# Table of Contents


**1 cadenaDeBloques_RPava 4**
1.1 Capa de Negocio 4
1.1.1 Diagrama de Capa de Negocio diagram 4
1.1.2 Nuevo Bloque 5
1.1.3 Roles Y Actores 5
1.1.3.1 Cliente 6
1.1.3.2 Minero 6
1.1.4 Servicios del Negocio 7
1.1.5 Transacción 7
1.2 Diagramas Estructurales 8
1.2.1 1. Diagrama de paquetes 8
1.2.1.1 1. Diagrama de paquetes diagram 8
1.2.2 2. Diagrama de Clases 9
1.2.2.1 2. Diagrama de Clases diagram 9
1.2.2.2 cadena 9
1.2.2.2.1 3.1 Diagrama detallado cadena diagram 10
1.2.2.2.2 Bloque 10
1.2.2.2.3 CadenaBloques 12
1.2.2.2.4 NodoMinador 13
1.2.2.2.5 PoolMinado 14
1.2.2.2.6 Transaccion 16
1.2.2.2.7 pruebasUnidadBloque 17
1.2.2.2.8 pruebasUnidadCadena 17
1.2.2.2.9 pruebasUnidadNodoMineria 18
1.2.2.2.10 pruebasUnidadPoolMinado 18
1.2.2.2.11 pruebasUnidadTransaccion 18
1.2.2.3 cliente 20
1.2.2.3.1 3.2 Diagrama detallado cliente diagram 20
1.2.2.3.2 Usuario 20
1.2.2.3.3 pruebasUnidadUsuario 21
1.2.2.4 simulador 23
1.2.2.4.1 3.3 Diagrama detallado simulador diagram 23
1.2.2.4.2 Simulador 23
1.2.2.4.3 pruebasUnidadSimulador 25
1.2.3 4. Diagrama de Objetos 26
1.2.3.1 4. Diagrama de Objetos diagram 26
1.2.4 5. Diagrama de componentes 27
1.2.4.1 5. Diagrama de componentes diagram 27
1.2.5 6. Diagrama de despliegue 28
1.2.5.1 6. Diagrama de despliegue diagram 28
1.2.5.2 Nodes 28
1.2.5.2.1 Nodes diagram 28
1.2.5.2.2 Clients 29
1.2.5.2.2.1 Clients diagram 29
1.2.5.2.3 Devices 30
1.2.5.2.3.1 Devices diagram 30
1.2.5.2.4 Servers 31

1.2.5.2.4.1 Servers diagram 31
1.2.5.3 Artifacts 32
1.2.5.3.1 Artifacts diagram 32
1.2.5.4 Topology 33
1.2.5.4.1 Network diagram 33
1.3 Diagramas de comportamiento 34
1.3.1 8. Diagramas de casos de uso 34
1.3.1.1 8. Diagramas de casos de uso diagram 34
1.3.1.2 Actores 34
1.3.1.2.1 Actores diagram 35
1.3.1.3 Primary Use Cases 36
1.3.1.3.1 Primary Use Cases diagram 36
1.3.2 9. Diagramas de Secuencia 37
1.3.2.1 9.1 Diagrama de Secuencia pruebasUnidadUsuario diagram 37
1.3.2.2 9.2 Diagramas de Secuencia pruebasUnidadSimulador diagram 38
1.3.2.3 9.3 Diagramas de Secuencia pruebasUnidadTransaccion diagram 39
1.3.3 10. Diagramas de Comunicación 41
1.3.3.1 10.1 Diagramas de Comunicación Usuario diagram 41
1.3.3.2 10.2 Diagramas de Comunicación pruebasUnidadSimulador diagram 41
1.3.3.3 10.3 Diagramas de Comunicación pruebaUnidadTrasaccion diagram 42
1.4 Extendidos 43
1.4.1 15. Requisitos de requerimientos 43
1.4.1.1 15. Requisitos requerimientos diagram 43
1.4.2 16. Metamodelo 45
1.4.2.1 16. Metamodelo diagram 45

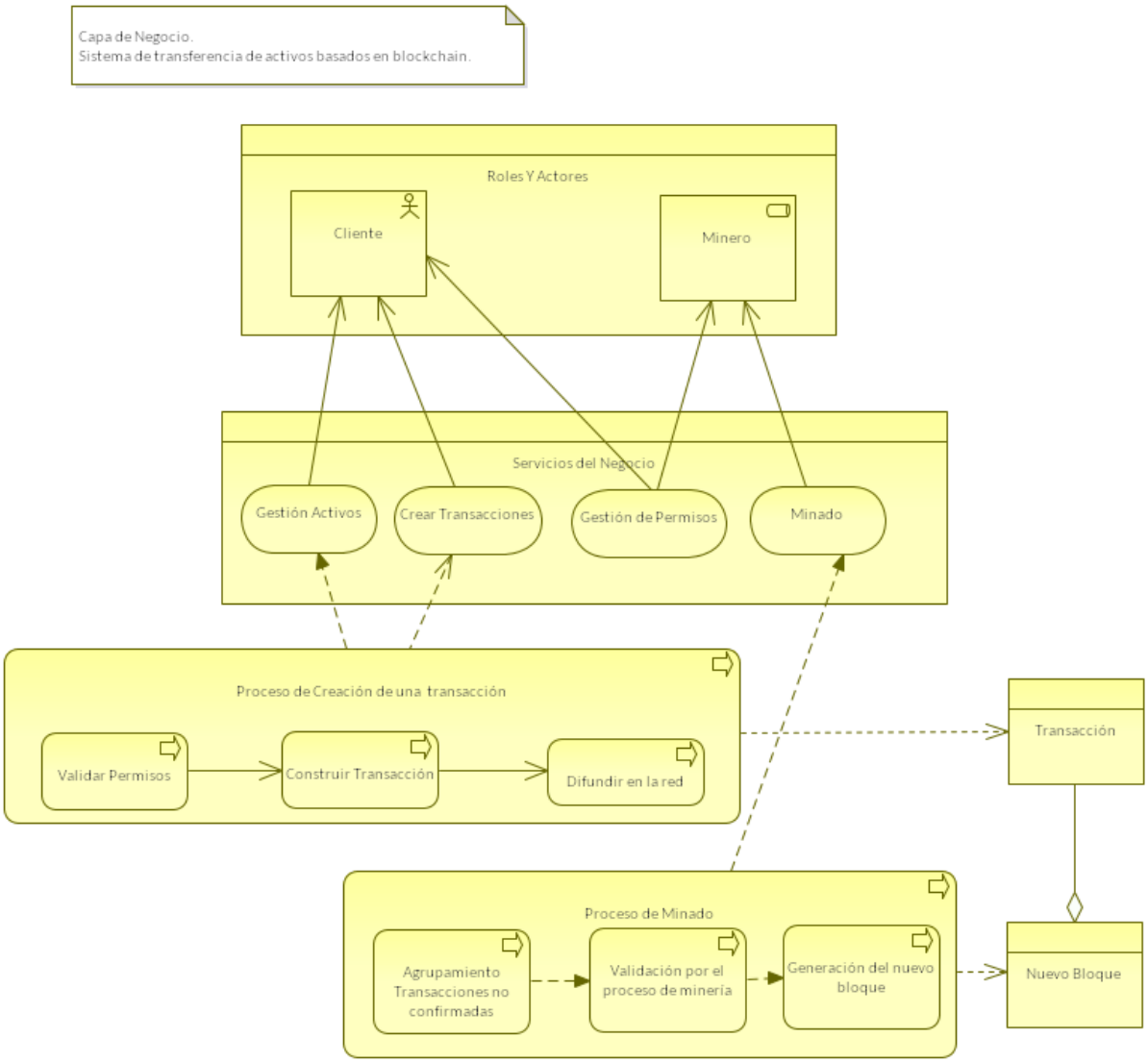


Figure 1: Diagrama de Capa de Negocio

## 1.1.2Nuevo Bloque

*BusinessObject «ArchiMate_BusinessObject» in package 'Capa de Negocio'*

Nuevo Bloque
Version 1.0 Phase 1.0 Proposed
blockchain created on 31/05/2018. Last modified 31/05/2018

| CONNECTORS |
| --- |
| **Access** «ArchiMate_Access» Source -> Destination<br>From: Proceso de Minado : BusinessProcess, Public<br>To: Nuevo Bloque : BusinessObject, Public |

## 1.1.3Roles Y Actores

*BusinessObject «ArchiMate_BusinessObject» in package 'Capa de Negocio'*

Roles Y Actores
Version 1.0 Phase 1.0 Proposed
blockchain created on 28/05/2018. Last modified 28/05/2018

**ELEMENTS OWNED BY Roles Y Actores**

Cliente : BusinessActor «ArchiMate_BusinessActor»

Es poseedor de los activos de información o valor que se gestiona en la blockchain. Está habilitado para transferir activos a otro cliente dentro del sistema. Si dispone de los privilegios de administrador puede asignar y revocar permisos a otros clientes.

Minero : BusinessRole «ArchiMate_BusinessRole»

Este actor es responsable de validar las transacciones de los clientes ejecutando el algoritmo de consenso definido. El bloque génesis de la cadena es generado por un usuario con privilegios.

### 1.1.3.1 Cliente

*BusinessActor «ArchiMate_BusinessActor» owned by 'Roles Y Actores', in package 'Capa de Negocio'*

Es poseedor de los activos de información o valor que se gestiona en la blockchain. Está habilitado para transferir activos a otro cliente dentro del sistema. Si dispone de los privilegios de administrador puede asignar y revocar permisos a otros clientes.

Cliente
Version 1.0 Phase 1.0 Proposed
ncapey created on 16/05/2018. Last modified 28/05/2018

**ASSOCIATIONS**

Serving (direction: Source -> Destination) «ArchiMate_Serving»

Source: Public (BusinessService) Gestión Activos «ArchiMate_BusinessService»

Target: Public (BusinessActor) Cliente «ArchiMate_BusinessActor»

Serving (direction: Source -> Destination) «ArchiMate_Serving»

Source: Public (BusinessService) Crear Transacciones «ArchiMate_BusinessService»

Target: Public (BusinessActor) Cliente «ArchiMate_BusinessActor»

Serving (direction: Source -> Destination) «ArchiMate_Serving»

Source: Public (BusinessService) Gestión de Permisos «ArchiMate_BusinessService»

Target: Public (BusinessActor) Cliente «ArchiMate_BusinessActor»

### 1.1.3.2 Minero

*BusinessRole «ArchiMate_BusinessRole» owned by 'Roles Y Actores', in package 'Capa de Negocio'*

Este actor es responsable de validar las transacciones de los clientes ejecutando el algoritmo de consenso definido. El bloque génesis de la cadena es generado por un usuario con privilegios.

Minero
Version 1.0 Phase 1.0 Proposed
ncapey created on 16/05/2018. Last modified 28/05/2018

**CONSTRAINTS**

Invariant. El usuario dispone de la propiedad de un activo de valor o de información para su transferencia
[ Proposed, Weight is 0. ]

**ASSOCIATIONS**

Serving (direction: Source -> Destination) «ArchiMate_Serving»

Source: Public (BusinessService) Minado «ArchiMate_BusinessService»
Target: Public (BusinessRole) Minero «ArchiMate_BusinessRole»

Serving (direction: Source -> Destination) «ArchiMate_Serving»

Source: Public (BusinessService) Gestión de Permisos «ArchiMate_BusinessService»
Target: Public (BusinessRole) Minero «ArchiMate_BusinessRole»

## 1.1.4Servicios del Negocio

*BusinessObject «ArchiMate_BusinessObject» in package 'Capa de Negocio'*

Servicios del Negocio
Version 1.0 Phase 1.0 Proposed
blockchain created on 28/05/2018. Last modified 31/05/2018

## 1.1.5Transacción

*BusinessObject «ArchiMate_BusinessObject» in package 'Capa de Negocio'*

Registro de la transferencia de valor, el cual contiene hash del origen y del destino y los datos del objeto a transferir

Transacción
Version 1.0 Phase 1.0 Proposed
blockchain created on 31/05/2018. Last modified 31/05/2018

**CONNECTORS**

**Access** «ArchiMate_Access» Source -> Destination
From: Proceso de Creación de una transacción : BusinessProcess, Public
To: Transacción : BusinessObject, Public

# 1.2 Diagramas Estructurales

*Package in package 'cadenaDeBloques_RPava'*

Diagramas Estructurales
Version 1.0 Phase 1.0 Proposed
ENVY created on 09/07/2018. Last modified 09/07/2018
Alias

## 1.2.11. Diagrama de paquetes

*Package in package 'Diagramas Estructurales'*

1. Diagrama de paquetes
Version Phase 1.0 Proposed
ENVY created on 09/07/2018. Last modified 09/07/2018
Alias

### 1.2.1.1 1. Diagrama de paquetes diagram

*Class diagram in package '1. Diagrama de paquetes'*

1. Diagrama de paquetes
Version 1.0
ENVY created on 09/07/2018. Last modified 09/07/2018

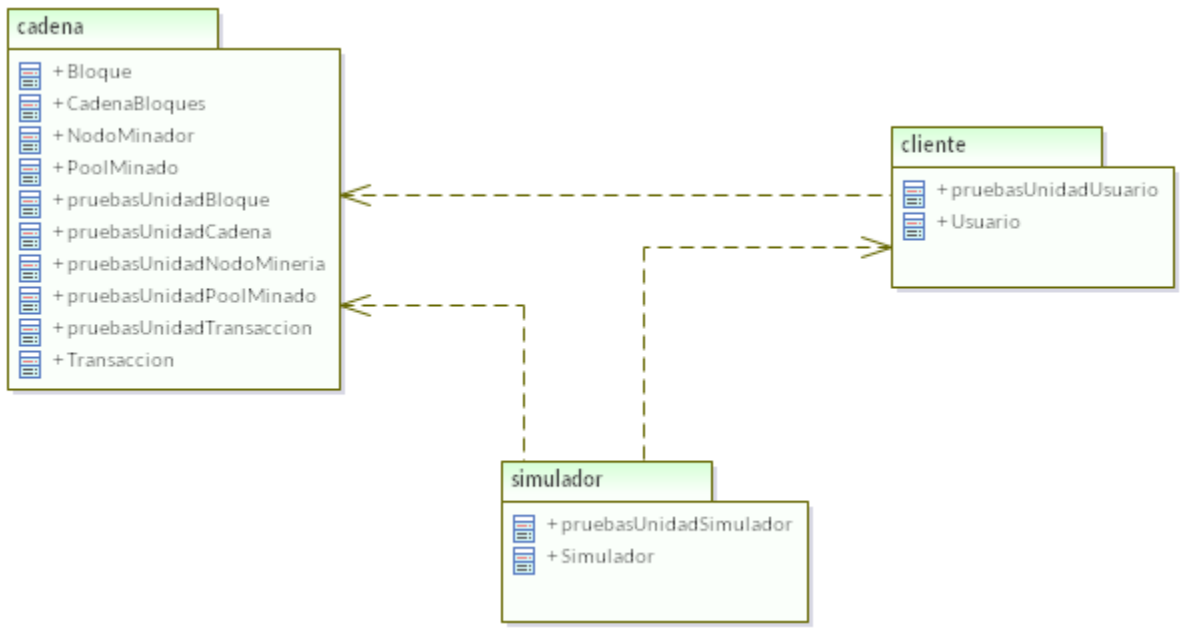


Figure 2: 1. Diagrama de paquetes

# 1.2.22. Diagrama de Clases

*Package in package 'Diagramas Estructurales'*

2. Diagrama de Clases
Version 1.0 Phase 1.0 Proposed
ENVY created on 09/07/2018. Last modified 09/07/2018
Alias

## 1.2.2.1 2. Diagrama de Clases diagram

*Class diagram in package '2. Diagrama de Clases'*

2. Diagrama de Clases
Version 1.0
ENVY created on 09/07/2018. Last modified 09/07/2018

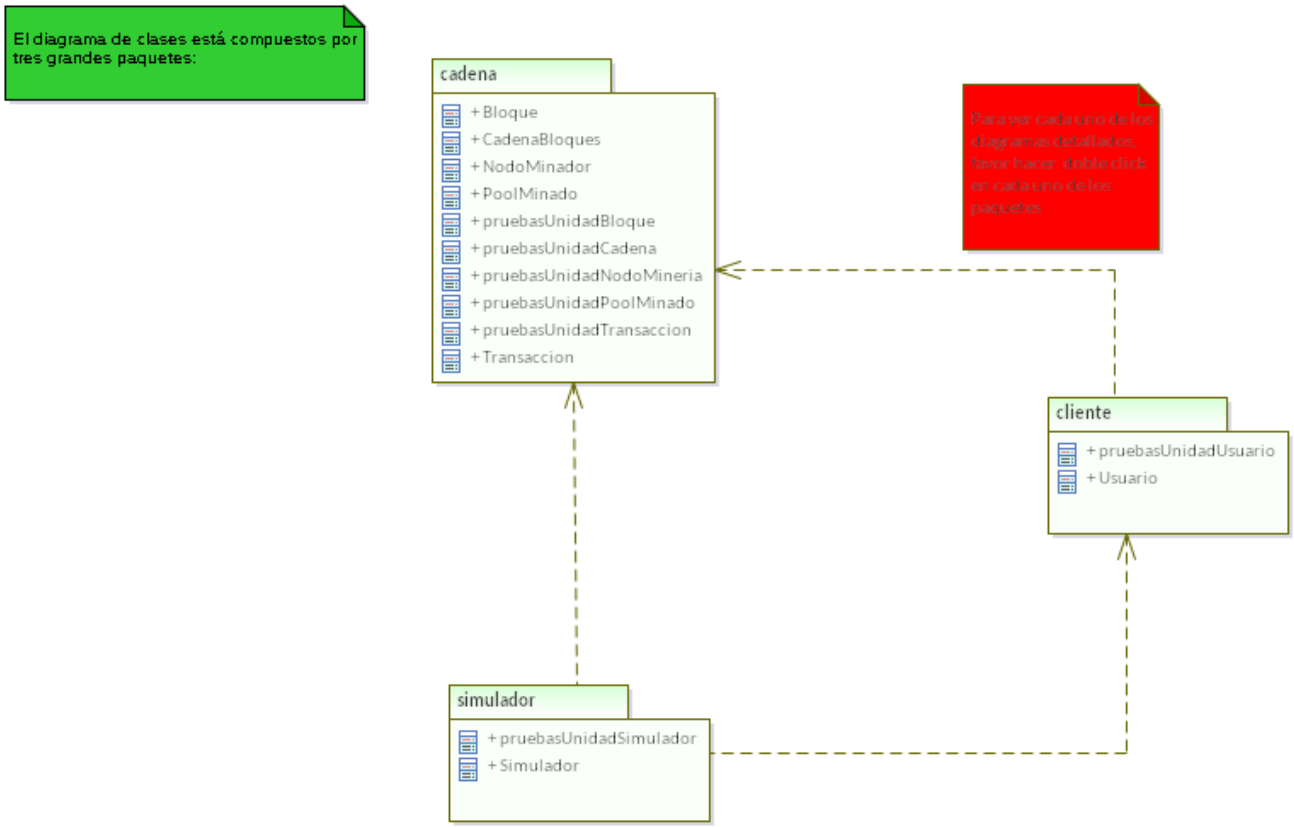


Figure 3: 2. Diagrama de Clases

## 1.2.2.2 cadena

*Package in package '2. Diagrama de Clases'*

cadena
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 09/07/2018
Alias

### 1.2.2.2.1 3.1 Diagrama detallado cadena diagram

*Class diagram in package 'cadena'*

3.1 Diagrama detallado cadena
Version 1.0
blockchain created on 17/06/2018. Last modified 09/07/2018

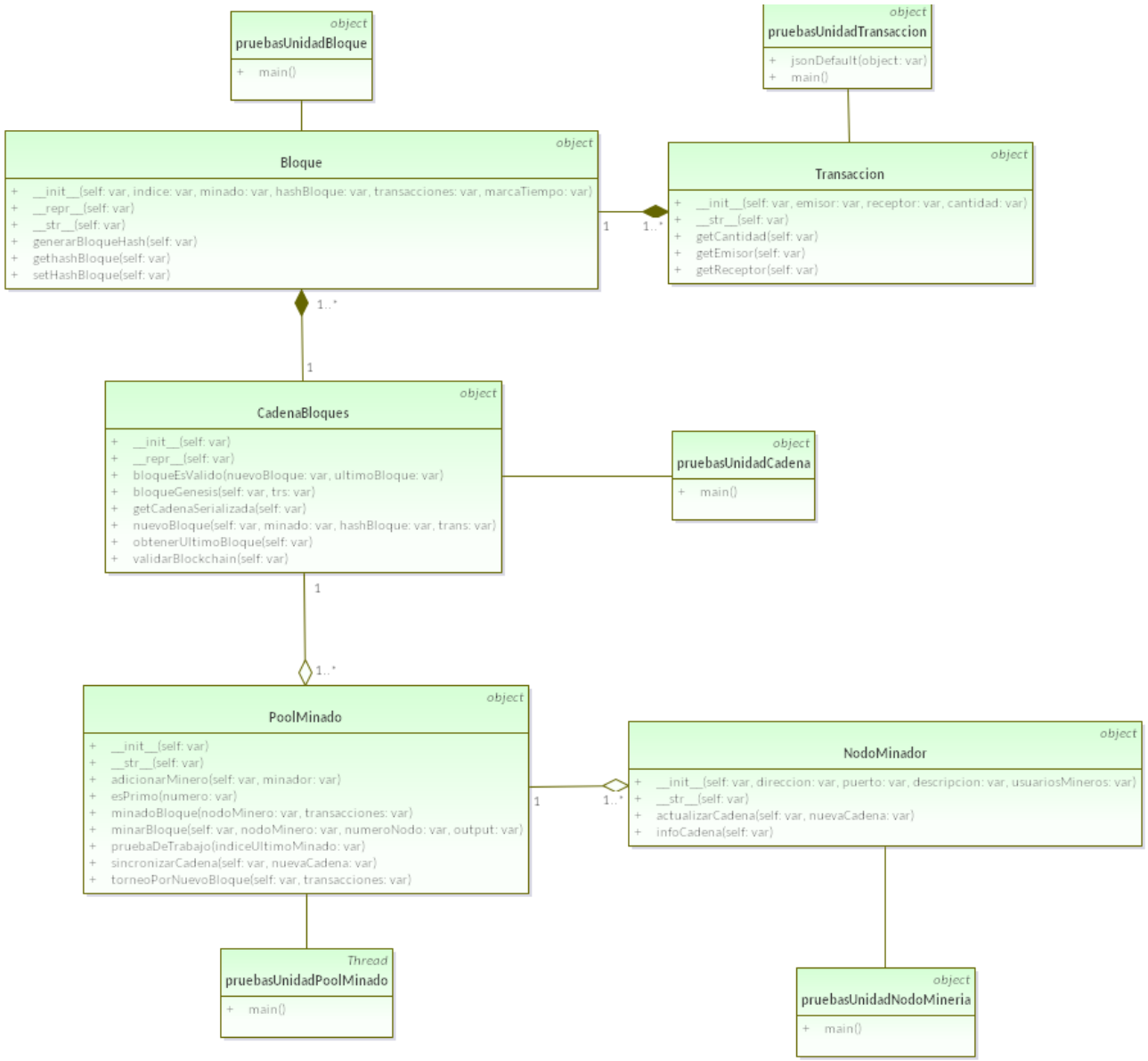


Figure 4: 3.1 Diagrama detallado cadena

### 1.2.2.2.2 Bloque

*Class in package 'cadena'*

Bloque
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018

Extends object

**OUTGOING STRUCTURAL RELATIONSHIPS**

Aggregation from Bloque to Transaccion

[ Direction is 'Source -> Destination'. ]

**INCOMING STRUCTURAL RELATIONSHIPS**

Aggregation from CadenaBloques to Bloque

[ Direction is 'Source -> Destination'. ]

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) pruebasUnidadBloque

Target: Public (Class) Bloque

**OPERATIONS**

__init__ (self : var , indice : var , minado : var , hashBloque : var , transacciones : var , marcaTiempo : var ) : Public

InicializaciÃ³n del bloque :param indice: Valor entero que representa la posiciÃ³n de cada bloque en la cadena :param minado: NÃºmero entero usado en el proceso de minerÃa :param hashBloque: hash del bloque actual en la cadena :param transacciones: Conjunto de transacciones del bloque :param marcaTiempo: Instante de tiempo de creaciÃ³n del nuevo bloque

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

__repr__ (self : var ) : Public

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

__str__ (self : var ) : Public

Genera un string con la informaciÃ³n del bloque :return: representaciÃ³n en String del bloque

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

generarBloqueHash (self : var ) : Public

Genera Hash la informaciÃ³n del bloque :return: Hash construido a partir de la informaciÃ³n del bloque

Properties:
decorators = @staticmethod

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

gethashBloque (self : var ) : Public

Obtiene el valor actual del hash del bloque :return: Hash del bloque

Properties:
ea_guid = {93941509-FCA2-4d45-B5EB-9F78D678FC4B}

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

setHashBloque (self : var ) : Public

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

**OPERATIONS**

### 1.2.2.2.3 CadenaBloques

*Class in package 'cadena'*

CadenaBloques
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**OUTGOING STRUCTURAL RELATIONSHIPS**

Aggregation from CadenaBloques to PoolMinado
[ Direction is 'Source -> Destination'. ]

Aggregation from CadenaBloques to Bloque
[ Direction is 'Source -> Destination'. ]

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) CadenaBloques Target: Public (Class) pruebasUnidadCadena

**OPERATIONS**

__init__ (self : var ) : Public

InicializaciÃ³n de la blockchain :param _cadena: Estructura de datos tipo array para almacenar la cadena de bloques.
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

__repr__ (self : var ) : Public
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

bloqueEsValido (nuevoBloque : var , ultimoBloque : var ) : Public

Verifica bajo cuatro reglas bÃ¡sicas la validez de un Bloque Regla 1: Secuencia ordenada de bloques por valor del Ãndice Regla 2: Secuencia ordenada de bloques por fecha de creaciÃ³n Regla 3: Hash diferente entre el Ãºltimo bloque de la cadena y el nuevo bloque Regla 4: Bloque correctamente minado
@return True si el bloque es vÃ¡lido, False en otro caso

Properties:
decorators = @staticmethod
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

bloqueGenesis (self : var , trs : var ) : Public

Crea el primer bloque en la cadena. Es el bloque seminal.
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

getCadenaSerializada (self : var ) : Public

**OPERATIONS**

Serializa la cadena de bloques en un documento JSON :return conjunto de bloques en formato JSON
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

nuevoBloque (self : var , minado : var , hashBloque : var , trans : var ) : Public

Construye e inserta el siguiente bloque de la cadena :param minado MÃ©todo de minado del bloque :param hashBloque Hash generado para el bloque :return Bloque nuevo bloque generado
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

obtenerUltimoBloque (self : var ) : Public

Devuelve una referencia al Ã°ltimo nodo de la blockchain
@return Ã°ltimo nodo
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

validarBlockchain (self : var ) : Public

Verifica la integridad de la cadena de bloques. para este procedimiento se valida que el Ãndice del siguiente bloque sea menor en una unidad del bloque anterior y que el hash de cada bloque sea correcto segÃ°n la polÃtica de minado establecida, nÃ°meros primos en este caso.
@return True si la cadena esta bien formada, False en otro caso
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

## 1.2.2.2.4 NodoMinador

*Class in package 'cadena'*

NodoMinador
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**INCOMING STRUCTURAL RELATIONSHIPS**

Aggregation from PoolMinado to NodoMinador
[ Direction is 'Source -> Destination'. ]

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) pruebasUnidadNodoMineria Target: Public (Class) NodoMinador

**OPERATIONS**

__init__ (self : var , direccion : var , puerto : var , descripcion : var , usuariosMineros : var ) : Public

CreaciÃ³n de un nodo para la minerÃa de blockchain :param _direcciÃ³n: DirecciÃ³n ip del nodo minero :param _puerto: puerto TCP/UDP donde se publica el servicio :param _descripcion: informaciÃ³n adicional del nodo minador :param _fechaCreacion: fecha de adiciÃ³n del nodo a la red :param _cadenaBloques: Recibe una copia de la cadena de bloques :param _hashRateNodo : Capacidad de resoluciÃ³n de hash por segundo del nodo :param _usuariosMineros: Conjunto de usuarios participantes en el nodo minero

**OPERATIONS**

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

__str__ (self : var ) : Public

Genera un string con la informaciÃ³n de un nodo de minerÃ- a :return: representaciÃ³n en String del nodo

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

actualizarCadena (self : var , nuevaCadena : var ) : Public

El proceso de minerÃ- a requiere que el nodo actualice la blockchain :param: Cadena de bloques actualizada

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

infoCadena (self : var ) : Public

Genera un string con la informaciÃ³n bÃ¡sica de la cadena de bloques :return: una cadena con el reporte del total de nodos de la cadena

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.2.5 PoolMinado

*Class in package 'cadena'*

PoolMinado
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**OUTGOING STRUCTURAL RELATIONSHIPS**

Aggregation from PoolMinado to NodoMinador

[ Direction is 'Source -> Destination'. ]

**INCOMING STRUCTURAL RELATIONSHIPS**

Aggregation from CadenaBloques to PoolMinado

[ Direction is 'Source -> Destination'. ]

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) PoolMinado Target: Public (Class) pruebasUnidadPoolMinado

**OPERATIONS**

__init__ (self : var ) : Public

InicializaciÃ³n del pool de minerÃa :param _nodos: conjunto de nodos mineros :param _hasRate Simula la capacidad de computo del pool de mineria :param _recompensaPorBloque cantidad que se recompensa a los nodos que resuelvan el problema :param _dificultad valor usado para simular el incremento de dificultad del proceso de minado

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

**OPERATIONS**

__str__ (self : var ) : Public

Genera un string con la informaciÃ³n del pool de minerÃ- a :return: representaciÃ³n en String del nodo
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

adicionarMinero (self : var , minador : var ) : Public

Agrega un nodo minero al conjunto de nodos minadores de la blockchain. Si el nodo adicionado aumenta el hashrate del pool ylo iguala o supera el 60% de la dificultad, esta Ã°ltima se incrementa en un 20%
@param minador: Nuevo nodo minero
@return True si el nodo de mineria se puede adicionar, False en otro caso
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

esPrimo (numero : var ) : Public

Determina si un nÃ°mero es primo
@param numero, NÃ°mero que necesitamos determinar si es primo
@return True si numero es primo, False en otro caso

Properties:
decorators = @staticmethod
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

minadoBloque (nodoMinero : var , transacciones : var ) : Public

Realiza el proceso de minado de un nuevo bloque en la cadena
@param nodoMinero que realiza el intento de minar el bloque
@param transacciones: Conjunto de Transacciones del nuevo bloque

Properties:
decorators = @staticmethod
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

minarBloque (self : var , nodoMinero : var , numeroNodo : var , output : var ) : Public

Algoritmo que realiza el minado de un bloque para un nodo en especifico.
@param nodoMinero, Nodo minero que realizarÃ¡ la prueba de trabajo
@param numeroNodo, identificador entero del nodo minero en el pool de minerÃa
@param output, ParÃ¡metro de salida donde se conserva la informaciÃ³n de la prueba de trabajo ejecutada por el iÃ©simo nodo minero.
@return minadoNuevo Valor de minado obtenido para el nuevo bloque
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

pruebaDeTrabajo (indiceUltimoMinado : var ) : Public

Algoritmo de minado por prueba de trabajo. En esta versiÃ³n se implementa una prueba basada en nÃ°meros primos, en la cual el valor de minado del bloque actual y del Ã°ltimo bloque de la cadena debe ser un nÃ°mero primo.
@param indiceUltimoMinado, Valor generado por la minerÃa para el Ã°ltimo bloque de la cadena
@return minadoNuevo Valor de minado obtenido para el nuevo bloque

Properties:
decorators = @staticmethod
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

sincronizarCadena (self : var , nuevaCadena : var ) : Public

**OPERATIONS**

Realiza una sincronizaciÃ³n de la cadena actualizada a todos los mineros
@param nuevaCadena, Cadena con el nuevo bloque
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

torneoPorNuevoBloque (self : var , transacciones : var ) : Public

Realiza un torneo para la adjudicaciÃ³n del nuevo bloque a un nodo minero. El nodo ganador serÃ¡ el que realice mayor trabajo.
@param transacciones, Bloque de transacciones que tendrÃ¡ el nuevo bloque de la cadena
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.2.6 Transaccion

*Class in package 'cadena'*

Transaccion
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**INCOMING STRUCTURAL RELATIONSHIPS**

Aggregation from Bloque to Transaccion
[ Direction is 'Source -> Destination'. ]

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) Transaccion Target: Public (Class) pruebasUnidadTransaccion

**OPERATIONS**

__init__ (self : var , emisor : var , receptor : var , cantidad : var ) : Public

CreaciÃ³n de una transacciÃ³na :param emisor: Hash de quiÃ©n envÃa :param receptor: Hash de quiÃ©n recibe :param cantidad: valor de la transacciÃ³n
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

__str__ (self : var ) : Public

Genera un string con la informaciÃ³n de la transacciÃ³n :return: representaciÃ³n en String de una transacciÃ³n
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

getCantidad (self : var ) : Public

Obtiene el valor de la transacciÃ³n :return: Valor de la transacciÃ³n
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

getEmisor (self : var ) : Public

Obtiene el valor actual del hash del emisor :return: Hash del emisor

**OPERATIONS**

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

getReceptor (self : var ) : Public

Obtiene el valor actual del hash del receptor :return: Hash del receptor

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.2.7 pruebasUnidadBloque

*Class in package 'cadena'*

pruebasUnidadBloque
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) pruebasUnidadBloque Target: Public (Class) Bloque

**OPERATIONS**

main () : Public

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.2.8 pruebasUnidadCadena

*Class in package 'cadena'*

pruebasUnidadCadena
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) CadenaBloques Target: Public (Class) pruebasUnidadCadena

**OPERATIONS**

main () : Public

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.2.9 pruebasUnidadNodoMineria

*Class in package 'cadena'*

pruebasUnidadNodoMineria
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) pruebasUnidadNodoMineria Target: Public (Class) NodoMinador

**OPERATIONS**

main () : Public
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.2.10 pruebasUnidadPoolMinado

*Class in package 'cadena'*

pruebasUnidadPoolMinado
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends Thread

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) PoolMinado Target: Public (Class) pruebasUnidadPoolMinado

**OPERATIONS**

main () : Public
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.2.11 pruebasUnidadTransaccion

*Class in package 'cadena'*

pruebasUnidadTransaccion
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**ASSOCIATIONS**

Association (direction: Unspecified)

**ASSOCIATIONS**

Source: Public (Class) Transaccion Target: Public (Class) pruebasUnidadTransaccion

**OPERATIONS**

jsonDefault (object : var ) : Public

Properties:
decorators = @staticmethod
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

main () : Public
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

## 1.2.2.3 cliente

*Package in package '2. Diagrama de Clases'*

cliente
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 09/07/2018
Alias

### 1.2.2.3.1 3.2 Diagrama detallado cliente diagram

*Class diagram in package 'cliente'*

3.2 Diagrama detallado cliente
Version 1.0
blockchain created on 17/06/2018. Last modified 09/07/2018

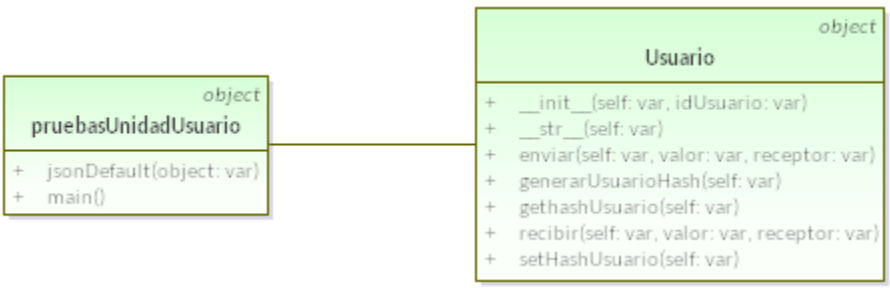


Figure 5: 3.2 Diagrama detallado cliente

### 1.2.2.3.2 Usuario

*Class in package 'cliente'*

Usuario
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) pruebasUnidadUsuario

Target: Public (Class) Usuario

**OPERATIONS**

__init__ (self : var , idUsuario : var ) : Public

CreaciÃ³n de un usuario en el sistema :param _idUsuario Identificador entero del usuario :param hashUsuario: CÃ³digo Hash del usuario :param saldo: Saldo del usuario :param marcaTiempo: Instante de tiempo de creaciÃ³n del usuario

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

__str__ (self : var ) : Public

**OPERATIONS**

Genera un string con la informaciÃ³n del usuario :return: representaciÃ³n en String del usuario
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

enviar (self : var , valor : var , receptor : var ) : Public

:param receptor: hash del usuario receptor de la transferencia :param valor: Cantidad que se desea transferir :return: La nueva transacciÃ³n que serÃ¡ enviada a validaciÃ³n
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

generarUsuarioHash (self : var ) : Public

Genera Hash la informaciÃ³n del bloque :return: Hash construido a partir de la informaciÃ³n del usuario
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

gethashUsuario (self : var ) : Public

Obtiene el valor actual del hash del usuario :return: Hash del bloque
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

recibir (self : var , valor : var , receptor : var ) : Public

:param receptor: hash del usuario receptor de la transferencia :param valor: Cantidad que se ha recibido
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

setHashUsuario (self : var ) : Public
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.3.3 pruebasUnidadUsuario

*Class in package 'cliente'*

pruebasUnidadUsuario
Version 1.0 Phase 1.0 Proposed
blockchain created on 17/06/2018. Last modified 17/06/2018
Extends object

**ASSOCIATIONS**

Association (direction: Unspecified)

Source: Public (Class) pruebasUnidadUsuario Target: Public (Class) Usuario

**OPERATIONS**

jsonDefault (object : var ) : Public

Properties:
decorators = @staticmethod
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

main () : Public

| OPERATIONS |
| --- |
| [ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

# 1.2.2.4 simulador

*Package in package '2. Diagrama de Clases'*

simulador
Version 1.0 Phase 1.0 Proposed
ENVY created on 09/07/2018. Last modified 09/07/2018
Alias

## 1.2.2.4.1 3.3 Diagrama detallado simulador diagram

*Class diagram in package 'simulador'*

3.3 Diagrama detallado simulador
Version 1.0
ENVY created on 09/07/2018. Last modified 09/07/2018

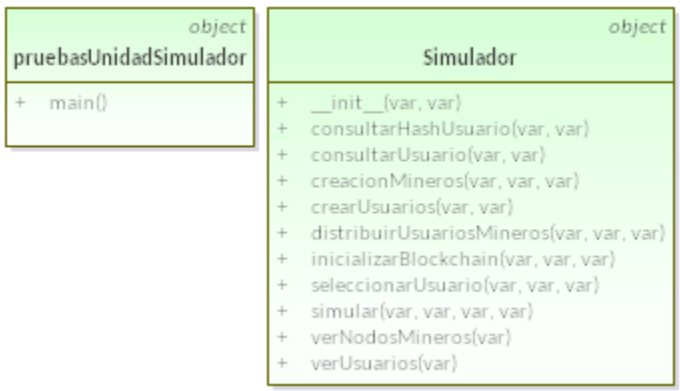


Figure 6: 3.3 Diagrama detallado simulador

## 1.2.2.4.2 Simulador

*Class in package 'simulador'*

Simulador
Version 1.0 Phase 1.0 Proposed
ENVY created on 09/07/2018. Last modified 09/07/2018
Extends object

**OPERATIONS**

__init__ (self : var , nombre : var ) : Public

CreaciÃ³n del escenario final de simulaciÃ³n :param _nombre Nombre Asignado al escenario :param _usuarios Conjunto de usuarios en la simulaciÃ³n :param _blockchain Cadena de Bloques :param _pool Conjunto de nodos mineros de la cadena
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

consultarHashUsuario (self : var , idUsuario : var ) : Public

Retorna el Hash de un usuario dado su nÃºmero de identificaciÃ³n :param idUsuario identificador entero de cada usuario en el sistema :return Hash de usuario que corresponde al identificador dado

Properties:

**OPERATIONS**

ea_guid = {6DF6F9B1-B625-40dc-9600-93CD25CCB976}

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

consultarUsuario (self : var , hashUsuario : var ) : Public

Consulta la informaciÃ³n de un usuario dado su hash :param hashUsuario LLave del diccionario :return usuario que corresponde con la llave dada

Properties:

ea_guid = {0C9F8C53-0611-4260-8A77-9CA712EBE8CC}

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

creacionMineros (self : var , cantidadMineros : var , usuariosNodo : var ) : Public

Adiciona los nodos mineros a la blockchain. Cada nodo minero tendrÃ¡ asociado un grupo de usuarios, quienÃ©s recibirÃ¡n recompensa por su esfuerzo de minado. :param cantidadMineros, Total de nodos Mineros para la cadena :param usuariosNodo, DistribuciÃ³n de usuarios mineros para cada nodo Minero

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

crearUsuarios (self : var , cantidadUsuarios : var ) : Public

Inicializa el conjunto de usuarios de la cadena de bloques :param cantidadUsuarios: Conjunto inicial de Usuarios en la SimulaciÃ³n

Properties:

ea_guid = {C4974C29-34C2-4af6-9DBC-C4ACDB022CA7}

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

distribuirUsuariosMineros (self : var , maxUsuarios : var , maxMineros : var ) : Public

Permite distribuir de manera uniforme los usuarios a los nodos mineros :param maxMineros: Conjunto de Nodos mineros de la blockchain :param maxUsuarios: Total de usuarios en el sistema :return arreglo con los usuarios distribuidos para cada nodo minero

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

inicializarBlockchain (self : var , maxUsuarios : var , cantidadMineros : var ) : Public

CreaciÃ³n del bloque gÃ©nesis de la cadena y asignaciÃ³n de nodos mineros :param cantidadMineros, Total de nodos Mineros para la cadena :param maxUsuarios: Total de usuarios en el sistema

Properties:

ea_guid = {5D0CE9D1-D370-42fc-905D-5D37B6B5A7F8}

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

seleccionarUsuario (self : var , cantidadPosibleReceptores : var , diferentea : var ) : Public

print("Torneo " + str(i) + " " + str(segundos) + " microseconds");

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

simular (self : var , maxTransaccionesPorBloque : var , totalBloquesCaena : var , cantidadUsuarios : var ) : Public

Realiza una simulaciÃ³n de evoluciÃ³n de la blockchain con los parÃ¡metros dados. :param cantidadUsuarios, Total de usuarios activos en la cadena :param cantidadMineros, Total de nodos Mineros para la cadena :param usuariosNodo, DistribuciÃ³n de usuarios mineros para cada nodo Minero

Properties:

ea_guid = {3CF6CA39-BA5C-4115-A9F3-BE0A7FC40F00}

**OPERATIONS**

[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

verNodosMineros (self : var ) : Public

Consulta el total de nodos mineros registrados en la cadena de bloques :return pool de mineros de la blockchain

Properties:
ea_guid = {A73B3E02-C1F3-480a-8755-F203EAC14FBC}
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

verUsuarios (self : var ) : Public

Consulta el total de usuarios registrados en un momento dado en la cadena de bloques :return RepresentaciÃ³n en string del conjunto de Usuarios en la SimulaciÃ³n

Properties:
ea_guid = {419E85C7-3FDF-4cb5-BF1A-E3AE0B5A3495}
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

### 1.2.2.4.3 pruebasUnidadSimulador

*Class in package 'simulador'*

pruebasUnidadSimulador
Version 1.0 Phase 1.0 Proposed
ENVY created on 09/07/2018. Last modified 09/07/2018
Extends object

**OPERATIONS**

main () : Public
[ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ]

# 1.2.34. Diagrama de Objetos

*Package in package 'Diagramas Estructurales'*

4. Diagrama de Objetos
Version 1.0 Phase 1.0 Proposed
ENVY created on 09/07/2018. Last modified 09/07/2018
Alias

## 1.2.3.1 4. Diagrama de Objetos diagram

*Object diagram in package '4. Diagrama de Objetos'*

4. Diagrama de Objetos
Version 1.0
ENVY created on 09/07/2018. Last modified 09/07/2018

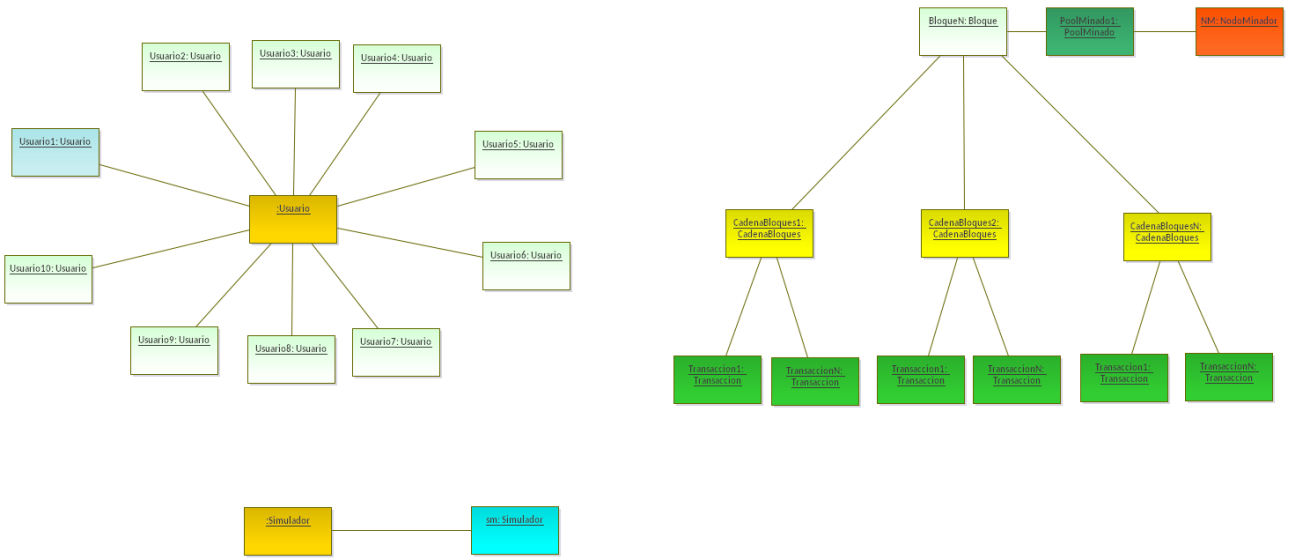


Figure 7: 4. Diagrama de Objetos

# 1.2.45. Diagrama de componentes

*Package in package 'Diagramas Estructurales'*

5. Diagrama de componentes
Version 1.0 Phase 1.0 Proposed
ENVY created on 09/07/2018. Last modified 09/07/2018
Alias

## 1.2.4.1 5. Diagrama de componentes diagram

*Component diagram in package '5. Diagrama de componentes'*

5. Diagrama de componentes
Version 1.0
ENVY created on 09/07/2018. Last modified 10/07/2018

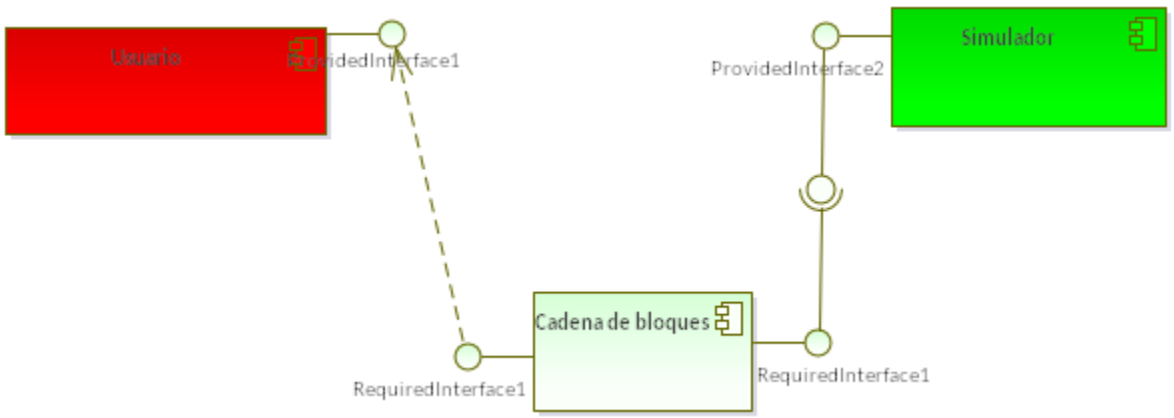


Figure 8: 5. Diagrama de componentes

# 1.2.56. Diagrama de despliegue

*Package in package 'Diagramas Estructurales'*

6. Diagrama de despliegue
Version Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

## 1.2.5.1 6. Diagrama de despliegue diagram

*Deployment diagram in package '6. Diagrama de despliegue'*

6. Diagrama de despliegue
Version 1.0
ENVY created on 10/07/2018. Last modified 10/07/2018

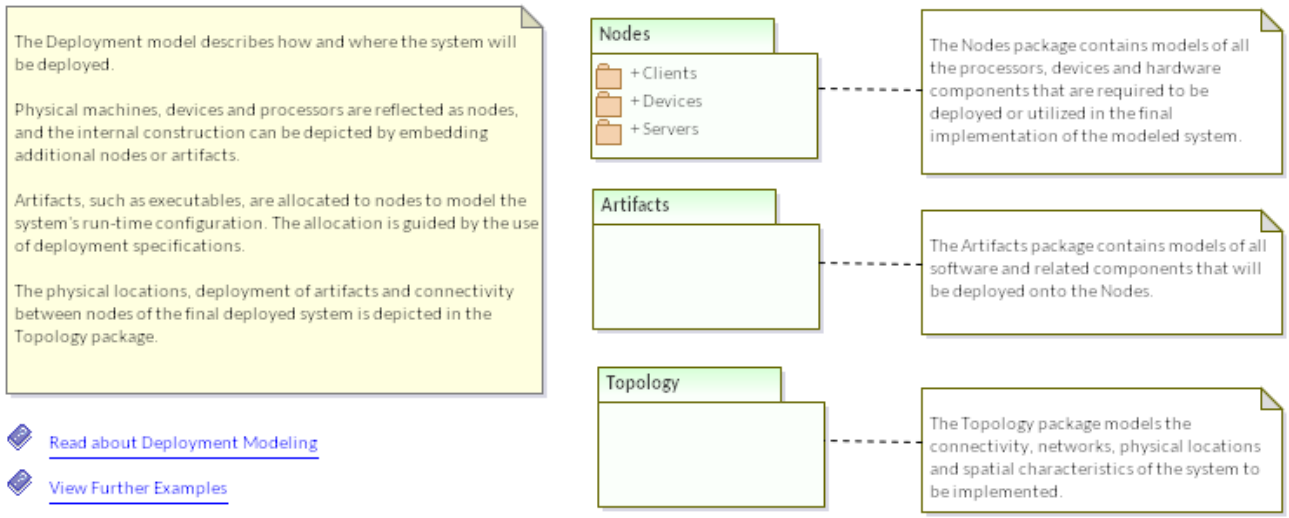


Figure 9: 6. Diagrama de despliegue

## 1.2.5.2 Nodes

*Package in package '6. Diagrama de despliegue'*

Nodes
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

### 1.2.5.2.1 Nodes diagram

*Deployment diagram in package 'Nodes'*

Nodes
Version 1.0

blockchain created on 10/07/2018. Last modified 10/07/2018



Figure 10: Nodes

#### 1.2.5.2.2 Clients

*Package in package 'Nodes'*

Clients
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

##### 1.2.5.2.2.1 Clients diagram

*Deployment diagram in package 'Clients'*

Clients
Version 1.0
blockchain created on 10/07/2018. Last modified 10/07/2018



Figure 11: Clients

### 1.2.5.2.3 Devices

*Package in package 'Nodes'*

Devices
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

#### 1.2.5.2.3.1 Devices diagram

*Deployment diagram in package 'Devices'*

Devices
Version 1.0
blockchain created on 10/07/2018. Last modified 10/07/2018

Figure 12: Devices

### 1.2.5.2.4 Servers

*Package in package 'Nodes'*

Servers
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

#### 1.2.5.2.4.1 Servers diagram

*Deployment diagram in package 'Servers'*

Servers
Version 1.0
blockchain created on 10/07/2018. Last modified 10/07/2018



Figure 13: Servers

### 1.2.5.3 Artifacts

*Package in package '6. Diagrama de despliegue'*

Artifacts
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

#### 1.2.5.3.1 Artifacts diagram

*Deployment diagram in package 'Artifacts'*

Artifacts
Version 1.0
blockchain created on 10/07/2018. Last modified 10/07/2018

Figure 14: Artifacts

## 1.2.5.4 Topology

*Package in package '6. Diagrama de despliegue'*

Topology
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

### 1.2.5.4.1 Network diagram

*Deployment diagram in package 'Topology'*

Network
Version 1.0
blockchain created on 10/07/2018. Last modified 10/07/2018

Figure 15: Network

# 1.3 Diagramas de comportamiento

*Package in package 'cadenaDeBloques_RPava'*

Diagramas de comportamiento
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

## 1.3.18. Diagramas de casos de uso

*Package in package 'Diagramas de comportamiento'*

8. Diagramas de casos de uso
Version Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

### 1.3.1.1 8. Diagramas de casos de uso diagram

*Use Case diagram in package '8. Diagramas de casos de uso'*

8. Diagramas de casos de uso
Version 1.0
blockchain created on 31/05/2018. Last modified 10/07/2018


Figure 16: 8. Diagramas de casos de uso

### 1.3.1.2 Actores

*Package in package '8. Diagramas de casos de uso'*

Actores
Version 1.0 Phase 1.0 Proposed
blockchain created on 31/05/2018. Last modified 10/07/2018
Alias

### 1.3.1.2.1 Actores diagram

*Use Case diagram in package 'Actores'*

Actores
Version 1.0
blockchain created on 31/05/2018. Last modified 10/07/2018


Figure 17: Actores

## 1.3.1.3 Primary Use Cases

*Package in package '8. Diagramas de casos de uso'*

Primary Use Cases
Version 1.0 Phase 1.0 Proposed
blockchain created on 31/05/2018. Last modified 10/07/2018
Alias

### 1.3.1.3.1 Primary Use Cases diagram

*Use Case diagram in package 'Primary Use Cases'*

Primary Use Cases
Version 1.0
blockchain created on 31/05/2018. Last modified 10/07/2018

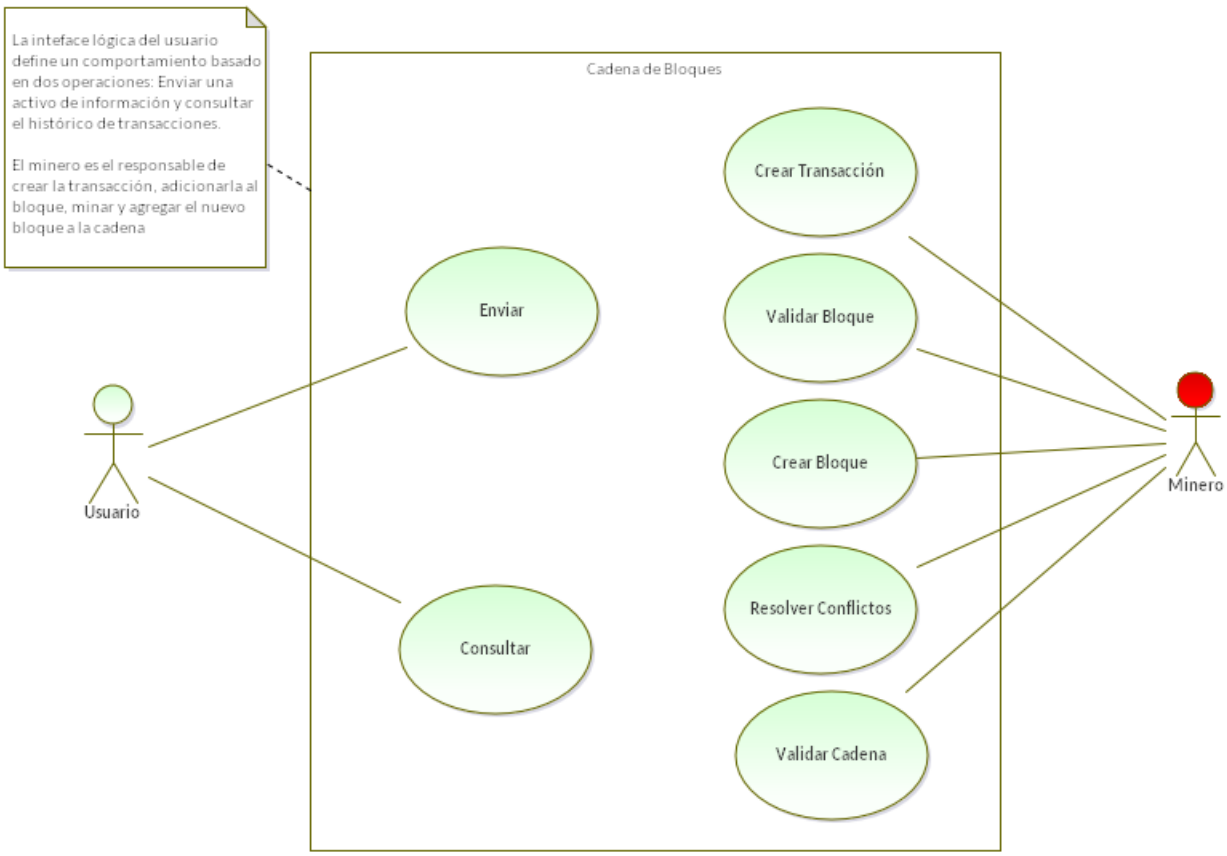


Figure 18: Primary Use Cases

# 1.3.29. Diagramas de Secuencia

*Package in package 'Diagramas de comportamiento'*

9. Diagramas de Secuencia
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

## 1.3.2.1 9.1 Diagrama de Secuencia pruebasUnidadUsuario diagram

*Interaction diagram in package '9. Diagramas de Secuencia'*

9.1 Diagrama de Secuencia pruebasUnidadUsuario
Version 1.0
ENVY created on 10/07/2018. Last modified 10/07/2018

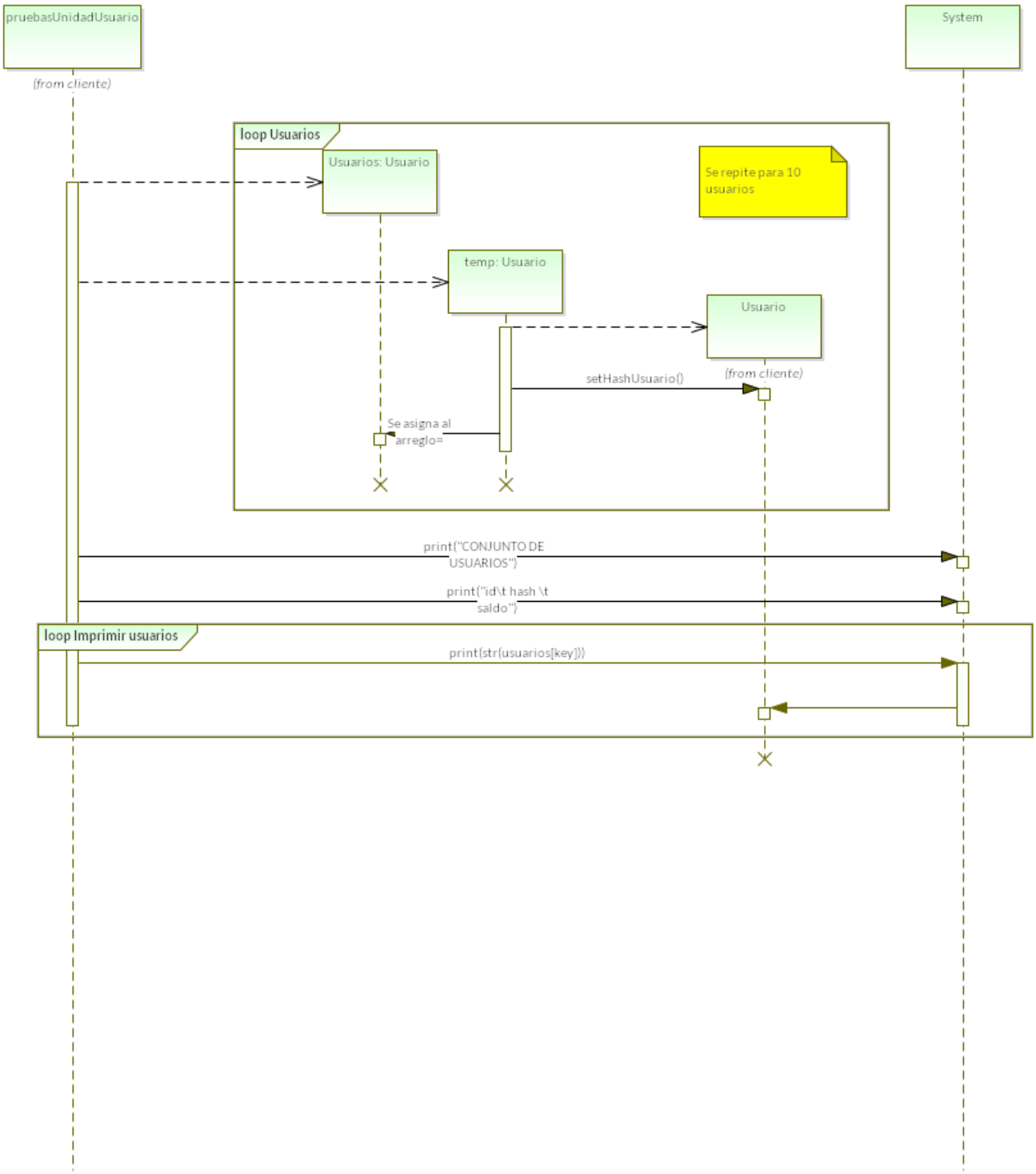


Figure 19: 9.1 Diagrama de Secuencia pruebasUnidadUsuario

## 1.3.2.2 9.2 Diagramas de Secuencia pruebasUnidadSimulador diagram

*Interaction diagram in package '9. Diagramas de Secuencia'*

9.2 Diagramas de Secuencia pruebasUnidadSimulador
Version 1.0
ENVY created on 10/07/2018. Last modified 10/07/2018

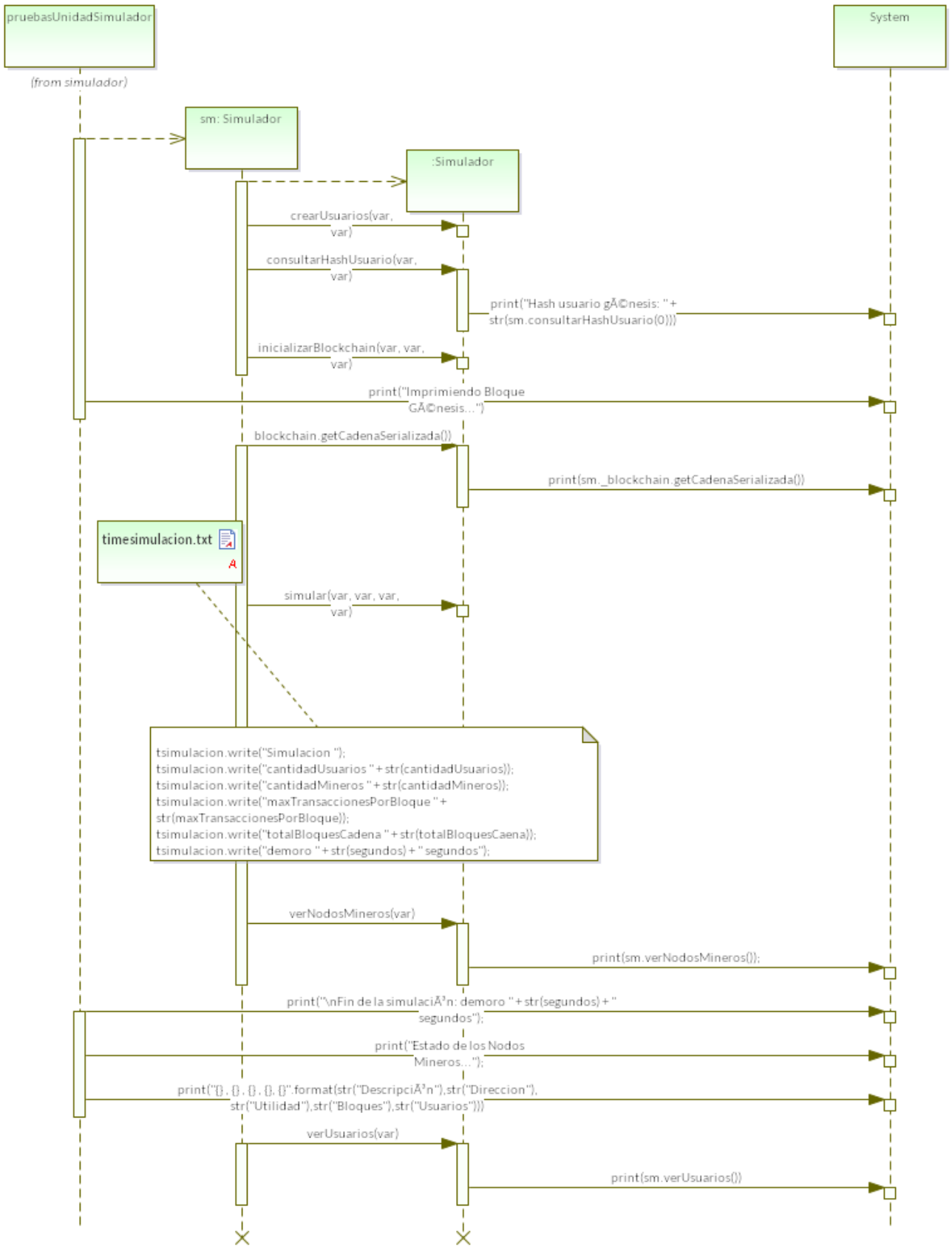


Figure 20: 9.2 Diagramas de Secuencia pruebasUnidadSimulador

## 1.3.2.3 9.3 Diagramas de Secuencia pruebasUnidadTransaccion diagram

*Interaction diagram in package '9. Diagramas de Secuencia'*

9.3 Diagramas de Secuencia pruebasUnidadTransaccion
Version 1.0
ENVY created on 10/07/2018. Last modified 10/07/2018

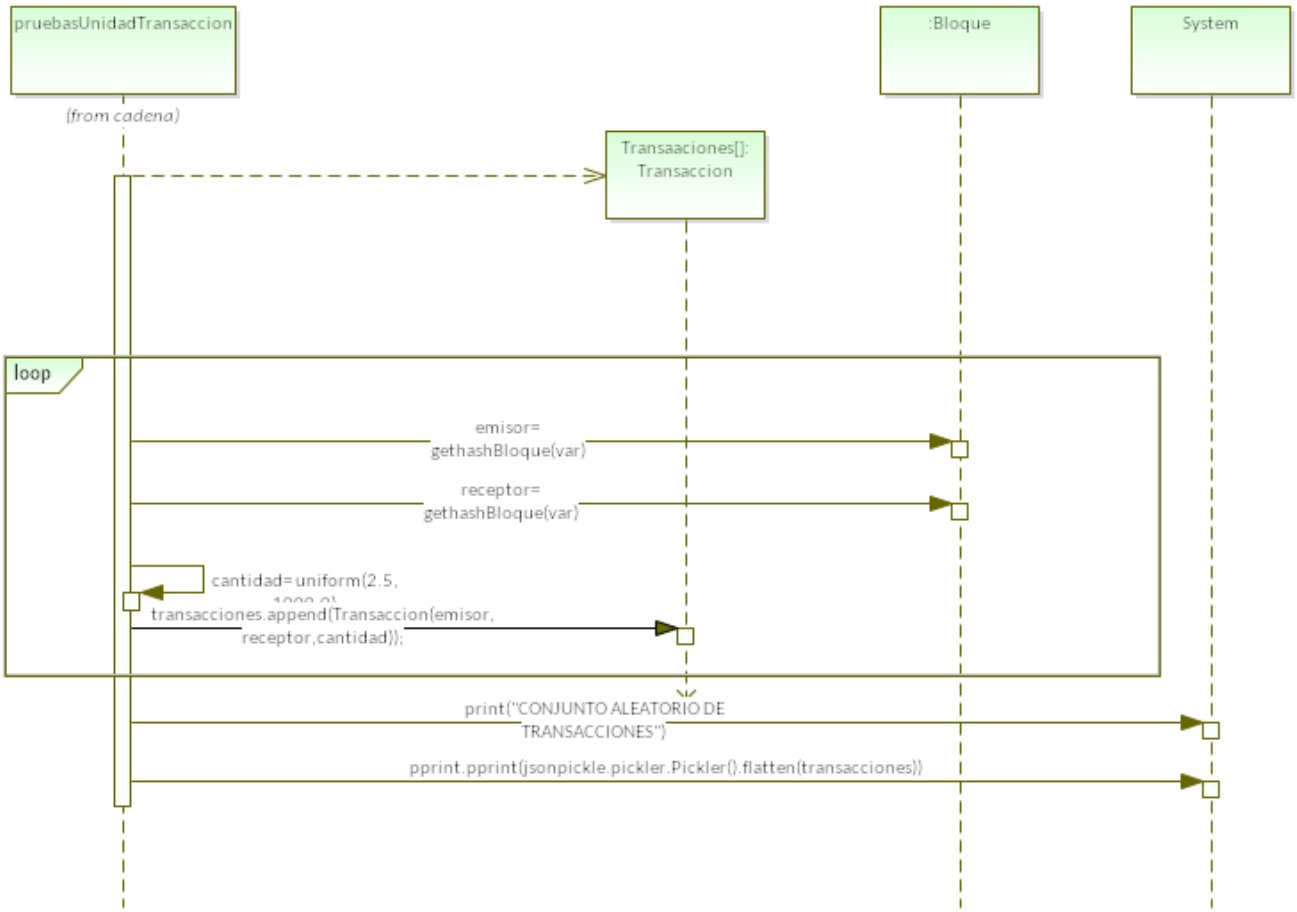


Figure 21: 9.3 Diagramas de Secuencia pruebasUnidadTransaccion

# 1.3.310. Diagramas de Comunicación

*Package in package 'Diagramas de comportamiento'*

10. Diagramas de Comunicación
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

## 1.3.3.1 10.1 Diagramas de Comunicación Usuario diagram

*Communication diagram in package '10. Diagramas de Comunicación'*

10.1 Diagramas de Comunicación Usuario
Version 1.0
ENVY created on 10/07/2018. Last modified 10/07/2018

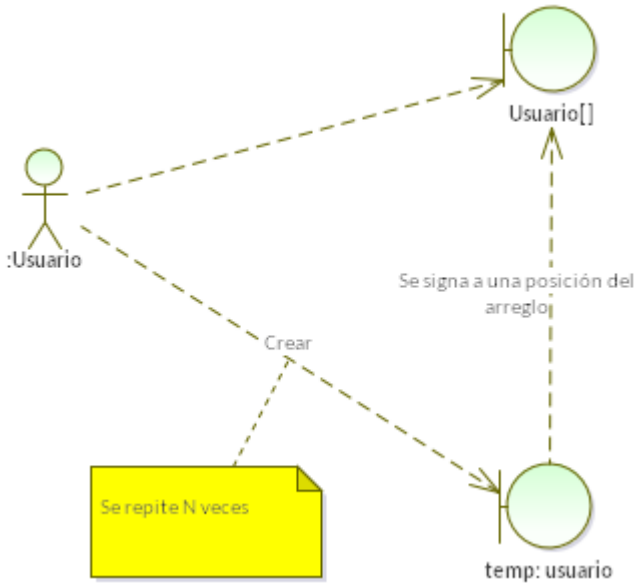


Figure 22: 10.1 Diagramas de Comunicación Usuario

## 1.3.3.2 10.2 Diagramas de Comunicación pruebasUnidadSimulador diagram

*Communication diagram in package '10. Diagramas de Comunicación'*

10.2 Diagramas de Comunicación pruebasUnidadSimulador
Version 1.0
ENVY created on 10/07/2018. Last modified 10/07/2018

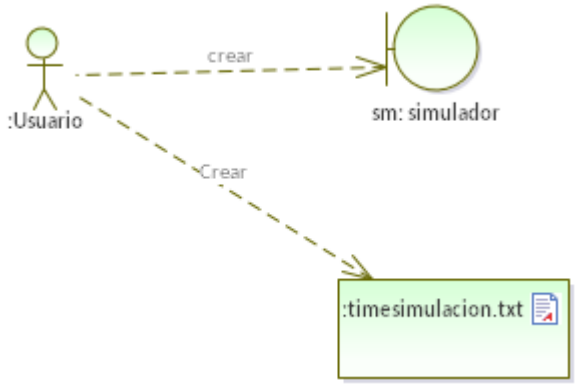


Figure 23: 10.2 Diagramas de Comunicación pruebasUnidadSimulador

## 1.3.3.3 10.3 Diagramas de Comunicación pruebaUnidadTrasaccion diagram

*Communication diagram in package '10. Diagramas de Comunicación'*

10.3 Diagramas de Comunicación pruebaUnidadTrasaccion
Version 1.0
ENVY created on 10/07/2018. Last modified 10/07/2018

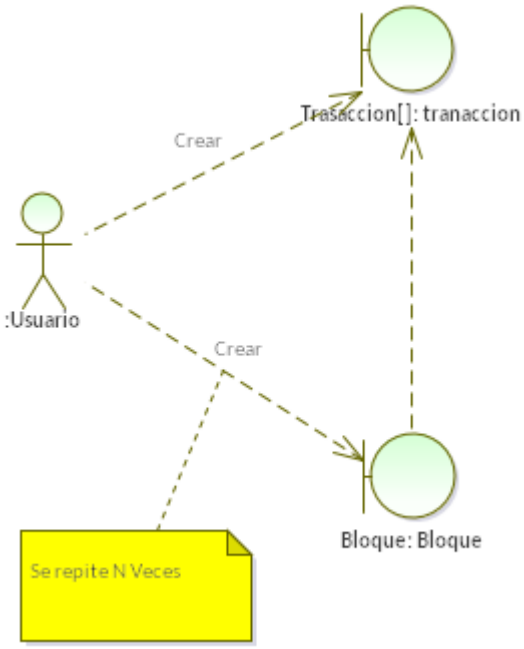


Figure 24: 10.3 Diagramas de Comunicación pruebaUnidadTrasaccion

# 1.4 Extendidos

*Package in package 'cadenaDeBloques_RPava'*

Extendidos
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

## 1.4.115. Requisitos de requerimientos

*Package in package 'Extendidos'*

15. Requisitos de requerimientos
Version 1.0 Phase 1.0 Proposed
ENVY created on 10/07/2018. Last modified 10/07/2018
Alias

### 1.4.1.1 15. Requisitos requerimientos diagram

*Requirements diagram in package '15. Requisitos de requerimientos'*

15. Requisitos requerimientos
Version 1.0
ENVY created on 10/07/2018. Last modified 10/07/2018

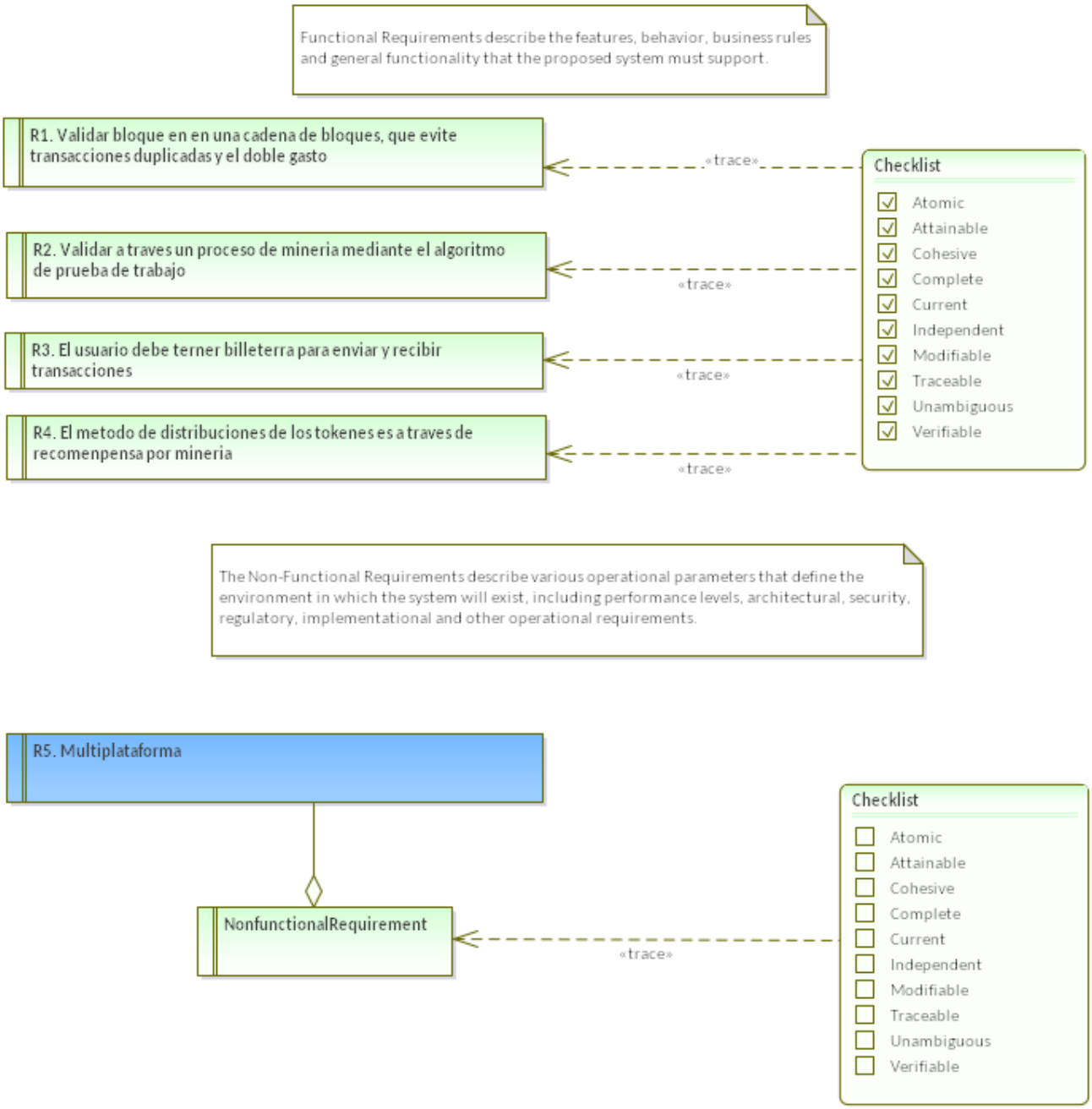


Figure 25: 15. Requisitos requerimientos

# BlockchainSimulation

March 20, 2019

Clase Transacción

In [7]:
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Define los elementos que componen una transacción.
"""

class Transaccion(object):

    def __init__(self, emisor, receptor, cantidad):
        """
        Creación de una transaccióna
        :param emisor: Hash de quién envía
        :param receptor:  Hash de quién recibe
        :param cantidad: valor de la transacción
        """
        self._emisor = emisor
        self._receptor = receptor
        self._cantidad = cantidad

    def getEmisor(self):
        """
        Obtiene el valor actual del hash del emisor
        :return: Hash del emisor
        """
        return self._emisor;

    def getReceptor(self):
        """
        Obtiene el valor actual del hash del receptor
        :return: Hash del receptor
        """
        return self._receptor;

    def getCantidad(self):
        """
        Obtiene el valor de la transacción
```

```
        :return: Valor de la transacción
        """
        return self._cantidad;

    def __str__(self):
        """
        Genera un string con la información de la transacción
        :return: representación en String de una transacción
        """
        return "{} , {} , {}".format(self._emisor, self._receptor, self._cantidad)
```

Prueba Unitaria Clase Transacción

In [2]:
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from Transaccion import Transaccion
from Bloque import Bloque
from random import uniform
from random import random
import jsonpickle
import jsonpickle.backend
import jsonpickle.handlers
import pprint

"""
En esta clase se realizan pruebas de unidad
"""
class pruebasUnidadTransaccion(object):

    @staticmethod
    def jsonDefault(object):
        return object.__dict__

    def main():
       transacciones = [];
       #Se construye una conjunto aleatorio de transacciones
       for i in range(5):
           emisor = Bloque.generarBloqueHash(random())
           receptor = Bloque.generarBloqueHash(random())
           cantidad = uniform(2.5, 1000.0)
           transacciones.append(Transaccion(emisor, receptor,cantidad));

       print("CONJUNTO ALEATORIO DE TRANSACCIONES")
       pprint.pprint(jsonpickle.pickler.Pickler().flatten(transacciones))

    if __name__ == "__main__":
        main()
```

```
CONJUNTO ALEATORIO DE TRANSACCIONES
[{'_cantidad': 203.66456670821438,
  '_emisor': 'cfa82607adac99aaabae7cec8b9b4b719d871e987a00ebb7d9c5cc1ee93a6a76',
  '_receptor': '5024b513039bbf695b6f0a71fd8ed7bd865a4cc72d75d566a5fff597e34dab19',
  'py/object': 'Transaccion.Transaccion'},
 {'_cantidad': 834.4563379913702,
  '_emisor': '9750962eef171090b7e20898752b770d0ba241ef7e6ecb0e05359964cbec8a43',
  '_receptor': 'd66d52d6a2a2cefca7d98b6c957700fd41ff8b3a3d910698305fd16f7fe68d26',
  'py/object': 'Transaccion.Transaccion'},
 {'_cantidad': 161.6765922669576,
  '_emisor': 'ecda4062b102f1fad39bdc8bbbc081fc138f97bb62ed48e71c84b434d4c96e6c',
  '_receptor': '36ce88519a21b62d93d9f69d21f9013b076a35b50f89b277db1701a1bbfb8734',
  'py/object': 'Transaccion.Transaccion'},
 {'_cantidad': 798.8531049411085,
  '_emisor': '153441ec7145aa14388a54bfa84ff0dcf2422cb4d0e0f074704256312c406a7d',
  '_receptor': 'cf5252f833c3f5a9e173af1bb600b15a6cb80ba823196bbdf5bed2f735d5b2ae',
  'py/object': 'Transaccion.Transaccion'},
 {'_cantidad': 440.52259568593405,
  '_emisor': '9a27161fe22b29060eb2fe4afa3c83be1320432ae5b13d32d6037342ad189fe7',
  '_receptor': '28114126f9ac3f36fc36c70b5a0f4b4372406d0f11d74d78b701951d57f386fd',
  'py/object': 'Transaccion.Transaccion'}]
```

Clase Usuario

```
In [3]: #!/usr/bin/env python3
        # -*- coding: utf-8 -*-
        import time
        import hashlib
        from Transaccion import Transaccion
        """
        """


        class Usuario(object):

            def __init__(self, idUsuario):
                """
                Creación de un usuario en el sistema
                :param _idUsuario Identificador entero del usuario
                :param hashUsuario:  Código Hash del usuario
                :param saldo: Saldo del usuario
                :param marcaTiempo: Instante de tiempo de creación del usuario
                """
                self._idUsuario  = idUsuario;
                self._saldo = 0;
                self._hashUsuario = 0;
                self._marcaTiempo = time.time();
```

```
        """
        Usuario Génesis toma 100 de saldo
        """
        if self._idUsuario == 0:
            self._saldo = 100;


    def gethashUsuario(self):
        """
        Obtiene el valor actual del hash del usuario
        :return: Hash del bloque
        """
        return self._hashUsuario;

    def setHashUsuario(self):
        self._hashUsuario = self.generarUsuarioHash();


    def generarUsuarioHash(self):
        """
        Genera Hash la información del bloque
        :return: Hash construido a partir de la información del usuario
        """
        return hashlib.sha256(str(self).encode()).hexdigest()


    def __str__(self):
        """
        Genera un string con la información del usuario
        :return: representación en String del usuario
        """
        return "{}, {} , {}".format(self._idUsuario, self._hashUsuario, self._saldo)

    def enviar(self, valor, receptor):
        """
        :param receptor: hash del usuario receptor de la transferencia
        :param valor: Cantidad que se desea transferir
        :return: La nueva transacción que será enviada a validación
        """
        if valor < self._saldo:
            self._saldo -= valor;
            self.recibir(valor, receptor);
            return Transaccion(self._hashUsuario, receptor,valor);

    def recibir(self, valor, receptor):
        """
        :param receptor: hash del usuario receptor de la transferencia
        :param valor: Cantidad que se ha recibido
```

```
            """
            receptor._saldo += valor;
```

Prueba Unitaria Clase Usuario

```
In [4]: #!/usr/bin/env python3
        # -*- coding: utf-8 -*-
        from Usuario import Usuario


        """
        En esta clase se realizan pruebas de unidad
        """
        class pruebasUnidadUsuario(object):

            @staticmethod
            def jsonDefault(object):
                return object.__dict__

            def main():
               usuarios = {};
               for i in range(10):
                   temp = Usuario(i);
                   temp.setHashUsuario();
                   usuarios[temp.gethashUsuario] = temp;

               print("CONJUNTO DE USUARIOS")
               print("id\t hash \t saldo");
               for key in usuarios:
                   print(str(usuarios[key]));


            if __name__ == "__main__":
                main()
```

```
CONJUNTO DE USUARIOS
id         hash          saldo
0, ae1bc1b04561fac552570296dfe154094dc4df4b6d4970ca81ec9507046375b2 , 100
1, 9d5eab463378fe259f33d07590aa6a913a1d099d5a276e353a955cb2da4311f2 , 0
2, b4aec9b924aa7f7d83851c90a7f5142be0cec41c7c46cff2fb89bf0bbf055f67 , 0
3, 0890bee71d46a465a64625124eee9a424a66908e11ce283ed4afaf5a9551cef0 , 0
4, 1206cfe53ef4288f00f4c53f5377601c07bdc95f6a5249b95183cc5db54ad93a , 0
5, 5d4a62b58203901f4e7502ccbdd5a2e52a234bb44545549fd5c87edef9202363 , 0
6, 11446965a28709ff7577c812099db7093ae7a31f8aa8ebe2a81d15ee876b0f16 , 0
7, 44b70f5a64febf0f1c47390efaa1d824a5503e3831d37b28b42c63f25813cf17 , 0
8, 06aeb9cf521c8e7114647117303ebf837f50bd39b38104ce5e3371bf54d1db61 , 0
9, 371eb3f89a125644770e3395faa25a50d0f7bf6f9fb15b271649b53640ed1a3b , 0
```

Clase Bloque

In [5]:
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import time
import hashlib
import copy
import json
"""
Define la estructura del bloque.
"""

class Bloque(object):

    def __init__(self, indice, minado, hashBloque, transacciones, marcaTiempo=None):
        """
        Inicialización del bloque
        :param indice: Valor entero que representa la posición de cada bloque en la ca
        :param minado: Número entero usado en el proceso de minería
        :param hashBloque:  hash del bloque actual en la cadena
        :param transacciones: Conjunto de transacciones del bloque
        :param marcaTiempo: Instante de tiempo de creación del nuevo bloque
        """
        self._indice = indice
        self._minado = minado
        self._hashBloque = hashBloque
        self._transacciones = copy.copy(transacciones)
        self._marcaTiempo = marcaTiempo or time.time()

    def gethashBloque(self):
        """
        Obtiene el valor actual del hash del bloque
        :return: Hash del bloque
        """
        return self._hashBloque;

    def setHashBloque(self):
        self._hashBloque = Bloque.generarBloqueHash(self)

    @staticmethod
    def generarBloqueHash(self):
        """
        Genera Hash la información del bloque
        :return: Hash construido a partir de la información del bloque
        """
        return hashlib.sha256(str(self).encode()).hexdigest()
```

```
        def __str__(self):
            """
            Genera un string con la información del bloque
            :return: representación en String del bloque
            """
            return "{} , {} , {}  , {}".format(self._indice, self._minado, self._hashBloque

        def __repr__(self):
                return json.JSONEncoder().encode(self)
```

Prueba Unitaria Clase Bloque

```
In [6]: #!/usr/bin/env python3
        # -*- coding: utf-8 -*-
        from Transaccion import Transaccion
        from Bloque import Bloque
        from random import randrange
        from random import uniform
        from random import random
        import jsonpickle
        import jsonpickle.backend
        import jsonpickle.handlers
        import pprint

        """
        En esta clase se realizan pruebas de unidad
        """
        class pruebasUnidadBloque(object):

            def main():
               #Cada bloque de prueba se ha creado con un conjunto aleatorio transacciones
               transacciones = [];
               #Se construye una conjunto aleatorio de transacciones para bloque genésis
               emisor = Bloque.generarBloqueHash(random())
               receptor = Bloque.generarBloqueHash(random())
               cantidad = uniform(2.5, 1000.0)
               transacciones.append(Transaccion(emisor, receptor,cantidad));
               #0 para indicar que el bloque no ha sido minado
               genesis = Bloque(0,0,Bloque.generarBloqueHash(0),transacciones);
               cadena = [genesis];

                #Se construye una pseudocadena con 5 bloques
               for i in range(1,5):
                      transacciones = [];
                      #Se construye una conjunto aleatorio de transacciones
                      for t in range(1+randrange(3)):
                          emisor = Bloque.generarBloqueHash(random())
                          receptor = Bloque.generarBloqueHash(random())
```

```
                        cantidad = uniform(2.5, 1000.0)
                        transacciones.append(Transaccion(emisor, receptor,cantidad));

                    cadena.append(Bloque(i,0,Bloque.generarBloqueHash(cadena[i-1].gethashBlo

            pprint.pprint(jsonpickle.pickler.Pickler().flatten(cadena))

        if __name__ == "__main__":
            main()
```

```
[{'_hashBloque': '5feceb66ffc86f38d952786c6d696c79c2dbc239dd4e91b46729d73a27fb57e9',
  '_indice': 0,
  '_marcaTiempo': 1529247369.137099,
  '_minado': 0,
  '_transacciones': [{'_cantidad': 877.3810119704357,
                      '_emisor': 'a9df00a99c4be24e257cbff30f230f5190b2d5ab36c4277267cfeae5631a
                      '_receptor': '21d5eca5521a35d601c86987fbc3ad932a29278b894b2015580902e894
                      'py/object': 'Transaccion.Transaccion'}],
  'py/object': 'Bloque.Bloque'},
 {'_hashBloque': '5122a1d1bc9d87662dcf5fb870adf8c55faf2ce12fad1bacac2fe7df88172466',
  '_indice': 1,
  '_marcaTiempo': 1529247369.1371431,
  '_minado': 0,
  '_transacciones': [{'_cantidad': 883.2276154566855,
                      '_emisor': '0ed246acb2dbb5b13218ff358c86629997caa3d9b420f41a58dc8a19b545
                      '_receptor': 'eaf44a27bb108fbb343214593237a7d556c4519b03ad8f270f4fe77062
                      'py/object': 'Transaccion.Transaccion'},
                     {'_cantidad': 276.6887479260855,
                      '_emisor': 'e0fc82ae30827fc813242d95aeecd56fbcbdcc11275ff771c85f6df1f62c
                      '_receptor': 'd9c8eb2e75add7b6aa45cf56f753c0a8b02ed13e62f2cbe0650e6c00e1
                      'py/object': 'Transaccion.Transaccion'}],
  'py/object': 'Bloque.Bloque'},
 {'_hashBloque': '3c857ab8e3936f0f125a97a86262fd3d1cd906f4b7acf1df385f1756f7f692b2',
  '_indice': 2,
  '_marcaTiempo': 1529247369.1371617,
  '_minado': 0,
  '_transacciones': [{'_cantidad': 197.136994685999,
                      '_emisor': 'fdad280e38116f37382db3633df35619589d9176cd80b05b5c5ac29a4fce
                      '_receptor': '259885aa40fad971e3fe0a945f15d229f6a9f9b10858ab3edbeda090ea
                      'py/object': 'Transaccion.Transaccion'}],
  'py/object': 'Bloque.Bloque'},
 {'_hashBloque': '3806743a62ad600a453105090db2884c747600ad27576f4e6ba18f2f56cc445b',
  '_indice': 3,
  '_marcaTiempo': 1529247369.1371787,
  '_minado': 0,
  '_transacciones': [{'_cantidad': 14.972984892336266,
                      '_emisor': 'd939634d05ed10d16ec0d1a429bf22dfb4f3dd3a9cebf0f7561949623ba5
                      '_receptor': '489d4d3461255a694e63ca6edfd22be63156b786582073ca09108632f7
```

```
                    'py/object': 'Transaccion.Transaccion'}],
  'py/object': 'Bloque.Bloque'},
 {'_hashBloque': '19870586ed9e6d23573bd38b361f090ec70a0260fbf5b6e9d667f239912ccf94',
  '_indice': 4,
  '_marcaTiempo': 1529247369.1372035,
  '_minado': 0,
  '_transacciones': [{'_cantidad': 64.61594012977609,
                      '_emisor': '55ee9138f8528cb62a3742b8a3c107fe6e83c86a2504f1fd605b56e462b4
                      '_receptor': '4e2620109ea5f93d34900c5694fc0010d8b07f986a179c8582aacdf969
                      'py/object': 'Transaccion.Transaccion'},
                     {'_cantidad': 770.3643679591898,
                      '_emisor': 'fefcb5c9349bc1c439102a5850fbf3deda3772d4a7e0faf2ee9f0e320c7a
                      '_receptor': 'df22b53cfb72b48b4a9a4c1a4b339abe9eb9994ff77c1593a76d6d5587
                      'py/object': 'Transaccion.Transaccion'}],
  'py/object': 'Bloque.Bloque'}]
```

Clase Nodo Minador

In [8]:
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Define la estructura de un nodo responsable de minar la cadena de bloques
"""
import time
import copy

class NodoMinador(object):

    def __init__(self, direccion, puerto, descripcion, usuariosMineros):
        """
        Creación de un nodo para la minería de blockchain
        :param _direcccción: Dirección ip del nodo minero
        :param _puerto:  puerto TCP/UDP donde se publica el servicio
        :param _descripcion: información adicional del nodo minador
        :param _fechaCreacion: fecha de adición del nodo a la red
        :param _cadenaBloques: Recibe una copia de la cadena de bloques
        :param _hashRateNodo : Capacidad de resolución de hash por segundo del nodo
        :param _usuariosMineros: Conjunto de usuarios participantes en el nodo minero
        """
        self._direccion = direccion;
        self._puerto = puerto;
        self._descripcion = descripcion;
        self._utilidad = 0;
        self._cantidadBloquesMinados = 0;
        self._fechaCreacion = time.time();
        self._hashRateNodo = 100;
        self._cadenaBloques = [];
```

```
            self._usuariosMineros = copy.copy(usuariosMineros);


        def actualizarCadena(self, nuevaCadena):
            """
            El proceso de minería requiere que el nodo actualice la blockchain
            :param: Cadena de bloques actualizada
            """
            self._cadenaBloques = copy.copy(nuevaCadena)

        def infoCadena(self):
            """
            Genera un string con la información básica de la cadena de bloques
            :return: una cadena con el reporte del total de nodos de la cadena
            """
            return "La cadena tiene {} bloques".format(str(len(self._cadenaBloques)))

        def __str__(self):
            """
            Genera un string con la información de un nodo de minería
            :return: representación en String del nodo
            """
            return "{} , {} , {} , {} , {}".format(self._descripcion , str(self._direccion)
```

Prueba Unitaria Clase Nodo Minador

```
In [9]: #!/usr/bin/env python3
        # -*- coding: utf-8 -*-
        from CadenaBloques import CadenaBloques
        from NodoMinador import NodoMinador
        from Transaccion import Transaccion
        from Usuario import Usuario
        from Bloque import Bloque
        from random import randrange
        from random import uniform
        from random import random
        import copy

        """
        En esta clase se realizan pruebas de unidad
        """
        class pruebasUnidadNodoMineria(object):
            def main():
                transacciones = [];
                blockchain = CadenaBloques()
                #Se construye una transacción aleatoria para bloque genésis
                emisor = Bloque.generarBloqueHash(random())
```

```
    receptor = Bloque.generarBloqueHash(random())
    cantidad = uniform(2.5, 1000.0)
    transacciones.append(Transaccion(emisor, receptor,cantidad));
    blockchain.bloqueGenesis(transacciones)

    totalUsuarios = 25;
    totalMineros = 5;

    usuarios = {};
    for i in range(totalUsuarios):
        temp = Usuario(i);
        temp.setHashUsuario();
        usuarios[temp.gethashUsuario] = temp;

    #Se construye una conjunto aleatorio de nodos mineros
    usuariosNodo = [];
    usuariosMineros = {};
    contador = totalUsuarios//totalMineros - 1;
    aux = 0;
    for key in usuarios:
        usuariosMineros[key] = usuarios[key];
        if(aux < contador):
           aux += 1;
        else:
            usuariosNodo.append(copy.copy(usuariosMineros));
            aux = 0;
            usuariosMineros = {};

    mineros = [];
    for i in range(len(usuariosNodo)):
        address = "127.0.0." + str(1+randrange(253))
        puerto = 1025+randrange(60000)
        descripcion = "nodo minero número " + str(i)
        nodo = NodoMinador(address, puerto,descripcion,usuariosNodo[i])
        nodo.actualizarCadena(blockchain)
        mineros.append(nodo);

    print("CONJUNTO ALEATORIO DE NODOS MINEROS")
    #pprint.pprint(jsonpickle.pickler.Pickler().flatten(mineros))
    for i in mineros:
        print(i);
        print("Usuarios Mineros en el Nodo");
        for key in i._usuariosMineros:
            print(i._usuariosMineros[key]);
        print("============================================================")


if __name__ == "__main__":
```

```
                main()

CONJUNTO ALEATORIO DE NODOS MINEROS
nodo minero número 0 , 127.0.0.143:45186 , 0 , 0 , 5
Usuarios Mineros en el Nodo
0, ae1bc1b04561fac552570296dfe154094dc4df4b6d4970ca81ec9507046375b2 , 100
1, 9d5eab463378fe259f33d07590aa6a913a1d099d5a276e353a955cb2da4311f2 , 0
2, b4aec9b924aa7f7d83851c90a7f5142be0cec41c7c46cff2fb89bf0bbf055f67 , 0
3, 0890bee71d46a465a64625124eee9a424a66908e11ce283ed4afaf5a9551cef0 , 0
4, 1206cfe53ef4288f00f4c53f5377601c07bdc95f6a5249b95183cc5db54ad93a , 0
============================================================
nodo minero número 1 , 127.0.0.136:31969 , 0 , 0 , 5
Usuarios Mineros en el Nodo
5, 5d4a62b58203901f4e7502ccbdd5a2e52a234bb44545549fd5c87edef9202363 , 0
6, 11446965a28709ff7577c812099db7093ae7a31f8aa8ebe2a81d15ee876b0f16 , 0
7, 44b70f5a64febf0f1c47390efaa1d824a5503e3831d37b28b42c63f25813cf17 , 0
8, 06aeb9cf521c8e7114647117303ebf837f50bd39b38104ce5e3371bf54d1db61 , 0
9, 371eb3f89a125644770e3395faa25a50d0f7bf6f9fb15b271649b53640ed1a3b , 0
============================================================
nodo minero número 2 , 127.0.0.81:20701 , 0 , 0 , 5
Usuarios Mineros en el Nodo
10, d3e0b66ec1e56b92a5c0b1bd7c57540697da963987253413c40b77636707f13c , 0
11, b29492fe4bf566a54a5ac57bfb49cd6441b5cc3dbb19b36804da13b6a7495032 , 0
12, e8a26469e9a9879b4ef23fb12236e77a66b7e1a55b060694537c39fdca4fec8c , 0
13, 526042ccba454c2762af9fefd0d418254372e3d3c9d8605dfa6217d1c5843a61 , 0
14, 7f74614d41796a901e5dd4fc962da90a80b2e0a7a9f3180d968697845d882e55 , 0
============================================================
nodo minero número 3 , 127.0.0.163:56998 , 0 , 0 , 5
Usuarios Mineros en el Nodo
15, c1f330b2fa35be43c2d2e695cc0d73ba2c074247ef4e10a82e4a4eca63307c80 , 0
16, da6a1600010abc6c5a6c9aea97886cf0559fe1ab067110a0dabb3e6c449c3835 , 0
17, 33d772cef882b98fe5495ddc7b1e7b369fb6301372761b36542aa7f2683a0788 , 0
18, 5fafa6531ad8489b778a631289e5d41cc2a6dee01b2d17dab578b9ecc47abdac , 0
19, ce12b6eece7dc590ac7af2b77258ba8c262d5d0b1f89b8aff89914e6eb3a86b9 , 0
============================================================
nodo minero número 4 , 127.0.0.232:41227 , 0 , 0 , 5
Usuarios Mineros en el Nodo
20, b252c76b28f15c2fce8bfb7b0153ce87336a0789e16c40fed7faa04df2fc864f , 0
21, 839604719f7a03403bda1bea57e369f0ee226a4aa8a092aee337252b330e7ab8 , 0
22, 18fa0626ea0690666f53c910244389a32c4fa7a13453987b094ddd813ec1904d , 0
23, 192990181fd61e86ffa1e28ec8741a76c1640e816bd75b00e8e7e4499b12c1af , 0
24, 1d918ed33bf67966f2ab8ea55bb3707f2ead718e01b6881c5ec03882587760e1 , 0
============================================================
```

Clase Cadena de Bloques

```
In [10]: #!/usr/bin/env python3
         # -*- coding: utf-8 -*-
```

```
"""
En esta clase almacena la cadena de bloques
"""

from Bloque import Bloque
from PoolMinado import PoolMinado
import json
import copy

class CadenaBloques(object):

    def __init__(self):
        """
        Inicialización de la blockchain
        :param _cadena: Estructura de datos tipo array para almacenar la cadena de bl
        """
        self._cadena = []


    def getCadenaSerializada(self):
        """
        Serializa la cadena de bloques en un documento JSON
        :return conjunto de bloques en formato JSON
        """
        salida = ""
        for i in range(len(self._cadena)):
            salida = salida + str(self._cadena[i]) + "\nFin Bloque " + str(i) + str("
        return salida

    def bloqueGenesis(self,trs):
        """
        Crea el primer bloque en la cadena. Es el bloque seminal.
        """
        self.nuevoBloque(2, Bloque.generarBloqueHash(0), trs)

    def nuevoBloque(self, minado, hashBloque,trans):
        """
        Construye e inserta el siguiente bloque de la cadena
        :param minado Método de minado del bloque
        :param hashBloque Hash generado para el bloque
        :return Bloque nuevo bloque generado
        """
        bloque = Bloque(
            indice = len(self._cadena),
            minado = minado,
            hashBloque = hashBloque,
            transacciones=copy.copy(trans)
        )
```

```
        self._cadena.append(bloque)
        return bloque


    def obtenerUltimoBloque(self):
        """
        Devuelve una referencia al último nodo de la blockchain
        @return último nodo
        """
        return self._cadena[-1]

    @staticmethod
    def bloqueEsValido(nuevoBloque, ultimoBloque):
        """
        Verifica bajo cuatro reglas básicas la validez de un Bloque
        Regla 1: Secuencia ordenada de bloques por valor del índice
        Regla 2: Secuencia ordenada de bloques por fecha de creación
        Regla 3: Hash diferente entre el último bloque de la cadena y el nuevo bloque
        Regla 4: Bloque correctamente minado
        @return True si el bloque es válido, False en otro caso
        """
        if ultimoBloque._indice + 1 != nuevoBloque._indice:
            return False
        elif nuevoBloque._marcaTiempo <= ultimoBloque._marcaTiempo:
            return False
        elif ultimoBloque.gethashBloque != nuevoBloque.gethashBloque:
            return False
        elif not CadenaBloques.esPrimo(nuevoBloque.minado + ultimoBloque.minado):
            return False
        return True



    def validarBlockchain(self):
        """
        Verifica la integridad de la cadena de bloques.
        para este procedimiento se valida que el índice del siguiente bloque sea
        menor en una unidad del bloque anterior y que el hash de cada bloque sea
        correcto según la política de minado establecida, números primos en este caso
        @return True si la cadena esta bien formada, False en otro caso
        """
        bloqueAnterior = self._cadena[0]
        for i in range (1,len(self._cadena)):
            bloqueActual = self._cadena[i]
            if (bloqueAnterior._indice + 1) - bloqueActual._indice != 0:
                print("indice malo")
                return False
            if not PoolMinado.esPrimo(bloqueAnterior._minado + bloqueActual._minado):
```

```
                    return False
                bloqueAnterior = copy.copy(bloqueActual)

            return True



    def __repr__(self):
        return json.JSONEncoder().encode(self._cadena)
```

Prueba Unitaria Clase Cadena de Bloques

```
In [11]: #!/usr/bin/env python3
         # -*- coding: utf-8 -*-
         from CadenaBloques import CadenaBloques
         from NodoMinador import NodoMinador
         from Transaccion import Transaccion
         from Bloque import Bloque
         from PoolMinado import PoolMinado
         from Usuario import Usuario
         from random import randrange
         from random import uniform
         from random import random
         import jsonpickle
         import jsonpickle.backend
         import jsonpickle.handlers
         import pprint

         """
         En esta clase se realizan pruebas de unidad
         """
         class pruebasUnidadCadena(object):
             def main():
                pool = PoolMinado()
                transacciones = [];
                blockchain = CadenaBloques()
                #Se construye una transacción aleatoria para bloque genésis
                emisor = Bloque.generarBloqueHash(random())
                receptor = Bloque.generarBloqueHash(random())
                cantidad = uniform(2.5, 1000.0)
                transacciones.append(Transaccion(emisor, receptor,cantidad));
                blockchain.bloqueGenesis(transacciones)

                print("Agregando nodos mineros para la blockchain ...");
                totalUsuarios = 25;
                totalMineros = 5;

                usuarios = {};
```

```
        for i in range(totalUsuarios):
            temp = Usuario(i);
            temp.setHashUsuario();
            usuarios[temp.gethashUsuario] = temp;

        #Se construye una conjunto aleatorio de nodos mineros
        usuariosNodo = [];
        usuariosMineros = {};
        contador = totalUsuarios//totalMineros - 1;
        aux = 0;
        for key in usuarios:
            usuariosMineros[key] = usuarios[key];
            if(aux < contador):
               aux += 1;
            else:
                usuariosNodo.append(copy.copy(usuariosMineros));
                aux = 0;
                usuariosMineros = {};

        mineros = [];
        for i in range(len(usuariosNodo)):
            address = "127.0.0." + str(1+randrange(253))
            puerto = 1025+randrange(60000)
            descripcion = "nodo minero número " + str(i)
            nodo = NodoMinador(address, puerto,descripcion,usuariosNodo[i])
            nodo.actualizarCadena(blockchain)
            mineros.append(nodo);

        #Se construye una cadena de bloques con 10 bloques
        for i in range(1,10):
               transacciones = [];
               #Se construye una conjunto aleatorio de transacciones
               for tr in range(1+randrange(3)):
                   emisor = Bloque.generarBloqueHash(random())
                   receptor = Bloque.generarBloqueHash(random())
                   cantidad = uniform(2.5, 1000.0)
                   transacciones.append(Transaccion(emisor, receptor,cantidad))

               pool.torneoPorNuevoBloque(transacciones)
               print("Iteracion " + str(i) +" La cadena es válida? " + str(blockchain.v

        pprint.pprint(jsonpickle.pickler.Pickler().flatten(blockchain))


    if __name__ == "__main__":
        main()
```

```
Agregando nodos mineros para la blockchain ...
```

```
-
Iteracion 1 La cadena es válida? True ahora tiene 1 bloques
-
Iteracion 2 La cadena es válida? True ahora tiene 1 bloques
-
Iteracion 3 La cadena es válida? True ahora tiene 1 bloques
-
Iteracion 4 La cadena es válida? True ahora tiene 1 bloques
-
Iteracion 5 La cadena es válida? True ahora tiene 1 bloques
-
Iteracion 6 La cadena es válida? True ahora tiene 1 bloques
-
Iteracion 7 La cadena es válida? True ahora tiene 1 bloques
-
Iteracion 8 La cadena es válida? True ahora tiene 1 bloques
-
Iteracion 9 La cadena es válida? True ahora tiene 1 bloques
{'_cadena': [{'_hashBloque': '5feceb66ffc86f38d952786c6d696c79c2dbc239dd4e91b46729d73a27fb57e9
              '_indice': 0,
              '_marcaTiempo': 1529247715.5939398,
              '_minado': 2,
              '_transacciones': [{'_cantidad': 905.2468928548867,
                                  '_emisor': '888f882083818ee6cc2298e1d918d041fbb4e0d51e0ff3d2
                                  '_receptor': '930cf99d263c3695e622b1173b56c487b174ce9a1f5b33
                                  'py/object': 'Transaccion.Transaccion'}],
              'py/object': 'Bloque.Bloque'}],
 'py/object': 'CadenaBloques.CadenaBloques'}
```

Clase Pool de Minado

```
In [12]: #!/usr/bin/env python3
         # -*- coding: utf-8 -*-
         """
         Define la organización para los nodos que van a minar la cadena.
         """
         from random import random
         from multiprocessing.pool import ThreadPool
         import numpy as np
         from random import randint


         class PoolMinado(object):

             def __init__(self):
                 """
                 Inicialización del pool de minería
```

```
        :param _nodos: conjunto de nodos mineros
        :param _hasRate Simula la capacidad de computo del pool de mineria
        :param _recompensaPorBloque cantidad que se recompensa a los nodos que resuel
        :param _dificultad  valor usado para simular el incremento de dificultad del
        """
        self._nodos =  [];
        self._hashRate = 0;
        self._recompensaPorBloque = 10;
        self._dificultad = 1000;

    def adicionarMinero(self, minador):
        """
        Agrega  un nodo minero al conjunto de nodos minadores de la blockchain. Si el
        el hashrate del pool ylo iguala o supera el 60% de la dificultad, esta última
        20%
        @param minador: Nuevo nodo minero
        @return True si el nodo de mineria se puede adicionar, False en otro caso
        """
        self._nodos.append(minador)
        self._hashRate += minador._hashRateNodo
        if self._dificultad * 0.6 < self._hashRate:
            self._dificultad *= 1.2;

        return True

    @staticmethod
    def pruebaDeTrabajo(indiceUltimoMinado):
        """
        Algoritmo de minado por prueba de trabajo.
        En esta versión se implementa una prueba basada en números primos, en la cual
        valor de minado del bloque actual y del último bloque de la cadena debe ser u
        @param indiceUltimoMinado, Valor generado por la minería para el último bloqu
        @return minadoNuevo Valor de minado obtenido para el nuevo bloque
        """
        minadoNuevo = indiceUltimoMinado + 1
        while not PoolMinado.esPrimo(minadoNuevo + indiceUltimoMinado):
            minadoNuevo += 1

        return minadoNuevo

    @staticmethod
    def esPrimo(numero):
        """
        Determina si un número es primo
        @param numero, Número que necesitamos determinar si es primo
        @return True si numero es primo, False en otro caso
        """
        if numero < 2:
```

```
            return False
        elif numero == 2:
            return True
        else:
            for x in range(2,numero):
                if(numero % x==0):
                    return False
        return True

    def minarBloque(self,nodoMinero, numeroNodo, output):
        """
        Algoritmo que realiza el minado de un bloque para un nodo en especifico.
        @param nodoMinero, Nodo minero que realizará la prueba de trabajo
        @param numeroNodo, identificador entero del nodo minero en el pool de minería
        @param output, Parámetro de salida donde se conserva la información de
        la prueba de trabajo ejecutada por el iésimo nodo minero.
        @return minadoNuevo Valor de minado obtenido para el nuevo bloque
        """
        trabajo = 0
        for i in range(self._dificultad - nodoMinero._hashRateNodo + randint(1,10000)):
            trabajo += 1
        ultimo = nodoMinero._cadenaBloques.obtenerUltimoBloque()
        nuevoValorMinado = PoolMinado.pruebaDeTrabajo(ultimo._minado)
        #print("[" + nodoMinero._descripcion + "] ha trabajado " + str(trabajo) + " n
        output[numeroNodo][0] = numeroNodo;
        output[numeroNodo][1] = nuevoValorMinado;
        output[numeroNodo][2] = trabajo;


    def torneoPorNuevoBloque(self,transacciones):
        """
        Realiza un torneo para la adjudicación del nuevo bloque a un nodo minero.
        El nodo ganador será el que realice mayor trabajo.
        @param transacciones, Bloque de transacciones que tendrá el nuevo bloque de l
        """
        try:
            #print("Total nodos " + str(len(self._nodos)))
            numOfThreads = int(len(self._nodos))
            pool = ThreadPool(numOfThreads)
            output = [];
            for x in range(numOfThreads):
                output.append([0]*3)

            for i in range(numOfThreads):
                pool.apply_async(self.minarBloque(self._nodos[i],i,output))
            pool.close()
            pool.join()
            utilidadObtenida = np.max(output);
```

```
                ganador = np.argmax(output)//(numOfThreads-1)+randint(0,1);
                #print("utilidad obtenida " + str(utilidadObtenida))
                self._nodos[ganador]._cantidadBloquesMinados += 1;
                self._nodos[ganador]._utilidad += utilidadObtenida;
                self.minadoBloque(self._nodos[ganador],transacciones);
                nueva = self._nodos[ganador]._cadenaBloques;
                self.sincronizarCadena(nueva);

                totalMinadores = len(self._nodos[ganador]._usuariosMineros);
                for key in self._nodos[ganador]._usuariosMineros:
                    self._nodos[ganador]._usuariosMineros[key]._saldo += utilidadObtenida/
                    #print(self._nodos[ganador]._usuariosMineros[key]._saldo);
                #print("=======================================================")

            except:
                #print ("Error: un hilo ha fallado. ",sys.exc_info())
                print("-");
    @staticmethod
    def minadoBloque(nodoMinero, transacciones):
        """
        Realiza el proceso de minado de un nuevo bloque en la cadena
        @param nodoMinero que realiza el intento de minar el bloque
        @param transacciones: Conjunto de Transacciones del nuevo bloque
        """
        ultimo = nodoMinero._cadenaBloques.obtenerUltimoBloque();
        nodoMinero._cadenaBloques.nuevoBloque(PoolMinado.pruebaDeTrabajo(ultimo._minad

    def sincronizarCadena(self,nuevaCadena):
        """
        Realiza una sincronización de la cadena actualizada a todos los mineros
        @param nuevaCadena, Cadena con el nuevo bloque
        """
        for i in range(len(self._nodos)):
            self._nodos[i].actualizarCadena(nuevaCadena);

    def __str__(self):
        """
        Genera un string con la información del pool de minería
        :return: representación en String del nodo
        """
        a = "";
        for i in range(len(self._nodos)):
            a += str(self._nodos[i]) + "\n";
        return a;
```

Prueba Unitaria Clase Pool de Minado

```
In [13]: #!/usr/bin/env python3
         # -*- coding: utf-8 -*-
```

```
from CadenaBloques import CadenaBloques
from Usuario import Usuario
from NodoMinador import NodoMinador
from Transaccion import Transaccion
from Bloque import Bloque
from PoolMinado import PoolMinado
from random import randrange
from random import uniform
from random import random
from threading import Thread

"""
En esta clase se realizan pruebas de unidad
"""
class pruebasUnidadPoolMinado(Thread):
    def main():
       pool = PoolMinado()
       transacciones = [];
       blockchain = CadenaBloques()
       #Se construye una transacción aleatoria para bloque genésis
       emisor = Bloque.generarBloqueHash(random())
       receptor = Bloque.generarBloqueHash(random())
       cantidad = uniform(2.5, 1000.0)
       transacciones.append(Transaccion(emisor, receptor,cantidad));
       blockchain.bloqueGenesis(transacciones)
       print("La cadena tiene un " + str(len(blockchain._cadena)) + " Bloque, el genés
       print("Agregando nodos mineros para la blockchain ...");

       maxUsuarios = 25;
       maxMineros = 5;
       maxTransaccionesPorBloque = 10;
       totalBloquesCaena = 100;

       usuarios = {};
       for i in range(maxUsuarios):
           temp = Usuario(i);
           temp.setHashUsuario();
           usuarios[temp.gethashUsuario] = temp;

       #Se construye una conjunto aleatorio de nodos mineros
       usuariosNodo = [];
       usuariosMineros = {};
       contador = maxUsuarios//maxMineros - 1;
       aux = 0;
       for key in usuarios:
           usuariosMineros[key] = usuarios[key];
           if(aux < contador):
              aux += 1;
```

```
                    else:
                        usuariosNodo.append(usuariosMineros);
                        aux = 0;
                        usuariosMineros = {};

                for i in range(len(usuariosNodo)):
                    address = "127.0.0." + str(1+randrange(253))
                    puerto = 1025+randrange(60000)
                    descripcion = "nodo minero " + str(i)
                    minerito = NodoMinador(address, puerto,descripcion,usuariosNodo[i]);
                    minerito.actualizarCadena(blockchain);
                    pool.adicionarMinero(minerito);


                print("agregados " + str(len(usuariosNodo)) + " mineros ...");
                print("Iniciando la simulación de crecimiento de la blockchain");
                #Se construye una cadena de bloques con 10 bloques
                for i in range(1,totalBloquesCaena):
                       transacciones = [];
                       #Se construye una conjunto aleatorio de transacciones
                       for tr in range(1+randrange(maxTransaccionesPorBloque)):
                           emisor = Bloque.generarBloqueHash(random())
                           receptor = Bloque.generarBloqueHash(random())
                           cantidad = uniform(2.5, 1000.0)
                           transacciones.append(Transaccion(emisor, receptor,cantidad))
                       #print("Bloque de transacciones: " + str(i) + " , Agregadas " + str(tr+
                       pool.torneoPorNuevoBloque(transacciones)
                       #print("La cadena tiene ahora " + str(len(blockchain._cadena)) + " Bloq
                print("Fin de la simulación");
                print("{Descripcion} , {Dirección} , {Puerto} , {Utilidad} , {#Bloques}, {#Mina
                print(pool);
                print("id\t hash \t saldo");
                for key in usuarios:
                    print(str(usuarios[key]));
            if __name__ == "__main__":
                main()
```

```
La cadena tiene un 1 Bloque, el genésis
Agregando nodos mineros para la blockchain ...
agregados 5 mineros ...
Iniciando la simulación de crecimiento de la blockchain
Fin de la simulación
{Descripcion} , {Dirección} , {Puerto} , {Utilidad} , {#Bloques}, {#Minadores}
nodo minero 0 , 127.0.0.152:60109 , 74630 , 8 , 5
nodo minero 1 , 127.0.0.123:51402 , 193207 , 21 , 5
nodo minero 2 , 127.0.0.112:18794 , 277198 , 30 , 5
nodo minero 3 , 127.0.0.137:48074 , 271195 , 30 , 5
nodo minero 4 , 127.0.0.27:32727 , 90562 , 10 , 5
```

```
id         hash          saldo
0, ae1bc1b04561fac552570296dfe154094dc4df4b6d4970ca81ec9507046375b2 , 114.926
1, 9d5eab463378fe259f33d07590aa6a913a1d099d5a276e353a955cb2da4311f2 , 14.926
2, b4aec9b924aa7f7d83851c90a7f5142be0cec41c7c46cff2fb89bf0bbf055f67 , 14.926
3, 0890bee71d46a465a64625124eee9a424a66908e11ce283ed4afaf5a9551cef0 , 14.926
4, 1206cfe53ef4288f00f4c53f5377601c07bdc95f6a5249b95183cc5db54ad93a , 14.926
5, 5d4a62b58203901f4e7502ccbdd5a2e52a234bb44545549fd5c87edef9202363 , 38.641400000000004
6, 11446965a28709ff7577c812099db7093ae7a31f8aa8ebe2a81d15ee876b0f16 , 38.641400000000004
7, 44b70f5a64febf0f1c47390efaa1d824a5503e3831d37b28b42c63f25813cf17 , 38.641400000000004
8, 06aeb9cf521c8e7114647117303ebf837f50bd39b38104ce5e3371bf54d1db61 , 38.641400000000004
9, 371eb3f89a125644770e3395faa25a50d0f7bf6f9fb15b271649b53640ed1a3b , 38.641400000000004
10, d3e0b66ec1e56b92a5c0b1bd7c57540697da963987253413c40b77636707f13c , 55.439599999999984
11, b29492fe4bf566a54a5ac57bfb49cd6441b5cc3dbb19b36804da13b6a7495032 , 55.439599999999984
12, e8a26469e9a9879b4ef23fb12236e77a66b7e1a55b060694537c39fdca4fec8c , 55.439599999999984
13, 526042ccba454c2762af9fefd0d418254372e3d3c9d8605dfa6217d1c5843a61 , 55.439599999999984
14, 7f74614d41796a901e5dd4fc962da90a80b2e0a7a9f3180d968697845d882e55 , 55.439599999999984
15, c1f330b2fa35be43c2d2e695cc0d73ba2c074247ef4e10a82e4a4eca63307c80 , 54.239000000000004
16, da6a1600010abc6c5a6c9aea97886cf0559fe1ab067110a0dabb3e6c449c3835 , 54.239000000000004
17, 33d772cef882b98fe5495ddc7b1e7b369fb6301372761b36542aa7f2683a0788 , 54.239000000000004
18, 5fafa6531ad8489b778a631289e5d41cc2a6dee01b2d17dab578b9ecc47abdac , 54.239000000000004
19, ce12b6eece7dc590ac7af2b77258ba8c262d5d0b1f89b8aff89914e6eb3a86b9 , 54.239000000000004
20, b252c76b28f15c2fce8bfb7b0153ce87336a0789e16c40fed7faa04df2fc864f , 18.112400000000004
21, 839604719f7a03403bda1bea57e369f0ee226a4aa8a092aee337252b330e7ab8 , 18.112400000000004
22, 18fa0626ea0690666f53c910244389a32c4fa7a13453987b094ddd813ec1904d , 18.112400000000004
23, 192990181fd61e86ffa1e28ec8741a76c1640e816bd75b00e8e7e4499b12c1af , 18.112400000000004
24, 1d918ed33bf67966f2ab8ea55bb3707f2ead718e01b6881c5ec03882587760e1 , 18.112400000000004
```

Clase Simulador

```
In [14]: #!/usr/bin/env python3
         # -*- coding: utf-8 -*-

         from random import randint
         from Usuario import Usuario
         from CadenaBloques import CadenaBloques
         from NodoMinador import NodoMinador
         from PoolMinado import PoolMinado
         from random import randrange
         from random import random
         from random import seed
         from datetime import datetime

         """
         Define la estructura del bloque.
         """
```

```
class Simulador(object):

    def __init__(self, nombre):
        """
        Creación del escenario final de simulación
        :param _nombre Nombre Asignado al escenario
        :param _usuarios Conjunto de usuarios en la simulación
        :param _blockchain Cadena de Bloques
        :param _pool Conjunto de nodos mineros de la cadena
        """
        self._nombre = nombre;
        self._usuarios = {};
        self._blockchain = CadenaBloques();
        self._pool = PoolMinado()

    def crearUsuarios(self, cantidadUsuarios):
        """
        Inicializa el conjunto de usuarios de la cadena de bloques
        :param cantidadUsuarios: Conjunto inicial de Usuarios en la Simulación
        """
        for i in range(cantidadUsuarios):
           temp = Usuario(i);
           temp.setHashUsuario();
           self._usuarios[temp.gethashUsuario()] = temp;

    def distribuirUsuariosMineros(self, maxUsuarios, maxMineros):
        """
        Permite distribuir de manera uniforme los usuarios a los nodos mineros
        :param maxMineros: Conjunto de Nodos mineros de la blockchain
        :param maxUsuarios: Total de usuarios en el sistema
        :return arreglo con los usuarios distribuidos para cada nodo minero
        """
        #Se construye una conjunto aleatorio de nodos mineros
        usuariosNodo = [];
        usuariosMineros = {};
        contador = maxUsuarios//maxMineros - 1;
        aux = 0;
        for key in self._usuarios:
            usuariosMineros[key] = self._usuarios[key];
            if(aux < contador):
                aux += 1;
            else:
                usuariosNodo.append(usuariosMineros);
                aux = 0;
                usuariosMineros = {};
        return usuariosNodo;

    def verUsuarios(self):
```

```
        """
        Consulta el total de usuarios registrados en un momento dado en la cadena de
        :return  Representación en string del conjunto de Usuarios en la Simulación
        """
        a = "";
        for key in self._usuarios:
            a += str(self._usuarios[key])+"\n";
        return a;

    def consultarHashUsuario(self, idUsuario):
        """
        Retorna el Hash de un usuario dado su número de identificación
        :param idUsuario identificador entero de cada usuario en el sistema
        :return Hash de usuario que corresponde al identificador dado
        """
        for key in self._usuarios:
            if(self._usuarios[key]._idUsuario == idUsuario ):
                return self._usuarios[key].gethashUsuario();

    def consultarUsuario(self, hashUsuario):
        """
        Consulta la información de un usuario dado su hash
        :param hashUsuario LLave del diccionario
        :return usuario que corresponde con la llave dada
        """
        return self._usuarios[hashUsuario];

    def inicializarBlockchain(self, maxUsuarios, cantidadMineros):
        """
        Creación del bloque génesis de la cadena y asignación de nodos mineros
        :param cantidadMineros, Total de nodos Mineros para la cadena
        :param maxUsuarios: Total de usuarios en el sistema
        """
        emisor = self.consultarUsuario(self.consultarHashUsuario(0));
        receptor = self.consultarUsuario(self.consultarHashUsuario(0));
        transacciones = [];
        transacciones.append(emisor.enviar(100,receptor.gethashUsuario()));
        self._blockchain.bloqueGenesis(transacciones);
        usuariosNodo = self.distribuirUsuariosMineros(maxUsuarios,cantidadMineros);
        self.creacionMineros(cantidadMineros,usuariosNodo);

    def creacionMineros(self, cantidadMineros, usuariosNodo):
        """
        Adiciona los nodos mineros a la blockchain. Cada nodo minero tendrá asociado
        un grupo de usuarios, quienés recibirán recompensa por su esfuerzo de minado.
        :param cantidadMineros, Total de nodos Mineros para la cadena
        :param usuariosNodo, Distribución de usuarios mineros para cada nodo Minero
        """
```

```
        for m in range(cantidadMineros):
            address = "127.0.0." + str(1+randint(1,250));
            puerto = 1025+randint(1000,10000);
            descripcion = "Minero " + str(m);
            minerito = NodoMinador(address, puerto,descripcion,usuariosNodo[m]);
            minerito.actualizarCadena(self._blockchain);
            self._pool.adicionarMinero(minerito);

    def simular(self, maxTransaccionesPorBloque,totalBloquesCaena, cantidadUsuarios):
        """
        Realiza una simulación de evolución de la blockchain con los parámetros dados
        :param cantidadUsuarios, Total de usuarios activos en la cadena
        :param cantidadMineros, Total de nodos Mineros para la cadena
        :param usuariosNodo, Distribución de usuarios mineros para cada nodo Minero
        """
        transacciones = [];
        emisor = self.consultarHashUsuario(0);
        receptor = self.consultarHashUsuario(1);
        cantidad =  self.consultarUsuario(emisor)._saldo * random();
        transacciones.append(self.consultarUsuario(emisor).enviar(cantidad, self.consu
        self._pool.torneoPorNuevoBloque(transacciones);
        cantidadPosibleReceptores = 1;
        for i in range(1,totalBloquesCaena):
            transacciones = [];
            if cantidadPosibleReceptores == cantidadUsuarios-1:
                cantidadPosibleReceptores = 1;
            #Se construye una conjunto aleatorio de transacciones
            for tr in range(1+randrange(maxTransaccionesPorBloque)):
                if cantidadPosibleReceptores == cantidadUsuarios-1:
                    cantidadPosibleReceptores = 1;
                intemisor = self.seleccionarUsuario(cantidadPosibleReceptores,-1);
                intreceptor = self.seleccionarUsuario(cantidadPosibleReceptores,intemis
                emisor = self.consultarHashUsuario(intemisor);
                receptor = self.consultarHashUsuario(intreceptor);
                cantidad =  self.consultarUsuario(emisor)._saldo * random();
                #print("Cantidad receptores " + str(cantidadPosibleReceptores)+ " rece
                print(".", end="");
                transacciones.append(self.consultarUsuario(emisor).enviar(cantidad, se
                cantidadPosibleReceptores += 1;
            instanteInicial = datetime.now()
            self._pool.torneoPorNuevoBloque(transacciones);
            instanteFinal = datetime.now()
            tiempo = instanteFinal - instanteInicial # Devuelve un objeto timedelta
            segundos = tiempo.microseconds
            print("Torneo " + str(i) + " " + str(segundos) + " microseconds");

    def seleccionarUsuario(self,cantidadPosibleReceptores, diferentea):
        seed();
```

```
            x = randint(0,cantidadPosibleReceptores+1);
            while x == diferentea:
                x = randint(0,cantidadPosibleReceptores+1);
            return x;

        def verNodosMineros(self):
            """
            Consulta el total de nodos mineros registrados en la cadena de bloques
            :return pool de mineros de la blockchain
            """
            return self._pool;
```

Prueba Unitaria Clase Simulador

```
In [15]: #!/usr/bin/env python3
         # -*- coding: utf-8 -*-
         from Simulador import Simulador
         from datetime import datetime

         """
         En esta clase se realizan pruebas de unidad
         """
         class pruebasUnidadSimulador(object):

             def main():
                sm = Simulador("Simulación del comportamiento de una cadena de bloques");
                #cantidadUsuarios = int(input("Cantidad de usuarios en la simulación: "));
                cantidadUsuarios = 20;
                sm.crearUsuarios(cantidadUsuarios);
                #cantidadMineros = int(input("Cantidad de Mineros de la blockchain: "));
                cantidadMineros = 5;
                #maxTransaccionesPorBloque = int(input("Maxima cantidad de transacciones por B
                maxTransaccionesPorBloque = 5;
                #totalBloquesCaena = int(input("Cantidad de Bloques en la Cadena: "));
                totalBloquesCaena = 1000;
                print("Creando la cadena de bloques...Adicionando Bloque Génesis...");
                print("Hash usuario génesis: " + str(sm.consultarHashUsuario(0)));
                sm.inicializarBlockchain(cantidadUsuarios,cantidadMineros);
                print("Imprimiendo Bloque Génesis...")
                print(sm._blockchain.getCadenaSerializada());

                instanteInicial = datetime.now()
                sm.simular(maxTransaccionesPorBloque,totalBloquesCaena,cantidadUsuarios);
                instanteFinal = datetime.now()
                tiempo = instanteFinal - instanteInicial # Devuelve un objeto timedelta
                segundos = tiempo.seconds;
                print("\nFin de la simulación: demoro " + str(segundos) + " segundos");
                print("Estado de los Nodos Mineros...");
```

```
            print("{} , {} , {} , {}, {}".format(str("Descripción"),str("Direccion"), str("
            print(sm.verNodosMineros());
            print("Estado de los Usuarios")
            print("{}, {} , {} ".format(str("#id"),str("Direccion Hash del Usario"), str("S
            print(sm.verUsuarios());
        if __name__ == "__main__":
            main()
```

```
Creando la cadena de bloques...Adicionando Bloque Génesis...
Hash usuario génesis: ae1bc1b04561fac552570296dfe154094dc4df4b6d4970ca81ec9507046375b2
Imprimiendo Bloque Génesis...
0 , 2 , 5feceb66ffc86f38d952786c6d696c79c2dbc239dd4e91b46729d73a27fb57e9  , 1529248135.5695615
Fin Bloque 0.

...
Fin de la simulación: demoro 107 segundos
Estado de los Nodos Mineros...
Descripción , Direccion , Utilidad , Bloques, Usuarios
Minero 0 , 127.0.0.20:2999 , 911962 , 100 , 4
Minero 1 , 127.0.0.132:9888 , 1955814 , 211 , 4
Minero 2 , 127.0.0.132:3869 , 2640455 , 282 , 4
Minero 3 , 127.0.0.48:7341 , 2844724 , 305 , 4
Minero 4 , 127.0.0.185:6264 , 958680 , 102 , 4

Estado de los Usuarios
#id, Direccion Hash del Usario , Saldo
0, ae1bc1b04561fac552570296dfe154094dc4df4b6d4970ca81ec9507046375b2 , 43.90571953228023
1, 9d5eab463378fe259f33d07590aa6a913a1d099d5a276e353a955cb2da4311f2 , 170.85615671626476
2, b4aec9b924aa7f7d83851c90a7f5142be0cec41c7c46cff2fb89bf0bbf055f67 , 83.2765994522879
3, 0890bee71d46a465a64625124eee9a424a66908e11ce283ed4afaf5a9551cef0 , 1623.9545723143294
4, 1206cfe53ef4288f00f4c53f5377601c07bdc95f6a5249b95183cc5db54ad93a , 125.85689736016461
5, 5d4a62b58203901f4e7502ccbdd5a2e52a234bb44545549fd5c87edef9202363 , 368.81631273914695
6, 11446965a28709ff7577c812099db7093ae7a31f8aa8ebe2a81d15ee876b0f16 , 193.74265994235944
7, 44b70f5a64febf0f1c47390efaa1d824a5503e3831d37b28b42c63f25813cf17 , 745.5559940692958
8, 06aeb9cf521c8e7114647117303ebf837f50bd39b38104ce5e3371bf54d1db61 , 290.65435766067515
9, 371eb3f89a125644770e3395faa25a50d0f7bf6f9fb15b271649b53640ed1a3b , 839.5940516116573
10, d3e0b66ec1e56b92a5c0b1bd7c57540697da963987253413c40b77636707f13c , 40.95932113167698
11, b29492fe4bf566a54a5ac57bfb49cd6441b5cc3dbb19b36804da13b6a7495032 , 3083.063430232901
12, e8a26469e9a9879b4ef23fb12236e77a66b7e1a55b060694537c39fdca4fec8c , 73.9606016876291
13, 526042ccba454c2762af9fefd0d418254372e3d3c9d8605dfa6217d1c5843a61 , 606.9877366153947
14, 7f74614d41796a901e5dd4fc962da90a80b2e0a7a9f3180d968697845d882e55 , 313.7516847236654
15, c1f330b2fa35be43c2d2e695cc0d73ba2c074247ef4e10a82e4a4eca63307c80 , 21.880188867958633
16, da6a1600010abc6c5a6c9aea97886cf0559fe1ab067110a0dabb3e6c449c3835 , 603.3334429737754
17, 33d772cef882b98fe5495ddc7b1e7b369fb6301372761b36542aa7f2683a0788 , 4.072810289791246
18, 5fafa6531ad8489b778a631289e5d41cc2a6dee01b2d17dab578b9ecc47abdac , 88.3769476493061
19, ce12b6eece7dc590ac7af2b77258ba8c262d5d0b1f89b8aff89914e6eb3a86b9 , 89.03551442943997
```

# BlokchainSimulator

1.0

Generated by Doxygen 1.8.13

# Contents


**1 Namespace Index** **1**

1.1 Namespace List . . . . . 1

**2 Hierarchical Index** **3**

2.1 Class Hierarchy . . . . . 3

**3 Class Index** **5**

3.1 Class List . . . . . 5

**4 File Index** **7**

4.1 File List . . . . . 7

**5 Namespace Documentation** **9**

5.1 blockchainsimulacion Namespace Reference . . . . . 9

5.1.1 Variable Documentation . . . . . 9

5.1.1.1 __author__ . . . . . 10

5.1.1.2 __copyright__ . . . . . 10

5.1.1.3 __email__ . . . . . 10

5.1.1.4 __maintainer__ . . . . . 10

5.1.1.5 __version__ . . . . . 10

5.2 blockChainTest2019 Namespace Reference . . . . . 10

5.3 Bloque Namespace Reference . . . . . 10

5.4 CadenaBloques Namespace Reference . . . . . 11

5.5 conf Namespace Reference . . . 11
5.5.1 Variable Documentation . . . 11
5.5.1.1 author . . . 12
5.5.1.2 copyright . . . 12
5.5.1.3 epub_author . . . 12
5.5.1.4 epub_copyright . . . 12
5.5.1.5 epub_exclude_files . . . 12
5.5.1.6 epub_publisher . . . 12
5.5.1.7 epub_title . . . 13
5.5.1.8 exclude_patterns . . . 13
5.5.1.9 extensions . . . 13
5.5.1.10 html_sidebars . . . 13
5.5.1.11 html_static_path . . . 14
5.5.1.12 html_theme . . . 14
5.5.1.13 htmlhelp_basename . . . 14
5.5.1.14 intersphinx_mapping . . . 14
5.5.1.15 language . . . 14
5.5.1.16 latex_documents . . . 15
5.5.1.17 latex_elements . . . 15
5.5.1.18 man_pages . . . 15
5.5.1.19 master_doc . . . 16
5.5.1.20 project . . . 16
5.5.1.21 pygments_style . . . 16
5.5.1.22 release . . . 16
5.5.1.23 source_suffix . . . 16
5.5.1.24 templates_path . . . 16
5.5.1.25 texinfo_documents . . . 17
5.5.1.26 todo_include_todos . . . 17

5.5.1.27 version . . . 17
5.6 NodoMinador Namespace Reference . . . 17
5.7 PoolMinado Namespace Reference . . . 17
5.8 pruebasUnidad Namespace Reference . . . 18
5.8.1 Function Documentation . . . 18
5.8.1.1 main() . . . 18
5.9 pruebasUnidadBloque Namespace Reference . . . 18
5.10 pruebasUnidadCadena Namespace Reference . . . 18
5.11 pruebasUnidadNodoMineria Namespace Reference . . . 18
5.12 pruebasUnidadPoolMinado Namespace Reference . . . 18
5.13 pruebasUnidadSimulador Namespace Reference . . . 19
5.14 pruebasUnidadTransaccion Namespace Reference . . . 19
5.15 pruebasUnidadUsuario Namespace Reference . . . 19
5.16 Simulador Namespace Reference . . . 19
5.17 Transaccion Namespace Reference . . . 19
5.18 Usuario Namespace Reference . . . 19

**6 Class Documentation** **21**

6.1 Bloque.Bloque Class Reference . . . 21
6.1.1 Detailed Description . . . 21
6.1.2 Constructor & Destructor Documentation . . . 22
6.1.2.1 __init__() . . . 22
6.1.3 Member Function Documentation . . . 22
6.1.3.1 __repr__() . . . 22
6.1.3.2 __str__() . . . 22
6.1.3.3 consultarBloque() . . . 23
6.1.3.4 generarBloqueHash() . . . 23
6.1.3.5 gethashBloque() . . . 23

6.1.3.6 setHashBloque() . . . 23
6.1.4 Member Data Documentation . . . 23
6.1.4.1 _hashBloque . . . 24
6.1.4.2 _indice . . . 24
6.1.4.3 _marcaTiempo . . . 24
6.1.4.4 _minado . . . 24
6.1.4.5 _transacciones . . . 24
6.2 blockchainsimulacion.Bloque Class Reference . . . 24
6.2.1 Detailed Description . . . 25
6.2.2 Constructor & Destructor Documentation . . . 25
6.2.2.1 __init__() . . . 25
6.2.3 Member Function Documentation . . . 26
6.2.3.1 __repr__() . . . 26
6.2.3.2 __str__() . . . 26
6.2.3.3 generarBloqueHash() . . . 26
6.2.3.4 gethashBloque() . . . 26
6.2.3.5 setHashBloque() . . . 27
6.2.4 Member Data Documentation . . . 27
6.2.4.1 _hashBloque . . . 27
6.2.4.2 _indice . . . 27
6.2.4.3 _marcaTiempo . . . 27
6.2.4.4 _minado . . . 27
6.2.4.5 _transacciones . . . 28
6.3 CadenaBloques.CadenaBloques Class Reference . . . 28
6.3.1 Detailed Description . . . 28
6.3.2 Constructor & Destructor Documentation . . . 28
6.3.2.1 __init__() . . . 29
6.3.3 Member Function Documentation . . . 29

6.3.3.1 __repr__() . . . 29
6.3.3.2 __str__() . . . 29
6.3.3.3 bloqueEsValido() . . . 29
6.3.3.4 bloqueGenesis() . . . 30
6.3.3.5 getCadenaSerializada() . . . 30
6.3.3.6 nuevoBloque() . . . 30
6.3.3.7 obtenerUltimoBloque() . . . 31
6.3.3.8 validarBlockchain() . . . 31
6.3.4 Member Data Documentation . . . 31
6.3.4.1 _cadena . . . 31
6.4 blockchainsimulacion.CadenaBloques Class Reference . . . 31
6.4.1 Detailed Description . . . 32
6.4.2 Constructor & Destructor Documentation . . . 32
6.4.2.1 __init__() . . . 32
6.4.3 Member Function Documentation . . . 32
6.4.3.1 __repr__() . . . 33
6.4.3.2 bloqueEsValido() . . . 33
6.4.3.3 bloqueGenesis() . . . 33
6.4.3.4 getCadenaSerializada() . . . 33
6.4.3.5 nuevoBloque() . . . 34
6.4.3.6 obtenerUltimoBloque() . . . 34
6.4.3.7 validarBlockchain() . . . 34
6.4.4 Member Data Documentation . . . 34
6.4.4.1 _cadena . . . 35
6.5 blockchainsimulacion.NodoMinador Class Reference . . . 35
6.5.1 Detailed Description . . . 35
6.5.2 Constructor & Destructor Documentation . . . 35
6.5.2.1 __init__() . . . 36

6.5.3 Member Function Documentation . . . 36
6.5.3.1 __str__() . . . 36
6.5.3.2 actualizarCadena() . . . 36
6.5.3.3 infoCadena() . . . 37
6.5.4 Member Data Documentation . . . 37
6.5.4.1 _cadenaBloques . . . 37
6.5.4.2 _cantidadBloquesMinados . . . 37
6.5.4.3 _descripcion . . . 37
6.5.4.4 _direccion . . . 37
6.5.4.5 _fechaCreacion . . . 38
6.5.4.6 _hashRateNodo . . . 38
6.5.4.7 _puerto . . . 38
6.5.4.8 _usuariosMineros . . . 38
6.5.4.9 _utilidad . . . 38
6.6 NodoMinador.NodoMinador Class Reference . . . 38
6.6.1 Detailed Description . . . 39
6.6.2 Constructor & Destructor Documentation . . . 39
6.6.2.1 __init__() . . . 39
6.6.3 Member Function Documentation . . . 40
6.6.3.1 __str__() . . . 40
6.6.3.2 actualizarCadena() . . . 40
6.6.3.3 infoCadena() . . . 40
6.6.4 Member Data Documentation . . . 40
6.6.4.1 _cadenaBloques . . . 41
6.6.4.2 _cantidadBloquesMinados . . . 41
6.6.4.3 _descripcion . . . 41
6.6.4.4 _direccion . . . 41
6.6.4.5 _fechaCreacion . . . 41

6.6.4.6 _hashRateNodo . . . 41
6.6.4.7 _puerto . . . 42
6.6.4.8 _usuariosMineros . . . 42
6.6.4.9 _utilidad . . . 42
6.7 PoolMinado.PoolMinado Class Reference . . . 42
6.7.1 Detailed Description . . . 43
6.7.2 Constructor & Destructor Documentation . . . 43
6.7.2.1 __init__() . . . 43
6.7.3 Member Function Documentation . . . 43
6.7.3.1 __str__() . . . 43
6.7.3.2 adicionarMinero() . . . 44
6.7.3.3 esPrimo() . . . 44
6.7.3.4 minadoBloque() . . . 44
6.7.3.5 minarBloque() . . . 45
6.7.3.6 pruebaDeTrabajo() . . . 45
6.7.3.7 sincronizarCadena() . . . 45
6.7.3.8 torneoPorNuevoBloque() . . . 46
6.7.4 Member Data Documentation . . . 46
6.7.4.1 _dificultad . . . 46
6.7.4.2 _hashRate . . . 46
6.7.4.3 _nodos . . . 46
6.7.4.4 _recompensaPorBloque . . . 47
6.8 blockchainsimulacion.PoolMinado Class Reference . . . 47
6.8.1 Detailed Description . . . 47
6.8.2 Constructor & Destructor Documentation . . . 47
6.8.2.1 __init__() . . . 48
6.8.3 Member Function Documentation . . . 48
6.8.3.1 __str__() . . . 48

6.8.3.2 adicionarMinero() . . . 48
6.8.3.3 esPrimo() . . . 49
6.8.3.4 minadoBloque() . . . 49
6.8.3.5 minarBloque() . . . 49
6.8.3.6 pruebaDeTrabajo() . . . 50
6.8.3.7 sincronizarCadena() . . . 50
6.8.3.8 torneoPorNuevoBloque() . . . 50
6.8.4 Member Data Documentation . . . 50
6.8.4.1 _dificultad . . . 51
6.8.4.2 _hashRate . . . 51
6.8.4.3 _nodos . . . 51
6.8.4.4 _recompensaPorBloque . . . 51
6.9 blockchainsimulacion.pruebasUnidadBloque Class Reference . . . 51
6.9.1 Detailed Description . . . 52
6.9.2 Member Function Documentation . . . 52
6.9.2.1 main() . . . 52
6.10 pruebasUnidadBloque.pruebasUnidadBloque Class Reference . . . 52
6.10.1 Detailed Description . . . 52
6.10.2 Member Function Documentation . . . 52
6.10.2.1 main() . . . 52
6.11 blockchainsimulacion.pruebasUnidadCadena Class Reference . . . 53
6.11.1 Detailed Description . . . 53
6.11.2 Member Function Documentation . . . 53
6.11.2.1 main() . . . 53
6.12 pruebasUnidadCadena.pruebasUnidadCadena Class Reference . . . 53
6.12.1 Detailed Description . . . 53
6.12.2 Member Function Documentation . . . 54
6.12.2.1 main() . . . 54

6.13 blockchainsimulacion.pruebasUnidadNodoMineria Class Reference . . . 54
6.13.1 Detailed Description . . . 54
6.13.2 Member Function Documentation . . . 54
6.13.2.1 main() . . . 54
6.14 pruebasUnidadNodoMineria.pruebasUnidadNodoMineria Class Reference . . . 55
6.14.1 Detailed Description . . . 55
6.14.2 Member Function Documentation . . . 55
6.14.2.1 main() . . . 55
6.15 pruebasUnidadPoolMinado.pruebasUnidadPoolMinado Class Reference . . . 55
6.15.1 Detailed Description . . . 55
6.15.2 Member Function Documentation . . . 56
6.15.2.1 main() . . . 56
6.16 blockchainsimulacion.pruebasUnidadPoolMinado Class Reference . . . 56
6.16.1 Detailed Description . . . 56
6.16.2 Member Function Documentation . . . 56
6.16.2.1 main() . . . 56
6.17 pruebasUnidadSimulador.pruebasUnidadSimulador Class Reference . . . 57
6.17.1 Detailed Description . . . 57
6.17.2 Member Function Documentation . . . 57
6.17.2.1 main() . . . 57
6.18 blockchainsimulacion.pruebasUnidadSimulador Class Reference . . . 57
6.18.1 Detailed Description . . . 57
6.18.2 Member Function Documentation . . . 58
6.18.2.1 main() . . . 58
6.19 pruebasUnidadTransaccion.pruebasUnidadTransaccion Class Reference . . . 58
6.19.1 Detailed Description . . . 58
6.19.2 Member Function Documentation . . . 58
6.19.2.1 jsonDefault() . . . 58

6.19.2.2 main() . . . 59
6.20 blockchainsimulacion.pruebasUnidadTransaccion Class Reference . . . 59
6.20.1 Detailed Description . . . 59
6.20.2 Member Function Documentation . . . 59
6.20.2.1 jsonDefault() . . . 59
6.20.2.2 main() . . . 60
6.21 pruebasUnidadUsuario.pruebasUnidadUsuario Class Reference . . . 60
6.21.1 Detailed Description . . . 60
6.21.2 Member Function Documentation . . . 60
6.21.2.1 jsonDefault() . . . 60
6.21.2.2 main() . . . 61
6.22 blockchainsimulacion.pruebasUnidadUsuario Class Reference . . . 61
6.22.1 Detailed Description . . . 61
6.22.2 Member Function Documentation . . . 61
6.22.2.1 jsonDefault() . . . 61
6.22.2.2 main() . . . 62
6.23 Simulador.Simulador Class Reference . . . 62
6.23.1 Detailed Description . . . 62
6.23.2 Constructor & Destructor Documentation . . . 62
6.23.2.1 __init__() . . . 63
6.23.3 Member Function Documentation . . . 63
6.23.3.1 consultarHashUsuario() . . . 63
6.23.3.2 consultarUsuario() . . . 63
6.23.3.3 creacionMineros() . . . 64
6.23.3.4 crearUsuarios() . . . 64
6.23.3.5 distribuirUsuariosMineros() . . . 64
6.23.3.6 inicializarBlockchain() . . . 65
6.23.3.7 logEjecucion() . . . 65

6.23.3.8 seleccionarUsuario() . . . 65
6.23.3.9 simular() . . . 65
6.23.3.10 verNodosMineros() . . . 66
6.23.3.11 verUsuarios() . . . 66
6.23.4 Member Data Documentation . . . 66
6.23.4.1 _blockchain . . . 66
6.23.4.2 _nombre . . . 66
6.23.4.3 _pool . . . 67
6.23.4.4 _usuarios . . . 67
6.24 blockchainsimulacion.Simulador Class Reference . . . 67
6.24.1 Detailed Description . . . 68
6.24.2 Constructor & Destructor Documentation . . . 68
6.24.2.1 __init__() . . . 68
6.24.3 Member Function Documentation . . . 68
6.24.3.1 consultarHashUsuario() . . . 68
6.24.3.2 consultarUsuario() . . . 69
6.24.3.3 creacionMineros() . . . 69
6.24.3.4 crearUsuarios() . . . 69
6.24.3.5 distribuirUsuariosMineros() . . . 70
6.24.3.6 inicializarBlockchain() . . . 70
6.24.3.7 seleccionarUsuario() . . . 70
6.24.3.8 simular() . . . 71
6.24.3.9 verNodosMineros() . . . 71
6.24.3.10 verUsuarios() . . . 71
6.24.4 Member Data Documentation . . . 71
6.24.4.1 _blockchain . . . 72
6.24.4.2 _nombre . . . 72
6.24.4.3 _pool . . . 72

6.24.4.4 _usuarios . . . 72
6.25 blockchainsimulacion.Transaccion Class Reference . . . 72
6.25.1 Detailed Description . . . 73
6.25.2 Constructor & Destructor Documentation . . . 73
6.25.2.1 __init__() . . . 73
6.25.3 Member Function Documentation . . . 73
6.25.3.1 __str__() . . . 73
6.25.3.2 getCantidad() . . . 74
6.25.3.3 getEmisor() . . . 74
6.25.3.4 getReceptor() . . . 74
6.25.4 Member Data Documentation . . . 74
6.25.4.1 _cantidad . . . 74
6.25.4.2 _emisor . . . 75
6.25.4.3 _receptor . . . 75
6.26 Transaccion.Transaccion Class Reference . . . 75
6.26.1 Detailed Description . . . 75
6.26.2 Constructor & Destructor Documentation . . . 75
6.26.2.1 __init__() . . . 76
6.26.3 Member Function Documentation . . . 76
6.26.3.1 __str__() . . . 76
6.26.3.2 getCantidad() . . . 76
6.26.3.3 getEmisor() . . . 77
6.26.3.4 getReceptor() . . . 77
6.26.4 Member Data Documentation . . . 77
6.26.4.1 _cantidad . . . 77
6.26.4.2 _emisor . . . 77
6.26.4.3 _receptor . . . 78
6.27 Usuario.Usuario Class Reference . . . 78

6.27.1 Detailed Description . . . 78
6.27.2 Constructor & Destructor Documentation . . . 78
6.27.2.1 __init__() . . . 79
6.27.3 Member Function Documentation . . . 79
6.27.3.1 __str__() . . . 79
6.27.3.2 enviar() . . . 79
6.27.3.3 generarUsuarioHash() . . . 80
6.27.3.4 gethashUsuario() . . . 80
6.27.3.5 recibir() . . . 80
6.27.3.6 setHashUsuario() . . . 80
6.27.4 Member Data Documentation . . . 81
6.27.4.1 _hashUsuario . . . 81
6.27.4.2 _idUsuario . . . 81
6.27.4.3 _marcaTiempo . . . 81
6.27.4.4 _saldo . . . 81
6.28 blockchainsimulacion.Usuario Class Reference . . . 81
6.28.1 Detailed Description . . . 82
6.28.2 Constructor & Destructor Documentation . . . 82
6.28.2.1 __init__() . . . 82
6.28.3 Member Function Documentation . . . 82
6.28.3.1 __str__() . . . 83
6.28.3.2 enviar() . . . 83
6.28.3.3 generarUsuarioHash() . . . 83
6.28.3.4 gethashUsuario() . . . 83
6.28.3.5 recibir() . . . 84
6.28.3.6 setHashUsuario() . . . 84
6.28.4 Member Data Documentation . . . 84
6.28.4.1 _hashUsuario . . . 84
6.28.4.2 _idUsuario . . . 84
6.28.4.3 _marcaTiempo . . . 84
6.28.4.4 _saldo . . . 84

**7 File Documentation** **85**

7.1 __init__.py File Reference . . . . . 85

7.2 cadena/Bloque.py File Reference . . . . . 85

7.3 cadena/CadenaBloques.py File Reference . . . . . 85

7.4 cadena/NodoMinador.py File Reference . . . . . 86

7.5 cadena/PoolMinado.py File Reference . . . . . 86

7.6 cadena/pruebasUnidad.py File Reference . . . . . 86

7.7 cadena/pruebasUnidadBloque.py File Reference . . . . . 86

7.8 cadena/pruebasUnidadCadena.py File Reference . . . . . 87

7.9 cadena/pruebasUnidadNodoMineria.py File Reference . . . . . 87

7.10 cadena/pruebasUnidadPoolMinado.py File Reference . . . . . 87

7.11 cadena/pruebasUnidadTransaccion.py File Reference . . . . . 87

7.12 cadena/Transaccion.py File Reference . . . . . 88

7.13 cliente/pruebasUnidadUsuario.py File Reference . . . . . 88

7.14 cliente/Usuario.py File Reference . . . . . 88

7.15 docs/blockchainsimulacion.py File Reference . . . . . 88

7.16 docs/conf.py File Reference . . . . . 89

7.17 docs/source/conf.py File Reference . . . . . 90

7.18 simulador/pruebasUnidadSimulador.py File Reference . . . . . 90

7.19 simulador/Simulador.py File Reference . . . . . 90

**Index** **91**

# Chapter 1

# Namespace Index

## 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:


blockchainsimulacion . . . . . 9
blockChainTest2019 . . . . . 10
Bloque . . . . . 10
CadenaBloques . . . . . 11
conf . . . . . 11
NodoMinador . . . . . 17
PoolMinado . . . . . 17
pruebasUnidad . . . . . 18
pruebasUnidadBloque . . . . . 18
pruebasUnidadCadena . . . . . 18
pruebasUnidadNodoMineria . . . . . 18
pruebasUnidadPoolMinado . . . . . 18
pruebasUnidadSimulador . . . . . 19
pruebasUnidadTransaccion . . . . . 19
pruebasUnidadUsuario . . . . . 19
Simulador . . . . . 19
Transaccion . . . . . 19
Usuario . . . . . 19

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:


object
- blockchainsimulacion.Bloque . . . 24
- blockchainsimulacion.CadenaBloques . . . 31
- blockchainsimulacion.NodoMinador . . . 35
- blockchainsimulacion.PoolMinado . . . 47
- blockchainsimulacion.pruebasUnidadBloque . . . 51
- blockchainsimulacion.pruebasUnidadCadena . . . 53
- blockchainsimulacion.pruebasUnidadNodoMineria . . . 54
- blockchainsimulacion.pruebasUnidadSimulador . . . 57
- blockchainsimulacion.pruebasUnidadTransaccion . . . 59
- blockchainsimulacion.pruebasUnidadUsuario . . . 61
- blockchainsimulacion.Simulador . . . 67
- blockchainsimulacion.Transaccion . . . 72
- blockchainsimulacion.Usuario . . . 81
- Bloque.Bloque . . . 21
- CadenaBloques.CadenaBloques . . . 28
- NodoMinador.NodoMinador . . . 38
- PoolMinado.PoolMinado . . . 42
- pruebasUnidadBloque.pruebasUnidadBloque . . . 52
- pruebasUnidadCadena.pruebasUnidadCadena . . . 53
- pruebasUnidadNodoMineria.pruebasUnidadNodoMineria . . . 55
- pruebasUnidadSimulador.pruebasUnidadSimulador . . . 57
- pruebasUnidadTransaccion.pruebasUnidadTransaccion . . . 58
- pruebasUnidadUsuario.pruebasUnidadUsuario . . . 60
- Simulador.Simulador . . . 62
- Transaccion.Transaccion . . . 75
- Usuario.Usuario . . . 78

Thread
- blockchainsimulacion.pruebasUnidadPoolMinado . . . 56
- pruebasUnidadPoolMinado.pruebasUnidadPoolMinado . . . 55

# Chapter 3

# Class Index

## 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:


Bloque.Bloque . . . 21
blockchainsimulacion.Bloque . . . 24
CadenaBloques.CadenaBloques . . . 28
blockchainsimulacion.CadenaBloques . . . 31
blockchainsimulacion.NodoMinador . . . 35
NodoMinador.NodoMinador . . . 38
PoolMinado.PoolMinado . . . 42
blockchainsimulacion.PoolMinado . . . 47
blockchainsimulacion.pruebasUnidadBloque . . . 51
pruebasUnidadBloque.pruebasUnidadBloque . . . 52
blockchainsimulacion.pruebasUnidadCadena . . . 53
pruebasUnidadCadena.pruebasUnidadCadena . . . 53
blockchainsimulacion.pruebasUnidadNodoMineria . . . 54
pruebasUnidadNodoMineria.pruebasUnidadNodoMineria . . . 55
pruebasUnidadPoolMinado.pruebasUnidadPoolMinado . . . 55
blockchainsimulacion.pruebasUnidadPoolMinado . . . 56
pruebasUnidadSimulador.pruebasUnidadSimulador . . . 57
blockchainsimulacion.pruebasUnidadSimulador . . . 57
pruebasUnidadTransaccion.pruebasUnidadTransaccion . . . 58
blockchainsimulacion.pruebasUnidadTransaccion . . . 59
pruebasUnidadUsuario.pruebasUnidadUsuario . . . 60
blockchainsimulacion.pruebasUnidadUsuario . . . 61
Simulador.Simulador . . . 62
blockchainsimulacion.Simulador . . . 67
blockchainsimulacion.Transaccion . . . 72
Transaccion.Transaccion . . . 75
Usuario.Usuario . . . 78
blockchainsimulacion.Usuario . . . 81

# Chapter 4

# File Index

## 4.1 File List

Here is a list of all files with brief descriptions:


__init__.py . . . 85
cadena/Bloque.py . . . 85
cadena/CadenaBloques.py . . . 85
cadena/NodoMinador.py . . . 86
cadena/PoolMinado.py . . . 86
cadena/pruebasUnidad.py . . . 86
cadena/pruebasUnidadBloque.py . . . 86
cadena/pruebasUnidadCadena.py . . . 87
cadena/pruebasUnidadNodoMineria.py . . . 87
cadena/pruebasUnidadPoolMinado.py . . . 87
cadena/pruebasUnidadTransaccion.py . . . 87
cadena/Transaccion.py . . . 88
cliente/pruebasUnidadUsuario.py . . . 88
cliente/Usuario.py . . . 88
docs/blockchainsimulacion.py . . . 88
docs/conf.py . . . 89
docs/source/conf.py . . . 90
simulador/pruebasUnidadSimulador.py . . . 90
simulador/Simulador.py . . . 90

# Chapter 5

# Namespace Documentation

## 5.1 blockchainsimulacion Namespace Reference

### Classes

- class Bloque
- class CadenaBloques
- class NodoMinador
- class PoolMinado
- class pruebasUnidadBloque
- class pruebasUnidadCadena
- class pruebasUnidadNodoMineria
- class pruebasUnidadPoolMinado
- class pruebasUnidadSimulador
- class pruebasUnidadTransaccion
- class pruebasUnidadUsuario
- class Simulador
- class Transaccion
- class Usuario

### Variables

- string __author__ = "Roberto A. Pava"
- string __maintainer__ = "Roberto A. Pava"
- string __email__ = "rapavad@correo.udistrital.edu.co"
- string __copyright__ = "Copyright 2018"
- string __version__ = "1.0"

#### 5.1.1 Variable Documentation

#### 5.1.1.1 __author__

```
string blockchainsimulacion.__author__ = "Roberto A. Pava"  [private]
```

Definition at line 115 of file blockchainsimulacion.py.

#### 5.1.1.2 __copyright__

```
string blockchainsimulacion.__copyright__ = "Copyright 2018"  [private]
```

Definition at line 118 of file blockchainsimulacion.py.

#### 5.1.1.3 __email__

```
string blockchainsimulacion.__email__ = "rapavad@correo.udistrital.edu.co"  [private]
```

Definition at line 117 of file blockchainsimulacion.py.

#### 5.1.1.4 __maintainer__

```
string blockchainsimulacion.__maintainer__ = "Roberto A. Pava"  [private]
```

Definition at line 116 of file blockchainsimulacion.py.

#### 5.1.1.5 __version__

```
string blockchainsimulacion.__version__ = "1.0"  [private]
```

Definition at line 119 of file blockchainsimulacion.py.

## 5.2 blockChainTest2019 Namespace Reference

## 5.3 Bloque Namespace Reference

### Classes

- class Bloque

## 5.4 CadenaBloques Namespace Reference

### Classes

- class CadenaBloques

## 5.5 conf Namespace Reference

### Variables

- list extensions
- list templates_path = ['_templates']
- string source_suffix = '.rst'
- string master_doc = 'index'
- string project = 'MyProject'
- string copyright = '2019, My Name'
- string author = 'My Name'
- string version = '0.0.1'
- string release = '0.0.1'
- string language = 'en'
- list exclude_patterns = ['_build', 'Thumbs.db', '.DS_Store']
- string pygments_style = 'sphinx'
- bool todo_include_todos = True
- string html_theme = 'alabaster'
- list html_static_path = ['_static']
- dictionary html_sidebars
- string htmlhelp_basename = 'MyProjectdoc'
- dictionary latex_elements
- list latex_documents
- list man_pages
- list texinfo_documents
- string epub_title = project
- string epub_author = author
- string epub_publisher = author
- string epub_copyright = copyright
- list epub_exclude_files = ['search.html']
- dictionary intersphinx_mapping = {'https://docs.python.org/': None}

### 5.5.1 Variable Documentation

#### 5.5.1.1 author

```
string conf.author = 'My Name'
```

Definition at line 53 of file conf.py.

#### 5.5.1.2 copyright

```
string conf.copyright = '2019, My Name'
```

Definition at line 52 of file conf.py.

#### 5.5.1.3 epub_author

```
string conf.epub_author = author
```

Definition at line 176 of file conf.py.

#### 5.5.1.4 epub_copyright

```
string conf.epub_copyright = copyright
```

Definition at line 178 of file conf.py.

#### 5.5.1.5 epub_exclude_files

```
list conf.epub_exclude_files = ['search.html']
```

Definition at line 190 of file conf.py.

#### 5.5.1.6 epub_publisher

```
string conf.epub_publisher = author
```

Definition at line 177 of file conf.py.

#### 5.5.1.7 epub_title

```
string conf.epub_title = project
```

Definition at line 175 of file conf.py.

#### 5.5.1.8 exclude_patterns

```
list conf.exclude_patterns = ['_build', 'Thumbs.db', '.DS_Store']
```

Definition at line 74 of file conf.py.

#### 5.5.1.9 extensions

```
list conf.extensions
```

**Initial value:**

```
1 =  ['sphinx.ext.autodoc',
2     'sphinx.ext.todo',
3     'sphinx.ext.viewcode']
```

Definition at line 34 of file conf.py.

#### 5.5.1.10 html_sidebars

```
dictionary conf.html_sidebars
```

**Initial value:**

```
1 =  {
2     '**': [
3         'relations.html',  # needs 'show_related': True theme option to display
4         'searchbox.html',
5     ]
6 }
```

Definition at line 106 of file conf.py.

#### 5.5.1.11 html_static_path

```
list conf.html_static_path = ['_static']
```

Definition at line 99 of file conf.py.

#### 5.5.1.12 html_theme

```
string conf.html_theme = 'alabaster'
```

Definition at line 88 of file conf.py.

#### 5.5.1.13 htmlhelp_basename

```
string conf.htmlhelp_basename = 'MyProjectdoc'
```

Definition at line 117 of file conf.py.

#### 5.5.1.14 intersphinx_mapping

```
dictionary conf.intersphinx_mapping = {'https://docs.python.org/':  None}
```

Definition at line 200 of file conf.py.

#### 5.5.1.15 language

```
conf.language = 'en'
```

Definition at line 69 of file conf.py.

#### 5.5.1.16 latex_documents

```
list conf.latex_documents
```

**Initial value:**

```
1 =  [
2     (master_doc, 'MyProject.tex', 'MyProject Documentation',
3      'My Name', 'manual'),
4 ]
```

Definition at line 143 of file conf.py.

#### 5.5.1.17 latex_elements

```
dictionary conf.latex_elements
```

**Initial value:**

```
1 =  {
2     # The paper size ('letterpaper' or 'a4paper').
3     #
4     'papersize': 'letterpaper',
5
6     # The font size ('10pt', '11pt' or '12pt').
7     #
8     'pointsize': '10pt',
9
10     # Additional stuff for the LaTeX preamble.
11     #
12     'preamble': '',
13
14     # Latex figure (float) alignment
15     #
16     'figure_align': 'htbp',
17 }
```

Definition at line 122 of file conf.py.

#### 5.5.1.18 man_pages

```
list conf.man_pages
```

**Initial value:**

```
1 =  [
2     (master_doc, 'myproject', 'MyProject Documentation',
3      [author], 1)
4 ]
```

Definition at line 153 of file conf.py.

#### 5.5.1.19 master_doc

```
string conf.master_doc = 'index'
```

Definition at line 48 of file conf.py.

#### 5.5.1.20 project

```
string conf.project = 'MyProject'
```

Definition at line 51 of file conf.py.

#### 5.5.1.21 pygments_style

```
string conf.pygments_style = 'sphinx'
```

Definition at line 77 of file conf.py.

#### 5.5.1.22 release

```
string conf.release = '0.0.1'
```

Definition at line 62 of file conf.py.

#### 5.5.1.23 source_suffix

```
string conf.source_suffix = '.rst'
```

Definition at line 45 of file conf.py.

#### 5.5.1.24 templates_path

```
list conf.templates_path = ['_templates']
```

Definition at line 39 of file conf.py.

#### 5.5.1.25 texinfo_documents

```
list conf.texinfo_documents
```

**Initial value:**

```
1 =  [
2     (master_doc, 'MyProject', 'MyProject Documentation',
3      author, 'MyProject', 'One line description of project.',
4      'Miscellaneous'),
5 ]
```

Definition at line 164 of file conf.py.

#### 5.5.1.26 todo_include_todos

```
bool conf.todo_include_todos = True
```

Definition at line 80 of file conf.py.

#### 5.5.1.27 version

```
string conf.version = '0.0.1'
```

Definition at line 60 of file conf.py.

## 5.6 NodoMinador Namespace Reference

### Classes

- class NodoMinador

## 5.7 PoolMinado Namespace Reference

### Classes

- class PoolMinado

## 5.8 pruebasUnidad Namespace Reference

### Functions

- def main ()

### 5.8.1 Function Documentation

#### 5.8.1.1 main()

```
def pruebasUnidad.main ( )
```

Definition at line 8 of file pruebasUnidad.py.

## 5.9 pruebasUnidadBloque Namespace Reference

### Classes

- class pruebasUnidadBloque

## 5.10 pruebasUnidadCadena Namespace Reference

### Classes

- class pruebasUnidadCadena

## 5.11 pruebasUnidadNodoMineria Namespace Reference

### Classes

- class pruebasUnidadNodoMineria

## 5.12 pruebasUnidadPoolMinado Namespace Reference

### Classes

- class pruebasUnidadPoolMinado

## 5.13 pruebasUnidadSimulador Namespace Reference

### Classes

- class pruebasUnidadSimulador

## 5.14 pruebasUnidadTransaccion Namespace Reference

### Classes

- class pruebasUnidadTransaccion

## 5.15 pruebasUnidadUsuario Namespace Reference

### Classes

- class pruebasUnidadUsuario

## 5.16 Simulador Namespace Reference

### Classes

- class Simulador

## 5.17 Transaccion Namespace Reference

### Classes

- class Transaccion

## 5.18 Usuario Namespace Reference

### Classes

- class Usuario

# Chapter 6

# Class Documentation

## 6.1 Bloque.Bloque Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, indice, minado, hashBloque, transacciones, marcaTiempo=None)
- def gethashBloque (self)
- def setHashBloque (self)
- def __str__ (self)
- def __repr__ (self)
- def consultarBloque (self)

### Static Public Member Functions

- def generarBloqueHash (self)

### Private Attributes

- _indice
- _minado
- _hashBloque
- _transacciones
- _marcaTiempo

### 6.1.1 Detailed Description

Definition at line 11 of file Bloque.py.

### 6.1.2 Constructor & Destructor Documentation

#### 6.1.2.1 __init__()

```
def Bloque.Bloque.__init__ (
              self,
              indice,
              minado,
              hashBloque,
              transacciones,
              marcaTiempo = None )
```

```
Inicialización del bloque
:param indice: Valor entero que representa la posición de cada bloque en la cadena
:param minado: Número entero usado en el proceso de minería
:param hashBloque:  hash del bloque actual en la cadena
:param transacciones: Conjunto de transacciones del bloque
:param marcaTiempo: Instante de tiempo de creación del nuevo bloque
```

Definition at line 13 of file Bloque.py.

### 6.1.3 Member Function Documentation

#### 6.1.3.1 __repr__()

```
def Bloque.Bloque.__repr__ (
              self )
```

Definition at line 54 of file Bloque.py.

#### 6.1.3.2 __str__()

```
def Bloque.Bloque.__str__ (
              self )
```

```
Genera un string con la información del bloque
:return: representación en String del bloque
```

Definition at line 47 of file Bloque.py.

#### 6.1.3.3 consultarBloque()

```
def Bloque.Bloque.consultarBloque (
              self )
```

Definition at line 57 of file Bloque.py.

#### 6.1.3.4 generarBloqueHash()

```
def Bloque.Bloque.generarBloqueHash (
              self )  [static]
```

```
Genera Hash la información del bloque
:return: Hash construido a partir de la información del bloque
```

Definition at line 39 of file Bloque.py.

#### 6.1.3.5 gethashBloque()

```
def Bloque.Bloque.gethashBloque (
              self )
```

```
Obtiene el valor actual del hash del bloque
:return: Hash del bloque
```

Definition at line 28 of file Bloque.py.

#### 6.1.3.6 setHashBloque()

```
def Bloque.Bloque.setHashBloque (
              self )
```

Definition at line 35 of file Bloque.py.

### 6.1.4 Member Data Documentation

#### 6.1.4.1 _hashBloque

`Bloque.Bloque._hashBloque [private]`

Definition at line 24 of file Bloque.py.

#### 6.1.4.2 _indice

`Bloque.Bloque._indice [private]`

Definition at line 22 of file Bloque.py.

#### 6.1.4.3 _marcaTiempo

`Bloque.Bloque._marcaTiempo [private]`

Definition at line 26 of file Bloque.py.

#### 6.1.4.4 _minado

`Bloque.Bloque._minado [private]`

Definition at line 23 of file Bloque.py.

#### 6.1.4.5 _transacciones

`Bloque.Bloque._transacciones [private]`

Definition at line 25 of file Bloque.py.

The documentation for this class was generated from the following file:

- cadena/Bloque.py

## 6.2 blockchainsimulacion.Bloque Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, indice, minado, hashBloque, transacciones, marcaTiempo=None)
- def gethashBloque (self)
- def setHashBloque (self)
- def __str__ (self)
- def __repr__ (self)

### Static Public Member Functions

- def generarBloqueHash (self)

### Private Attributes

- _indice
- _minado
- _hashBloque
- _transacciones
- _marcaTiempo

## 6.2.1 Detailed Description

Definition at line 243 of file blockchainsimulacion.py.

## 6.2.2 Constructor & Destructor Documentation

### 6.2.2.1 __init__()

```
def blockchainsimulacion.Bloque.__init__ (
              self,
              indice,
              minado,
              hashBloque,
              transacciones,
              marcaTiempo = None )
```

```
Inicialización del bloque
:param indice: Valor entero que representa la posición de cada bloque en la cadena
:param minado: Número entero usado en el proceso de minería
:param hashBloque:  hash del bloque actual en la cadena
:param transacciones: Conjunto de transacciones del bloque
:param marcaTiempo: Instante de tiempo de creación del nuevo bloque
```

Definition at line 245 of file blockchainsimulacion.py.

### 6.2.3 Member Function Documentation

#### 6.2.3.1 __repr__()

```
def blockchainsimulacion.Bloque.__repr__ (
              self )
```

Definition at line 286 of file blockchainsimulacion.py.

#### 6.2.3.2 __str__()

```
def blockchainsimulacion.Bloque.__str__ (
              self )

Genera un string con la información del bloque
:return: representación en String del bloque
```

Definition at line 279 of file blockchainsimulacion.py.

#### 6.2.3.3 generarBloqueHash()

```
def blockchainsimulacion.Bloque.generarBloqueHash (
              self )  [static]

Genera Hash la información del bloque
:return: Hash construido a partir de la información del bloque
```

Definition at line 271 of file blockchainsimulacion.py.

#### 6.2.3.4 gethashBloque()

```
def blockchainsimulacion.Bloque.gethashBloque (
              self )

Obtiene el valor actual del hash del bloque
:return: Hash del bloque
```

Definition at line 260 of file blockchainsimulacion.py.

#### 6.2.3.5 setHashBloque()

```
def blockchainsimulacion.Bloque.setHashBloque (
              self )
```

Definition at line 267 of file blockchainsimulacion.py.

### 6.2.4 Member Data Documentation

#### 6.2.4.1 _hashBloque

```
blockchainsimulacion.Bloque._hashBloque  [private]
```

Definition at line 256 of file blockchainsimulacion.py.

#### 6.2.4.2 _indice

```
blockchainsimulacion.Bloque._indice  [private]
```

Definition at line 254 of file blockchainsimulacion.py.

#### 6.2.4.3 _marcaTiempo

```
blockchainsimulacion.Bloque._marcaTiempo  [private]
```

Definition at line 258 of file blockchainsimulacion.py.

#### 6.2.4.4 _minado

```
blockchainsimulacion.Bloque._minado  [private]
```

Definition at line 255 of file blockchainsimulacion.py.

#### 6.2.4.5 _transacciones

`blockchainsimulacion.Bloque._transacciones [private]`

Definition at line 257 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.3 CadenaBloques.CadenaBloques Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self)
- def getCadenaSerializada (self)
- def bloqueGenesis (self, trs)
- def nuevoBloque (self, minado, hashBloque, trans)
- def obtenerUltimoBloque (self)
- def validarBlockchain (self)
- def __repr__ (self)
- def __str__ (self)

### Static Public Member Functions

- def bloqueEsValido (nuevoBloque, ultimoBloque)

### Private Attributes

- _cadena

### 6.3.1 Detailed Description

Definition at line 12 of file CadenaBloques.py.

### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 __init__()

```
def CadenaBloques.CadenaBloques.__init__ (
              self )
```

```
Inicialización de la blockchain
:param _cadena: Estructura de datos tipo array para almacenar la cadena de bloques.
```

Definition at line 14 of file CadenaBloques.py.

### 6.3.3 Member Function Documentation

#### 6.3.3.1 __repr__()

```
def CadenaBloques.CadenaBloques.__repr__ (
              self )
```

Definition at line 108 of file CadenaBloques.py.

#### 6.3.3.2 __str__()

```
def CadenaBloques.CadenaBloques.__str__ (
              self )
```

```
Genera un string con la información de la cadena
:return: representación en String del nodo
```

Definition at line 111 of file CadenaBloques.py.

#### 6.3.3.3 bloqueEsValido()

```
def CadenaBloques.CadenaBloques.bloqueEsValido (
              nuevoBloque,
              ultimoBloque )  [static]
```

```
Verifica bajo cuatro reglas básicas la validez de un Bloque
Regla 1: Secuencia ordenada de bloques por valor del índice
Regla 2: Secuencia ordenada de bloques por fecha de creación
Regla 3: Hash diferente entre el último bloque de la cadena y el nuevo bloque
Regla 4: Bloque correctamente minado
@return True si el bloque es válido, False en otro caso
```

Definition at line 65 of file CadenaBloques.py.

#### 6.3.3.4 bloqueGenesis()

```
def CadenaBloques.CadenaBloques.bloqueGenesis (
              self,
              trs )
```

```
Crea el primer bloque en la cadena. Es el bloque seminal.
```

Definition at line 34 of file CadenaBloques.py.

#### 6.3.3.5 getCadenaSerializada()

```
def CadenaBloques.CadenaBloques.getCadenaSerializada (
              self )
```

```
Serializa la cadena de bloques en un documento JSON
:return conjunto de bloques en formato JSON
```

Definition at line 22 of file CadenaBloques.py.

#### 6.3.3.6 nuevoBloque()

```
def CadenaBloques.CadenaBloques.nuevoBloque (
              self,
              minado,
              hashBloque,
              trans )
```

```
Construye e inserta el siguiente bloque de la cadena
:param minado Método de minado del bloque
:param hashBloque Hash generado para el bloque
:return Bloque nuevo bloque generado
```

Definition at line 40 of file CadenaBloques.py.

#### 6.3.3.7 obtenerUltimoBloque()

```
def CadenaBloques.CadenaBloques.obtenerUltimoBloque (
              self )
```

```
Devuelve una referencia al último nodo de la blockchain
@return último nodo
```

Definition at line 57 of file CadenaBloques.py.

#### 6.3.3.8 validarBlockchain()

```
def CadenaBloques.CadenaBloques.validarBlockchain (
              self )
```

```
Verifica la integridad de la cadena de bloques.
para este procedimiento se valida que el índice del siguiente bloque sea
menor en una unidad del bloque anterior y que el hash de cada bloque sea
correcto según la política de minado establecida, números primos en este caso.
@return True si la cadena esta bien formada, False en otro caso
```

Definition at line 86 of file CadenaBloques.py.

### 6.3.4 Member Data Documentation

#### 6.3.4.1 _cadena

```
CadenaBloques.CadenaBloques._cadena  [private]
```

Definition at line 19 of file CadenaBloques.py.

The documentation for this class was generated from the following file:

- cadena/CadenaBloques.py

## 6.4 blockchainsimulacion.CadenaBloques Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self)
- def getCadenaSerializada (self)
- def bloqueGenesis (self, trs)
- def nuevoBloque (self, minado, hashBloque, trans)
- def obtenerUltimoBloque (self)
- def validarBlockchain (self)
- def __repr__ (self)

### Static Public Member Functions

- def bloqueEsValido (nuevoBloque, ultimoBloque)

### Private Attributes

- _cadena

## 6.4.1 Detailed Description

Definition at line 503 of file blockchainsimulacion.py.

## 6.4.2 Constructor & Destructor Documentation

### 6.4.2.1 __init__()

```
def blockchainsimulacion.CadenaBloques.__init__ (
              self )
```

```
Inicialización de la blockchain
:param _cadena: Estructura de datos tipo array para almacenar la cadena de bloques.
```

Definition at line 505 of file blockchainsimulacion.py.

## 6.4.3 Member Function Documentation

#### 6.4.3.1 __repr__()

```
def blockchainsimulacion.CadenaBloques.__repr__ (
              self )
```

Definition at line 597 of file blockchainsimulacion.py.

#### 6.4.3.2 bloqueEsValido()

```
def blockchainsimulacion.CadenaBloques.bloqueEsValido (
              nuevoBloque,
              ultimoBloque )  [static]
```

```
Verifica bajo cuatro reglas básicas la validez de un Bloque
Regla 1: Secuencia ordenada de bloques por valor del índice
Regla 2: Secuencia ordenada de bloques por fecha de creación
Regla 3: Hash diferente entre el último bloque de la cadena y el nuevo bloque
Regla 4: Bloque correctamente minado
@return True si el bloque es válido, False en otro caso
```

Definition at line 554 of file blockchainsimulacion.py.

#### 6.4.3.3 bloqueGenesis()

```
def blockchainsimulacion.CadenaBloques.bloqueGenesis (
              self,
              trs )
```

```
Crea el primer bloque en la cadena. Es el bloque seminal.
```

Definition at line 523 of file blockchainsimulacion.py.

#### 6.4.3.4 getCadenaSerializada()

```
def blockchainsimulacion.CadenaBloques.getCadenaSerializada (
              self )
```

```
Serializa la cadena de bloques en un documento JSON
:return conjunto de bloques en formato JSON
```

Definition at line 513 of file blockchainsimulacion.py.

#### 6.4.3.5 nuevoBloque()

```
def blockchainsimulacion.CadenaBloques.nuevoBloque (
              self,
              minado,
              hashBloque,
              trans )


Construye e inserta el siguiente bloque de la cadena
:param minado Método de minado del bloque
:param hashBloque Hash generado para el bloque
:return Bloque nuevo bloque generado
```

Definition at line 529 of file blockchainsimulacion.py.

#### 6.4.3.6 obtenerUltimoBloque()

```
def blockchainsimulacion.CadenaBloques.obtenerUltimoBloque (
              self )


Devuelve una referencia al último nodo de la blockchain
@return último nodo
```

Definition at line 546 of file blockchainsimulacion.py.

#### 6.4.3.7 validarBlockchain()

```
def blockchainsimulacion.CadenaBloques.validarBlockchain (
              self )


Verifica la integridad de la cadena de bloques.
para este procedimiento se valida que el índice del siguiente bloque sea
menor en una unidad del bloque anterior y que el hash de cada bloque sea
correcto según la política de minado establecida, números primos en este caso.
@return True si la cadena esta bien formada, False en otro caso
```

Definition at line 575 of file blockchainsimulacion.py.

### 6.4.4 Member Data Documentation

#### 6.4.4.1 _cadena

```
blockchainsimulacion.CadenaBloques._cadena  [private]
```

Definition at line 510 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.5 blockchainsimulacion.NodoMinador Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, direccion, puerto, descripcion, usuariosMineros)
- def actualizarCadena (self, nuevaCadena)
- def infoCadena (self)
- def __str__ (self)

### Private Attributes

- _direccion
- _puerto
- _descripcion
- _utilidad
- _cantidadBloquesMinados
- _fechaCreacion
- _hashRateNodo
- _cadenaBloques
- _usuariosMineros

### 6.5.1 Detailed Description

Definition at line 359 of file blockchainsimulacion.py.

### 6.5.2 Constructor & Destructor Documentation

#### 6.5.2.1 __init__()

```
def blockchainsimulacion.NodoMinador.__init__ (
              self,
              direccion,
              puerto,
              descripcion,
              usuariosMineros )
```

```
Creación de un nodo para la minería de blockchain
:param _direccción: Dirección ip del nodo minero
:param _puerto:  puerto TCP/UDP donde se publica el servicio
:param _descripcion: información adicional del nodo minador
:param _fechaCreacion: fecha de adición del nodo a la red
:param _cadenaBloques: Recibe una copia de la cadena de bloques
:param _hashRateNodo : Capacidad de resolución de hash por segundo del nodo
:param _usuariosMineros: Conjunto de usuarios participantes en el nodo minero
```

Definition at line 361 of file blockchainsimulacion.py.

### 6.5.3 Member Function Documentation

#### 6.5.3.1 __str__()

```
def blockchainsimulacion.NodoMinador.__str__ (
              self )
```

```
Genera un string con la información de un nodo de minería
:return: representación en String del nodo
```

Definition at line 397 of file blockchainsimulacion.py.

#### 6.5.3.2 actualizarCadena()

```
def blockchainsimulacion.NodoMinador.actualizarCadena (
              self,
              nuevaCadena )
```

```
El proceso de minería requiere que el nodo actualice la blockchain
:param: Cadena de bloques actualizada
```

Definition at line 383 of file blockchainsimulacion.py.

#### 6.5.3.3 infoCadena()

```
def blockchainsimulacion.NodoMinador.infoCadena (
              self )
```

```
Genera un string con la información básica de la cadena de bloques
:return: una cadena con el reporte del total de nodos de la cadena
```

Definition at line 390 of file blockchainsimulacion.py.

### 6.5.4 Member Data Documentation

#### 6.5.4.1 _cadenaBloques

```
blockchainsimulacion.NodoMinador._cadenaBloques  [private]
```

Definition at line 379 of file blockchainsimulacion.py.

#### 6.5.4.2 _cantidadBloquesMinados

```
blockchainsimulacion.NodoMinador._cantidadBloquesMinados  [private]
```

Definition at line 376 of file blockchainsimulacion.py.

#### 6.5.4.3 _descripcion

```
blockchainsimulacion.NodoMinador._descripcion  [private]
```

Definition at line 374 of file blockchainsimulacion.py.

#### 6.5.4.4 _direccion

```
blockchainsimulacion.NodoMinador._direccion  [private]
```

Definition at line 372 of file blockchainsimulacion.py.

#### 6.5.4.5 _fechaCreacion

`blockchainsimulacion.NodoMinador._fechaCreacion [private]`

Definition at line 377 of file blockchainsimulacion.py.

#### 6.5.4.6 _hashRateNodo

`blockchainsimulacion.NodoMinador._hashRateNodo [private]`

Definition at line 378 of file blockchainsimulacion.py.

#### 6.5.4.7 _puerto

`blockchainsimulacion.NodoMinador._puerto [private]`

Definition at line 373 of file blockchainsimulacion.py.

#### 6.5.4.8 _usuariosMineros

`blockchainsimulacion.NodoMinador._usuariosMineros [private]`

Definition at line 380 of file blockchainsimulacion.py.

#### 6.5.4.9 _utilidad

`blockchainsimulacion.NodoMinador._utilidad [private]`

Definition at line 375 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.6 NodoMinador.NodoMinador Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, direccion, puerto, descripcion, usuariosMineros)
- def actualizarCadena (self, nuevaCadena)
- def infoCadena (self)
- def __str__ (self)

### Private Attributes

- _direccion
- _puerto
- _descripcion
- _utilidad
- _cantidadBloquesMinados
- _fechaCreacion
- _hashRateNodo
- _cadenaBloques
- _usuariosMineros

## 6.6.1 Detailed Description

Definition at line 9 of file NodoMinador.py.

## 6.6.2 Constructor & Destructor Documentation

### 6.6.2.1 __init__()

```
def NodoMinador.NodoMinador.__init__ (
              self,
              direccion,
              puerto,
              descripcion,
              usuariosMineros )
```

```
Creación de un nodo para la minería de blockchain
:param _direcccción: Dirección ip del nodo minero
:param _puerto:  puerto TCP/UDP donde se publica el servicio
:param _descripcion: información adicional del nodo minador
:param _fechaCreacion: fecha de adición del nodo a la red
:param _cadenaBloques: Recibe una copia de la cadena de bloques
:param _hashRateNodo : Capacidad de resolución de hash por segundo del nodo
:param _usuariosMineros: Conjunto de usuarios participantes en el nodo minero
```

Definition at line 11 of file NodoMinador.py.

### 6.6.3 Member Function Documentation

#### 6.6.3.1 __str__()

```
def NodoMinador.NodoMinador.__str__ (
              self )
```

```
Genera un string con la información de un nodo de minería
:return: representación en String del nodo
```

Definition at line 47 of file NodoMinador.py.

#### 6.6.3.2 actualizarCadena()

```
def NodoMinador.NodoMinador.actualizarCadena (
              self,
              nuevaCadena )
```

```
El proceso de minería requiere que el nodo actualice la blockchain
:param: Cadena de bloques actualizada
```

Definition at line 33 of file NodoMinador.py.

#### 6.6.3.3 infoCadena()

```
def NodoMinador.NodoMinador.infoCadena (
              self )
```

```
Genera un string con la información básica de la cadena de bloques
:return: una cadena con el reporte del total de nodos de la cadena
```

Definition at line 40 of file NodoMinador.py.

### 6.6.4 Member Data Documentation

#### 6.6.4.1 _cadenaBloques

`NodoMinador.NodoMinador._cadenaBloques [private]`

Definition at line 29 of file NodoMinador.py.

#### 6.6.4.2 _cantidadBloquesMinados

`NodoMinador.NodoMinador._cantidadBloquesMinados [private]`

Definition at line 26 of file NodoMinador.py.

#### 6.6.4.3 _descripcion

`NodoMinador.NodoMinador._descripcion [private]`

Definition at line 24 of file NodoMinador.py.

#### 6.6.4.4 _direccion

`NodoMinador.NodoMinador._direccion [private]`

Definition at line 22 of file NodoMinador.py.

#### 6.6.4.5 _fechaCreacion

`NodoMinador.NodoMinador._fechaCreacion [private]`

Definition at line 27 of file NodoMinador.py.

#### 6.6.4.6 _hashRateNodo

`NodoMinador.NodoMinador._hashRateNodo [private]`

Definition at line 28 of file NodoMinador.py.

#### 6.6.4.7 _puerto

`NodoMinador.NodoMinador._puerto [private]`

Definition at line 23 of file NodoMinador.py.

#### 6.6.4.8 _usuariosMineros

`NodoMinador.NodoMinador._usuariosMineros [private]`

Definition at line 30 of file NodoMinador.py.

#### 6.6.4.9 _utilidad

`NodoMinador.NodoMinador._utilidad [private]`

Definition at line 25 of file NodoMinador.py.

The documentation for this class was generated from the following file:

- cadena/NodoMinador.py

## 6.7 PoolMinado.PoolMinado Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self)
- def adicionarMinero (self, minador)
- def minarBloque (self, nodoMinero, numeroNodo, output)
- def torneoPorNuevoBloque (self, transacciones)
- def sincronizarCadena (self, nuevaCadena)
- def __str__ (self)

### Static Public Member Functions

- def pruebaDeTrabajo (indiceUltimoMinado)
- def esPrimo (numero)
- def minadoBloque (nodoMinero, transacciones)

### Private Attributes

- _nodos
- _hashRate
- _recompensaPorBloque
- _dificultad

## 6.7.1 Detailed Description

Definition at line 12 of file PoolMinado.py.

## 6.7.2 Constructor & Destructor Documentation

### 6.7.2.1 __init__()

```
def PoolMinado.PoolMinado.__init__ (
              self )
```

```
Inicialización del pool de minería
:param _nodos: conjunto de nodos mineros
:param _hasRate Simula la capacidad de computo del pool de mineria
:param _recompensaPorBloque cantidad que se recompensa a los nodos que resuelvan el problema
:param _dificultad  valor usado para simular el incremento de dificultad del proceso de minado
```

Definition at line 14 of file PoolMinado.py.

## 6.7.3 Member Function Documentation

### 6.7.3.1 __str__()

```
def PoolMinado.PoolMinado.__str__ (
              self )
```

```
Genera un string con la información del pool de minería
:return: representación en String del nodo
```

Definition at line 148 of file PoolMinado.py.

#### 6.7.3.2 adicionarMinero()

```
def PoolMinado.PoolMinado.adicionarMinero (
              self,
              minador )
```

```
Agrega  un nodo minero al conjunto de nodos minadores de la blockchain. Si el nodo adicionado aumenta
el hashrate del pool ylo iguala o supera el 60% de la dificultad, esta última se incrementa en un
20%
@param minador: Nuevo nodo minero
@return True si el nodo de mineria se puede adicionar, False en otro caso
```

Definition at line 27 of file PoolMinado.py.

#### 6.7.3.3 esPrimo()

```
def PoolMinado.PoolMinado.esPrimo (
              numero )  [static]
```

```
Determina si un número es primo
@param numero, Número que necesitamos determinar si es primo
@return True si numero es primo, False en otro caso
```

Definition at line 58 of file PoolMinado.py.

#### 6.7.3.4 minadoBloque()

```
def PoolMinado.PoolMinado.minadoBloque (
              nodoMinero,
              transacciones )  [static]
```

```
Realiza el proceso de minado de un nuevo bloque en la cadena
@param nodoMinero que realiza el intento de minar el bloque
@param transacciones: Conjunto de Transacciones del nuevo bloque
```

Definition at line 131 of file PoolMinado.py.

#### 6.7.3.5 minarBloque()

```
def PoolMinado.PoolMinado.minarBloque (
              self,
              nodoMinero,
              numeroNodo,
              output )
```

```
Algoritmo que realiza el minado de un bloque para un nodo en especifico.
@param nodoMinero, Nodo minero que realizará la prueba de trabajo
@param numeroNodo, identificador entero del nodo minero en el pool de minería
@param output, Parámetro de salida donde se conserva la información de
la prueba de trabajo ejecutada por el iésimo nodo minero.
@return minadoNuevo Valor de minado obtenido para el nuevo bloque
```

Definition at line 74 of file PoolMinado.py.

#### 6.7.3.6 pruebaDeTrabajo()

```
def PoolMinado.PoolMinado.pruebaDeTrabajo (
              indiceUltimoMinado )  [static]
```

```
Algoritmo de minado por prueba de trabajo.
En esta versión se implementa una prueba basada en números primos, en la cual el
valor de minado del bloque actual y del último bloque de la cadena debe ser un número primo.
@param indiceUltimoMinado, Valor generado por la minería para el último bloque de la cadena
@return minadoNuevo Valor de minado obtenido para el nuevo bloque
```

Definition at line 43 of file PoolMinado.py.

#### 6.7.3.7 sincronizarCadena()

```
def PoolMinado.PoolMinado.sincronizarCadena (
              self,
              nuevaCadena )
```

```
Realiza una sincronización de la cadena actualizada a todos los mineros
@param nuevaCadena, Cadena con el nuevo bloque
```

Definition at line 140 of file PoolMinado.py.

#### 6.7.3.8 torneoPorNuevoBloque()

```
def PoolMinado.PoolMinado.torneoPorNuevoBloque (
              self,
              transacciones )
```

```
Realiza un torneo para la adjudicación del nuevo bloque a un nodo minero.
El nodo ganador será el que realice mayor trabajo.
@param transacciones, Bloque de transacciones que tendrá el nuevo bloque de la cadena
```

Definition at line 94 of file PoolMinado.py.

### 6.7.4 Member Data Documentation

#### 6.7.4.1 _dificultad

```
PoolMinado.PoolMinado._dificultad  [private]
```

Definition at line 25 of file PoolMinado.py.

#### 6.7.4.2 _hashRate

```
PoolMinado.PoolMinado._hashRate  [private]
```

Definition at line 23 of file PoolMinado.py.

#### 6.7.4.3 _nodos

```
PoolMinado.PoolMinado._nodos  [private]
```

Definition at line 22 of file PoolMinado.py.

#### 6.7.4.4 _recompensaPorBloque

`PoolMinado.PoolMinado._recompensaPorBloque [private]`

Definition at line 24 of file PoolMinado.py.

The documentation for this class was generated from the following file:

- cadena/PoolMinado.py

## 6.8 blockchainsimulacion.PoolMinado Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self)
- def adicionarMinero (self, minador)
- def minarBloque (self, nodoMinero, numeroNodo, output)
- def torneoPorNuevoBloque (self, transacciones)
- def sincronizarCadena (self, nuevaCadena)
- def __str__ (self)

### Static Public Member Functions

- def pruebaDeTrabajo (indiceUltimoMinado)
- def esPrimo (numero)
- def minadoBloque (nodoMinero, transacciones)

### Private Attributes

- _nodos
- _hashRate
- _recompensaPorBloque
- _dificultad

### 6.8.1 Detailed Description

Definition at line 710 of file blockchainsimulacion.py.

### 6.8.2 Constructor & Destructor Documentation

#### 6.8.2.1 __init__()

```
def blockchainsimulacion.PoolMinado.__init__ (
              self )
```

```
Inicialización del pool de minería
:param _nodos: conjunto de nodos mineros
:param _hasRate Simula la capacidad de computo del pool de mineria
:param _recompensaPorBloque cantidad que se recompensa a los nodos que resuelvan el problema
:param _dificultad  valor usado para simular el incremento de dificultad del proceso de minado
```

Definition at line 712 of file blockchainsimulacion.py.

### 6.8.3 Member Function Documentation

#### 6.8.3.1 __str__()

```
def blockchainsimulacion.PoolMinado.__str__ (
              self )
```

```
Genera un string con la información del pool de minería
:return: representación en String del nodo
```

Definition at line 846 of file blockchainsimulacion.py.

#### 6.8.3.2 adicionarMinero()

```
def blockchainsimulacion.PoolMinado.adicionarMinero (
              self,
              minador )
```

```
Agrega  un nodo minero al conjunto de nodos minadores de la blockchain. Si el nodo adicionado aumenta
el hashrate del pool ylo iguala o supera el 60% de la dificultad, esta última se incrementa en un
20%
@param minador: Nuevo nodo minero
@return True si el nodo de mineria se puede adicionar, False en otro caso
```

Definition at line 725 of file blockchainsimulacion.py.

#### 6.8.3.3 esPrimo()

```
def blockchainsimulacion.PoolMinado.esPrimo (
              numero )  [static]
```

```
Determina si un número es primo
@param numero, Número que necesitamos determinar si es primo
@return True si numero es primo, False en otro caso
```

Definition at line 756 of file blockchainsimulacion.py.

#### 6.8.3.4 minadoBloque()

```
def blockchainsimulacion.PoolMinado.minadoBloque (
              nodoMinero,
              transacciones )  [static]
```

```
Realiza el proceso de minado de un nuevo bloque en la cadena
@param nodoMinero que realiza el intento de minar el bloque
@param transacciones: Conjunto de Transacciones del nuevo bloque
```

Definition at line 829 of file blockchainsimulacion.py.

#### 6.8.3.5 minarBloque()

```
def blockchainsimulacion.PoolMinado.minarBloque (
              self,
              nodoMinero,
              numeroNodo,
              output )
```

```
Algoritmo que realiza el minado de un bloque para un nodo en especifico.
@param nodoMinero, Nodo minero que realizará la prueba de trabajo
@param numeroNodo, identificador entero del nodo minero en el pool de minería
@param output, Parámetro de salida donde se conserva la información de
la prueba de trabajo ejecutada por el iésimo nodo minero.
@return minadoNuevo Valor de minado obtenido para el nuevo bloque
```

Definition at line 772 of file blockchainsimulacion.py.

#### 6.8.3.6 pruebaDeTrabajo()

```
def blockchainsimulacion.PoolMinado.pruebaDeTrabajo (
              indiceUltimoMinado )  [static]
```

```
Algoritmo de minado por prueba de trabajo.
En esta versión se implementa una prueba basada en números primos, en la cual el
valor de minado del bloque actual y del último bloque de la cadena debe ser un número primo.
@param indiceUltimoMinado, Valor generado por la minería para el último bloque de la cadena
@return minadoNuevo Valor de minado obtenido para el nuevo bloque
```

Definition at line 741 of file blockchainsimulacion.py.

#### 6.8.3.7 sincronizarCadena()

```
def blockchainsimulacion.PoolMinado.sincronizarCadena (
              self,
              nuevaCadena )
```

```
Realiza una sincronización de la cadena actualizada a todos los mineros
@param nuevaCadena, Cadena con el nuevo bloque
```

Definition at line 838 of file blockchainsimulacion.py.

#### 6.8.3.8 torneoPorNuevoBloque()

```
def blockchainsimulacion.PoolMinado.torneoPorNuevoBloque (
              self,
              transacciones )
```

```
Realiza un torneo para la adjudicación del nuevo bloque a un nodo minero.
El nodo ganador será el que realice mayor trabajo.
@param transacciones, Bloque de transacciones que tendrá el nuevo bloque de la cadena
```

Definition at line 792 of file blockchainsimulacion.py.

### 6.8.4 Member Data Documentation

#### 6.8.4.1 _dificultad

`blockchainsimulacion.PoolMinado._dificultad [private]`

Definition at line 723 of file blockchainsimulacion.py.

#### 6.8.4.2 _hashRate

`blockchainsimulacion.PoolMinado._hashRate [private]`

Definition at line 721 of file blockchainsimulacion.py.

#### 6.8.4.3 _nodos

`blockchainsimulacion.PoolMinado._nodos [private]`

Definition at line 720 of file blockchainsimulacion.py.

#### 6.8.4.4 _recompensaPorBloque

`blockchainsimulacion.PoolMinado._recompensaPorBloque [private]`

Definition at line 722 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.9 blockchainsimulacion.pruebasUnidadBloque Class Reference

Inherits object.

### Public Member Functions

- def main ()

### 6.9.1 Detailed Description

Definition at line 312 of file blockchainsimulacion.py.

### 6.9.2 Member Function Documentation

#### 6.9.2.1 main()

```
def blockchainsimulacion.pruebasUnidadBloque.main ( )
```

Definition at line 314 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.10 pruebasUnidadBloque.pruebasUnidadBloque Class Reference

Inherits object.

### Public Member Functions

- def main ()

### 6.10.1 Detailed Description

Definition at line 16 of file pruebasUnidadBloque.py.

### 6.10.2 Member Function Documentation

#### 6.10.2.1 main()

```
def pruebasUnidadBloque.pruebasUnidadBloque.main ( )
```

Definition at line 18 of file pruebasUnidadBloque.py.

The documentation for this class was generated from the following file:

- cadena/pruebasUnidadBloque.py

## 6.11 blockchainsimulacion.pruebasUnidadCadena Class Reference

Inherits object.

### Public Member Functions

- def main ()

### 6.11.1 Detailed Description

Definition at line 627 of file blockchainsimulacion.py.

### 6.11.2 Member Function Documentation

#### 6.11.2.1 main()

```
def blockchainsimulacion.pruebasUnidadCadena.main ( )
```

Definition at line 628 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.12 pruebasUnidadCadena.pruebasUnidadCadena Class Reference

Inherits object.

### Public Member Functions

- def main ()

### 6.12.1 Detailed Description

Definition at line 21 of file pruebasUnidadCadena.py.

### 6.12.2 Member Function Documentation

#### 6.12.2.1 main()

```
def pruebasUnidadCadena.pruebasUnidadCadena.main ( )
```

Definition at line 22 of file pruebasUnidadCadena.py.

The documentation for this class was generated from the following file:

- cadena/pruebasUnidadCadena.py

## 6.13 blockchainsimulacion.pruebasUnidadNodoMineria Class Reference

Inherits object.

### Public Member Functions

- def main ()

### 6.13.1 Detailed Description

Definition at line 428 of file blockchainsimulacion.py.

### 6.13.2 Member Function Documentation

#### 6.13.2.1 main()

```
def blockchainsimulacion.pruebasUnidadNodoMineria.main ( )
```

Definition at line 429 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.14 pruebasUnidadNodoMineria.pruebasUnidadNodoMineria Class Reference

Inherits object.

### Public Member Functions

- def main ()

### 6.14.1 Detailed Description

Definition at line 16 of file pruebasUnidadNodoMineria.py.

### 6.14.2 Member Function Documentation

#### 6.14.2.1 main()

```
def pruebasUnidadNodoMineria.pruebasUnidadNodoMineria.main ( )
```

Definition at line 17 of file pruebasUnidadNodoMineria.py.

The documentation for this class was generated from the following file:

- cadena/pruebasUnidadNodoMineria.py

## 6.15 pruebasUnidadPoolMinado.pruebasUnidadPoolMinado Class Reference

Inherits Thread.

### Public Member Functions

- def main ()

### 6.15.1 Detailed Description

Definition at line 17 of file pruebasUnidadPoolMinado.py.

### 6.15.2 Member Function Documentation

#### 6.15.2.1 main()

```
def pruebasUnidadPoolMinado.pruebasUnidadPoolMinado.main ( )
```

Definition at line 18 of file pruebasUnidadPoolMinado.py.

The documentation for this class was generated from the following file:

- cadena/pruebasUnidadPoolMinado.py

## 6.16 blockchainsimulacion.pruebasUnidadPoolMinado Class Reference

Inherits Thread.

### Public Member Functions

- def main ()

### 6.16.1 Detailed Description

Definition at line 880 of file blockchainsimulacion.py.

### 6.16.2 Member Function Documentation

#### 6.16.2.1 main()

```
def blockchainsimulacion.pruebasUnidadPoolMinado.main ( )
```

Definition at line 881 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.17 pruebasUnidadSimulador.pruebasUnidadSimulador Class Reference

Inherits object.

### Public Member Functions

- def main ()

### 6.17.1 Detailed Description

Definition at line 9 of file pruebasUnidadSimulador.py.

### 6.17.2 Member Function Documentation

#### 6.17.2.1 main()

```
def pruebasUnidadSimulador.pruebasUnidadSimulador.main ( )
```

Definition at line 11 of file pruebasUnidadSimulador.py.

The documentation for this class was generated from the following file:

- simulador/pruebasUnidadSimulador.py

## 6.18 blockchainsimulacion.pruebasUnidadSimulador Class Reference

Inherits object.

### Public Member Functions

- def main ()

### 6.18.1 Detailed Description

Definition at line 1148 of file blockchainsimulacion.py.

### 6.18.2 Member Function Documentation

#### 6.18.2.1 main()

```
def blockchainsimulacion.pruebasUnidadSimulador.main ( )
```

Definition at line 1150 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.19 pruebasUnidadTransaccion.pruebasUnidadTransaccion Class Reference

Inherits object.

### Public Member Functions

- def main ()

### Static Public Member Functions

- def jsonDefault (object)

### 6.19.1 Detailed Description

Definition at line 16 of file pruebasUnidadTransaccion.py.

### 6.19.2 Member Function Documentation

#### 6.19.2.1 jsonDefault()

```
def pruebasUnidadTransaccion.pruebasUnidadTransaccion.jsonDefault (
              object )  [static]
```

Definition at line 19 of file pruebasUnidadTransaccion.py.

#### 6.19.2.2 main()

```
def pruebasUnidadTransaccion.pruebasUnidadTransaccion.main ( )
```

Definition at line 22 of file pruebasUnidadTransaccion.py.

The documentation for this class was generated from the following file:

- cadena/pruebasUnidadTransaccion.py

## 6.20 blockchainsimulacion.pruebasUnidadTransaccion Class Reference

Inherits object.

### Public Member Functions

- def main ()

### Static Public Member Functions

- def jsonDefault (object)

### 6.20.1 Detailed Description

Definition at line 79 of file blockchainsimulacion.py.

### 6.20.2 Member Function Documentation

#### 6.20.2.1 jsonDefault()

```
def blockchainsimulacion.pruebasUnidadTransaccion.jsonDefault (
              object )  [static]
```

Definition at line 82 of file blockchainsimulacion.py.

#### 6.20.2.2 main()

```
def blockchainsimulacion.pruebasUnidadTransaccion.main ( )
```

Definition at line 85 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.21 pruebasUnidadUsuario.pruebasUnidadUsuario Class Reference

Inherits object.

### Public Member Functions

- def main ()

### Static Public Member Functions

- def jsonDefault (object)

### 6.21.1 Detailed Description

Definition at line 9 of file pruebasUnidadUsuario.py.

### 6.21.2 Member Function Documentation

#### 6.21.2.1 jsonDefault()

```
def pruebasUnidadUsuario.pruebasUnidadUsuario.jsonDefault (
              object )  [static]
```

Definition at line 12 of file pruebasUnidadUsuario.py.

#### 6.21.2.2 main()

```
def pruebasUnidadUsuario.pruebasUnidadUsuario.main ( )
```

Definition at line 15 of file pruebasUnidadUsuario.py.

The documentation for this class was generated from the following file:

- cliente/pruebasUnidadUsuario.py

## 6.22 blockchainsimulacion.pruebasUnidadUsuario Class Reference

Inherits object.

### Public Member Functions

- def main ()

### Static Public Member Functions

- def jsonDefault (object)

### 6.22.1 Detailed Description

Definition at line 203 of file blockchainsimulacion.py.

### 6.22.2 Member Function Documentation

#### 6.22.2.1 jsonDefault()

```
def blockchainsimulacion.pruebasUnidadUsuario.jsonDefault (
              object )  [static]
```

Definition at line 206 of file blockchainsimulacion.py.

#### 6.22.2.2 main()

```
def blockchainsimulacion.pruebasUnidadUsuario.main ( )
```

Definition at line 209 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.23 Simulador.Simulador Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, nombre)
- def crearUsuarios (self, cantidadUsuarios)
- def distribuirUsuariosMineros (self, maxUsuarios, maxMineros)
- def verUsuarios (self)
- def consultarHashUsuario (self, idUsuario)
- def consultarUsuario (self, hashUsuario)
- def inicializarBlockchain (self, maxUsuarios, cantidadMineros)
- def creacionMineros (self, cantidadMineros, usuariosNodo)
- def simular (self, maxTransaccionesPorBloque, totalBloquesCaena, cantidadUsuarios)
- def seleccionarUsuario (self, cantidadPosibleReceptores, diferentea)
- def verNodosMineros (self)
- def logEjecucion (self, file, mensaje)

### Private Attributes

- _nombre
- _usuarios
- _blockchain
- _pool

### 6.23.1 Detailed Description

Definition at line 18 of file Simulador.py.

### 6.23.2 Constructor & Destructor Documentation

#### 6.23.2.1 __init__()

```
def Simulador.Simulador.__init__ (
              self,
              nombre )
```

```
Creación del escenario final de simulación
:param _nombre Nombre Asignado al escenario
:param _usuarios Conjunto de usuarios en la simulación
:param _blockchain Cadena de Bloques
:param _pool Conjunto de nodos mineros de la cadena
```

Definition at line 20 of file Simulador.py.

### 6.23.3 Member Function Documentation

#### 6.23.3.1 consultarHashUsuario()

```
def Simulador.Simulador.consultarHashUsuario (
              self,
              idUsuario )
```

```
Retorna el Hash de un usuario dado su número de identificación
:param idUsuario identificador entero de cada usuario en el sistema
:return Hash de usuario que corresponde al identificador dado
```

Definition at line 75 of file Simulador.py.

#### 6.23.3.2 consultarUsuario()

```
def Simulador.Simulador.consultarUsuario (
              self,
              hashUsuario )
```

```
Consulta la información de un usuario dado su hash
:param hashUsuario LLave del diccionario
:return usuario que corresponde con la llave dada
```

Definition at line 85 of file Simulador.py.

#### 6.23.3.3 creacionMineros()

```
def Simulador.Simulador.creacionMineros (
              self,
              cantidadMineros,
              usuariosNodo )
```

```
Adiciona los nodos mineros a la blockchain. Cada nodo minero tendrá asociado
un grupo de usuarios, quienés recibirán recompensa por su esfuerzo de minado.
:param cantidadMineros, Total de nodos Mineros para la cadena
:param usuariosNodo, Distribución de usuarios mineros para cada nodo Minero
```

Definition at line 107 of file Simulador.py.

#### 6.23.3.4 crearUsuarios()

```
def Simulador.Simulador.crearUsuarios (
              self,
              cantidadUsuarios )
```

```
Inicializa el conjunto de usuarios de la cadena de bloques
:param cantidadUsuarios: Conjunto inicial de Usuarios en la Simulación
```

Definition at line 33 of file Simulador.py.

#### 6.23.3.5 distribuirUsuariosMineros()

```
def Simulador.Simulador.distribuirUsuariosMineros (
              self,
              maxUsuarios,
              maxMineros )
```

```
Permite distribuir de manera uniforme los usuarios a los nodos mineros
:param maxMineros: Conjunto de Nodos mineros de la blockchain
:param maxUsuarios: Total de usuarios en el sistema
:return arreglo con los usuarios distribuidos para cada nodo minero
```

Definition at line 43 of file Simulador.py.

#### 6.23.3.6 inicializarBlockchain()

```
def Simulador.Simulador.inicializarBlockchain (
              self,
              maxUsuarios,
              cantidadMineros )

Creación del bloque génesis de la cadena y asignación de nodos mineros
:param cantidadMineros, Total de nodos Mineros para la cadena
:param maxUsuarios: Total de usuarios en el sistema
```

Definition at line 93 of file Simulador.py.

#### 6.23.3.7 logEjecucion()

```
def Simulador.Simulador.logEjecucion (
              self,
              file,
              mensaje )
```

Definition at line 178 of file Simulador.py.

#### 6.23.3.8 seleccionarUsuario()

```
def Simulador.Simulador.seleccionarUsuario (
              self,
              cantidadPosibleReceptores,
              diferentea )
```

Definition at line 164 of file Simulador.py.

#### 6.23.3.9 simular()

```
def Simulador.Simulador.simular (
              self,
              maxTransaccionesPorBloque,
              totalBloquesCaena,
              cantidadUsuarios )

Realiza una simulación de evolución de la blockchain con los parámetros dados.
:param cantidadUsuarios, Total de usuarios activos en la cadena
:param cantidadMineros, Total de nodos Mineros para la cadena
:param usuariosNodo, Distribución de usuarios mineros para cada nodo Minero
```

Definition at line 122 of file Simulador.py.

#### 6.23.3.10 verNodosMineros()

```
def Simulador.Simulador.verNodosMineros (
              self )
```

```
Consulta el total de nodos mineros registrados en la cadena de bloques
:return pool de mineros de la blockchain
```

Definition at line 171 of file Simulador.py.

#### 6.23.3.11 verUsuarios()

```
def Simulador.Simulador.verUsuarios (
              self )
```

```
Consulta el total de usuarios registrados en un momento dado en la cadena de bloques
:return  Representación en string del conjunto de Usuarios en la Simulación
```

Definition at line 65 of file Simulador.py.

### 6.23.4 Member Data Documentation

#### 6.23.4.1 _blockchain

```
Simulador.Simulador._blockchain  [private]
```

Definition at line 30 of file Simulador.py.

#### 6.23.4.2 _nombre

```
Simulador.Simulador._nombre  [private]
```

Definition at line 28 of file Simulador.py.

#### 6.23.4.3 _pool

`Simulador.Simulador._pool [private]`

Definition at line 31 of file Simulador.py.

#### 6.23.4.4 _usuarios

`Simulador.Simulador._usuarios [private]`

Definition at line 29 of file Simulador.py.

The documentation for this class was generated from the following file:

- simulador/Simulador.py

## 6.24 blockchainsimulacion.Simulador Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, nombre)
- def crearUsuarios (self, cantidadUsuarios)
- def distribuirUsuariosMineros (self, maxUsuarios, maxMineros)
- def verUsuarios (self)
- def consultarHashUsuario (self, idUsuario)
- def consultarUsuario (self, hashUsuario)
- def inicializarBlockchain (self, maxUsuarios, cantidadMineros)
- def creacionMineros (self, cantidadMineros, usuariosNodo)
- def simular (self, maxTransaccionesPorBloque, totalBloquesCaena, cantidadUsuarios)
- def seleccionarUsuario (self, cantidadPosibleReceptores, diferentea)
- def verNodosMineros (self)

### Private Attributes

- _nombre
- _usuarios
- _blockchain
- _pool

### 6.24.1 Detailed Description

Definition at line 976 of file blockchainsimulacion.py.

### 6.24.2 Constructor & Destructor Documentation

#### 6.24.2.1 __init__()

```
def blockchainsimulacion.Simulador.__init__ (
              self,
              nombre )
```

```
Creación del escenario final de simulación
:param _nombre Nombre Asignado al escenario
:param _usuarios Conjunto de usuarios en la simulación
:param _blockchain Cadena de Bloques
:param _pool Conjunto de nodos mineros de la cadena
```

Definition at line 978 of file blockchainsimulacion.py.

### 6.24.3 Member Function Documentation

#### 6.24.3.1 consultarHashUsuario()

```
def blockchainsimulacion.Simulador.consultarHashUsuario (
              self,
              idUsuario )
```

```
Retorna el Hash de un usuario dado su número de identificación
:param idUsuario identificador entero de cada usuario en el sistema
:return Hash de usuario que corresponde al identificador dado
```

Definition at line 1033 of file blockchainsimulacion.py.

#### 6.24.3.2 consultarUsuario()

```
def blockchainsimulacion.Simulador.consultarUsuario (
              self,
              hashUsuario )
```

```
Consulta la información de un usuario dado su hash
:param hashUsuario LLave del diccionario
:return usuario que corresponde con la llave dada
```

Definition at line 1043 of file blockchainsimulacion.py.

#### 6.24.3.3 creacionMineros()

```
def blockchainsimulacion.Simulador.creacionMineros (
              self,
              cantidadMineros,
              usuariosNodo )
```

```
Adiciona los nodos mineros a la blockchain. Cada nodo minero tendrá asociado
un grupo de usuarios, quienés recibirán recompensa por su esfuerzo de minado.
:param cantidadMineros, Total de nodos Mineros para la cadena
:param usuariosNodo, Distribución de usuarios mineros para cada nodo Minero
```

Definition at line 1065 of file blockchainsimulacion.py.

#### 6.24.3.4 crearUsuarios()

```
def blockchainsimulacion.Simulador.crearUsuarios (
              self,
              cantidadUsuarios )
```

```
Inicializa el conjunto de usuarios de la cadena de bloques
:param cantidadUsuarios: Conjunto inicial de Usuarios en la Simulación
```

Definition at line 991 of file blockchainsimulacion.py.

#### 6.24.3.5 distribuirUsuariosMineros()

```
def blockchainsimulacion.Simulador.distribuirUsuariosMineros (
              self,
              maxUsuarios,
              maxMineros )
```

```
Permite distribuir de manera uniforme los usuarios a los nodos mineros
:param maxMineros: Conjunto de Nodos mineros de la blockchain
:param maxUsuarios: Total de usuarios en el sistema
:return arreglo con los usuarios distribuidos para cada nodo minero
```

Definition at line 1001 of file blockchainsimulacion.py.

#### 6.24.3.6 inicializarBlockchain()

```
def blockchainsimulacion.Simulador.inicializarBlockchain (
              self,
              maxUsuarios,
              cantidadMineros )
```

```
Creación del bloque génesis de la cadena y asignación de nodos mineros
:param cantidadMineros, Total de nodos Mineros para la cadena
:param maxUsuarios: Total de usuarios en el sistema
```

Definition at line 1051 of file blockchainsimulacion.py.

#### 6.24.3.7 seleccionarUsuario()

```
def blockchainsimulacion.Simulador.seleccionarUsuario (
              self,
              cantidadPosibleReceptores,
              diferentea )
```

Definition at line 1118 of file blockchainsimulacion.py.

#### 6.24.3.8 simular()

```
def blockchainsimulacion.Simulador.simular (
              self,
              maxTransaccionesPorBloque,
              totalBloquesCaena,
              cantidadUsuarios )
```

```
Realiza una simulación de evolución de la blockchain con los parámetros dados.
:param cantidadUsuarios, Total de usuarios activos en la cadena
:param cantidadMineros, Total de nodos Mineros para la cadena
:param usuariosNodo, Distribución de usuarios mineros para cada nodo Minero
```

Definition at line 1080 of file blockchainsimulacion.py.

#### 6.24.3.9 verNodosMineros()

```
def blockchainsimulacion.Simulador.verNodosMineros (
              self )
```

```
Consulta el total de nodos mineros registrados en la cadena de bloques
:return pool de mineros de la blockchain
```

Definition at line 1125 of file blockchainsimulacion.py.

#### 6.24.3.10 verUsuarios()

```
def blockchainsimulacion.Simulador.verUsuarios (
              self )
```

```
Consulta el total de usuarios registrados en un momento dado en la cadena de bloques
:return  Representación en string del conjunto de Usuarios en la Simulación
```

Definition at line 1023 of file blockchainsimulacion.py.

### 6.24.4 Member Data Documentation

#### 6.24.4.1 _blockchain

```
blockchainsimulacion.Simulador._blockchain  [private]
```

Definition at line 988 of file blockchainsimulacion.py.

#### 6.24.4.2 _nombre

```
blockchainsimulacion.Simulador._nombre  [private]
```

Definition at line 986 of file blockchainsimulacion.py.

#### 6.24.4.3 _pool

```
blockchainsimulacion.Simulador._pool  [private]
```

Definition at line 989 of file blockchainsimulacion.py.

#### 6.24.4.4 _usuarios

```
blockchainsimulacion.Simulador._usuarios  [private]
```

Definition at line 987 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.25 blockchainsimulacion.Transaccion Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, emisor, receptor, cantidad)
- def getEmisor (self)
- def getReceptor (self)
- def getCantidad (self)
- def __str__ (self)

### Private Attributes

- _emisor
- _receptor
- _cantidad

## 6.25.1 Detailed Description

Definition at line 15 of file blockchainsimulacion.py.

## 6.25.2 Constructor & Destructor Documentation

### 6.25.2.1 __init__()

```
def blockchainsimulacion.Transaccion.__init__ (
              self,
              emisor,
              receptor,
              cantidad )
```

```
Creación de una transaccióna
:param emisor: Hash de quién envía
:param receptor:  Hash de quién recibe
:param cantidad: valor de la transacción
```

Definition at line 17 of file blockchainsimulacion.py.

## 6.25.3 Member Function Documentation

### 6.25.3.1 __str__()

```
def blockchainsimulacion.Transaccion.__str__ (
              self )
```

```
Genera un string con la información de la transacción
:return: representación en String de una transacción
```

Definition at line 49 of file blockchainsimulacion.py.

#### 6.25.3.2 getCantidad()

```
def blockchainsimulacion.Transaccion.getCantidad (
              self )
```

```
Obtiene el valor de la transacción
:return: Valor de la transacción
```

Definition at line 42 of file blockchainsimulacion.py.

#### 6.25.3.3 getEmisor()

```
def blockchainsimulacion.Transaccion.getEmisor (
              self )
```

```
Obtiene el valor actual del hash del emisor
:return: Hash del emisor
```

Definition at line 28 of file blockchainsimulacion.py.

#### 6.25.3.4 getReceptor()

```
def blockchainsimulacion.Transaccion.getReceptor (
              self )
```

```
Obtiene el valor actual del hash del receptor
:return: Hash del receptor
```

Definition at line 35 of file blockchainsimulacion.py.

### 6.25.4 Member Data Documentation

#### 6.25.4.1 _cantidad

```
blockchainsimulacion.Transaccion._cantidad  [private]
```

Definition at line 26 of file blockchainsimulacion.py.

#### 6.25.4.2 _emisor

`blockchainsimulacion.Transaccion._emisor [private]`

Definition at line 24 of file blockchainsimulacion.py.

#### 6.25.4.3 _receptor

`blockchainsimulacion.Transaccion._receptor [private]`

Definition at line 25 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

## 6.26 Transaccion.Transaccion Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, emisor, receptor, cantidad)
- def getEmisor (self)
- def getReceptor (self)
- def getCantidad (self)
- def __str__ (self)

### Private Attributes

- _emisor
- _receptor
- _cantidad

### 6.26.1 Detailed Description

Definition at line 7 of file Transaccion.py.

### 6.26.2 Constructor & Destructor Documentation

#### 6.26.2.1 __init__()

```
def Transaccion.Transaccion.__init__ (
              self,
              emisor,
              receptor,
              cantidad )


Creación de una transaccióna
:param emisor: Hash de quién envía
:param receptor:  Hash de quién recibe
:param cantidad: valor de la transacción
```

Definition at line 9 of file Transaccion.py.

### 6.26.3 Member Function Documentation

#### 6.26.3.1 __str__()

```
def Transaccion.Transaccion.__str__ (
              self )


Genera un string con la información de la transacción
:return: representación en String de una transacción
```

Definition at line 41 of file Transaccion.py.

#### 6.26.3.2 getCantidad()

```
def Transaccion.Transaccion.getCantidad (
              self )


Obtiene el valor de la transacción
:return: Valor de la transacción
```

Definition at line 34 of file Transaccion.py.

#### 6.26.3.3 getEmisor()

```
def Transaccion.Transaccion.getEmisor (
              self )

Obtiene el valor actual del hash del emisor
:return: Hash del emisor
```

Definition at line 20 of file Transaccion.py.

#### 6.26.3.4 getReceptor()

```
def Transaccion.Transaccion.getReceptor (
              self )

Obtiene el valor actual del hash del receptor
:return: Hash del receptor
```

Definition at line 27 of file Transaccion.py.

### 6.26.4 Member Data Documentation

#### 6.26.4.1 _cantidad

```
Transaccion.Transaccion._cantidad  [private]
```

Definition at line 18 of file Transaccion.py.

#### 6.26.4.2 _emisor

```
Transaccion.Transaccion._emisor  [private]
```

Definition at line 16 of file Transaccion.py.

#### 6.26.4.3 _receptor

```
Transaccion.Transaccion._receptor  [private]
```

Definition at line 17 of file Transaccion.py.

The documentation for this class was generated from the following file:

- cadena/Transaccion.py

## 6.27 Usuario.Usuario Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, idUsuario)
- def gethashUsuario (self)
- def setHashUsuario (self)
- def generarUsuarioHash (self)
- def __str__ (self)
- def enviar (self, valor, receptor)
- def recibir (self, valor, receptor)

### Private Attributes

- _idUsuario
- _saldo
- _hashUsuario
- _marcaTiempo

### 6.27.1 Detailed Description

Definition at line 11 of file Usuario.py.

### 6.27.2 Constructor & Destructor Documentation

#### 6.27.2.1 __init__()

```
def Usuario.Usuario.__init__ (
              self,
              idUsuario )
```

```
Creación de un usuario en el sistema
:param _idUsuario Identificador entero del usuario
:param hashUsuario:  Código Hash del usuario
:param saldo: Saldo del usuario
:param marcaTiempo: Instante de tiempo de creación del usuario
```

Definition at line 13 of file Usuario.py.

### 6.27.3 Member Function Documentation

#### 6.27.3.1 __str__()

```
def Usuario.Usuario.__str__ (
              self )
```

```
Genera un string con la información del usuario
:return: representación en String del usuario
```

Definition at line 51 of file Usuario.py.

#### 6.27.3.2 enviar()

```
def Usuario.Usuario.enviar (
              self,
              valor,
              receptor )
```

```
:param receptor: hash del usuario receptor de la transferencia
:param valor: Cantidad que se desea transferir
:return: La nueva transacción que será enviada a validación
```

Definition at line 58 of file Usuario.py.

#### 6.27.3.3 generarUsuarioHash()

```
def Usuario.Usuario.generarUsuarioHash (
              self )
```

```
Genera Hash la información del bloque
:return: Hash construido a partir de la información del usuario
```

Definition at line 43 of file Usuario.py.

#### 6.27.3.4 gethashUsuario()

```
def Usuario.Usuario.gethashUsuario (
              self )
```

```
Obtiene el valor actual del hash del usuario
:return: Hash del bloque
```

Definition at line 32 of file Usuario.py.

#### 6.27.3.5 recibir()

```
def Usuario.Usuario.recibir (
              self,
              valor,
              receptor )
```

```
:param receptor: hash del usuario receptor de la transferencia
:param valor: Cantidad que se ha recibido
```

Definition at line 69 of file Usuario.py.

#### 6.27.3.6 setHashUsuario()

```
def Usuario.Usuario.setHashUsuario (
              self )
```

Definition at line 39 of file Usuario.py.

### 6.27.4 Member Data Documentation

#### 6.27.4.1 _hashUsuario

```
Usuario.Usuario._hashUsuario  [private]
```

Definition at line 23 of file Usuario.py.

#### 6.27.4.2 _idUsuario

```
Usuario.Usuario._idUsuario  [private]
```

Definition at line 21 of file Usuario.py.

#### 6.27.4.3 _marcaTiempo

```
Usuario.Usuario._marcaTiempo  [private]
```

Definition at line 24 of file Usuario.py.

#### 6.27.4.4 _saldo

```
Usuario.Usuario._saldo  [private]
```

Definition at line 22 of file Usuario.py.

The documentation for this class was generated from the following file:

- cliente/Usuario.py

## 6.28 blockchainsimulacion.Usuario Class Reference

Inherits object.

### Public Member Functions

- def __init__ (self, idUsuario)
- def gethashUsuario (self)
- def setHashUsuario (self)
- def generarUsuarioHash (self)
- def __str__ (self)
- def enviar (self, valor, receptor)
- def recibir (self, valor, receptor)

### Private Attributes

- _idUsuario
- _saldo
- _hashUsuario
- _marcaTiempo

## 6.28.1 Detailed Description

Definition at line 122 of file blockchainsimulacion.py.

## 6.28.2 Constructor & Destructor Documentation

#### 6.28.2.1 __init__()

```
def blockchainsimulacion.Usuario.__init__ (
              self,
              idUsuario )
```

```
Creación de un usuario en el sistema
:param _idUsuario Identificador entero del usuario
:param hashUsuario:  Código Hash del usuario
:param saldo: Saldo del usuario
:param marcaTiempo: Instante de tiempo de creación del usuario
```

Definition at line 124 of file blockchainsimulacion.py.

## 6.28.3 Member Function Documentation

#### 6.28.3.1 __str__()

```
def blockchainsimulacion.Usuario.__str__ (
              self )
```

```
Genera un string con la información del usuario
:return: representación en String del usuario
```

Definition at line 162 of file blockchainsimulacion.py.

#### 6.28.3.2 enviar()

```
def blockchainsimulacion.Usuario.enviar (
              self,
              valor,
              receptor )
```

```
:param receptor: hash del usuario receptor de la transferencia
:param valor: Cantidad que se desea transferir
:return: La nueva transacción que será enviada a validación
```

Definition at line 169 of file blockchainsimulacion.py.

#### 6.28.3.3 generarUsuarioHash()

```
def blockchainsimulacion.Usuario.generarUsuarioHash (
              self )
```

```
Genera Hash la información del bloque
:return: Hash construido a partir de la información del usuario
```

Definition at line 154 of file blockchainsimulacion.py.

#### 6.28.3.4 gethashUsuario()

```
def blockchainsimulacion.Usuario.gethashUsuario (
              self )
```

```
Obtiene el valor actual del hash del usuario
:return: Hash del bloque
```

Definition at line 143 of file blockchainsimulacion.py.

#### 6.28.3.5 recibir()

```
def blockchainsimulacion.Usuario.recibir (
              self,
              valor,
              receptor )
```

```
:param receptor: hash del usuario receptor de la transferencia
:param valor: Cantidad que se ha recibido
```

Definition at line 180 of file blockchainsimulacion.py.

#### 6.28.3.6 setHashUsuario()

```
def blockchainsimulacion.Usuario.setHashUsuario (
              self )
```

Definition at line 150 of file blockchainsimulacion.py.

### 6.28.4 Member Data Documentation

#### 6.28.4.1 _hashUsuario

```
blockchainsimulacion.Usuario._hashUsuario  [private]
```

Definition at line 134 of file blockchainsimulacion.py.

#### 6.28.4.2 _idUsuario

```
blockchainsimulacion.Usuario._idUsuario  [private]
```

Definition at line 132 of file blockchainsimulacion.py.

#### 6.28.4.3 _marcaTiempo

```
blockchainsimulacion.Usuario._marcaTiempo  [private]
```

Definition at line 135 of file blockchainsimulacion.py.

#### 6.28.4.4 _saldo

```
blockchainsimulacion.Usuario._saldo  [private]
```

Definition at line 133 of file blockchainsimulacion.py.

The documentation for this class was generated from the following file:

- docs/blockchainsimulacion.py

# Chapter 7

# File Documentation

## 7.1 __init__.py File Reference

### Namespaces

- blockChainTest2019

## 7.2 cadena/Bloque.py File Reference

### Classes

- class Bloque.Bloque

### Namespaces

- Bloque

## 7.3 cadena/CadenaBloques.py File Reference

### Classes

- class CadenaBloques.CadenaBloques

### Namespaces

- CadenaBloques

## 7.4 cadena/NodoMinador.py File Reference

### Classes

- class NodoMinador.NodoMinador

### Namespaces

- NodoMinador

## 7.5 cadena/PoolMinado.py File Reference

### Classes

- class PoolMinado.PoolMinado

### Namespaces

- PoolMinado

## 7.6 cadena/pruebasUnidad.py File Reference

### Namespaces

- pruebasUnidad

### Functions

- def pruebasUnidad.main ()

## 7.7 cadena/pruebasUnidadBloque.py File Reference

### Classes

- class pruebasUnidadBloque.pruebasUnidadBloque

### Namespaces

- pruebasUnidadBloque

## 7.8 cadena/pruebasUnidadCadena.py File Reference

### Classes

- class pruebasUnidadCadena.pruebasUnidadCadena

### Namespaces

- pruebasUnidadCadena

## 7.9 cadena/pruebasUnidadNodoMineria.py File Reference

### Classes

- class pruebasUnidadNodoMineria.pruebasUnidadNodoMineria

### Namespaces

- pruebasUnidadNodoMineria

## 7.10 cadena/pruebasUnidadPoolMinado.py File Reference

### Classes

- class pruebasUnidadPoolMinado.pruebasUnidadPoolMinado

### Namespaces

- pruebasUnidadPoolMinado

## 7.11 cadena/pruebasUnidadTransaccion.py File Reference

### Classes

- class pruebasUnidadTransaccion.pruebasUnidadTransaccion

### Namespaces

- pruebasUnidadTransaccion

## 7.12 cadena/Transaccion.py File Reference

### Classes

- class Transaccion.Transaccion

### Namespaces

- Transaccion

## 7.13 cliente/pruebasUnidadUsuario.py File Reference

### Classes

- class pruebasUnidadUsuario.pruebasUnidadUsuario

### Namespaces

- pruebasUnidadUsuario

## 7.14 cliente/Usuario.py File Reference

### Classes

- class Usuario.Usuario

### Namespaces

- Usuario

## 7.15 docs/blockchainsimulacion.py File Reference

### Classes

- class blockchainsimulacion.Transaccion
- class blockchainsimulacion.pruebasUnidadTransaccion
- class blockchainsimulacion.Usuario
- class blockchainsimulacion.pruebasUnidadUsuario
- class blockchainsimulacion.Bloque
- class blockchainsimulacion.pruebasUnidadBloque
- class blockchainsimulacion.NodoMinador
- class blockchainsimulacion.pruebasUnidadNodoMineria
- class blockchainsimulacion.CadenaBloques
- class blockchainsimulacion.pruebasUnidadCadena
- class blockchainsimulacion.PoolMinado
- class blockchainsimulacion.pruebasUnidadPoolMinado
- class blockchainsimulacion.Simulador
- class blockchainsimulacion.pruebasUnidadSimulador

### Namespaces

- blockchainsimulacion

### Variables

- string blockchainsimulacion.__author__ = "Roberto A. Pava"
- string blockchainsimulacion.__maintainer__ = "Roberto A. Pava"
- string blockchainsimulacion.__email__ = "rapavad@correo.udistrital.edu.co"
- string blockchainsimulacion.__copyright__ = "Copyright 2018"
- string blockchainsimulacion.__version__ = "1.0"

## 7.16 docs/conf.py File Reference

### Namespaces

- conf

### Variables

- list conf.extensions
- list conf.templates_path = ['_templates']
- string conf.source_suffix = '.rst'
- string conf.master_doc = 'index'
- string conf.project = 'MyProject'
- string conf.copyright = '2019, My Name'
- string conf.author = 'My Name'
- string conf.version = '0.0.1'
- string conf.release = '0.0.1'
- string conf.language = 'en'
- list conf.exclude_patterns = ['_build', 'Thumbs.db', '.DS_Store']
- string conf.pygments_style = 'sphinx'
- bool conf.todo_include_todos = True
- string conf.html_theme = 'alabaster'
- list conf.html_static_path = ['_static']
- dictionary conf.html_sidebars
- string conf.htmlhelp_basename = 'MyProjectdoc'
- dictionary conf.latex_elements
- list conf.latex_documents
- list conf.man_pages
- list conf.texinfo_documents
- string conf.epub_title = project
- string conf.epub_author = author
- string conf.epub_publisher = author
- string conf.epub_copyright = copyright
- list conf.epub_exclude_files = ['search.html']

## 7.17 docs/source/conf.py File Reference

### Namespaces

- conf

### Variables

- dictionary conf.intersphinx_mapping = {'https://docs.python.org/': None}

## 7.18 simulador/pruebasUnidadSimulador.py File Reference

### Classes

- class pruebasUnidadSimulador.pruebasUnidadSimulador

### Namespaces

- pruebasUnidadSimulador

## 7.19 simulador/Simulador.py File Reference

### Classes

- class Simulador.Simulador

### Namespaces

- Simulador

# Index


__author__
    blockchainsimulacion, 9
__copyright__
    blockchainsimulacion, 10
__email__
    blockchainsimulacion, 10
__init__
    blockchainsimulacion::Bloque, 25
    blockchainsimulacion::CadenaBloques, 32
    blockchainsimulacion::NodoMinador, 35
    blockchainsimulacion::PoolMinado, 47
    blockchainsimulacion::Simulador, 68
    blockchainsimulacion::Transaccion, 73
    blockchainsimulacion::Usuario, 82
    Bloque::Bloque, 22
    CadenaBloques::CadenaBloques, 28
    NodoMinador::NodoMinador, 39
    PoolMinado::PoolMinado, 43
    Simulador::Simulador, 62
    Transaccion::Transaccion, 75
    Usuario::Usuario, 78
__init__.py, 85
__maintainer__
    blockchainsimulacion, 10
__repr__
    blockchainsimulacion::Bloque, 26
    blockchainsimulacion::CadenaBloques, 32
    Bloque::Bloque, 22
    CadenaBloques::CadenaBloques, 29
__str__
    blockchainsimulacion::Bloque, 26
    blockchainsimulacion::NodoMinador, 36
    blockchainsimulacion::PoolMinado, 48
    blockchainsimulacion::Transaccion, 73
    blockchainsimulacion::Usuario, 82
    Bloque::Bloque, 22
    CadenaBloques::CadenaBloques, 29
    NodoMinador::NodoMinador, 40
    PoolMinado::PoolMinado, 43
    Transaccion::Transaccion, 76
    Usuario::Usuario, 79
__version__
    blockchainsimulacion, 10
_blockchain
    blockchainsimulacion::Simulador, 71
    Simulador::Simulador, 66
_cadena
    blockchainsimulacion::CadenaBloques, 34
    CadenaBloques::CadenaBloques, 31
_cadenaBloques
    blockchainsimulacion::NodoMinador, 37
    NodoMinador::NodoMinador, 40
_cantidad
    blockchainsimulacion::Transaccion, 74
    Transaccion::Transaccion, 77
_cantidadBloquesMinados
    blockchainsimulacion::NodoMinador, 37
    NodoMinador::NodoMinador, 41
_descripcion
    blockchainsimulacion::NodoMinador, 37
    NodoMinador::NodoMinador, 41
_dificultad
    blockchainsimulacion::PoolMinado, 50
    PoolMinado::PoolMinado, 46
_direccion
    blockchainsimulacion::NodoMinador, 37
    NodoMinador::NodoMinador, 41
_emisor
    blockchainsimulacion::Transaccion, 74
    Transaccion::Transaccion, 77
_fechaCreacion
    blockchainsimulacion::NodoMinador, 37
    NodoMinador::NodoMinador, 41
_hashBloque
    blockchainsimulacion::Bloque, 27
    Bloque::Bloque, 23
_hashRate
    blockchainsimulacion::PoolMinado, 51
    PoolMinado::PoolMinado, 46
_hashRateNodo
    blockchainsimulacion::NodoMinador, 38
    NodoMinador::NodoMinador, 41
_hashUsuario
    blockchainsimulacion::Usuario, 84
    Usuario::Usuario, 81
_idUsuario
    blockchainsimulacion::Usuario, 84
    Usuario::Usuario, 81
_indice
    blockchainsimulacion::Bloque, 27

Bloque::Bloque, 24
_marcaTiempo
blockchainsimulacion::Bloque, 27
blockchainsimulacion::Usuario, 84
Bloque::Bloque, 24
Usuario::Usuario, 81
_minado
blockchainsimulacion::Bloque, 27
Bloque::Bloque, 24
_nodos
blockchainsimulacion::PoolMinado, 51
PoolMinado::PoolMinado, 46
_nombre
blockchainsimulacion::Simulador, 72
Simulador::Simulador, 66
_pool
blockchainsimulacion::Simulador, 72
Simulador::Simulador, 66
_puerto
blockchainsimulacion::NodoMinador, 38
NodoMinador::NodoMinador, 41
_receptor
blockchainsimulacion::Transaccion, 75
Transaccion::Transaccion, 77
_recompensaPorBloque
blockchainsimulacion::PoolMinado, 51
PoolMinado::PoolMinado, 46
_saldo
blockchainsimulacion::Usuario, 84
Usuario::Usuario, 81
_transacciones
blockchainsimulacion::Bloque, 27
Bloque::Bloque, 24
_usuarios
blockchainsimulacion::Simulador, 72
Simulador::Simulador, 67
_usuariosMineros
blockchainsimulacion::NodoMinador, 38
NodoMinador::NodoMinador, 42
_utilidad
blockchainsimulacion::NodoMinador, 38
NodoMinador::NodoMinador, 42

actualizarCadena
blockchainsimulacion::NodoMinador, 36
NodoMinador::NodoMinador, 40
adicionarMinero
blockchainsimulacion::PoolMinado, 48
PoolMinado::PoolMinado, 43
author
conf, 11

blockChainTest2019, 10
blockchainsimulacion, 9
__author__, 9
__copyright__, 10
__email__, 10
__maintainer__, 10
__version__, 10
blockchainsimulacion.Bloque, 24
blockchainsimulacion.CadenaBloques, 31
blockchainsimulacion.NodoMinador, 35
blockchainsimulacion.PoolMinado, 47
blockchainsimulacion.pruebasUnidadBloque, 51
blockchainsimulacion.pruebasUnidadCadena, 53
blockchainsimulacion.pruebasUnidadNodoMineria, 54
blockchainsimulacion.pruebasUnidadPoolMinado, 56
blockchainsimulacion.pruebasUnidadSimulador, 57
blockchainsimulacion.pruebasUnidadTransaccion, 59
blockchainsimulacion.pruebasUnidadUsuario, 61
blockchainsimulacion.Simulador, 67
blockchainsimulacion.Transaccion, 72
blockchainsimulacion.Usuario, 81
blockchainsimulacion::Bloque
__init__, 25
__repr__, 26
__str__, 26
_hashBloque, 27
_indice, 27
_marcaTiempo, 27
_minado, 27
_transacciones, 27
generarBloqueHash, 26
gethashBloque, 26
setHashBloque, 26
blockchainsimulacion::CadenaBloques
__init__, 32
__repr__, 32
_cadena, 34
bloqueEsValido, 33
bloqueGenesis, 33
getCadenaSerializada, 33
nuevoBloque, 33
obtenerUltimoBloque, 34
validarBlockchain, 34
blockchainsimulacion::NodoMinador
__init__, 35
__str__, 36
_cadenaBloques, 37
_cantidadBloquesMinados, 37
_descripcion, 37
_direccion, 37
_fechaCreacion, 37
_hashRateNodo, 38
_puerto, 38
_usuariosMineros, 38
_utilidad, 38
actualizarCadena, 36
infoCadena, 36

blockchainsimulacion::PoolMinado
__init__, 47
__str__, 48
_dificultad, 50
_hashRate, 51
_nodos, 51
_recompensaPorBloque, 51
adicionarMinero, 48
esPrimo, 48
minadoBloque, 49
minarBloque, 49
pruebaDeTrabajo, 49
sincronizarCadena, 50
torneoPorNuevoBloque, 50
blockchainsimulacion::Simulador
__init__, 68
_blockchain, 71
_nombre, 72
_pool, 72
_usuarios, 72
consultarHashUsuario, 68
consultarUsuario, 68
creacionMineros, 69
crearUsuarios, 69
distribuirUsuariosMineros, 69
inicializarBlockchain, 70
seleccionarUsuario, 70
simular, 70
verNodosMineros, 71
verUsuarios, 71
blockchainsimulacion::Transaccion
__init__, 73
__str__, 73
_cantidad, 74
_emisor, 74
_receptor, 75
getCantidad, 73
getEmisor, 74
getReceptor, 74
blockchainsimulacion::Usuario
__init__, 82
__str__, 82
_hashUsuario, 84
_idUsuario, 84
_marcaTiempo, 84
_saldo, 84
enviar, 83
generarUsuarioHash, 83
gethashUsuario, 83
recibir, 83
setHashUsuario, 84
blockchainsimulacion::pruebasUnidadBloque
main, 52
blockchainsimulacion::pruebasUnidadCadena
main, 53
blockchainsimulacion::pruebasUnidadNodoMineria
main, 54
blockchainsimulacion::pruebasUnidadPoolMinado
main, 56
blockchainsimulacion::pruebasUnidadSimulador
main, 58
blockchainsimulacion::pruebasUnidadTransaccion
jsonDefault, 59
main, 59
blockchainsimulacion::pruebasUnidadUsuario
jsonDefault, 61
main, 61
Bloque, 10
Bloque.Bloque, 21
Bloque::Bloque
__init__, 22
__repr__, 22
__str__, 22
_hashBloque, 23
_indice, 24
_marcaTiempo, 24
_minado, 24
_transacciones, 24
consultarBloque, 22
generarBloqueHash, 23
gethashBloque, 23
setHashBloque, 23
bloqueEsValido
blockchainsimulacion::CadenaBloques, 33
CadenaBloques::CadenaBloques, 29
bloqueGenesis
blockchainsimulacion::CadenaBloques, 33
CadenaBloques::CadenaBloques, 29

cadena/Bloque.py, 85
cadena/CadenaBloques.py, 85
cadena/NodoMinador.py, 86
cadena/PoolMinado.py, 86
cadena/Transaccion.py, 88
cadena/pruebasUnidad.py, 86
cadena/pruebasUnidadBloque.py, 86
cadena/pruebasUnidadCadena.py, 87
cadena/pruebasUnidadNodoMineria.py, 87
cadena/pruebasUnidadPoolMinado.py, 87
cadena/pruebasUnidadTransaccion.py, 87
CadenaBloques, 11
CadenaBloques.CadenaBloques, 28
CadenaBloques::CadenaBloques
__init__, 28
__repr__, 29
__str__, 29
_cadena, 31
bloqueEsValido, 29

bloqueGenesis, 29
getCadenaSerializada, 30
nuevoBloque, 30
obtenerUltimoBloque, 30
validarBlockchain, 31
cliente/Usuario.py, 88
cliente/pruebasUnidadUsuario.py, 88
conf, 11
author, 11
copyright, 12
epub_author, 12
epub_copyright, 12
epub_exclude_files, 12
epub_publisher, 12
epub_title, 12
exclude_patterns, 13
extensions, 13
html_sidebars, 13
html_static_path, 13
html_theme, 14
htmlhelp_basename, 14
intersphinx_mapping, 14
language, 14
latex_documents, 14
latex_elements, 15
man_pages, 15
master_doc, 15
project, 16
pygments_style, 16
release, 16
source_suffix, 16
templates_path, 16
texinfo_documents, 16
todo_include_todos, 17
version, 17
consultarBloque
Bloque::Bloque, 22
consultarHashUsuario
blockchainsimulacion::Simulador, 68
Simulador::Simulador, 63
consultarUsuario
blockchainsimulacion::Simulador, 68
Simulador::Simulador, 63
copyright
conf, 12
creacionMineros
blockchainsimulacion::Simulador, 69
Simulador::Simulador, 63
crearUsuarios
blockchainsimulacion::Simulador, 69
Simulador::Simulador, 64

distribuirUsuariosMineros
blockchainsimulacion::Simulador, 69
Simulador::Simulador, 64
docs/blockchainsimulacion.py, 88
docs/conf.py, 89
docs/source/conf.py, 90

enviar
blockchainsimulacion::Usuario, 83
Usuario::Usuario, 79
epub_author
conf, 12
epub_copyright
conf, 12
epub_exclude_files
conf, 12
epub_publisher
conf, 12
epub_title
conf, 12
esPrimo
blockchainsimulacion::PoolMinado, 48
PoolMinado::PoolMinado, 44
exclude_patterns
conf, 13
extensions
conf, 13

generarBloqueHash
blockchainsimulacion::Bloque, 26
Bloque::Bloque, 23
generarUsuarioHash
blockchainsimulacion::Usuario, 83
Usuario::Usuario, 79
getCadenaSerializada
blockchainsimulacion::CadenaBloques, 33
CadenaBloques::CadenaBloques, 30
getCantidad
blockchainsimulacion::Transaccion, 73
Transaccion::Transaccion, 76
getEmisor
blockchainsimulacion::Transaccion, 74
Transaccion::Transaccion, 76
getReceptor
blockchainsimulacion::Transaccion, 74
Transaccion::Transaccion, 77
gethashBloque
blockchainsimulacion::Bloque, 26
Bloque::Bloque, 23
gethashUsuario
blockchainsimulacion::Usuario, 83
Usuario::Usuario, 80

html_sidebars
conf, 13
html_static_path
conf, 13

html_theme
conf, 14
htmlhelp_basename
conf, 14

infoCadena
blockchainsimulacion::NodoMinador, 36
NodoMinador::NodoMinador, 40
inicializarBlockchain
blockchainsimulacion::Simulador, 70
Simulador::Simulador, 64
intersphinx_mapping
conf, 14

jsonDefault
blockchainsimulacion::pruebasUnidadTransaccion, 59
blockchainsimulacion::pruebasUnidadUsuario, 61
pruebasUnidadTransaccion::pruebasUnidadTransaccion, 58
pruebasUnidadUsuario::pruebasUnidadUsuario, 60

language
conf, 14
latex_documents
conf, 14
latex_elements
conf, 15
logEjecucion
Simulador::Simulador, 65

main
blockchainsimulacion::pruebasUnidadBloque, 52
blockchainsimulacion::pruebasUnidadCadena, 53
blockchainsimulacion::pruebasUnidadNodoMineria, 54
blockchainsimulacion::pruebasUnidadPoolMinado, 56
blockchainsimulacion::pruebasUnidadSimulador, 58
blockchainsimulacion::pruebasUnidadTransaccion, 59
blockchainsimulacion::pruebasUnidadUsuario, 61
pruebasUnidad, 18
pruebasUnidadBloque::pruebasUnidadBloque, 52
pruebasUnidadCadena::pruebasUnidadCadena, 54
pruebasUnidadNodoMineria::pruebasUnidadNodoMineria, 55
pruebasUnidadPoolMinado::pruebasUnidadPoolMinado, 56
pruebasUnidadSimulador::pruebasUnidadSimulador, 57
pruebasUnidadTransaccion::pruebasUnidadTransaccion, 58
pruebasUnidadUsuario::pruebasUnidadUsuario, 60
man_pages
conf, 15
master_doc
conf, 15
minadoBloque
blockchainsimulacion::PoolMinado, 49
PoolMinado::PoolMinado, 44
minarBloque
blockchainsimulacion::PoolMinado, 49
PoolMinado::PoolMinado, 44

NodoMinador, 17
NodoMinador.NodoMinador, 38
NodoMinador::NodoMinador
__init__, 39
__str__, 40
_cadenaBloques, 40
_cantidadBloquesMinados, 41
_descripcion, 41
_direccion, 41
_fechaCreacion, 41
_hashRateNodo, 41
_puerto, 41
_usuariosMineros, 42
_utilidad, 42
actualizarCadena, 40
infoCadena, 40
nuevoBloque
blockchainsimulacion::CadenaBloques, 33
CadenaBloques::CadenaBloques, 30

obtenerUltimoBloque
blockchainsimulacion::CadenaBloques, 34
CadenaBloques::CadenaBloques, 30

PoolMinado, 17
PoolMinado.PoolMinado, 42
PoolMinado::PoolMinado
__init__, 43
__str__, 43
_dificultad, 46
_hashRate, 46
_nodos, 46
_recompensaPorBloque, 46
adicionarMinero, 43
esPrimo, 44
minadoBloque, 44
minarBloque, 44
pruebaDeTrabajo, 45
sincronizarCadena, 45
torneoPorNuevoBloque, 45
project
conf, 16
pruebaDeTrabajo
blockchainsimulacion::PoolMinado, 49
PoolMinado::PoolMinado, 45

pruebasUnidad, 18
    main, 18
pruebasUnidadBloque, 18
pruebasUnidadBloque.pruebasUnidadBloque, 52
pruebasUnidadBloque::pruebasUnidadBloque
    main, 52
pruebasUnidadCadena, 18
pruebasUnidadCadena.pruebasUnidadCadena, 53
pruebasUnidadCadena::pruebasUnidadCadena
    main, 54
pruebasUnidadNodoMineria, 18
pruebasUnidadNodoMineria.pruebasUnidadNodoMineria, 55
pruebasUnidadNodoMineria::pruebasUnidadNodoMineria
    main, 55
pruebasUnidadPoolMinado, 18
pruebasUnidadPoolMinado.pruebasUnidadPoolMinado, 55
pruebasUnidadPoolMinado::pruebasUnidadPoolMinado
    main, 56
pruebasUnidadSimulador, 19
pruebasUnidadSimulador.pruebasUnidadSimulador, 57
pruebasUnidadSimulador::pruebasUnidadSimulador
    main, 57
pruebasUnidadTransaccion, 19
pruebasUnidadTransaccion.pruebasUnidadTransaccion, 58
pruebasUnidadTransaccion::pruebasUnidadTransaccion
    jsonDefault, 58
    main, 58
pruebasUnidadUsuario, 19
pruebasUnidadUsuario.pruebasUnidadUsuario, 60
pruebasUnidadUsuario::pruebasUnidadUsuario
    jsonDefault, 60
    main, 60
pygments_style
    conf, 16

recibir
    blockchainsimulacion::Usuario, 83
    Usuario::Usuario, 80
release
    conf, 16

seleccionarUsuario
    blockchainsimulacion::Simulador, 70
    Simulador::Simulador, 65
setHashBloque
    blockchainsimulacion::Bloque, 26
    Bloque::Bloque, 23
setHashUsuario
    blockchainsimulacion::Usuario, 84
    Usuario::Usuario, 80
Simulador, 19
Simulador.Simulador, 62
simulador/Simulador.py, 90
simulador/pruebasUnidadSimulador.py, 90
Simulador::Simulador
    __init__, 62
    _blockchain, 66
    _nombre, 66
    _pool, 66
    _usuarios, 67
    consultarHashUsuario, 63
    consultarUsuario, 63
    creacionMineros, 63
    crearUsuarios, 64
    distribuirUsuariosMineros, 64
    inicializarBlockchain, 64
    logEjecucion, 65
    seleccionarUsuario, 65
    simular, 65
    verNodosMineros, 65
    verUsuarios, 66
simular
    blockchainsimulacion::Simulador, 70
    Simulador::Simulador, 65
sincronizarCadena
    blockchainsimulacion::PoolMinado, 50
    PoolMinado::PoolMinado, 45
source_suffix
    conf, 16

templates_path
    conf, 16
texinfo_documents
    conf, 16
todo_include_todos
    conf, 17
torneoPorNuevoBloque
    blockchainsimulacion::PoolMinado, 50
    PoolMinado::PoolMinado, 45
Transaccion, 19
Transaccion.Transaccion, 75
Transaccion::Transaccion
    __init__, 75
    __str__, 76
    _cantidad, 77
    _emisor, 77
    _receptor, 77
    getCantidad, 76
    getEmisor, 76
    getReceptor, 77

Usuario, 19
Usuario.Usuario, 78
Usuario::Usuario
    __init__, 78
    __str__, 79
    _hashUsuario, 81

_idUsuario, 81
_marcaTiempo, 81
_saldo, 81
enviar, 79
generarUsuarioHash, 79
gethashUsuario, 80
recibir, 80
setHashUsuario, 80

validarBlockchain
blockchainsimulacion::CadenaBloques, 34
CadenaBloques::CadenaBloques, 31
verNodosMineros
blockchainsimulacion::Simulador, 71
Simulador::Simulador, 65
verUsuarios
blockchainsimulacion::Simulador, 71
Simulador::Simulador, 66
version
conf, 17