

CLAIMS:

1. A computer-implemented method of controlling the visibility and/or performance of a contract, the method comprising the steps:
 - 5 (a) storing a contract on or in a computer-based repository;
 - (b) broadcasting a transaction to a blockchain, the transaction comprising:
 - i) at least one unspent output (UTXO); and
 - ii) metadata comprising an identifier indicative of the location where the contract is stored;
 - 10 (c) interpreting the contract as open or valid until the unspent output (UTXO) is spent on the blockchain; and
 - (d) renewing or rolling the contract on by:
generating a new key using data relating to a previous key associated with the contract;
 - 15 generating a script comprising the new key, the location of the contract and a hash of the contract; and
paying an amount of currency to the script.
2. A method according to claim 1 wherein the transaction further comprises a
20 deterministic redeem script address, preferably wherein the redeem script address is a pay-to-script-hash (P2SH) address.
3. A method according to claim 2 and further comprising the step of terminating the contract by broadcasting a further transaction to the blockchain to spend the output
25 (UTXO).
4. A method according to claim 3 wherein the further transaction comprises:
an input which is the output (UTXO); and
an unlocking script comprising a signature; the metadata; and a public key.
30
5. A method according to any preceding claim wherein the contract defines:
 - i) at least one condition; and

ii) at least one action whose performance is dependent upon the evaluation of the condition; and/or

wherein the metadata comprises:

- i) an address or representation of an address of where the contract is stored in the computer-based repository; and/or
- ii) a hash of the contract.

6. A method according to any preceding claim and comprising the step:

checking whether the contract has been terminated by determining whether the unspent transaction UTXO is in the list of unspent transaction outputs for the blockchain.

7. A method according to any preceding claim, wherein

- (i) the contract is stored in a Distributed Hash Table (DHT); and/or
- (ii) the method comprises the step of:

broadcasting a transaction to the blockchain comprising an instruction to spend the output at a specified date and/or time, preferably wherein the instruction is a CheckLockTimeVerify instruction.

8. A method according to any preceding claim wherein

- (i) access to some or all of the contents of the contract is restricted to at least one designated authorised party; and or
- (ii) the contract comprises a Deterministic Finite Automaton (DFA) to implement the contract;
preferably wherein the Deterministic Finite Automaton is defined using a codification scheme.

9. A method according to claim 8 wherein the Deterministic Finite Automaton is implemented using:

- i) at least one blockchain transaction, preferably using a scripting language;
- ii) a computing agent arranged to monitor the state of the blockchain; and/or
- iii) a set of instructions for a digital wallet.

10. A computer-implemented method of controlling the visibility and/or performance of a contract, the method comprising the steps:
- (a) storing a contract on or in a computer-based repository;
 - (b) broadcasting a transaction to a blockchain, the transaction comprising:
 - 5 i) at least one unspent output (UTXO); and
 - ii) metadata comprising an identifier indicative of the location where the contract is stored; and
 - (c) interpreting the contract as open or valid until the unspent output (UTXO) is spent on the blockchain; and
 - 10 (d) generating a sub-contract derived from the contract, wherein the sub-contract is associated with a deterministic address and is generated by:
 - iii) using a new public key derived using a seed;
 - iv) storing the sub-contract in or on the repository with a reference to the contract, and broadcasting a transaction to the blockchain comprising a script which includes
 - 15 the reference; and/or
 - v) adding a reference to the sub-contract to the metadata of the existing contract.
11. A method according to claim 10 wherein the transaction further comprises a deterministic redeem script address, preferably wherein the redeem script address is a
- 20 pay-to-script-hash (P2SH) address.
12. A method according to claim 11 and further comprising the step of completing the contract by broadcasting a further transaction to the blockchain to spend the output (UTXO);
- 25 preferably wherein the further transaction comprises:
- an input which is the output (UTXO); and
 - an unlocking script comprising a signature; the metadata; and a public key.
13. A method according to any one of claims 10 to 12 wherein:
- 30 (i) the contract defines:
- a) at least one condition; and

b) at least one action whose performance is dependent upon the evaluation of the condition; and/or

(ii) the metadata comprises:

- a) an address or representation of an address of where the contract is stored in the computer-based repository; and/or
- b) a hash of the contract.

14. A method according to any one of claims 10 to 13 and comprising the step: checking whether the contract has been terminated by determining whether the unspent transaction UTXO is in the list of unspent transaction outputs for the blockchain.

15. A method according to any one of claims 10 to 14, wherein the contract is stored in a Distributed Hash Table (DHT).

16. A method according to any one of claims 10 to 15 and comprising the step of: broadcasting a transaction to the blockchain comprising an instruction to spend the output at a specified date and/or time, preferably wherein the instruction is a CheckLockTimeVerify instruction.

17. A method according to any one of claims 10 to 16 wherein:

- i) access to some or all of the contents of the contract is restricted to at least one designated authorised party; and/or
- ii) the contract comprises a Deterministic Finite Automaton (DFA) to implement the contract;

preferably wherein:

- the Deterministic Finite Automaton is defined using a codification scheme; and/or
- the Deterministic Finite Automaton is implemented using:
 - i) at least one blockchain transaction, preferably using a scripting language;
 - ii) a computing agent arranged to monitor the state of the blockchain; and/or
 - iii) a set of instructions for a digital wallet.

18. A system arranged to perform the method of any preceding claim.