranged with SSL/TLS digital certificates or with PGP keys, a user will send root to data and timestamp of its upload. The receiver will hash the timestamp to search data inside Storj and provide its meaning of root to data. If root hash with data hash, known to a node, is equal to hash in a token (double data and root hash with timestamp hash), the system will allow the receiving party to download the data.

Hashes are stored inside the ERC721 token, which will also contain an Owner_ID. The Owner_ID system will also work to facilitate the restricting of access to any unlicensed user. The secure data being stored inside a token guarantees that it is tamper free and assured by Ethereum source code.

Details on Ethereum specifics, including security of data storage on nodes are here:
https://github.com/ethereum/yellowpaper

Oracles + Nodes

CTi's oracles and node security rely on the security mechanisms implemented inside the Ethereum nodes' software.

Oracle connections with external data sources are established only with encrypted channels based on SSL/TLS certificates. In these channels, the Oracle will send messages with RESTful JSON API.

To reinforce security in the permissioned blockchain, each node will be connected with a TLS cert or VPN connection. Nodes will use Linux OS for servers. If a node consists of more than on working station, it will be a requirement to connect them within MESH-network.