Característiques Plataforma SCIPIO

Participants

|  |  |  |
| --- | --- | --- |
| **PARTICIPANT** | **What value adds** | **What it gets** |
| **Investor** | Provides financing for CAPEX investment | Value Accruation in Token |
| **Installer** | Plans & Executes Investment. Coordination with local authorities. | Payment of invoiced installation. |
| **ServiceProvider** | Offers ISP | Revenues from end users through compensation table. |
| **SuperNode** | Offers connectivity to end user and access to decentralized services | Revenues from end users through compensation table. |
| **EndUser** | Makes investment interesting for Investors | Internet Connectivity. Accés to Blockchain. |
| **PA (Public Authority)** | Facilitates investment and network development | Reduced connectivity costs due to cost-oriented philosophy of common networks. |
| **Scipio Foundation** | Streamlines CAPEX Investment Automated Conflict Resolution  Public real-time network monitoring and usage  Bandwitdh trading  Democratising access to internet and blockchain. | Capex-fee: 2.5% charged to investors. 0.5% charged to Professionals.  Revenue fee: 10%. 50%, 100% for Type A/B/C operators). |
| **Token Investor** | Initial Funding for Platform Development and Marketing | Token Appreciation |

# SmartContracts

## SuperNodeManager

Each SuperNode has to be created by its owner. The creation can be called by anybody and will receive as arguments in constructor:

Name, Owner, Role\_owner, Zona

This SmartContract should:

* Upload New Expenses related to the SuperNode
* Get the expenses validated by the Foundation (initially each SuperNode validates them).
* Add new nodes to wich it connects.
* Add inflow in GB of Supernode
* Add inflow in monetary units (payments collected)
* Add outflow in GB served to nodes
* Get total contributed ammount.
* Get total Inflow of money
* Get total Outflow of GB served.

Struct SuperNode

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type |  | Comments |
| nameSuperNode | string | public |  |
| ownerSuperNode | address | public |  |
| role\_owner | Mapping address  mapping  enum  bool | Public | Investor / Installer / ServiceProvider / SuperNode / EndUser / PA / Scipio / Token Owner |
| superNodeZone | string | public |  |
| SuperNodePositionLat | string | public |  |
| SuperNodePositionLong | string | public |  |
| SuperNodeMallada | bool |  |  |
| superNodeEstado | enum |  |  |
| superNodeConnectedNodes | Mapping address  bool |  |  |
| superNodeHeight | uint |  |  |
| superNodeLinks | Address[] |  | Array of SuperNodes it is connected to |
| superNodeCost | Uint |  | Amount in wei of total cost of installation that is required to fund. |
| superNodeOPExpenses | Uint |  | Total Ammount of expenses (maintenance ,etc) |
| superNodeExpectedYearlyOpCosts | Uint |  | Expected yearly operating costs of node (maintenance, etc). |
| Expenses | Struct[] |  |  |
| expensesTotalAmmount | uint |  |  |
| periodEndUserBalances | Mapping address  uint |  |  |
| periodSupplyBalances | Mapping address  uint |  |  |

Struct Participant

|  |  |  |
| --- | --- | --- |
| Elements | Type | Comments |
| This.owner | address |  |
| Ammount | uint |  |
| Issuer NIF | string |  |
| issuerName |  |  |
| expenseDate | bytes |  |
| expenseStatus | enum | Presupuestado / Programada / En Construcción / Finalizada |
| expenseFinalised | Bool |  |
| expenseIPFS | String | Hash of IPFS link to document |
| inicioProgramada | Uint256 |  |
|  |  |  |
|  |  |  |

### STRUCT Expense

|  |  |  |
| --- | --- | --- |
| Elements | Type | Comments |
| This.owner | address |  |
| Ammount | uint |  |
| Issuer NIF | string |  |
| issuerName |  |  |
| expenseDate | bytes |  |
| expenseStatus | enum | Presupuestado / Programada / En Construcción / Finalizada |
| expenseFinalised | Bool |  |
| expenseIPFS | String | Hash of IPFS link to document |
| inicioProgramada | Uint256 |  |
|  |  |  |
|  |  |  |

SuperNode Methods:

* Afegir nodes
* Get list of active nodes connected
* Afegir Links
* Get list of active links
* Contribuir al finançament inicial del node.

## Participants

Logic: This Smart Contract will have all the business logic that applies to platform participants.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type |  | Comments |
| Name | string | public |  |
| Owner | address | public |  |
| Role\_owner | Mapping address  mapping  enum  bool | Public | Investor / Installer / ServiceProvider / SuperNode / EndUser / PA / Scipio / Token Owner |
| Zona | string | public |  |
| Position\_lat | string | public |  |
| Position\_long | string | public |  |
| Mallada | bool |  |  |
| Estado | enum |  |  |
| connectedNodes | Mapping address  bool |  |  |
| Height | uint |  |  |
| Links | Address[] |  | Array of SuperNodes it is connected to |
| superNodeCost | Uint |  | Amount in wei of total cost of installation that is required to fund. |
| superNodeOPExpenses | Uint |  | Total Ammount of expenses (maintenance ,etc) |
| Expenses | Struct[] |  |  |
| expensesTotalAmmount | uint |  |  |
| periodEndUserBalances | Mapping address  uint |  |  |
| periodSupplyBalances | Mapping address  uint |  |  |

## SuperNodeSponsorship

Logic: manage the value transfer between different participants from the beginning of the process until the end of the SuperNode life.

* Should allow addresses who “own” a SuperNode (even if it is in funding phase) to start a FundingRequest.
* Should track monetary contribution and stablish pro-rata of each contributor on the SuperNode. Future value accruition from revenues generated from this SuperNode will be given based in this pro-rata.
* Should let contributors approve investments. If more than 50% of contributors approve them, it is executed.

## Monitoring

Logic: tracks the relative balance of internet bandwith exchanged between all participants, which will be used in the end of the period in the compensation table. This SC should:

* Keep an updated state of the state channels currently opened and provide this info to the CompensationSystem at the end of the period.
* Should reinitialize once compensation has been executed for period.

## CompensationSystem

Logic: Common-network specific system that calculates how much each participant has provided to the network in terms of monetary units and how much it has got in exchange. Automatic Resolution of outcomes. This SC should:

* Should Check for Contribution in EUR of each of the participants and its relative %.
* Should check for Consumption in TB of each of the participants and its relative %.
* Should calculate the resulting Balance (%), deducting % Consumption from % Contribution and multiplying it per the total amount contributed.
* Should include Scipio Platform as management costs.

## Invoicing

Logic: should regulate how customers are invoiced. It should:

* Stablish Invoicing Type: per GB / served, fix-rate. Until end of GB allowance.
* Stablish periodicity: once per month
* Clearly communicate the parts of the invoice, where do they go (desglose factura)
* For user, should allow easy consultation of historical invoices and manage subscription options.
* For SuperNode, it should allow easy management of connected nodes.

## ConflictResolution

## BandwitdhAuction

Logic: manages the processes that allow investors and token holders to trade their tokens in exchange of service or money.

TO BE DONE:

Create Standard ERC20 Token for Platform/ICO Token. Features:

* Non Mineable.
* Burneable
* Hard-Capped.
* Tradeable

Create Internal Bandwith Token

* Mintable based on permission
* Tradeable

Token Dynamics

Platform Token

It allows preferential access to bandwitdh auctions and allows for participation in the Bandwitdh Pool as per % of tokens held.

Internal BandWitdh Token (BWT)

It is created throught a smart contract and sold by installatators in order to sell upfront internet bandwitdh in order to finance the project. The smart contract regulates: 1) type of connection, 2) time stamp when the installation will be operative; 3) price tag (€/TBh). This smart contract represents a standard ISP purchase agreement between ISP and bandwitdh buyer.

1 BWT equals to 1 TB / s.

Example:

Let’s say that a local operator wants to extend the fiber cable installation from the closest village to his village. The total ammount to invest is 100.000€, from which he wants to fund 50.000€ by issuing and selling tokenised future bandwitdh. Therefore he creates 810.000 BWTs and sets initial price at 20% discount of market price (0.061 € / TB). He sets the timeframe, location, id, and set up details. The platform previously does the project due dilligence.

Maximum ammount of raisable funding:50% of total investment. The rest should be raised by promotors.

0.9% of the tokens sold are donated to the Platform Token Holders through the Pool. Platform Token Holders may:

1. Use the BWT tokens for their own personal / community use whenever the service starts.
2. Sell the tokenised Banwitdh before service starts to any other user.
3. Automatically sell the Bandwith to Wholesale market once the bandwith capacity is operative and receive payment in fiat or crypto.

If the token holder decides to choose the third option, he can i) cash out the proceeds immediately or at a future date; or ii) reinvest the received amount in common network enhancement and keep storing value via bandwith tokens. The value of this token will never drop below its book value - the market cost of internet bandwitdh.

300 MB / s cost 60€/month. That means 0.077 € / TB aprox.

If a connection provides 500.000 GB over a period of one hour (3600 seconds), it means it has provided 138 Bandwitdh Tokens.

0.9% of this SCP will go to the Common Pool.



