



國立臺灣大學電機資訊學院資訊工程學系

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

基於類比特幣區塊鏈上的虛擬資產發行與管理的治理

架構之設計實做

Governance structure for virtual asset issuance and
management on bitcoin-like Blockchain

張鈞為

Chun-Wei Chang

指導教授：廖世偉 博士

Advisor: Shih-Wei Liao, Ph.D.

中華民國 105 年 7 月

July 2016

中文摘要



在現在社會中，隨著科技的大幅進步，越來越多的事物開始跟數位生活結合。其中包括了資產的數位化-數位資產(數位貨幣)。從過去最簡單的數位點數概念，透過儲值的方式在遊戲或線上商城消費，到近幾年瘋狂流行的比特幣(Bitcoin)，以及現在世界各地不斷研究的數位金融科技，都可以視為是資產數位化的實例。

其中又以比特幣的發展最為快速。比特幣底層的核心技術我們稱為區塊鏈(Blockchain)。基於區塊鏈的匿名、安全及不可篡改等等特性，吸引了很多使用者在比特幣網路上進行交易。區塊鏈技術也吸引了成千上萬的開發者前來開發，不論是修改區塊鏈本身的設計，或是在現有的區塊鏈上做出新的應用，都讓區塊鏈成為這兩年來最熱門的資訊名詞之一。

但原本的比特幣區塊鏈只支援單一貨幣(即 Bitcoin)，因為這種特性，造成了比特幣在對應到一些現實生活中的行為上會出現一些困難，也大大限制了區塊鏈的可使用性。於是有人開始思考如何在區塊鏈上抽象地做出多種可流動的Token。常見的像 bitcoin 2.0 等 meta-layer，如 OmniLayer。又或是自己修改後另開不同的區塊鏈，如 GCOIN blockchain，提供了一般區塊鏈不存在的「治理架構」的概念。

本研究致力於設計一個將 GCOIN 特有的治理架構，實做在比特幣區塊鏈上的 meta-layer(即 Omni)，目標做出一個既讓人放心使用，又能讓區塊鏈服務合乎更多現實生活所需要的功能。

Abstraction



In this decade, the digitization of our daily life became more important these days. Especially digital currency, it became one of the most popular digitization topic in this trend.

Bitcoin, the most popular digital currency nowadays, also became one of the hottest research topic. In Bitcoin network, the most important feature is “blockchain”, a public distributed ledger. Due to the safety that Bitcoin blockchain guarantee, Bitcoin attracts many users to do transaction on it.

Also, blockchain technique attracts many developers to research and develop on it. However, because of some nature limitation on Bitcoin blockchain, sometimes Bitcoin cannot match user’s need. To break these limitations, many different new designs show up these years.

Some developers develop applications on original Bitcoin blockchain. For example, many bitcoin 2.0 meta-layer applications, like OmniLayer, provide their service in this way.

Other developers try to design their own blockchain to replace original Bitcoin blockchain. For example, GCOIN use their new blockchain design to provide multiple properties and governance structure features.

Both two ways have their own pros and cons. In this paper, we implemented GCOIN governance structure feature on OmniLayer, and try to provide a trustable and safe solution for those who want to use GCOIN features on Bitcoin blockchain.

目錄



中文摘要.....	I
ABSTRACTION	II
目錄.....	III
圖目錄	IV
表目錄	IV
第 1 章 緒論.....	1
1.1 研究背景.....	1
1.2 研究動機.....	2
第 2 章 現今區塊鏈技術概況.....	3
2.1 BITCOIN 及 BLOCKCHAIN 概述.....	3
2.2 GCOIN 概述	4
2.3 OMNILAYER 及 OMNICORE 概述	5
第 3 章 治理架構設計	6
3.1 OMNILAYER DATA PACKET 簡介	6
3.2 GOVERNANCE 設計	10
3.2.1 投票系統設計	10
3.2.2 License/Alliance 設計.....	13
第 4 章 治理架構相關實做	15
4.1 DATABASE ENTRY 實做.....	15
4.2 RPC 實做	15
第 5 章 實驗及測試流程	17
5.1 LICENSE 測試流程	17
5.2 ALLIANCE 測試流程	17
第 6 章 結論及未來展望	18
參考資料.....	20



圖目錄

圖 2-1 OmniLayer (前 MasterCoin)的階層關係示意圖	6
圖 3-1 Transaction 及其 input/output 例圖[4]	7
圖 3-2 Overview of Transaction Spending	7
圖 3-3 OmniLayer data packet 示意圖	9
圖 3-4 Class B 用來裝資訊的假 Address 示意圖	9
圖 3-5 License 申請示意圖	11
圖 3-6 申請 Alliance 流程示意圖	12

表目錄

表 3-1 Alliance 資訊主要欄位	14
表 3-2 License 資訊主要欄位	15
表 5-1 OmniLayer vs GCOIN Layer	18

第1章 緒論



1.1 研究背景

如摘要所言，現今各種數位資產慢慢地流行起來，其中又以比特幣，又或是說區塊鏈技術最受注目。究竟什麼樣的區塊鏈服務最能處理人們生活上的不便，是近幾年做這類研究的開發人員最為關心的事。

比特幣的流通在這五年左右廣泛地得到大家的肯定，但畢竟它一開始並不是被設計為一個給多資產流通的網路，如果要使用於更廣的應用情境，難免會有所不足。為了解決比特幣本身使用上的彈性不足，許多開發人員試著開發新的區塊鏈服務，常見的有以下兩種方法。

第一，直接修改區塊鏈的資料結構。讓記錄交易的資料結構直接多上一個欄位來記錄這次的交易是屬於哪一種 **Token**。優點是可塑性很高，開發者 A 想要加上什麼新的功能就只需要自己改並自己營運屬於自己的私有區塊鏈 (**Private blockchain, or permission blockchain**) 即可。但缺點就是，如果使用者不相信、不放心 A 的私有區塊鏈，他們會選擇相信更穩定如比特幣般公開區塊鏈 (**Public blockchain, or permission-less blockchain**)，這樣 A 的區塊鏈仍然無人使用。這個不信任可能來自於不信任區塊鏈的營運者或交易驗證者，或是單純覺得使用者不夠多而拒絕使用。

GCOIN 團隊在過去兩年採用了此方法，開發了 **GCOIN 區塊鏈ⁱ**。透過修改區塊鏈的資料結構，GCOIN 不只支援 **Multi-colored token**，他還支援了許多現實生活中各組織 (**ex.公司、聯盟、金融機構等**) 所需要的管理資產的發行、鑄造等治理架構(在後面會詳述其治理架構設計)。除此之外，GCOIN 也修改了本來 **Bitcoin**

較為緩慢的驗證交易方法，使得 GCOIN 可以更貼切現實生活中的某些應用場景。雖然 GCOIN 創造了不同以往的新設計，但是在現實還是聽到了一些不信任此區塊鏈的聲音。

第二，使用已存在的公開區塊鏈，但是在區塊鏈加上屬於自己的資料，這種作法我們常將他稱為 bitcoin 2.0 meta-layer。我們可以把區塊鏈視為一個大家都看得到的公開帳本。以往傳統的做法，大家只在上面記錄有關比特幣的交易(input/output 的形式)，但是其實是有辦法在每筆 transaction 加上額外的資訊(稍後會再解說外加資訊的方法)，在這些資訊中記錄比特幣之外的抽象交易，如此一來就能在已存在的公開區塊鏈上流動多種數位資產了。優缺點剛好跟第一種方式相反，選擇了大家信任的公開區塊鏈，但也會被原有的區塊鏈的天然限制給侷限一些彈性。

市面上有許多開發在比特幣區塊鏈的產品都算第二種。他們運用自家的定義的資訊內容，就可以在比特幣區塊鏈來做比特幣之外的服務。像是 Omni (前 Mastercoin)，他們提供了多種自定義的交易類型，提供了多種資產流動，每個人都可以發行屬於自己的資產。除此之外他們還實做出像是在比特幣區塊鏈上進行群眾募資，或是不同幣別之間的掛賣、拍賣等功能。

這兩種做法各有其優缺點，如何在之間找到各自的應用場景，又或是說未來如何找到區塊鏈的 Killer App，將會區塊鏈是未來的發展重點。

1.2 研究動機

經過我們的調查，現在市面上的 meta-layer 類型的服務，雖然有各自的特色，但是確沒有 GCOIN 最重要的特色之一：治理架構。以 Omni 為例，他們雖能讓每個使用者(每個 address)自行發行各種自己想要的數位資產，但在 Omni layer 上沒有 License distributor 的概念。換句話說，「任何人」都可以在 Omni 上

發行「任意種類」、「任意數量」的數位資產而不受到任何監控。

在一般的現實生活中，想在某領域發行自己的資產(貨幣、股票、點數等)，往往需要有一個更高層的管理者群組，對各資產作出判斷、核准的動作，以確保此數位資產的安全性、合法性足夠，才允許發行者發行該資產。也就是說，現實生活中常需要有以下流程：欲發行者向管理申請發行某資產→管理人員審核、批准申請→發行者才得以發行資產。GCOIN 做到了以上的治理架構，發行者需拿到由 Alliance 核許的發行權(License)才能發行資產。

因此我們萌發了一個想法：把 GCOIN 的特長，也就是治理架構，加以調整並重新設計成適合跑在比特幣區塊鏈上 meta-layer 的版本，再將此治理架構的概念實做在一個已存在、穩定及開源的比特幣區塊鏈 meta-layer 服務(OmniLayer)之上。如此一來，就能設計並實做一個能達到既保有 GCOIN 在治理架構的特點，又能讓不放心私有鏈的使用者安心的新區塊鏈應用。

因此，本篇論文將會致力於設計一個基於 OmniLayer 現有「在比特幣區塊鏈上加上資訊」的技術，並將 GCOIN 治理架構的概念做調整，將其完整實作在比特幣的公開區塊鏈之上。為了和本來的 OmniLayer 做出區隔，我們將此實做稱作 GCOIN Layer，本文以下也都用 GCOIN Layer 來代表。

第2章 現今區塊鏈技術概況

2.1 Bitcoin 及 Blockchain 概述

比特幣為一個 2009 年由中本聰(或譯作中本哲史，英文讀音為 Nakamura Satoshi)推出以區塊鏈做為帳本的全球通行數位貨幣。它並沒有一個中央控管的

Server 及 Database，而是採用 p2p 及分散化的方式運行，所有的工作基本上都是由整個比特幣網路的參與者共同維護。

在比特幣的架構中，區塊(block)可以視為收納多筆交易(Transaction)的資料結構。何謂區塊鏈？顧名思義即為區塊們串接起來而形成的一條鏈(chain of blocks)，所有的已被驗證過的交易記錄都被記在區塊鏈內。

每一個區塊，都是由比特幣礦工(miner)不斷試著將一至多個交易收至其中，並設法通過一連串密碼學方法的驗證。一旦有任一礦工成功建出新的區塊(即通過驗證)之後，會把新形成的區塊接上區塊鏈並廣播到整個網路，通知所有人此區塊已被驗證並繼續建立下一個區塊。中本聰稱此「收集與驗證交易以生成新的區塊」這個動作叫挖礦(mining)，進行挖礦的機器稱為礦工ⁱⁱ，因為礦工一旦成功生出新的區塊會得到比特幣做為獎賞，就有如挖礦一般。一旦新的區塊被加到區塊鏈，所有在此區塊中的交易即被視為已被驗證合法的。新的交易/區塊被只要被驗證合法過後，所有在鏈上的資料就無法被修改、刪除。

因此，在比特幣網路上，每個使用者(address)的餘額或是所有交易等資訊，只要從區塊鏈上一掃過一次後即可得知。透過這種特性，區塊鏈也可以永久記錄一些小量非交易的資訊在上面。我們將透過這個特性將我們要的資料存在區塊鏈上。詳細的做法會在之後的章節提到。

2.2 GCOIN 概述

GCOIN 區塊鏈協議為 GCOIN 開發團隊致力於提供一套開放的數位金融的基礎建設，特別針對金融應用及電子商務所需的交易速度、可擴展性等特性加以改進、優化。

不同於傳統比特幣區塊鏈，為了提高交易結算速度(比特幣約每 10 分鐘結算一次)，改良原本由挖礦建立的分散式共識演算法(Decentralized Consensus

Algorithm)，以達到特定業者所需。不過這一部份不是本篇文章的重點，將不會太詳細著墨在此。

除此之外，GCOIN 在本篇最重要的特點為「多中心階層架構」，也就是前面多次提到的治理架構。GCOIN 區塊鏈在他們的網路中設計不同角色的參與者，大致上分為聯盟成員(Alliance member，以下都簡稱為大寫開頭的 Alliance)和發行者(Issuer)。在 GCOIN 的區塊鏈中，Alliance 來負責驗證交易的正確性，以及核發每種資產的發行許可(以下簡稱大寫開頭的 License)。而發行者故名思義即為發行特定資產的角色，他們得到由 Alliance 的發行許可之後，即可享有該資產的發行、鑄造權。反之，沒有資產 License 的人是沒有資格發行、鑄造該資產的。

2.3 OmniLayer 及 Omnicore 概述

OmniLayer 為實做在使用者和比特幣之間的抽象層(meta-layer)，而 Omnicore 指的是使用者用來在 OmniLayer 執行的程式。OmniLayer 提供多樣原本比特幣不存在的各種服務。提供的服務十分多樣化，如發行自己的資產、在區塊鏈上發行募資、不同數位資產之間的各種交易等等。即使對一個沒有資訊背景的使用者，也可以透過幾個簡單的步驟在 OmniLayer 上發行、管理屬於自己的資產。

OmniLayer 透過比特幣區塊鏈的 transaction 記錄屬於自己的資訊，一旦有新的 OmniLayer 資訊被記錄到區塊鏈時，各個 OmniLayer 上的節點會去了解該資訊所含有的意思。舉例來說，Allen 發了一個記錄「送一個 Allen 幣給小明」資訊的比特幣交易。一旦這筆交易被認證收到區塊鏈中之後，各個 OmniLayer 上的節點就會收到此筆交易的資訊，他們就會知道「Allen 送了 1 Allen 幣給小明」，因此他們在各自電腦所存的狀態就會更新，把 Allen 的餘額扣一並把小明的餘額加一。

因此如上方的例子，只要 OmniLayer 上的大家看到的區塊鏈資訊一樣，只需各自一一從頭掃下來看過一次，最後大家就能得到相同的結果。

OmniLayer 的各節點在比特幣網路中其實和一般的比特幣節點沒什麼不同，看到一樣的區塊鏈，看到一樣的交易，交易的格式、方式也都一樣。唯一不同的就是，一般的比特幣節點看不懂、也不會去處理屬於 OmniLayer 上的資訊。

有關 Omni 如何在比特幣區塊鏈上包裝資訊的技術，在下一章節會有詳細的解說。

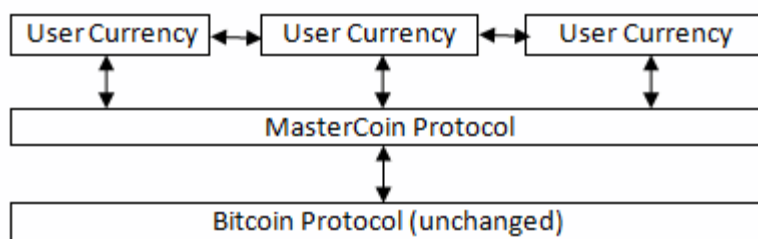


圖 2-1 OmniLayer (前 MasterCoin)的階層關係示意圖ⁱⁱⁱ

第3章 治理架構設計

3.1 OmniLayer data packet 簡介

在正式進入第三章前，會先介紹 OmniLayer 如何將資訊包入比特幣的交易之中，因這一段不屬於我們的設計，所以放在第 3 章第 0 節來作為第 3 章的前導。

在這之前，得先簡單介紹比特幣區塊鏈的交易行為。每一筆比特幣的交易都有一至多筆的 Input 及 Output，該筆交易即是記錄拿了哪些錢做為 Input 來使用，並看這些錢要給哪些人，再一一包成 Output 的樣子。這些 Output 未來將會變成其他 transaction 的 Input，也就是說所有的 Input 其實就是以前的 transaction 的某筆 Output。

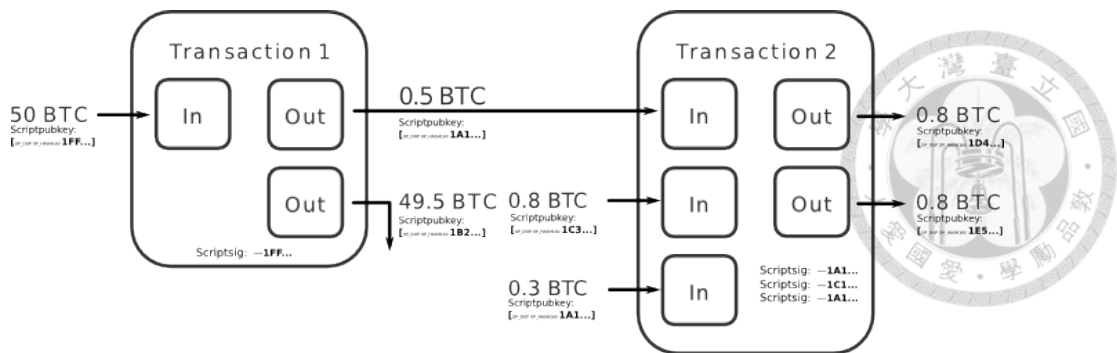


圖 3-1 Transaction 及其 input/output 例圖^{iv}

正常來說每一筆 Output 都會記錄著一筆金額的比特幣及一段 script language，而這一段 script language 用來負責「鎖住」這筆錢，我們稱之為 scriptPubKey，或是 locking script。scriptPubKey 通常會指定一至多個 public key 作為自己的解鎖鑰匙，只有擁有和這(些)public key 相對應 private key 的人才有辦法生出對應的簽章(signature)。透過簽章可以生出相對應的 script 來解鎖，我們稱這段解鎖的 script 為 scriptSig，或是 unlocking script。有這段 scriptSig 的人才能把這筆 Output 拿來使用，未來才可以做為另一筆交易的 Input。下圖 3.有如何花費 Output 的圖解。比特幣區塊鏈交易的 Input 及 Output 概念基本上是這樣，至於各種 script 的細節我們不在此討論。

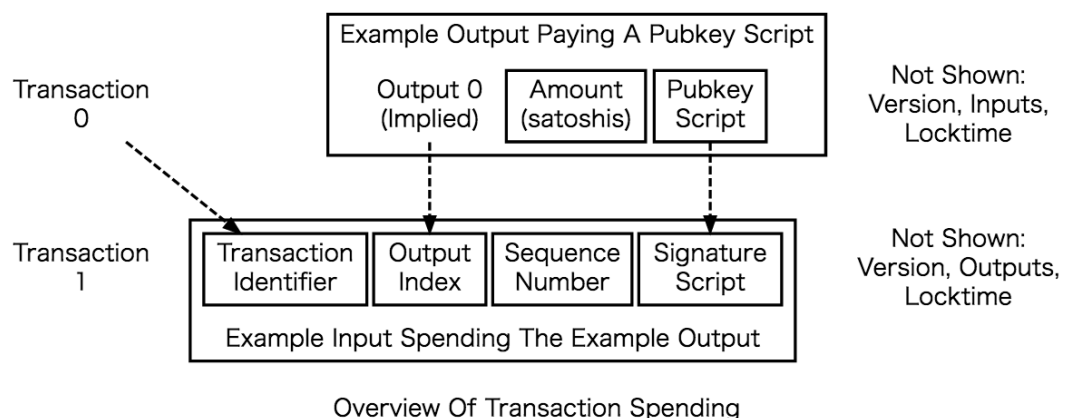


圖 3-2 Overview of Transaction Spending.

此圖可以看出 transaction 1 將 transaction 0 的 output 拿來做 input。^v

但是有一種特別的 Output 並不會傳送任何比特幣，他單純只用來記錄一些簡

單的 data，一般我們稱為 data output。此 output 的 script 很簡單，寫成：

OP_RETURN <data>，要記錄的資料就直接接在 OP_RETURN 之後即可。此種方法包的 data 上限為 80 byte。

接下來我們切入正題，Omni 如何在比特幣區塊鏈上存入自己的資料。Omni 官方定義了兩種方式：Class B 和 Class C。(Class A 目前沒有使用了就不在此介紹)

Class C

Class C 即是使用 data output 的方式寫入資料。Omni 在 OP_RETURN 後的 data 固定由'O', 'M', 'N', 'I' 這四個字元開始。如此一來，Omni 上的節點只需要去認由這四個字元開始的 data output 就可以了。但由於 data output 本身只能存 80 byte 的資料，扣掉這四個辨識字元後，只能存 76 個 byte。因此當 Omni 需要記錄超過 76 byte 的話，就無法使用 Class C 的方法。

Class B

當 data 大小超過 Class C 可以承載的上限後，就得使用其他方法來讓這些 data 上到區塊鏈。為了解決這個問題，Omni 他們使用 1-of-n multi-signature 的 transaction，通常 $n = 3$ 。要包成一個 1-of-3 multi-signature 的 transaction 需要三個 public key，其中除了第一個 public key 是發送者自己外，剩下 2 個 public key 就是 Omni 塞資料的地方。也就是說這兩個 public key 本身是 Omnicore 幫你虛構出來的。一個 public key 有 33 byte，Omnicore 利用中間 31 個 byte 來存 data，最後再用沒存 data 的頭尾兩個 byte 來調整這一串資料成為比特幣上合法的 public key。其中，用來存 data 的 31 byte，第一個 byte 拿來做為 data 順序的編號，所以每一個假的 public key 可以存 30 個 byte 的 data。Data 的順序編號範圍從 1~255(1 byte)，所以一筆交易最多可以存入共 $255 \times 30 = 7650$ byte。這個數字遠大於

Class C (data output)可以存入的資料大小(80 byte)，如此一來當所需的資訊有很多時也可以順利的上到比特幣的區塊鏈上。^{vi}

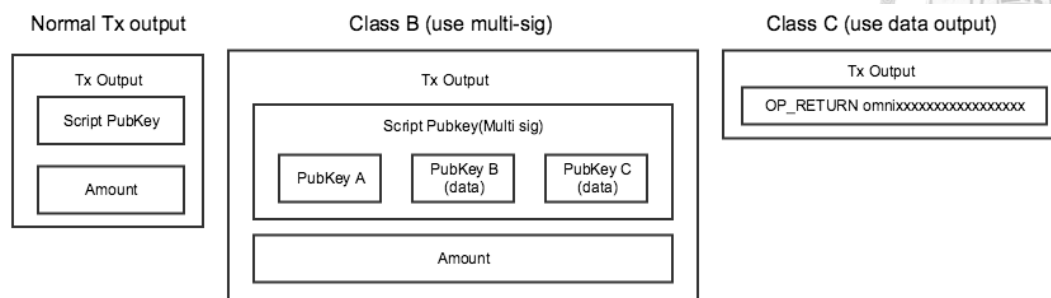


圖 3-3 OmniLayer data packet 示意圖

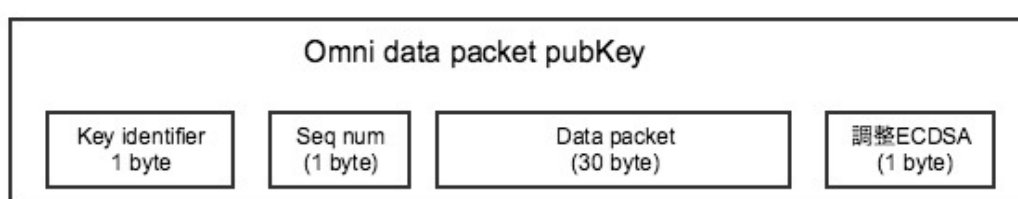


圖 3-4 Class B 用來裝資訊的假 Address 示意圖

Class C 是用'o', 'm', 'n', 'i' 這四個字完來判斷是屬於 OmniLayer 的 data output，同樣的在 Class B 中也有類似的手法。所有 OmniLayer 上的 Class b transaction 都會有一個 output 是給一個固定的 address，Omni 官方稱之為 Exodus address。只要發現有 transaction 的 output 是傳給 Exodus address 的話，OmniCore 就會試著去處理他，看是不是一個合法的 Class B transaction。

以上是 OmniLayer 在包裝他們 data packet 的方法簡介。所以如上面所說，所有的 Omni transactions 其實都是某種形式的比特幣交易，因此他們也都需要得到比特幣區塊鏈上的礦工來被記錄、驗證。所以，Omni transaction 仍需和一般交易一樣，每一筆得付一點點手續費(Transaction fee)給礦工們，所以某種程度上使用 OmniLayer 不是完全是免費的。

最後，為了避免不必要的問題，在 GCOIN Layer 中我們將 Class C 的開頭"omni"，改為"gcoin"，並將 exodus address 改為我們個人用的 address，如此一來 GCOIN Layer 和原本的 OmniLayer 就互不處理各自的 transaction，就不會有任

何衝突了。



3.2 Governance 設計

大部分「bitcoin 2.0」類的 meta-layer 技術(如 Omni)並沒有 License distributor 的概念，所有的 address 都可以任意發行自己的數位資產，也就是說沒有對 License 申請/審核/管理的概念。

我們希望在類比特幣區塊鏈般的分散式帳本上做出如 GCOIN 般，有某個「管理者」，或是「Alliance」的角色，可以做像核發某特定資產的 License 給某一 address 之類的行為，而這也是我們自稱為 GCOIN Layer 的主因。

我們的設計大致可以分為兩個部分，一為 Alliance 的投票系統，另一個為 License/Alliance 資料結構的設計。

3.2.1 投票系統設計

我們設計了一個適用於所有的「申請」流程的投票系統，包括申請 License、申請成為 Alliance 等需要 Alliance 認可後才可以擁有權限的行為，我們都透過我們設計的投票系統來解決。

在 GCOIN Layer 中，所有的申請和投票都是一個 transaction。每個 Alliance 都有權限可以針對特定申請送出 Vote transaction，其中記錄著投票目標以及贊成或反對。一旦投票目標超過了合格門檻，該筆申請即被接受。

以下的例子為一個 Address 想申請新的 License(申請成為 Issuer)的流程：

1. Address “A” 發出申請 License 的 Transaction (Apply Tx)
2. 交由比特幣區塊鏈上的礦工將此 Apply Tx 收入新的區塊中

3. 一旦被驗證記錄下來，所有在 GCOIN Layer 上的使用者收到新的區塊之後，在掃讀新的 transaction 時，會看到剛剛由 A 發出的 Apply Tx，就可以得知 A 所發出的申請
4. 每個 Alliance 都可決定對此申請要贊成還是反對，並透過發起 Vote Tx 記錄結果。Vote tx 會記錄投票目標及贊成/反對。
5. Vote Tx 和 Apply Tx 一樣交由區塊鏈上的礦工來驗證。一旦足夠多 Vote Tx (足以判斷合格/失敗的投票量) 也被收入新的區塊之後，此 Apply Tx 即算處理完畢。
6. 如果最後的贊成票有過過門檻(預設為 2/3 Alliance)的話，則該 Address “A”成為此資產的 Issuer。反之，則代表此申請無效。

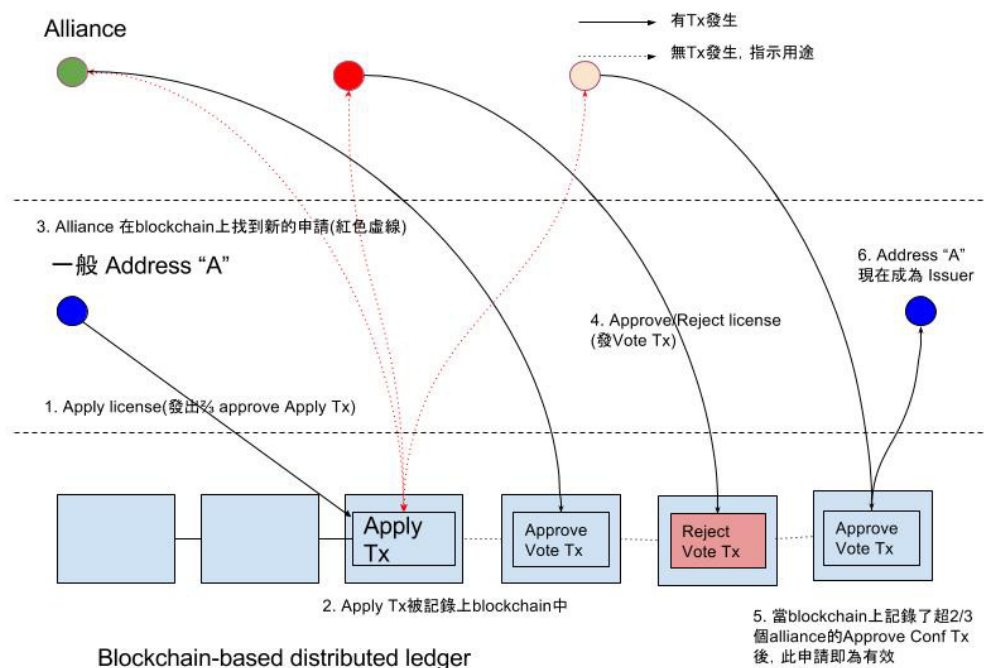


圖 3-5 License 申請示意圖

同理，如果一個普通的 Address 想要加入成為 Alliance，也可以透過類似的流程。

1. 一般 Address “A”發出一個申請成為聯盟成員的 Apply Tx
2. 交由比特幣區塊鏈上的礦工將此 Apply Tx 收入新的區塊中
3. 一旦被驗證記錄下來，所有在 GCOIN Layer 上的使用者收到新的區塊之後，在掃讀新的 transaction 時，會看到剛剛由 A 發出的 Apply Tx，就可以得知 A 所發出的申請
4. 每個 Alliance 可決定對此申請要贊成還是反對，並透過發起 Vote Tx 記錄結果。Vote Tx 會記錄投票目標(A 申請成為聯盟的一員)及贊成/反對。
5. Vote Tx 和 Apply Tx 一樣交由區塊鏈上的礦工來驗證。一旦足夠多 Vote Tx(足以判斷合格/失敗的投票量)也被收入新的區塊之後，此 Apply Tx 即算處理完畢。
6. 如果最後的贊成票有過過門檻(預設為 2/3 Alliance)的話，則該 Address “A”成為此新的 Alliance。反之，則代表此申請無效。

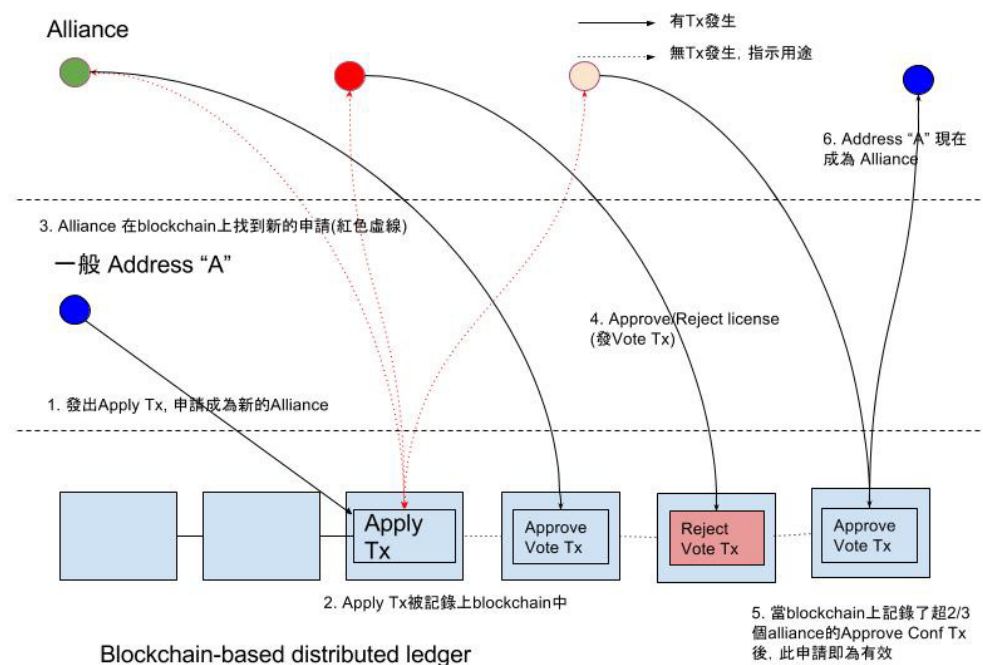


圖 3-6 申請 Alliance 流程示意圖

我們這個發想是來自於 GCOIN 的 Alliance/License 特性再加以改變的，和 GCOIN 不同的是，GCOIN 的 License 只需要單一 Alliance 核發即可，並沒有經過投票這個手續。在我們這篇論文中，不論是 Alliance 或 License 我們都是透過投票來完成審核。



3.2.2 License/Alliance 設計

以下會解釋我們如何實際設計 License 及 Alliance 的設計。有關所有 License 及 Alliance 資料的存取我們都用 Level db 來實做。

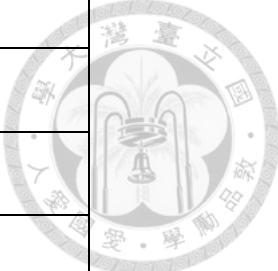
Alliance

Omni 中本來沒有 Alliance 的概念。在 GCOIN 裡，Alliance 代表著聯盟成員，他們負責區塊鏈的驗證、license 的核可、新 alliance 成員的加入等等需較高權限的行為。像是所有的投票相關功能，只有 Alliance 有權力可以發送。

在比特幣區塊鏈上，由於礦工是不需要經由許可的(permission-less)，因此我們設計主要參考只 GCOIN 中 Alliance 成員的治理架構部份，包括 License 和新的聯盟成員的核可。

我們自定義的 Alliance Info 中存有基本的成員資訊，包括名字、address、網址、描述等等。除此之外，記錄成員的身分狀態，記錄該 address 是否為一個合格的成員。附表為 Alliance info 的主要內容。

Type	name	description
std::string	address	alliance's address.
std::string	name	alliance's name.
std::string	url	website url



std::string	data	Some description.
uint256	txid	creation tx hash.
uint32	approve_threshold	
uint32	approve_count	
uint32	status	Approved/pending/rejected

表 3-1 Alliance 資訊主要欄位

在 GCOIN Layer 中，除了一開始設定好預設的 Alliance 外，新的成員想加入聯盟也可以透過聯盟成員的投票。一旦經過投票通過，就可以加入聯盟成為新的一員。

License

理想的 License 應作為一記錄，記錄數位資產的發行權利及義務。

在 Omni 中本來就有定義 Smart property info，但他只用來記錄該 property 的一些基本資訊，但並沒有類似「許可」的概念。因此在 GCOIN Layer 中我們調整了 Omni 本來就有的 smart property info，另外加上了記錄投票數及投票門檻的欄位，如此一來就可以當作我們的 license 了。

有了 License 的概念後，某 address 如要發行該資產時，我們會去檢查：

1. 該 License 的發行者是不是這個 address
2. 該 License 的贊成票是否有超過合格門檻

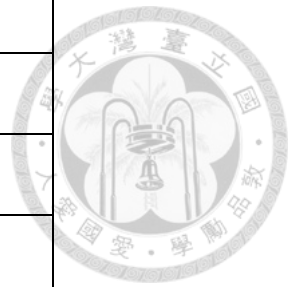
如果這兩個條件都通過，此 address 才有發行這個資產的權力。附表為

License info 的主要內容。

Type	Name	Description
int	Property type	Divisible/indivisible
std::string	Name	Property name

std::string	url	Website url
std::string	data	Some description.
std::string	issuer	Issuer's address.
uint16	approve_threshold	
uint16	approve_count	
uint32	status	approved/pending/rejected

表 3-2 License 資訊主要欄位



第4章 治理架構相關實做

4.1 Database Entry 實做

我們針對以上設計，在 Omnicore 上做出了幾個新的類別(c++ class)及新的 RPC (remote procedure call)。有關 Alliance 和 License 的類別實做，大致上就是照都在上面的設計的章節所說的，在此節我就不特別詳細說明。

4.2 RPC 實做

這一節會介紹幾個比較重要的新 rpc 實做。

gcoin_apply_license (申請新 License)

此 rpc 我們是直接從 Omnicore 本來現有的 tx type 54 (new property with managed number of tokens)來修改。本來 type 54 即為申請一個新資產的 rpc，不過並沒有加入審核的概念在內。我們在此 rpc 呼叫時，新加入了合格門檻的概念，新創立的資產如沒有足夠的贊成，是無法擅自發行的。

gcoin_vote_for_license (投票給某 License)

此 rpc 是給聯盟成員針對特定 License 投票使用，使用我們自定義的 tx type 500。使用時需指定特定 License 的 id，並外加贊成或反對，送出後即可成為一張合格的選票。Layer 成員收到投票後，會將指定的 license 的得票數做更新。

目前我們不支援重複投票，也就是一個成員對一個 license 最多只能投一次票且一旦投出後無法修改內容。

gcoin_grantlicense (發行、鑄造某 License 所代表的 token)

此 rpc 我們是直接從 Omnicore 本來現有的 tx type 55(Granting tokens for a managed property)來修改。在原本的 omni 中，這個 rpc 本來就是和 type 54 一起合作使用，不過我們多加了 License 狀態的檢查，只讓已合格的 License 的 Issuer 有權發行該資產。

gcoin_apply_alliance (申請成為 Alliance)

此 rpc 是給非聯盟成員申請成為新成員使用，使用我們自定的 tx type 400。使用時需填入 Alliance info 所該有的一些基本欄位，如名稱、網址、基本描述等資訊，以供其他 Alliance 審核。透過此 rpc 送出 transaction 後，此 layer 上所有人都可以看到申請者成為未被確認的 Alliance。

目前不支援重複申請，同一個 address 無法做兩次 Alliance 的申請。

gcoin_vote_for_alliance (投票給某 Alliance)

此 rpc 是給已合格的 Alliance 針對特定 Alliance 申請做為投票使用，使用我們自定義的 tx type 501。使用時需指定投票對象的 address，並外加贊成或反對，送出後即可成為一張合格的選票。Layer 成員收到投票後，會將指定的 pending Alliance 的得票數做更新。

目前我們不支援重複投票，也就是一個成員對一個 address 最多只能投一次票且一旦投出後無法修改內容。

gcoin_get_alliance_info_list (列出所有 Alliance 的詳情)

此 rpc 為單純列出所有 Alliance 詳情用，包含已合格、等待中的 Alliance。



第5章實驗及測試流程

5.1 License 測試流程

1. 創立一個全新的 Layer，包括設定 Class c 的 marker 和 Exodus Address (同時這個 Address 也會是初始的聯盟成員)。
2. 用不同於 Exodus 的 address 申請一個 License。門檻應是總 Alliance 的 $\frac{2}{3}$ ，因初始只有一個 Alliance，所以門檻應為 1。
3. 收到新的申請後，應看得到該筆 License 狀態為”等待中”，此時還無法發行該資產。然後再用 exodus address 投出贊成票。
4. 收到投票後，應看得到該筆 License 狀態改為成功。之後申請者就可以自由發行該資產。當然，非申請者仍不能發行該資產。

5.2 Alliance 測試流程

1. 用不同於 Exodus address 的 address 申請成為聯盟成員。門檻應是總 Alliance 的 $\frac{2}{3}$ ，因初始只有一個 Alliance，所以門檻應為 1。
2. 收到新的申請後，應看得到該筆 License 狀態為”等待中”。再用 exodus address 投出贊成票。
3. 收到投票後，應看得到該筆 License 狀態改為成功。之後申請者就可以自由發行該資產。
4. 往後所有申請的門檻將會因為新的 Alliance 出現而增高，而此 address 往後

也都享有投票權。



我們以上兩個流程來回交叉做了多次，以確保邏輯沒有問題。不該發行的人不能發行、不能投票的人不能投票等等，經過測試果然照我們所期望。以下是 GCOIN Layer 和本來的 OmniLayer 的功能比較表。

	OmniLayer	GCOIN Layer
支援多種資產流動	O	O
License 申請	O	O
License 的審核	X	O
Alliance 申請	X	O
Alliance 的審核	X	O

表 5-1 OmniLayer vs GCOIN Layer

第6章 結論及未來展望

我們成功設計出了一套「基於比特幣區塊鏈」的「多資產的治理、管理架構」，成功解決以下兩個看似相同但實際上是不同的面相問題。第一，我們成功實做出目前在比特幣區塊鏈上的各家 meta-layer 架構中所缺的，所謂透過不同層級來達到治理、管理多資產的功能。第二，我們把 GCOIN 的治理概念搬到比特幣區塊鏈上，讓本來對私有鏈/permission chain 有疑慮的人可以放心在公有鏈/permission-less chain 上享有如 GCOIN 一般的治理功能。

基本上我們在 GCOIN Layer 的 Alliance/License 及投票系統的設計，在上一段

所描述的及一些額外的測試中都是沒有問題的，GCOIN Layer 運行得和我們預期設計的一樣。不過，針對一些惡意攻擊的部份我們沒有研究得太詳細。關於這點，如果以後要正式對外開放的話，可能還要在那之前還需要再經過更多、嚴謹的安全性測試才能正式上線。

另外最後有幾點要注意的是：

第一，使用比特幣區塊鏈挖礦需要時間，交易不是送出後馬上就能被驗證，

因此可能會有延遲。運氣不好的話可能要等上一陣子。

第二，我們目前所有的測試都是在比特幣的測試鏈(Bitcoin Testnet)上。測試鏈

和 Bitcoin Mainnet 行為上應為一樣，所以正確性應不需質疑。選擇使

用測試鏈的主因單純是因為在測試鏈上測的話，tx fee 不使用真的比

特幣。

第三，我們沒有很嚴謹的處理當 blockchain 發生 fork 所可能造成的問題。舉

例來說，如果當 fork 發生了，本來以為已存在的申請而投出的票，交

易可能會因為在另一條 blockchain 上順序亂掉，造成本來以為已投的票

失效。甚至如果產生了 double spending 的問題而本來存在的交易最後

失效，會是更麻煩的問題。這點還需要有更詳細的研究。

最後，除了比特幣外，是不是其他的類區塊鏈的公開帳本系統也可以使用這套設計？說不定以後這個 layer 可以隨著用戶所需，隨時可以抽換底層的區塊鏈，而不僅限於比特幣區塊鏈，如此一來可以創造出更多彈性。

參考資料



-
- ⁱ GCOIN Whitepaper, GCOIN Foundation, <https://github.com/OpenNetworking/gcoin-community/blob/develop/README.md>
 - ⁱⁱ Mining, bitcoin wiki, <https://en.bitcoin.it/wiki/Mining>
 - ⁱⁱⁱ Visualizaton, OmniLayer Spec, <https://github.com/OmniLayer/spec#visualization>
 - ^{iv} CoinJoin, bitcoin Wiki, <https://en.bitcoin.it/wiki/CoinJoin>
 - ^v Bitcoin develop Guide, Bitcoin.org, <https://bitcoin.org/en/developer-guide#transactions>
 - ^{vi} Class B transaction, OmniLayer Spec, <https://github.com/OmniLayer/spec#class-b-transactions-also-known-as-the-multisig-method>