

## Universal Tokenisation System for Blockchain-based Cryptocurrencies

**Technical Field**

[0001] The present disclosure relates generally to a solution for the control and/or transfer of an asset, or the transfer of ownership of an asset. In particular, it relates to a method of creating, transferring ownership and redeeming tokens which represent assets. The present disclosure has particular application with creating tokens associated with transactions on a peer-to-peer distributed ledger such as, for example, the Bitcoin blockchain. The token may be representative of a contractual right, smart contract or other form of asset.

**Background**

[0002] Commercial transactions may involve the transfer of property rights. Such rights may include real property, or personal property (including both tangible and intangible property). Furthermore, contracts between parties may also include contractual rights that bind both parties. In the digital economy, there may be an expectation for transactions to be performed in a timely manner and across vast distances. This expectation, along with practical limitations, mean that traditional forms of transferring property, such as physical delivery of hardcopy of documents representing a contract, negotiable instrument, etc. or the tangible property itself is not desirable. In more recent times, blockchains have been used for the transfer of digital assets.

[0003] A blockchain is an electronic ledger which is implemented as a computer-based decentralised, distributed, peer-to-peer system made up of blocks which in turn are made up of transactions. Each transaction (Tx) is a data structure that encodes the transfer of control of a digital asset between participants in the blockchain system, and includes at least one input and at least one output. Each block contains a hash of the previous block so that blocks become chained together to create a permanent, unalterable record of all transactions which have been written to the blockchain since its inception. Transactions contain small programs known as scripts embedded into their inputs and outputs, which specify how and by whom the outputs of the transactions can be accessed. On the Bitcoin platform, these scripts are written using a stack-based scripting language.

[0004] In order for a transaction to be written to the blockchain, it must be “validated”. Network nodes (miners) perform work to ensure that each transaction is valid, with invalid transactions rejected from the network. Software clients installed on the nodes perform this validation work on an unspent transaction (UTXO) by executing its locking and unlocking scripts. If execution of the locking and unlocking scripts evaluate to TRUE, the transaction is valid and the transaction is written to the blockchain. Thus, in order for a transaction to be written to the blockchain, it must be i) validated by the first node that receives the transaction – if the transaction is validated, the node relays it to the other nodes in the network; and ii) added to a new block built by a miner; and iii) mined, i.e. added to the public ledger of past transactions.

[0005] Although blockchain technology is most widely known for the use of cryptocurrency implementation, digital entrepreneurs have begun exploring the use of both the cryptographic security system Bitcoin is based on and the data that can be stored on the Blockchain to implement new systems. It would be highly advantageous if the blockchain could be used for automated tasks and processes which are not limited to the realm of cryptocurrency. Such solutions would be able to harness the benefits of the blockchain (e.g. a permanent, tamper proof records of events, distributed processing etc) while being more versatile in their applications.

[0006] One area of current research is the use of the blockchain for the implementation of “smart contracts”. These are computer programs designed to automate the execution of the terms of a machine-readable contract or agreement. Unlike a traditional contract which would be written in natural language, a smart contract is a machine executable program which comprises rules that can process inputs in order to produce results, which can then cause actions to be performed dependent upon those results.

[0007] Another area of blockchain-related interest is the use of ‘tokens’ (or ‘coloured coins’) to represent and transfer real-world or virtual entities via the blockchain. A potentially sensitive or secret item can be represented by the token which has no discernable meaning or value. The token thus serves as an identifier that allows the asset to be referenced from the blockchain.

[0008] Any discussion of documents, acts, materials, devices, articles or the like which has been included in the present specification is not to be taken as an admission that any or all of these matters form part of the prior art base or were common general knowledge in the field relevant to the present disclosure as it existed before the priority date of each claim of this application.

[0009] In this document we use the term 'blockchain' to include all forms of electronic, computer-based, peer-to-peer, distributed ledgers. These include, but are not limited to consensus-based blockchain and transaction-chain technologies, permissioned and un-permissioned ledgers, shared ledgers and variations thereof. The most widely known application of blockchain technology is the Bitcoin ledger, although other blockchain implementations have been proposed and developed. While Bitcoin may be referred to herein for the purpose of convenience and illustration, it should be noted that the invention is not limited to use with the Bitcoin blockchain and alternative blockchain implementations and protocols fall within the scope of the present invention.

[0010] Throughout this specification the word "comprise", or variations such as "comprises" or "comprising", will be understood to imply the inclusion of a stated element, integer or step, or group of elements, integers or steps, but not the exclusion of any other element, integer or step, or group of elements, integers or steps.

## Summary

[0011] There is provided an invention as defined in the appended claims.

[0012] The present invention may provide solutions for the secure control and/or transfer of an asset or right via a blockchain. Additionally or alternatively, it may enable control and/or transfer of ownership of the asset or right. This may be a digital or virtual asset such as a smart contract, or a real-world/physical asset. The invention may use tokenisation techniques to facilitate this control or transfer. The invention may enable the transfer to be performed in a secure manner, incorporating the use of cryptographic keys, while not requiring any alteration of the underlying blockchain protocol.

[0013] The invention provides, not least: enhanced optimisation of memory usage for electronic transfers, improved security and data integrity through the use of hashing techniques, improved security through the removal of a need for a trusted third party, and enhanced anonymity of data. This list of advantages is not limiting or exhaustive.

[0014] The invention may require the interaction and inter-communication of various distinct and separate computer-based resources, such as one or more user devices and a distributed computer system (blockchain) which includes computing nodes arranged to execute blockchain-related software and protocols.

[0015] The invention may provide a method comprising the steps of:

- generating a blockchain transaction (Tx) having an output (TxO) related to a digital asset (B1) and a hash (H1) of a redeem script (RS1) which comprises:
  - metadata comprising a token which is a representation of, or a reference to, a tokenised entity;
  - and
  - at least one (preferably two or more) public cryptographic keys.

The digital asset (B1) may be a quantity of cryptocurrency eg Bitcoin. The redeem script may be provided within a locking script of the transaction output TxO. The metadata may be provided in the redeem script at a location which is designated in a blockchain protocol as a location for a cryptographic key.

The method may further include the step of submitting the transaction Tx to a blockchain. In effect, the cryptocurrency (B1) may thus be locked on the blockchain in association with the token. The quantity of cryptocurrency (B1) can only be spent (redeemed) upon provision of an unlocking script which meets the requirements of the locking script for the output TxO. In particular, a redeem script must be presented which, when hashed, matches the hash provided in the locking script of TxO. As the locking script for output TxO comprises the hash of the redeem script which in turn includes the token (in the metadata), the cryptocurrency (B1) is associated with the token. Upon presentation of the correct unlocking (redeem) script, ownership of the cryptocurrency (B1) may be transferred to the redeeming party or user i.e. it is spent.

The terms “spending”, “transferring”, “redeeming” or “transferring ownership/control of” may be used interchangeably herein. Also, term “user” may be used herein to refer to either a human user or a machine-based resource.

The public key(s) may be associated with a corresponding private key, to form a cryptographic key pair. The corresponding private key(s) may be required in order to unlock the transaction output (TxO) and thus enable transfer of the digital asset and/or ownership thereof.. The tokenised entity may be stored on or off the blockchain. It may be a digital asset such as a (smart) contract or some other form/type of asset or entity. The token may be provided in the redeem script such that it appears to be, or is interpreted by the blockchain protocol, as a cryptographic key. Thus, the underlying blockchain protocol may be agnostic to the presence of the token and/or other metadata provided in the redeem script. However, the metadata can be interpreted and used as a token by a user who is party to the process of the invention.

Thus, the invention may comprise embodiments or aspects which enable the issuance of a digital token to a user in a cryptographically enforced and secure manner via a blockchain. A corresponding system may be provided, the system arranged to implement the method of any embodiment described above and comprising a blockchain network and associated nodes.

[0016] Additional or alternative wordings, features or embodiments of the invention are now provided. Features described in relation to one or more aspects or embodiments of the invention may be used in relation to one or more other aspects or embodiments.

[0017] The invention may provide a computer-implemented method of creating a first token (T1) by an issuer (I). The first token (T1) may be associated with a first quantity of an encrypted, electronically transferrable digital asset (B1).

[0018] In addition or alternative to the method described above, the method may comprise one or more of the following steps: receiving, over a communications network, a request from a first user (A) for a first token (T1); determining a first user public key (P1A), wherein the first user public key (P1A) forms a cryptographic pair with a first user private key (V1A); and

allocating a first quantity of an encrypted, electronically transferrable digital asset (B1) for association with the first token (T1); determining a first hash (H1) of a first redeem script (RS1), wherein the first redeem script (RS1) is based on: at least a first metadata (MD1) that includes information associated with the first token (T1); the first user public key (P1A); and a first issuer public key (P1I) associated with the issuer (I), wherein the first issuer public key (P1I) forms a cryptographic pair with a first issuer private key (V1I); sending, over the communications network, a first data output (O1) to a peer-to-peer distributed ledger comprising: an indication of a transaction of the first quantity of said digital asset (B1) to the first user (A); and the first hash (H1), wherein the first hash (H1) is associated with the first quantity of said digital asset (B1), to provide the first token (T1) that is associated with the first user (A) and issuer (I).

Determining a first hash (H1) of a first redeem script (RS1), wherein the first redeem script (RS1) is based on at least a first metadata (MD1) that includes information associated with the first token (T1), the first user public key (P1A) and a first issuer public key (P1I) associated with the issuer (I), and sending, over the communications network, a first data output (O1) comprising the first hash (H1) provides a number of advantages. Firstly, because the information regarding the token is securely embedded in a public ledger such as a blockchain, security of data transmission is provided, while avoiding the need for a trusted third party, since the transacting parties can rely on the details of the related transactions being locked in a publicly verifiable manner. Also, anonymity of the transactions is preserved, and since the first redeem script is hashed, it would be impracticably difficult to change the values of the metadata without causing a change in the corresponding hash values of the redeem script. The advantage is also provided that the first metadata can be embedded in one or more of the places available for public keys in the redeem script, thereby enabling nodes not suitable for processing the metadata to simply transmit the redeem script to a further node, as opposed to blocking its process. This in turn improves computational efficiency of the related transactions. The further advantage is provided that the metadata may contain a pointer to terms and conditions of the contract, thereby enabling this information to be stored in an off-block repository. This enables the amount of processing and memory resources used to be reduced, since the transaction can be processed without the need to transmit its entire transactional history, while also enabling the details of the related transactions to be reliably verified afterwards. A further advantage is provided that control data can be incorporated into the metadata, for example an access code for a barrier in the case of the token representing a

ticket for a venue or a travel ticket or voucher. A yet further advantage is provided that the token can be divided, enabling two or more transaction outputs, each of which can relate to a tokenised or untokenised digital asset.

[0019] The first data output (O1) may facilitate recording of a pay to script hash transaction.

[0020] In the method, the step of receiving a request from a first user (A) for a token (T) may comprise receiving an offer or an acceptance of a contract. The step of receiving a request from a first user (A) for a token (T) may further comprise receiving at least one or more terms and conditions of a contract.

[0021] The method may further comprise sending at least one or more terms and conditions of a contract to the first user (A).

[0022] The information in the first metadata (MD1) may comprise a hash of at least one or more terms and conditions of a contract. The information in the first metadata (MD1) comprises information on one or more of: a type of contract; one or more terms and conditions of a contract; a pointer to terms and conditions of a contract; and information on how to process the transaction.

[0023] The method may further comprise storing the first redeem script (RS1) in a data store.

[0024] In the method, the first redeem script (RS1) may be in the format:

<NumSigs MD1 ... P1A P1I ... NumKeys OP\_CHECKMULTISIG>

wherein

NumSigs is the number of signatures required to redeem the first token (T1);

NumKeys is the total of public key slots in the script, including the metadata and public keys;

OP\_CHECKMULTISIG is an operation comparing signatures with the public key slots in sequential order.

[0025] The method may further comprise: determining whether the first user (A) has an account (ACA) with the issuer (I) to facilitate transactions associated with the first token (T1), wherein if the first user (A) does not have an account, the method further comprises: sending, over a communications network, a request to open an account (ACA) for the first user (A), wherein the account (ACA) is associated with the cryptographic pair including the first user private key (V1A) and the first user public key (P1A) for the first user (A).

[0026] In the method, the step of allocating a first quantity of said digital asset (B1) for association with the first token (T1) may comprise: determining a first token value (TV1) of the first token (T1); determining a pegging rate (PR1) for the first token (T1); and determining the first quantity of said digital asset (B1) based on the pegging rate (PR1) and the first token value (TV1).

[0027] In one alternative, the step of allocating a first quantity of said digital asset (B1) for association with the first token (T1) may comprise: determining a minimum threshold of said digital asset (MT1) of the first token (T1); and determining a first quantity of said digital asset (B1) that is at or above the minimum threshold of said digital asset (MT1).

[0028] A computer-implemented method of redeeming a first token (T1) associated with a first quantity of said digital asset (B1) according to the method of creating a first token (T1) described above, the method comprising the issuer: receiving, over the communications network, a request from the first user (A) to redeem the first token (T1); determining the first redeem script (RS1) associated with the first token (T1); receiving the first user private key (V1A); signing, with the first user private key (V1A) and the first issuer private key (V1I), the first redeem script (RS1) to unlock the first quantity of said digital asset (B1) associated with the first token (T1); and sending, over the communications network, a second data output (O2) to the peer-to-peer distributed ledger comprising an indication of a transaction of the first quantity of said digital asset (B1) to the issuer (I).

[0029] In the method of redeeming a first token (T1), the first token (T1) may have a token value of a first portion (R1) and a second portion (R2), and wherein the request from the first

user (A) to redeem the first token (T1) comprises a request to redeem a value of the first portion (R1), the method further comprising: determining the first user public key (P1A); and allocating a second quantity of said digital asset (B2) for association with a second token (T2), wherein the second token has a second token value (TV2) based on the second portion (R2). The method also comprises determining a second hash (H2) of a second redeem script (RS2), wherein the second redeem script (RS2) is based on: at least a second metadata (MD2) that is based, at least in part, on the first metadata (MD1) associated with the first token (T1); the first user public key (P1A); and- the first issuer public key (P1I) associated with the issuer (I). In this method, the second data output (O2) to the public ledger may further comprise: an indication of a transaction of at least the second quantity of said digital asset (B2) to the first user (A); and the second hash (H2), wherein the second hash (H2) is associated with the second quantity of said digital asset (B2), to provide the second token (T2) that is associated with the first user (A) and the issuer (I).

[0030] A computer-implemented method of creating a third token (T3) by an issuer (I), wherein the third token is associated with a transfer of value from a first token (T1), according to the method of creating a first token (T1) described above, the method comprising: receiving, over the communications network, a request from the first user (A), and/or a second user (B), to create the third token (T3); determining the first redeem script (RS1) associated with the first token (T1); receiving the first user private key (V1A); and signing, with the first user private key (V1A) and the first issuer private key (V1I), the first redeem script (RS1) to unlock the first quantity of said digital asset (B1) associated with the first token (T1). The method further includes determining a second user public key (P1B), wherein the second user public key (P1B) forms a cryptographic pair with a second user private key (V1B); and allocating a third quantity of said digital asset (B3) for association with the third token (T3). The method further includes determining a third hash (H3) of a third redeem script (RS3), wherein the third redeem script (RS3) is based on: at least a third metadata (MD3) that is based, at least in part, on the first metadata (MD1) associated with first the token (T1); the second user public key (P1B); and the first issuer public key (P1I). The method also includes sending, over the communications network, a third data output (O3) to the peer-to-peer distributed ledger comprising: an indication of a transaction of at least the third quantity of said digital asset (B3) to the second user (B); and the third hash (H3), wherein the third hash (H3) is associated with the third quantity of said digital asset (B3) to provide the third token (T3) that is associated with the second user (B) and the issuer (I).

[0031] In the method of creating a third token (T3), the first token (T1) may have a token value of a first portion (R1) and a second portion (R2), and wherein the request to create the third token (T3) comprises a request to create the third token (T3) with a third token value (TV3) based on the first portion (R1), the method further comprising: determining the first user public key (P1A); and allocating a second quantity of said digital asset (B2) for association with a second token (T2), wherein the second token has a second token value (TV2) based on the second portion (R2). The method further includes determining a second hash (H2) of a second redeem script (RS2), wherein the second redeem script (RS2) is based on; at least a second metadata (MD2) that is based, at least in part, on the first metadata (MD1) associated with the first token (T1); the first user public key (P1A); the first issuer public key (P1I) associated with the issuer (I). In the method, the third data output (O3) to the peer-to-peer distributed ledger may further comprise: an indication of a transaction of at least the second quantity of said digital asset (B2) to the first user (A); and the second hash (H2), wherein the second hash (H1) is associated with the second quantity of said digital asset (B2), to provide the second token (T2) that is associated with the first user (A) and the issuer (I).

[0032] In the method of creating a third token (T3), the step of allocating a second quantity of said digital asset (B2) may comprise: determining a pegging rate (PR2) for the second token (T2); and determining the second quantity of said digital asset (B2) based on the pegging rate (PR2) and the second token value (TV2).

[0033] In the method of creating a third token (T3), the step of allocating a second quantity of said digital asset (B2) may comprise: determining a minimum threshold of said digital asset (MT2) of the second token (T2); and determining the second quantity of said digital asset (B2) that is at or above the minimum threshold of said digital asset (MT2) of the second token (T2).

[0034] In the method of creating a third token (T3), the second quantity of said digital asset (B2) and/or the third quantity of said digital asset (B3) includes, at least part, the first quantity of said digital asset (B1).

[0035] The method of creating a third token (T3) may further comprise: determining a fourth quantity of said digital asset (B4) as a transaction fee, wherein the first data output (O1), second data output (O2) and third data output (O3) to the peer-to-peer distributed ledger

further comprises an indication of a transaction of the fourth quantity of said digital asset (B4) as a transaction fee.

[0036] In the above described methods, the peer-to-peer distributed ledger may comprise the bitcoin block chain.

[0037] A computer-implemented method of redeeming a first token (T1) associated with a first quantity of said digital asset (B1) as defined above, the method comprising the issuer: receiving, over the communications network, a request from the first user (A) to redeem the first token (T1); determining the first redeem script (RS1) associated with the first token (T1); sending, over the communications network, the first redeem script (RS1) for signing by the first user (A); receiving, over the communications network, a first redeem script signed by the first user (RS1A) with the first user private key (V1A); signing, with the first issuer private key (V1I), the first redeem script signed by the first user (RS1A) to unlock the first quantity of said digital asset (B1) associated with the first token (T1); sending, over the communications network, a second data output (O2) to the peer-to-peer distributed ledger comprising an indication of a transaction of the first quantity of said digital asset (B1) to the issuer (I).

[0038] A computer-implemented method of creating a third token (T3) by an issuer (I), wherein the third token is associated with a transfer of value from a first token (T1), according to the method of creating a first token (T1) described above, the method comprising: receiving, over the communications network, a request from the first user (A), and/or a second user (B), to create the third token (T3); determining the first redeem script (RS1) associated with the first token (T1); sending, over the communications network, the first redeem script (RS1) for signing by the first user (A); receiving, over the communications network, a first redeem script signed by the first user (RS1A) with the first user private key (V1A); signing, with the first issuer private key (V1I), the first redeem script signed by the first user (RS1A) to unlock the first quantity of said digital asset (B1) associated with the first token (T1); determining a second user public key (P1B), wherein the second user public key (P1B) forms a cryptographic pair with a second user private key (V1B); allocating a third quantity of said digital asset (B3) for association with the third token (T3); determining a third hash (H3) of a third redeem script (RS3), wherein the third redeem script (RS3) is based on: at least a third metadata (MD3) that is based, at least in part, on the first metadata (MD1) associated with

first the token (T1); the second user public key (P1B); and the first issuer public key (P1I); sending, over the communications network, a third data output (O3) to the peer-to-peer distributed ledger comprising: an indication of a transaction of at least the third quantity of said digital asset (B3) to the second user (B); and the third hash (H3), wherein the third hash (H3) is associated with the third quantity of said digital asset (B3) to provide the third token (T3) that is associated with the second user (B) and the issuer (I).

[0039] The invention may further provide a tokenisation method. It may be implemented using a distributed peer-to-peer network, such as a blockchain. Thus, all of the known advantages of blockchain technology can be harnessed by the invention. The invention may provide an improved solution for secure transfer of data items via a blockchain platform. It may be described as a solution for efficient secure transfer of tokens on, for example, a blockchain.

[0040] The method may provide a solution for representing a contract for the supply of at least one asset and/or service. The contract may be a transferable contract.

[0041] The invention is not limited with regard to the nature, type, quantity etc of the asset and/or service being transferred by the contract. The method may provide a codification scheme for tokenising any type of transferable contract.

[0042] The method may comprise the step of generating a blockchain transaction. The transaction may comprise three parameters or data items. This data may be indicative of:

[0043] i) an amount of shares available under the contract (this may be referred to herein as 'NumShares');

[0044] ii) a quantity of transfer units to be transferred from a sender to at least one recipient (this may be referred to herein as 'ShareVal'); and

iii) a factor for calculating a value for the quantity of transfer units (this may be referred to herein as a 'pegging rate').

[0045] An advantage of the invention is that it can be used to encapsulate or represent contracts as tokens on a blockchain. This can be achieved using only the three parameters

described above. In effect, the contract can be specified using a minimum of these three data items. As the invention provides a solution which can be used for any type of transferable contract, common algorithms can be devised and applied.

[0046] These three data items may be provided as metadata within the transaction. Advantageously, only a small amount of metadata (in bytes) is required to represent any type of contract. Thus, the invention provides an efficient and effective yet powerful mechanism for the transfer of tokens on a peer-to-peer distributed system such as a blockchain.

[0047] The data may further comprise an indication of the availability of the asset(s) i.e. goods and/or service(s) to be supplied under the contract. Additionally or alternatively, it may comprise an indication of the type and/or quantity of the asset(s) or service(s) being supplied under the contract.

[0048] The transaction may in practice also include a fourth parameter or data item, which identifies that particular asset being transferred, e.g. a house or racehorse, to which the transaction relates.

[0049] The method may further comprise the step of sending the transaction to the at least one recipient (or an address associated with the at least one recipient).

[0050] The transaction may comprise the contract; and/or information for accessing the contract or accessing a file containing the contract. This provides the advantage of a facility to transfer either the contract or information corresponding to the location of a file containing the contract via the blockchain.

[0051] The transaction may comprise a hash of the contract. This provides the advantage of a means to verify the authenticity of the contract.

[0052] The transaction may comprise at least one cryptographic signature. This provides the advantage of identifying an origin of the transaction.

[0053] The transaction may comprise at least one locking script and at least one public key. This provides the advantage of preventing the contract from being redeemed by someone other than an intended recipient of the transaction.

[0054] The transfer units may relate to a cryptocurrency, which may or may not be Bitcoin related. The transfer units may be an amount of Bitcoin, and the blockchain may be the Bitcoin Blockchain. This provides the advantage that the method may be implemented on existing infrastructure.

[0055] Thus, the invention provides a simple and efficient scheme for the secure transfer of contracts on a blockchain.

[0056] A computer program comprising machine-readable instructions to cause a processing device to implement any one of the methods described above.

[0057] A device including a processing device to perform the method according to any one of the methods described above.

### **Brief Description of Drawings**

[0058] Examples of the present disclosure will be described with reference to:

[0059] Fig. 1 is a schematic of an example system to create and redeem tokens;

[0060] Fig. 2(a) is a diagram illustrating an example of a first type of transaction between a first user and an issuer that includes creating a token;

[0061] Fig. 2(b) is a diagram illustrating an example of a second type of transaction between a first user and an issuer that redeems a token;

[0062] Fig. 2(c) is a diagram illustrating an example of a third type of transaction between the first user and a second user, and facilitated by the issuer, that results in a transfer of value of the token from the first user to the second user; and

[0063] Fig. 3(a) is a diagram illustrating an example of a fourth type of transaction between the first user and the issuer that redeems a portion of value of the token;

[0064] Fig. 3(b) is a diagram illustrating an example of a fifth type of transaction between the first user and the second user, and facilitated by the issuer, that results in a transfer of a portion of the value of the token from the first user to the second user;

[0065] Fig. 4 is a flow chart of a computer-implemented method for creating a token;

[0066] Fig. 5 is a flow chart of a computer-implemented method of registration of a user;

[0067] Fig. 6 is a flow chart of another example of a computer-implemented method of creating a token;

[0068] Fig. 7 is a flow chart of a computer-implemented method of redeeming a token;

[0069] Fig. 8 is a flow chart of additional steps in the computer-implemented method of Fig. 7;

[0070] Fig. 9 is a flow chart of a computer-implemented method of transferring value from a first user to a second user, facilitated by the issuer;

[0071] Fig. 10 is a flow chart of additional steps in the computer-implemented method of Fig. 9;

[0072] Fig. 11 is a flow chart of a variation of a computer-implemented method of redeeming a token whereby the redeem script is sent to the first user for signing;

[0073] Fig. 12 is a flow chart of a variation of a computer-implemented method of transferring value from a first user to a second user, facilitated by the issuer, whereby the redeem script is sent to the first user for signing; and

[0074] Fig. 13 illustrates a schematic example processing device.

## **Description of Embodiments**

### Overview of the system

[0075] A method and system to create, redeem and transfer tokens will now be described. Fig. 1 illustrates a system 1 that includes an issuer (I) 3, a first user (A) 5 and a second user (B) 7. The issuer (I) 3 creates tokens and may be, for example, a bank, another financial institution, mint, company, etc. The issuer (I) 3 is associated first processing device 13, for performing the methods 100, 200, 300, and is in communications with a communications network 8. Although the first processing device 13 is illustrated as a single node, it is to be appreciated that the method 100, 200, 300 described herein may be performed by more than one processing device 13 or nodes associated with the issuer (I) 3 and one or more steps may be distributed and performed at different nodes. The issuer (I) may also have an associated data store 11.

[0076] The first user (A) 5 is associated with a second processing device 15 that is in communication, over a communications network 8, with the first processing device 13 of the issuer (I) 3. The first user (A) 5 may request the creation of tokens from the issuer (I) 3, redeeming of tokens with the issuer (I) 3, or request part of or all the value of a token be transferred to the second user (B) 7.

[0077] The second user (B) 7 is associated with a third processing device 17 that is in communication, over a communications network 8, with the first processing device 13. In some examples, the second processing device 15 and the third processing device 17 may be a computer, a mobile communication device, or other electronic device used by the respective first and second user 5, 7. In other examples, the second processing device 15 and the third processing device 17 may be virtual machines accessible by the first and second users via a terminal or other interface.

[0078] Also illustrated is a peer-to peer distributed ledger 9 to record transactions. The peer-to-peer distributed ledger 9 may be associated with one or more processing devices 19 to receive and record transactions. An example of a peer-to-peer distributed ledger includes the block chain, which is a distributed database of transactions based on the bitcoin protocol. Thus the processing device 19 associated with the ledger may be processing devices used by “miners”.

#### Overview of transactions involving tokens

[0079] In a non-limiting example, there are three general types of transactions that involve tokens as illustrated in Fig. 2. In this example the issuer (I) is a financial institution that also manages an electronic wallet of the first user (A) 5 and second user (B) 7.

[0080] The first type of transaction, as shown in Fig. 2(a), involves the creation of a first token (T1) where the first user (A) transfers fiat currency (e.g. \$1,000 AUD) to the issuer (I) 3. In exchange for the fiat currency, the issuer (I) “tokenises” a first quantity of cryptocurrency (B1) such that it has a token value and transfers this first quantity of cryptocurrency (B1) to the first user (A) 5. The first token (T1) may be representative of a contract, such as a contract where the issuer (I) agrees to redeem the bearer of the first token (T1) for a specified fiat currency amount (e.g. \$1,000 AUD). Therefore, the first token (T1) may be similar to a negotiable instrument. By “cryptocurrency” is meant an encrypted, electronically transferrable digital asset such as, but not limited to, Bitcoin.

[0081] The second type of transaction, as shown in Fig. 2(b), involves the first user (A) 5 redeeming the first token (T1) with the issuer (I) 3. In this transaction, the first quantity of cryptocurrency (B1) is transferred from the first user (A) 5 to the issuer (I) 3. In return, the issuer (I) 3 sends the redeemed value, in the form of fiat currency, to the first user (A) 5. The first quantity of cryptocurrency (B1) transferred to the issuer (I) 3 may be spent in other subsequent transactions. Whether this first quantity of cryptocurrency (B) with the issuer (I) 3 remains “tokenised” or is converted to “untokenised” cryptocurrency may be a choice for the issuer (I) 3.

[0082] The third type of transaction, as shown in Fig. 2(c), involves the first user (A) 5 transferring the value of the first token (T1) to the second user (B) 7. In this example, the first quantity of cryptocurrency (B1) that represents the first token (T1) is transferred from the first user (A) 5 to the second user (B) 7. The issuer (I) 3 is involved in this transaction as the issuer (I) needs to sign the redeem script (discussed below) to authorise the transaction. The result of this transaction is the second user (B) 7 having the first quantity of cryptocurrency (B1) that has a token value which may be “spent” by redeeming with the issuer (I) 3 (in a similar manner as discussed above) or transferred to another user.

[0083] In some instances, only a portion of the value of a first token (T1) may be spent by the first user (A) 5. This will be described as the fourth and fifth types of transactions with

reference to Figs. 3(a) and 3(b) which are variations of the second and third types of transactions discussed above. In these examples, the first token (T1) has a total token value made up of a first portion (R1) plus a second portion (R2), and the first user (A) 5 wishes to “spend” the first portion (R1) and have the second portion (R2) returned as change.

[0084] In the fourth type of transaction, as show in Fig. 3(a), the first user (A) 5 has a first quantity of cryptocurrency (B1) representing a first token (T1) from a previous transaction. The first user (A) then redeems the first portion (R1) of the token (T1) by transferring a quantity of cryptocurrency as tokens with a value of the first portion (R1) and in return receives fiat currency with a value representing the first portion (R1). Since only the first portion (R1) is redeemed, the remaining second portion (R2) stays with the first user (A) 5. This is shown as the second quantity of cryptocurrency (B2) as tokens with a value representing the second portion (R2). In one example, the second quantity of cryptocurrency (B2) is the first quantity of cryptocurrency (B1) less the quantity transferred to the issuer (I) 3.

[0085] In the fifth type of transaction, as shown in Fig. 3(b), the first user also has a first quantity of cryptocurrency (B1) representing a first token (T1) from a previous transaction. The first user (A) then transfers the first portion (R1) of the tokens (T1) by transferring a third quantity of cryptocurrency (B3) as tokens with a value of the first portion (R1) to the second user (B) 7. Since only the first portion (R1) is transferred, the remaining second portion (R2) stays with the first user (A) 5. This is shown as the second quantity of cryptocurrency (B2) as tokens with a value representing the second portion (R2). In one example, the second quantity of cryptocurrency (B2) and the third quantity of cryptocurrency (B3) is derived from the first quantity of cryptocurrency (B1).

[0086] Detailed examples of the above mentioned types of transactions will now be described.

#### **First type of transaction - Issuer creates a first token (T1) for the first user (A)**

[0087] A non-limiting example application of the method described herein includes a first user (A) 5 that deposits a fiat currency amount (e.g. \$1000 AUD) to the issuer (I) (e.g. a financial institution) who in turn creates a first token (T1) for the first user (A) 5 that represents the value of the deposited fiat currency. Depending on the particular terms and

conditions, the first user (A) 5 may redeem the first token (T1) at a future date for a value associated with the deposited fiat currency. The terms and conditions may also allow the first user (A) 5 to have at least part of the value of the token transferred to a second user (B). Such terms and conditions may be specific to the token or may be general terms and conditions between the users 5, 7 and the issuer (I) 3.

#### Overview of the method of creating a token

[0088] An example of a method 100 of creating a first token (T1) by the first processing device 13 at the issuer (I) 3 will be now be described referring to Fig. 2(a) and 4. The method 100 includes receiving 110, over the communications network 8, a request from the first user (A) 5 for a first token (T1). The method also includes determining 120 a first user public key (P1A) which forms a cryptographic pair with a first user private key (V1A).

[0089] The method 100 includes allocating 130 a first quantity of cryptocurrency (B1) for association with the first token (T1). The method 100 further includes determining 140 a first hash (H1) of a first redeem script (RS1), wherein the first redeem script (RS1) is based on: at least a first metadata (MD1) that includes information associated with the first token (T1); the first user public key (P1A); and a first issuer public key (P1I) associated with the issuer (I), wherein the first issuer public key (P1I) forms a cryptographic pair with a first issuer private key (V1I).

[0090] The method 100 also includes sending 150, over the communications network 8, a first data output (O1) to the peer-to-peer distributed ledger 9. The first data output (O1) includes: an indication of a transaction of the first quantity of cryptocurrency (B1) to the first user (A) 5; and the first hash (H1), wherein the first hash (H1) is associated with the first quantity of cryptocurrency (B1), to provide the first token (T1) that is associated with the first user (A) 5 and issuer (I).

[0091] Thus the method 100 allows creation of a token whereby a record of the token is sent to the peer-to-peer distributed ledger 9. An advantage of recording this transaction on the peer-to-peer distributed ledger 9 is that it may allow the recipient, such as the first user (A) 5 to validate the existence of the token (T1). Furthermore, since the at least first metadata (MD1) that includes information associated with the first token (T1) is hashed, this allows the

validation of the transaction (which is on the public record) against the information associated with the token. In one example, information associated with the first token (T1) may be terms and conditions of a contract. Thus including the terms and conditions in the first redeem script to be hashed may advantageously provide comfort to the first user (A) 5 (or any other user) that the terms and conditions cannot be varied as any variation would alter the first hash (H1). Since the first hash (H1) was sent and recorded on the peer-to-peer distributed ledger 9 at the time of creating the first token (T1), it would be impossible (or difficult) to alter the terms and conditions at a later time that would provide an identical first hash (H1).

A detailed example of the issuer (I) 3 creating tokens for the first user (A) 5 will now be described, including the initial process of registration 400.

#### Method of registration 400

[0092] A method of registration 400 will now be described with reference to Fig. 5. The method 100 may include determining 312 whether the first user (A) 5 has an account with the issuer (I). In particular, this may include determining if the first user (A) 5 has an account that is suitable for facilitating transactions associated with the first token (T1).

[0093] If the first user (A) 5 does not have an account, the method may further include sending 314, over a communications network 8, a request to open an account for the first user (A). The request to open account may include sending, to the first user (A) 5, general details of terms and conditions for an account with the issuer (I) as well as a request to the first user (A) 5 to accept the terms and conditions. This may also include sending a request for details of the first user (A) 5.

[0094] In addition, sending a request to open an account may also include sending a request for generation of a cryptographic pair including a first user private key (V1A) and the first user public key (P1A) for the first user (A) 5. In some examples, this may include sending a request to another node associated with the issuer (I) to generate the first user private key (V1A) and the first user public key (P1A), whereby the another node in turn generates and sends to the first user (A) 5. In other examples, this may include sending to the first user (A) 5 a request to generate, at the second processing device 15 associated with the first user (A) 5, the first user private key (V1A) and the first user public key (P1A). It is to be appreciated that

these key associated with the first user (A) 5 may be generated by, or at, other means, so long as the first user private key (V1A) is kept secure and used only with authorisation with the first user (A) 5.

[0095] The first user (A) 5 may receive 714 the request to open an account and in turn send 716, to the issuer (I) 3, information to open the account.

[0096] The method of registration 400 may also include creating 316 an electronic wallet associated with the account of the first user (A) 5 and storing information associated with this electronic wallet and account in the data store 11.

[0097] In some examples, the first user private key (V1A) may be stored in the electronic wallet, whereby ideally the first user private key (V1A) is only accessible with authorisation from the first user (A) 5. For example, the electronic wallet may have multiple private keys associated with the first user (A) 5, whereby the first private keys are available when the first user (A) 5 successfully, and securely, logs on with issuer (I) 3 (such as in a virtual machine environment or a terminal). The first user (A) 5, may then selectively authorise the issuer (3) to retrieve and use these private keys from the data store 11 for transactions. In some examples, the user private keys are not stored in the electronic wallet, but are able to be recreated by the issuer (I) 3 with authorisation from the respective user. In yet another examples, the user private keys may be split keys, with the electronic wallet having one portion and the user having the remaining portion, whereby they can be combined to recreate the private key.

[0098] In other examples, the first user private key (V1A) may be kept separate to the issuer (I), first processing device 13, and data store 11. For example, the first user (A) 5 may keep a hard copy of the first user private key (V1A) in a safe, or in a secure part of a personal electronic device, computer, or storage device.

[0099] It is to be appreciated that steps in the method 400 may be performed during method 100, for example after the step of receiving 110 a request from a first user (A) 5 for a first token (T1). In other examples, the method or registration 400 may be performed beforehand.

Detailed method of creating a token 100

[0100] The method 100 of creating a first token (T1) will now be described in detail with reference to Figs. 2(a), 4 and 6 (which shows the methods 100 and 500 performed by the issuer (I) 3 and first user (A) 5 respectively). In this example, the creating tokens will be discussed in the context of the first user (A) 5 depositing cash with the issuer (I) 3 in return for tokens representing the deposited cash. However, it is to be appreciated that this is a non-limiting example and that the tokens can be created in the context of other transactions. For example, the token may represent any other contract, negotiable instrument, tangible property, etc., and may represent a transferrable property including an access code for a barrier, for example in the case of the token representing a ticket for a venue or a travel ticket or voucher.

Agreeing on terms and conditions for the token

[0101] After or before the method of registration 400, the first user (A) 5 may send 510 a request for a first token (T1). In one example, the first user (A) 5 makes this request by depositing fiat currency, for example \$1000 AUD, with a request to have this amount in tokens (T1).

[0102] In one example, the request sent by the first user (A) 5, may include an offer for a contract. This offer may include one or more terms and conditions of a contract. For example, the first user (A) 5 may include in the request that the tokens associated with the deposit of \$1000 AUD should have a fixed pegging rate to cryptocurrency. For example, a request that the pegging rate is 1000 satoshi/cent (AUD). It is to be appreciated that other terms and conditions may be included in the offer, such as account keeping fees, transaction fees, how the tokens can be redeemed, etc.

[0103] As shown in Fig. 6, the first processing device 13 of the issuer (I) receives 110, over the communications network 8, the request from the first user (A) 5 for the first token (T1) and, in some cases, at least some of the terms and conditions. The issuer (I) may then determine 112 whether to accept the request, propose a counter offer that includes a modification of the terms and conditions of the request, or reject the request. The method 100 then includes sending 114, over the communications network 8, the result of the determination in step 112.

[0104] The first user (A) 5 may then receive 520, over the communications network 8, the result of the determination in step 112 that includes the acceptance, the counter offer, or rejection of the request.

[0105] In one alternative example, the request sent 510 to the issuer (I) may simply include a request for a first token (T1). In this case, the issuer (I) may send to an offer, including terms and conditions, to the first user (A) 5. The first user (A) 5 may, in turn, determine whether to accept the offer, propose a counter offer, or reject the offer, which is then sent to the issuer (I).

[0106] It is to be appreciated that steps 510, 520, and 110, 112, 114 may be modified in to include multiple rounds of offers and counter offers that are sent and received between the issuer (I) and first user (A) 5 until they are in agreement.

[0107] In some alternatives, the terms and conditions may be standardised, whereby the user accept the terms and conditions by performing the steps in the methods 100, 500. In one example, the issuer (I) may have standardised offers for tokens for their customers, including the first user (A) 5. Such offers for tokens may be listed publicly, such as on a public exchange or on the issuer's website. Standing offers may also be provided by the issuer (I) to the first user (A) 5 privately, such as by email, through an application, or by logging in a secure website.

[0108] The terms and conditions associated for the token may be stored in the data store 11, sent to a third party for storage, or torrented.

#### Determining the first user public key 120

[0109] The method 100 includes determining 120 the first user public key (P1A). In one example, the first user public key (P1A) may be sent 530 from the first user (A) 5, over the communications network 8, to the issuer (I). In another example, the first user public key (P1A) may be associated stored in the data store 11 (which may, for example, be received and stored during registration of the first user (A) 5). Thus the step of determining 120 the first user public key (P1A) includes retrieving the key from the data store 11. In yet another example, the first user public key (P1A) may be received, over the communications network

8, from a third party. The third party may include, for example, a trusted third party that acts as a public directory, such as a certification authority.

Allocating a first quantity of cryptocurrency for association with the token 130

[0110] The method 100 includes allocating 130 a first quantity of cryptocurrency (B1) for association with the first token (T1). In order for a record of a transaction involving the first token (T1) to be recorded on the peer-to-peer distributed ledger (which in this example is the block chain), the token must be associated with a quantity of cryptocurrency. In turn, that quantity of cryptocurrency is recorded on the peer-to-peer distributed ledger as a transaction from the issuer (I) 3 to the first user (A) 5. In other words, a blockchain transaction Tx is submitted to the blockchain network for inclusion in the ledger. The Tx includes an output which can be used to transfer the quantity of cryptocurrency (or ownership/control thereof) from one party eg issuer to another eg the first user.

[0111] The allocation of the first quantity of cryptocurrency (B1) for association with the first token (T1) may be based on a ratio of the token value. For example, a pegging rate (PR1) may be specified for the first token (T1). Thus the step of allocating 130 a first quantity of cryptocurrency (B1) may include determining a first quantity of cryptocurrency (B1) based on the pegging rate (PR1) and the first token value (TV1). As an illustrative example, the pegging rate (PR1) may be 1000 satoshis/cent AUD and the first token value (TV1) is \$1000 AUD. Thus the first quantity of cryptocurrency (B1) may be 10,000,000.

[0112] The quantity of cryptocurrency to be allocated for a token may be influenced by some of the following considerations. Firstly, the allocated quantity of cryptocurrency ideally has a market value (for this purpose, this means the market value of the cryptocurrency in itself, assuming it has a value, without reference to the token value) that is less than the value of the token (“token value”). This is desirable so that there is no motivation to use the quantity of cryptocurrency for the underlying value rather than as a token. This may be analogous to cash coins where it is desirable to have the face value of the coin to be higher than the metal the coin is minted from, so that there is no desire to melt the coins for the value of the metal. In some examples, the token value is multiples larger than the underlying value of the quantity of cryptocurrency. However, it is to be appreciated that some token may not have a fixed or easily determinable token value. For example, the token may be

representative of a contract to perform work, whereby the value may change day to day. In other examples, the contract may only have a value that is determinable on the day it is redeemed.

[0113] Another consideration is that the quantity of cryptocurrency allocated should not be too large, relative to the token value or the value of the transaction, since recording a transaction of the quantity of cryptocurrency on the peer-to-peer distributed ledger may be at a cost, such as incurring a transaction fee. In some examples, the transaction fee is based on the quantity of cryptocurrency in the transaction and therefore it may be desirable to keep the quantity of cryptocurrency for the token at a minimum level.

[0114] On the other hand, the quantity of cryptocurrency allocated for association with the token cannot be infinitely small. Firstly, the cryptocurrency may have a minimum denomination amount, and for example, Bitcoin has a minimum amount of one satoshi (where 1 bitcoin (BTC) = 10,000,000 satoshi). Secondly, a transaction of cryptocurrency may be limited to a minimum size or else it will not be recorded (or the cost of the transaction will be close to, or exceed, the cost of performing the transaction). This minimum amount, in some examples, is a “dust” limit. Thus in some examples, allocating a quantity of cryptocurrency for a token must be above a minimum threshold of cryptocurrency (MT1). Therefore the method 100 may include determining the minimum threshold of cryptocurrency (MT1) suitable for the first token (T1) and determining a first quantity of cryptocurrency (B1) that is at or above the minimum threshold of cryptocurrency (MT1). In one example, the minimum threshold of cryptocurrency (MT1), in “Bitcoin”, is 546 satoshis.

[0115] Another consideration when allocating the quantity of cryptocurrency for a token is divisibility of the quantity of cryptocurrency for subsequent tokens. For example, the first token (T1) may have a token value (TV1) of \$1000 AUD and the first user (A) 5 may wish to transfer \$800 AUD of the token value to the second user (B) 7 and keep the remaining \$200 AUD tokens. Such a transaction would involve a transaction with the first token (T1) that results in a second token (T2) representing \$200 AUD that stays with the first user (A) 5 as change and creating a third token (T3) representing \$800 AUD to be transferred to the second user (B) 7. Thus the result of this transfer is two tokens, the second token (T2) and third token (T3), where each of these tokens must also be allocated a quantity of cryptocurrency. If the first quantity of cryptocurrency (B1) was minimal, for example at the “dust” limit, then

further amounts of cryptocurrency will need to be sourced so that each of the new tokens created are also associated with sufficient quantities of cryptocurrency to satisfy a minimum threshold. Therefore, there may be advantages to allocating a sufficient quantity of cryptocurrency (B1) for the first token (T1) such that the amount is sufficient to be divided up to be used for an anticipated number of subsequent tokens.

In one example, the terms and conditions may specify the quantity of cryptocurrency or the minimum value or denomination of a token. For example, the term and conditions may set the minimum denomination of token value to \$10 AUD. Therefore, allocating a first quantity of cryptocurrency (B1) for a first token (T1) with a token value (TV1) of \$1000 AUD may include determining a first quantity that will ensure that there is sufficient cryptocurrency if the entire token value (TV1) is divided up to the smallest denomination. In this example, the token value (TV1) may be divided to 100 subsequent tokens (calculated by  $\$1000/\$10$ ). Therefore a suitable first quantity of cryptocurrency (B1) may be 100 times the “dust” limit.

Determining a first hash (H1) of a first redeem script (RS1) 140

[0116] The method further includes determining 140 a first hash (H1) of a first redeem script (RS1). In one example, the hash of the redeem script may be used to provide a pay to script hash (P2SH) address for a pay to script hash transaction. An example includes the hash functions used in P2SH script in the Bitcoin protocol. This may include a combination of SHA 256 followed by RIPEMD160.

[0117] The first redeem script (RS1) is a script that may be used to unlock the first token (T1) which, as discussed later, includes a transaction of the first quantity of cryptocurrency (B1). When unlocking the first token (T1) certain conditions of the first redeem script (RS1) must be met to unlock the transaction. In particular, the signatures of the first user (A) 5 and issuer (I) are required. An example of the first redeem script (RS1) will now be described.

The first redeem script (RS1)

[0118] The first redeem script (RS1) is based on: at least a first metadata (MD1) that includes information associated with the first token, the first user public key (P1A) and the first issuer public key (P1I).

*(i) Redeem script in P2SH in general*

[0119] As background, in a pay to script hash method the redeem script may take the form of:

<NumSigs PubK1 PubK2 ... PubK15 NumKeys OP\_CHECKMULTISIG>

where

NumSigs – is the number “m” of valid signatures required to satisfy the redeem script to unlock the transaction

PubK1, PubK2 ... PubK15 – are the public keys that correspond to signatures that unlock the transaction (up to a maximum of 15 public keys)

NumKeys – is the number “n” of public keys (which must be 15 or less)

[0120] To unlock the above redeem script, at least a number “m” of signatures corresponding to the public keys are required. In some examples, the order of the public keys are important and the number “m” out of “n” signatures for signing must be done in sequence. For example, say that “m” is two and the number of public keys “n” is fifteen. Say that two signatures are available for use, say Sig1 (corresponding to PubK1) and Sig 15 (corresponding to PubK15), the redeem script must be signed by Sig1 first followed by Sig15.

*(ii) The first redeem script (RS1) using P2SH*

[0121] Turning back to the present example, the first redeem script (RS1) that utilises P2SH may include the at least first metadata (MD1) in the redeem script. In particular, the at least first metadata (MD1) may be embedded in one or more of the 15 places available for the public keys in the redeem script.

[0122] Therefore in one example, the first redeem script (RS1) may take the form of:

<NumSigs Metadata1 Metadata2...PubK1 PubK2...NumKeys OP\_CHECKMULTISIG>

where

NumSigs – is the number “m” of valid signatures required to satisfy the redeem script to unlock the transaction

Metadata1 and Metadata2– includes metadata that takes the place of a public key

PubK1 and PubK2 – are actual public keys. In one example, PubK1 may be the first user public key (P1A) and PubK2 may be the issuer public key (P1I)

NumKeys - is the is total number of positions taken by the metadata and the public keys (which must be 15 or less)

[0123] The advantage of this is that the metadata will be included in the first redeem script (RS1), which in turn will be hashed and the record of which will be included in the peer-to-peer distributed ledger 9. Therefore it would be difficult, if not impossible, to change the values of the metadata without resulting in a change of the corresponding hash of the first redeem script hash (RS1).

[0124] A practical advantage may be illustrated by the following example. The first user (A) 5 and the issuer (I) 3 may wish to enter into a contract with particular terms and conditions. The contract may include the issuer (I) creating a token, whereby the specific terms and conditions are included in the metadata embedded in the redeem script. A hash of the redeem script is then recorded on the peer-to-peer distributed ledger 9, which becomes a record of the transaction that is difficult or impossible to change. Say the issuer (I) attempts to deceive the first user (A) 5, and for example, attempts to modify a term and alleges that the modified term was in the originally agreed contract. The first user (A) 5 may be able to contest this by placing the modified term in the metadata of the redeem script and hashing it, and then showing that this does not match the redeem script recorded on the peer-to-peer distributed ledger. Therefore including information associated with the token in the at least first metadata may be useful for ensuring the integrity of the token.

[0125] It is to be appreciated that the metadata in the redeem script may itself include a hash of other information. For example if the terms and conditions is lengthy, a hash of the terms and conditions may be used to provide a shorter metadata.

[0126] The first redeem script (RS1) may be stored in the data store 11 as a record and for redeeming the first token (T1). In some alternative examples, the first redeem script may be sent to the first user (A) 5, or a third party .

#### The metadata

[0127] In the present example, the first redeem script (RS1) takes the form:

<2 Metadata1 Metadata2 P1A P1I 4 OP\_CHECKMULTISIG>

[0128] Thus the at least first metadata (MD1) includes both Metadata1 and Metadata2 that occupies two of the places in the redeem script. This is followed by two public keys in sequence, the first user public key (P1A) and the first issuer public key (P1I). The NumSigs is 2 which mean two signatures are required to unlock the transaction.

[0129] The metadata may include information regarding the token in a number of ways. As discussed, in one example the terms and conditions may be included in the metadata. In another example, a hash of the terms and conditions may be included in the metadata. In yet another example, the metadata may include a pointer to a file that contains the terms and conditions of a contract. In further embodiments, combinations including one or more of the above may be included in the metadata.

#### *(i) Metadata with pointer to terms and conditions*

[0130] A specific example of the first metadata (MD1) is illustrated in Table 1 below.

| Field     | Sub-field         | Bytes | Value | Comments                                                              |
|-----------|-------------------|-------|-------|-----------------------------------------------------------------------|
| Metadata1 | ContractType      | 4     |       | Coded value indicates type of contract.                               |
|           | ContractPointer   | 16    |       | IPv6 address of the actual contract file location                     |
|           | ContractTypeData1 | 12    |       | Format depends on value of ContractType. Padded with zeros            |
| Metadata2 | ContractHash      | 20    |       | RIPEMD-160(SHA256(actual contract file addressed by ContractPointer)) |
|           | ContractTypeData2 | 12    |       | Format depends on value of ContractType. Padded with zeros            |

Table1

[0131] This example includes a minimum amount of information in relation to the token and transaction. This example includes providing a pointer to the contract which may be useful

where the size of the contract precludes including such details in the metadata. Furthermore, since the metadata may be made public, or transmitted over an unsecure network, it may be desirable that specific details of the token be veiled or hidden for privacy reasons.

[0132] The first 4 bytes of metadata1 indicates the type of contract. For example, the contract type may be for 'Fiat Currency'. The next 16 bytes holds the IP address of the location of the actual electronic contract file, making allowance for IPv6 addresses. Note that in some embodiments, this value may point to the seed of a torrent file such that the contract file can be distributed over the cloud rather than being centralised. The following 12 bytes contains data specific to the type of contract .

[0133] The first 20 bytes of metadata2 is a hash of the actual contract file using RIPEMD-160 over SHA256 applied to the file. As the actual contract file is retrievable this allows validation of the transaction against the contract. Note that the contract file itself may be completely public (unencrypted and human readable) or may be encrypted for privacy, depending on the requirements of the specific embodiment. The content of the remaining 12 bytes of metadata2 may be used depending on the type of contract.

*(ii) Metadata with key parameters of the token*

[0134] Another specific example of the first metadata (MD1) is illustrated in Table 2 below:

| Field     | Sub-field        | Bytes | Value      | Comments                                                                  |
|-----------|------------------|-------|------------|---------------------------------------------------------------------------|
| Metadata1 | ContractType     | 4     | 0x00000001 | Indicates Fiat Currency                                                   |
|           | ContractPointer  | 16    |            | IPv6 address of the actual contract file location                         |
|           | FiatDenomination | 2     |            | Coded value indicating currency (e.g. 0x0001=CAD, 0x0002=PHP, etc)        |
|           | PeggingRate      | 1     |            | Coded value represents the BTC/fiat pegging rate.                         |
|           | TransactionType  | 1     |            | Coded value represents type of output (issuance/payment/redemption)       |
|           | Padding          | 8     | 0x00000... | Spare bytes                                                               |
| Metadata2 | ContractHash     | 20    |            | RIPEMD-160(SHA256(the actual contract file addressed by ContractPointer)) |
|           | Padding          | 12    | 0x00000... | Spare bytes                                                               |

Table 2

[0135] In this example, some key parameters of the token included in the metadata. By key parameters, this may include information relevant the token itself or information that may assist processing of the transaction. In particular, the bytes allocated to the Sub-field "ContractTypeData1" in Table 1 above has been used for indicating: fiat denomination, pegging rate, and transaction type.

[0136] Importantly, inclusion of the key parameters in the metadata may assist greater processing efficiency as the issuer (I) 3 may, in some cases, process the tokens in transactions without retrieving the contract file for the key information required to process the transaction.

[0137] In addition to the above information, other information relating to the history of the token or the tokens preceding the token may be included. For example, if the first user (A) 5 wishes to redeem a portion of the first token (T1) and a second token (T2) is created by the issuer (I) to represent the value of the remaining portion, the issuer may embed information in the metadata to associate the second token (T2) with the first token (T1). This may assist the issuer (I) to account for and keep track of the tokens without the expense of tracing through the history of transactions which, for an issuer (I) such as a bank, can be an intensive task.

[0138] In Table 2, the metadata contains a 2-byte field to indicate the fiat currency (FiatDenomination) and 1-byte field called PeggingRate. The pegging rate is set by the Issuer (I). Several different rates may be set for the same fiat currency, however, a different token (with a different contract) will be needed for each different rate. The choice of rate may be at the discretion of the Issuer (I), however who may take similar considerations for the pegging rate as for the allocation of the quantity of cryptocurrency for the token as discussed above.

[0139] In one example, the PeggingRate is an 8-bit coded value as follows:

Leftmost bit will be used as a flag:

- 1 = rate expressed as satoshis/cent ('cent' refers to one hundredth of the fiat currency, which is the minimum fiat amount)
- 0 = rate expressed as cents/satoshi

The rightmost seven bits represents the rate as a power of ten in binary, for example:

- USD 10000010 means rate of 100 satoshis/cent (flag is on)
- PHP 00000000 means rate of 1 centavo/satoshi (flag is off)
- IDR 00000001 means rate of 10 rupiah/satoshi (flag is off)

[0140] In one example, TransactionType is a 1-byte field indicating whether the transaction is an "issuance" (in which a token is created from cryptocurrency), a payment (in which at least part of the token value is transferred from one user to another user); or a redemption (in which tokens are transferred to the issuer (I) and converted back to regular cryptocurrency).

[0141] In some examples, the “Padding” in both the Metadata1 and Metadata2 may include randomly generated values for each transaction. The result is that each of Metadata1 and Metadata2 will vary between transactions. An advantage is that this may lower the risk, and motivation, of an unscrupulous person trying to determine a private key that would match one or both of Metadata1 or Metadata2 as a cryptographic pair (for the purpose of using such a private key to sign the redeem script). This may be important for standardised tokens where the remaining bulk of the Metadata1 or Metadata2 is the same.

#### The public keys

[0142] The first user public key (P1A) and the issuer public key (P1I) are respectively paired with corresponding first user private key (V1A) and issuer private key (P1I). It is to be appreciated that the public keys may be known widely to the public, whilst in other examples it may be desirable to communicate the public keys as required. In any case, only the public key is required for the redeem script as the corresponding private keys are only required when signing and unlocking the redeem script (such as when redeeming the token).

[0143] As noted above, in some alternatives the first and second users 5, 7 may access their electronic wallet through a virtual machine environment or a terminal. The electronic wallet may be hosted by the issuer (I) 3 (or a server associated with the issuer (I) 3) wherein the private key(s) of a corresponding user are stored in the data store 11 but can only be accessed (or recreated) by the issuer (I) 3 with authorisation from that user. In such cases, the first and second users 5, 7 may authorise their private keys to be provided to the issuer (I) 3 to unlock the redeem script. This may include authorising the user’s private key(s) to be sent to the first processing device 13 of the issuer (I) 3, wherein the first processing device 13 may unlock the redeem script with the user’s private key(s) (e.g. P1A, P1B) and the first issuer public key (P1I).

#### Sending a first data output (O1) to a peer-to-peer distributed ledger 150

[0144] The method 100 further includes sending 150, over a communications network 8, a first data output (O1) to a peer-to-peer distributed ledger 9. This first data output (O1) may include an indication of a transaction of the first quantity of cryptocurrency (B1) to the first user (A) 5. That is, recording that the underlying quantity of cryptocurrency (B1) associated

the first token (T1) has been transferred to the first user (A) 5. The first data output (O1) also includes the first hash (H1) discussed above. The first hash (H1) is associated with the first quantity of cryptocurrency (B1), to provide a record of the first token (T1) that is associated with the first user (A) 5 and the issuer (I).

[0145] Importantly, the first hash (H1) is on the peer-to-peer distributed ledger 9 which can be used to prove or verify the existence of the token (T1), the relationship between the issuer (I) and first user (A) 5, and/or the terms and conditions of the token.

[0146] The method may also include storing 160 the first redeem script (RS1) in a data store 11 for later use.

[0147] A specific example of a transaction that creates a first token (T1) will now be described with reference to Fig. 2(a).

First user (A) 5 deposits \$1000 AUD to the Issuer (I) for equivalent value in a token

[0148] In this example, the first user (A) 5 wishes to deposit \$1000 AUD to the issuer (I), who, in return creates a first token (T1), with a token value (TV1) of \$1000 AUD, by associating this with a first quantity of cryptocurrency (B1) of 10,000,000.

[0149] To create tokens, the issuer (I) needs to have cryptocurrency. This may be sourced from previous transactions, or sourced in response to the request from the first user (A) 5 for a first token (T1). This is shown on the left hand side of Fig. 2(a) as the “First quantity of (untokenised) cryptocurrency”.

[0150] Table 3 below shows an originating transaction output, in the form of transaction-ID / Satoshis amount/ locking script. This originating transaction output represents the cryptocurrency that the issuer (I) has acquired from a previous transaction and in which at least some of the cryptocurrency will be used for association with the first token.

|                                                                 |
|-----------------------------------------------------------------|
| ID-201                                                          |
| 50,000,000                                                      |
| OP_DUP OP_HASH160 <PubK-Issuer hash> OP_EQUALVERIFY OP_CHECKSIG |

Table 3

[0151] The first line “ID-201” is a transaction identifier to identify this transaction. The next line is the number of satoshis in this transaction, which is 50,000,000. The third line is the locking script (output script) for this transaction. The redeem script in this output, <PubK-Issuer hash> , shows that this output has been locked with the first issuer public key (P1I). That is, this transaction can be unlocked using the issuer’s corresponding first issuer private key (V1I).

[0152] As discussed above, the method 100 includes allocating a first quantity of cryptocurrency (B1) suitable for the first token (T1). However, the quantities of cryptocurrency that the issuer (I) has on hand may not exactly match the first quantity of cryptocurrency (B1). In the present example, the required first quantity of cryptocurrency (B1) is 10,000,000 which is much less than the 50,000,000 in from the transaction ID-201.

[0153] Accordingly, a transaction to create a first token (T1) will include providing change of cryptocurrency back to the issuer (I) for the excess amounts of cryptocurrency that was not required for the token.

[0154] Furthermore, the creation of the token 100 may be a transaction that requires a payment of a transaction fee to a miner. This is illustrated with reference to Table 4 below which shows the transaction for creating the tokens.





10,000,000 of the first output and 39,999,000 of the second output). Thus there is a deficit of 1,000 satoshis. In this example, the 1,000 satoshis is a transaction fee (e.g. miner's fee).

### **Second type of transaction – First user (A) redeems tokens with issuer (I)**

#### Overview of redeeming a token

[0164] In this example, the issuer (I) is a service provider that provides electronic wallets for the users 5, 7, wherein the private keys of the users are kept secure in a data store 11 associated with the issuer (I) 3. Therefore in this example, the users 5, 7 (or their respective processing devices 15, 17) do not sign the redeem script. Instead, the issuer (I) 3, with authorisation from the users 5, 7, signs the redeem script. This may be illustrated in the methods 200 and 600 shown in Fig. 7 where the first user (A) 5 sends 610 a request to redeem a first token to the issuer (I) 3. Either implicitly, or explicitly, this request to redeem a first token also includes an authorisation by the first user (A) 5 for the issuer (I) 3 to use the first user private key (P1A) to redeem the first token.

[0165] The method 200 includes receiving 210, over the communications network 8, a request 610 from the first user (A) 5 to redeem the first token (T1). The method 200 includes determining 220 the first redeem script (RS1) associated with the first token (T1).

[0166] The method also includes the issuer (I) 3 receiving 235 the first user private key (V1A). In one example, this includes retrieving the first user private key (V1A) from the data store 11. It would be appreciated that the user private keys, which may be contained in the electronic wallet administered by the issuer (I) should be kept secure. In another alternative, the issuer (I) 3 may receive the first user private key (V1A) from another entity or node.

[0167] The issuer may then sign 245, with the user private key (P1A) and the first issuer private key (P1I) the first redeem script. This may be advantageous as the issuer (I) 3, who is the service provider for the first user (A) 5, can perform these steps securely at the first processing device 13 and without sending the first redeem script (RS1), signed or unsigned, over the communications network 8.

[0168] The method 200 also includes sending 260, over the communications network 8, a second data output (O2) to the peer-to-peer distributed ledger 9 comprising an indication of a transaction of the first quantity of cryptocurrency (B1) to the issuer (I).

[0169] Thus the method 200 returns the first quantity of cryptocurrency (B1) associated with the first token (T1) back to the issuer (I). In one example, since the first redeem script (RS1) is signed with the private keys of both the first user (A) 5 and the issuer (I), the recipient of the first quantity of cryptocurrency (B1) in this transaction, being the issuer (I) 3, may then spend the first quantity of cryptocurrency (B1) for other transactions – whether as cryptocurrency alone or with other associated tokens.

[0170] A specific example of a transaction to redeem the first token (T1) will now be described.

First user (A) 5 redeems the first token (T1) for \$1000 AUD from the issuer (I)

[0171] In this example, the first user (A) 5 wishes redeem the first token (T1) with the issuer (I) for the token value as shown in Fig. 2(b). This results in a transaction of the first quantity of cryptocurrency (B1) from the first user (A) 5 to the issuer (I), referred to as transaction ID-510 below. In return, the issuer (I) provides \$1000 AUD in fiat currency to the first user (A) 5.

[0172] In this example, the first token is redeemed by unlocking the first quantity of cryptocurrency (B1) and transferring them to the issuer (I) 3. The transfer of the first quantity of cryptocurrency (B1) back to the issuer (I) allows to issuer (I) 3 to then spend the first quantity of cryptocurrency (B1) for future transactions. The issuer (I) 3 may also “detokenize” the first quantity of cryptocurrency (B1) by one or more transactions that removes the metadata (which may include the redeeming transaction that transferred the first quantity of cryptocurrency (B1) back to the issuer (I) 3). The issuer (I) may further spend this cryptocurrency without the restriction of requiring authorisation (such as a signature) from the first user (A) 5 or other user.

[0173] Before describing the transaction to redeem the first token, ID-510, we will briefly describe the originating transaction outputs (from transaction ID-210 and ID-610) that are the

inputs to the present redemption transaction, ID-510. The two inputs in general, include the first quantity of cryptocurrency (B1) associated with the first token (T1), and another quantity of cryptocurrency which, at least in part, is used to pay the transaction fee (e.g. miner’s fee).

[0174] From the earlier example, the first user (A) 5 received the first quantity of cryptocurrency (B1) in transaction ID-210. The outputs that went to the first user (A) 5 in transaction ID-210 may be summarised as:

|                                          |
|------------------------------------------|
| ID-210                                   |
| 10,000,000                               |
| OP_HASH160 <redeem script hash> OP_EQUAL |

Table 5

[0175] The second line in Table 5 represents the first quantity of cryptocurrency (B1) associated with the first token (T1) which numbers in 10,000,000 satoshis. The third line represents the output script, which is equivalent to line 11 in Table 4 described above. From the earlier example, the transaction, ID-210, that created the first token (T1) had two outputs, but only the first output that corresponded to the first quantity of cryptocurrency (B1) are relevant to the redemption transaction ID-510. The second output in transaction ID-210 was change back to the issuer (I) shown in Table 4.

[0176] The issuer (I) also needs to pay the transaction fee (e.g. miner’s fee) for the redemption transaction, ID-510, which is paid in part from a quantity of cryptocurrency received from a previous transaction, ID-610. This quantity of cryptocurrency may be summarised as:

|                                                                 |
|-----------------------------------------------------------------|
| ID-610                                                          |
| 9,999,000                                                       |
| OP_DUP OP_HASH160 <PubK-Issuer hash> OP_EQUALVERIFY OP_CHECKSIG |

Table 6

[0177] The second line of Table 6 indicates the amount of cryptocurrency from previous transaction which is 9,999,000. The third line of Table 6 is the output script from this



[0181] The second input is shown at lines 9 to 13, which is previous transaction, ID-610, used to fund the present transaction, ID-510. The ScriptSig at line 12 requires signing with the first issuer private key (V1I) to the previous output script that included the first issuer public key (P1I).

[0182] The first output is shown at lines 15 to 17, that has an output of 10,000,000 satoshis. This corresponds to the first quantity of cryptocurrency (B1) from the first token (T1). The output script is at line 17, and the corresponding redeem script is:

```
1 metadata1 metadata2 PubK-Issuer 3 OP_CHECKMULTISIG
```

[0183] This redeem script includes the metadata from the first token as well as the issuer public key (P1I) shown as PubK-Issuer. This redeem script requires one of three signatures to spend the 10,000,000 satoshis. In practice, the first issuer private key (V1I) may be used to sign and spend the cryptocurrency for subsequent transactions. It is notable that the first user public key (P1A) is not in this redeem script. This is because this quantity of cryptocurrency has been redeemed with the issuer (I) and therefore may be considered as spent by the first user (A) 5. Accordingly, the issuer (I) should be free to spend this quantity of cryptocurrency without requiring authorisation (such as implicit authorisation through a signature of the first user (A) 5).

[0184] The issuer 3 may then conduct a further transaction to redeem the redeem script from output script (in line 17) using the issuer public key (P1I), whereby the further transaction results in an output script without the metadata.

[0185] Although the above mentioned first output retains metadata from the first token (T1) in the redeem script, it is to be appreciated that in some alternatives that this metadata need not be included in the first output as the first token (T1) has been redeemed and thus “untokenised”. That is, the first quantity of cryptocurrency (B1) may be disassociated, during a redeem transaction, with the first token (T1) by removing the corresponding first and/or second metadata (MD1/MD2). Furthermore, it is to be appreciated that the output script may be in other forms as specified by the issuer (I).

[0186] The second output is shown at lines 18 to 20, that has an output of 9,998,000 satoshis. This is in contrast to the input corresponding to transaction ID-610 that had 9,999,000 satoshis. The difference of 1,000 satoshis reflects the miners fee for this transaction.

#### **Fourth type of transaction – First user (A) redeems a first portion with issuer (I)**

##### Redeeming a portion of the value of the first token (T1)

[0187] In the above example, the first user (A) 5 redeemed the entire value of the first token (T1). However, in some examples, the first user (A) 5 may want to only redeem a portion of the value of the first token (T1).

[0188] Referring to Figs. 3(a) and 8, the first token (T1) has a token value that may include the sum of a first portion (R1) and a second portion (R2). Thus the first user (A) 5 may send 610, over the communications network 8, a request to redeem a value of the first portion (R1) of the first token (T1). In turn, the issuer (I) 3 receives 210, over the communications network 8, the request from the first user (A) 5 to redeem the first token (T1). The issuer may then perform steps 220, 230, 240 and 250 as described above for redeeming the first token (T1).

[0189] However, since the first user (A) 5 has made a request to redeem a value of the first portion (R1) of the total token value (T1), the value of the remaining second portion (R2) will need to be allocated to a second token (T2) back to the first user (A) 5. The second token will now be described with reference to Fig. 8.

[0190] The first user (A) 5 may send 645, over the communication network 8) the first user public key (P1A) to the issuer (I) 3 for creating the second token (T2). In turn, the method 200 then includes the issuer (I) determining 255 the first user public key (P1A) from the first user (A) 5. It is to be appreciated that the issuer (I) may already have the first user public key (P1A) from a previous transaction (or in the electronic wallet) and in such cases, it may not be necessary for the first user public key (P1A) to be sent again by the first user (A) 5. Instead, the first user public key (P1A) may be received from a data store 11 and/or a third party.

[0191] In yet another alternative, the first user (A) 5 may wish to use a different cryptographic pair for the second token (T2). Thus the step of sending 645 and determining 255 the first user public key may include a first user public key that is different to the one associated with the first token (T1).

[0192] The method 200 then includes allocating 265 a second quantity of cryptocurrency (B2) for association with a second token (T2), wherein the second token has a second token value (TV2) based on the second portion (R2). This step of allocating 265 a second quantity of cryptocurrency (B2) may include similar considerations as allocating a first quantity of cryptocurrency (B1) described above.

[0193] In some examples the pegging rate (PR2) of the second token (T2) is the same as the pegging rate (PR1) of the first token (T1), which may be desirable for the first user (A) 5 as the terms and conditions of the first (T1) and second (T2) token remain the same, with the exception of quantum of the token value.

[0194] In other examples, the second quantity of cryptocurrency (B2) may need to be at or above a minimum threshold of cryptocurrency (MT2) that is different to the minimum threshold of cryptocurrency (MT1) for the first token (T1). Thus allocating 265 the second quantity of cryptocurrency (B2) may include determining a minimum threshold of cryptocurrency (MT2) of the second token (T2) and determining a second quantity of cryptocurrency (B2) that is at or above the minimum threshold of cryptocurrency (MT2) of the second token (T2).

[0195] The method 200 further comprises determining 275 a second hash (H2) of a second redeem script (RS2), wherein the second redeem script (RS2) is based on: at least a second metadata that is based, at least in part, on the first metadata (MD2) associated with the first token (T1); the first user public key (P1A); and the first issuer public key (P1I) associated with the issuer (I).

[0196] The at least second metadata (MD2) may include, for example, association with one or more of the terms and conditions of the first token (T1). Therefore, the second token (T2) may have the same or similar characteristics as the first token (T1), albeit with a different token value. In some particular examples, the at least second metadata (MD2) of the second

token (T2) is the same as the at least first metadata (MD1) of the first token. In such examples, the second redeem script (RS2) of the second token (T2) is the same as the first redeem script (RS1) of the first token (T1). Accordingly the second hash (H2) associated with the second token (T2) is also the same as the first hash (H1) associated with the first token (T1). This may be advantageous as it may be easy to verify the second hash (H2) of the second token (T2) with by comparing it with the first hash (H1) of the first token (T1). This may also reduce the storage space, associated with storing the second hash (H2) (or subsequent hashes) as they will be the same.

[0197] As noted above, the first user public key (P1A) for the second token (T2) may, in some alternatives, be different to the first user public key associated with the first token (T1), and similarly, the first issuer public key (P1I) associated with the issuer (I) for the second token (T2) may also be different. For example, the issuer (I) and/or first user (A) 5 may wish to use different cryptographic pairs for security reasons.

[0198] In this example, the step of sending 260, over the communications network 8, a second data output (O2) to the peer-to-peer distributed ledger may further comprise an indication of a transaction of the second quantity of cryptocurrency (B2) to the first user (A) 5 and the second hash (H2), wherein the second hash (H2) is associated with the second quantity of cryptocurrency (B2), to provide the second token (T2) that is associated with the first user (A) 5 and the issuer (I). Thus the first user (A) 5 is provided with a second token (T2) that is based on the value of the second portion (R2) and, in some examples, carry similar characteristics to the first token (T1).

[0199] An example of redeeming a first portion is illustrated in Fig. 3(a) where the first user (A) 5 redeems the first token (T1) with the issuer (I) that includes a request to redeem the value of a first portion (R1) of the first token (T1), where the first portion is equivalent to \$500 AUD fiat currency. In return, the issuer (I) 3 provides \$500 AUD in fiat currency and a second quantity of cryptocurrency (B2) to provide a second token (T2) to the first user (A) 5. The second quantity of cryptocurrency (B2) is associated with the second token that may be representative of the value of the second portion (R2) of \$500 AUD.

**Third type of transaction – First user (A) transfers value to second user (B)**

Overview of a transfer of value from the first user (A) 5 to the second user (B)

[0200] The present disclosure also includes a method 300 of creating one or more additional tokens by the issuer (I) 3 as illustrated in Fig. 9. These additional tokens may be created as a result of, for example, the first user (A) 5 wishing to transfer the value, or portion thereof, the first token (T1) to the second user (B). This may be achieved by creating a third token (T3) that is associated with the second user (B) 7 and the issuer (I) 3.

[0201] This may advantageously allow the first user (A) 5 to, in effect, transfer the same or similar rights associated with a first token (T1), to the second user (B). Although a new token, in the form of a third token (T3), is created, the third token (T3) may have similar characteristics as the first token (T1). For example, the tokens may have associated metadata that is the same or similar. This may be useful, for example, if the same or similar terms and conditions applicable between the first user (A) 5 and the issuer (I) 3 should also apply between the second user (B) 7 and the issuer (I) 3.

[0202] In some circumstances, the first user (A) 5 may wish to transfer at least part of the value of the first token (T1) to the second user (B) as shown in Fig. 2(c). In one example, this is achieved by a transaction of the first quantity of cryptocurrency (B1) associated with the first token (T1) from the first user (A) 5 to the second user (B) 7. In a transaction where the entire value of the first token (T1) is transferred to the second user (B) 8, this may involve the creation of a third token (T3) with the first quantity of cryptocurrency B1, that is transferred to the second user (B) 7. Effectively, the third token (T3) is the transfer of the first token (T1), and rights associated with the first token (T1), to the second user (B) 7.

[0203] In this example, the transfer of value from the first user (A) 5 to the second user (B) involves the issuer (I) 3 as an intermediary to facilitate the transfer. This is distinguished from a direct transaction of the first quantity of cryptocurrency (B1) from the first user (A) 5 to the second user (B) 7. Involving the issuer (I) in this transfer of value may be advantageous for a number of reasons. Firstly, involving the issuer (I) reduces the risk of the first quantity of cryptocurrency (B1) from being transferred and spent by the first user (A) 5 as ordinary cryptocurrency as opposed to using the first quantity of cryptocurrency (B1) as a token. This is prevented by requiring the issuer (I) to sign the redeem script. Secondly, involving the issuer (I) may allow the issuer (I) 3 to keep track of the token(s) and particular

rights and/or liabilities associated with particular users. This may be useful for accounting, financial reporting and/or regulatory purposes.

[0204] An example of transferring the value will now be described in detail with reference to Figs. 2(c) and 9 where the methods 300, 700, 800 are performed by the issuer (I) 3, the first user (A) 5 and the second user (B) 7 respectively. The first user (A) 5 may send 710, over the communications network 8, a request to create the third token (T3), where the third token (T3) is associated with the first token (T1). In conjunction, or alternatively, the second user (B) 7 may send 810, over the communications network 8, a request to create the third token (T3). Whether these request are sent by one or both the first user (A) 5 and/or second user (B) 7 may depend on the terms and conditions of the first token (T1).

[0205] The issuer (I) then receives 310, over the communications network 8, the request to create the third token (T3). It is to be appreciated that the requests from the first user (A) 5 and second user (B) may be sent via another party in the communications network 8. In addition, the request may be in piecemeal, with part of the request coming from the first user (A) 5 and the other part of the request from the second user (B) 7.

[0206] The method 300 then includes determining 320 the first redeem script (RS1) associated with the first token (T1).

[0207] The method 300 also includes receiving 335 the first user private key (V1A). In one example, this includes retrieving the first user private key (V1A) from the data store 11. The method further includes the issuer (I) 3 signing 345, with the user private key (P1A) and the first issuer private key (P1I) the first redeem script. The steps 335 and 345 are analogous to the steps 235 and 245 in method 200 described above for redeeming the first token (T1) and similar considerations may also apply.

[0208] To create the third token (T3) the second user public key (P1B) is required. This second user public key (P1B) is a cryptographic pair with a second user private key (V1B). The issuer (I) 3 may determines 360 the second user public key (P1B) in a number of ways. Firstly, the issuer (I) 3 may be a service provider of the second user (B) 7 and the second user public key (P1B) may be stored in a data store 11 of the issuer (I). Alternatively, the second user public key (P1B) may have been received by the issuer (I) during an earlier transaction

and accordingly, the second user public key (P1B) may, in some cases, be retrieved from the data store 11 of the issuer (I). In some other alternatives, the second user public key (P1B) may be received via a third party in the communications network 8. In yet another alternative, the second user (B) 7 may send 820, over the communications network 8, the second user public key (P1B) to the issuer (I) 3.

[0209] The method 300 further includes allocating 370 a third quantity of cryptocurrency (B3) for association with the third token (T3). In some examples where the total value of the first token (T1) is to be transferred to the second user (B), it may be appropriate for the third quantity of cryptocurrency (B3) to be allocated from, and be the same as, the first quantity of cryptocurrency (B1). In other alternatives (such as the fifth type of transaction which will be discussed in further detail below), only a portion of the total value of the first token (T1) is to be transferred to the second user (B) and a corresponding proportion may be allocated for the third quantity of cryptocurrency (B3). In yet further examples, the third quantity of cryptocurrency (B3) may be allocated from other cryptocurrency that is not associated with the first quantity of cryptocurrency (B1). It is to be appreciated that the considerations for allocating 370 the third quantity of cryptocurrency (B3) may be the same or similar to those when allocating 130 the first quantity of cryptocurrency (B1) in method 100 and allocating 265 the second quantity of cryptocurrency (B2) in method 200.

[0210] The method further includes determining 380 a third hash (H3) of a third redeem script (RS3), wherein the third redeem script (RS3) is based on: at least a third metadata (MD3) that is based, at least in part, on the first metadata (MD1) associated with the first token; the second user public key (P1B); and the first issuer public key (P1I). This may include similar or the same considerations as determining 140 a first hash (H1) of a first redeem script (RS1) in method 100 or determining 275 a second hash (H2) of a second redeem script (RS2) in method 200.

[0211] The method 300 further includes sending 390, over the communications network, a third data output (O3) to the peer-to-peer distributed ledger comprising: an indication of a transaction of at least the third quantity of cryptocurrency (B3) to the second user (B); and the third hash (H3), wherein the third hash (H3) is associated with the third quantity of cryptocurrency (B3) to provide the third token (T3) that is associated with the second user (B)

7 and the issuer (I). This is similar to steps 150 and 260 described above and like variations and alternatives may be applicable.

**Fifth type of transaction – First user (A) transfers a first portion to second user (B)**

[0212] In a further example, only a first portion (R1) of the total value of the first token (T1) is transferred to the second user (B) 7, and in such cases, the remaining second portion (R2) of the total value may be included in a second token (T2) that is refunded back to the first user (A) 5. This may be similar to refunding a second portion (R2) of the value described above in method 200. Thus the request to create the third token (T3) may comprise, explicitly or implicitly, a request to create the third token (T3) with a third token value (TV3) based on the first portion (R1).

[0213] The refund back to the first user (A) 5, in the form of the second token (T2), will be described with reference to Figs. 3(b) and 10. To create the second token (T2), the method 300 includes determining 355 the first user public key (P1A). This may be achieved in a number of ways as described above and may include receiving the first user public key (P1A) that is sent 745, over the communications network 8, from the first user (A) 5.

[0214] The method further includes allocating 365 a second quantity of cryptocurrency (B2) for association with the second token (T2), wherein the second token has a second token value (TV2) based on the second portion (R2). The method 300 also includes determining 375 a second hash (H2) of a second redeem script (RS2) that is based on: the second metadata (MD2) that is based, at least in part, on the first metadata (MD1) associated with the first token (T1); the first user public key (P1A); and the first issuer public key (P1I) associated with the issuer (I) 3. Thus the step of sending 390 the third data output (O3) to the peer-to-peer distributed ledger further includes: an indication of a transaction of at least the second quantity of cryptocurrency (B2) to the first user (A) 5; and the second hash (H2), wherein the second hash (H2) is associated with the second quantity of cryptocurrency (B2), to provide the second token (T2) that is associated with the first user (A) 5 and the issuer (I) 3.

An example of transferring value from the first user (A) 5 to the second user (B)

[0215] A specific example of a transaction, ID-110, will now be described. Referring to Fig. 3(b), the first user (A) 5 has tokens with a total value of \$10.00 AUD. The first user (A) 5 wishes to transfer a first portion (R1) of \$7.30 AUD to the second user (B) as a third token (T3) and have the remaining second portion (R2) of \$2.70 provided as change in the form of second token (T2) back to the first user (A) 5.

[0216] In this example, the first token (T1) includes two blocks of tokens, with each block representing a value of \$5.00 AUD. This may be representative of tokens that have standardised values, (e.g. in \$5.00 blocks) or representative of the first user (A) 5 having acquired the two blocks from different transactions. Each of these blocks include 50,000 satoshis which, at a pegging rate of 100 satoshis /cent is equivalent to \$5.00 AUD. This is illustrated below in Table 8 as transactions ID-101 and ID-102, which are transactions that created the first token (T1).

|                                          |
|------------------------------------------|
| ID-101                                   |
| 50,000                                   |
| OP_HASH160 <redeem script hash> OP_EQUAL |

|                                          |
|------------------------------------------|
| ID-102                                   |
| 50,000                                   |
| OP_HASH160 <redeem script hash> OP_EQUAL |

|                                                                 |
|-----------------------------------------------------------------|
| ID-103                                                          |
| 10,000,000                                                      |
| OP_DUP OP_HASH160 <PubK-Issuer hash> OP_EQUALVERIFY OP_CHECKSIG |

Table 8

[0217] Line 3 of both ID-101 and ID-102 represents the output script of the respective transactions, which would be similar to line 11 in Table 4 described above.

[0218] The issuer (I) also needs to pay a transaction fee (miner's fee) for this transaction. This transaction fee may be paid in part from a quantity of cryptocurrency received from a previous transaction, ID-103, as shown in Table 8. This shows a previous transaction of 10,000,000 satoshis which will, in part be used to fund the transaction. This is similar to the previous transaction ID-610 described above with reference to Table 6.

[0219] The transaction, ID-110, to transfer the value to the second user (B) will now be discussed with reference to Table 9 below.

|                                                                    |                         |
|--------------------------------------------------------------------|-------------------------|
| ID-110                                                             | Transaction-ID          |
| Version number                                                     | Version number          |
| 3                                                                  | Number of inputs        |
| ID-101                                                             | Prev Trans Output       |
| IDX-00                                                             | Prev Trans Output index |
| Script length                                                      | Script length           |
| Sig-V1A Sig-V1I <2 metadata1 metadata2 P1A P1I 4 OP_CHECKMULTISIG> | ScriptSig               |
| 00000000000000000000000000000001                                   | Sequence number         |
| ID-102                                                             | Prev Trans Output       |
| IDX-00                                                             | Prev Trans Output index |
| Script length                                                      | Script length           |
| Sig-V1A Sig-V1I <2 metadata1 metadata2 P1A P1I 4 OP_CHECKMULTISIG> | ScriptSig               |
| 00000000000000000000000000000011                                   | Sequence number         |
| ID-103                                                             | Prev Trans Output       |
| IDX-00                                                             | Prev Trans Output index |
| Script length                                                      | Script length           |
| Sig-issuer PubK-issuer                                             | ScriptSig               |
| 00000000000000000000000000000100                                   | Sequence number         |
| 3                                                                  | Number of outputs       |
| 73,000                                                             | Output value            |
| Output script length                                               | Output script length    |
| OP_HASH160 <redeem script hash> OP_EQUAL                           | Output script           |
| 27,000                                                             | Output value            |
| Output script length                                               | Output script length    |
| OP_HASH160 <redeem script hash> OP_EQUAL                           | Output script           |
| 9,999,000                                                          | Output value            |
| Output script length                                               | Output script length    |
| OP_DUP OP_HASH160 <PubK-issuer hash> OP_EQUALVERIFY OP_CHECKSIG    | Output script           |
| LockTime                                                           | LockTime                |

Table 9

[0220] The third line of Table 9 indicates there are three inputs, and line 19 indicates there are three outputs. Two of the inputs represent the first token (T1) and the third input is for paying the transaction fee. The first output is representative of the transfer of value to the second user (B) 7, the second output representative of the change of tokens back to the first user (A) 5, and third output is the change of cryptocurrency back to the issuer (I).

[0221] The first input, based on previous transaction ID-101, is shown at lines 4 to 8, which is the input of a first block of 50,000,000 satoshis, being half of the first quantity of cryptocurrency (B1) and represent \$5.00AUD in value. Line 7 shows the ScriptSig to allow spending of this quantity of cryptocurrency. This shows that the first redeem script (RS1) requires a two of four signatures, and in particular the signing with the first user private key

(V1A) and the first issuer private key (V1I). Line 8 indicates a sequence number where this first input is marked to the first output.

[0222] The second input, based on previous transaction ID-102, is shown at lines 9 to 13, which is the input of a second block of 50,000,000 satoshis, being the second half of the first quantity of cryptocurrency (B1) and represents \$5.00AUD in value. Line 12 shows a ScriptSig similar to line 7 above. Line 12 indicates a sequence number where this second input is marked to output to both the first output and the second output. This is because this second block of 50,000,000 satoshis will be split, with 23,000 satoshis to the first output and 27,000 satoshis to the second output.

[0223] The third input is shown at lines 14 to 18, which is based on previous transaction ID-103 that is used to fund the present transaction ID-110. The ScriptSig at line 17 requires signing with the first issuer private key (V1I) to the previous output script that included the first issuer public key (P1I).

[0224] The first output is shown at lines 20 to 22 and has an output of 73,000 satoshis which is the third quantity of cryptocurrency (B3) for the third token (T3). In this example, the pegging rate for the third token (T3) is 100 satoshis / cent (which is the same pegging rate as the first token (T1)) and therefore the third quantity of cryptocurrency (B3) has a third token value (TV3) of \$7.30AUD, which is based on the first portion (R1) of \$7.30 AUD.

[0225] The output script is at line 22, and the corresponding redeem script for this example is:

2 metadata1 metadata2 P1B P1I 4 OP\_CHECKMULTISIG

[0226] This includes the second user public key (P1B) and the first issuer public key (P1I) as described above. Importantly, the second user public key (P1B) is used since the third token (T3) is for the second user (B) 7 to redeem. The metadata 1 and metadata2 may include metadata as described above, including an indication that this is a “payment” or “transfer” transaction between users. Thus the first output provides the third token (T3) that can be redeemed by the second user (B) 7 for a value of \$7.30 AUD with the issuer (I) 3.

[0227] The second output is shown at lines 23 to 24 and has an output of 27,000 satoshis which is the second quantity of cryptocurrency (B2) for the second token (T2) back to the first user (A) 5. In this example, the same pegging rate of 100 satoshis / cent and therefore the second quantity of cryptocurrency (B2) has a second token value (TV3) of \$2.70 AUD, which is based on the remaining second portion (R2) of \$2.70 AUD. The output script is at line 25, and the corresponding redeem script for this example is:

```
2 metadata1 metadata2 P1A P1I 4 OP_CHECKMULTISIG
```

[0228] This includes the first user public key (P1A) and the first issuer public key (P1I). Importantly, the first user public key (P1A) is used since the second token (T2) is for the first user (A) 5 to redeem. The metadata may also include an indication that this is part of a “payment” or “transfer” transaction between users.

[0229] The third output is shown at lines 26 to 28 and reflects the issuer’s change for the transaction. In the present transaction, the transaction fee is 1,000 satoshis and accordingly, the issuer (I) can expect to have change from the third input of 10,000,000 satoshis. Line 26 is an output value for the third output which is 9,999,000. Since the third output is the change back to the issuer (I), the issuer should be free to spend the third output. Accordingly, the output script at line 28 only includes the first issuer public key (P1I) which is represented by <PubK-Issuer hash>.

[0230] The above example shows that a single transaction may have a mixture of cryptocurrency that is “tokenised” and “non-tokenised”. In one example, it may be important to verify that the input token values is equal to the output token values. Thus the issuer (I) may verify that a first token value (TV1) of the first token (T1) is equal to the sum of the second token value (TV2) and the third token value (TV3).

[0231] In the above example, the transaction fee is paid by the issuer (I), who may pass on these costs by other means. It is to be appreciated that in some alternatives, the transaction fee may be paid directly by the first user and/or second user. For example, the first user may be required to contribute cryptocurrency to be used for payment of the transaction fee. In another examples, a portion of the first, second or third quantity of cryptocurrency may be used in each transaction to pay the transaction fee. In yet another alternative, each transaction

may include an additional output that creates an additional token for the miner facilitating the transaction to redeem with the issuer (I) 3.

**Variation – The users sign the redeem script with respective private keys**

[0232] In the above examples, the issuer (I) are the service providers for the first user (A) 5 and second user (B) 7 and manage respective electronic wallets. Accordingly, the issuer (I) 3 can access, with authorisation from the users, the respective user's private keys. This includes retrieving the user private keys from a data store 11.

[0233] In some alternative examples, it may be desirable from the users to keep the private keys themselves. This separation may allow the users to have greater control over their private keys. It may also be more secure, since a person having access to all the information in the data store 11 of the issuer (I) 3, including the first issuer private key (VI1), will not be able to unlock the redeem scripts as they would not have the respective user's private keys.

[0234] Accordingly, one variation of the method may include sending the redeem scripts to the users 5, 7 for signature with their respective private keys. In such examples, the issuer (I) 3 does not need to have possession of the user's private keys.

[0235] Such a variation of methods 200', 600' of redeeming the first token (T1) will now be described with reference to Fig. 11.

[0236] The first user (A) 5 may send 610, over the communications network 8, a request to redeem the first token (T1). In turn, the issuer (I) 3 receives 210, over the communications network 8, the request from the first user (A) 5 to redeem the first token (T1). The method 200 then includes determining 220 the first redeem script (RS1) associated with the first token (T1). In one example, this may include retrieving the first redeem script (RS1) from the data store 11. In another example, this may include recreating the first redeem script (RS1) with data from one or more sources. For example, this may include retrieving the at least first metadata (MD1) and the first issuer public key (PI1) from the data store 11 and receiving, over the communications network 8, the first user public key (P1A). This data may then be combined to recreate the first redeem script (RS1).

[0237] The method 200' then includes sending 230, over the communications network 8, the first redeem script (RS1) for signing by the first user (A) 5. In turn, the first user (A) 5 receives 620 the first redeem script (RS1). It is to be appreciated that in some alternative examples, the step of sending the first redeem script (RS1) to the first user (A) 5 may be performed at other times. For example, the first redeem script (RS1) may be sent to the first user (A) 5 during or after the issuer (I) 3 creates the first token (T1). In another alternative, the first redeem script (RS1) may be retrieved by the first user (A) 5 from a data store. In yet other alternatives, the first user (A) 5 may determine the first redeem script (RS1) associated with the first token (T1) independently. For example, the first user (A) 5 may retrieve the first metadata (MD1), first issuer public key (P1I) and first user public key (P1A) from one or more sources to determine the first redeem script (RS1).

[0238] The first user (A) 5 then signs 630, with the first user private key (V1A), the first redeem script (RS1) to provide a first redeem script signed by the first user (RS1A). The first redeem script signed by the first user (RS1A) is then sent 640, over the communications network 8, from the first user (A) 5 to the issuer (I) 3.

[0239] In turn, the method 200' includes receiving 240, over the communications network 8, the first redeem script signed by the first user (RS1A). The method 200 further includes signing 250, with the first issuer private key (V1I), the first redeem script signed by the first user (RS1A) to unlock the first quantity of cryptocurrency (B1) associated with the first token (T1).

[0240] The method 200' further includes the step of sending 260, over the communications network 8, a second data output (O2) to the peer-to-peer distributed ledger comprising an indication of a transaction of the first quantity of cryptocurrency (B1) to the issuer (I). In one example, the first quantity of cryptocurrency (B1) is transferred back to the issuer (I) and may no longer be associated with the first token (T1), since the first token (T1) has been redeemed. In some cases, the metadata associated with the first quantity of cryptocurrency (B1) may be removed to "untokenise" the cryptocurrency. This may be done in the same transaction or in a subsequent transaction, which may be a choice for the issuer (I) 3.

[0241] Notably, the above method 200' requires both the first user (A) 5 and the issuer (I) to sign the first redeem script (RS1). This may be advantageous to prevent, or reduce the risk, of

accidental or inadvertent spending of the first quantity of cryptocurrency (B1) by the first user (A) 5 beyond the intended token purpose. For example, if the first user (A) 5 attempted to spend the first quantity of cryptocurrency (B1) with another user (other than the issuer (I)), such a transaction will not proceed since the first issuer private key (V1I) is required to unlock the first quantity of cryptocurrency (B1). On the other hand, requiring the first user (A) 5 to sign the first redeem script (RS1) provides a degree of security for redeeming of the first quantity of cryptocurrency (B1), as the first user (A) 5 may control the first user private key (V1A) and selectively use it for authorised transactions.

[0242] In addition, having the issuer (I) sign the redeem script last can improve security as it may avoid sending a completely signed redeem script over a communications network that may not be secure. For example, when redeeming the first token (T1) the order of the public keys may dictate that the first user (A) 5 signs the first redeem script (RS1), after which it is sent to the issuer (I) for the final signature. Since the issuer (I) provides the final signature for unlocking, this lowers the risks that a person intercepting communications between the issuer (I) and the first user (A) 5 can fraudulently access the token (T1) and/or the first quantity of cryptocurrency (B1).

[0243] Similar steps may also be used when transferring the value of the first token (T1) from the first user (A) 5 to the second user (7) as shown in the methods 300', 700', 800' illustrated Fig. 12. The methods 300', 700', 800' include similar steps to those described in methods 300, 700, 800 described above with reference to Fig. 9, but with the following exceptions. Instead of the issuer (I) 3 receiving 335 the first user private key (V1A) and using it to sign the first redeem script (RS1), this is done by the first user (A) 5. Thus the method 300' includes the issuer (I) 3 sending 330, over the communications network 8, the first redeem script (RS1) for signing by the first user (A) 5.

[0244] The first user (A) 5 receives 720 the first redeem script and signs 730 the first redeem script with the first user private key (V1A). This provides the first redeem script signed by the first user (RS1A) which is then sent 740, over the communications network 8, to the issuer (I) 3.

[0245] The method 300' then includes receiving 340, over the communications network 8, a first redeem script signed by the first user (RS1A). This is followed by signing 350, with the

first issuer private key (V1I), the first redeem script signed by the first user (RS1A) to unlock the first quantity of cryptocurrency (B1) associated with the first token (T1).

[0246] The method 300' may further include steps 360, 370, 380 and 390 to complete creation of the third token (T3) in a similar manner to the method 300 described above.

#### **[0247] Tokens and codification process**

A contract is transferable if the rights it defines are conferred upon the holder or owner of the contract. An example of a non-transferable contract is one in which the participants are named - that is, where the rights are conferred upon a specific named entity rather than the holder of the contract. Only transferable contracts are discussed herein.

A token represents a specific contract that details or defines rights conferred by a contract. The actual contract may be a file stored in a distributed manner. For example, it may be stored in the cloud. In a preferred embodiment, the token is a representation of the contract in the form of a bitcoin transaction.

A divisible token is one in which the value on a transaction output may be subdivided into smaller amounts allocated across multiple tokens (i.e. allocated across multiple transactions). The archetype is tokenised fiat currency. Divisible contracts are defined as those that specify a non-zero PeggingRate. For divisible contracts the tokenised value transferred in the transaction output is tied to the underlying bitcoin (BTC) value via the PeggingRate. That is, the contract specifies the holder's rights in terms of a pegging-rate. For non-divisible tokens there is no PeggingRate and the contract specifies the holder's rights in terms of a fixed value (e.g. like a bearer bond: 'this contract is redeemable for exactly \$1000' or a voucher 'this contract is redeemable for one haircut'). For non-divisible contracts the underlying transaction BTC value is irrelevant to the contract value.

The phrase "Underlying BTC value" refers to the bitcoin amount (BTC) attached to the transaction output. In the Bitcoin protocol every transaction output must have non-zero BTC amount to be considered valid. In fact, the BTC amount must be greater than a set minimum (known as 'dust') which, at the time of writing, is currently set to 546 satoshis. 1 bitcoin is defined as being equal to 100 million satoshis. As the bitcoin transactions are here used only as a means of facilitating an exchange of ownership, the actual underlying BTC amount is arbitrary: the true value lies in the contract specification. In theory every token could be carried by dust.

In the protocol of the present invention, specifically for divisible tokens, the underlying BTC value does have a meaning: it bears a relationship to the contract value via a PeggingRate. The PeggingRate is itself arbitrary and is chosen so as to keep the underlying BTC amount small. The reason for using a PeggingRate rather than simply underlying every token transaction with dust is because the protocol of the present invention facilitates divisibility: when a token is split into several transaction outputs of smaller amounts it is not necessary to adjust the original contract. Rather, the contract value of each subdivided token is simply calculated based on the PeggingRate and the subdivided amount of underlying BTC value.

A limited token is one in which a total issuance value is fixed (or 'limited') by a fixed non-zero number of shares as defined by a quantity called NumShares. Therefore, no further shares may be issued under a limited contract. For example a contract for part ownership of a race horse is limited to 100% of the race horse (e.g. 100 shares at 1% each or 10 shares at 10% each, etc.). An unlimited contract implies that the issuer is able to underwrite further issuances of shares, for example by adding the required amount of fiat currency into their Reserve Account. NumShares must be explicitly stated on all contracts. Limited contracts must have  $\text{NumShares} > 0$ ; unlimited contracts are denoted by setting  $\text{NumShares} = 0$ .

The archetypical example is a currency reserve (analogous to a gold reserve) such that the total value held in the reserve bank account matches the total value in promissory notes in existence (i.e. unredeemed tokens). This concept extends beyond currency reserves to include stock inventory. For example, an issuer of licensed printed t-shirt tokens may start with an inventory of 10,000 T-shirts in stock and may issue a divisible token to represent those 10,000 t-shirts (where, say, each share = 1 t-shirt). The original token could be subdivided and each subdivided token would be redeemable for a number of t-shirts according to the transaction output's underlying BTC value as defined by the PeggingRate. If demand increases, however, the issuer may decide to issue further shares (i.e. increase the number of shares in circulation by (say) another 10,000). In such cases it is incumbent on the issuer to deposit a further 10,000 t-shirts into his reserve account (i.e. stock warehouse) in order to underwrite the further issuance. Thus, the total number of t-shirts in stock (where stock acts as 'reserve account') at any one time = the total number of unredeemed shares.

PeggingRates only apply to divisible contracts, wherein the value of a share (represented by a quantity called ShareVal) is pegged to the underlying BTC amount. For example, the contract might specify that the issuer promises to redeem the token at a rate of \$10,000 for every underlying 1 BTC. That would mean (for example) that a transaction with a tokenised underlying output value of 15,400 satoshis would be redeemable for \$1.54. A value of 0 for the PeggingRate indicates that the contract is non-divisible (i.e. can only be transferred whole, like a bearer bond). When the PeggingRate is set to 0 (meaning non-

divisible token) the underlying BTC value is not relevant to the contract value and can be set at any amount. Normally in this case it is desirable to keep the underlying BTC amount as small as possible (i.e. set to dust) to minimise operating costs.

NumShares is the total (fixed) number of shares available under the (Limited) contract. For limited contracts NumShares must be a whole number greater than zero. For unlimited contracts NumShares is not fixed as more shares can be issued at any time (provided they are underwritten), which is denoted by setting the value to 0.

A share is defined as the unit of transfer and the ShareVal is the value of that unit. For example, for fiat currency, the unit of transfer may be set to 1 cent. Or, for example, it may be set to 50 cents, in which case transfers may only be executed in 'lots' of 50 cents. ShareVal may also be expressed as a percentage: for example if a breeder wants to sell a racehorse in 10 equal shares then the ShareVal = 10%. ShareVal must be > 0 and must be defined on the contract.

TotalIssuance represents the total value of shares issued. This value only relates to limited contracts as for unlimited contracts the issuance is not fixed and more shares may be issued.

If the shares are expressed as a percentage then the TotalIssuance = 100% by definition.

For limited contracts NumShares, ShareVal, and TotalIssuance are related in the following way:

$$\text{NumShares} \times \text{ShareVal} = \text{TotalIssuance}.$$

A value of 0 for TotalIssuance implies it is an unlimited contract. An example of an unlimited contract is fiat currency (so TotalIssuance is set to 0); examples of limited contracts are: (i) limited edition commemorative coins (1000 minted, where 1 share = 1 coin): TotalIssuance = 1000 x 1 = 1000 coins; and (ii) seats at a ticketed venue, where TotalIssuance = total number of seats available.

The circulation is defined as the total value of unspent tokens (i.e. as determined by transactions in UTXO – unspent transaction output). The full set of all unspent transactions is kept in a list available to all bitcoin nodes. For example, if an issuer initially issues \$10,000 as fiat currency type tokens and over time \$5500 worth of tokens are redeemed, then the circulation = \$4500 (being the value of unredeemed tokens). This value should reconcile to the balance in the associated reserve account.

It should be noted that in some (atypical) situations the circulation might fall below the reserve account balance, though never the reverse. For example, consider a breeder issuing 10 shares in a race horse (where by definition TotalIssuance = 100%). A buyer may redeem their token by sending it back to the

breeder, and if she detokenises it there will only be 9 shares in circulation = 90% of the horse, although of course the whole 100% horse is in the reserve (= stable). In this situation the reserve excess (i.e. the unaccounted 10% ownership) implicitly belongs to the issuer.

Although this situation is benign and falls within the scope of the invention, a protocol may be enforced wherein 100% of shares must be explicitly accounted for (i.e. the breeder in this illustrative situation is not allowed to detokenise the token).

Example 1 – Firewood by weight. In this example, a contract reads: “The holder is entitled to firewood at a rate of 20kg for every underlying 600 satoshi.” The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = 20kg; and PeggingRate = 600 satoshis/share. These parameters define an unlimited and divisible contract, wherein a share in the contract has a value of 20kg of firewood, and wherein each multiple of 600 satoshis within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed.

Example 2 – Firewood by bag. In this example, the contract reads: “The holder is entitled to one 20kg bag of firewood.” The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = 1 bag; and PeggingRate = 0. These parameters define an unlimited and indivisible contract, wherein a share in the contract has a value of one 20kg bag of firewood, and wherein any amount of underlying bitcoin within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed.

Example 3 – A \$1000 note. In this example, the contract reads: “The holder is entitled to \$1000 exactly”. The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = \$1000; PeggingRate = 0. These parameters define an unlimited and indivisible contract, wherein a share in the contract has a value of \$1000, and wherein any amount of underlying bitcoin within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed.

Example 4 – Commemorative coins #1. In this example, the contract reads: “The holder is entitled to a limited edition (1000 coins) Year 2000 Olympic Silver coin (maximum one per customer)”. The metadata is defined to represent the following critical parameters: NumShares = 1000; ShareVal = 1 coin; PeggingRate = 0. These parameters define an indivisible contract limited to 1000 shares, wherein a share in the contract has a value of 1 coin, and wherein any amount of underlying bitcoin within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is 1000 coins.

Example 5 – Commemorative coins #2. In this example, the contract reads: “The holder is entitled to limited edition (10,000 coins) Year 2000 Olympic Bronze coins at a rate of 1 coin for every underlying

600 satoshis". The metadata is defined to represent the following critical parameters: NumShares = 10,000; ShareVal = 1 coin; PeggingRate = 600 satoshis/share. These parameters define a divisible contract limited to 10,000 shares, wherein a share in the contract has a value of 1 coin, and wherein each multiple of 600 satoshis within the transaction corresponds to one share in the contract. In this embodiment, the TotalIssuance is 10,000 coins.

Example 6 – Fiat currency #1. In this example, the contract reads: "The holder is entitled to Canadian dollars at a rate of \$10,000 for every underlying bitcoin. Unit of transfer is 50 cents". The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = 50 cents; PeggingRate = 5000 satoshis/share. These parameters define an unlimited and divisible contract, wherein a share in the contract has a value of 50 Canadian cents, and wherein each multiple of 5000 satoshis within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed.

Example 7 – Fiat currency #2. In this example, the contract reads: "The holder is entitled to Australian dollars AUD at a rate of \$10,000 for every underlying bitcoin. Unit of transfer is 1 cent". The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = 1 cent; PeggingRate = 100 satoshis/share. These parameters define an unlimited and divisible contract, wherein a share in the contract has a value of 1 Australian cent, and wherein each multiple of 100 satoshis within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed. Incidentally, it will be seen that the minimum AUD that can actually be transferred in this example is 6 cents. Any less would result in an underlying BTC value lower than the current minimum required for a valid transaction.

Example 8 – Shared house. In this example, the contract reads: "The holder is entitled to part ownership of the property at (address) at a rate of 10% for every underlying 600 satoshis". The metadata is defined to represent the following critical parameters: NumShares = 10; ShareVal = 10%; PeggingRate = 600 satoshis/share. These parameters define a divisible contract limited to ten shares, wherein a share in the contract has a value of 10%, and wherein each multiple of 600 satoshis within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is 100% ownership of a house.

Example 9 – A race horse. In this example, the contract reads: "The holder is entitled to part ownership of 'Naka's Delight' at a rate of 1% for every underlying 600 satoshis". The metadata is defined to represent the following critical parameters: NumShares = 100; ShareVal = 1%; PeggingRate = 600 satoshis/share. These parameters define a divisible contract limited to one hundred shares, wherein a share in the contract has a value of 1%, and wherein each multiple of 600 satoshis within the transaction

corresponds to one share in the contract. In this example, the TotalIssuance is 100% ownership of a horse.

Example 10 – Allocated seat ticket. In this example, the contract reads: “The holder is entitled to seat B54 at ‘Dead Lizard’ Concert on 14 February 2016 at the Central Concert Hall”. The metadata is defined to represent the following critical parameters: NumShares = 1; ShareVal = 1 ticket; PeggingRate = 0. These parameters define an indivisible contract limited to one share, wherein a share in the contract has a value of 1 ticket, and wherein any amount of underlying bitcoin within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is 1 ticket. The ticket may include an access code to a barrier on entry into the event venue, thereby providing feedback that the ticket has been redeemed.

Example 11 – A voucher for a celebrity date. In this example, the contract reads: “The holder is entitled to a one-off prize dinner date on 31 March 2016 with George Kludgy at the Spiffy Hotel in central Sydney including a taxi ride home”. The metadata is defined to represent the following critical parameters: NumShares = 1; ShareVal = 1 date; PeggingRate = 0. These parameters define an indivisible contract limited to one share, wherein a share in the contract has a value of 1 date, and wherein any amount of underlying bitcoin within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is 1 date.

Example 12 – Vouchers for haircuts. In this example, the contract reads: “The holder is entitled to one haircut and blowdry, valid any weekday except public holidays”. The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = 1 voucher; PeggingRate = 0. These parameters define an unlimited and indivisible contract, wherein a share in the contract has a value of 1 voucher, and wherein any amount of underlying bitcoin within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed.

Example 13 – T-shirts. In this example, the contract reads: “The holder is entitled to ‘Dead Lizard’ souvenir t-shirts for the 2016 World Tour at a rate of 1 t-shirt for every 1000 satoshis”. The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = 1 t-shirt; PeggingRate = 1000. These parameters define an unlimited and divisible contract, wherein a share in the contract has a value of 1 t-shirt, and wherein each multiple of 1000 satoshis within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed.

Example 14 – Non-allocated seat ticket. In this example, the contract reads: “The holder is entitled to entry to The Jazz Jivers concert at Sadie’s Pub on 29 April 2016 at a rate of 1 entry ticket for every underlying 1000 satoshis. Only 137 spaces are available”. The metadata is defined to represent the

following critical parameters: NumShares = 137; ShareVal = 1 ticket; PeggingRate = 1000. These parameters define a divisible contract limited to 137 shares, wherein a share in the contract has a value of 1 ticket, and wherein each multiple of 1000 satoshis within the transaction corresponds to one share in the contract. In this embodiment, the TotalIssuance is 137 tickets.

Example 15 – A music file. In this example, the contract reads: “The holder is entitled to one copy of The Dead Lizard’s album ‘Chameleon Rising’”. The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = 1 album; PeggingRate = 0. These parameters define an unlimited and indivisible contract, wherein a share in the contract has a value of 1 file corresponding to or containing the album, and wherein any amount of underlying bitcoin within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed.

Example 16 – Item of furniture from a catalogue. In this example, the contract reads: “The holder is entitled to this stunning and unique antique Georgian escritoire in excellent condition”. The metadata is defined to represent the following critical parameters: NumShares = 1; ShareVal = 1 item; PeggingRate = 0. These parameters define an indivisible contract limited to one share, wherein a share in the contract has a value of 1 item, and wherein any amount of underlying bitcoin within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is 1 item.

Example 17 – Golf balls by the batch. In this example, the contract reads: “The holder is entitled to premium quality Tigger Wodes class ‘A’ golf balls at a rate of 12 balls for every underlying 600 satoshis”. The metadata is defined to represent the following critical parameters: NumShares = 0; ShareVal = 12 golf balls; PeggingRate = 600. These parameters define an unlimited and divisible contract, wherein a share in the contract has a value of 12 golf balls, and wherein each multiple of 600 satoshis within the transaction corresponds to one share in the contract. In this example, the TotalIssuance is not fixed.

### **Processing device**

[0248] As noted above, the issuer (I) 3, first user (A) 5 and second user (B) 7 may be associated with a first processing device 13, second processing device 15, and a third processing device 17. The peer-to-peer distributed ledger 9 may also be associated with multiple processing devices 19.

[0249] Such a processing device may be part of an electronic device, such as a computer, tablet computer, mobile communication device, computer server etc. In addition to a processing device, the electronic device may include a data store 11 and a user interface.

[0250] Fig. 13 illustrates an example of a processing device 13, 15, 17, 19. The processing device 13, 15, 17, 19 includes a processor 1510, a memory 1520 and an interface device 1540 that communicate with each other via a bus 1530. The memory 1520 stores instructions and data for implementing the method 100, 200, 300, 400, 500, 600, 700, 800 described above, and the processor 1510 performs the instructions from the memory 1520 to implement the methods. The interface device 1540, may include a communications module that facilitates communication with the communications network 5 and, in some examples, with the user interface and peripherals such as data store 11. It should be noted that although the processing device 1501 may be independent network elements, the processing device may also be part of another network element. Further, some functions performed by the processing device may be distributed between multiple network elements. For example, the issuer 3 may have multiple processing devices 23 to perform method 100, 200, 300, 400 in a secure local area network associated with the issuer (I) 3.

[0251] Where this disclosure describes that a user, issuer, merchant, provider or other entity performs a particular action (including signing, issuing, determining, calculating, sending, receiving, creating etc.), this wording is used for the sake of clarity of presentation. It should be understood that these actions are performed by the computing devices operated by these entities.

[0252] Signing may comprise executing a cryptographic function. The cryptographic function has an input for a clear text and an input for a key, such as a private key. A processor may execute the function to calculate a number or string that can be used as a signature. The signature is then provided together with the clear text to provide a signed text. The signature changes completely if the message text or the key changes by a single bit. While calculating the signature requires little computational power, recreating a message that has a given signature is practically impossible. This way, the clear text can only be changed and accompanied by a valid signature if the private key is available. Further, other entities can easily verify the signature using the publicly available public key.

[0253] In most circumstances, encrypting and decrypting comprises a processor executing a cryptographic function to calculate an output string representing the encrypted message or a clear text message respectively.

[0254] Keys, tokens, metadata, transactions, offers, contracts, signatures, scripts, metadata, invitations, and the like refer to data represented as numbers, text or strings stored on data memory, such as variables in program code of type “string” or “int” or other types or text files.

[0255] An example of the peer-to-peer ledger is the bitcoin Blockchain. Transferring funds or paying fees in bitcoin currency comprises creating a transaction on the bitcoin Blockchain with the funds or fees being output from the transaction. An example of a bitcoin transaction includes an input transaction hash, a transaction amount, one or more destinations, a public key of a payee or payees and a signature created by using the input transaction as the input message and a private key of a payer to calculate the signature. The transaction can be verified by checking that the input transaction hash exists in a copy of the bitcoin Blockchain and that the signature is correct using the public key. To ensure that the same input transaction hash has not been used elsewhere already, the transaction is broadcast to a network of computing nodes (‘miners’). A miner accepts and records the transaction on the Blockchain only if the input transaction hash is not yet connected and the signatures are valid. A miner rejects the transaction if the input transaction hash is already linked to a different transaction.

[0256] Allocating cryptocurrency for a token comprises creating a transaction with the allocated cryptocurrency and the token represented in a metadata field in the transaction.

[0257] When two items are associated, this means that there is a logical connection between these items. In a database, for example, identifiers for the two items may be stored in the same records to make the two items associated with each other. In a transaction, identifiers for the two items may be included in the transaction string to make the two items associated with each other.

[0258] Using the bitcoin protocol, redeeming a script and/or unlocking a token comprises calculating a signature string of the script and/or transaction using the private key. The script

may require more than one signature derived from different private keys or other conditions. The output of this transaction is then provided to a miner.

[0259] Authorising another entity may comprise calculating a signature string of a transaction using a private key and providing the signature string to the entity to allow the entity to use the signature to verify the transaction.

[0260] A user having an account with another entity may comprise the entity storing information about the user, such as email address, name and potentially public keys. For example, the entity may maintain a database, such as SQL, OrientDB, MongoDB or others. In some examples, the entity may also store one or more of the user's private keys.

[0261] It will be appreciated by persons skilled in the art that numerous variations and/or modifications may be made to the above-described embodiments, without departing from the broad general scope of the present disclosure. The present embodiments are, therefore, to be considered in all respects as illustrative and not restrictive.