



(21)申請案號：106105713

(22)申請日：中華民國 106 (2017) 年 02 月 21 日

(51)Int. Cl. : **G06Q20/40 (2012.01)**

(30)優先權：	2016/02/23	英國	1603123.9
	2016/02/23	英國	1603125.4
	2016/02/23	英國	1603117.1
	2016/02/23	英國	1603114.8
	2016/04/01	英國	1605571.7
	2016/11/15	英國	1619301.3

(71)申請人：恩鏈控股有限公司 (安地卡及巴布達) NCHAIN HOLDINGS LIMITED (AG)  
安地卡及巴布達

(72)發明人：賴特 克雷格 WRIGHT, CRAIG (AU)；薩凡納 斯特凡 SAVANAH, STEPHANE (GB)

(74)代理人：王尊民

申請實體審查：無 申請專利範圍項數：27 項 圖式數：13 共 68 頁

(54)名稱

區塊鏈執行智能化合同的註冊及自動化管理方法

REGISTRY AND AUTOMATED MANAGEMENT METHOD FOR BLOCKCHAIN-ENFORCED SMART CONTRACTS

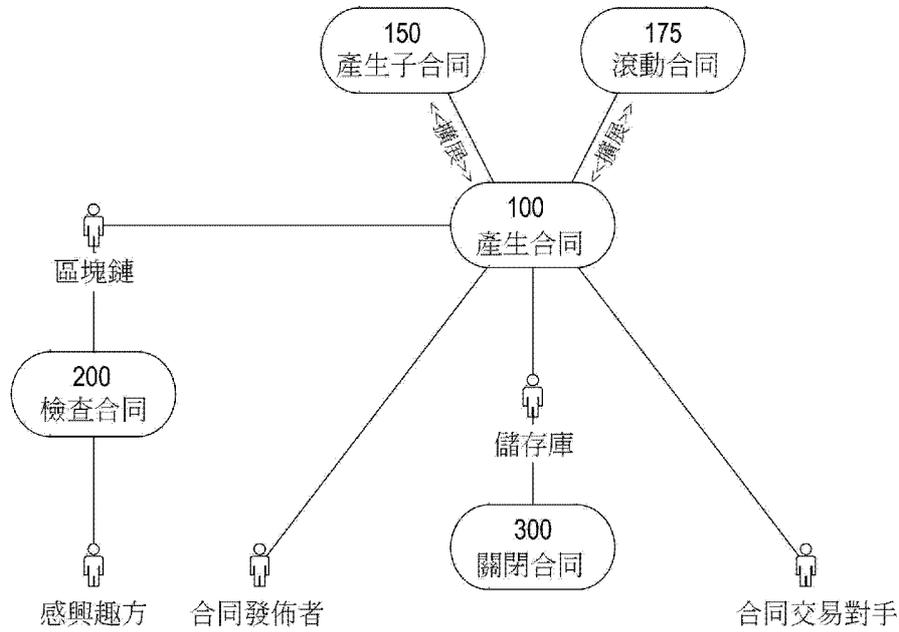
(57)摘要

本發明涉及一種令牌化、區塊鏈及智能化合同技術，其提供可將合同自動化管理簡化之技術安排。本發明包括使用基於電腦的儲存庫來儲存合同之方法及系統，接著將合同由區塊鏈上的交易表示，交易腳本中的元數據包括合同的雜湊值及在儲存庫中標識合同之位置。交易亦包括一未花費的交易輸出(UTXO)，代表狀態為開放(即尚未終止)合同。在稍後的時間點藉由花費該交易輸出以終止合同，例如使用 nLockTime + CheckLockTimeVerify (CLTV)。通過將該概念與其他技術和計算組件結合，本發明可提供一種強大的機制來實現各種任務，如更新或滾動合同，或將其劃分為複數子合同或複數條件。此外，由於合同的地位和存在是通過該區塊鏈之證明，因此提供了永久、公開可見且不可變的合同記錄。

The invention relates to the fields of tokenisation, blockchain and smart contract technologies. It provides a technical arrangement which simplifies the automated management of contracts. The invention comprises a method and system which use a computer-based repository for storage of the contract. The contract is then represented by a transaction on the blockchain. Metadata within the transaction's script includes a hash of the contract and a means of identifying its location within the repository. The transaction also includes an unspent output (UTXO) which indicates its status as an open (ie not terminated) contract. The contract is terminated by spending the output at a later point in time, for example, using nLockTime + CheckLockTimeVerify (CLTV). By combining this concept with other techniques and computing components, the invention can provide a powerful mechanism for implementing various tasks such as renewing or rolling over the contract, or dividing it into sub-contracts or conditions. Furthermore, as the

status and existence of the contract is evidence via the blockchain, this provides a permanent, publicly visible and non-alterable record of the contract.

指定代表圖：



第1圖

## 【發明說明書】

### 【中文發明名稱】

區塊鏈執行智能化合同的註冊及自動化管理方法

### 【英文發明名稱】

Registry and Automated Management Method for Blockchain-Enforced Smart Contracts

### 【技術領域】

【0001】 本發明係有關一種電腦協議，特別是指一種條件控制過程的驗證、執行及/或實現，例如與合同有關的過程。本發明特別適合於與區塊鏈網路一起使用，並且可以通過智能合同來使用。

### 【先前技術】

【0002】 區塊鏈是由不可更改的區塊所組成的分散式電腦系統，而這些區塊由交易組成，每個區塊包含前一個區塊的雜湊，以便區塊自動鏈接在一起，創建自成立以來已寫入區塊鏈中的所有交易的記錄。交易包含被嵌入其輸入和輸出腳本中的小程序，其指定可存取交易的輸出方式及由誰輸出，每個未花費的交易(例如未花費的交易輸出(unspent transaction output, UTXO))都可以用作新交易的輸入。

【0003】 儘管其他區塊鏈實現已被提出和開發，但最廣為人知的區塊鏈技術的應用為比特幣分類帳，雖然本文中為了便利和說明而使用比特幣，但應當注意的是，本發明不限於與比特幣區塊鏈一起使用，並且替代的區塊鏈實現亦落入本發明的範圍內。

【0004】 區塊鏈技術以使用加密貨幣實現而聞名，但在最近數位企業家已開始探索使用基於比特幣的加密貨幣安全系統和可以儲存在區塊鏈上的資料來實現新系統。這些包括但不限於：

- 儲存元數據
- 實施數位令牌
- 實施並管理合同

【0005】 現代合同管理的一個關鍵問題是，它傾向於是特定本地商店和手動維護的合同副本，因此，如「智能合同」之電腦協議已經開始引起注意，因為其可部分或全部實現合同的自動執行或履行。智能合同可提供諸如增強安全

性和降低交易成本等優點，然而，雖然有一些已知的技術解決方案旨在確保這些合同一旦儲存就不能被修改，卻沒有一般公認的公共登記冊來檢查合同的有效性，即合同是否仍然開放或已被終止。

【0006】 因此，希望提供一種以電腦實現的機制，其可控制公眾對合同存在的可見性，並且有助於相關方以自動方式管理、執行和維護諸如合同之類的基於性能的過程的能力(即通過機器而不是人力管理)。重要的是，此機制將提供技術能力來指定合同內定義的行為的控制條件和觸發器。

【0007】 應當注意，在此定義和描述的本發明不限於與該詞的法律意義上的合同一起使用。合同可為定義可在特定條件下觸發的一組行為的文件、檔案或其他機制。控制條件可以公開實現。本發明不應被視為限於在法律或商業導向的背景下使用，術語「合同」不應被解釋為具有限制性的含義，例如，合同可為火車、航空公司或音樂會場地的票，並且在其上打印諸如機器可讀條碼的存取代碼以提供障礙物解鎖。

【0008】 因此，本發明即提出一種即時傳送設備相關訊息之系統，有效解決上述該等問題，具體架構及其實施方式將詳述於下：

#### 【發明內容】

【0009】 本發明之主要目的在提供一種控制合同之可見性及執行之電腦執行方法，該方法包括下列步驟：

- (a) 將一合同儲存於一電腦中的一儲存庫；
- (b) 將一交易廣播到一區塊鏈，該交易包括：
  - i) 至少一未花費的交易輸出(unspent output； 以下簡稱UTXO、「交易輸出」或「輸出」)；以及
  - ii) 元數據，包括指示儲存該合同之位置的一識別碼(identifier)；以及
- (c) 更新或滾動該合同，包括下列步驟：
  - 利用一先前金鑰之相關資料產生一新金鑰，該先前金鑰與該合同相關聯；
  - 產生一腳本，其包括該新金鑰、該合同之該位置及該合同的一雜湊；以及
  - 支付一數量之貨幣給該腳本。

【0010】 通過使用與合同相關的先前金鑰相關的資料產生新金鑰來更新或滾動合同，產生包括新金鑰的腳本、合同的位置和合同的雜湊，以及支付一數

量之貨幣給腳本，由於新金鑰與先前金鑰相關聯的優點，授權方可通過與更新或滾動合同的連接來查看源合同，從而能夠驗證合同中的滾動，而不會有任何確定性或隱私性的損失。另一的優點在於，將合同儲存在電腦的離線儲存庫中(即不形成區塊鏈的一部分)可減少儲存器和處理量，而不會有確定性或隱私性的損失，因為與更新或滾動合同相關聯的金鑰又與源合同的金鑰相關聯。

【0011】 在「滾動」情況下，UTXO可能被花費而發送到「新」滾動合同。但是，亦可通過在鎖定時間之前將輸出花費以取消現存合同，將整個合同全部取消。

【0012】 本發明可提供用於控制合同的可見性及/或性能的電腦實現的方法和系統。「可視性」可能意味著合同的存在及/或內容可用或可及的方式以及與誰合作。合同可能是「智能合同」。該方法可為自動化智能合同方法。其可為自動化監控合同存在、有效性及/或執行過程的方法。由於合同可以以區塊鏈交易的至少一部分的形式表示，本發明可以被稱為令牌化方法/系統。交易中的元數據可提供用於表示及/或存取合同的區塊鏈接實現的令牌。

【0013】 本發明可提供一種方法/系統，該方法/系統可將合同儲存在儲存庫(註冊表)中，其中可以使用合同的雜湊可做為用於尋找合同的查找金鑰。

【0014】 該方法可包括下列步驟：

將一合同儲存於設於一電腦中之一儲存庫；

將一交易廣播到一區塊鏈，該交易包括：

- i) 至少一未花費的交易輸出(unspent output, UTXO)；以及
- ii) 元數據，包括指示儲存該合同之位置的一識別碼(identifier)。

【0015】 合同可能被解釋為公開或有效，直到UTXO在區塊鏈上被花費。區塊鏈可能是也可能不是比特幣區塊鏈，這提供了利用UTXO代表表示區塊鏈上之合同的狀態或有效性的新機制之優點。

【0016】 該方法可包括使用區塊鏈外電腦實現的過程、代理商或其他實體來觀察區塊鏈的狀態，並且基於輸出當前是否未被使用而以某種方式行為的步驟。該過程可被安排為將UTXO解釋為合同狀態的指標，換言之，雖然輸出仍然在區塊鏈的UTXO列表內，即交易仍然未被使用，但可表示元數據指向或引用的合同的有效性或「開放」狀態。一旦使用UTXO，合同可能被認為是完整的(已

終止)，此情況可以在合同中說明。然而，一旦UTXO已經被使用，元數據可能會繼續包含一個指標或引用的合同和合同的雜湊，所以合同可以保留其功能。

【0017】該方法可以包括發佈合同的存在之步驟，其可通過以下步驟來實現：

- 合同發佈者可創建新的合同文件並將其發佈到儲存庫。商店的位置和該文件的安全雜湊可以儲存以供以後使用；
- 在 $n$ 個多簽名結構中創建涵蓋合同檔案的贖回腳本，其中：
  - $m$ 為至少一個；及
  - $n$ 是 $m$ 加上元數據區塊的數量
- 在腳本中至少包含一公鑰；這可能是合同發佈者的公鑰，但也可能需要其他簽名
- 支付一定數量的貨幣，例如比特幣到腳本中，最好通過P2SH交易
- 等待直到交易被發佈到區塊鏈上，並提取已發佈交易的交易ID
- 創建一個新的交易，將鎖定時間設置為合同的到期時間，將交易的輸出支付回公鑰雜湊；或
- 對於滾動期限合同：使用自動計算代理商檢測區塊鏈上的交易，並等到合同到期時間才觸發代碼將其轉換為新合同；或
- 對於基於完成的合同(其中 $x$ 個實體同意合同已經履行)：創建 $n$ 個多簽名中之 $m$ 個簽名的交易，並將其發送給這些實體以在完成時共同簽署。

【0018】儲存庫可為區塊鏈外之儲存資源，換言之，儲存庫可能不會構成區塊鏈本身的一部分。基於電腦的儲存庫可為或包括伺服器。儲存庫可為在基於電腦的資源上提供的資料庫或其他儲存設施。儲存庫可被索引、允許搜索。儲存庫可包括分散式雜湊表。合同可儲存在分散式雜湊表(DHT)中或與分散式雜湊表(DHT)相關聯。

【0019】交易可進一步包括確定性贖回或鎖定腳本地址。該地址可為付費腳本雜湊(P2SH)地址，因此，利用公開給區塊鏈的交易可公開獲得合同的存在(或合同中的定義元素)，其係由合同之發佈者判斷或提供付費腳本雜湊地址來公開；及/或合同的元數據。

【0020】該方法可進一步包括通過將(額外)交易廣播到區塊鏈來花費UTXO

以終止合同的步驟。額外交易更包括一輸出(UTXO)的輸入；以及包括簽名的解鎖腳本；元數據；及公鑰。這可以通過使用區塊鏈交易來花費該輸出以提供自動終止合同之優點。

【0021】 合同可以定義：i)至少一條件；及ii)至少一行為，其效能取決於條件的評估。條件可為可以被評估為真或假的測試。條件可為合同的一部分(例如一個條款)。完成或履行條件可能需要履行合同，若評估為真，則該條件可能已被完成。

【0022】 元數據可包括：i)將合同儲存在基於電腦的儲存庫中的地址或一地址代表；及/或ii)合同的雜湊。

【0023】 該方法可包括觀察區塊鏈狀態之步驟，包括搜尋區塊鏈以找到一包含UTXO之交易；亦可包括藉由確定UTXO是否在區塊鏈的UTXO列表中，來檢查合同是否已被終止的步驟。該監視或檢查過程可為自動的。可由適當編程的計算代理商或資源來執行。基本上如後面章節「用於本發明的說明性計算代理商」的部分中所描述，代理商可以基於UTXO的已用或未用狀態來執行操作。因此，UTXO的狀態可以控制或影響一個偏移計算代理商的行為。

【0024】 該方法可包括將交易廣播到區塊鏈的步驟，該步驟包括在指定日期及/或時間花費該交易輸出的指令。該指令可為CheckLockTimeVerify指令。

【0025】 存取合同的部分或全部內容可能只限於至少一個指定的授權方，換言之，可能需要授權才能存取或查看部分或全部合同。在一些實施例中，保護機制可應用於合同本身，例如檔案的一個或多個部分可以被保護，但整體內容可為公開的。此部分保護可以適用於合同內的資訊加密以及雜湊檢測改變其內容。

【0026】 合同可能包括一確定性有限自動器(DFA)來實施合同。確定性有限自動器可以法律編纂方案(codification scheme)定義。確定性有限自動器可執行，使用：

- i) 至少一區塊鏈交易，其利用一腳本語言；
- ii) 一計算代理商，其用以監視該區塊鏈之狀態；及/或
- iii) 一數位錢包的一組指令。

【0027】 本發明之另一方面提供一種控制合同之可見性及效能的電腦執行

方法，該方法包括下列步驟：

- (a) 將一合同儲存於一電腦中的儲存庫；
- (b) 將一交易廣播到一區塊鏈，該交易包括：
  - i) 至少一UTXO；以及
  - ii) 元數據，包括指示儲存該合同之位置的一識別碼(identifier)；以及
- (c) 從合同中衍化(derived)產生一子合同，子合同與一確定性位址相關聯並由以下產生：
  - iii) 產生一新金鑰，其利用一種子進行衍化；
  - iv) 通過合同的一參考，將子合同儲存於儲存庫，並向區塊鏈廣播包含一腳本的一交易，腳本包括該參考；以及
  - v) 將子合同的一參考加入到現存之合同的元數據中。

【0028】 通過產生從合同中衍化出的子合同，其中子合同與確定性地址相關聯，並且通過使用種子導出的新公鑰來產生，利用合同的一參考將子合同儲存在儲存庫中或其上，並將包含一腳本之交易廣播到區塊鏈，此腳本包括參考及/或添加對現有合同的元數據的子的引用的腳本的塊的廣播交易，這提供了可以獨立地管理子合同而沒有損失的優點，因為子合同在加密方面與源合同有關。此外，可將子合同儲存在區塊鏈外之儲存庫中，以最小化儲存器和處理資源。

【0029】 該方法可包括利用基於電腦的代理商、基於合同內容以監控區塊鏈及/或執行動作，此代理商基本上如下面章節標題「說明本發明使用之計算代理商」的部分中所述。

【0030】 本發明還可提供一種電腦執行之系統，用以執行上述任何方法步驟，或本文描述的方法的任何實施例。本發明可提供一種用於控制合同的可視性及/或效能之電腦執行之系統，該系統包括：

- 一電腦中之一儲存庫，用以儲存合同；以及
- 一區塊鏈，包括一交易，交易中包括：
  - i) 至少一UTXO；以及
  - ii) 元數據，包括指示儲存合同之位置的一識別碼(identifier)。

【0031】 元數據亦可儲存合同之雜湊，合同可為智能合同。

【0032】 儲存庫可包括一資料庫，其可包括一分散式雜湊表(Distributed Hash Table, DHT)，可被索引、可檢索。其可包括至少一安全機制以控制合同之存取。

【0033】 系統還可以包括適當配置、基於計算的實體或代理商。代理商可用以監視及/或搜索區塊鏈，可基於區塊鏈的狀態來執行至少一個動作，可安排以確定UTXO是否已經花費，可基於UTXO是否已經花費來執行一個或多個動作。

【0034】 此處所描述之關於一實施例或方面的任何特徵亦可用於任何其它實施例或方面，例如，關於該方法描述的任何特徵也可以相對於該系統使用，反之亦然。

【0035】 現在提供本發明可提供的一些優點的非全面列表。

【0036】 本發明可提供簡化結構化控制條件的自動化管理之技術，這在本文中可以被稱為「合同」，這反過來使當發生爭議時能夠更容易地同意合同的狀態。本發明還可提供一種利用電腦自動判斷合同有效性的方式來保持合同安全的公開記錄，並在驗證時將其細節釋放給授權實體的機制。因此，本發明可提供一種安全性增強的控制機制，其允許或禁止以智能方式存取資源。

【0037】 本發明還提供通過電腦系統向觀眾發佈合同的能力，使合同的細節可僅限於授權實體，但是公知存在的合同是公知的。換言之，公眾可以知道A和B之間有合同，這可以被公開驗證，但除其存在之外的任何東西都限於授權方(通常只能是A和B)。

【0038】 其還提供了電腦執行的機制，允許合同有時限(即在特定時間或給定日期之後到期)；條件限制(即一旦在合同中指定的可交付成果已被滿足)或開放式(即合同繼續在一通知期間滾動以終止合同)。

【0039】 其可提供一種以公開方式通知終止合同的機制，例如，在花費交易中使用 $nLockTime + CheckLockTimeVerify$ (CLTV)來「制定」到期。

【0040】 其可提供一種以確定性方式來構建子合同層級的機制，以控制合同的不同方面進行分割，例如，在技術開發過程中，需求階段可能具有與開發階段不同的一組控制觸發器。

【0041】 由於本發明可以在區塊鏈平台上執行，且可擴展區塊鏈的功能，使其可以技術上不同的方式使用，本發明可提供改進之區塊鏈系統或平台。

【0042】本發明可用於將任何UTXO轉換成智能合同，例如用於數位存取。例如，考慮一種情況，當消費者支付商家一存取服務費一段時間後，若商家的付款地址被以一智能合同執行，則本發明可用於實現該服務的存取控制機制，可產生一支票(check)以確保款項已支付，以及用於在期限結束時將價值清算到商家賬戶的自動化流程。

【圖式簡單說明】

【0043】

第 1 圖為本發明實施例中如何使用區塊鏈交易來執行各種合同相關任務之概述。

第 2a圖為一簡單狀態機，包含二種狀態：(i)合同打開；(ii)合同關閉。

第 2b圖為第 2a圖情境之元數據定義，此元數據攜帶(比特幣)交易輸出，其指定合同位置及有效證明(透過雜湊)。

第 2c圖顯示第 2a和 2b圖的情境相關的「發行」交易，其最初將合同(之雜湊)儲存在區塊鏈上。

第 2d圖利用花費比特幣將第 2a至 2c圖之合同取消。

第 3a圖為元數據之一情境，其創建並發佈具有隱藏所有權的資產到區塊鏈。

第 3b圖說明一用於「資助」第 3a圖資產之交易。亦即，將一些比特幣放入資產的公鑰中，以便該資產可以為其交易提供資金(如 3c所示的公開交易)。

第 3c圖說明將第 3a及 3b圖之資產發佈之一區塊鏈交易。

第 3d圖說明將第 3a、3b及 3c圖相關之合同關閉的交易說明，當需要取消合同時，將使用UTXO。在此情況下，資產和資產的隱藏所有者都被要求簽名。

第 4a圖所示為涉及租賃合同的場景之一狀態機模型。

第 4b圖所示為第 4a圖之情境的元數據。

第 4c圖說明將第 4a及 4b圖之資產所有權發佈到區塊鏈的一交易。

第 5a圖所示為合同正在滾動的場景之一狀態機模型。

第 5b圖所示為第 5a圖之情境的元數據。

第 5c圖為可公開第 5a及 5b圖之初始合同、且合同可初始滾動到區塊鏈上的一交易。

第 5d圖所示為第 5a至 5d圖之合同終止之交易。

第 6a圖為合同條件的場景之狀態機模型。

第 6b圖所示為第 6a圖的場景之元數據。

第 6c圖所示為可用於創建初始合同和兩個子合同並將其發佈之交易。

第 6d圖所示為關於第 6a至 6c圖的場景之交易。

第 7 圖至第 13 圖所示為用於從父金鑰導出子金鑰之技術的各個方面，該技術適用於本發明的各方面。

#### 【實施方式】

【0044】 內置於區塊鏈中的智能合同可以通過直接嵌入到比特幣交易(即鎖定/解鎖腳本)及/或通過外部基於電腦應用程式的邏輯來實施。此基於電腦的外部應用程式可能被稱為「代理」、「神諭(oracles)」或「機器人」。此外，一些合同條件可以通過其他比特幣交易元素(如InLockTime字段)來執行。

【0045】 本發明所述中，只要在表示合同的區塊鏈上存在有效的未花費交易輸出(UTXO)，則合同被解釋為保持有效。應當理解，由於各種機制(例如程式化計算代理)的行為受到合同本身的條件或規定的控制，可以影響和改變此種未花費的狀態，例如，合同規定將在某個日期過期，或當某個值達到指定的閾值時，該期限就屆滿。

【0046】 利用未花費的交易輸出(UTXO)來表示合同的原則可與諸如加密技術的其他特徵組合使用，可執行複雜的場景和活動。有效地，無符號的UTXO的內容及腳本中相關聯的元數據使其能被花費，允許該交易充當指向區塊鏈外的儲存庫的指標或引用，此儲存庫中包含合同正式詳細資訊。在這裡，「區塊鏈外」代表不是區塊鏈本身的一部分，這提供了一種機制，任何人都可以使用基於軟體的組件或工具來確定合同是否已被終止或通過檢查區塊鏈仍然有效/打開。一旦合同終止，這將被記錄在區塊鏈上做為一花費輸出，並可供公眾查閱。該區塊鏈交易成為合同存在和現狀的永久性、不可變的和公開的記錄。

【0047】 儲存庫(其也可以稱為「註冊表」或「註冊」)可以用各種方式來實現，包括如分散式雜湊表(distributed hash table, DHT)。合同的雜湊可以產生並做為元數據儲存在區塊鏈交易中，可以用作查找關鍵字以供從區塊鏈中引用合同。在交易元數據中也提供了對合同位置的引用，例如可提供儲存庫的連結(URL)。雖然元數據是公開的，但合同本身可能不是或可能受到部分保護。

【0048】標準比特幣功能(如CheckLockTimeVerify(CLTV))可允許合同在將來的某一時刻有正式的自動化到期，使用區塊鏈讓此有效期成為安全(不可更改)的公開記錄。該概念結合使用下面描述的多個加密金鑰，使CLTV模型可自動滾動或續訂合同，除非被明確取消。

【0049】將確定性子金鑰之使用與本發明所述的令牌化機制組合，允許創建關於合同的子合同或時間表。

【0050】此外，使用區塊鏈外計算代理(oracles)可允許合同條件內置於受信任的第三方並由其修改，這意味著代理人的行為可能受合同定義中提供的條件(例如「IF」條件語句)的影響。

【0051】金鑰項目

【0052】此處可使用下列項目：

- 合同發佈者：  
此實體代表負責將合同發佈到區塊鏈上的行為者。
- 感興趣的一方：  
該實體代表可能需要確定特定合同是否仍然存在的行為者，或可能需要確定合同的細節。
- 儲存庫：  
該實體代表保護/儲存區塊鏈智能合同引用的合同的一結構化表示的位置。
- 合同交易對手：  
該實體代表特定合同的交易對手。請注意，在許多情況下，該實體將不存在
- 合同：  
這是儲存在儲存庫中的結構化檔案或文件，其係從區塊鏈中引用。合同可以是任何類型的合同或協議，可能包括例如金融合同、業權合同、服務合同等。合同的內容可以是公開的或私人的。正式的合同使用編纂方案以有條理的方式表達。

【0053】合同模型

【0054】合同模型之基本元件如下：

- 一編纂方案，允許對任何類型的合同進行完整的描述。該方案可能是一種新的構造，或者可以使用現有的設施，如XBRL，XML，JSON等）；

- 一確定性有限自動器 (Deterministic Finite Automaton, DFA)，用於執行在編纂方案中明確定義的合同。其係由以下組成：
  - 一組參數，以及這些參數的來源在哪裡；
  - 一組狀態定義
  - 一組狀態之間的轉換，包括轉換的觸發器和轉換過程中遵循的規則。
  - 規則定義表
- 本合同實例的具體參數的定義；
- 確保和保護合同的機制；
- 「瀏覽器」，使合同能夠以正式的法律語言被人閱讀；及
- 「編譯器」，將編纂方案轉換為oracle代碼及/或腳本，如比特幣腳本。

#### 【0055】 執行合同

【0056】 當合同在儲存庫中註冊時，相關聯的地址(例如URL和hash)可用作區塊鏈交易中的元數據，將區塊鏈上的交易與控制合同本身相關聯。這可以用各種形式實施，但是在「編纂方案」一節中為了完整性而提供了適當的編纂方案。

【0057】 關於合同定義中包含的確定性有限自動器(DFA)是如何實現的，有許多不同的方法：

- 做為區塊鏈交易或交易序列。各種形式的DFA可以在比特幣腳本語言中直接實現；本領域技術人員將理解這一點，並且本發明不限制通過區塊鏈交易來實現DFA的方式；
- 做為基於代理(例如oracle)的過程或過程序列。下面標題為「本發明使用的說明性計算代理」部分描述了定義和運行合適的代理，以監視區塊鏈和其它可能的外部資源的基本過程。
- 做為智能電子錢包的一組說明。在此內容中，智能電子錢包實際上只是一個可以處理某些合同條件的本地oracle程序，例如將交易輸入分配給區塊鏈交易。

【0058】 注意，給定的合同定義可以做為上述三種機制的混合來實現，其中每個合同狀態轉換實際上是單獨的實現。有許多從合同定義創建實施的方法，包括手動製作相關的交易/代碼。

【0059】 公開發佈合同之存在

【0060】 為了公開合同(或合同中定義的元素)的存在，使用付費腳本雜湊地址(P2SH)將交易Tx發佈到區塊鏈。P2SH交易是收件人必須提供與腳本雜湊相匹配的腳本，以及使腳本評估為true的資料，以便花費該交易。本發明的實施例中，可從以下輕易判斷付費腳本雜湊位址(pay-to-script-hash address, P2SH)：

- 合同發佈者；及
- 合同的元數據。

【0061】 在本發明的一些實施例中，未花費的交易可以被解釋為合同狀態的指標。可利用區塊鏈外處理、以基於輸出是否未被使用來監視區塊鏈並以某種方式行事。換言之，雖然該輸出仍然在區塊鏈的UTXO列表中(即，交易仍然未被花費)，但是這表示元數據指向或引用的合同的有效性。一旦此輸出花費完，合同就被認為是完整的。此條件(只要UTXO存在，合同仍然有效/開放)可以是合同本身的條件。然而，在其他實施例中協議的必要規定不是一個替代的終止條件，請注意，即使在交易已經被花費(因此在UTXO列表中不再存在)之後，交易仍然永久地駐留在區塊鏈上，並且仍然保留指標或引用合同以及合同之雜湊，甚至可在花費後保留其功能。

【0062】 子合同/條件

【0063】 子合同是與現有合同直接相關的合同。條件是現有合同內的條款，必須符合合同條款。

【0064】 在本發明的實施例中，子合同和條件可以相同方式執行，例如可以以與確定性贖回腳本地址做為UTXO實現的合同。在這兩種情況下，當UTXO花費時，實體可以解釋為完整的(在條件的情況下，表示條件已經滿足)。如上所述，元數據將仍然包含一個指標或對儲存庫內實體的位置的引用，且還包含其雜湊。因此，在其他實施例中，根據合同規定的條件，子合同或條件可以保持存在並且即使在輸出已經消耗之後也保留功能。

【0065】 有一些機制可用於創建條件或子合同的確定性地址：

- 使用種子資訊導出一新公鑰；
- 在儲存庫內創建和發佈子合同，並參考主合同，將其做為元數據引用；及
- 將條件/子合同引用添加到現有合同的元數據中。

**【0066】 確保合同**

**【0067】** 合同的正式表示(即指定合同内容的文件或档案)可根据具体合同的正式需要以各种方式确保，尽管在所有情况下，合同存在的公开记录将会在元数据记录所包含的区块链上发布(有关特定元数据结构详细资讯，请参见「编纂计划」一节)。

**【0068】** 从该区块链记录中，被授权的实体将可学习正式表示(formal representation)的位置及杂凑，以确定自发布交易以来，正式表示形式尚未被修改。

**【0069】** 然而，有可能通过多种方法进一步确保形式表示本身：

- 文件储存库本身可以呈现存取控制机制；及
- 合同本身可以通过标准加密技术来保护，以限制存取相关解密金钥的实体。

**【0070】** 在许多情况下，合同本身将受到部分保护。举例而言，文件中的某些部分可能受到保护，而整个内容是公开的，例如公开了如何实施固定利率贷款的细节，但只有缔约方知道哪些人拿出多少贷款及以什么样的利率。

**【0071】** 此部分保护适用于合同中的资讯的加密，及杂凑检测改变其内容。

**【0072】** 对于一些合同而言，细节可在其一生中修改，不应要求重新签发合同本身。此可以通过确定合同子集上的杂凑的范围来实现。可能有用之例子为实施单位信托，支持单位信托的合同可能不变，但单位受益人可通过售卖合同进行修改。在一实施例中，可利用子合同达到记录变化。

**【0073】 终止合同**

**【0074】** 由于区块链提供永久性、不可变更的交易记录，因此只需删除相关的合同文件即可终止合同。这意味着安全的合同储存库必须具有与通过多个标准机制支持的区块链本身相同的储存和保留规则，而这代表解决方案必须提供一种直接利用区块链记录以侦测合同到期的机制。

**【0075】** 终止方法被定义为合同中的一项条件，并且可以用各种方式实现，所有这些方法在概念上被本发明所涵盖。在本发明的较佳实施例中，通过代表合同的UTXO之花费来处理终止。

**【0076】** 对于一些合同类型，合同的到期可以与合同本身的出版同时发

佈。創建二有效交易，一者為發佈合同並獲得代表合同的交易輸出，另一者則用於花費該輸出。第二個交易具有一CheckLockTimeVerify集合，以在給定的未來日期(代表合同的結束)中花費輸出。根據以前的評論，此為本發明之標準方式，但不是唯一的方法。

【0077】此自動花費可以擴展到支持合同的滾動(例如，若合同不被取消，則自動延長十二個月)。在此情況下，UTXO被花費而將其發送到「新」的滾動合同。然而，亦可通過在鎖定時間之前花費輸出以取消舊者，從而取消整個合同。

【0078】使用範例模型

【0079】第1圖所示為本發明一實施例中一使用範例模型之概述，其說明了使用範例模型如何使用標準的比特幣交易以直接在比特幣腳本中實現DFA的元素。以下提供了金鑰範例以說明目的。

【0080】創建合同

【0081】合同發佈者(本例中的主要行為人)希望將合同發佈到區塊鏈上以供公眾了解，該過程如表1所述：

步驟	細節
100.10	合同發佈者創建一個新的合同文件並將其發佈到儲存庫中，儲存商店的位置及該文件的安全雜湊值以供將來使用。請注意，根據合同文件本身的性質，此儲存庫可以是公開的、私有的或半私有的。儲存庫被索引，允許其通過各種屬性進行檢索。
100.20	合同發佈者創建一個涵蓋合同文件的贖回腳本，以n個多簽名結構中之m個，其中： - m是至少一個；及 - n是m加上元數據區塊的數量(將為至少2)。 必須始終為此腳本提供之一公鑰係為合同發佈者的公鑰。然而，根據合同條款，也可能需要其他簽名。
100.30	合同發佈者支付名義金額的貨幣(如比特幣)給贖回腳本，其為步驟100.20中通過標準P2SH交易計算的贖回腳本。
100.40	合同發佈者等待直到交易已發佈到區塊鏈上，並提取已發佈交易的交易ID。
100.50	對於固定期限合同，合同發佈者隨後創建一個新交易，鎖定時間設置為合同到期時間，將步驟100.40的輸出支付回合同發佈者的公鑰雜湊。 對於滾動期限合同，基於電腦的代理可接收交易，並等到合同到期時間，然後觸發下表3的「翻轉」範例，將其轉到合同上。 對於基於完成的合同(其中y個實體中的x個同意合同已經履行)，n個多簽名交易中

的m個被創建並發佈給這些實體以在完成時共同簽署)。

表1

【0082】 此情況有兩個關鍵的實施例或變化，下面將詳細說明：

- 從現有合同中創建子合同
- 將現有合同轉為新的合同(續約)

【0083】 創建一子合同

【0084】 在此情況下，合同發佈者希望從現有合同中創建子合同，此過程如下表2所述：

步驟	細節
150.10	<p>合同發佈者從其公鑰中創建一個新的子金鑰，用以在從父合同中導出子金鑰資訊時使用一種子值以創建父合同。這可以是合同發佈者希望(並承諾)的任何衍生，但適當種子的範例可包括：</p> <ul style="list-style-type: none"> <li>- 在步驟100.40中創建合同UTXO的交易ID /索引；或</li> <li>- 在步驟100.20中創建贖回腳本雜湊。</li> </ul> <p>應該指出的是，這個例子假設上述公鑰是合同發佈者的公鑰；然而，本領域技術人員將理解，沒有任何東西可防止其衍變成子金鑰(即子合同的子合同)。</p>
150.20	<p>根據正在創建的子合同的性質，合同發佈者可：</p> <ul style="list-style-type: none"> <li>- 使用主合同文件的位置和雜湊；或</li> <li>- 創建一個新的合同文件，其中包含內嵌之主文件的連結，儲存文件的位置和該文件的安全雜湊以供以後使用；或</li> <li>- 創建一個新的合同文件，其中包含內嵌之主文件的連結，以及所涵蓋的原始合同文件中的字段列表。實際上，這是一份規定了子合同涵蓋另一文件的特定部分的文件，而不是複製原始資訊。</li> </ul> <p>請注意，根據合同文件本身的性質，此儲存庫可以是公開的、私有的或半私有的。</p>
150.30	<p>合同發佈者創建一個涵蓋合同文件的贖回腳本，以n個多簽名結構中的m個，其中：</p> <ul style="list-style-type: none"> <li>- m是至少一個；及</li> <li>- n是m加上元數據區塊的數量(將為至少2)。</li> </ul> <p>必須始終為此腳本提供的一公鑰是合同發佈者的公鑰。然而，根據合同條款，也可能需要其他簽名。</p>
150.40	<p>合同發佈者支付名義金額的貨幣(如比特幣)給贖回腳本，其為步驟150.30中通過標準付費腳本雜湊(P2SH)交易計算的贖回腳本。</p>
150.50	<p>合同發佈者等待直到交易已發佈到區塊鏈上，並提取已發佈交易的交易ID。</p>
150.60	<p>對於固定期限子合同，合同發佈者隨後創建一新的交易，鎖定時間設置為合同到期時間，將步驟150.50的輸出支付回合同發佈者的公鑰雜湊。</p>

表2

【0085】 在一個或多個實施例中，子合同可被獨立監視。舉例而言，考慮

一個財產建造合同，需要驗船師在合同上簽名，合同規定「由<x>簽署」。為了實現這一點，步驟150.60被創建並分發到<x>進行簽名。償還腳本(repay script)不是時間鎖定的，而是做為n個多簽名元素中的m個所創建，其中所需的簽名者是<x>。在一些實施例中，交易將具有兩個輸出：<x>的費用加上步驟150.50中產生的UTXO的支付。

**【0086】 實施例：滾動現有合同**

**【0087】** 在此實施例中，合同發佈者希望將現有合同轉為新合同，如下表3所示：

步驟	細節
175.10	合同發佈者將通過驗證先前的UTXO是否已經被花費來檢查區塊鏈，以確定合同是否已被取消。若已花費，則過程結束。
175.20	合同發佈者從用於創建父合同的公鑰中創建一個新的子金鑰，並將其做為從父合同序列衍生出子金鑰資訊的一種子值。這可以是合同發佈者希望(並承諾)的任何確定性衍生，但可以是： <ul style="list-style-type: none"> <li>- 序列號(例如，滾動實例「1」)；或</li> <li>- 滾動合同的日期範圍</li> </ul> 上述假設公鑰為合同發佈者的公鑰，但實際上沒有什麼可阻止這是一個衍生子金鑰(即子合同的子合同)。有關如何創建子金鑰的實施例，請參閱「子金鑰產生方法」。
175.30	合同發佈者採用現有合同文件的位置和雜湊。請注意，根據合同文件本身的性質，此儲存庫可以是公共的、私有的或半私有的。
175.40	合同發佈者創建一個涵蓋合同文件的贖回腳本，以n個多簽名結構中的m個，其中： <ul style="list-style-type: none"> <li>- m是至少一個；及</li> <li>- n是m加上元數據區塊的數量(將為至少2)。</li> </ul> 必須始終提供給此腳本的二公鑰為合同發佈者和客戶的公鑰。然而，根據合同條款，也可能需要其他簽名。
175.50	合同發佈者透過標準P2SH交易將步驟175.40中計算的一數量之比特幣支付給贖回腳本。
175.60	合同發佈者等待直到交易已發佈到區塊鏈上，並提取已發佈交易的交易ID。
175.70	一個過程(例如基於機器人或基於oracle的執行)將接收交易，並等到合同到期時間再重新觸發表3的「滾動」過程，若未被取消，則再次將其重新滾動。

表3

**【0088】 實施例：確認合同**

**【0089】** 在這個實施例中，有興趣的一方希望確認一合同存在，以涵蓋他

正在查詢的活動，此過程如下表4所示：

步驟	細節
200.10	有意者將檢查區塊鏈以確認與他們感興趣的合同有關的UTXO是否已經被花費掉。若UTXO尚未花費，則合同仍然有效。若UTXO仍然未花費，但是有一個鎖定時間交易掛起，則這將決定合同的到期時間。當UTXO被花費，則合同在某些方面已經完成了。

表4

【0090】 上述主要變量假設有意者知道可通過其他途徑管理合同的交易（一般情況下，即合同發佈者或合同交易對手）。但任何可存取合同文件和合同發佈者的知識的實體將能夠通過以下方式進行檢查：

- 衍生UTXO交易的贖回腳本；及
- 掃描區塊鏈以找到與贖回腳本雜湊匹配的UTXO。

【0091】 實施例：關閉合同

【0092】 在此實施例中，合同發佈者或合同交易方希望關閉現有合同，此過程如下表5所述：

步驟	細節
300.10	關閉的發起者將藉由驗證以前的UTXO是否已經被花費來檢查該區塊鏈是否已被取消。若已花費，則該過程以合同已經關閉的方式結束。
300.20	若存在一現有的關閉交易，則該發起者將簡單地簽署該交易並提交到區塊鏈上
300.30	若沒有現有的關閉交易，則發起者將創建交易，交易輸入是上一個合同的UTXO，且解鎖腳本為其簽名、與合同關聯的元數據及其公鑰。
300.40	當交易被接受到區塊鏈時，公眾知道合同已經關閉(儘管只有參與者將知道具體原因)。

表5

【0093】 合同條件

【0094】 上述相同的機制可用以監視給定合同中的條件，例如檢查點。舉例而言，若一個合同被確定為100個比特幣，20個比特幣將在檢查點1到5支付，則上述子合同模型可以用於獲得主合同和五個子合同。這些子合同中的每一個可以使用相同或不同的簽名者(例如公證人或類似的簽名者)來標記為完整的。以此方式，公開記錄可以保持，並表明符合合同的條件已經得到滿足。接著可將此概念與一程序或應用程式(「機器人」)相結合，一旦合同被標記為完整，其可以用來觸發20個比特幣付款。

【0095】為了說明目的，下面提供了一些範例場景，其中顯示了本發明的一些應用。在所有這些情況下，合同本身的内容被認為是無關緊要且非限制性的。

【0096】實施例情境1：資產之公共註冊表

【0097】在此情境下，Bob決定將其資產(例如他的家)的所有權發佈到區塊鏈上。在這個階段還沒有做任何事情；其僅為一種資產，然後可在隨後的交易中使用。在此情況下，合同沒有終止日期。第2a圖顯示了具有兩個狀態的簡單狀態機：(i)合同開放及(ii)合同關閉。第2b圖顯示了比特幣交易輸出中承載的元數據定義，並通過雜湊表指定合同位置和有效性證明。第2c圖顯示了最初在區塊鏈上儲存合同的「發行」交易(儘管其實際上僅儲存雜湊而不是實際的合同)。第2d圖藉由花費比特幣取消合同。

【0098】實施例情境2：隱藏所有權資產的創建和註冊

【0099】這是一個略微增強版本的情境1，其中Bob想將資產發佈到區塊鏈上，但不想直接顯示其所有權。

【0100】在此情況下，Bob首先從他的公鑰中創建一個子金鑰來表示資產，該子金鑰隨後做為資產細節的一部分發佈到區塊鏈上。在此情況下，資產沒有終止日期。(下面提供了一個可以產生子金鑰的方法的詳細說明，請參見下面的「子金鑰產生方法」)。

【0101】此情境的狀態機與情境1相同，如第2a圖所示。第3a圖顯示了此場景的元數據定義，元數據在比特幣交易輸出中進行，並指定合同位置和有效性證明(通過雜湊)。第3b圖顯示了「投資」資產的交易，亦即，將一些比特幣放入資產的公鑰中，以便該資產可以為其交易提供資金(如第3c圖中的發佈交易)。第3b圖沒有顯示Bob創建的資產子金鑰，因為其並非比特幣交易。

【0102】第3c圖顯示資產發佈的區塊鏈交易。第3d圖顯示合同關閉的交易，當需要取消合同時，將花費UTXO，在此情況下，資產和資產的隱藏所有者都要求簽名。

【0103】實施例情境3：租賃合同

【0104】在這種說明性的情況下，Bob與Eve簽訂租約，期限為三年。合同的條款將指定一些付款。付款的細節與本發明無關。但合同條款是固定的，沒

有違約條款。

【0105】這有一個簡單的狀態機模型，如第4a圖所示。第4b圖顯示了這種情況的元數據。第4c圖顯示了將資產的所有權發佈到區塊鏈上的交易。首先，Bob為資產提供了一些資金，然後資產自己公佈。

【0106】實施例情境4：滾動合同

【0107】在此說明情況下，Bob決定每年從Eve出租房屋，他需要提供兩個月的通知，在更新日期取消租賃，否則將自動滾動。這有一個簡單的狀態機模型，如第5a圖所示。第5b圖顯示了這種情況的元數據。第5c圖顯示了將初始合同和合同的初始翻轉到區塊鏈的交易。

【0108】第一年後，Bob繼續租賃，並沒有終止。EVE-S3-T2發佈後立即由自動化電腦代理，並再次推出一年。應指出的是，EVE也可以使用自己的內部邏輯來完成。

【0109】第二年後，Bob選擇終止租賃，並使用與EVE-S3-T3相同的輸入提出交易。但由於此交易尚未提交，因此輸入未被使用，如果Bob的交易首先發佈到區塊鏈，則會使EVE-S3-T3無效。雖然所涉及的數額是微不足道的，但是機器人不會簽署交易，除非輸出是針對Eve的公鑰雜湊(或任何合同實際聲明的)。Bob的合同終止交易如第5d圖所示。

【0110】實施例情境5：合同條件

【0111】在此說明情況下，Bob與一批建商簽訂了一份合同，交付一份新的財產，並規定了合同中需要獨立簽發的一些條件(首先是批准當地規劃當局的計劃)。這有一個簡單的狀態機模型，如第6a圖所示。第6b圖顯示了這種情況的元數據。第6c圖顯示Bob創建初始合同和兩個子合同(在導出相關子金鑰之後，可使用下面描述的子金鑰產生技術)並將其發佈。第6d圖顯示了何時簽署計劃許可的交易。

【0112】編纂方案

【0113】用於引用合同的元數據可以用各種方式進行格式化，然而，這裡描述了合適的編纂方案。

【0114】如果其定義的權利授予合同的持有人或所有者，合同可轉讓。不可轉讓合同的一個例子是參與者被命名的例子，即將權利賦予特定的命名實體

而不是合同的持有人。該編纂方案只討論可轉讓的合同。

【0115】 令牌代表具體或定義合同賦予的特定合同。在本發明中，令牌是以比特幣交易的形式表示合同。

【0116】 該編纂方法使用包括三個參數或資料項的元數據。此資料可表示：

- i) 根據合同可獲得的股份數量(這可以稱為「NumShares」)；
- ii) 要從發送方轉移到至少一接收者的轉移單元的數量(在本文中稱之為「ShareVal」)；及
- iii) 用於計算轉移單元數量的值的因子(在本文中稱為「掛鉤率(pegging rate)」)。

【0117】 該編纂方案其一優點在於可僅使用上述三個參數來將代碼封裝或表示為區塊鏈上的令牌。實際上，可以使用這三個資料項中的最小值來指定合同。由於該編纂方案可用於任何類型的可轉讓合同，因此可以設計和應用常見的算法。這些元數據項的進一步細節如下提供。

【0118】 可分割令牌是可以將交易輸出上的值細分為跨多個令牌分配的較小數量(即跨多個交易分配)的令牌，原型是令牌化的法定貨幣。可分割合同定義為指定非零PeggingRate的合同。對於可分割合同，交易輸出中轉移的令牌值通過PeggingRate與底層比特幣值相關聯，亦即合同規定了持有人在掛鉤率方面的權利。對於不可分割令牌，沒有PeggingRate，合同規定了持有人在固定價值方面的權利(例如像無記名債券：「本合同可以贖回1000美元」或憑證「本合同可贖回一次理髮」)。對於不可分割合同而言，基本交易比特幣值與合同價值無關。

【0119】 術語「基礎比特幣值」是指附加到交易輸出的比特幣量。在比特幣協議中，每個交易輸出必須有非零比特幣量是被認為有效的。事實上，比特幣金額必須大於設定的最低值(稱為「灰塵」)，在寫入時，目前設定為546 satosis。1比特幣定義為等於1億satoshis。由於比特幣交易在此僅用作促成所有權交換的手段，實際的基本比特幣金額是任意的：真正的價值在於合同規範。理論上，每個令牌都可以被灰塵所攜帶。

【0120】 根據目前的編纂方案，專門針對可分割令牌，底層比特幣值具有一定的意義：其通過PeggingRate與合同價值相關聯。PeggingRate本身是任意的，並且被選擇以便保持底層比特幣量小。使用PeggingRate而不是將每一個含

有灰塵的令牌交易簡單底層化的原因是，因為本發明的協議有助於可分割性：當令牌被分割成較小數量的多個交易輸出時，不需要調整原始合同。相反，每個可分割令牌的合同價值只是根據PeggingRate和底層比特幣值的分割量進行簡單計算。

【0121】 有限的令牌是一個總發行價值由固定的非零數量的股份固定(或「限制」)，由NumShares的數量定義。因此，根據有限合同，不得再發行股票。例如，賽馬的部分所有權合同限於比賽的100%(例如100股每股1%，10股每股10%等)。無限合同意味著發佈者能夠承保進一步發行股票，例如將所需金額的法定貨幣加入其儲備賬戶。NumShares必須在所有合同中明確說明。有限合同必須有 $\text{NumShares} > 0$ ；通過設置 $\text{NumShares} = 0$ 表示無限制合同。

【0122】 典型的例子是貨幣儲備(類似於黃金儲備)，使得儲備銀行賬戶中的總價值與存在的期票中的總值(即未贖回的令牌)相符。此概念超出了貨幣儲備，包括庫存盤點。例如，許可印刷的T恤的令牌發佈者可以從庫存的10,000個T恤庫存開始，並且可發出一可分割令牌來代表那些10,000件T恤(其中，每個股份= 1個T恤)。可以根據由PeggingRate定義的交易輸出的底層比特幣值，將原始令牌細分，每個細分令牌可贖回多件T恤。然而，若需求增加，發佈者可能會決定再發行股票(即增加)流通股數(說)另外10,000股)。在這種情況下，發佈者有義務在其儲備賬戶(即庫存倉庫)內另外存入1萬件T恤，以承保進一步的發行。因此，任何時候股票(股票做為「儲備賬戶」)的T恤總數=未贖回股份的總數。

【0123】 PeggingRates僅適用於可分割合同，其中股份的價值(以ShareVal的數量表示)與潛在的比特幣金額掛鉤。舉例而言，合同可能規定，發佈者承諾以每個底層1 比特幣的10,000美元贖回令牌，這意味著(例如)具有15,400個satoshis的令牌化潛在產值的交易將可以贖回1.54美元。PeggingRate值為0時，表示合同是不可分割的(即只能整數轉移，就像承載債券)。當PeggingRate設置為0(意味著不可忽略的令牌)時，底層比特幣值與合同值無關，可以設置為任何數量。通常在這種情況下，希望保持底層比特幣量盡可能小(即設置為灰塵)，以最小化操作成本。

【0124】 NumShares是根據(有限責任合同)可獲得的總數(固定)股數。對於有限合同，NumShares必須是大於零的整數。對於無限合同，NumShares不是固

定的，因為隨時可以發行更多的股票(只要它們被包銷)，通過將值設置為0來表示。

【0125】 A股被定義為轉移單位，ShareVal是該單位的價值。例如，對於法定貨幣，轉讓單位可以設定為1分。或者，舉例而言，可設置為50美分，在這種情況下，轉移只能以「很多」50美分執行。ShareVal也可以用百分比表示：例如，若育種者想要以10股份的價格出售賽馬，則ShareVal = 10%。ShareVal必須 > 0，且必須在合同上定義。

【0126】 總額為發行股份之總額，該值僅與有限合同有關，因為無限合同發行不固定，可能發行更多股份。若股份以百分比表示，則定義TotalIssuance = 100%。

【0127】 有限合同的NumShares、ShareVal及TotalIssuance與以下方式相關：  

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

【0128】 TotalIssuance的值為0代表其為一無限合同。無限合同的一個例子是法定貨幣(所以TotalIssuance設置為0)；有限合同的例子是：(i)限量紀念幣(鑄造1000枚，1股= 1枚硬幣)：TotalIssuance = 1000 x 1 = 1000枚硬幣；及(ii)在售票處的席位，其中TotalIssuance = 可用席位總數。

【0129】 循環(circulation)被定義為未用令牌的總值(即由未花費交易輸出(UTXO)中的交易確定)。所有未花費交易輸出之完整集合保存在可用於所有比特幣節點的列表中。舉例而言，若發佈者最初發行10,000美元做為法定貨幣類型的令牌，且隨著時間的推移，價值5500美元的令牌被贖回，則流通=4500美元(做為未贖回令牌的值)。該值應與相關儲備賬戶中的餘額調節。

【0130】 子金鑰產生方法

【0131】 上述表3及範例情境所述的情況從一原始(主)金鑰產生子金鑰是有利的。在此提供一種用於執行這一點的方法。

【0132】 第7圖顯示系統1包括通過通訊網路5與第二節點7通訊的第一節點3。第一節點3具有相關聯的第一處理設備23，第二節點5具有相關聯的第二處理設備27，第一和第二節點3、7可包括諸如電腦、電話、平板電腦、移動通信設備、電腦伺服器等的電子設備。在一實施例中，第一節點3可為客戶端(使用者)設備，第二節點7可為伺服器。伺服器可能是電子錢包提供商的伺服器。

【0133】第一節點3與具有第一節點主私鑰( $V_{1c}$ )和第一節點主公鑰( $P_{1c}$ )的第一非對稱加密對相關聯。第二節點7與具有第二節點主私鑰( $V_{1s}$ )和第二節點主公鑰( $P_{1s}$ )的第二非對稱加密對相關聯。換言之，第一和第二節點各自擁有相應的公私金鑰對。

【0134】第一和第二節點3、7之第一和第二非對稱加密對可在註冊過程中產生，例如電子錢包的註冊過程。每個節點的公鑰可以公開共享，例如通過通訊網路5。

【0135】為了在第一節點3和第二節點7兩者確定共同秘密(common secret, CS)，節點3、7分別執行方法300、400的步驟，而不通過通訊網路5與私鑰通訊。

【0136】由第一節點3執行的方法300包括基於至少第一節點主私鑰( $V_{1c}$ )和產生器值(GV)來確定330第一節點第二私鑰( $V_{2c}$ )。產生器值可基於在第一和第二節點之間共享的訊息(M)，其可包括通過通訊網路5共享訊息，下面將進一步詳細描述。方法300還包括至少基於第二節點主公鑰( $P_{1s}$ )和產生器值(GV)確定第二節點第二公鑰( $P_{2s}$ )(步驟370)。方法300包括基於第一節點第二私鑰( $V_{2c}$ )和第二節點第二公鑰( $P_{2s}$ )確定共同秘密(CS)(步驟380)。

【0137】重要的是，亦可藉由方法400在第二節點7處確定相同的共同秘密(CS)。方法400包括基於第一節點主公鑰( $P_{1c}$ )及產生器值(GV)確定430一第一節點第二公鑰( $P_{2c}$ )。方法400還包括基於第二節點主私鑰( $V_{1s}$ )和產生器值(GV)確定470第二節點第二私鑰( $V_{2s}$ )。方法400包括基於第二節點第二私鑰( $V_{2s}$ )和第一節點第二公鑰( $P_{2c}$ )確定480共同秘密(CS)。

【0138】通訊網路5可以包括局域網、廣域網、蜂窩網路、無線電通訊網路、互聯網等，其中可經由諸如電線、光纖或無線的通信介質傳輸資料的這些網路易於竊聽，例如通過竊聽者11。方法300、400可允許第一節點3和第二節點7獨立確定共同秘密，而不在通訊網路5上傳送共同秘密。

【0139】因此，共同秘密(CS)的優點是可被每個節點安全且獨立地確定，而不必通過潛在不安全的通訊網路5發送私鑰。反過來說，共同秘密可以用作秘密金鑰(或做為秘密金鑰的基礎)。

【0140】方法300、400可以包括額外的步驟，請參考第11圖，方法300可在第一節點3處包括基於訊息(M)和第一節點第二私鑰( $V_{2c}$ )產生的簽名訊息(SM1)的

步驟350。方法300還包括步驟360通過通訊網路將第一簽名訊息(SM1)發送到第二節點7。另外，第二節點7可以執行接收第一簽名訊息(SM1)的步驟440。方法400還包括步驟450驗證具有第一節點第二公鑰( $P_{2c}$ )的第一簽名訊息(SM2)，及步驟460基於驗證第一簽名訊息(SM1)的結果認證第一節點3。有利於第二節點7認證所聲稱的第一節點(其中產生了第一簽名訊息的位置)是第一節點3，這是基於僅第一節點3可存取第一節點主私鑰( $V_{1c}$ )，因此只有第一節點3可確定用以產生第一簽名訊息(SM1)的第一節點第二私鑰( $V_{2c}$ )。應當理解，類似地，可以在第二節點7處產生第二簽名訊息(SM2)，並將其發送到第一節點3，使第一節點3可認證第二節點7，諸如在對等情況。

【0141】在第一和第二節點之間共享訊息(M)可以用各種方式實現。在一實施例中，訊息可以在第一節點3處產生，再通過通訊網路5發送到第二節點7，或可在第二節點7處產生訊息，然後通過通訊網路發送到第一節點3。在一些實施例中，訊息(M)可為公開的，因此可通過不安全的網路5發送。一個或多個訊息(M)可儲存在資料儲存器13、17、19中，本領域技術人員將意識到可以用各種方式實現訊息共享。

【0142】有利的是，共同秘密(CS)重新產生之的記錄可在不需將記錄本身被私人儲存或安全地傳送的情況下維持。

【0143】註冊方法100、200

【0144】請參考第9圖所述註冊方法100、200的實施例，其中方法100由第一節點3執行，方法200由第二節點7執行。這包括建立與第一和第二非對稱加密對相應的第一和第二節點3、7。

【0145】非對稱加密對包括相關聯的私鑰和公鑰，例如在公鑰加密中使用的金鑰。在本例中，使用橢圓曲線密碼術(ECC)和橢圓曲線運算的屬性產生非對稱加密對。

【0146】在方法100、200中包括在共同ECC系統上同時使用第一和第二節點110、210並使用基礎值(G)。(注意：基礎值可以稱為公共產生器，但術語「基礎值」用於避免與產生器值GV混淆)。在一實施例中，共同ECC系統可基於做為比特幣使用的ECC系統的secp256K1。基礎值(G)可以被選擇、隨機產生或分配。

【0147】接著回到第一節點3，方法100包括在共同ECC系統和基礎值(G)上

建立110，可包括從第二節點7或第三節點9接收共同ECC系統和基礎值。或者，使用者介面15可以與第一節點3相關聯，由此使用者可以選擇性地提供共同ECC系統及/或基礎值(G)。在另一實施例中，共同ECC系統及/或基礎值(G)中的一個或兩個可由第一節點3隨機選擇。第一節點3可在通訊網路5上發送指示使用共同ECC的通知系統，其具有到第二節點7的基礎值(G)。繼而，第二節點7可以通過發送指示使用共同ECC系統和基礎值(G)的確認的通知來確定210。

【0148】方法100還包括120第一節點3產生包括第一節點主私鑰( $V_{1c}$ )和第一節點主公鑰( $P_{1c}$ )的第一非對稱加密對。這包括至少部分地基於在共同ECC系統中指定的允許範圍內的隨機整數來產生第一節點主私鑰( $V_{1c}$ )。還包括根據以下公式基於第一節點主私鑰( $V_{1c}$ )和基礎值(G)的橢圓曲線點乘法來確定第一節點主公鑰( $P_{1c}$ )：

$$P_{1c} = V_{1c} \times G \quad (\text{公式1})$$

因此，第一非對稱加密對包括：

$V_{1c}$ ：由第一個節點保密的第一節點主私鑰。

$P_{1c}$ ：公開的第一節點主公鑰。

【0149】第一節點3可以將第一節點主私鑰( $V_{1c}$ )和第一節點主公鑰( $P_{1c}$ )儲存在與第一節點3相關聯的第一資料儲存器13中。為了安全起見，第一節點主私鑰( $V_{1c}$ )可以儲存在第一資料儲存器13的安全部分中，以確保金鑰保持私有。

【0150】方法100還包括130通過通訊網路5將第一節點主公鑰( $P_{1c}$ )發送到第二節點7，如第9圖所示。步驟220第二節點7可以在接收第一節點主公鑰( $P_{1c}$ )時，將第一節點主公鑰( $P_{1c}$ )儲存在與第二節點7相關聯的第二資料儲存器17中(步驟230)。

【0151】與第一節點3類似，第二節點7的方法200包括產生240包括第二節點主私鑰( $V_{2s}$ )和第二節點主公鑰( $P_{2s}$ )的第二非對稱加密對。第二節點主私鑰( $V_{2s}$ )也是允許範圍內的隨機整數。反過來說，第二節點主公鑰( $P_{2s}$ )由以下公式確定：

$$P_{2s} = V_{2s} \times G \quad (\text{公式2})$$

因此，第二非對稱加密對包括：

$V_{2s}$ ：由第二節點保密的第二節點主私鑰。

$P_{1s}$ ：公開的第二節點主公鑰。

【0152】 第二節點7可以將第二非對稱加密對儲存在第二資料儲存器17中。方法200還包括將第二節點主公鑰( $P_{1s}$ )發送到第一節點3(步驟250)，反過來說，第一節點3可接收140 並儲存150第二節點主公鑰( $P_{1s}$ )。

【0153】 應當理解，在一些替代方案中，可以在與第三節點9(例如可信第三方)相關聯的第三資料儲存器19處接收和儲存相應的主公鑰。這可能包括充當公共目錄的第三方，如證書頒發機構。因此，在一些實施例中，當僅確定需要共同秘密(CS)時，第二節點7才能請求和接收第一節點主公鑰( $P_{1c}$ )(反之亦然)。

【0154】 註冊步驟可能僅需要做為初始設置發生一次。

【0155】 會談初始和利用第一節點3判斷共同秘密

【0156】 請參考第10圖所述之確定共同秘密(CS)的實施例。共同秘密(CS)可以用於第一節點3和第二節點7之間的特定會談、時間、交易或其他目的，使用相同的共同秘密(CS)可能是不可取或不可靠的。因此，共同秘密(CS)可在不同的會談、時間、交易等之間改變。

【0157】 下面提供了上述之安全傳輸技術的說明。

【0158】 步驟310產生一訊息(M)

【0159】 在該範例中，由第一節點3執行的方法300包括產生310訊息(M)。訊息(M)可以是隨機、偽隨機或使用者定義的。在一範例中，訊息(M)基於Unix時間和隨機數(和任意值)。例如，訊息(M)可以被提供為：

$$\text{Message (M)} = \text{UnixTime} + \text{nonce} \quad (\text{公式3})$$

【0160】 在一些範例中，訊息(M)是任意的。然而，應當理解，訊息(M)可以具有在一些應用中可能有用的選擇性值(諸如Unix時間等)。

【0161】 方法300包括步驟315通過通訊網路5將訊息(M)發送到第二節點7。當訊息(M)不包括關於私鑰的資訊時，訊息(M)可通過不安全的網路發送。

【0162】 步驟320判斷一產生器值

【0163】 方法300還包括基於訊息(M)確定產生器值(GV)的步驟320。在該範例中，這包括確定訊息的密碼雜湊。密碼雜湊算法的一個例子包括SHA-256以產生256位元產生器值(GV)，其為：

$$GV = \text{SHA-256}(M) \quad (\text{公式4})$$

【0164】 應當理解，可使用其他雜湊算法，這可能包括安全雜湊算法(SHA)系列中的其他雜湊算法。一些特定範例包括SHA-3子集，包括SHA3-224、SHA3-256、SHA3-384、SHA3-512、SHAKE128、SHAKE256。其他雜湊算法可包括RACE完整原語評估訊息摘要(RACE Integrity Primitives Evaluation Message Digest, RIPEMD)家族。一具體實施例可包括RIPEMD-160。其他雜湊函數可包括基於Zémor-Tillich雜湊函數和基於背包(knapsack)的雜湊函數的族。

【0165】 步驟330判斷一第一節點第二私鑰

【0166】 接著，方法300包括基於第一節點主私鑰( $V_{1c}$ )和產生器值(GV)確定第一節點第二私鑰( $V_{2c}$ )的步驟330。依據以下公式的第一節點主私鑰( $V_{1c}$ )和產生器值(GV)的純量加法：

$$V_{2c} = V_{1c} + GV \quad (\text{公式5})$$

【0167】 因此，第一節點第二私鑰( $V_{2c}$ )不是隨機值，而是從第一節點主私鑰確定性地導出。加密對中的相應公鑰、即第一節點第二公鑰( $P_{2c}$ )，具有以下關係：

$$P_{2c} = V_{2c} \times G \quad (\text{公式6})$$

將公式5中的 $V_{2c}$ 加入到公式6中：

$$P_{2c} = (V_{1c} + GV) \times G \quad (\text{公式7})$$

其中+運算符指的是純量加法， $\times$ 運算符是指橢圓曲線點乘法。注意橢圓曲線加密代數是分散式，公式7可以表示為：

$$P_{2c} = V_{1c} \times G + GV \times G \quad (\text{公式8})$$

最後，公式1可以被替換為公式7以提供：

$$P_{2c} = P_{1c} + GV \times G \quad (\text{公式9.1})$$

$$P_{2c} = P_{1c} + \text{SHA-256}(M) \times G \quad (\text{公式9.2})$$

【0168】 在公式8到9.2中，+運算符是指橢圓曲線點加法。因此，給定第一節點主公鑰( $P_{1c}$ )和訊息(M)的資訊可導出對應的第一節點第二公鑰( $P_{2c}$ )。第二節點7可以具有這樣的知識來獨立地確定第一節點第二公鑰( $P_{2c}$ )，將在方法400下面進一步詳細討論。

【0169】 步驟350依據訊息和第一節點第二私鑰產生第一簽名訊息( $SM1$ )

【0170】 方法300還包括基於訊息(M)和確定的第一節點第二私鑰( $V_{2c}$ )產生

第一簽名訊息(SM1)之步驟350。產生簽名訊息包括應用數字簽名算法對訊息(M)進行數字簽名。在一範例中，這包括在橢圓曲線數字簽名算法(ECDSA)中將第一節點第二私鑰( $V_{2c}$ )應用於訊息以獲得第一簽名訊息(SM1)。ECDSA的範例包括基於具有secp256k1、secp256r1、secp384r1、secp521r1的ECC系統。

【0171】 第一簽名訊息(SM1)可利用第二節點7對應的第一節點第二公鑰( $P_{2c}$ )來驗證。第二節點7可使用第一簽名訊息(SM1)之驗證來認證第一節點3，將在下面的方法400中討論。

【0172】 *步驟370' 判斷一第二節點第二公鑰*

【0173】 然後，第一節點3可以確定第二節點第二公鑰( $P_{2s}$ )，如步驟370。如上所述，第二節點第二公鑰( $P_{2s}$ )可至少基於第二節點主公鑰( $P_{1s}$ )和產生器值(GV)。在此例中，由於公鑰被確定為具有與基礎值(G)相乘的橢圓曲線點的私鑰(步驟370')，所以可用類似於公式6的方式來表示第二節點第二公鑰( $P_{2s}$ )如下：

$$P_{2s} = V_{2s} \times G \quad (\text{公式10.1})$$

$$P_{2s} = (P_{1s} + GV) \times G \quad (\text{公式10.2})$$

【0174】 公式10.2的數學證明與上述相同，用於為第一節點第二公鑰( $P_{2c}$ )導出公式9.1。應當理解，步驟370中第一節點3可獨立確定第二節點7的第二節點第二公鑰。

【0175】 *步驟380 判斷第一節點3的共同秘密*

【0176】 接著，第一節點3可基於已確定的第一節點第二私鑰( $V_{2c}$ )和已確定的第二節點第二公鑰( $P_{2s}$ )來確定共同秘密(CS)，如步驟380。共同秘密(CS)可由第一節點3通過以下公式確定：

$$S = V_{2c} \times P_{2s} \quad (\text{公式11})$$

【0177】 *在第二節點7上執行方法400*

【0178】 現在將描述在第二節點7執行的相應方法400。應當理解，這些步驟中部分類似於由第一節點3執行的步驟。

【0179】 方法400包括通過通訊網路5從第一節點3接收訊息(M)之步驟410，可包括在步驟315由第一節點3發送的訊息(M)。接著步驟420第二節點7基於訊息(M)確定產生器值(GV)。由第二節點7確定產生器值(GV)的步驟類似於上述由第一節點執行的步驟320。在該實施例中，第二節點7獨立於第一節點3執行該確定

步驟420。

【0180】 下一步驟包括基於第一節點主公鑰( $P_{1c}$ )和產生器值(GV)確定430第一節點第二公鑰( $P_{2c}$ )。在該範例中，由於公鑰被在步驟430'確定為具有與基礎值(G)的橢圓曲線點相乘的私鑰，故可用類似於公式9的方式來表達第一節點第二公鑰( $P_{2c}$ )如下：

$$P_{2c} = V_{2c} \times G \quad (\text{公式12.1})$$

$$P_{2c} = P_{1c} + GV \times G \quad (\text{公式12.2})$$

公式12.1和12.2的數學證明與上面公式10.1和10.2討論的數學證明相同。

【0181】 *第二節點7認證第一節點3*

【0182】 方法400可包括由第二節點7執行以認證第一節點3是第一節點3的步驟。如前所述，這包括步驟440從第一節點3接收第一簽名訊息(SM1)。第二節點7接著可使用在步驟430確定的第一節點第二公鑰( $P_{2c}$ )來驗證第一簽名訊息(SM1)上的簽名，如步驟450。

【0183】 數位簽名的驗證可根據如上所述的橢圓曲線數字簽名算法(ECDSA)來完成，重要的是，由於 $V_{2c}$ 和 $P_{2c}$ 形成加密對，故用第一節點第二私鑰( $V_{2c}$ )簽名的第一個簽名訊息(SM1)只能被對應的第一個節點第二個公鑰( $P_{2c}$ )正確驗證。由於這些金鑰對於在第一節點3的註冊時產生的第一節點主私鑰( $V_{1c}$ )和第一節點主公鑰( $P_{1c}$ )是確定性的，所以可使用驗證第一簽名訊息(SM1)做為認證發送第一簽名訊息(SM1)的第一節點與註冊時的第一節點3相同。因此，第二節點7還可基於驗證(450)第一簽名訊息的結果來執行認證(460)第一節點3的步驟。

【0184】 *第二節點7決定共同秘密*

【0185】 方法400還可以包括第二節點7基於第二節點主私鑰( $V_{1s}$ )和產生器值(GV)確定第二節點第二私鑰( $V_{2s}$ )(步驟470)。類似於由第一節點3執行的步驟330，第二節點第二私鑰( $V_{2s}$ )可基於根據以下公式的第二節點主私鑰( $V_{1s}$ )和產生器值(GV)的純量相加：

$$V_{2s} = V_{1s} + GV \quad (\text{公式13.1})$$

$$V_{2s} = V_{1s} + \text{SHA-256}(M) \quad (\text{公式13.2})$$

接著，第二節點7可獨立於第一節點3之外，依據以下公式，於步驟480基於第二節點第二私鑰( $V_{2s}$ )和第一節點第二公鑰( $P_{2c}$ )確定共同秘密(CS)：

$$S = V_{2s} \times P_{2c} \quad (\text{公式14})$$

【0186】 由第一節點3和第二節點7確定的共同秘密(CS)的證明

【0187】 由第一節點3確定的共同秘密(CS)與在第二節點7處確定的共同秘密(CS)相同。接著將描述公式11和公式14提供相同共同秘密(CS)的數學證明。

【0188】 由第一節點3確定的共同秘密(CS)，公式10.1可以被替換為公式11如下：

$$S = V_{2c} \times P_{2s} \quad (\text{公式11})$$

$$S = V_{2c} \times (V_{2s} \times G)$$

$$S = (V_{2c} \times V_{2s}) \times G \quad (\text{公式15})$$

【0189】 由第二節點7確定的共同秘密(CS)，公式12.1可以如下代入公式14：

$$S = V_{2s} \times P_{2c} \quad (\text{公式14})$$

$$S = V_{2s} \times (V_{2c} \times G)$$

$$S = (V_{2s} \times V_{2c}) \times G \quad (\text{公式16})$$

【0190】 由於ECC代數是可交換的，故公式15和公式16是等價的，因為：

$$S = (V_{2c} \times V_{2s}) \times G = (V_{2s} \times V_{2c}) \times G \quad (\text{公式17})$$

【0191】 共同秘密(CS)和秘密金鑰(secret key)

【0192】 共同秘密(Common secret, CS)可做為秘密金鑰，或做為在第一節點3和第二節點7之間安全通訊的對稱式金鑰演算法中之金鑰的基礎。

【0193】 共同秘密(CS)可以是橢圓曲線點( $x_s$ ,  $y_s$ )的形式，其可使用由節點3、7所約定的標準公知操作來轉換成標準密鑰格式。例如， $x_s$ 值可為AES256加密之256位整數金鑰。對於需要此長度金鑰的任何應用，也可以使用RIPEMD160將其轉換為160位整數。

【0194】 共同秘密(CS)可根據需要確定。重要的是，第一節點3不需要儲存共同秘密(CS)，因為其可基於訊息(M)重新確定。在一些範例中，所使用的訊息(M)可以儲存在資料儲存器13、17、19(或其他資料儲存器)中，而與主私鑰所需的安全級別相同。在一些範例中，訊息(M)可以是公開的。然而，根據一些應用，若共同秘密(CS)被保持為與第一節點主私鑰( $V_{1c}$ )一樣安全，則可將共同秘密(CS)儲存在與第一節點相關聯的第一資料儲存器(X)中。

【0195】該技術可以於基於單一主私鑰加密對來確定可對應於多個安全秘密金鑰的多個共同秘密，是有利的。

【0196】產生器值的層級(金鑰)

【0197】舉例而言，可確定一系列連續的產生器值(GV)，其中可以基於先前的產生器值(GV)來確定每個連續的GV。例如，代替重複步驟310至370及410至470以產生連續的單用途金鑰，通過節點之間的事先協商，先前使用的產生器值(GV)可由雙方重複重新建立產生器值的層級。實際上，基於訊息(M)的雜湊的產生器值可用於下一代產生器值(GV')的下一代訊息(M')，如此一來，可使連續幾代的共享秘密被計算，而不需要進一步的協議建立傳輸，特別是針對每一代共同秘密的多個訊息的傳輸。下一代通用秘密(CS')可以如下計算。

【0198】首先，第一節點3和第二節點7皆獨立地確定產生器值(GV')的下一代。其類似於步驟320和420，但適用於以下公式：

$$M' = \text{SHA-256}(M) \quad (\text{公式18})$$

$$GV' = \text{SHA-256}(M') \quad (\text{公式19.1})$$

$$GV' = \text{SHA-256}(\text{SHA-256}(M)) \quad (\text{公式19.2})$$

【0199】接著，第一節點3可確定類似於上述步驟370和330的第二節點第二公鑰( $P_{2s}'$ )和第一節點第二私鑰( $V_{2c}'$ )的下一代，但使用以下公式進行調整：

$$P_{2s}' = P_{1s} + GV' \times G \quad (\text{公式20.1})$$

$$V_{2c}' = V_{1c} + GV' \quad (\text{公式20.2})$$

【0200】然後，第二節點7可以確定類似於上述步驟430和470的第一節點第二公鑰( $P_{2c}'$ )和第二節點第二私鑰( $V_{2s}'$ )的下一代，但是使用以下公式進行調整：

$$P_{2c}' = P_{1c} + GV' \times G \quad (\text{公式21.1})$$

$$V_{2s}' = V_{1s} + GV' \quad (\text{公式21.2})$$

【0201】接著，第一節點3和第二節點7可各自確定下一代共同秘密(CS')。特別地，第一節點3以下式確定下一代共同秘密(CS')：

$$CS' = V_{2c}' \times P_{2s}' \quad (\text{公式22})$$

【0202】第二節點7以下式確定下一代共同秘密(CS')：

$$CS' = V_{2s}' \times P_{2c}' \quad (\text{公式23})$$

【0203】可以用相同的方式計算更多代(CS''、CS'''等)來創建鏈層級。該技術要求第一節點3和第二節點7都跟蹤原始訊息(M)或原始計算的產生器值(GV)，以及與之相關的節點。由於這是公開的資訊，所以沒有關於保留這些資訊的安全問題。因此，這些資訊可能會保留在「雜湊表」(將雜湊值鏈接到公共密鑰)上並通過網路5自由分發(例如使用洪流)。此外，若層級結構中的任何單獨的共同秘密(CS)受到損害，若私鑰 $V_{1c}$ 、 $V_{1s}$ 保持安全，則不會影響層級中任何其他常見秘密的安全性。

#### 【0204】金鑰的樹狀結構

【0205】除了如上所述的鏈(線性)層級之外，可創建樹狀結構形式的層級結構。使用樹狀結構，可確定用於不同目的各種金鑰，例如認證金鑰、加密金鑰、簽名金鑰、支付金鑰等，由此這些金鑰都鏈接到單個安全維護的主私鑰，這在第12圖中已最好地顯示了具有各種不同金鑰的樹狀結構901，其中的每一個都可用來與另一方共享秘密。樹狀分支可以通過幾種方式完成，其中三種如下所述。

#### 【0206】(i)主金鑰生殖(Master key spawning)

【0207】在鏈層級中，通過向原始主私鑰添加乘法重新產生的訊息來創建每個新的「連結」(公鑰/私鑰對)。例如，(為了清楚，僅顯示第一節點3的私鑰)：

$$V_{2c} = V_{1c} + \text{SHA-256}(M) \quad (\text{公式24})$$

$$V_{2c}' = V_{1c} + \text{SHA-256}(\text{SHA-256}(M)) \quad (\text{公式25})$$

$$V_{2c}'' = V_{1c} + \text{SHA-256}(\text{SHA-256}(\text{SHA-256}(M))) \quad (\text{公式26})$$

... 等等。

【0208】要創建分支，可以使用任何鏈做為子主私鑰。例如 $V_{2c}'$ 可以做為子主私鑰( $V_{3c}$ )，通過向常規主私鑰增加雜湊：

$$V_{3c} = V_{2c}' + \text{SHA-256}(M) \quad (\text{公式27})$$

【0209】子主私鑰( $V_{3c}$ )本身可以具有下一代密鑰( $V_{3c}'$ )，例如：

$$V_{3c}' = V_{3c} + \text{SHA-256}(\text{SHA-256}(M)) \quad (\text{公式28})$$

這提供了使用如第13圖所示主私鑰產生方法的樹狀結構903。

#### 【0210】(ii)邏輯組合

【0211】在這種方法中，樹中的所有節點(公/私鑰對)被產生為鏈(或以任何

其他方式)，並且樹中的節點之間的邏輯關係由樹中的每個節點維護使用指標簡單地與樹中的父節點相關聯。因此，該指標可用於確定會談的共同秘密(CS)相關的公鑰/私鑰對。

**【0212】** (iii)訊息多重性

**【0213】** 通過在鏈或樹中的任何一點引入新訊息，可產生新的私鑰/公鑰對。訊息本身可為任意的或可具有某種意義或功能(例如可能與「真實」銀行帳號相關等)。可能需要安全地保留用於形成新的私鑰/公鑰對的這種新訊息。

**【0214】** 說明本發明使用的計算代理

**【0215】** 本發明可利用計算資源或代理來執行合同過程的自動化方面。下面提供了合適的代理的範例，儘管可以使用其他方式實現。

**【0216】** 代理可以與區塊鏈結合操作，在圖靈機的實現中將其用作不可擦除的磁帶。該代理與區塊鏈網路並行運作，監督和處理(循環(looping))程序的執行。循環過程旨在執行給定的任務，例如程序的自動化或設備或系統的控制。此種平行的資源監視區塊鏈的狀態，可能導致交易被寫入區塊鏈。在某種意義上，它使用區塊鏈做為圖靈機的不可擦除帶，具有以下定義和特徵：

1. 區塊鏈做為圖靈機的磁帶。區塊鏈中的每一交易皆表示磁帶上的一單元。該單元可以包含有限字母表中的符號。
2. 磁帶頭可從已經寫入區塊鏈的區塊中讀取資訊。
3. 磁帶頭可將包含許多交易的新區塊寫入區塊鏈的末端，但不能寫入已經存在的區塊。因此，區塊鏈磁帶是不可擦除的。
4. 每個交易的元數據可被儲存為多簽名付費腳本雜湊(P2SH)交易的一部分。

**【0217】** 代理的一個重要功能是充當一個可監視區塊鏈當前狀態的自動化實體。其還可從任何區塊鏈外的資源接收信號或輸入。根據區塊鏈狀態及/或接收到的輸入，代理可執行某些動作。代理決定要執行哪些動作可能涉及也可能不涉及「現實世界」(即禁區)及/或區塊鏈行動(例如創建和廣播新交易)的行為。代理所採取的操作可能由區塊鏈狀態觸發。代理還可決定將要廣播到比特幣網路的下一組交易，然後寫入區塊鏈。

**【0218】** 代理的行動平行運作，同時運行到區塊鏈(比特幣)網路。在某種意義上，這擴展了區塊鏈(如比特幣)腳本的功能。這種連續監視實現了「循環」

控制流結構，使組合代理和區塊鏈系統圖靈完備。

【0219】圖靈機包括兩個堆疊：

- 資料堆疊：由區塊鏈表示，如上所述。
- 控制堆疊：由代理功能表示，其儲存與重複控制流功能有關的資訊。

【0220】控制堆疊與資料堆疊的分離提供了可防止比特幣核心發生無限循環之優點，從而減輕了拒絕服務攻擊。

【0221】代理之管理和運行可通過任何類型的循環構造(例如FOR-NEXT、REPEAT UNTIL等)循環的子程序。本文描述的實施例包括使用「重複」構造的一實施例的過程。使用者可指定索引(i)和限制(J)。這些代表當前迭代次數(通常從0開始計數)和重複循環的總次數。

【0222】對於每次迭代(iteration)：

1. 索引遞增1。對於退出條件，當索引達到限制時，迭代將停止
2. 執行包含“if condition then action”(ICTA)語句的代碼區塊；該動作可為在區塊鏈上或區塊鏈外的任何動作；
3. 計算該子程序的加密雜湊。其可做為交易的一部分儲存在區塊鏈中。由於雜湊對於每個代碼是唯一的，故其可驗證哪個代碼已被使用。

【0223】循環體包括一代碼區塊，每個代碼區塊皆包含一個“If condition then action”(ICTA)語句。這將監視區塊鏈的當前狀態，以滿足以下條件：

- 開始或觸發條件(例如達到特定日期)。
- 重複條件(即與先前迭代相關聯的元數據或雜湊)。
- 停止條件(即循環的最後一次迭代)。

【0224】ICTA語句使代理可根據區塊鏈的當前狀態來決定下一個交易。進行下一個交易涉及將交易廣播到比特幣網路上，並將新交易寫入區塊鏈。此動作係做為該迭代執行的記錄。一旦交易已被寫入區塊鏈，管理器將隨後發現先前的迭代已執行並寫入區塊鏈，並將執行下一個迭代。當索引(i)達到代碼區塊中指定的限制(J)時，下一個會繼續，直到重複循環退出。

【0225】每個交易皆以可重複使用的方式保存在區塊鏈中。在比特幣實現中，交易中的每個簽名都附加了一個SIGHASH標誌，此標誌可採用不同的值，每個值都表示可修改交易的其他部分，而不涉及該簽名的所有者。一可重複使

用的交易在其中一個交易輸入中有SIGHASH標誌「SigHash\_AnyoneCanPay」，允許任何人對交易的投入做出貢獻。此參數使代理的ICTA功能可執行並重複多次，且具有不同的輸入。該功能的使用可以限於授權方 - 例如通過可重複使用交易的版權。

【0226】 ICTA代碼區塊的“if condition”部分可監視任何類型的條件，其類似於其他編程語言(例如C、C++、Java)，而不限於儲存在區塊鏈上的資訊。例如，可監視日期和時間(若達到一定的日期和時間)或監測天氣(即當溫度低於攝氏10度並正在下雨時)、監視合同或信託的條件(即當A公司購買B公司時)。

【0227】 ICTA代碼區塊中的“Then action”部分可執行多個操作。本發明不限於可以採取的動作的數量或類型。該操作不限於區塊鏈上的交易，儘管包含與該操作相關的元數據的交易可能會寫入區塊鏈。

【0228】 元數據可為任何形式。然而，在一實施例中，元數據可將一超連結儲存到一檔案中，此檔案包含更多與該動作相關的資料或指令。元數據可將包含更多資料或與該動作有關的指令的雜湊表的超連結以及做為雜湊表的查找關鍵字之動作的雜湊儲存在一起。

【0229】 代理人的控制堆疊可以用特定於每個使用者之需要的多種方式來實現。例如，控制堆疊的重複循環可以基於任何圖靈完備語言。一種可能的語言選擇是基於Forth風格的堆疊語言，使用此種語言的一優點為，其使編程風格與已知和廣泛使用的比特幣腳本保持一致。

【0230】 使用比特幣腳本的備用堆疊做為資料儲存空間

【0231】 比特幣腳本包含命令，也稱為操作代碼，讓使用者可將資料移動到替代堆疊上，稱為「alt堆疊」。

【0232】 操作碼為：

- OP\_TOALTSTACK - 將資料從主堆疊的頂部移動到alt堆疊的頂部。
- OP\_FROMALTSTACK - 將資料從alt堆疊的頂部移動到主堆疊的頂部。

【0233】 這使得來自中間計算步驟的資料可儲存在alt堆疊中，類似於允許將資料儲存在計算器上的「儲存器」功能。在一實施例中，alt堆疊用於配置比特幣腳本以解決小計算任務並在計算中返回結果。

【0234】 使用一代碼註冊表管理代理

【0235】該代理還管理其擁有並運行的所有代碼的註冊表，此註冊表的結構類似於將特定金鑰映射到特定值的查找表或字典。金鑰和值的配對分別由代碼區塊(H<sub>1</sub>)的雜湊和儲存代碼的IPv6地址表示。要使用金鑰H<sub>1</sub>檢索代碼區塊時，查找表用於檢索關聯值(此為儲存代碼的位置)，並相應檢索源代碼。代碼註冊表的執行可以有所不同。

【0236】代理代碼的交易元數據及循環的重新產生

【0237】在特定迭代中重新產生代理循環所需的資訊，其做為元數據儲存在區塊鏈記錄的交易中。

【0238】以這種方式，區塊鏈上的交易儲存或提供關於循環的給定迭代的存取資訊係在代理上執行。該資訊可包括與循環相關聯的任何變量的值，例如索引i，以及任何其他必需資訊，例如在代碼區塊中使用的參數值或位置相關資料，所需資訊可被存取。

【0239】元數據本身做為多簽名付費腳本雜湊腳本(P2SH)的一部分儲存在交易中。使用交易記錄的元數據還可記錄過去代碼執行的審計跟蹤。

【0240】代理可在每次迭代中重新產生重複循環代碼區塊的多種方法。代碼區塊可能被硬編碼到代理程序本身中，或可儲存在私有或公開可用的文件中，或儲存為專用或公共雜湊表文件中的條目，或上述之組合。代碼區塊可為具有硬編碼變量的靜態，或可為靜態的但包含可填充的參數。參數可以是任何資料格式的單一值，亦可為小塊代碼，或是上述之組合。這些參數可通過直接從交易中的元數據(例如比特幣交易)或外部來源(例如內部資料庫、私人/公共文件或雜湊表)上述的任何組合中獲取來填充參數。指向外部參數值的指標可能儲存在交易的元數據中。

【0241】以下步驟提供了代理如何在第i次迭代中重新產生重複循環代碼區塊的一實施例。在此實施例中，代碼註冊表為雜湊表，其中雜湊值做為表的查找關鍵字，並儲存在交易的元數據中。

1. 代理監視區塊鏈中的交易，交易包含與代碼註冊表中條目匹配之代碼區塊的雜湊。
2. 代理發現包含相應雜湊(H<sub>1</sub>)的交易。
3. 代理讀取「元數據代碼雜湊(Metadata-CodeHash)」，取得CodeHash字段以獲取

H1，並使用其檢索代碼(C1)。若RIPEMD-160(SHA256(C1))等於H1，則代碼未被更改，可以繼續進行下一步。

4. 代理讀取儲存有索引I的「元數據代碼雜湊」，並在第i次迭代中重新產生代碼。換言之，循環在適當的迭代被「重新加載」。
5. 使用者的簽名包含在P2SH命令中，以驗證元數據的來源。
6. 代理讀取元數據輸出雜湊 (Metadata-OutputHash) 及元數據輸出指標 (Metadata-OutputPointer)以檢索先前步驟的輸出，若循環的這個迭代需要這些資料。

【0242】 應當注意，上述實施例說明並非限制本發明，且本領域技術人員將能夠設計許多備選實施例而不脫離由所附申請專利範圍限定之本發明範圍。在申請專利範圍中，放置在括號中的任何附圖標記不應被解釋為限制申請專利範圍。詞語「包括」及「包含」等不排除除了任何申請專利範圍或說明書做為整體列出的元件或步驟之外的元件或步驟的存在。元素的單數引用並不排除這些元素的複數引用，反之亦然。本發明可通過包括幾個不同元件的硬體，以及借助於適當編程的電腦來實現。在列舉了若干裝置的設備請求項中，這些裝置的其中幾個可以由同一個硬體元件實現。在相互不同的附屬項中記載某些措施的事實並不表示這些措施的組合不能有效使用。

#### 【符號說明】

##### 【0243】

- 3 第一節點
- 5 通訊網路
- 7 第二節點
- 9 第三節點
- 11 竊聽者
- 13、17、19 資料儲存器
- 15 使用者介面
- 23 第一處理設備
- 27 第二處理設備



201732706

申請日: 106/02/21

IPC分類: G06Q 20/40 (2012.01)

**【發明摘要】****【中文發明名稱】**

區塊鏈執行智能化合同的註冊及自動化管理方法

**【英文發明名稱】**

Registry and Automated Management Method for Blockchain-Enforced Smart Contracts

**【中文】**

本發明涉及一種令牌化、區塊鏈及智能化合同技術，其提供可將合同自動化管理簡化之技術安排。本發明包括使用基於電腦的儲存庫來儲存合同之方法及系統，接著將合同由區塊鏈上的交易表示，交易腳本中的元數據包括合同的雜湊值及在儲存庫中標識合同之位置。交易亦包括一未花費的交易輸出(UTXO)，代表狀態為開放(即尚未終止)合同。在稍後的時間點藉由花費該交易輸出以終止合同，例如使用nLockTime + CheckLockTimeVerify(CLTV)。通過將該概念與其他技術和計算組件結合，本發明可提供一種強大的機制來實現各種任務，如更新或滾動合同，或將其劃分為複數子合同或複數條件。此外，由於合同的地位和存在是通過該區塊鏈之證明，因此提供了永久、公開可見且不可變的合同記錄。

**【英文】**

The invention relates to the fields of tokenisation, blockchain and smart contract technologies. It provides a technical arrangement which simplifies the automated management of contracts. The invention comprises a method and system which use a computer-based repository for storage of the contract. The contract is then represented by a transaction on the blockchain. Metadata within the transaction's script includes a hash of the contract and a means of identifying its location within the repository. The transaction also includes an unspent output (UTXO) which indicates its status as an open (ie not terminated) contract. The contract is terminated by spending the output at a later point in time, for example, using nLockTime + CheckLockTimeVerify (CLTV). By combining this concept with other techniques and computing components, the invention can provide a powerful mechanism for implementing various tasks such as renewing or rolling over the contract, or dividing it into sub-contracts or conditions. Furthermore, as the status and existence of the contract is evidence via the blockchain, this provides a

permanent, publicly visible and non-alterable record of the contract.

【指定代表圖】：第 1 圖。

【代表圖之符號簡單說明】

無

## 【發明申請專利範圍】

【第1項】一種控制合同之可見性及效能之電腦執行方法，該方法包括下列步驟：

- (a) 將一合同儲存於設於一電腦中之一儲存庫；
- (b) 將一交易廣播到一區塊鏈，該交易包括：
  - i) 至少一未花費交易輸出(unspt transaction output, UTXO)；以及
  - ii) 元數據，包括指示儲存該合同之位置的一識別碼(identifier)；以及
- (c) 更新或滾動該合同，包括下列步驟：

利用一先前金鑰之相關資料產生一新金鑰，該先前金鑰與該合同相關聯；  
產生一腳本，其包括該新金鑰、該合同之該位置及該合同的一雜湊；以及  
支付一數量之貨幣給該腳本。

【第2項】如請求項 1 所述之方法，其中該交易更包括一確定性(deterministic)贖回腳本位址，其為一支付腳本雜湊(pay-to-script-hash, P2SH)位址。

【第3項】如請求項 2 所述之方法，更包括利用向該區塊鏈廣播一額外交易(further transaction)以花費該UTXO以終止該合同之步驟。

【第4項】如上述請求項任一項所述之方法，其中該額外交易包括：  
一輸入，其為該輸出(UTXO)；以及  
一解鎖腳本，包含一簽名；該元數據；以及一公鑰。

【第5項】如上述請求項任一項所述之方法，其中該合同之定義為：  
i) 至少一條件；以及  
ii) 至少一行為，其效能取決於該條件的評估。

【第6項】如上述請求項任一項所述之方法，其中該元數據包括：  
i) 一位址或表示一位址，為該合同儲存在該電腦中之該儲存庫之位置；以及/或  
ii) 該合同的一雜湊。

【第7項】如上述請求項任一項所述之方法，更包括下列步驟：  
藉由判斷該UTXO是否在該區塊鏈的一UTXO清單中，以確認該合同是否已終止。

【第8項】如上述請求項任一項所述之方法，其中該合同係儲存於一分散式雜湊表(Distributed Hash Table, DHT)中。

【第9項】如上述請求項任一項所述之方法，更包括下列步驟：

向該區塊鏈廣播一交易，包括一指令，用以在一指定日期及/或時間花費該輸出(UTXO)，該指令為一CheckLockTimeVerify指令。

【第10項】如上述請求項任一項所述之方法，其中獲得該合同之部分或全部內容僅限於至少一指定的授權方。

【第11項】如上述請求項任一項所述之方法，其中該合同包括一確定性有限自動器(Deterministic Finite Automaton, DFA)，以實現該合同。

【第12項】如請求項11所述之方法，其中該確定性有限自動器係使用一法律編纂方案(codification scheme)定義。

【第13項】如請求項11或12所述之方法，其中該確定性有限自動器係使用：

- i) 至少一區塊鏈交易，其利用一腳本語言；
- ii) 一計算代理商，其用以監視該區塊鏈之狀態；及/或
- iii) 一數位錢包的一組指令。

【第14項】一種控制合同之可見性及效能之電腦執行方法，該方法包括下列步驟：

- (a) 將一合同儲存於設於一電腦中之一儲存庫；
- (b) 將一交易廣播到一區塊鏈，該交易包括：
  - i) 至少一未交易之輸出(unspent output, UTXO)；以及
  - ii) 元數據，包括指示儲存該合同之位置的一識別碼(identifier)；以及
- (c) 從該合同中衍化(derived)產生一子合同，該子合同與一確定性位址相關聯並由以下產生：
  - iii) 產生一新金鑰，其利用一種子進行衍化；
  - iv) 通過該合同的一參考，將該子合同儲存於該儲存庫，並向該區塊鏈廣播包含一腳本的一交易，該腳本包括該參考；以及
  - v) 將該子合同的一參考加入到現存之該合同的該元數據中。

【第15項】如請求項14所述之方法，其中該交易更包括一確定性(deterministic)贖回腳本位址，其為一支付腳本雜湊(pay-to-script-hash, P2SH)位址。

【第16項】如請求項15所述之方法，更包括利用向該區塊鏈廣播一額外交易(further transaction)以花費該UTXO以終止該合同之步驟。

【第17項】如請求項 14 至 16 所述之方法，其中該額外交易包括：  
一輸入，其為該輸出(UTXO)；以及  
一解鎖腳本，包含一簽名；該元數據；以及一公鑰。

【第18項】如請求項 14 至 17 所述之方法，其中該合同之定義為：  
i) 至少一條件；以及  
ii) 至少一行為，其效能取決於該條件的評估。

【第19項】如請求項 14 至 18 所述之方法，其中該元數據包括：  
i) 一位址或一位址表示，為該合同儲存在該電腦中之該儲存庫之位置；以及/或  
ii) 該合同之一雜湊。

【第20項】如請求項 14 至 19 所述之方法，更包括下列步驟：  
藉由判斷該UTXO是否在該區塊鏈的一UTXO清單中，以確認該合同是否已終止。

【第21項】如請求項 14 至 20 所述之方法，其中該合同係儲存於一分散式雜湊表(Distributed Hash Table, DHT)中。

【第22項】如請求項 14 至 21 所述之方法，更包括下列步驟：  
向該區塊鏈廣播一交易，包括一指令，用以在一指定日期及/或時間花費該輸出(UTXO)，該指令為一CheckLockTimeVerify指令。

【第23項】如請求項 14 至 22 所述之方法，其中獲得該合同之部分或全部內容僅限於至少一指定的授權方。

【第24項】如請求項 14 至 23 所述之方法，其中該合同包括一確定性有限自動器(Deterministic Finite Automaton, DFA)，以實現該合同。

【第25項】如請求項 24 所述之方法，其中該確定性有限自動器係使用一法律編纂方案(codification scheme)定義。

【第26項】如請求項 24 或 25 所述之方法，其中該確定性有限自動器係使用：

- i) 至少一區塊鏈交易，其利用一腳本語言；
- ii) 一計算代理商，其用以監視該區塊鏈之狀態；及/或
- iii) 一數位錢包的一組指令。

【第27項】一種系統，用以執行上述任一請求項所述之方法。



















































