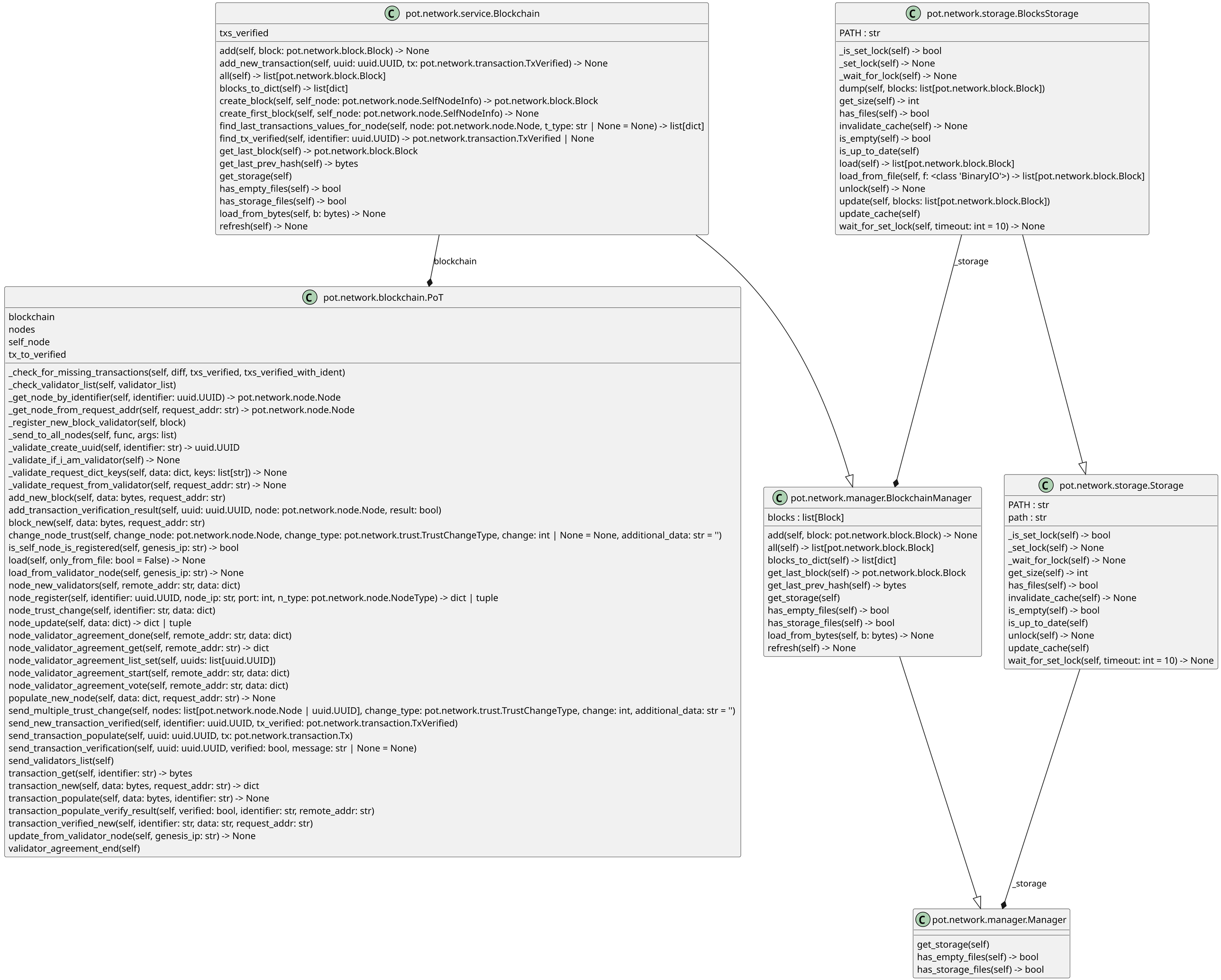
pot.network.service.Blockchain
txs_verified
add(self, block: pot.network.block.Block) -> None
add_new_transaction(self, uuid: uuid.UUID, tx: pot.network.transaction.TxVerified) -> None
all(self) -> list[pot.network.block.Block]
blocks_to_dict(self) -> list[dict]
create_block(self, self_node: pot.network.node.SelfNodeInfo) -> pot.network.block.Block
create_first_block(self, self_node: pot.network.node.SelfNodeInfo) -> None
find_last_transactions_values_for_node(self, node: pot.network.node.Node, t_type: str | None = None) -> list[dict]
find_tx_verified(self, identifier: uuid.UUID) -> pot.network.transaction.TxVerified | None
get_last_block(self) -> pot.network.block.Block
get_last_prev_hash(self) -> bytes
get_storage(self)
has_empty_files(self) -> bool
has_storage_files(self) -> bool
load_from_bytes(self, b: bytes) -> None
refresh(self) -> None
pot.network.storage.BlocksStorage
PATH : str
_is_set_lock(self) -> bool
_set_lock(self) -> None
_wait_for_lock(self) -> None
dump(self, blocks: list[pot.network.block.Block])
get_size(self) -> int
has_files(self) -> bool
invalidate_cache(self) -> None
is_empty(self) -> bool
is_up_to_date(self)
load(self) -> list[pot.network.block.Block]
load_from_file(self, f: <class 'BinaryIO'>) -> list[pot.network.block.Block]
unlock(self) -> None
update(self, blocks: list[pot.network.block.Block])
update_cache(self)
wait_for_set_lock(self, timeout: int = 10) -> None
blockchain
_storage
pot.network.blockchain.PoT
blockchain
nodes
self_node
tx_to_verified
_check_for_missing_transactions(self, diff, txs_verified, txs_verified_with_ident)
_check_validator_list(self, validator_list)
_get_node_by_identifier(self, identifier: uuid.UUID) -> pot.network.node.Node
_get_node_from_request_addr(self, request_addr: str) -> pot.network.node.Node
_register_new_block_validator(self, block)
_send_to_all_nodes(self, func, args: list)
_validate_create_uuid(self, identifier: str) -> uuid.UUID
_validate_if_i_am_validator(self) -> None
_validate_request_dict_keys(self, data: dict, keys: list[str]) -> None
_validate_request_from_validator(self, request_addr: str) -> None
add_new_block(self, data: bytes, request_addr: str)
add_transaction_verification_result(self, uuid: uuid.UUID, node: pot.network.node.Node, result: bool)
block_new(self, data: bytes, request_addr: str)
change_node_trust(self, change_node: pot.network.node.Node, change_type: pot.network.trust.TrustChangeType, change: int | None = None, additional_data: str = '')
is_self_node_is_registered(self, genesis_ip: str) -> bool
load(self, only_from_file: bool = False) -> None
load_from_validator_node(self, genesis_ip: str) -> None
node_new_validators(self, remote_addr: str, data: dict)
node_register(self, identifier: uuid.UUID, node_ip: str, port: int, n_type: pot.network.node.NodeType) -> dict | tuple
node_trust_change(self, identifier: str, data: dict)
node_update(self, data: dict) -> dict | tuple
node_validator_agreement_done(self, remote_addr: str, data: dict)
node_validator_agreement_get(self, remote_addr: str) -> dict
node_validator_agreement_list_set(self, uuids: list[uuid.UUID])
node_validator_agreement_start(self, remote_addr: str, data: dict)
node_validator_agreement_vote(self, remote_addr: str, data: dict)
populate_new_node(self, data: dict, request_addr: str) -> None
send_multiple_trust_change(self, nodes: list[pot.network.node.Node | uuid.UUID], change_type: pot.network.trust.TrustChangeType, change: int, additional_data: str = '')
send_new_transaction_verified(self, identifier: uuid.UUID, tx_verified: pot.network.transaction.TxVerified)
send_transaction_populate(self, uuid: uuid.UUID, tx: pot.network.transaction.Tx)
send_transaction_verification(self, uuid: uuid.UUID, verified: bool, message: str | None = None)
send_validators_list(self)
transaction_get(self, identifier: str) -> bytes
transaction_new(self, data: bytes, request_addr: str) -> dict
transaction_populate(self, data: bytes, identifier: str) -> None
transaction_populate_verify_result(self, verified: bool, identifier: str, remote_addr: str)
transaction_verified_new(self, identifier: str, data: str, request_addr: str)
update_from_validator_node(self, genesis_ip: str) -> None
validator_agreement_end(self)
pot.network.manager.BlockchainManager
blocks : list[Block]
add(self, block: pot.network.block.Block) -> None
all(self) -> list[pot.network.block.Block]
blocks_to_dict(self) -> list[dict]
get_last_block(self) -> pot.network.block.Block
get_last_prev_hash(self) -> bytes
get_storage(self)
has_empty_files(self) -> bool
has_storage_files(self) -> bool
load_from_bytes(self, b: bytes) -> None
refresh(self) -> None
pot.network.storage.Storage
PATH : str
path : str
_is_set_lock(self) -> bool
_set_lock(self) -> None
_wait_for_lock(self) -> None
get_size(self) -> int
has_files(self) -> bool
invalidate_cache(self) -> None
is_empty(self) -> bool
is_up_to_date(self)
unlock(self) -> None
update_cache(self)
wait_for_set_lock(self, timeout: int = 10) -> None
_storage
pot.network.manager.Manager
get_storage(self)
has_empty_files(self) -> bool
has_storage_files(self) -> bool