



北京大学

硕士研究生学位论文

题目： 区块链的实名交易实时监督系统的
设计与实现

姓 名： _____

学 号： _____ 1601210903

院 系： _____

专 业： _____

研究方向： _____

导 师： _____

二〇一八年一月

版权声明

任何收存和保管本论文各种版本的单位和个人，未经本论文作者同意，不得将本论文转借他人，亦不得随意复制、抄录、拍照或以任何方式传播。否则一旦引起有碍作者著作权之问题，将可能承担法律责任。

摘要

比特幣是一個集成網絡學、密碼學、貨幣銀行學的加密貨幣，加密貨幣市場中，有數以千計的貨幣種類在市場流動著。值得一提的是，至今比特幣的點對點式電子現金系統還未出現過錯誤。比特幣最大的特色在於去中心化與匿名化，以去中心化的基礎建構出一個其他人無法管控的點對點金流，但也因為其匿名之特性，存在著三項問題，分別為無法追蹤、無法得到稅收以及在交易的過程中無法保障消費者權益。第一，在比特幣系統中，沒有任何一個使用者可以要求每一個人落實實名制，政府主管機關與相關人士難以追查每一筆資金的真正持有者，進而增加洗錢防制的困難性。第二，稅收更是國家經濟基礎運作的資金來源，現今的國家並無支持以比特幣交易相關的收銀系統或是制定出相關的稅務標準，使得政府無法從加密貨幣這方面的金融交易獲得稅收。第三，現有的比特幣交易模型皆為匿名與匿名之間的交易，並無開立收據，無法保障消費者權益。

本論文設計與實現一個区块链匿名加密貨幣為主的實名交易監督系統，論文工作包括五項。第一，交易模型分析：分析五種交易模型，發現匿名支付給實名交易，達到保障消費者權益、保護消費者隱私以及使政府可以課徵稅收。第二，系統設計：以匿名支付給實名交易為基礎設計支持加密貨幣的系統架構。第三，系統實現：使用 Java 編程語言，實現主要系統、商店和商品信息管理子系統、移動裝置收款及交易子系統以及客戶端移動支付和交易子系統。第四，系統優化：解決區塊鏈速度緩慢，引入多重簽章算法構建實時監督系統。第五，功能與性能測試：實現系統後，對本系統進行功能測試，且對原始監督系統與實時監督系統進行性能測試與分析。

本論文貢獻如下：

1. 引入匿名支付給實名的交易模型至加密貨幣系統。
2. 設計與實現於加密貨幣中匿名支付給實名的交易監督系統。
3. 導入多重簽章算法，設計與實現實時的交易監督系統，提升交易速度。
4. 實現於加密貨幣中，消費者匿名同時也讓消費者擁有消費者權益。
5. 藉由將商家實名，使政府主管機關可以獲得稅收的有效加密貨幣交易監督模型。

关键词：比特币，区块链，多重签章

Design and Realization of blockchain real-name transaction monitoring system

Chen Po Wei (Data Mining and Business Intelligence)

Directed by Prof. Lijie

ABSTRACT

Bitcoin is a cryptocurrency that integrates cybernetics, cryptography, and currency banking. In the cryptocurrency market, there are thousands of currency types that flow in the market. It is worth mentioning that Bitcoin's peer-to-peer e-cash system has not made mistakes so far. The biggest feature of Bitcoin is decentralization and anonymization, and a decentralized foundation creates a peer-to-peer flow that others cannot control. However, because of its anonymity, there are three problems: the inability to track the flow of funds, the inability to receive taxes, and the inability to protect consumer rights in the course of transactions. First, in the Bitcoin system, no one user can require everyone to implement the real name system, and it is difficult for the government authorities and related parties to trace the true holder of each fund, thereby increasing the difficulty of money-laundering prevention. Second, taxation is a source of funding for the operation of the country's economic base. Today's countries do not support the use of bitcoin transactions related to the cash register system or formulate relevant tax standards, making it impossible for the government to obtain tax revenues from cryptocurrency financial transactions. Third, the existing bitcoin transaction models are all transactions between anonymous and anonymous transactions, and no receipts have been issued to protect consumer rights.

This thesis designs and implements a real-name transaction monitoring system based on blockchain anonymous cryptocurrency. The work of the thesis includes five items. First, transaction model analysis: analyzing five transaction models and discovering anonymous payments to real-name transactions to protect consumer rights, protect consumer privacy, and enable the government to collect taxes. Second, the system design: Designed to support the cryptocurrency system architecture based on anonymous payments to real-name transactions. Third, the system is implemented: Using the Java programming language, the main system, store and merchandise information management subsystem, mobile device collection and

payment subsystem, and client mobile payment and transaction subsystem are implemented. Fourth, system optimization: to solve the slow blockchain speed, the introduction of multiple signature algorithms to build a real-time monitoring system. Fifth, functional and performance tests: After the system is implemented, functional tests are performed on the system, and performance testing and analysis are performed on the original supervisory system and the real-time supervisory system.

The contributions of this paper are as follows:

1. Introduce anonymous payment to real-name trading models to the cryptocurrency system.
2. Design and implementation of anonymous payments to cryptocurrencies to real-name transaction monitoring systems.
3. Import multiple signature algorithms, design and implement real-time transaction monitoring systems, and increase transaction speed.
4. In cryptocurrency, consumer anonymity also allows consumers to have consumer rights.
5. By putting the real name of the merchant, the government authorities can obtain an effective cryptocurrency transaction supervision model for taxation.

KEYWORDS: Bitcoin, Blockchain, Multiple signatures

目录

| | |
|---------------------------------------|-----------|
| 第一章 绪论 | 1 |
| 1.1 選題背景 | 1 |
| 1.2 研究目標與內容 | 8 |
| 1.3 論文組織結構 | 10 |
| 第二章 相關理論與技術 | 13 |
| 2.1 比特幣 (Bitcoin) | 13 |
| 2.2 比特幣地址 (Bitcoin Address) | 13 |
| 2.3 區塊鏈 (Blockchain) | 25 |
| 2.4 點對點網絡與加密貨幣安全 | 26 |
| 第三章 系統需求分析 | 29 |
| 3.1 功能性需求分析 | 29 |
| 3.2 交易模型分析 | 29 |
| 3.3 特性要因分析 | 33 |
| 第四章 系統概要設計 | 35 |
| 第五章 系統詳細設計與實現 | 43 |
| 5.1 區塊鏈的實名交易監督系統設計 | 43 |
| 5.2 系統模塊設計 | 45 |
| 5.3 系統實現 | 56 |
| 第六章 系統測試 | 59 |
| 6.1 功能測試 | 59 |
| 6.2 性能測試 | 65 |
| 第七章 总结与展望 | 73 |
| 参考文献 | 75 |
| 致谢 | 79 |
| 北京大学学位论文原创性声明和使用授权说明 | 81 |

图目录

| | |
|--|----|
| 图 1.1 2017 加密貨幣總市值走勢圖 ^[10] | 2 |
| 图 1.2 加密貨幣歷年最高總市值折線圖 ^[10] | 3 |
| 图 1.3 Bitcoin、Western Union ^[16] 、PayPal ^[17] 以及 VISA 每秒支持交易量比較圖 ^[18] | 5 |
| 图 1.4 比特幣區塊鏈成長走勢圖 ^[19] | 6 |
| 图 1.5 Darklaunder 洗錢模型 ^[25] | 7 |
| 图 2.1 LBC 突舉比特幣私鑰算力狀態圖 ^[33] | 14 |
| 图 2.2 區塊疊加示意圖 | 18 |
| 图 2.3 比特幣地址生成流程圖 | 21 |
| 图 2.4 Green Address 錢包生成流程圖 | 23 |
| 图 2.5 Green Address 交易發起流程圖 | 24 |
| 图 2.6 比特幣區塊鏈結構圖 | 25 |
| 图 2.7 Merkle Tree 示意圖 | 26 |
| 图 2.8 比特幣全節點分佈圖 ^[53] | 27 |
| 图 3.1 系統用例圖 | 30 |
| 图 3.2 各種交易模型示意圖 | 31 |
| 图 3.3 匿名對匿名交易示意圖 | 31 |
| 图 3.4 匿名對實名交易示意圖 | 31 |
| 图 3.5 實名對實名交易示意圖 | 32 |
| 图 3.6 實名對實名再對實名交易示意圖 | 32 |
| 图 3.7 實名對匿名再對實名交易示意圖 | 33 |
| 图 3.8 魚骨圖（1） | 34 |
| 图 4.1 BRTMS 系統功能模塊圖 | 35 |
| 图 4.2 商家端建置與管理商品資訊子系統功能模塊圖 | 36 |
| 图 4.3 商家端行動收銀與交易明細系統功能模塊圖 | 37 |
| 图 4.4 顧客端行動支付與交易明細系統功能模塊圖 | 37 |
| 图 4.5 BRTMS 和商家註冊流程的核心架構圖 | 38 |
| 图 4.6 BRTMS 的整體架構與功能示意圖 | 39 |

| | |
|---|----|
| 图 4.7 多重簽章優化後的 BRTMS 整體示意圖 | 41 |
| 图 5.1 BRTMS 數據庫實體關係圖 | 43 |
| 图 5.2 用戶註冊與登入模塊類圖 | 46 |
| 图 5.3 職工與商家註冊時序圖 | 46 |
| 图 5.4 商家產品管理類圖 | 47 |
| 图 5.5 商家產品管理時序圖 | 48 |
| 图 5.6 職工管理模塊類圖 | 49 |
| 图 5.7 商家職工管理時序圖 | 50 |
| 图 5.8 顧客交易管理模塊類圖 | 51 |
| 图 5.9 顧客交易管理時序圖 | 52 |
| 图 5.10 顧客交易管理 GA 模塊類圖 | 53 |
| 图 5.11 商家交易管理模塊類圖 | 54 |
| 图 5.12 商家交易管理時序圖 | 55 |
| 图 5.13 商家交易管理模塊類圖 | 56 |
| 图 5.14 SMIMSS 的 Java 應用程序的註冊和登錄界面 | 57 |
| 图 5.15 在 SMIMSS 中插入或更新授權商家的產品目錄 | 57 |
| 图 5.16 登錄、等待結帳的商品、刪除商品及支付確認 | 57 |
| 图 5.17 在 CMPTSS App 中，交易確認，付款確認、交易歷史記錄和發票 | 58 |
| 图 6.1 集成子系統測試 | 61 |
| 图 6.2 BRTMS 使用者用例圖 | 61 |
| 图 6.3 验收测试（Acceptance Testing） | 62 |
| 图 6.4 使用區塊鏈瀏覽器驗證存儲在比特幣區塊鏈中的交易過程 | 68 |

表目录

| | |
|---|----|
| 表 1.1 各國政府對比特幣態度統整表 | 9 |
| 表 2.1 比特幣簡介 | 13 |
| 表 2.2 ECC 與 RSA 的密鑰對生成時間比較表 ^[47] | 16 |
| 表 2.3 算法 BLISS、RSA、ECDSA 安全級數比較圖表 ^[48] | 16 |
| 表 2.4 MD5、SHA-0、SHA-1、SHA-2 和 SHA-3 比較表 | 17 |
| 表 2.5 Base58 編碼表 | 20 |
| 表 2.6 Base64 編碼表 | 20 |
| 表 3.1 交易關係比較表 | 33 |
| 表 5.1 商家信息表 | 44 |
| 表 5.2 產品信息表 | 44 |
| 表 5.3 交易信息表 | 44 |
| 表 5.4 用戶信息表 | 45 |
| 表 5.5 職工信息表 | 45 |
| 表 5.6 商家產品信息表 | 45 |
| 表 6.1 小米 3 手機規格 | 60 |
| 表 6.2 Google Nexus 5X 手機規格 | 60 |
| 表 6.3 IT1 測試用例 | 62 |
| 表 6.4 IT2 測試用例 | 63 |
| 表 6.5 IT3 測試用例 | 63 |
| 表 6.6 AT1 測試用例 | 64 |
| 表 6.7 AT2 測試用例 | 64 |
| 表 6.8 AT3 測試用例 | 65 |
| 表 6.9 集成子系統測試結果 | 66 |
| 表 6.10 驗收測試結果 | 66 |
| 表 6.11 子系統與測試用例關係表 | 67 |
| 表 6.12 需求與集成測試用例的關係表 | 67 |
| 表 6.13 需求與驗收測試案例的關係表 | 67 |

| | |
|--|----|
| 表 6.14 第一次 Testnet 執行實驗之數據分析 (2017 年 9 月 6 日) | 69 |
| 表 6.15 第二次 Testnet 執行實驗數據 (2018 年 3 月 21 日) | 69 |
| 表 6.16 第二次以 Testnet 執行實驗數據分析 (2018 年 3 月 21 日) | 69 |
| 表 6.17 初步的 Green Address 實驗測試數據 (2017 年 7 月 25 日) | 70 |
| 表 6.18 第一次以 GreenAddress 執行實驗之數據分析 (2017 年 9 月 6 日) . . . | 71 |
| 表 6.19 第二次以 GreenAddress 執行實驗之數據 (2018 年 3 月 21 日) | 71 |
| 表 6.20 第二次以 GreenAddress 執行實驗之數據分析 (2018 年 3 月 21 日) . . | 71 |
| 表 6.21 原始系統與實時系統比較表 | 72 |

第一章 緒論

現金法定貨幣，收據及交易數據庫存在著一些缺點。如現金很難杜絕假鈔的橫行，收據有著偽造的可能，在交易數據庫中信息不一致，數據庫被 DDOS 攻擊，交易數據被竄改，數據庫損毀，都是在傳統交易過程中曾出現過的窘境。

於 2009 年加密貨幣 - 比特幣的問世，以密碼學、網絡學與貨幣銀行學為基礎創建了新一代的網絡貨幣。各式網絡貨幣中又以比特幣最為廣泛使用，其防堵被竄改、公開交易數據檢視、使用者具匿名性、自動運作不須人為運營等等多項特性深受現今使用者喜愛。至今區塊鏈技術已成為 IBM、摩根大通、微軟、谷歌與英特爾等重點開發項目，被視為改善銀行運作效率、降低運營成本、提升信息安全、建立公開數據的最佳方法。為解決現金、收益及交易數據庫存在之問題，本文採用以區塊鏈為基礎的加密貨幣比特幣為基礎，進行商業化收銀系統開發。不僅是基於比特幣算法穩定、交易公開透明、不可被竄改等特性，同時本論文更加入監督標籤，促使在匿名交易轉為部分實名交易之過程中，監管部門能有更好的新興貨幣技術的提升，亦可建立自動化的稅務審查機制，大幅降低人事成本，進而實現提升交易系統信息之可靠度與其穩定性。

1.1 選題背景

追溯著加密貨幣市場的演進，於 2009 年時，比特幣並非第一個加密貨幣，在比特幣之前已經有著很多類似的加密貨幣開發實驗，但是一直無法做出一個穩定點對點式的電子現金系統，關於其製作瓶頸之部分將於後段章節闡述。在比特幣穩定發展之後，有著許多對比特幣有興趣的研究者，以穩定的比特幣系統為基礎修改了許多基本的協議。於 2011 年相繼創造出了貨幣，將其稱之為山寨幣。山寨幣早期較為著名包括有萊特幣 (LiteCoin, LTC)^[1]、狗幣 (DogeCoin, DOGE)^[2]、域名幣 (NameCoin, NMC)^[3]，於 2014 年也有人認為比特幣挖礦使用到了大量的哈希運算，這樣的大量運算也浪費了許多的社會資源，進而開發出較具意義的工作量證明挖礦算法，其中較為著名的如素數幣 (Primecoin, XPM)^[4]。於 2015 年底也誕生了現在最為著名的以太坊經典 (Ethereum Classic, ETC)^[5]、以太坊 (Ethereum, ETH)^[6]，以太坊最重大突破設計在於將編程語言虛擬機移植到了區塊鏈架構上，這使得區塊鏈技術不再僅止於點對點的電子現金系統，也創造出了屬於以太坊的編程語言 Solidity^[7]，使以太坊在虛擬機 (Ethereum Virtual Machine, EVM)^[8] 中可以使用 Solidity 創建智能合約，合約可以建構去中心化的應用程序，如去中心化的交易所，將交易所去中心化可以有效的防治 DDOS 攻擊^[9]，降低交

交易所因為黑客攻擊而倒閉的可能性。

1.1.1 加密貨幣市場

加密貨幣中最具代表性的是比特幣，但除了比特幣之外也存在許多模仿比特幣的加密貨幣，有的是為其利益，有的是鑒於比特幣的各種不足，進而希望藉由其他貨幣改善比特幣不夠完美之處。加密貨幣市場中有成千上萬種的加密貨幣，其中較廣為人知的加密貨幣會在 Cryptocurrency Market Capitalizations^[10] 的排行榜中出現，截至 2018 年 2 月 8 日該排行榜已經收入了 1510 種加密貨幣。在 Cryptocurrency Market Capitalizations 統計的數據當中，可知整體的加密貨幣市場，如圖 1.1 所示，於 2018 年 1 月 7 日創下了歷史新高，加密貨幣市場的總市值也高達了 829,579,000,000 美金，相當於五兆人民幣的總市值。



图 1.1 2017 加密貨幣總市值走勢圖^[10]

經由 Cryptocurrency Market Capitalizations 數據顯示，整體加密貨幣市場自 2013 年起已經高達 150 億美金，2014 年與 2015 年間總市值減少到近乎 2013 年的一半。針對比特幣的價格波動，論文 "Have the security flaws surrounding Bitcoin effected the currency's value?."^[11] 作出詳盡的市場調研，致力於探討在各個比特幣市場大事件中對比特幣價格的波動影響，針對影響的程度該論文給出影響指數，當中影響最為嚴重的是於 2014 年 2 月發生的日本交易所 Mt.Gox 倒閉事件，因為早期的加密貨幣市場中無完善的法律規範，各國對加密貨幣的接受度有所不同，日本對金融科技的接受度相較於較為開放的情況下成立了全世界第一家比特幣交易所 Mt.Gox，也因為交易所不夠普及，使得大部分的加密貨幣交易都集中在 Mt.Gox 交易所中，使 Mt.Gox 倒閉事件成為震盪市場價格重大因子之一，也造成 2014 與 2015 年的加密貨幣市場低迷。而在 2017 年，比特幣又以 2016 年總市值之 35 倍的姿態攀上新高點，主要是因為美國最大的期權交易中心芝

加哥期權交易所於 2017 年 12 月 10 日支宣布支持比特幣期貨交易，此舉將比特幣價格推升到 20,000 美金的歷史新高，圖 1.2 為 2013 年至 2018 年歷年的加密貨幣總市值的統計圖表。



图 1.2 加密貨幣歷年最高總市值折線圖^[10]

1.1.2 加密貨幣的優勢

於 2009 年 Satoshi Nakamoto 發布了比特幣系統，成為全世界第一個加密貨幣的雛形。其透明的交易信息、區塊鏈交易數據無法修改和刪除、匿名與自治系統等特性，促使區塊鏈技術衝破現存傳統中心化之金融機構技術上的藩籬。以下將逐一說明加密貨幣的七項優點，分別為區塊鏈結算系統不間斷的運行、遠距離支付、貨幣為使用者持有、開放和透明的交易信息、區塊鏈交易數據無法修改和刪除、交易匿名性以及自治性系統。

第一，區塊鏈結算系統不間斷的運行。基於區塊鏈技術與點對點網絡的架構，以比特幣為例，自 2009 年至今，所有的比特幣交易事件皆會存儲在比特幣區塊鏈當中，區塊鏈既無法刪除也無法修改，比特幣區塊鏈會以點對點網絡的方式存儲在比特幣網絡中的全節點^[12]，目前比特幣網絡中的全節點高達 10552 個。與傳統中心化的銀行數據庫相比，可能會因為銀行的服務器維護，導致交易無法順利進行，甚至可能有黑客的入侵導致銀行或是個人資產有重大的損失。點對點網絡提供穩定的數據庫元數據，不會因為數據庫的停機而無法繼續使用，實現其 24 小時不間斷之運作。

第二，基於點對點網路架構完成遠距離支付。於跨國匯款從美國轉帳至中國一百萬美金的場景中，需要經過的手續較為繁瑣，資金有可能需要經過多個國家才可以抵達目的地，在經過各個國家的過程中，需要支付各國的手續費，也需要等待各個國家辦理該業務的時間，即使當資金順利抵達了目的地銀行，目的地銀行也需要花將近三

至五日的工作日確認該筆金額的來源。屆時領款人亦需要前往銀行核實完整的身份驗證、解釋資金用途，才得以領取這筆跨國資金。比特幣系統當中，有著 24 小時不間斷運作的優點，也因為點對點網絡架構，使得比特幣無需經由傳統金融機構繁瑣的步驟完成國際匯款，於比特幣系統中無系統壅塞的情況下，平均 10 分鐘即可入帳，實現其短時間內即可完成遠距離支付之運作。

第三，加密貨幣為使用者持有。傳統的金融體系中，資金的存儲、流動往往需要經過銀行，使用者將所有的資產存入銀行，拿到的是一串數字的銀行餘額，銀行是一個中心化的機構，有著最高的權利。中央社的新聞^[13]指出，臺灣各地於 2017 年接連於土地銀行、日盛銀行、彰化銀行、京城銀行、兆豐銀行皆傳出銀行行員監守自盜的行為，總金額高達一億三千萬新臺幣。在比特幣系統中，比特幣有如金幣般存放在個人的比特幣地址當中，使用者為真實持有著貨幣，即使是比特幣系統亦無權利動用該筆比特幣資產，唯有比特幣地址的私鑰持有者，實現其只為持有者所用之安全基礎。

第四，公開的交易信息。基於區塊鏈架構，所有的交易信息皆公開的方式存儲於區塊鏈中，並且可信任與方便的取得元數據，以下將針對可信任與原數據進行探討。

1. 可信任：在公有鏈的基本架構上，所有的交易記錄都是公開透明的存儲在區塊鏈當中，比特幣網絡的使用者都可以檢視該筆交易，所有人都可以檢查每個交易記錄的正確性，公開的交易信息亦提升交易數據之可信性。
2. 元數據：除了以區塊鏈技術為基礎建構出可信任的系統之外，開放和透明的特性讓更多的開發商或新公司更容易獲得交易的元數據。畢竟，在傳統金融體系中，所有交易記錄均由中央金融機構存儲，從中央金融機構提取原始交易信息並不容易，區塊鏈的開放性和透明性促使金融公司降低了獲取原始數據努力的門檻。公司或學者可以透過元數據制定出可視化的開發計畫，甚至可以運用大量數據來分析以前從未見過的有價值觀點。

第五，區塊鏈交易數據無法修改和刪除。在區塊鏈結構中，通過嚴格驗證的所有信息都記錄在區塊鏈中，且使用者及系統平台都不賦與刪除及修改之權限。根據區塊鏈的特點，舊區塊的哈希值在連接區塊鏈的過程中，舊區塊的哈希值會被存儲在新區塊。只要區塊中的值被修改，即使僅有 1 bit 的變化，也會產出完全不同的哈希值，這也就是所謂的雪崩效應（Avalanche effect）^[14]。由於上述結構特性，區塊鏈中所有的信息都不會被改變，倘若區塊中記載的比特幣交易在其中一個比特幣全節點驗證的結果被竊改，則該區塊將不被比特幣系統接受。因此，所有已經存儲在區塊鏈中的交易記錄將不能被修改和刪除，進而實現其架構安全之穩固。

第六，區塊鏈系統中所有的使用者皆為匿名。現今社會中，個人信息保護已成為企業最重要的課題。在區塊鏈系統中創建的所有賬戶都不會與真實世界中的實體建立

直接關聯，也因為沒直接關聯所以建立匿名。區塊鏈系統中的所有賬戶都是由匿名個體創建，匿名的設計可以有效保護消費者的隱私。然而，VISA 交易與比特幣系統截然不同，在使用 VISA 支付系統前，我們會向 VISA 公司的主機提交大量個人信息，這可能會產生個人信息洩露的風險。在區塊鏈技術中，其匿名之特性可以有效地避免這個問題。

第七，自治系統。在區塊鏈系統中，區塊鏈的運作依賴於一些算法，包括共識算法。因此，在這種自治系統中，沒有人（例如節點或礦工）可以直接改變系統運作的規則。如果在比特幣系統中發現需要更正的嚴重錯誤，可以使用比特幣改進提案（Bitcoin Improvement Proposals, BIP）^[15] 升級比特幣系統。在實施比特幣改進提案之前，提議的比特幣改進提案需要得到比特幣系統中超過一定數量的礦工算力支持。由於這種以投票機制升級系統的門檻相當高，使得區塊鏈系統通常不會有大的變化，但也因為變化不大而相對穩定。

1.1.3 加密貨幣的劣勢

在區塊鏈技術中，有著三項項瓶頸，分別為每秒處理的交易量（Transactions Per Second, TPS）僅為七筆的限制、洗錢防治困難、低可擴展性，以下將逐一探討。

第一，每秒處理的交易量上限僅為七筆。圖1.3為國際上較為廣泛使用的支付系統之每秒支持交易量比較圖，以 VISA 為例，其以公司中心化運營的方式可以支持高達每秒 2,000 筆交易。但是以區塊鏈技術為基礎的比特幣最大能夠接受的每秒處理交易量僅為 7 筆。一般認為只要提升區塊大小的限制，就可以提升每秒處理的交易量。以下說明提升每秒處理的交易量的同時也存下述的兩項問題，分別為區塊鏈成長速度過快造成節點崩潰以及區塊同步延遲造成區塊鏈分岔：



图 1.3 Bitcoin、Western Union^[16]、PayPal^[17] 以及 VISA 每秒支持交易量比較圖^[18]

1. 區塊鏈成長速度過快會造成去比特幣全節點不堪負荷：從 2009 年至今的比特幣區塊鏈大小已達到 156GB，這樣的成長速度因為比特幣區塊大小的最大值被設置為 1MB。圖1.4為過去比特區塊鏈大小，圖中可以發現，於 2016 年開始，比特幣區塊鏈的成長速度為一直線，這表示著比特幣網絡中持續維持在供不應求的

图 1.4 比特幣區塊鏈成長走勢圖^[19]

狀況。為解決比特幣每秒支持交易量上限的窘迫，現今對比特幣的每秒處理交易量有許多優化的方案，其中包括解除比特幣區塊大小 1MB 的限制。在一個區塊上限為 1MB 的限制下，滿載的比特幣系統中，比特幣區塊鏈平均每十分鐘會增 1MB，每小時會增加 6MB，每天會增加 144MB，每月會增加 4.2GB，每年會增加高達 50GB，要達到 1TB 的區塊鏈大小還需要 8 年，在 8 年後的未來存儲 1TB 的數據量應該不會有太大的負擔。倘若解除 1MB 的區塊限制，在系統的每秒處理交易量看似可以接受更多的交易成倍成長，面臨 1TB 的比特幣區塊鏈數據會在更短的時間內出現，倘若存儲區塊鏈的成本超過了摩爾定律的成長曲線，會進一步造成使用者自願成為比特幣全節點的意願度降低，使得比特幣網絡的全節點數變少，導致比特幣點對點網絡逐漸轉向中心化網絡發展，失去一開始點對點網絡的意義。

2. 造成區塊鏈最新區塊同步延遲。對於區塊鏈的區塊同步延遲同時也會造成比特幣網絡的影響，J. Göbel 於“Increased block size and Bitcoin blockchain dynamics”^[20] 有著詳細的研究，在上修區塊大小上限的議題上，使用者自願成為比特幣全節點意願度下降，亦有機會在比特幣點對點網絡建構出的區塊鏈同步上造成延遲，在 1055 個比特幣全節點當中，平均每十分鐘會有礦工於其中一個全節點生成一個最新的區塊，該最新的區塊會以點對點網絡協議同步到 1055 個節點上。在比特幣系統中，長年來的過程經驗可以發現在礦工生成 1MB 的區塊後同步到全網節點可以在創造下一個區塊之前完成。倘若將區塊大小修改為 2MB 或是更大，會

使得比特幣全節點的最新區塊同步延遲現象更加明顯，同步延遲會使得區塊鏈分岔，造成 1055 個比特幣全節點的信息不一致，近一步造成整個比特幣點對點網絡崩潰。

第二，洗錢防治困難。匿名性為比特幣系統一大特色，比特幣的地址生成的熵是 256 bits，亂數是在 2^{256} 的組態空間中隨機選取，這樣的地址與現實生活中的身份並無任何關聯，使得黑市交易、洗錢防治變的困難，甚至有更為前沿的加密貨幣 Monero^[21] 導入了環簽章（Ring Signature）^[22] 算法、Zcash^[23] 導入零知識證明算法^[24]，使得原本公開透明的區塊鏈，變得無法檢視，進而造成加密貨幣在洗錢防治上更加的困難。2017 年由 Thibault de Balthasar and Julio Hernandez-Castro 所提出的論文 "An Analysis of Bitcoin Laundry Services."^[25]，致力探究比特幣匿名交易下的資金流動模型，試圖以機械學習的方法找出比特幣洗錢模型作為洗錢的工具，圖 1.5 為該論文針對黑市交易中的洗錢服務運營商 Darklaunder 進行洗錢機械學習識別有著傑出的成果。



图 1.5 Darklaunder 洗錢模型^[25]

第三，低可擴展性。區塊鏈架構為了防止個是的攻擊，所以在架構訂定以及接口設計都有著嚴謹的規範。以下將闡述造成比特幣低可擴展性的原因：

1. 修改比特幣協議製作添加外部信息的區塊鏈：比特幣區塊鏈技術是一個嚴謹的架構，倘若要創造可以支持外部信息的結構需要重新創造全新的加密貨幣，大部分的加密貨幣不支持外部輸入，外部的信息輸入皆無法保證信息的正確性，近一步

造成垃圾進垃圾出（Garbage in, garbage out, GIGO）的問題，倘若錯誤的信息存儲在無法刪除、修改的區塊鏈下，只是強化該筆錯誤信息的錯誤。如食品履歷區塊鏈，致力於將食品生產到超市的過程逐一記錄在區塊鏈上，但如果一開始在輸入信息時，無法保證其信息之正確性，該食品履歷區塊鏈則毫無意義。

2. 於區塊頭或交易信息添加外部信息：比特幣區塊鏈上，可以添加一些信息於區塊上，該信息會永久保存於區塊鏈上，除了在區塊上新增信息，在比特幣單筆交易信息上，亦可填寫一些私人信息，但這樣的空間大小有限，且現今的比特幣價格日趨上漲，比特幣交易手續費是以單筆交易大小計算，這將使得在交易中添加些個人信息變得更加昂貴。

在比特幣系統中，其區塊鏈僅用於記錄交易記錄，不能擴展更多功能和應用程序。有很多開發者希望將比特幣系統擴展到智能合約等其他應用程序。但是，後來發現改變原始比特幣系統框架是具有挑戰性的工作。因此，全球第二大加密貨幣以太坊（Ethereum, ETH）的作者 Vitalik 也創建了以太坊虛擬機（Ethereum Virtual Machine, EVM）。其所創建的智能合約可以在統一的以太坊平臺上運行，而以太坊也藉此突破了比特幣之技術瓶頸。

1.1.4 國際情勢分析

比特幣是一種全新的價值交換的媒介，因為過於前沿在各個國家所秉持的態度皆有所不同，大致可將個國家對比特幣的政策分為兩大類，分別為接納與禁止，在接納的分類當中又分為歡迎放任以及監管。在歡迎放任的政策下的國家包括伊朗、以色列、法國、美國、沙特阿拉伯王國以及烏克蘭，上述國家政策上皆已抱持著開放且不監管的方式接納加密貨幣，這些國家認為加密貨幣對於國家金融科技的發展具有前景；在分類在接納但是實施監管的國家中，包括韓國、菲律賓、英國、馬來西亞、俄羅斯、日本以及香港地區皆認為加密貨幣技術可能存在的高風險，且針對加密貨幣的洗錢防制以及非法集資進行嚴格管理，也進一步制定相關的法律，使得國家可以應對新型加密貨幣的糾紛；在禁止得分鐘中包括辛巴威、摩洛哥、印尼以及中國，這些國家認為比特幣是一個非法的貨幣，因為涉及到洗錢問題、難以實施外匯管制。但是對於區塊鏈技術上技術探討，在所有國家中都是蓬勃發展。如表1.1：

1.2 研究目標與內容

基於比特幣在各個國家的蓬勃發展且應用在相當多的領域，應用的領域包括交易所、去中心化交易所、兌幣所、遊戲點數、網路購物、資產的保存、募資以及遠距離支付，甚至是各國的銀行和金融機構都逐步投入人力及資金研究區塊鏈相關技術，甚至

表 1.1 各國政府對比特幣態度統整表

| 序號 | 國家 | 接納 | | 禁止 |
|----|-------------|---------------------|---------------------|-----------------|
| | | 歡迎/放任 | 監管 | |
| 1 | 伊朗 | 監管得當歡迎 比特幣發展 | | |
| 2 | 以色列 | 歡迎：欲發展 國際 ICO 中心 | | |
| 3 | 法國 | 放任：與實體 經濟無關 | | |
| 4 | 美國 | 暫不構成威脅 | | |
| 5 | 新加坡 | 不監管但注意 周邊活動 | | |
| 6 | 沙特阿拉 伯王國 | 加密貨幣尚未 成熟，不監管 | | |
| 7 | 烏克蘭 | 不屬於貨幣 | | |
| 8 | 韓國 | | 很快將監管 交易所 | |
| 9 | 菲律賓 | | 計劃監管 ICO | |
| 10 | 英國 | | 考慮監管 | |
| 11 | 馬來西亞 | | 正規化監管框架 | |
| 12 | 俄羅斯 | | 2018 年 7 月前 完成立法 | |
| 13 | 日本 | | 任命加密貨幣 監察長 | |
| 14 | 香港 | | ICO 須受法規監管 | |
| 15 | 辛巴威 | | | 定法前，屬非法 |
| 16 | 摩洛哥 | | | 禁止加密貨幣交易 |
| 17 | 印尼 | | | 2018 年前 全面禁止 |
| 18 | 中國 | | | 禁止加密貨幣 |

是訂定各國相關的法律。在比特幣資產的流動上存在著許多的問題，第一，比特幣的帳戶是由亂數產生器生成，該比特幣帳戶與在現實生活中的使用者不無直接關聯，使得比特幣洗錢變更加盛行。第二，現今的比特幣交易皆為匿名對匿名支付，並無正式的收據開立，使得消費者在購物後，倘若商品存在問題，無法得到法律上的保障。第三，因為比特幣是匿支付匿名交易，使得政府無法查閱商家的收入進一步課徵稅收，造成逃漏稅的灰色地帶。

本文將解決上述三項問題，設計與實現一個区块链的实名交易實時監督系統達到下列幾點目標：

1. 將商家比特幣地址實名制，構建政府可以檢視商家營收的区块链的实名交易實時監督系統，使店家可以進行註冊，再由政府進行商家註冊後的審查。
2. 構建商家端行動收銀與交易明細系統，使得商家可以開立收據以保障消費者應有的消費者權益。
3. 使得政府可以課徵加密貨幣的相關稅收。
4. 以多重簽章技術提升比特幣交易速度，達到實時的比特幣交易系統。

為實現上述目標，本文首先將詳細探討區塊鏈技術的優勢，藉由深度了解區塊鏈技術的優勢可以取之優點，如區塊鏈是一個不間斷、不可篡改、去中心化、公開數據庫、穩定的系統。第二，探討區塊鏈技術的劣勢，區塊鏈技術並非完美，存在著洗錢、逃漏稅、無法保障消費者權益以及交易速度慢的問題，藉由瞭解問題，並設計系統解決。第三，交易模型分析，透過交易模型分析探討在現金、電子貨幣以及加密貨幣存在的交易模型，並在諸多交易模型中發現，匿名支付給實名的交易行為，可以保障消費者隱私，同時保障消費者權益，更使得政府可以對商家的營收進行檢視。第四，系統設計，完成交易分析，並著手設計數據庫模型、主系統、三個子系統的系統架構。第五，系統實現，利用 Java 編程語言實現主系統以及三個子系統。第六，系統優化，比特幣的多重簽章算法，實現實時的比特幣交易監督系統。第七，功能測試，測試以 Java 編成程序所有的功能是否能夠完整運作。第八，性能測試，測試在未進行優化的系統與已經優化的系統性能的差異。

1.3 論文組織結構

在本節中將逐一說明本論文的組織結構共分為七章，分別為緒論、相關理論與技術、系統需求分析、系統設計、系統實現、系統測試以及总结与展望：

第一章：簡述區塊鏈技術解決現有數據庫存在的信息不一至、信息安全問題以及數據庫損毀問題。說明在加密貨幣市長中不僅只是比特幣，亦有許多新穎技術為特色的貨幣競爭。簡要探討區塊鏈技術的優點，逐一探討比特幣的缺點及區塊鏈技術瓶頸。

最後於研究目標與內容當中，闡述本文的要解決的問題以及如何解決。

第二章：介紹比特幣的發展，闡述比特幣地址生成運用到的算法、生成過程以及多重簽章算法。並詳細說明區塊鏈結構及所有變數與區塊鏈相關的意義。

第三章：首先對五種交易模型進行分析，進而得知匿名對實名的交易，既可以保障消費隱私且兼具保有消費者權益，並透過需求分析探討本系統需構建的功能模塊。

第四章：設計区块链的实名交易监督系统的架構，以及說明三個以系統的模塊功能。設計本系統的數據庫結構，設計 ER 模型，並闡述本系統的運作流程。最後則將多重簽章算法引入本系統，使得本系統成為實時交易监督系统。

第五章：展示商店和商品信息管理子系統、商家移動裝置收款及交易子系統以及客戶端移動支付和交易子系統的介面。

第六章：透過功能測試逐一檢視所有功能是否能夠正常運行，經由性能測試分析原始系統與優化後的系統效能之間的差異。

第七章：說明比特幣區塊鏈系統透過本文所提出之系統，可以達到保障消費者權益同時也兼顧保有效費者隱私，也使得政府可以透過該系統檢視交易明細課徵稅收，而商家亦可對庫存及產品進行管理，達到三贏的局面。

第二章 相關理論與技術

2.1 比特幣 (Bitcoin)

表 2.1 比特幣簡介

| | |
|-----------------|-----------------------|
| 第一個區塊生成時間 | 2009 年 1 月 3 日 |
| 比特幣預計總產量 | 21,000,000 BTC |
| 比特幣目前總產量 | 16,921,800 BTC |
| 最新區塊高度 | 513743 |
| 比特幣總市值 (人民幣) | 5 兆 |
| 比特幣全節點的數量 | 11147 個 |
| 單日比特幣交易金額 (BTC) | 124,017.02430718 BTC |
| 單日交易筆數 | 196,606 筆 |
| 比特幣區塊鏈大小 | 188.89 GB |
| 平均區塊大小 | 0.75 MB |
| 平均生成單一區塊所需時間 | 9.67 分鐘 |
| 單日產出比特幣數量 | 1,775 BTC |
| 挖礦難易度參數 | 3,290,605,988,755 |
| 全網挖礦算力 | 23,555,075.18 THash/s |

比特幣 (Bitcoin, BTC) 表2.1是比特幣系統的相關參數簡介，比特幣是一個點對點式的電子現金系統，集成了非對稱式金鑰密碼學 (Asymmetric Key Cryptography)^[26]、簽章密碼學 (Signature Cryptography)^[27]、零知識證明密碼學 (Zero Knowledge Proof Cryptography)^[28]、哈希函數密碼學 (Hash Function Cryptography)、共識算法 (Consensus Algorithm)^[29] 諸多技術建構了一個分散式、不需要仰賴中心化機構加以維護的交易帳本。在接下來的章節中將逐一進行詳盡的說明每個技術在各個環節中所扮演的角色。

2.2 比特幣地址 (Bitcoin Address)

比特幣地址為比特幣的載體，深入瞭解比特幣的地址生成相關算法、比特幣地址生成過程、多重簽章，可以近一步應用在區塊鏈的實名交易監督系統。

2.2.1 比特幣地址生成相關算法

在點對點的現金系統中，首先必須先生成一個地址，在比特幣的協議中有著既定的程序生成地址。運用到的技術包括亂數產生器、secp256k1^[29]、SHA-256 (哈希函數)^[30]、

RIPEMD-160（哈希函數）^[31]、Base58^[32]。接下來將詳述每一個函數的運作過程以及意義，最後說明比特幣交易地址生成的每一個步驟。

（一）亂數產生器（Random number generator）

亂數在密碼學中是個相當重要的一環，在比特幣系統中更是重要，畢竟生成的亂數會變成比特幣的私鑰，私鑰是簽署資產轉移的唯一方式，在比特幣地址中的亂數產生器會產出一個 256 bits 長度的亂數，也就是私鑰，256 bits 的長度可以表現的組態空間為 2^{256} ，換算成十進位表示為 1.1579209×10^{77} ，要在這組態空間中，以亂數產生同樣的一把私鑰是一件困難的事，但也有國際的實驗室^[33] 團隊正在努力的窮舉比特幣 2^{256} 的組態空間，如圖2.1所示，根據 LBC 公佈的數據顯示，目前已經完成了 2.330109×10^{16} 個地址探索。雖然 10^{16} 的級別與 10^{77} 的級別相距甚遠，但 LBC 已探索的組態空間中擊中了 15 個比特幣地址，該團隊也成功將這 15 個地址下的 1.180899 個比特幣轉走。

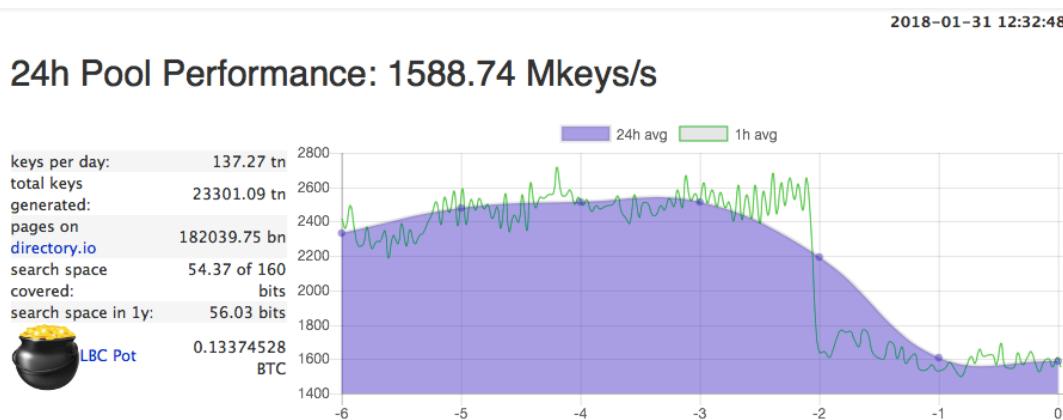


图 2.1 LBC 穷举比特币私钥算力状态图^[33]

如何建構一個亂數，在過往的亂數產生器往往會加入時間作為參數，但對於一個攻擊者而言，只需要去猜測在這段時間內目標者所有生成的可能性有極高的機率可猜出亂數。而亂數在密碼學中常會是一把私鑰的生成，在 https 協議中，服務器端與客戶端，建立一個加密連線的過程中也需要一個亂數去建立一個高安全性的加密通道-傳輸層安全性協定（Transport Layer Security， TLS）^[34]，在 SSH 協議中也採用了亂數。

在過去的歷史事件中，發現 Android 手機版以及平板版的亂數產生器中存在著不隨機，於 2013 年 8 月比特幣開發者 Mike Hearn 提及“All private keys generated on Android phones/tablets are weak and some signatures have been observed to have colliding R values”^[35]，Bitcoin.org 也發布了警告^[36] 簡要說明該事件的原因，以及表明影響到的比特幣錢包客戶端有 Bitcoin Wallet、BitcoinSpinner、Mycelium Bitcoin Wallet、blockchain.info。這樣的錯誤源於 Android 本身支持的亂數產生器並不隨機，隨後 Android 解釋了亂數的問題

並加以修正。在這 Android 手機亂數不夠亂的事件中，有自願者自發性地公佈自己的損失狀態，總金額為 55.82152538 個比特幣^[37]，但因為比特幣屬於被動的性質，無人主動回報即不會加入統計中，所以總損失估計會超過 55.82152538 個比特幣。

(二) 非對稱事加密算法 Secp256k1

在密碼學中有分對稱式加密與非對稱式加密，對稱式加密又分為信息流加密與信息塊加密，信息流加密著名的是由美國密碼學家 Ron Rivest 教授設計，包括 RC2 (1987 年)^[38]、RC4 (1987 年)^[39]、RC5 (1994 年)^[40]、RC6 (1998 年)^[41]；信息塊加密著名的有數據加密標準 (Data Encryption Standard, DES, 1975 年)^[42]、三重數據加密算法 (Triple Data Encryption Algorithm, Triple DES, 1998 年)^[43]、高級加密標準 (Advanced Encryption Standard, AES, 1998 年)^[44]；非對稱式加密最廣為人知的有 RSA (Rivest – Shamir – Adleman, 1977 年)^[45]、橢圓曲線密碼學 (Elliptic curve cryptography, ECC, 1985 年)^[46]。非對稱式加密與對稱式加密最大的不同在於，對稱式加密在加密解密的過程中只需要一把鑰匙，而非對稱式加密會生成兩把鑰匙分別為私鑰與公鑰，在算法的設計上一開始會以亂數產生一把私鑰，再經由非對稱式加密算法推導出公鑰，推導出的公鑰在非對稱式密碼學中並無直接的方法可以反推至私鑰，如此一來確立私鑰的安全性。非對稱式密碼的使用場景有兩種，第一種是希望收到加密信息的使用者 Alice，Alice 會生成私鑰存儲在自己本地端的電腦中，並將推導出的公鑰公佈在網絡上，這時希望聯繫 Alice 的使用者 Bob 在網絡上取得公鑰後，Bob 會以 Alice 的公鑰進行加密，之後將密文寄送給 Alice，在傳遞信息的過程中，即使網絡存在著監聽，也無法將信息順利解密，唯有 Alice 收到信息後使用 Alice 原本產生該公鑰的私鑰，才可以解出明文。第二種則應用在比特幣的交易之數字簽名以及交易驗證交易，比特幣地址的創建過程中會透過 secp256k1 生成私鑰公鑰對，在創建比特幣交易的過程中，使用該地址的私鑰對該地址未花費的輸出 (Unspent Transaction Output, UTXO) 進行數字簽名，完成數字簽名後會與公鑰以及交易信息一起廣播到比特幣網絡的交易緩存池當中，比特幣交易緩存池存在於所有比特幣全節點當中，主要存儲所有未被收入到比特幣區塊鏈內的所有交易，也就是零確認交易，等待礦工將該筆交易收入至比特幣區塊鏈當中。比特幣採用的 secp256k1 是屬於橢圓曲線密碼學中的一個版本，不同的橢圓曲線版本的差異在於不同的初始參數，包括橢圓曲線方程

$$y^2 = x^3 + ax + b$$

$p=FFFFFFFFFFFFFFFFFFF...FFFFFEFFFC2F$
為巨大的素數、G 點被稱為生成點的常數點亦稱為基點。至於為什麼選擇 ECC 而非

RSA 的主要原因，其一在於 ECC 在生成密鑰對所需的時間更佳快速，圖2.2為 Nicholas Jansma 於 2004 年針對 ECC 與 RSA 的密鑰對生成時間與數字簽名所需時間的論文^[47]顯示，當 ECC 產生 571 bits 的密鑰長度，RSA 要達到相同的安全性需要生成 15360 bits，這也導致生成時間產生高達 471 倍之差距。

表 2.2 ECC 與 RSA 的密鑰對生成時間比較表^[47]

| 密钥长度 | | 时间(秒) | |
|------|-------|-------|--------|
| ECC | RSA | ECC | RSA |
| 163 | 1024 | 0.08 | 0.16 |
| 33 | 2240 | 0.18 | 7.47 |
| 283 | 3072 | 0.27 | 9.8 |
| 409 | 7680 | 0.64 | 133.9 |
| 571 | 15360 | 1.44 | 679.06 |

除了在密鑰對生成時間 ECC 有著比 RSA 更高效的算法外，在安全性上 ECC 可以更短的密鑰長度達到與 RSA 相同的安全強度，L Ducas 針對 ECC、RSA、BLISS^[48]做出了深度的安全性探討^[48]，圖2.3同樣達到 80 bits 的安全性級數，RSA 1024 需要 1024 bits，ECDSA 160^[49] 僅需要 160 bits，該篇論文除了探討 RSA 與 ECDSA 之外，更大的部分在闡述量子計算機對於既有的傳統密碼帶來的抨擊，有機會快速窮舉 2^{256} 的比特幣私鑰，在未來量子計算機的蓬勃發展擁有 2000 qbits 運算能力，量子計算機可以快速窮舉破解所有的比特幣私鑰。因此發展針對量子計算機設計的數字簽名算法成為密碼學上嶄新的議題，而 BLISS 則為針對量子計算機所設計的抗量子計算的簽章算法。

表 2.3 算法 BLISS、RSA、ECDSA 安全級數比較圖表^[48]

| 算法 | 安全性 | 签名大小 | 私鑰大小 | 公鑰大小 | 簽名(微秒) | 簽名/秒 | 验证(微秒) | 驗證/秒 |
|-----------|-------------|--------|--------|--------|--------|------|--------|------|
| BLISS-0 | <=60bits | 3.3kb | 1.5kb | 3.3kb | 0.241 | 4k | 0.017 | 59k |
| BLISS-I | 128bits | 5.6kb | 2kb | 7kb | 0.124 | 8k | 0.03 | 33k |
| BLISS-II | 128bits | 5kb | 2kb | 7kb | 0.48 | 2k | 0.03 | 33k |
| BLISS-III | 160bits | 6kb | 3kb | 7kb | 0.203 | 5k | 0.031 | 32k |
| BLISS-IV | 192bits | 6.5kb | 3kb | 7kb | 0.375 | 2.5k | 0.032 | 31k |
| RSA-1024 | 72-80bits | 1kb | 1kb | 1kb | 0.167 | 6k | 0.004 | 91k |
| RSA-2048 | 103-112bits | 2kb | 2kb | 2kb | 1.18 | 0.8k | 0.038 | 27k |
| RSA-4096 | >128bits | 4kb | 4kb | 4kb | 8.66 | 0.1k | 0.138 | 7.5k |
| ECDSA-160 | 80bits | 0.32kb | 0.16kb | 0.16kb | 0.058 | 17k | 0.205 | 5k |
| ECDSA-256 | 128bits | 0.5kb | 0.25kb | 0.25kb | 0.106 | 9.5k | 0.384 | 2.5k |
| ECDSA-384 | 192bits | 0.75kb | 0.37kb | 0.37kb | 0.195 | 5k | 0.853 | 1k |

(三) 哈希算法 SHA-256

SHA-256 是 SHA (Secure Hash Algorithm, FIPS 182-2)^[30] 哈希算法的家族之一。SHA 家族當中有著四大分支，分別為 SHA-0、SHA-1、SHA-2 和 SHA-3，如表2.4所示。各種哈希算法的差異在於運算初始變數、算法所採用的運算子、接受的信息長度以及迴圈樹的不同。上述的參數差異皆由聯邦資訊處理標準 (Federal Information Processing Standards, FIPS) 中定義。表2.4中 MD5 不為 SHA 家族成員之一，但 MD5 為最早被廣泛使用的哈希算法，因此作為借鑒的標準。SHA-0 為 SHA 家族中被最早提出的架構，輸出的長度為 160 bits，而 SHA-1 提出後並無太大的變動。哈希算法的主要功能在於，將信息或是檔案建立一個對一的指紋，也就是哈希值，而該指紋長度會依照算法的設計輸出的長度而略有不同，表2.4中顯示的輸出哈希值長度，而長度也意味著該指紋的組態空間應設的大小。倘若有著不同的輸入，但映射到了相同的指紋，則將此現象稱之為碰撞。通常在信息安全領域中，只要發現該哈希算法存在著碰撞，就會被棄用。甚至是專家們提早數年提出警告，要求提早更換該哈希算法，並更換上新制定的哈希算法做應用。哈希算法的功能包括藉由生成哈希值進行檔案校驗、工作量證明算法設計以及區塊鏈中的哈希指針。

表 2.4 MD5、SHA-0、SHA-1、SHA-2 和 SHA-3 比較表

| 算法 | 分支 | 輸出 哈希值 長度 (bits) | 最大輸入 信息長度 (bits) | 迴圈 次數 | 使用到的運算子 | 碰撞 攻擊 (bits) |
|-----------|----------|---------------------------|------------------------|----------|--|--------------------|
| MD5 參考 | - | 128 | ∞ | 64 | And, Xor, Rot, Or, Add (mod 2^{32}) | < 64 已碰撞 |
| SHA-0 | - | 160 | $2^{64} - 1$ | 80 | | < 80 已碰撞 |
| SHA-1 | - | | | | | < 80 已碰撞 |
| SHA-2 | SHA-224 | 224 | $2^{64} - 1$ | 64 | And, Xor, Rot, Or, Shr, Add (mod 2^{32}) | 112 |
| | SHA-256 | 256 | | | | 128 |
| | SHA-384 | 384 | $2^{128} - 1$ | 80 | And, Xor, Rot, Or, Shr, Add (mod 2^{64}) | 192 |
| | SHA-512 | 512 | | | | 256 |
| SHA-3 | SHA3-224 | 224 | ∞ | 24 | And, Xor, Rot, Not | 112 |
| | SHA3-256 | 256 | | | | 128 |
| | SHA3-384 | 384 | | | | 192 |
| | SHA3-512 | 512 | | | | 256 |

1. 生成哈希值進行檔案校驗：哈希值在過往的應用中，往往作為檔案完整性的校驗，軟件供應者會在網站中提供各式不同算法的哈希值供給使用者下載完軟件之後，使用者將檔案輸入哈希算法中生成出哈希值後進行比對。但倘若該哈希算法存

碰撞的發生，這會使得不同的檔案存在著同樣的哈希值。這也意味著軟體提供平台所提供的軟體可能存在著摻入惡意代碼後，還能生成同樣的哈希指紋，失去了檔案完整性的校驗的功能，這便成為信息安全中的重大漏洞，因此在表2.4中的 MD5、SHA-0 以及 SHA-1 皆為已經棄用之算法。

2. 工作量證明算法設計：起出工作量證明算法的概念為設計一個求解困難，但驗算的過程中卻相當簡單快速。如對一個大數做因式分解是一個相當困難的事，但要驗算其結果僅需要將所有的解相乘確認是否為該大數即可證明。同樣的理念在比特幣系統中是以 hash-puzzle 的方式實現，hash-puzzle 是利用哈希函數有著不可預期的特性，不可預期指的是假設輸入連續性的數值 1 到 n 進入到哈希函數生成哈希值，而生成的哈希值無法觀察出關聯性，可說是完全不相關的數值，且無法預期下個輸入的哈希值輸出。比特幣系統中的困難度變數，在 hash-puzzle 中定義了何謂真正的答案。在 SHA-256 當中輸出的值為 256 bits 的哈希值，困難度變數則規定了在這 256bit 當中，自最左邊起必須為零的位數的門檻，而在困難度變數要求必須為零的位數變多，則意味著要在連續性的數值 1 到 n 的 hash-puzzle 中，符合門檻的解越少。在 hash-puzzle 中，求得一個符合困難度變數的哈希值是困難的，但要驗算求得的解是否符合困難度的門檻相當快速，這符合起出工作量證明算法的概念。

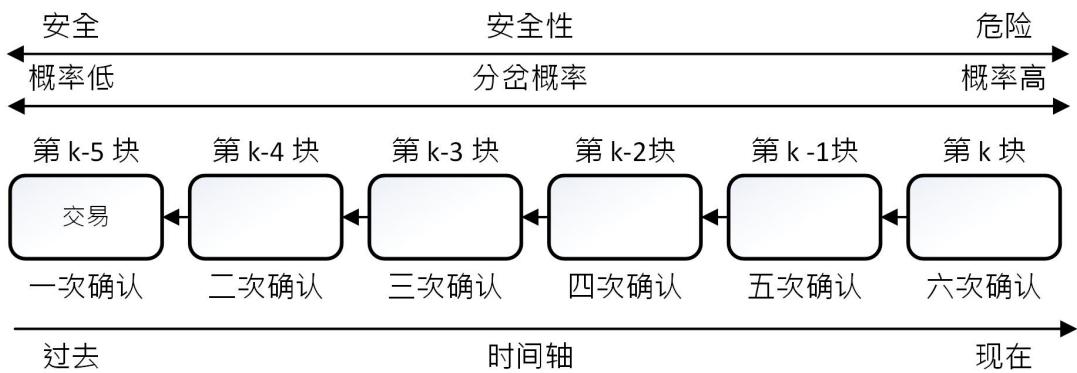


图 2.2 區塊疊加示意圖

3. 區塊鏈中的哈希指針：比特幣區塊鏈的區塊頭當中，當前區塊存儲著前區塊頭的雙重哈希的哈希值。當前區塊則會如資料結構鏈結串列一樣，直到鏈結到第一個比特幣區塊創世區塊，區塊鏈中的哈希指針將區塊鏈結在一起，也就形成了比特幣區塊鏈的基礎。在比特幣挖礦的過程中，礦工參與著最新區塊的 hash-puzzle 的解題，尋找一個符合困難度變數的輸入。但 hash-puzzle 的工作證明當中解可能不只一個，但只要符合困難度變數的解都可以創建一個新的區塊，屆時就會在當前的區塊上同時產生分岔，分岔的區塊在比特幣系統中皆為有效，但經過多個區

塊的疊加之後，變可以抉擇出最長的鏈，該最長鏈則成為主鏈，而其他的分岔鏈，則成為孤兒塊被丟棄，當中曾記載的交易信息無效，造成該分岔鏈的交易信息回溯。比特幣區塊鏈的分岔好發於最新的區塊，如圖2.2所示，而越舊的區塊則越不易發生，所以在比特幣交易中默認該筆交易要在六個區塊確認後，該公司才會承認該筆交易有效。

(四) Base58 編碼

表2.5為Base58編碼表，Base58編碼的首次出現自Satoshi Nakamoto所提出的論文^[50]當中，Base58編碼是源自於Base64編碼表，如表2.6中所示。Base64編碼中包括大寫英文26個字母、小寫英文26個字母、阿拉伯數字以及字符"/"和"+"。在比特幣地址生成過程中Base58的功能是將比特幣地址的公鑰哈希值重新編碼，比特幣地址的公鑰哈希值是二進制的資料型態，既使是將二進制碼轉換成十六進制碼輸出也是對人類辨識上有一定的不便性。倘若採用Base64對比特幣公鑰哈希值進行編碼有效縮短二進制碼的長度，但Base64的編碼存在著不適於作為地址的特殊字符"+"和"-".在Base58編碼中移除了特殊字符之外，移除了較不易人類判讀的相關字符數字"0"與大寫英文字母"O"，因為該兩字符在不同字型的體現相當相似，大寫英文字母"I"以及小寫英文字母"l"，在人類判讀上也有些為相似。因此於Base64移除上述6個字符後，便形成了Base58編碼表。值得一提的是，Base58編碼與Base64編碼表中的字符排序有些許異動，Base58編碼表將數字的部分一致了最前面，這使得比特幣地址以Base58編碼後的第一個字符呈現為"1"的主要原因。

2.2.2 比特幣地址生成過程

於上述章節中已經詳細說明比特幣在地址生成的過程中，使用到的所有的相關算法技術以及算法在現今的比特幣系統中存在的問題，於本節中將闡述如何將比特幣地址生相關算法帶入比特幣地址生成的過程中，地址生成的總共分為八個步驟，分別為生成私鑰、生成公鑰、生成公鑰SHA-256、生成公鑰SHA-256的RIPEMD-160、取得版本號、校驗碼生成、版本號、公鑰SHA-256的RIPEMD-160和校驗碼合併，最後一步則是將合併的結果以Base58編碼創建出一個比特幣地址。

1. 生成私鑰：使用亂數產生器產生一個長度為256 bits的亂數，而此亂數即成為該比特幣地址的私鑰。在比特幣的系統當中，私鑰可以透過橢圓曲線簽章算法secp256k1簽名一筆交易，廣播治比特幣網路當中。因為私鑰為該地址資金轉移的關鍵，黑客攻擊的對象皆會聚焦在比特幣私鑰，因此私鑰的保存成為比特幣系統中最為熱門的課題之一。

表 2.5 Base58 編碼表

| 数值 | 字符 | 数值 | 字符 | 数值 | 字符 | 数值 | 字符 |
|----|----|----|----|----|----|----|----|
| 0 | 1 | 16 | H | 32 | Z | 48 | q |
| 1 | 2 | 17 | J | 33 | a | 49 | r |
| 2 | 3 | 18 | K | 34 | b | 50 | s |
| 3 | 4 | 19 | L | 35 | c | 51 | t |
| 4 | 5 | 20 | M | 36 | d | 52 | u |
| 5 | 6 | 21 | N | 37 | e | 53 | v |
| 6 | 7 | 22 | P | 38 | f | 54 | w |
| 7 | 8 | 23 | Q | 39 | g | 55 | x |
| 8 | 9 | 24 | R | 40 | h | 56 | y |
| 9 | A | 25 | S | 41 | i | 57 | z |
| 10 | B | 26 | T | 42 | j | | |
| 11 | C | 27 | U | 43 | k | | |
| 12 | D | 28 | V | 44 | m | | |
| 13 | E | 29 | W | 45 | n | | |
| 14 | F | 30 | X | 46 | o | | |
| 15 | G | 31 | Y | 47 | p | | |

表 2.6 Base64 編碼表

| 数值 | 字符 | 数值 | 字符 | 数值 | 字符 | 数值 | 字符 |
|----|----|----|----|----|----|----|----|
| 0 | A | 16 | Q | 32 | g | 48 | w |
| 1 | B | 17 | R | 33 | h | 49 | x |
| 2 | C | 18 | S | 34 | i | 50 | y |
| 3 | D | 19 | T | 35 | j | 51 | z |
| 4 | E | 20 | U | 36 | k | 52 | 0 |
| 5 | F | 21 | V | 37 | l | 53 | 1 |
| 6 | G | 22 | W | 38 | m | 54 | 2 |
| 7 | H | 23 | X | 39 | n | 55 | 3 |
| 8 | I | 24 | Y | 40 | o | 56 | 4 |
| 9 | J | 25 | Z | 41 | p | 57 | 5 |
| 10 | K | 26 | a | 42 | q | 58 | 6 |
| 11 | L | 27 | b | 43 | r | 59 | 7 |
| 12 | M | 28 | c | 44 | s | 60 | 8 |
| 13 | N | 29 | d | 45 | t | 61 | 9 |
| 14 | O | 30 | e | 46 | u | 62 | + |
| 15 | P | 31 | f | 47 | v | 63 | / |

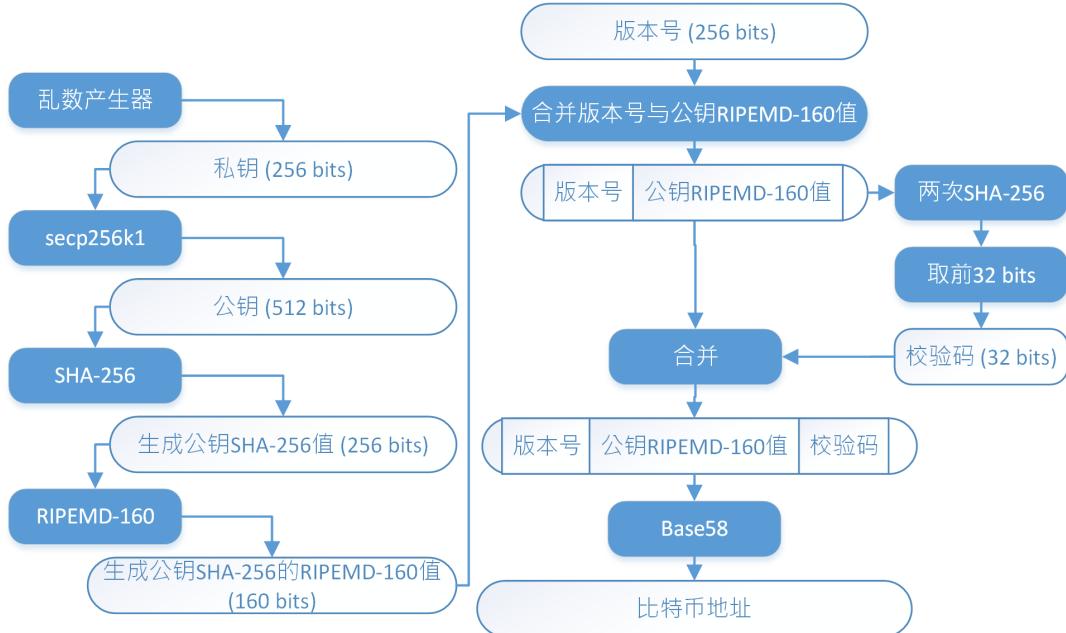


图 2.3 比特幣地址生成流程圖

2. 生成公鑰：在以亂數產生器生成私鑰之後，接下來將運用到非對稱密鑰學中的私鑰公鑰轉換算法，於比特幣系統中採用的是 secp256k1，secp256k1 是橢圓曲線密鑰學中的其中一個版本，不同的版本差異在於採用不同的變數。於比特幣地址生成的過程中，secp256k1 負責將上步驟生成的私鑰透過橢圓曲線算法計算出公鑰，生成的公鑰長度為 512 bits，比前步驟 256 bits 的私鑰多了一倍的長度。在比特幣交易中，私鑰可以簽署一筆交易，在簽署完之後便會將公鑰、簽名以及交易信息廣播至比特幣網路當中等待被收入到比特幣區塊鏈當中，此時該筆交易準備被存儲在比特幣交易緩存持當中，屆時可以使用 secp256k1 算法對透過公鑰、簽名以及交易信息進行驗算，倘若計算出的值為真則為有效，則會讓該筆交易繼續待在交易緩存持當中，等待被收入區塊鏈內。
3. 生成公鑰 SHA-256：SHA-256 為 SHA 家族之一，也是哈希算法的一種，因此符合哈希算法的特徵包括雪崩效應、不可預測、不可逆（單向性）以及校驗檔案是否完整性的諸多特性。在該步驟將前步驟生成長度為 512 bits 的公鑰作為 SHA-256 的輸入，產出長度僅為公鑰長度的一半的公鑰 SHA-256 哈希值。算該步驟使得知道公鑰 SHA-256 的攻擊者，更加難以用窮舉攻擊推導出該地址的公鑰。
4. 生成公鑰 SHA-256 的 RIPEMD-160：RIPEMD-160 也是哈希算法的一種，特色與哈希算法的特徵相同，RIPEMD-160 與 SHA-256 較為不同的部分在於 RIPEMD-160 生成的哈希值長度為 160 bits。在前步驟中，對公鑰進行 SHA-256 計算，公鑰已經有了第一層的保護，而在該步驟中，再次透過 RIPEMD-160 取得哈希值，這

使得攻擊者既使取得比特幣地址，必須針對 RIPEMD-160 進行破譯，再進一步對 SHA-256 才有機會取得該地址的公鑰，也因為這樣的設計，對於僅收取比特幣不曾花費比特幣的比特幣地址，於黑客攻擊上造成相當大的難度。

5. 取得版本號：在比特幣系統初始的設計中，已經定義了些不同功能的比特幣地址，這些特殊功能的比特幣地址有著特殊的版本號，最為常見的為以"1"為頭的比特幣地址，該地址為比特幣系統中最早被使用且最為普遍的地址版本，該地址為一把私鑰進行比特幣地址推倒，所以僅需要一把鑰匙就可以移動該地址下的比特幣資產；第二種是以"3"為地址開頭的比特幣地址，該地址採用多重簽章技術，該技術為後續經過多項 BIP 才完成落實於比特幣系統當中，該地址生成的過程中，。在第五個步驟中會加入版本號加以區分不同的地址。
6. 校驗碼生成：校驗碼為比特幣地址生成過程中重要的一環，可在支付比特幣的過程中降低因為手誤而將比特幣轉入到不存在（不符合比特幣地址生成規則）地址的可能性。對公鑰 SHA-256 的 RIPEMD-160 再做兩次 SHA-256，取該哈希值前 32 bits 的值作為校驗碼。
7. 版本號、公鑰 SHA-256 的 RIPEMD-160 和校驗碼合併：版本號、第四個步驟的產生之公鑰 RIPEMD-160 及第五個步驟產生之校驗碼合併。
8. 合併的結果以 Base58 編碼：將第六步驟組進行合併組合的結果，利用 Base58 進行編碼，Base58 修改自 Base64，其與 Base64 最大不同之處在於移除了"0"、"O"、"I"、"l"、"+"、"/" 的字符，可以降低人工在判讀地址的錯誤率。

2.2.3 多重簽章（Multi-Signature）

比特幣區塊鏈技術，雖然已經利用工作量證明的方式解決了雙重支付（Double-spending）問題^{[51][52]}，但工作量證明的算法所設定之題目困難度會直接影響到每一個比特幣區塊的產出時間，這個比特幣區塊的產出時間也考慮到比特幣全節點於全世界各地的網絡同步狀況，倘若今天的區塊生成時間過短會造成全世界的比特幣節點之區塊數據不一致，這樣的數據不一致將導致比特幣區塊鏈出現分岔，在更嚴重一點甚至會造成比特幣網絡的瓦解。

雙重支付問題存在於比特幣交易在未被區塊鏈確認收入到區塊鏈之前，都有機會受到惡意的攻擊者雙重支付同一筆款項。現今的比特幣區塊產出速度為十分鐘一塊，即便附上足夠的手續費也須等待將近十分鐘的時間，倘若是在手續費不足的情況下，該筆比特幣交易甚至會在比特幣交易緩存池中滯留一週的時間。在手續費足夠的情況下，十分鐘的確認時間會對實體店面的小額交易處理非常的不友善，為了在既有的比特幣區塊鏈的框架底下能夠提升交易速度，因此 Green Address 技術致力於在一開始創建

交易的同時管控雙重支付交易的發生，他們採用了 2-of-2 多重簽章，也就是創建一個特殊的比特幣地址，這個比特幣地址的持有人有兩個代表人，分別為使用者與 Green Address 機構節點，這筆交易的建立必須要雙方同時簽署交易才被允許廣播至比特幣網絡中。若是遇到交易塞車，且節點緩存池空間不足的問題時，比特幣節點會優先遺棄手續費最低的交易，視同該筆交易不曾存在過，故若真的遇到交易被遺棄的情況，Green Address 機構節點也會透內部的數據庫記錄再次廣播此筆交易，並確保此筆交易可以被收入至區塊內。Green Address 機構節點也就成為了交易創建的把關者，過濾所有的雙重支付攻擊的發生，也避免交易因為比特幣網絡塞車而交易被礦工遺棄的情形。在這樣的機制下，只要是用 Green Address 錢包交易即可確認雙重支付攻擊是不會發生的，對商家或是收款人而言，可以得到在即時交易中不被雙重支付攻擊的保障，提升在未進入區塊鏈的交易可確定性，進而創造出即時交易的可行性。

(一) Green Address 錢包生成過程

此節將詳細闡述 Green Address 錢包生成過程的重要步驟，如圖2.4所示。

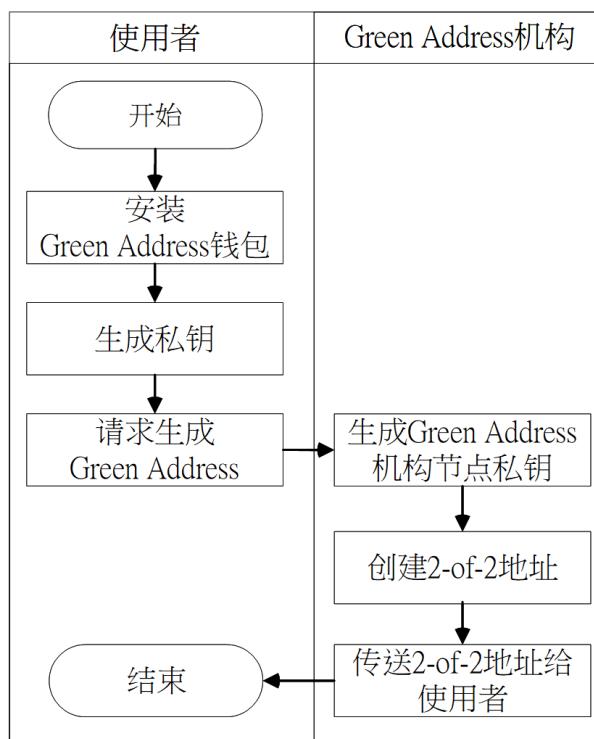


圖 2.4 Green Address 錢包生成流程圖

1. 使用者安裝 Green Address 比特幣錢包，並向 Green Address 機構節點請求創建 2-of-2 多重簽章比特幣地址。
2. 使用者與 Green Address 機構節點分別生成兩把私鑰，共同創建 Green Address 比

特幣地址。

3. 當交易發起時，使用者使用自己的私鑰簽署該筆交易。
4. 將該筆交易傳送到 Green Address 機構節點。
5. Green Address 機構節點收到後，檢查該筆交易是否存在雙重支付攻擊。
6. 確認無攻擊跡象後便廣播至比特幣網絡中。

(二) Green Address 交易發起流程

說明完 Green Address 地址是如何創建之後，本節將詳細說明如何運用多重簽章地址發起交易至比特幣網絡中，如圖2.5所示。

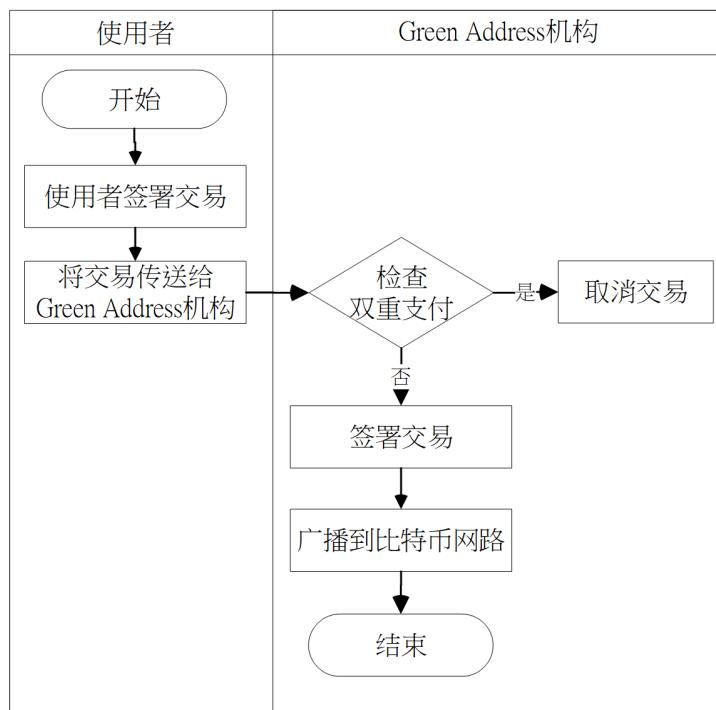


图 2.5 Green Address 交易发起流程圖

1. 使用者使用原本創建 Green Address 的私鑰，並完成簽署交易。
2. 因為是多重簽章地址，所以該交易需傳送至 Green Address 機構節點。
3. Green Address 機構節點收到交易信息後檢查該交易的發起地址是否存在雙重支付，倘若有雙重支付則遺棄；若無雙重支付則往下一個步驟。
4. Green Address 機構以 Green Address 的私鑰簽署該筆交易。
5. 將該筆交易封包廣播至比特幣網絡。

2.3 區塊鏈 (Blockchain)

自 2009 年以來，加密貨幣比特幣的誕生引發了新的貨幣革命浪潮，基於密碼學，點對點網絡，共識算法和區塊鏈技術，它們被結合成比特幣等加密貨幣。到目前為止，它在九年內發生大量的襲擊和欺詐事件後仍然在積極努力。比特幣一直是互聯網上最具代表性的加密貨幣，同時是區塊鏈技術最重要的應用之一，以下我們將描述區塊鏈技術的一些細節。

2.3.1 區塊頭 (Block Header)

比特幣區塊鏈可視為一種專門存儲交易信息的數據庫，該數據庫的結構嚴謹。區塊鏈之所以稱職為鍊是由許多區塊構成，區塊頭存在於區塊中記錄區塊中的重要信息共六項，分別為區塊版本、前區塊的哈希值、Merkle Root、難易度、時間戳以及Nonce，以下將逐一說明：



图 2.6 比特幣區塊鏈結構圖

1. 區塊版本 (32 bits)：該欄位存儲比特幣區塊鏈中的區塊版本。
2. 前區塊的哈希值 (256 bits)：記錄前一個區塊的哈希值。根據當前區塊的前一個區塊哈希值進而形成哈希指針，所有塊可以因為哈希指針連接在一起形成比特幣區塊鏈，不僅可以在區塊與區塊間建立虛擬鏈接，還可以使得區塊更難以被篡改。而通過新區塊不斷疊加在舊區塊過程，舊區塊的哈希值將繼續傳遞到最新的區塊。若區塊上面堆疊更多的區塊，促使的哈希值間接引用越多次，因此較早創建的區塊更難以修改。
3. Merkle Root (256 bits)：Merkle Root 的生成方法是將當前區塊的所有交易為 n 個進行排序後，Merkle Root 為 Merkle Tree 的樹根，交易為樹葉 n 個，將每個樹葉



图 2.7 Merkle Tree 示意图

進行兩次 SHA-256 哈希算法取得哈希值得到 n 個哈希值，再將哈希值兩兩配對合併進行兩次 SHA-256，得到 $n * 2^{-1}$ 個哈希值後，在 k 輪後會使得 $n * 2^{-k} = 1$ 時，合併到只剩下一個哈希值，最後一個哈希值則為 Merkle Root，如圖所示2.7，圖中的 Hash() 函數為雙重 SHA-256，在區塊鏈中的 Merkle Root 可用於快速檢查當前區塊中所有存儲交易的正確性。

4. 難易度 (32 bits): 難易度參數主要調控比特幣挖礦過程中採用工作量證明算法的變量，執得一提的是比特幣的難易度參數為動態調整。在過去的加密貨幣的設計中，有著因為沒有動態修改區塊難度，而導致區塊鏈生成速度太快，甚至導致區塊鏈系統崩潰。
5. 時間戳 (32 bits): 以年、月、日、小時和秒的格式記錄區塊生成時間。
6. Nonce (32 bits): Nonce 記錄著礦工在進行挖礦時，必須要不斷的嘗試 Nonce 參數，直到符合難易度參數，才可以創建一個全新的比特幣區塊。該值為 32 bits，意為著礦工嘗試的組態空間為 2^{32} 個可能性。

2.4 點對點網絡與加密貨幣安全

去中心化的加密貨幣系統給社會和傳統中心化的金融體系，以及政府帶來了很重大的衝擊，Satoshi Nakamoto 建構了一個不需要中央銀行發行貨幣的貨幣系統，在比特幣的貨幣發行上全靠區塊鏈既定的算法。除了貨幣發行，也將交易記錄的帳本以明文的方式存儲在去中心化的區塊鏈中，以比特幣為例，現今完整的比特幣區塊鏈帳本已經高達 180GB，這樣保存完整交易數據的計算機稱之為全節點，在比特幣去中心化的網絡中，如圖2.8所示，截至 2018 年 1 月 25 比特幣網絡中全節點數量為 10552 個^[53]，全節點的數量決定了比特幣帳本的可靠度，倘若有著更多的全節點，會使得比特幣網絡堅不可摧，更難去修改歷史發生過的交易數據。

GLOBAL BITCOIN NODES DISTRIBUTION
Reachable nodes as of Sat Feb 17 2018 15:52:54
GMT+0800 (CST).

11147 NODES

24-hour charts »

Top 10 countries with their respective number of reachable nodes are as follow.

| RANK | COUNTRY | NODES |
|------|--------------------|---------------|
| 1 | United States | 2909 (26.10%) |
| 2 | Germany | 2049 (18.38%) |
| 3 | China | 827 (7.42%) |
| 4 | France | 749 (6.72%) |
| 5 | Netherlands | 502 (4.50%) |
| 6 | Canada | 418 (3.75%) |
| 7 | Russian Federation | 376 (3.37%) |
| 8 | United Kingdom | 362 (3.25%) |
| 9 | n/a | 284 (2.55%) |
| 10 | Singapore | 224 (2.01%) |

More (103) »

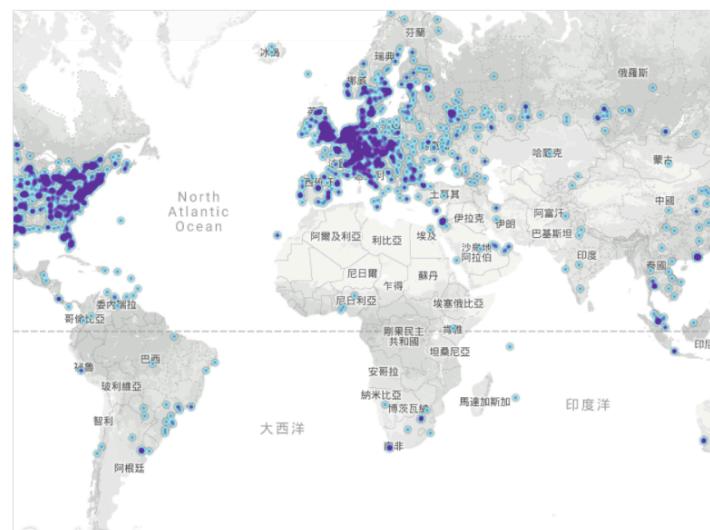


图 2.8 比特幣全節點分佈圖^[53]

第三章 系統需求分析

3.1 功能性需求分析

3.2 交易模型分析

在設計一個區塊鏈的實名交易監督系統之前，必須要針對不同的交易模型做探討。在區塊鏈的實名交易監督系統中，會以加密貨幣的觀點重新設計一個新的支付系統，重新深究匿名者與匿名者之間的交易模式、實名與實名之間的交易模式、匿名與實名之間的交易模式、實名客戶透過實名第三方再實名商家的交易模式以及實名客戶透過匿名第三方再實名商家的交易模式，這五種交易模式所代表著意義與時代革新帶來的技術變革。

3.2.1 現金交易模型

(一) 匿名客戶對匿名商家

最早人類的交易行為可以探究到以物易物的交易行為模式，進而發展出銅幣、紙幣、金幣，甚至是現今常聽聞的金本位制度。在交易的過程中商家無法知道消費者的真實身分，而在一些沒有收據的環境下，如雜貨店或是攤販、一些沒有開收據的商家，消費者也不知道商家的真實身分，故將此交易模式定義為”匿名者與匿名者之間的交易模式”。在這樣的交易模式下，消費者保持匿名，對消費者而言，可以有效的保護消費者的個人信息安全，因為在貨幣的持有方式，並不需要登記姓名，資產轉移的過程中一概不需要。對於消費者而言，雖然消費者的匿名保護了自己的個人信息，但商家的交易信息也是匿名，對整個交易結果若有爭議，這便是追訴無門的結果。而在交易並未被有效記錄的情況下，政府對稅收的計算，會進入無法計算的灰色地帶。圖3.3為匿名的消費者對未開立收據或是實名制的商家的交易模型。

(二) 匿名客戶對實名商家

在另一個場景中，在交易進行的過程中，消費者為匿名，商家為實名，對消費者而言因為自己本身並無綁定個人信息，故對個人信息有很大的保障，消費者消費物件的店傢具有實名而開立收據，對消費者而言會得到消費記錄的保障，對消費的糾紛有店家可以追溯。對政府，因為交易的紀載使得稅收的計算變得容易。圖3.4為匿名消費者使用現金對已經實名制或是開立收據的商家之消費模型。

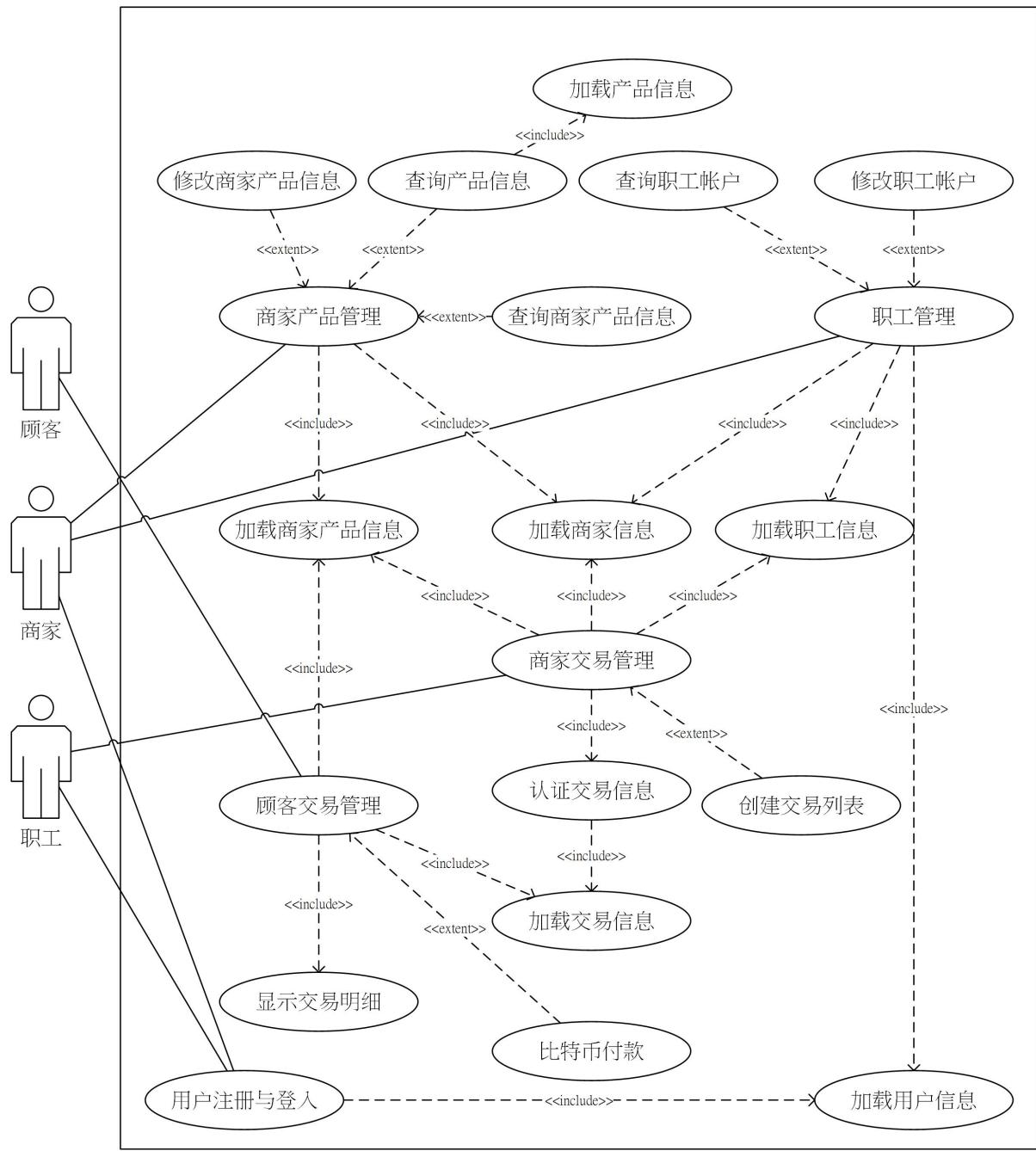


图 3.1 系统用例圖

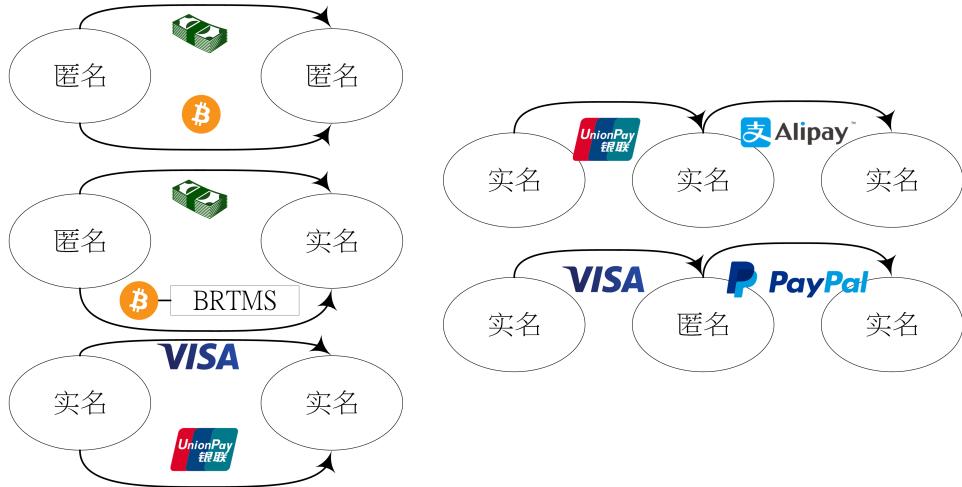


图 3.2 各種交易模型示意圖

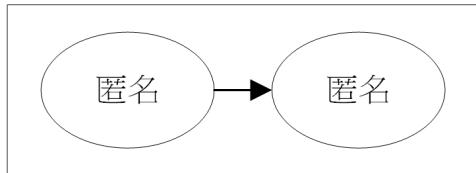


图 3.3 匿名對匿名交易示意圖

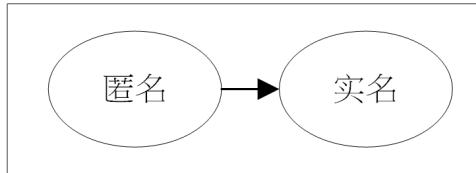


图 3.4 匿名對實名交易示意圖

3.2.2 電子貨幣交易模型

(一) 實名客戶支付實名商家

在現在最為常見的塑膠貨幣支付管道 VISA 中，因為當年的設計並無類似區塊鏈去中心化的理論、技術提出，因而資金的轉移設計會是通過銀行進行帳戶與帳戶之間的資金轉移，也因為這樣的設計，造成交易的基礎被規範在實名與實名之間的交易模式，這樣的交易模式，雖然快速且方便，但在無形之中透漏了許多消費者的個人信息。在交易手續費方面，VISA 的手續在跨國刷卡的場景下，皆需要收取高達百分之一點五的手續費，對於消費者而言使用 VISA 作為支付會帶來不小的負擔。圖3.5為透過實名制支付管道對商家進行交易的模型。

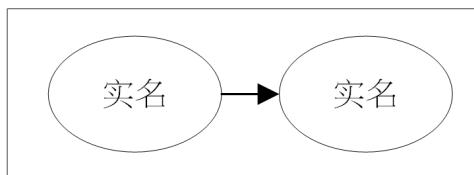


图 3.5 實名對實名交易示意圖

(二) 實名客戶透過實名第三方再實名商家

在中國 VISA 較不為常見，但除了 VISA 電子支付還有中國本身自營的銀聯，但因為中國的銀行眾多林立，且在科技化的世代中隨身帶著許多的卡片會造成不便，所以支付寶致力於將所有的卡片電子化，將所有中國在地銀行卡統合在一起，透過結合所有品牌的銀行卡以達有效提升卡片交易的方便性與使用率，圖3.6為以實名制的銀聯卡透過支付寶進行支付給實名制的店家的交易模型。

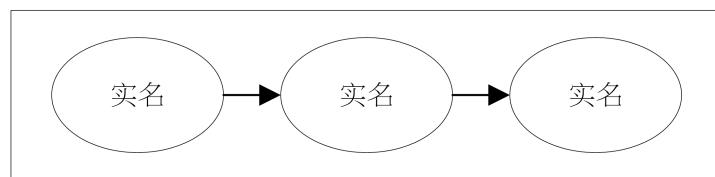


图 3.6 實名對實名再對實名交易示意圖

(三) 實名客戶透過匿名第三方再實名商家

為減低在進行交易的過程中使用 VISA 支付管道會透露太多的消費者個人信息的問題，PayPal 便致力於將消費者銀行卡的相關個人信息存儲在 PayPal 身上，PayPal 再以公司的身分，將資金轉移給商家，消費者與店家的交易中間多了一個仲介的角色，也讓這樣的交易模式看似匿名的消費者對上實名的商家。但在這樣的交易過程中，消費

者的信任需要寄附在 PayPal 身上，畢竟大部分的銀行卡與個人信息皆存儲在 PayPal 公司內，PayPal 公司的信息安全將成為最重要的議題。圖3.7為先以實名制的支付管道將資金轉移到代支付的 PayPal 公司，PayPal 代為支付的過程中將原資金來源的個人相關信息保護在 PayPal 公司中，以製造出一種匿名支付方法保護消費者的個人信息之模型。

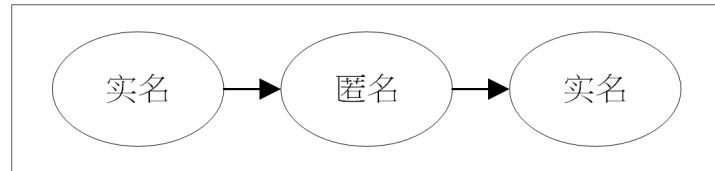


图 3.7 實名對匿名再對實名交易示意圖

3.2.3 交易關係比較

綜合上述五種交易方式，我們可以統整出一張交易關係比較表，如表3.1。我們可以發現除了部分數位元貨幣以及用現金與未開收據的店家做交易，這兩種方式最能保障消費者，但後者會讓商家成為匿名交易者，否則現今絕大多數的交易商家皆為實名制，即是為了保障消費者的權益，然而目前最廣為人知的數位貨幣是比特幣，在它的區塊鏈上只能查看透過雜湊處理所得的地址，無法得知交易雙方的真實資訊，倘若比特幣被用來執行非法交易、或是交易產生爭議，都無法輕易認定區塊鏈上的交易與現實生活中的交易是有關聯的。因此本系統致力於將比特幣從匿名對匿名的交易，強化成匿名對實名的交易，一方面便於政府監督社會上的金流，另一方面也可讓使用者在以比特幣交易時更有保障。

表 3.1 交易關係比較表

| | 顧客 | 仲介單位 | 商家 | 商品 |
|--------|----|------|-------|-------|
| 現金 | 匿名 | 無 | 匿名/實名 | 匿名/實名 |
| VISA | 實名 | 無 | 實名 | 實名 |
| 支付寶 | 實名 | 實名 | 實名 | 實名 |
| PayPal | 實名 | 匿名 | 實名 | 實名 |
| 加密貨幣 | 匿名 | 無 | 匿名/實名 | 匿名/實名 |

3.3 特性要因分析

透過特性要因分析可以將区块链的实名交易监督系统大致分為四個主題，如圖3.8所示，分別為信息安全技術、加密貨幣錢包、近場通訊技術以及數據庫。針對四項主軸，最為一個金流系統，信息安全是不可或缺的環節，著重於商家認證機制、用戶權力控

管、身份識別管理、使用者訪問控制四個方向；本系統致力於奠定匿名對實名的加密貨幣系統，必須對區塊鏈技術、公鑰私鑰生成算法、點對點交易技術、錢包地址產出以及貨幣發行技術五個方向進行探討；在交易場景中，本系統採用近場通信技術，因此需要對商品 RFID 標籤建置、讀取商品 RFID 標籤以及 Android Beam 傳輸商品交易進行基礎的 API 調研；為了使的加密貨幣實名制的實現，數據庫必須存儲與政府和商家相關的信息。此時數據庫加密、個資去識別化安全以及數據庫連接便相當重要。

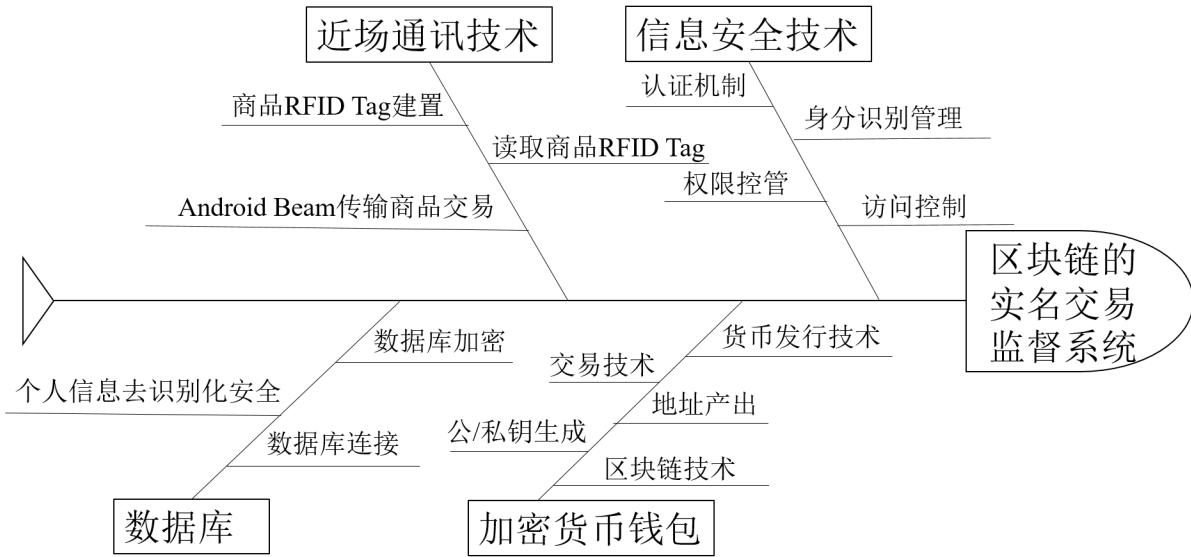


图 3.8 魁骨圖（1）

第四章 系統概要設計

本論文提出區塊鏈的實名交易監督系統（Blockchain Real-name Transaction Monitoring System, BRTMS），以下簡稱 BRTMS，BRTMS 以加密貨幣比特幣實作，BRTMS 包含三個子系統，商家和商品信息管理子系統（Store and Merchandise Information Management Sub-System, SMIMSS）、商家移動裝置收款及交易子系統（Store Mobile payment Collection and Transaction Sub-System, SMCTSS）、客戶端移動支付和交易子系統（Client Mobile Payment and Transaction Sub-System, CMPTSS），圖4.1為主系統與子系統的功能模塊圖。

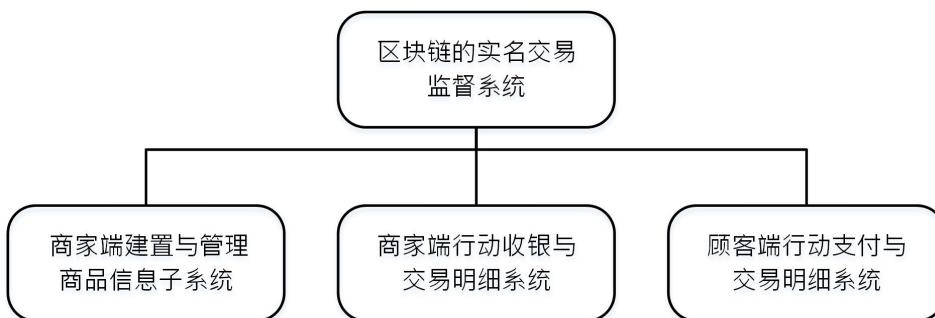


图 4.1 BRTMS 系統功能模塊圖

文將開發讓商家能夠結合商品與 RFID 標籤，以達到快速建構與管理商品資料庫之系統，並且讓商家及顧客可以運用手機 NFC 功能來實際運作比特幣的行動支付流程。店家只需要掃描商品上的 RFID 標籤，即可快速建立交易清單，再利用 NFC 功能與顧客之行動裝置進行訊息交換，輕鬆地將商家的收款地址以及交易資料傳送給顧客，收到資料後便能快速地以顧客之比特幣行動電子錢包付款，並將交易細項儲存下來，以便未來商家與顧客能夠快速查詢比特幣行動支付的交易紀錄。本系統主要是以完成區塊鏈之數位加密貨幣的收款監督系統為主要目標，本論文進而將積極利用自由軟體的利基：使用成本低、進入門檻低、開放原始碼、社群能力強、共通性及移植性強、資通安全性高等優勢來開發本論文收款監督系統的應用服務平台。本系統範圍包含建置下各項子系統如下：

1. 商家端建置與管理商品資訊子系統 (Store and Merchandise Information Management Sub-System, SMIMSS): 本系統可以讓商家在進貨時，快速地將 RFID 標籤之識別碼與進貨商品資訊集成在一起，並且透過本系統新增、修改或刪除資料庫內部的資訊，包括產品名稱、詳細資訊，存貨數量等資訊，商家與顧客便可依照該資

料庫取得當前商品資訊與狀態。不僅讓商家的存貨資訊更加清楚明瞭，也可以提供顧客更多的即時服務，圖4.2所示。

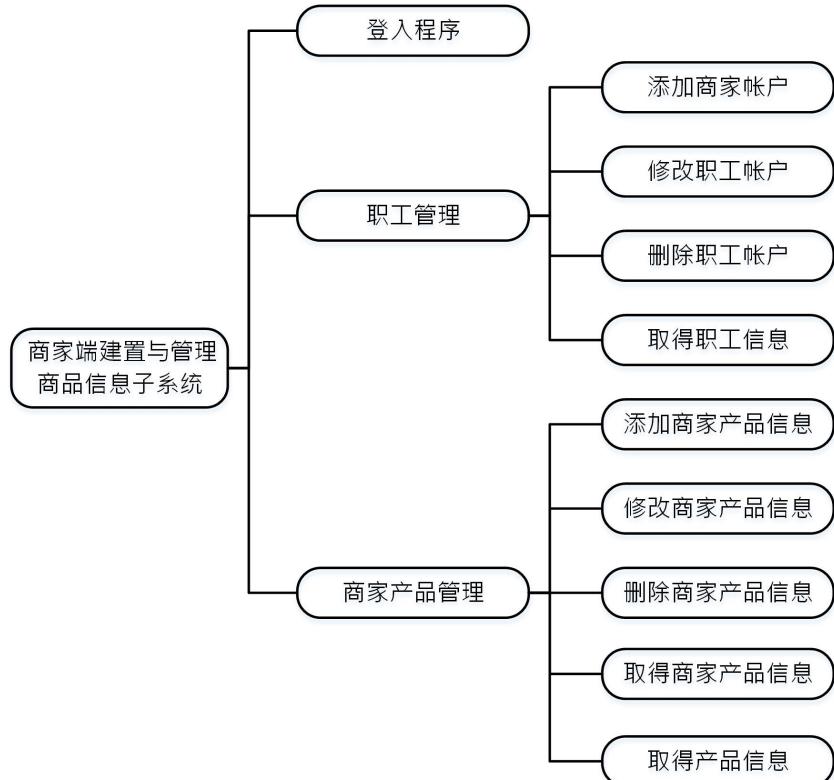


图 4.2 商家端建置與管理商品資訊子系統功能模塊圖

2. 商家端行動收銀與交易明細系統 (Store Mobile payment Collection and Transaction Sub-System, SMCTSS): 本系統使商家在結帳時，能夠以手機 NFC 功能掃描商品上的 RFID 標籤，即可簡單地建立交易清單，並透過 NFC 與顧客手機碰觸，將交易清單以及商家之比特幣收款地址等等重要交易資訊一併傳遞給顧客，可以簡短結帳的速度，使結帳效率大幅提升，圖4.3所示。
3. 顧客端行動支付與交易明細系統 (Client Mobile Payment and Transaction Sub-System, CMPTSS): 顧客在結帳時，不必再麻煩的拿出信用卡或是零錢包，只需要拿出手機讓職工以 NFC 將交易清單與比特幣地址轉送給自己，即可自動連結至比特幣電子錢包的應用程式當中，並且自動填妥相關資料，如: 交易金額、收款地址等等與此同時也能將交易紀錄儲存於客戶端，以便日後顧客快速取得過往的交易紀錄，除此之外亦可讓廣大的民眾體驗數位加密貨幣與行動支付帶來的便利生活，圖4.4所示。



图 4.3 商家端行動收銀與交易明細系統功能模塊圖

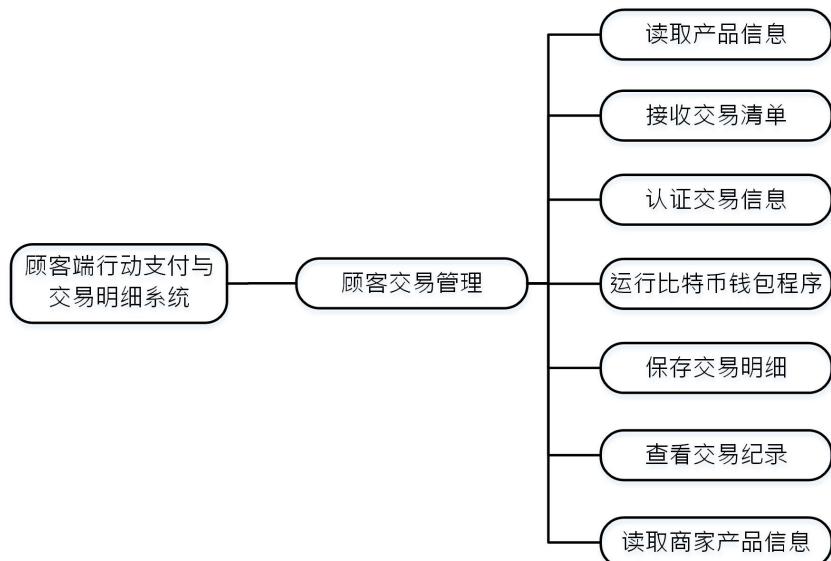


图 4.4 顧客端行動支付與交易明細系統功能模塊圖

4.0.1 商家和商品信息管理子系統 (SMIMSS)

圖4.5為區塊鏈的實名交易監督系統架構，商家需要在以下 4 個步驟中對商家和商品信息管理子系統（SMIMSS）進行註冊：

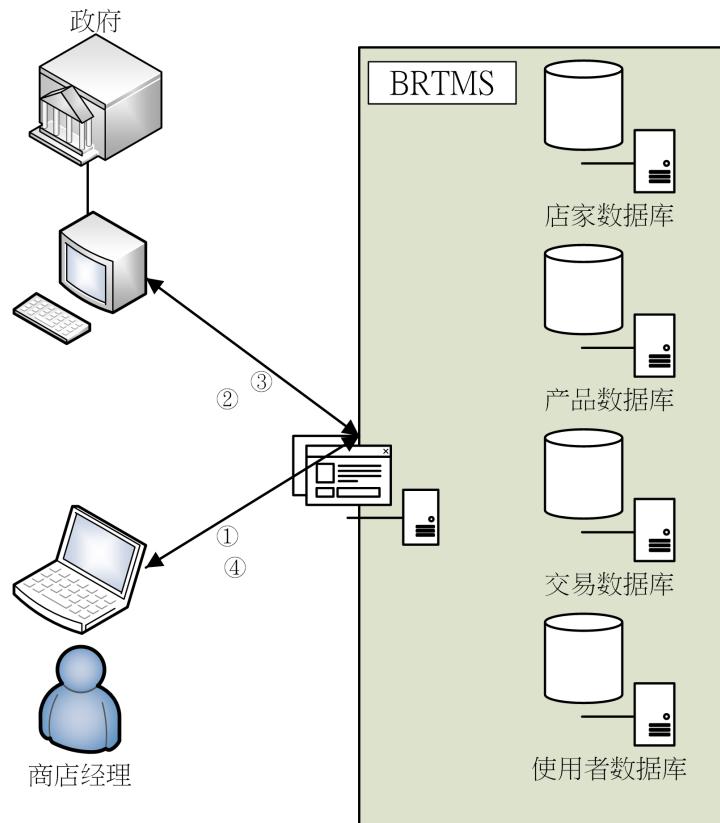


图 4.5 BRTMS 和商家註冊流程的核心架構圖

1. 商家必須在 BRTMS 註冊一個帳戶，並附有政府法規的商業證明。
2. 區塊鏈的實名交易監督系統將自動向相應的政府金融監管機構提交商業申請，以審查該商家的加密貨幣交易業務。
3. 如果政府批准商家的加密貨幣業務申請，服務器將激活商家在該收集監控系統中創建商家帳戶。
4. 商家可以自由地登錄賬戶並添加商家想要出售的產品信息，亦可透過職工管理添加修改刪除職工信息。

4.0.2 BRTMS 架構與運作流程

具體的加密貨幣商家收銀金流監控系統運行過程如圖4.6所示，我們以圖4.5所設計出的系統架構進行擴增，首先我們需要與區塊鏈檢視器 (blockchain explorer) 對接，而之所以該監控系統需要與區塊鏈檢視器對接是為了能夠最直接的比對交易被記錄的成

交狀況，能夠達到即時性以及正確性，倘若擔憂單方面的依賴相信區塊鏈檢視器的成果，亦可以使用多家區塊鏈檢視器進行交叉參考，以避免因為一家公司的錯誤所帶來的影響。區塊鏈檢視器的目的是為了能夠快速地準確地比對該筆交易的成交，做為整筆交易提出到結算的環節之一。

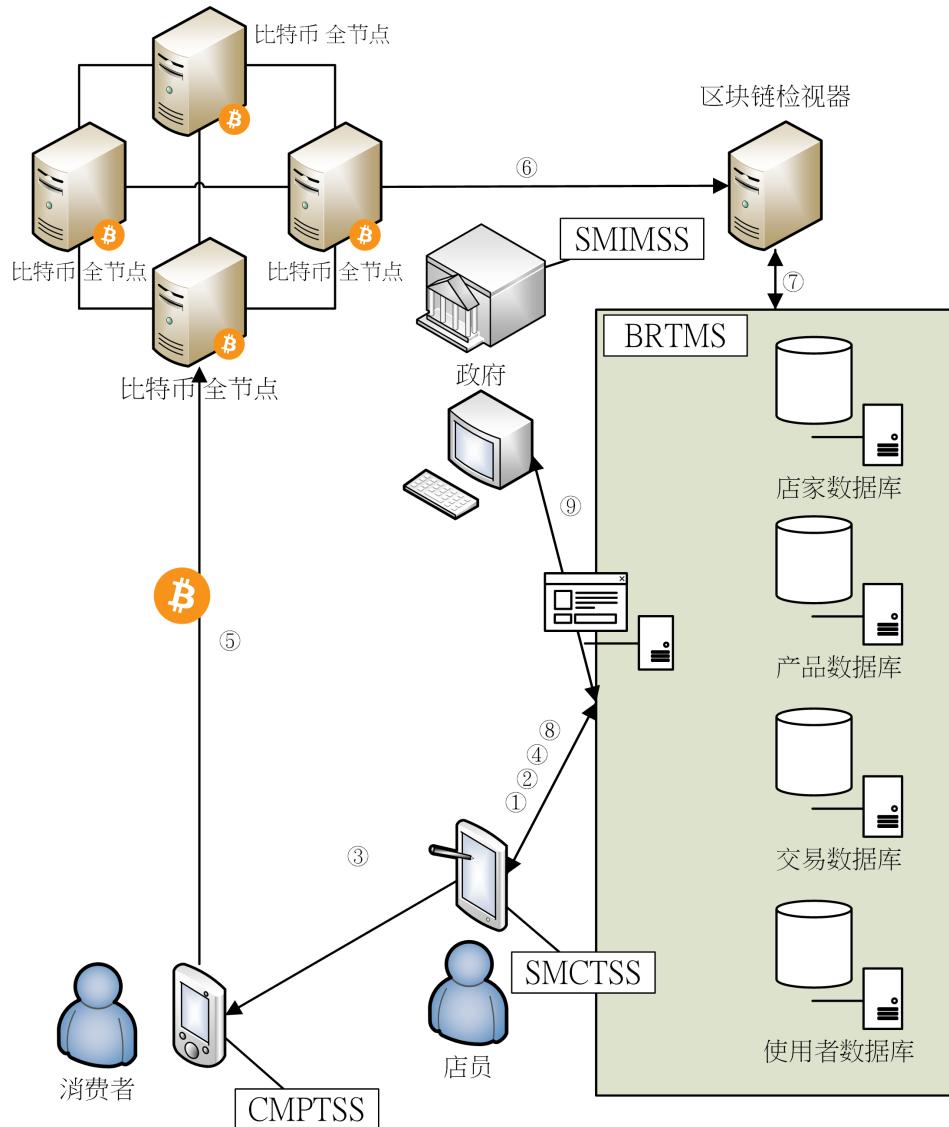


图 4.6 BRTMS 的整体架构与功能示意图

圖4.6區塊鏈的實名交易監督系統創建步驟描述如下：

1. 商家職工將登錄到圖4.5中的第一個步驟，所示的先前步驟創建的帳戶，使用手持式平板電腦或智能手機訪問 SMCTSS 中的服務。如前所述，在能夠登錄到系統之前，商家帳戶必須由政府機構審計。
2. 在成功登錄 SMCTSS 用於商家加密貨幣流量監控系統時，移動設備將加載通過 SMIMSS 註冊的商家產品信息，然後創建產品目錄。商家的職工可以根據客戶的

需求選擇所需的產品和數量。

3. 職工使用設備完成客戶選定商品的產品信息後，移動設備上的 NFC 技術可用於將產品信息從附近職工的移動設備傳遞給消費者的移動設備，而無需物理交互。然後，消費者可以很容易地將自己的消費信息記錄成為發票等參考。在接收從商家職工設備向顧客設備購買產品消息的後，顧客設備還將向商家的移動設備發送其自己的比特幣支付地址的信息。
4. 商家的手持設備收到客戶確認購買所選產品的相應信息後，會將交易信息的副本發送給 SMIMSS 監控系統。消費者信息包括交易序列號、商家 ID 號碼、商品號碼、購買的商品的數量以及加密貨幣的收款人地址以及消費者的支付地址。
5. 收到消費者交易信息後，即完成此次的加密貨幣支付。同時，此次交易的加密貨幣將發佈到比特幣網絡中進行驗證和記錄。
6. 區塊鏈瀏覽器將開始分析在比特幣網絡中緩存池的所有交易以及區塊鏈中記錄的交易。
7. 擬議的交易監控系統 BRTMS 將向區塊鏈檢視器提出請求。這個請求數據不僅包括存儲在 BRTMS 中的交易副本之加密貨幣收款人地址（如圖4.5的第 4 步所述），還包括客戶預期付款的加密貨幣支付地址。區塊鏈瀏覽器使用請求數據來檢查交易是否存儲在區塊鏈中，或者交易還在等待確認。如果交易已被確認並存儲在區塊鏈中，則交易數據庫中“交易確認”字段的值將更改為“1”，否則其默認值為“0”。
8. 當“交易確認”字段中的值為“1”時，“交易已完成”消息可發送至商家平板電腦上運行的商家和商品信息管理子系統（SMCTSS）。
9. 政府財政監督部門可以審查擬議 BRTMS 中的所有交易信息，以作為稅收審計參考。

4.0.3 實時 BRTMS 架構與運作流程

在這樣的機制下，雖然 Green Address 沒有減少交易確認時間，但是只要是用 Green Address 即可確保雙重支付攻擊是不會發生的，對商家或是收款人而言，可以得到在即時交易中不被雙重支付攻擊的保障，提升未進入區塊鏈的交易可信度，進而創造出即時交易的可行性。

本節將詳述本系統之商家註冊、Green Address 錢包創建與驗證交易的運作流程與相關數據庫架構，如圖4.7所示。

1. 商家以通過政府機構的審查稽覈的帳戶登入該系統。
2. 系統載入該店家註冊的商品信息，職工可以依照客戶的需求進行點單選取數量。
3. 快速建立交易清單，並透過 Green Address 建立一個全新的比特幣收款地址，再

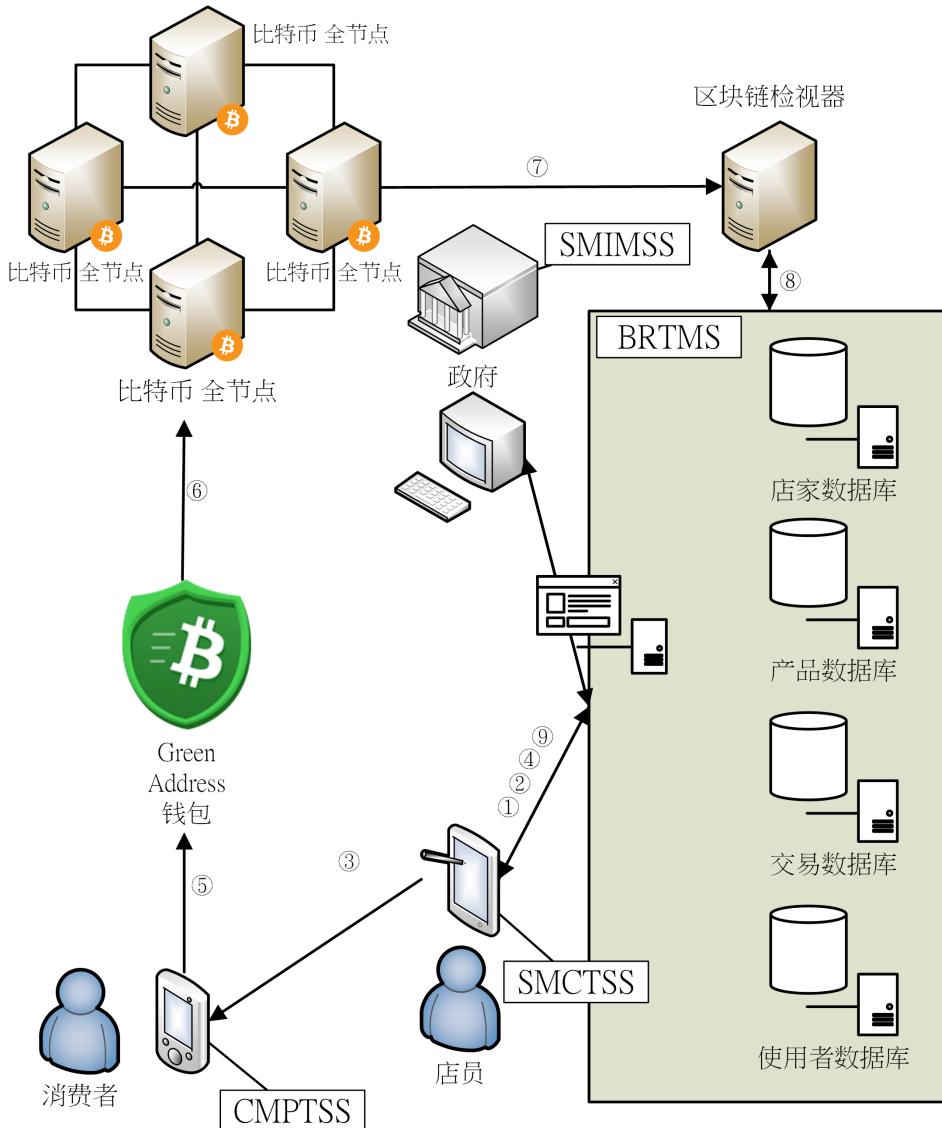


图 4.7 多重簽章優化後的 BRTMS 整體示意圖

以 Android Beam 的方式將交易信息輕鬆地傳達給消費者。

4. 在商家職工的平板電腦收到這筆交易信息之後，會對本監督系統重送一個副本進行存檔。該交易信息包括由監督系統所提出的交易流水號、商家編號等信息。
5. 消費者收到交易信息後，手機會自動開啟 Green Address 的付款頁面，確認金額無誤之後便能進行支付，此時便會以客戶的比特幣私鑰簽署交易，並等待 Green Address 機構節點的認證及發布。
6. Green Address 機構節點收到交易請求，並完成驗證非雙重支付攻擊後，以代理節點對應地址的私鑰簽署本次交易，並廣播至比特幣節點中。
7. 區塊鏈檢視器便會開始分析網絡中所有存在緩存池中的交易，以及已經被記錄到區塊鏈中的交易。
8. 本交易監督系統會向區塊鏈檢視器查找，檢查該筆交易是否已經存在於緩存池當中，若已經確認進入緩存池，則認定該筆交易成立並完成付款。
9. 在交易確認之後，便向商家職工的平板電腦送出交易已經成交的信息，此時完成交易，於此同時也將該筆交易信息建置於系統數據庫內。

第五章 系統詳細設計與實現

5.1 區塊鏈的實名交易監督系統設計

5.1.1 BRTMS 數據庫設計

區塊鏈的實名交易監督系統應用了六個信息表，分別為商家信息、產品信息、交易信息、用戶信息、職工信息以及商家產品信息，以下將逐一說明：

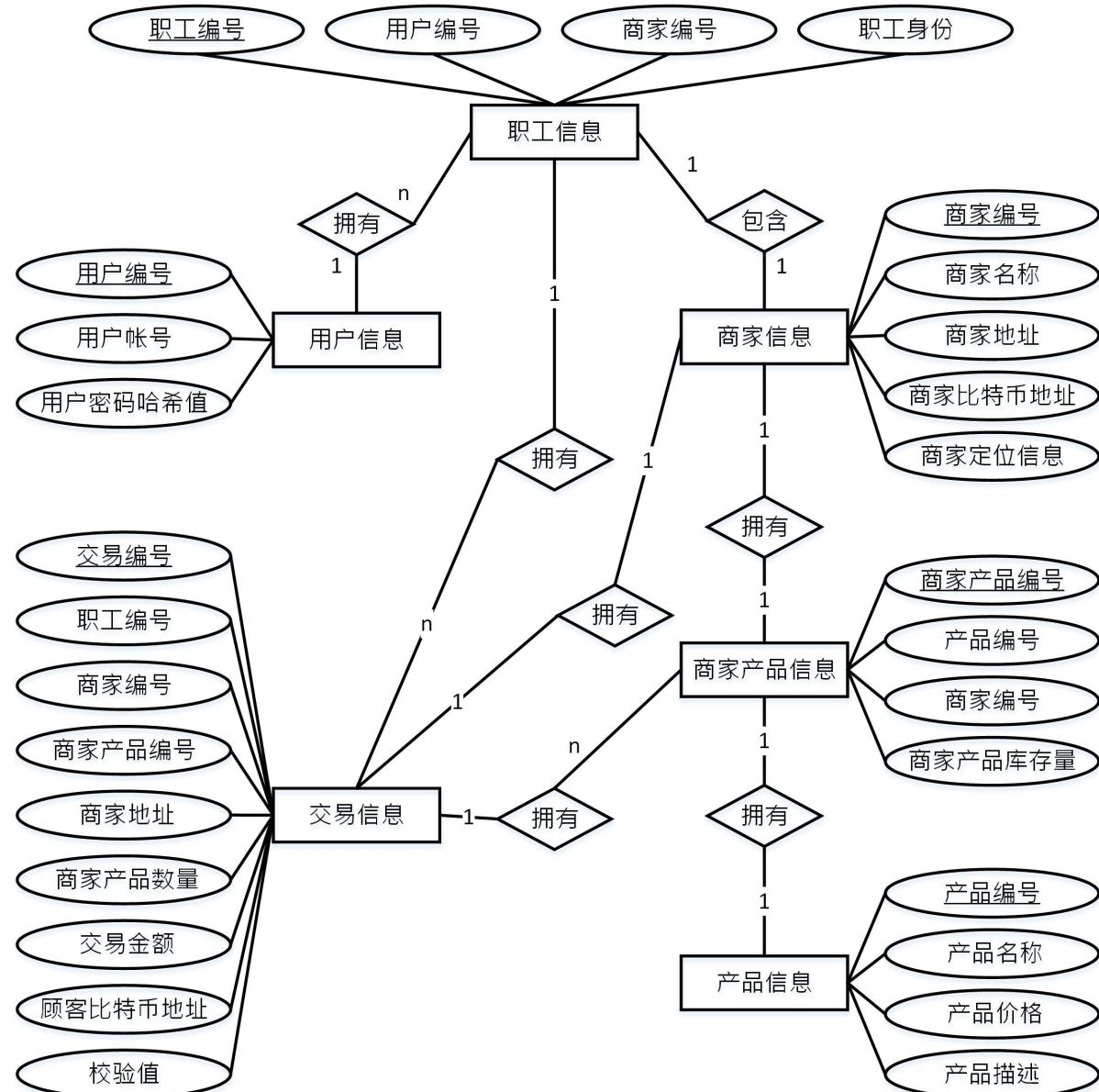


图 5.1 BRTMS 數據庫實體關係圖

- 商家信息表：存储正在审核中的企业信息或已经过审核的企业信息。表5.1存储的信息包括商家编号、商家名称、商家地址、商家比特币地址以及商家定位信息。

表 5.1 商家信息表

| 序號 | 字段名 | 字段說明 | 類型 | 是否為空 | 主外鍵 |
|----|-------------------|---------|---------------|------|-----|
| 1 | STORE_ID | 商家編號 | int | 否 | PK |
| 2 | STORE_NAME | 商家名稱 | navarchar(20) | 否 | |
| 3 | STORE_ADDRESS | 商家地址 | navarchar(50) | 否 | |
| 4 | STORE_BTCAADDRESS | 商家比特幣地址 | navarchar(50) | 否 | |
| 5 | STORE_GPS | 商家定位信息 | navarchar(30) | 否 | |

- 产品信息表：只有授权用户才能登录添加或修改交易产品信息。表5.2产品信息表内容包括产品编号、产品名称、产品价格以及产品描述信息。

表 5.2 产品信息表

| 序號 | 字段名 | 字段說明 | 類型 | 是否為空 | 主外鍵 |
|----|---------------------|------|---------------|------|-----|
| 1 | PRODUCT_ID | 產品編號 | int | 否 | PK |
| 2 | PRODUCT_NAME | 產品名稱 | navarchar(10) | 否 | |
| 3 | PRODUCT_PRICE | 產品價格 | float | 否 | |
| 4 | PRODUCT_DESCRIPTION | 產品描述 | navarchar(50) | 是 | |

- 交易信息表：表5.3记录包括交易编号、职工编号、商家编号、商家产品编号、商家地址、商家产品数量、交易金额、顾客比特币地址和最后确认字段的值。

表 5.3 交易信息表

| 序號 | 字段名 | 字段說明 | 類型 | 是否為空 | 主外鍵 |
|----|-----------------------|---------|---------------|------|-----|
| 1 | TX_ID | 交易編號 | int | 否 | PK |
| 2 | STAFF_ID | 職工編號 | int | 否 | FK |
| 3 | STORE_ID | 商家編號 | int | 否 | FK |
| 4 | STOREPRODUCT_ID | 商家產品編號 | int | 否 | FK |
| 5 | STORE_ADDRESS | 商家地址 | navarchar(50) | 否 | FK |
| 6 | STOREPRODUCT_QUANTITY | 商家產品數量 | int | 否 | |
| 7 | TX_AMOUNT | 交易金額 | float | 否 | |
| 8 | CONSUMER_BTCAADDRESS | 顧客比特幣地址 | navarchar(50) | 否 | |
| 9 | CHEK | 校驗值 | bool | 否 | |

- 用户信息表：表5.4存储所有使用者信息，包括政府、商家及顾客之个人的账户编号与账号，而使用者密码则以哈希值的方式保存以增加用户安全性。
- 职工信息表：表5.5信息表存储各个商家拥有的职工信息，包括各职工编号、用户编号商家编号及职工身份。
- 商家产品信息表：表5.6存储各家商家当前商家产品存货信息，由商家产品编号、产品编号、商家编号及商家产品库存量所组成。

表 5.4 用戶信息表

| 序號 | 字段名 | 字段說明 | 類型 | 是否為空 | 主外鍵 |
|----|-------------------|---------|--------------|------|-----|
| 1 | USER_ID | 用戶編號 | int | 否 | PK |
| 2 | USER_ACCOUNT | 用戶帳號 | nvarchar(30) | 否 | |
| 3 | USER_PASSWORDHASH | 用戶密碼哈希值 | nvarchar(30) | 否 | |
| 4 | GOVT_AUTH | 政府審查 | bool | 否 | |

表 5.5 職工信息表

| 序號 | 字段名 | 字段說明 | 類型 | 是否為空 | 主外鍵 |
|----|--------------|------|--------------|------|-----|
| 1 | STAFF_ID | 職工編號 | int | 否 | PK |
| 2 | USER_ID | 用戶編號 | int | 否 | FK |
| 3 | STORE_ID | 商家編號 | int | 否 | FK |
| 4 | STAFF_STATUS | 職工身份 | nvarchar(20) | 否 | |

5.2 系統模塊設計

在本系統中共有三種使用者，分別為顧客、商家以及職工，首先是用戶登入與註冊模塊，其餘總共有四個管理模塊分別為商家產品管理模塊、職工管理模塊、顧客交易管理模塊和商家交易管理模塊，以下將說明各個模塊類圖設計以及時序圖運作。

(一) 用戶註冊與登入模塊

在本系統中僅職工與商家需要進行註冊，並且需要經過政府的審查批准。職工與商家皆為使用者皆可使用用戶註冊模塊。

圖5.3為職工與商家註冊時序圖，以下為流程說明：

- 首先職工或是商家前往用戶註冊頁面。
- 前往用戶信息填寫頁面填寫用戶帳戶信息以及用戶密碼。
- 完成填寫後，為了提升使用者密碼信息安全，將使用者密碼以哈希算法計算後以密碼哈希值保存。將填寫完的帳號信息以及密碼哈希值提交到用戶信息表。
- 用戶信息表則傳為添加成功。屆時的用戶信息已經存儲到用戶信息表內但是尚未被激活，需要等待政府進行審查。
- 政府向用戶信息請求表拿取所有還未被激活的用戶信息。

表 5.6 商家產品信息表

| 序號 | 字段名 | 字段說明 | 類型 | 是否為空 | 主外鍵 |
|----|------------------------|---------|-----|------|-----|
| 1 | STOREPRODUCT_ID | 商家產品編號 | int | 否 | PK |
| 2 | PRODUCT_ID | 產品編號 | int | 否 | FK |
| 3 | STORE_ID | 商家編號 | int | 否 | FK |
| 4 | STOREPRODUCT_INVENTORY | 商家產品庫存量 | int | 否 | |

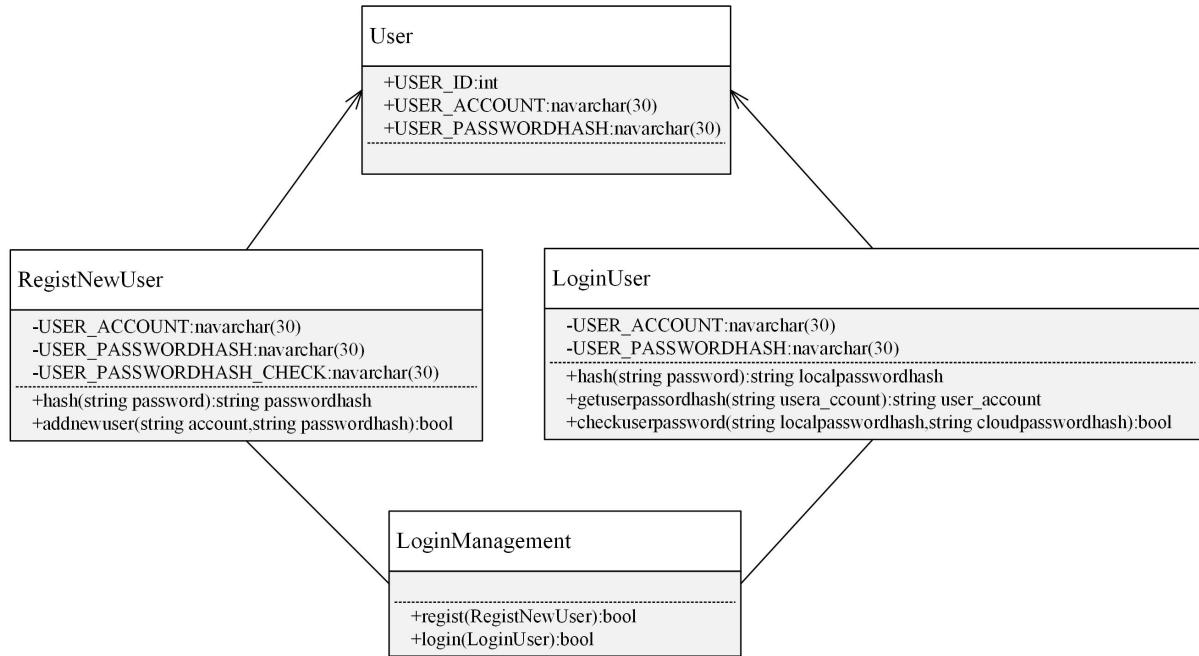


图 5.2 用 户 註 冊 與 登 入 模 塊 類 圖

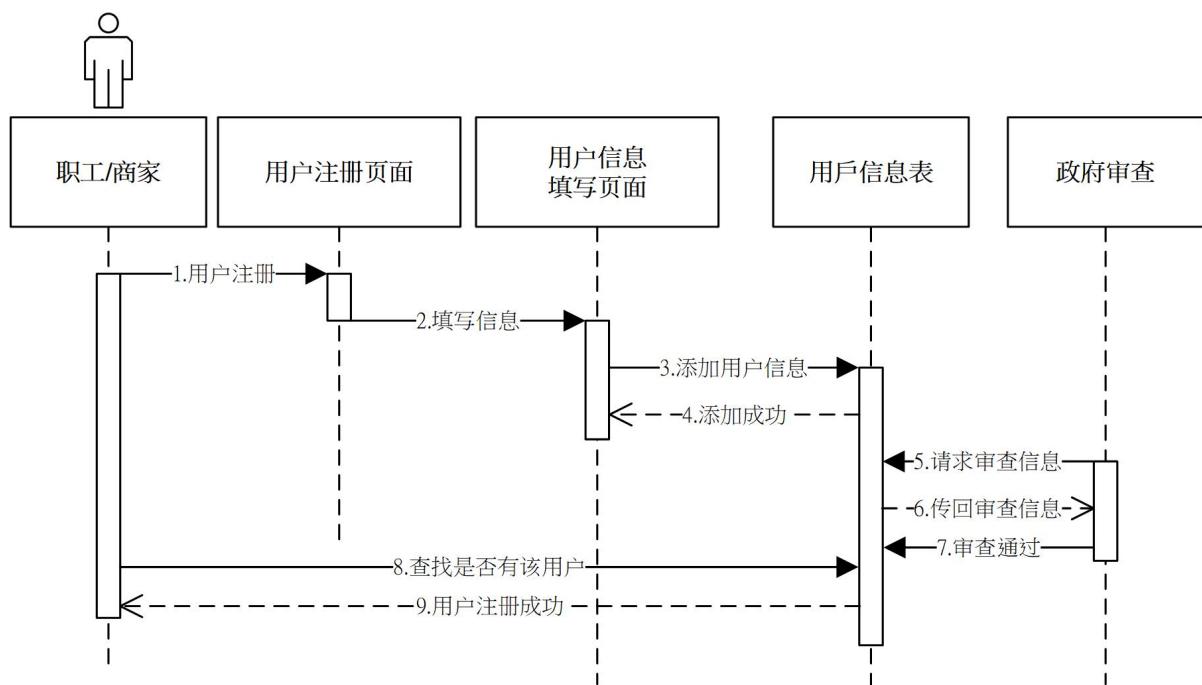


图 5.3 职 工 與 商 家 註 冊 時 序 圖

6. 數據庫中的用戶信息表傳回，政府請求的用戶信息清單。
7. 政府進行用戶審查，在完成用戶審查後，則向用戶信息表內輸入審查通過。
8. 此時，用戶再次訪問用戶信息表當中，用戶帳號是否已經激活成功。
9. 用戶信息表回傳已經驗證通過。

(二) 產品管理模塊

商家的運營需要管理本身所需販售的商品，在商家完成註冊用戶帳號並且通過政府審核之後，便可檢視所有的產品，近一步可以透過產品管理模塊新增、修改以及刪除商家商品。

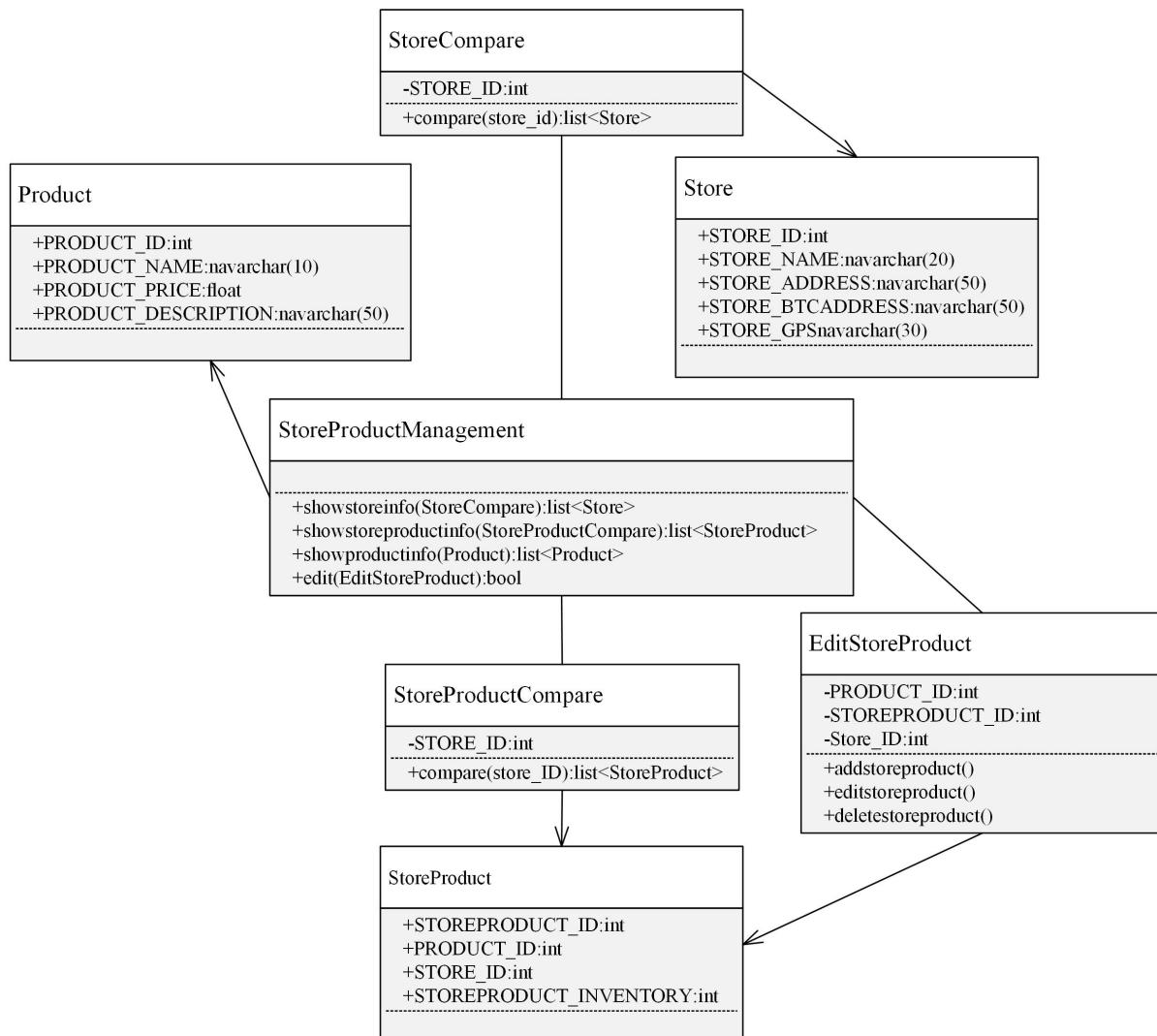


图 5.4 商家產品管理類圖

圖5.5為商家產品管理時序圖，以下為流程說明：

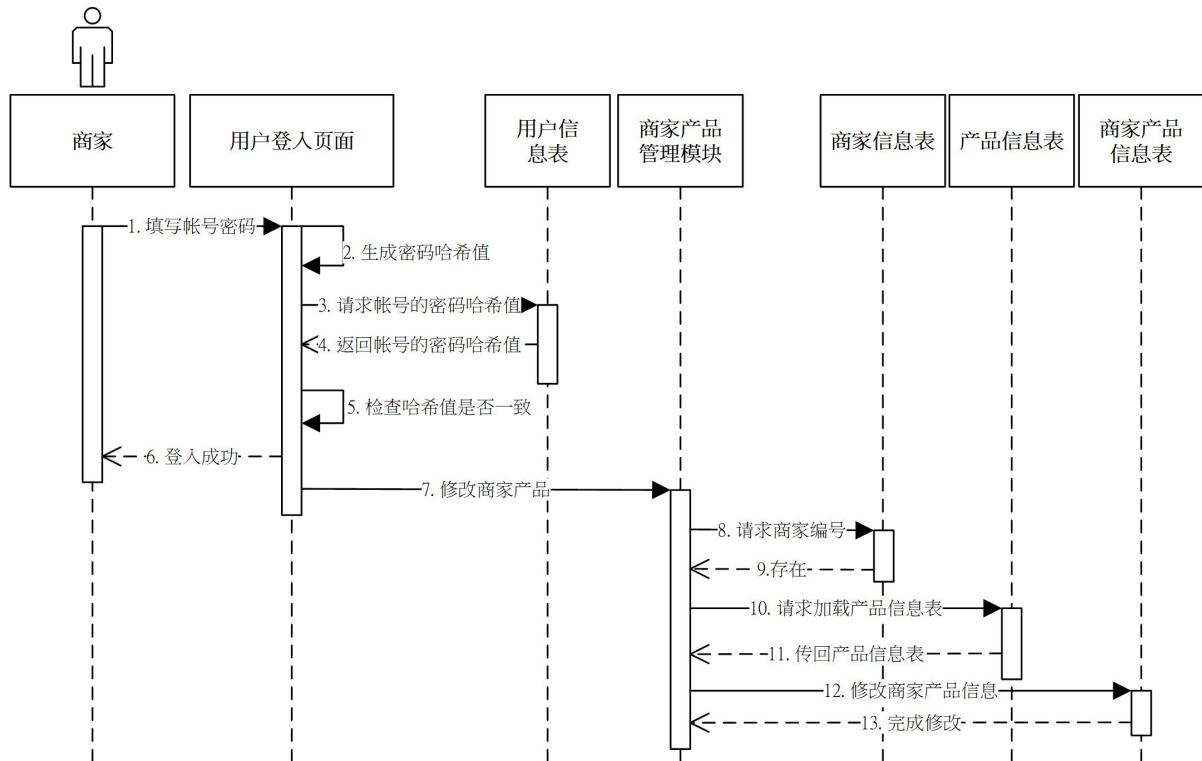


图 5.5 商家產品管理時序圖

1. 商家在完成用戶註冊後並完成政府的審核之後，便可填寫使用用戶與用戶密碼至到登入頁面。
2. 登入頁面會將用戶提交之密碼透過哈希算法生成密碼哈希值。
3. 用戶登入頁面向用戶信息表詢問該用戶帳號的密碼哈希值。
4. 用戶信息表將該帳戶的密碼哈希值傳回給用戶登入頁面保存。
5. 用戶頁面將本地密碼哈希值和從數據庫信息表中請求保存的密碼哈希值進行比對是否相同。
6. 倘若本地與數據庫中的哈希值一至，則為都入成功。
7. 屆時可以進入商家產品管理模塊。
8. 為確認預修改的商家是否存在，所以向數據庫中的商家信息表請求該商家編號是否存在。
9. 商家信息表返回存在的信息。
10. 向數據庫中的產品信息表當中請求所有的產品信息。
11. 產品信息表回傳所有的產品信息。
12. 此時商家已經確認商家信息是否存在，且已經取得所有的產品信息。商家向商家產品信息表提交商家要增加的產品編號以及商家本身的商家編號，此時生成商家產品編號。

13. 商家信息表在完成添加商家產品信息之後，傳回添加成功的信息到商家產品管理模塊。

(三) 職工管理模塊

商家需要職工進行商家的運營，在家用戶完成政府審查之後，便可以進入職工管理模塊，透過提交用員編號以及商家編號新增、修改以及刪除職工信息。

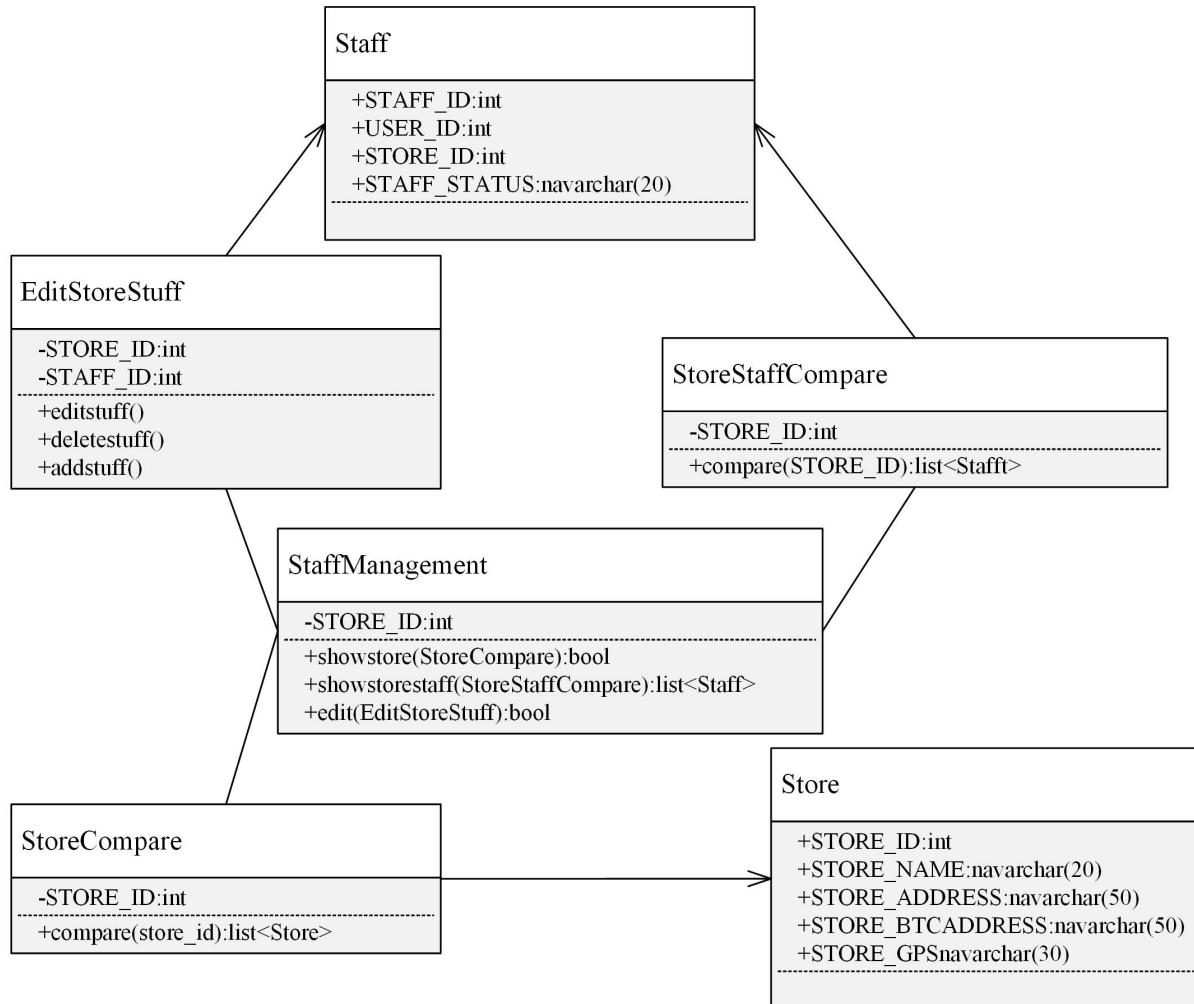


图 5.6 職工管理模塊類圖

圖5.7為商家職工管理時序圖，以下為流程說明：

1. 商家前往用員登入頁面輸入用員帳號密碼。
2. 用員登入頁面自動使用哈希算法將用員密碼轉換成密碼哈希值。
3. 用員登入頁面向數據庫中的用員信息表請求該用員帳號的密碼哈希值。
4. 數據庫用員信息表回傳用員密碼哈希值。
5. 用員登入頁面比對本地端的用員密碼哈希值與數據庫中的密碼哈希值是否一致。

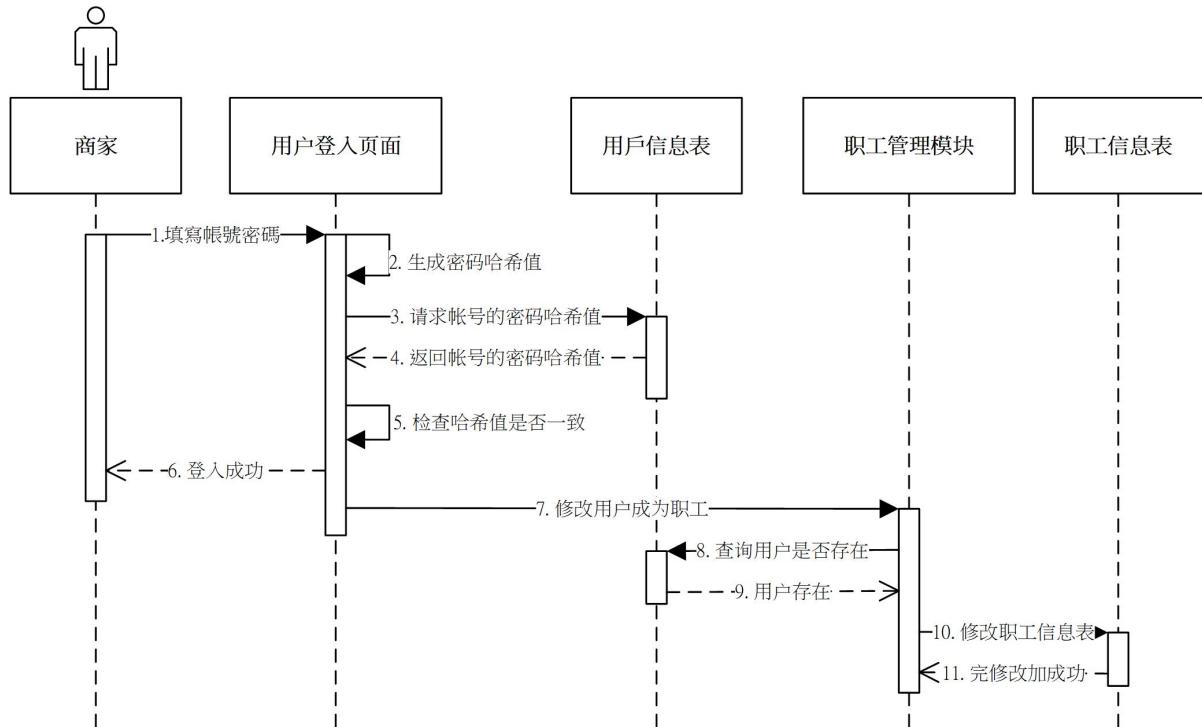


图 5.7 商家职工管理时序图

6. 倘若數據庫中的密碼哈希值與用戶登入頁面生成的密碼哈希值一致，則登入成功。
7. 商家提交商家編號以及預添加的用戶編號。
8. 職工管理模塊向用戶信息表查詢該用戶是否存在。
9. 用戶信息表傳回用戶存在信息。
10. 職工管理模塊向數據庫中的職工信息表提交用戶編號、商家編號以及職工編號修改職工信息。
11. 職工信息表傳回修改完成。

(四) 顧客交易管理模塊

圖5.9為顧客交易管理時序圖，以下為流程說明：

1.

(五) 商家交易管理模塊

圖5.12

流程說明：

1.

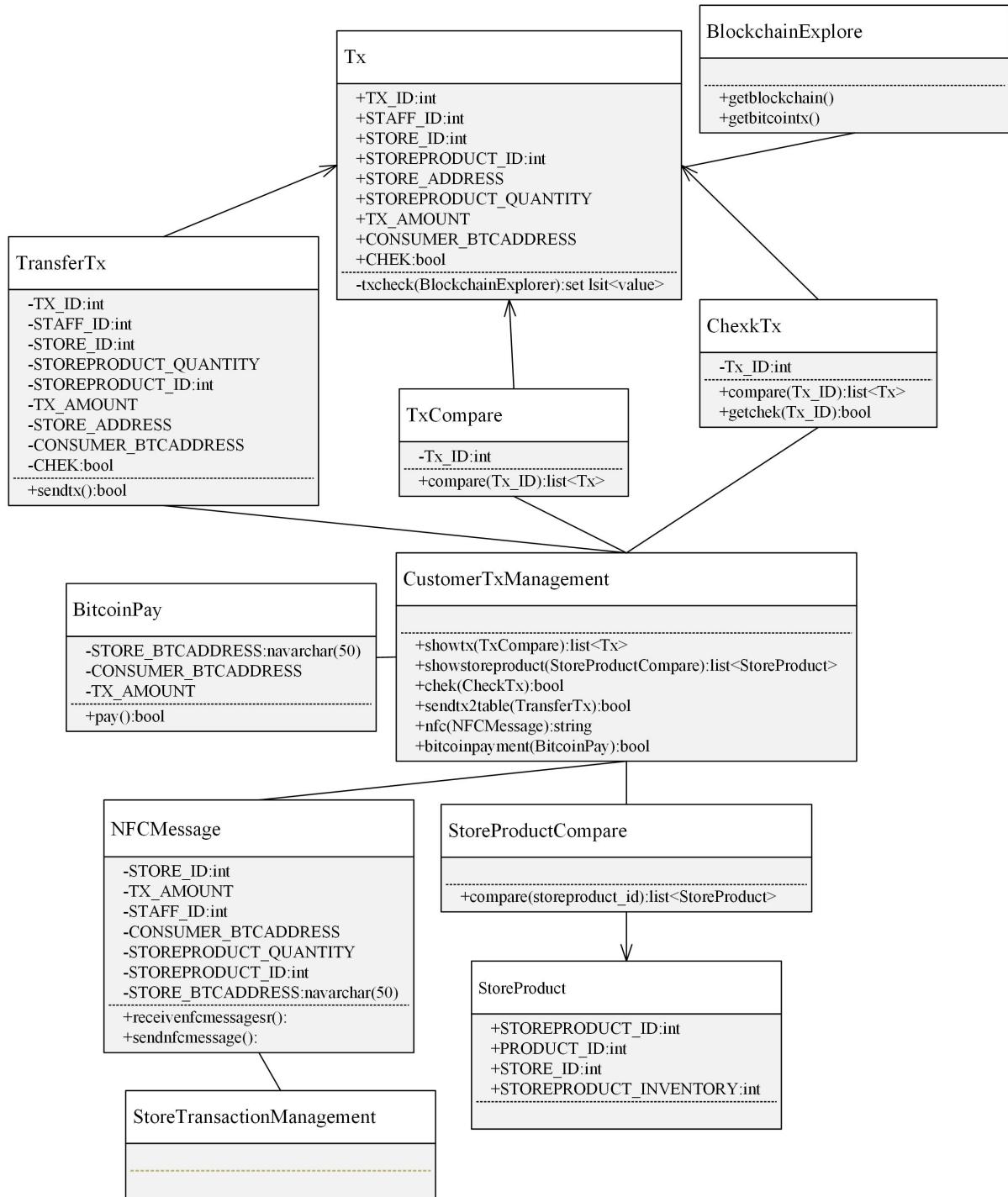


图 5.8 顧客交易管理模塊類圖

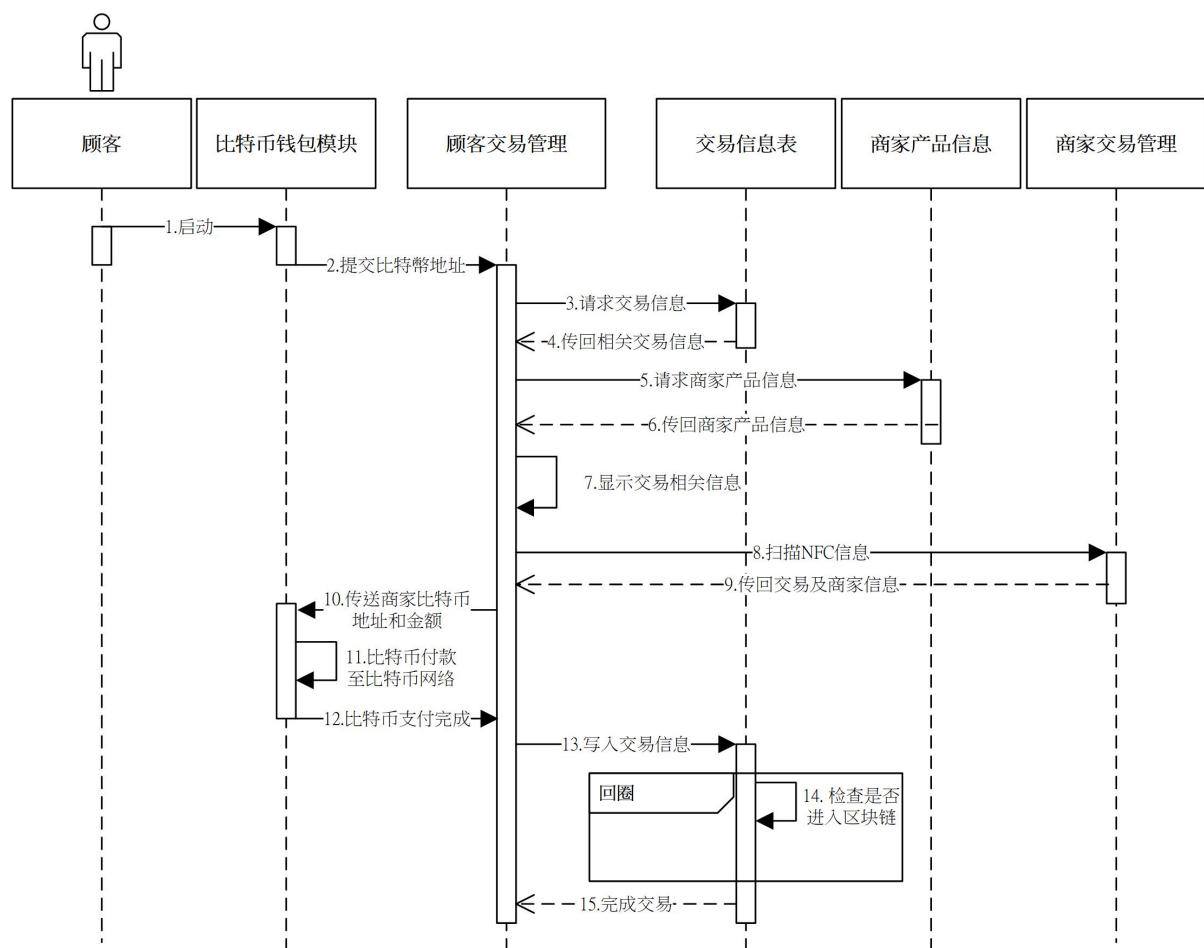


图 5.9 顧客交易管理時序圖

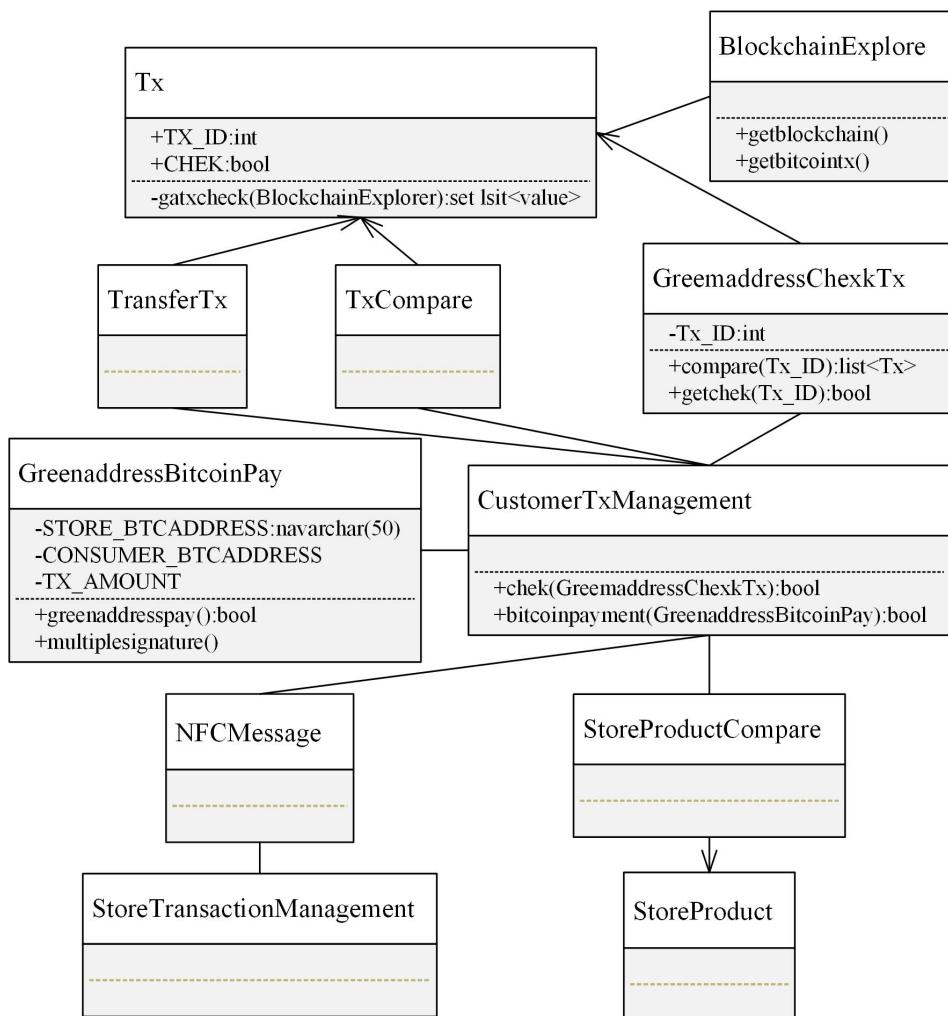


图 5.10 顧客交易管理 GA 模塊類圖

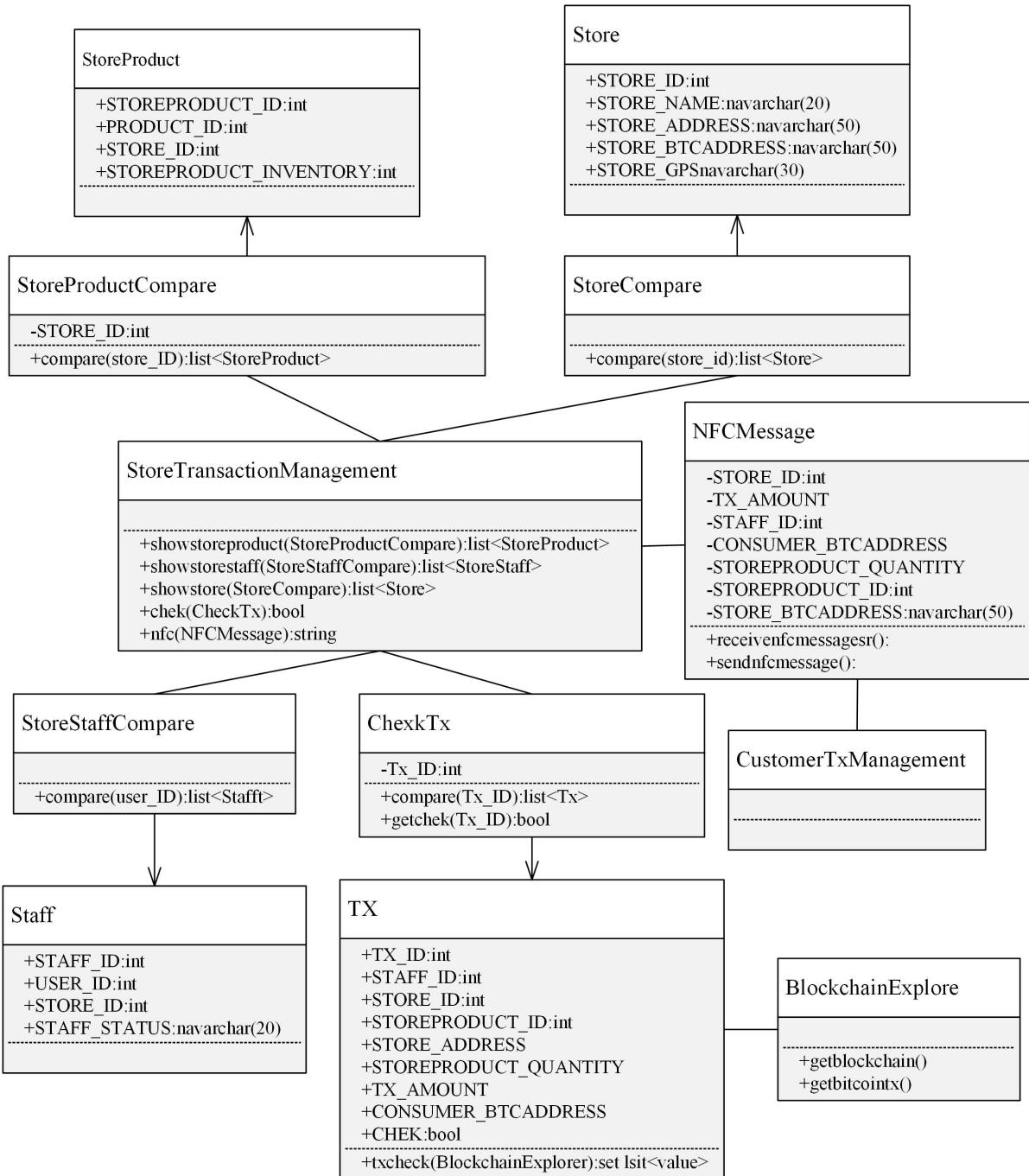


图 5.11 商家交易管理模塊類圖

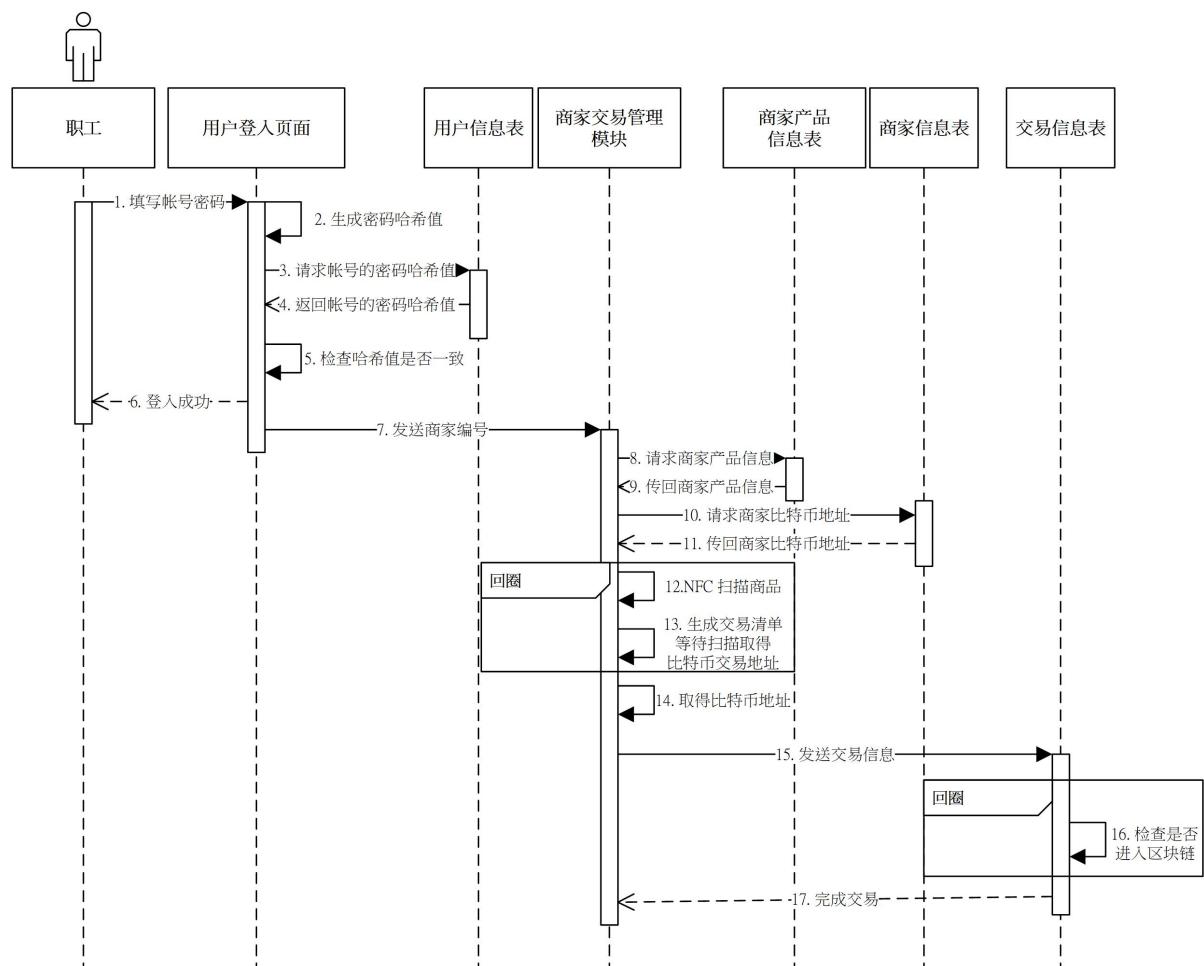


图 5.12 商家交易管理時序圖

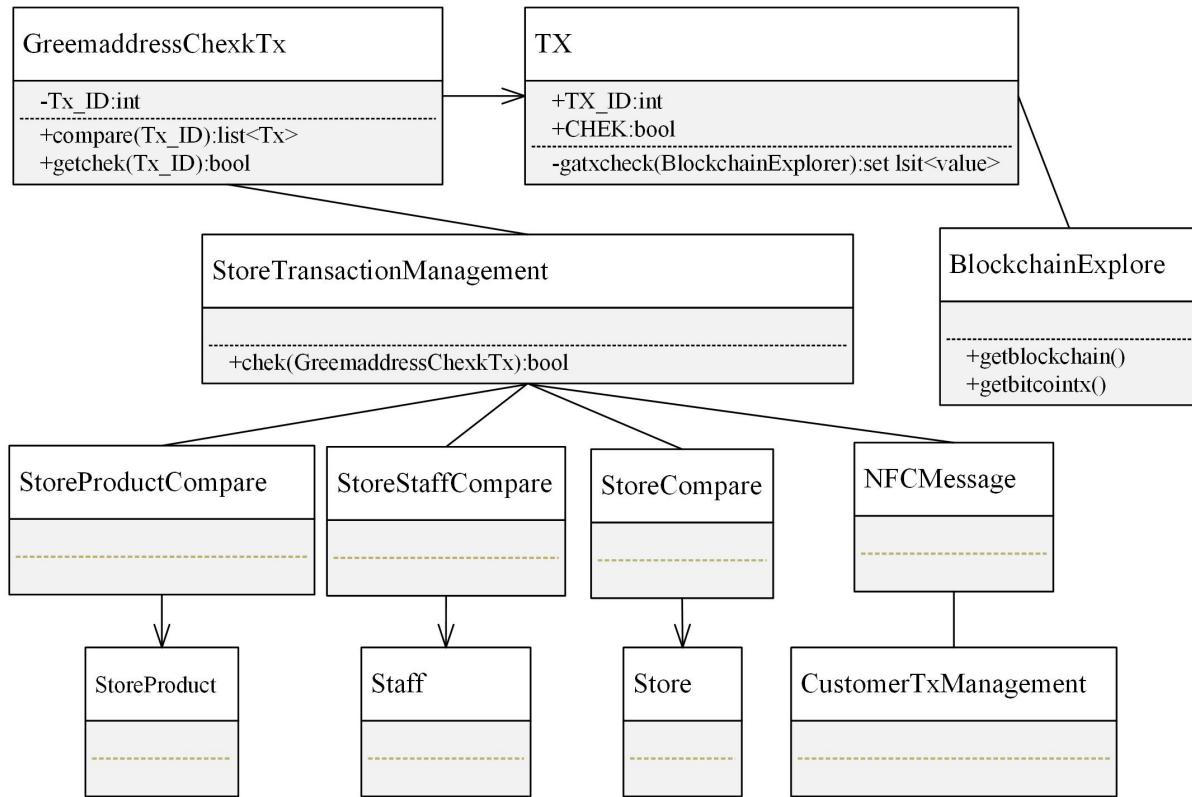


图 5.13 商家交易管理模塊類圖

5.3 系統實現

為了驗證和證明所提議的 BRTMS 用於比特幣支付收款監督的可行性和有效性，我們將其運行在用於商家商品管理和維護的 Java 應用程序的 SMIMSS 子系統，用於商家職工的運行在 Android App 上的 SMCTSS 以及運行在 App 上的用於客戶的 CMPTSS。如圖5.14所示，SMIMSS Java 應用程序可以幫助商家登錄到系統或創建一個新帳戶。授權商家成功登錄系統後，商家可以插入或更新產品列表，如圖5.15所示。實施的 SMIMSS Java 應用程序執行前面部分中所述的功能。

商家的產品信息可以通過 RFID 標籤掃描，存儲到雲端數據庫中，商家職工可以使用我們實現的 SMCTSS Android 客戶端，啟用 NFC 監聽器，從購物車中的客戶購買產品中讀取 RFID 標籤信息。在如圖5.16所示的第一項活動中，商家職工必須登錄才能獲得授權訪問 SMCTSS 功能。然後，在第二項活動中，SMCTSS 應用程序可以通過使用 SMIMSS 中應用的雲數據庫檢查產品 RFID 標籤信息並將其展示給客戶，從而將掃描的產品列入購物車。在圖5.16的第三項活動中，顧客可以要求職工取消購買物品以確認最終購買。最後，SMCTSS 應用程序將自動使用比特幣測試網絡（Bitcoin Testnet）^[54]幫助職工確認發布此加密貨幣交易的收款人地址，如圖5.16所示。

同時，客戶將使用與 SMCTSS App 相對應的 CMPTSS Android App 通過比特幣加

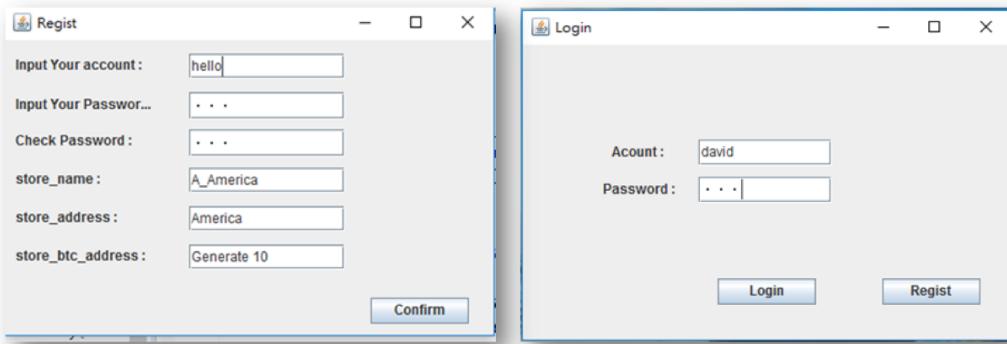


图 5.14 SMIMSS 的 Java 應用程序的註冊和登錄界面

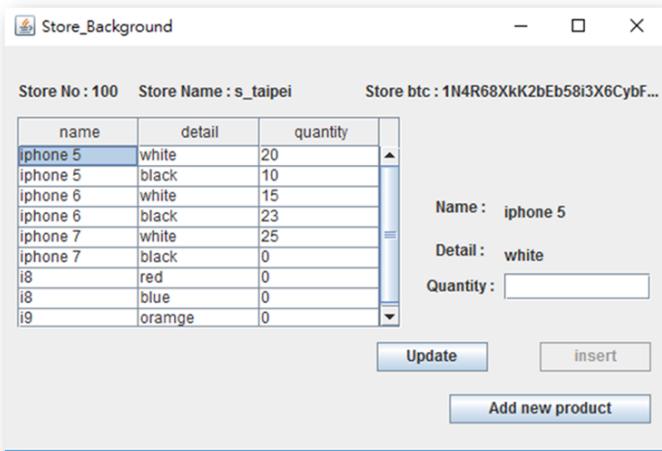


图 5.15 在 SMIMSS 中插入或更新授權商家的產品目錄

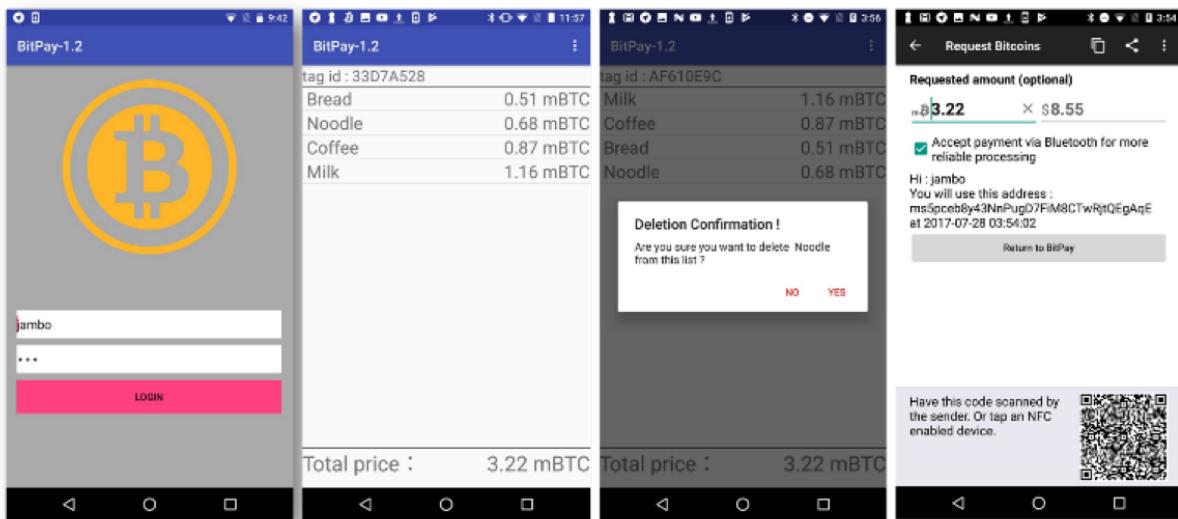


图 5.16 登錄、等待結帳的商品、刪除商品及支付確認

密貨幣完成採購產品交易。如圖5.17所示，第一個活動表示顧客確認購買產品創建交易數據庫的交易清單，第二個活動顯示包括金額和付款人比特幣地址在內的付款確認，第三個活動顯示交易歷史記錄的交易作為買方甚至是賣方，最後在第四項活動中顯示了該筆交易詳細採購產品的發票。

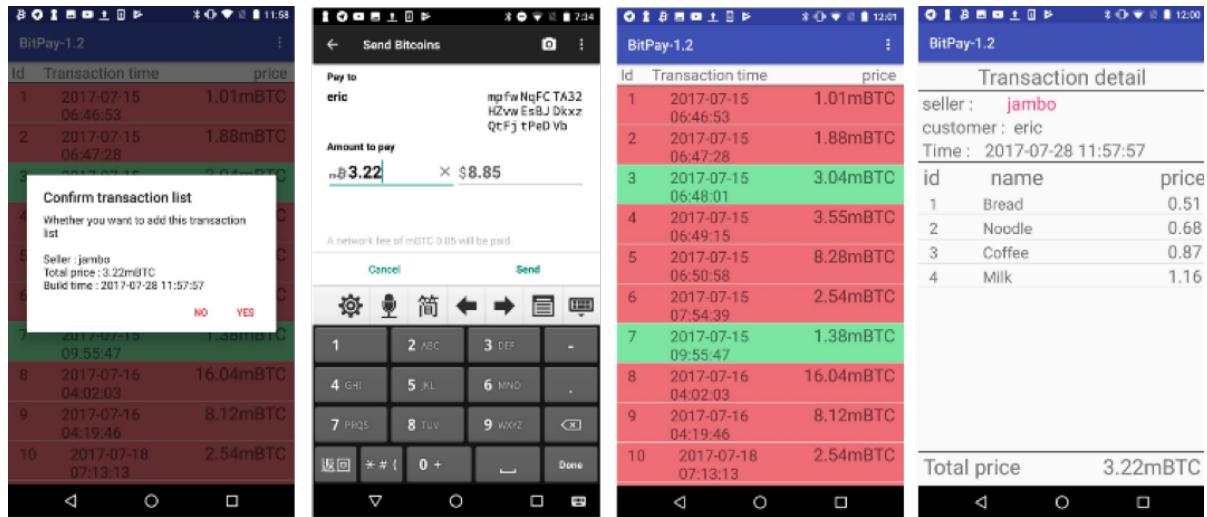


图 5.17 在 CMPTSS App 中，交易確認，付款確認、交易歷史記錄和發票

第六章 系統測試

於本章將進行功能測試與性能測試，執行功能測試已確認本文所提出之系統的所有功能是否皆順利運行，透過性能測試可知本系統的交易速度在優化前與優化後之間的差異。

6.1 功能測試

本段主要是描述區塊鏈的實名交易監督系統的測試計畫。確認在系統集成前，必須先確認所有的設計元件均可正確的輸出，在此我們著重於集成系統測試 (Integration Test) 及驗收測試 (Acceptance Test)。本章節內容將依據系統需求規格書與系統設計，描述關於集成測試的相關計畫與內容。並希望透過此章節之描述與實踐，達到順利進行測試工作之目的。

1. 接受標準：本測試計畫需要滿足下列的測試接受準則：
 - (a) 本系統需要對所有列為必要 (Critical、Important、Desirable) 之需求作完整測試。
 - (b) 測試程序需要依照本測試計畫所訂定的程序進行，所有測試結果需要能符合預期測試結果方能接受。
 - (c) 以測試案例為單位，當測試未通過時，需要進行該單元的測試，其接受的準則與前一項規定相同。
2. 測試環境說明包括所採用的硬件與軟件規格，分別如下：
 - (a) 硬件規格分為系統主機以及週邊設備：
 - i. 系統主機：一台以上主機，每台主機 CPU 為 Intel P4 1.0GHz 或以上，256 MB RAM 或以上，60G 以上硬碟空間。
 - ii. 周邊設備：一台以上智慧型手機，與用來代表虛擬商品的數個 RFID 標籤；已可供測試 NFC 用的智慧型手機包含小米 3 WCDMA 版，詳細規格於表6.1，Google Nexus 5X，詳細規格於表6.2。
 - (b) 軟件規格：關於測試環境所需的軟體規格說明，如下列所示：作業系統：Window 10、Android 6.0.1/7.1.1
3. 測試地點：在銘傳大學桃園校區資工系實驗室，透過 Android 手機進行的交易模擬實驗，測試環境如圖4.6的示意。
4. 測試時間：

表 6.1 小米 3 手機規格

| | |
|------|--|
| 系統頻率 | GSM 四頻、WCDMA |
| 作業系統 | Android 4.3 |
| 處理器 | Qualcomm Snapdragon 800 2.3 GHz 四核心 |
| 記憶體 | 2GB RAM、16GB ROM |
| 記憶卡 | 不支援 |
| 顯示螢幕 | 5 吋 1670 萬色 IPS(1920×1080 pixels)、441ppi |
| 相機 | 1300 萬畫素 (F2.2、28mm)、200 萬副鏡頭、1080p |
| 電池 | 3050 mAh(不可換) |
| 尺寸 | 144x73.6x8.1mm |
| 重量 | 145g |

表 6.2 Google Nexus 5X 手機規格

| | |
|------|--|
| 系統頻率 | GSM 四頻、WCDMA |
| 作業系統 | Android 6.0 |
| 處理器 | Qualcomm Snapdragon 800 1.8 GHz 六核 |
| 記憶體 | 2GB RAM、16GB ROM |
| 記憶卡 | 不支援 |
| 顯示螢幕 | 5 吋 1670 萬色 IPS(1920×1080 pixels)、441ppi |
| 相機 | 1300 萬畫素 (F2.2、28mm)、200 萬副鏡頭、1080p |
| 電池 | 2,700 mAh(不可換) |
| 尺寸 | 147x72.6x7.9mm |
| 重量 | 136g |

- (a) 各子系統之內部元件集成測試 (Module Test)(2017/2/25 2017/6/8)
- (b) 區塊鏈的實名交易監督系統集成測試 (Integration Test) (2017/6/8 2017/6/21)
- (c) 區塊鏈的實名交易監督系統接受度測試 (Acceptance Test) (2017/7/10 2017/7/21)

5. 查核點：

- (a) 各子系統之內部元件集成測試 (2017/5/10)
- (b) 區塊鏈的實名交易監督系統集成測試 (2017/7/1)
- (c) 區塊鏈的實名交易監督系統接受度測試 (2017/7/1)

6. 集成測試規劃 (Integration Testing) :

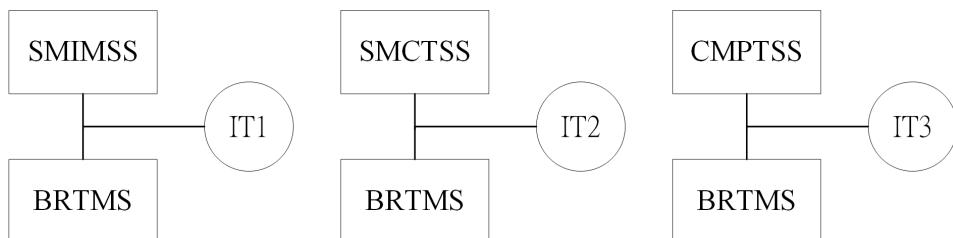


图 6.1 集成子系統測試

7. 驗收測試規劃 (Acceptance Testing, AT)：本系統須達成以下三組接受用例陳列的所有功能，測試本論文搭設計與搭建的系統功能是否能夠順利運行。測試的角色有兩個，分別為管理員以及使用者，如圖6.2為BRTMS用例示意圖，預計測試服務器的組態設定、手機的組態設定以及數據庫的組態設定：

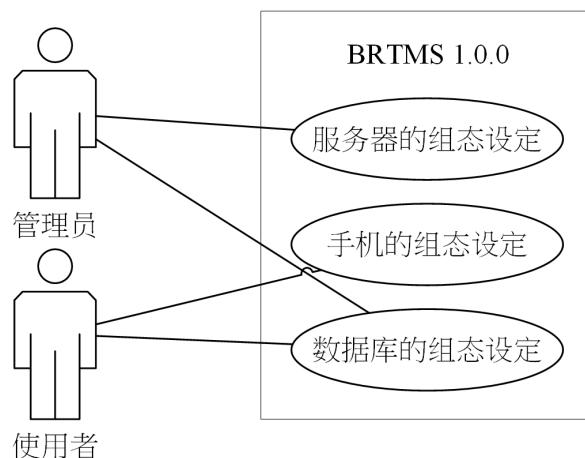


图 6.2 BRTMS 使用者用例圖

圖6.3為三組驗收測試的示意圖，對本區塊鏈的實名交易監督系統進行測試。

6.1.1 测试用例

1. 集成测试 (Integration Test)

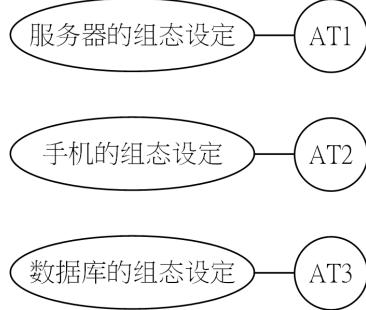


图 6.3 验收测试 (Acceptance Testing)

(a) IT1 测试用例：表6.3為 IT1 测试用例，測試對象為商家端建置與管理商品資訊子系統。目的：驗證 [SMIMSS 1.1.0] 子系統能否正確管理商品資訊。

表 6.3 IT1 测试用例

| | |
|---------|---|
| 用例 ID | IT1 |
| 用例名稱 | 集成 SMIMSS 至 BRTMS |
| 測試目標 | [SMIMSS 1.1.0]、[BRTMS 1.0.0] |
| 依賴關係 | SMIMSS-F-001~ SMIMSS-F-005 |
| 嚴重程度 | 1(Critical) |
| 用例描述 | 1. 能夠新增店家帳戶 2. 能夠新增/修改/刪除店員帳戶 3. 能夠新增/刪除/修改商品資訊 4. 能夠取得產品資訊 5. 能夠接收交易資訊 |
| 預期結果 | 1. 成功新增店家帳戶 2. 成功新增/修改/刪除店員帳戶 3. 成功新增/修改/刪除商品資訊 4. 成功取得產品資訊 5. 成功接收交易資訊 |
| Cleanup | 無 |

(b) IT2 测试用例：表6.4為 IT2 测试用例麼目標，測試對象為商家端行動收銀與交易明細系統。目的：驗證 [SMCTSS 1.2.0] 子系統是否能夠完成一筆行動支付之交易。

(c) IT3 测试用例：表6.5為 IT3 测试用例，目標檢測對象為顧客端行動支付與交易明細系統。目的：驗證 [CMPTSS1.3.0] 能正確接收 SMCTSS 所傳送的交易資料，並以其交易資訊執行以比特幣付款之動作。可以查詢商品資訊，且能夠儲存並且查看使用者過往之交易紀錄。

2. 验收测试用例 (Acceptance Testing Cases)：验收测试用例目的在於測試母系統 BRTMS、子系統 SMIMSS、SMCTSS 與 CMPTSS 是否能夠順利的進行信息傳遞

表 6.4 IT2 測試用例

| | |
|---------|--|
| 用例 ID | IT2 |
| 用例名稱 | 集成 SMCTSS 至 BRTMS |
| 測試目標 | [SMCTSS1.2.0]、[BRTMS 1.0.0] |
| 依賴關係 | SMCTSS-F-001~ SMCTSS-F-007 |
| 嚴重程度 | 1(Critical) |
| 用例描述 | 1. 能夠登入店員帳戶 2. 能夠掃描 NFC 標籤 3. 能夠讀取商品資訊 4. 能夠建立交易清單 5. 能夠傳送交易資訊 6. 能夠認證交易資訊 7. 能夠儲存交易明細 |
| 預期結果 | 1. 成功登入店員帳戶 2. 成功掃描 NFC 標籤 3. 成功讀取商品資訊 4. 成功建立交易清單 5. 成功傳送交易資訊 6. 成功認證交易資訊 7. 成功儲存交易明細 |
| Cleanup | 無 |

表 6.5 IT3 測試用例

| | |
|---------|---|
| 用例 ID | IT3 |
| 用例名稱 | 集成 CMPTSS 至 BRTMS |
| 測試目標 | [CMPTSS.1.3.0]、[BRTMS 1.0.0] |
| 依賴關係 | CMPTSS-F-001~ CMPTSS-F-007 |
| 嚴重程度 | 1(Critical) |
| 用例描述 | 1. 能夠登入客戶帳號 2. 能夠讀取商品資訊 3. 能夠接收交易清單 4. 能夠認證交易資訊 5. 能夠執行行動支付 6. 能夠儲存交易明細 7. 能夠查看交易紀錄 |
| 預期結果 | 1. 成功登入客戶帳號 2. 成功讀取商品資訊 3. 成功接收交易清單 4. 成功認證交易資訊 5. 成功執行行動支付 6. 成功儲存交易紀錄 7. 成功查看交易紀錄 |
| Cleanup | 無 |

完成交互。

(a) AT1 测试用例: 表6.6所示, 目標測試管理人員是否能夠順利使用子系統SMIMSS順利與母系統BRTMS 教戶。目的: 驗證使用用例 (Use case) 1, 透過組態檔案的修改對伺服器進行組態設定。

表 6.6 AT1 测试用例

| | | |
|---------|---------------------------------|--------------------------|
| 用例 ID | AT1 | |
| 用例名稱 | 伺服器的組態設定 | |
| 測試目標 | [SMIMSS 1.1.0] [BRTMS 1.0.0] | |
| 依賴關係 | BRTMS-F-001 | |
| 嚴重程度 | 1 | |
| 用例描述 | 使用者操作 | 系統響應 |
| | 1. 管理人員依照環境設定伺服器組態。 | |
| | | 2. 伺服器依照管理人員所做的組態設定啟動服務。 |
| 預期結果 | 成功啟動伺服器的相關服務。 | |
| Cleanup | 無 | |

(b) AT2 测试用例: 如表6.7所示, 參與者為使用者。目的: 驗證用例 (Use case) 2 透過組態檔案的修改對手機進行組態設定。

表 6.7 AT2 测试用例

| | | |
|---------|----------------------------------|--------------------------|
| 用例 ID | AT2 | |
| 用例名稱 | 手機的組態設定 | |
| 測試目標 | [SMCTSS 1.2.0] [CMPTSS 1.3.0] | |
| 依賴關係 | BRTMS-F-002~ BRTMS-F-003 | |
| 嚴重程度 | 1 | |
| 用例描述 | 使用者操作 | 系統響應 |
| | 1. 使用者修改手機組態設定參數。 | |
| | | 2. 手機依照使用者在設定檔中所填入的數值運作。 |
| 預期結果 | 成功完成手機的組態設定 | |
| Cleanup | 無 | |

(c) AT3 测试用例: 如表6.8所示, 目的: 驗證用例 (Use case) 3, 透過組態檔案的修改對資料庫進行組態設定。

表 6.8 AT3 測試用例

| | | |
|---------|--|--------------------------|
| 用例 ID | AT3 | |
| 用例名稱 | 資料庫的組態設定 | |
| 測試目標 | [SMIMSS 1.1.0] [SMCTSS 1.2.0] [CMPTSS 1.3.0] | |
| 依賴關係 | BRTMS-F-001~ BRTMS-F-003 | |
| 嚴重程度 | 1 | |
| 用例描述 | 使用者操作 | 系統響應 |
| | 1. 管理者設定資料庫組態。 | |
| | | 2. 資料庫依照管理人員所做的組態設定啟動服務。 |
| | 3. 使用者修改資料庫之資料及檔案。 | |
| | | 4. 資料庫依照使用者所做的組態設定啟動服務。 |
| 預期結果 | 成功設定完成資料庫的相關設定。 | |
| Cleanup | 無 | |

6.1.2 測試結果和分析

1. 集成測試用例 (Integration Testing Cases) 表6.9為 IT 1、為 IT 2、為 IT 3 的集成子系統測試結果，皆順利運作。
2. 驗收測試用例 (Acceptance Testing Cases) 表6.10為前節所設計的三種 AT 1、AT 2 與 AT 3 的驗收測試結果，皆順利運行。
3. 可追蹤性 (Traceability)
 - (a) 子系統與測試用例：表6.11為測試組 IT 1、IT 2、IT 3、AT 1、AT 2、AT 3 與子系統 SMIMSS、SMCTSS、CMPISS 的關係表。
 - (b) 需求與測試用例：表6.12為本系統需求與集成測試用例的關係表，表6.13為需求與驗收測試案例的關係表。

6.2 性能測試

根據比特幣點對點架構，儘管客戶和商家之間的交易細節已經快速存儲到雲數據庫，但官方確認交易與當前比特幣區塊鏈的交易通常需要更長的時間，因為需要確保確認的數量在交易廣播比特幣點對點網絡並存儲到緩存池後，是否存在雙重支付。本文設計了区块链的实名交易监督系统以及引用多重簽章算法重新設計的区块链的实名

表 6.9 集成子系統測試結果

| 测试用例 | 結果(通過/不通過) | Comment |
|------|------------|---|
| IT1 | 通過 | 1. 成功新增店家帳戶 2. 成功新增/修改/刪除店員帳戶 3. 成功新增/修改/刪除商品資訊 4. 成功取得產品資訊 5. 成功接收交易資訊 |
| IT2 | 通過 | 1. 成功登入店員帳戶 2. 成功掃描 NFC 標籤 3. 成功讀取商品資訊 4. 成功建立交易清單 5. 成功傳送交易資訊 6. 成功認證交易資訊 7. 成功儲存交易明細 |
| IT3 | 通過 | 1. 成功登入客戶帳號 2. 成功讀取商品資訊 3. 成功接收交易清單 4. 成功認證交易資訊 5. 成功執行行動支付 6. 成功儲存交易紀錄 7. 成功查看交易紀錄 |
| RATE | 90% | BRTMS 開發透過手機讓商家及顧客以手機傳送交易資訊，如：商品名稱、商品金額，商家收款地址。並且及時將商品資訊更新至伺服器之資料庫，以便商家控管商品資訊狀態，同時讓顧客可以享受數位加密貨幣的方便性。 |

表 6.10 驗收測試結果

| 测试用例 | 結果(通過/不通過) | Comment |
|------|------------|---------------------------------------|
| AT1 | 通過 | 成功啟動伺服器的相關服務。 |
| AT2 | 通過 | 成功完成手機的組態設定。 |
| AT3 | 通過 | 成功設定完成資料庫的相關設定。 |
| Rate | 100% | BRTMS 可透過組態設定的方式來設定各個子系統的環境參數。 |

表 6.11 子系統與測試用例關係表

| | SMIMSS 1.1.0 | SMCTSS 1.2.0 | CMPISS 1.3.0 |
|-----|--------------|--------------|--------------|
| IT1 | X | | |
| IT2 | | X | |
| IT3 | | | X |
| AT1 | X | | |
| AT2 | | X | X |
| AT3 | X | X | X |

表 6.12 需求與集成測試用例的關係表

| | IT1 | IT2 | IT3 |
|--------------|-----|-----|-----|
| BRTMS-F-001 | X | | |
| BRTMS-F-002 | | X | |
| BRTMS-F-003 | | | X |
| SMIMSS-F-001 | X | | |
| SMIMSS-F-002 | X | | |
| SMIMSS-F-003 | X | | |
| SMIMSS-F-004 | X | | |
| SMIMSS-F-005 | X | | |
| SMCTSS-F-001 | | X | |
| SMCTSS-F-002 | | X | |
| SMCTSS-F-003 | | X | |
| SMCTSS-F-004 | | X | |
| SMCTSS-F-005 | | X | |
| SMCTSS-F-006 | | X | |
| SMCTSS-F-007 | | X | |
| CMPTSS-F-001 | | | X |
| CMPTSS-F-002 | | | X |
| CMPTSS-F-003 | | | X |
| CMPTSS-F-004 | | | X |
| CMPTSS-F-005 | | | X |
| CMPTSS-F-006 | | | X |
| CMPTSS-F-007 | | | X |

表 6.13 需求與驗收測試案例的關係表

| | AT1 | AT2 | AT3 |
|-------------|-----|-----|-----|
| BRTMS-F-001 | X | | X |
| BRTMS-F-002 | | X | X |
| BRTMS-F-003 | | X | X |

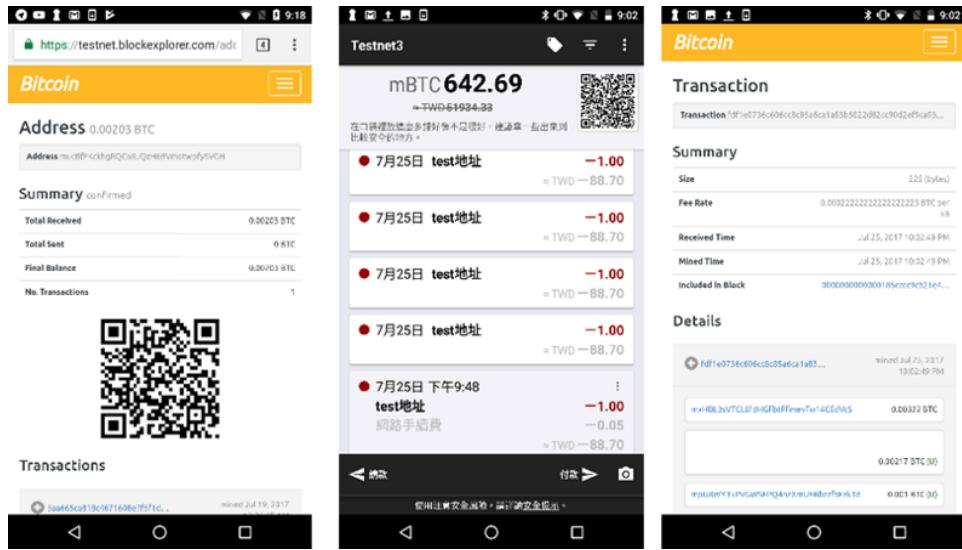


图 6.4 使用區塊鏈瀏覽器驗證存儲在比特幣區塊鏈中的交易過程

交易實時監督系統，以下將分別針對有無採用多重簽章算法的性能測試。

6.2.1 系统性能測試

為了驗證本文提出的 BRTMS 不會通過使用比特幣等加密貨幣影響交易完成時間，我們於 2017 年 7 月 25 日、2017 年 9 月 6 日以及 2018 年 3 月 21 日，在 Testnet 實驗中連續記錄了 20 筆交易信息，每 1 秒發出一筆交易，分別歷時 20 分鐘。

1. 初始測試（2017 年 7 月 25 日）：首先，我們使用區塊鏈檢視器（Blockchain Explorer）^[55]，如 6.4 的快照所示，透過使用 Testnet 依序進行 20 筆比特幣交易，如圖 6.4 的中間快照所示，最後 20 筆交易完成時間全部記錄在區塊鏈檢視器。實驗結果顯示，實驗中的所有交易都在 3 秒鐘左右（平均 2.97 秒，標準差小於 1 秒）發送到比特幣網絡緩存池，平均交易完成時間在比特幣區塊鏈中確認為 522.33 秒（小於 9 分鐘），標準差大約為 339 秒。
2. 第一次測試（2017 年 9 月 6 日）：表 6.14 為第一次實驗數據，該次的比特幣交易法起廣播至比特幣交易緩存池的時間平均為 1.918 秒，標準差為 0.55586 秒。但為了預防雙花攻擊需要等待平均 654.8 秒（小於 11 分鐘，標準差 346.63 秒）的時間。
3. 第二次測試（2018 年 3 月 21 日）：表 6.15，表 6.16

在未引用多重簽章算法的監督系統中，第一次與第二次的測試相比並無太大的差異，交易完成時間與比特幣區塊生成所需時間相同，平均時間皆為十分鐘。根據比特幣 Testnet 上的初步實驗結果顯示，本文提出的 BRTMS 可以快速有效地執行区块链的实名交易监督系统。

表 6.14 第一次 Testnet 執行實驗之數據分析 (2017 年 9 月 6 日)

| | 進入緩存池時間 (秒) | 進入區塊鏈時間 (秒) | 完成交易時間 (秒) |
|----------|-------------|---------------|---------------|
| 平均 | 1.918 | 654.8 | 654.8 |
| 樣本標準差 | 0.55586 | 346.63 | 346.63 |
| 95% 信賴區間 | 1.69~2.15 | 511.72~797.88 | 511.72~797.88 |
| 99% 信賴區間 | 1.61~2.23 | 460.9~848.7 | 460.9~848.7 |

表 6.15 第二次 Testnet 執行實驗數據 (2018 年 3 月 21 日)

| 交易 次數 | 付款時間 | 進入緩存池 所花時間 (秒) | 進入緩存池 時間點 | 寫入區塊鏈 所花時間 (秒) | 寫入區塊 時間點 |
|----------|---------|-------------------|--------------|-------------------|-------------|
| 1 | 0:00:00 | 3 | 0:00:03 | 136 | 0:02:16 |
| 2 | 0:01:00 | 1 | 0:01:01 | 76 | 0:02:16 |
| 3 | 0:02:04 | 3 | 0:02:07 | 12 | 0:02:16 |
| 4 | 0:03:00 | 1 | 0:03:01 | 438 | 0:10:18 |
| 5 | 0:04:00 | 1 | 0:04:01 | 378 | 0:10:18 |
| 6 | 0:05:00 | 1 | 0:05:01 | 318 | 0:10:18 |
| 7 | 0:06:00 | 2 | 0:06:02 | 258 | 0:10:18 |
| 8 | 0:07:00 | 1 | 0:07:01 | 198 | 0:10:18 |
| 9 | 0:08:00 | 1 | 0:08:01 | 138 | 0:10:18 |
| 10 | 0:09:00 | 1 | 0:09:01 | 78 | 0:10:18 |
| 11 | 0:10:00 | 1 | 0:10:01 | 18 | 0:10:18 |
| 12 | 0:11:00 | 2 | 0:11:02 | 810 | 0:24:30 |
| 13 | 0:12:00 | 1 | 0:12:01 | 750 | 0:24:30 |
| 14 | 0:13:00 | 1 | 0:13:01 | 690 | 0:24:30 |
| 15 | 0:14:00 | 1 | 0:14:01 | 630 | 0:24:30 |
| 16 | 0:15:00 | 1 | 0:15:01 | 570 | 0:24:30 |
| 17 | 0:16:00 | 2 | 0:16:02 | 510 | 0:24:30 |
| 18 | 0:17:00 | 2 | 0:17:02 | 450 | 0:24:30 |
| 19 | 0:18:00 | 1 | 0:18:01 | 390 | 0:24:30 |
| 20 | 0:19:00 | 1 | 0:19:01 | 330 | 0:24:30 |

表 6.16 第二次以 Testnet 執行實驗數據分析 (2018 年 3 月 21 日)

| | 進入緩存池時間 (秒) | 進入區塊鏈時間 (秒) | 完成交易時間 (秒) |
|----------|-------------|---------------|---------------|
| 平均 | 1.12 | 287.12 | 287.12 |
| 樣本標準差 | 0.68 | 246.59 | 246.59 |
| 95% 信賴區間 | 0.84~1.4 | 185.33~388.91 | 185.33~388.91 |
| 99% 信賴區間 | 0.74~1.5 | 149.18~425.06 | 149.18~425.06 |

6.2.2 實時系統性能測試

本主要介紹比特幣測試幣於 Green Address 錢包進行交易的效能實驗與結果分析，包含該實驗的目的、方法及結果分析。實驗的目的是要確認我們的系統在商家端進行行動支付時能快速、精準且高效率的進行交易，並瞭解使用一般 Testnet 錢包與 Green Address 錢包作為交易媒介的確認交易時間的差距。表6.17為2017年7月25日初步的採用多重簽章算法的測試數據，數據顯示交易存儲到區塊鏈的平均時間為10分鐘，但Green Address 採用的多重簽章算法，可以有效拒絕雙重花費的交易發起，因此可將交易的有效時間，從交易進入區塊鏈的時間，重新訂定為交易進入比特幣交易緩存持的時間。

表 6.17 初步的 Green Address 實驗測試數據 (2017 年 7 月 25 日)

| 付款時間 | 進入緩存池 所花時間 (秒) | 進入緩存池 時間點 | 入塊時間 | 寫入區塊 費時間 |
|----------|-------------------|--------------|----------|-------------|
| 06:36:25 | 02:14 | 06:36:27.14 | 06:52:00 | 16:25 |
| 06:38:25 | 1.075 | 06:38:26.075 | 06:52:00 | 14:25 |
| 06:40:25 | 1.722 | 06:40:26.722 | 06:52:00 | 12:25 |
| 06:42:25 | 2.953 | 06:42:27.953 | 06:52:00 | 10:25 |
| 06:44:25 | 1.511 | 06:44:26.511 | 06:52:00 | 08:25 |
| 06:46:25 | 2.269 | 06:46:27.269 | 06:52:00 | 06:25 |
| 06:48:25 | 1.831 | 06:48:26.831 | 06:52:00 | 04:25 |
| 06:50:25 | 1.227 | 06:50:26.227 | 06:52:00 | 02:25 |
| 06:52:25 | 2.026 | 06:52:27.026 | 07:12:01 | 20:24 |
| 06:54:25 | 1.257 | 06:54:26.257 | 07:12:01 | 18:24 |
| 06:56:25 | 1.511 | 06:56:26.511 | 07:12:01 | 16:24 |
| 06:58:25 | 2.815 | 06:58:27.815 | 07:12:01 | 14:24 |
| 07:00:26 | 1.544 | 07:00:27.544 | 07:12:01 | 12:25 |
| 07:02:25 | 1.767 | 07:02:26.767 | 07:12:01 | 10:24 |
| 07:04:25 | 1.52 | 07:04:26.52 | 07:12:01 | 08:24 |
| 07:06:25 | 1.953 | 07:06:26.953 | 07:12:01 | 06:24 |

本次的實驗分為兩部份，分別是透過比特幣 Testnet 錢包以及使用本論文所採用的 Green Address 比特幣錢包上執行 20 次付款，皆以相同地址收款，交易金額都設定為 0.00001BTC，實驗時間為 2017 年 9 月 6 日與 2018 年 2 月 6 日，每隔一分鐘執行一次付款的動作，總共歷時 20 分鐘。兩款錢包同時發起交易，並透過區塊鏈檢視器進行記錄時間，最後再比較使用一般比特幣錢包及 Green Address 錢包兩者之間的差距。

1. 第一次測試 (2017 年 9 月 6 日): 表6.18
2. 第二次測試 (2018 年 3 月 21 日): 表6.19, 表6.20

本次實驗分別記錄以 Testnet 錢包及 Green Address 錢包執行 20 次交易的進入緩存

表 6.18 第一次以 GreenAddress 執行實驗之數據分析 (2017 年 9 月 6 日)

| | 進入緩存池時間 (秒) | 進入區塊鏈時間 (秒) | 完成交易時間 (秒) |
|----------|-------------|---------------|------------|
| 平均 | 2.11 | 654.24 | 2.11 |
| 樣本標準差 | 0.65 | 346.9 | 0.65 |
| 95% 信賴區間 | 1.84~2.38 | 511.05~797.43 | 1.84~2.38 |
| 99% 信賴區間 | 1.75~2.47 | 460.19~848.29 | 1.75~2.47 |

表 6.19 第二次以 GreenAddress 執行實驗之數據 (2018 年 3 月 21 日)

| 交易 次數 | 付款時間 | 進入緩存池 所花時間 (秒) | 進入緩存池 時間點 | 寫入區塊鏈 所花時間 (秒) | 寫入區塊 時間點 |
|----------|---------|-------------------|--------------|-------------------|-------------|
| 1 | 0:00:01 | 2 | 0:00:03 | 619 | 0:10:18 |
| 2 | 0:01:00 | 1 | 0:01:01 | 559 | 0:10:18 |
| 3 | 0:02:02 | 2 | 0:02:04 | 496 | 0:10:18 |
| 4 | 0:03:00 | 1 | 0:03:01 | 438 | 0:10:18 |
| 5 | 0:04:00 | 1 | 0:04:01 | 378 | 0:10:18 |
| 6 | 0:05:00 | 2 | 0:05:02 | 318 | 0:10:18 |
| 7 | 0:06:00 | 1 | 0:06:01 | 258 | 0:10:18 |
| 8 | 0:07:00 | 1 | 0:07:01 | 198 | 0:10:18 |
| 9 | 0:08:00 | 2 | 0:08:02 | 138 | 0:10:18 |
| 10 | 0:09:00 | 2 | 0:09:02 | 78 | 0:10:18 |
| 11 | 0:10:00 | 1 | 0:10:01 | 18 | 0:10:18 |
| 12 | 0:11:00 | 2 | 0:11:02 | 810 | 0:24:30 |
| 13 | 0:12:00 | 2 | 0:12:02 | 750 | 0:24:30 |
| 14 | 0:13:00 | 1 | 0:13:01 | 690 | 0:24:30 |
| 15 | 0:14:00 | 2 | 0:14:02 | 630 | 0:24:30 |
| 16 | 0:15:00 | 1 | 0:15:01 | 570 | 0:24:30 |
| 17 | 0:16:00 | 2 | 0:16:02 | 510 | 0:24:30 |
| 18 | 0:17:00 | 2 | 0:17:02 | 450 | 0:24:30 |
| 19 | 0:18:00 | 1 | 0:18:01 | 390 | 0:24:30 |
| 20 | 0:19:00 | 2 | 0:19:02 | 330 | 0:24:30 |

表 6.20 第二次以 GreenAddress 執行實驗之數據分析 (2018 年 3 月 21 日)

| | 進入緩存池時間 (秒) | 進入區塊鏈時間 (秒) | 完成交易時間 (秒) |
|----------|-------------|---------------|------------|
| 平均 | 1.55 | 431.4 | 1.55 |
| 樣本標準差 | 0.51 | 220.84 | 0.51 |
| 95% 信賴區間 | 1.34~1.76 | 340.24~522.56 | 1.34~1.76 |
| 99% 信賴區間 | 1.26~1.84 | 307.86~554.94 | 1.26~1.84 |

池等待時間和寫入區塊等待時間。若以 Testnet 錢包交易，必須等到交易寫入才能保證此筆交易不會被礦工遺棄，也算真的完成這筆交易；但若以 Green Address 錢包發起交易就大不相同，當交易進入緩存池，即使遇到交易被礦工遺棄的情況，Green Address 機構節點也會重新發起此筆交易，保證讓交易寫入區塊，所以只要進入緩存池我們就可以視為交易完成，透過兩者錢包的交易數據，本文分析兩種錢包交易的時間數據。

表 6.21 原始系統與實時系統比較表

| | 完成交易時間 (秒) |
|------------|------------|
| 原始系統交易平均時間 | 287.12 |
| 實時系統交易平均時間 | 1.55 |

透過本次的實驗可以發現雖然以兩種錢包交易進入區塊的等待時間完全相同，但因為 Green Address 錢包的特性，只要進入緩存池就算完成交易確認，因此 Green Address 錢包的完成交易確認的時間遠遠快於一般 Testnet。相信以此方式作為主要支付管道，可以省去消費者在現金支付時掏零錢、算錢及找零等繁瑣的動作及時間，以此達成提升日常生活中的便利性與安全性。

第七章 总结与展望

在文論本中，我們建議並實施一個名為 BRTMS 的區塊鏈實名交易監督系統，不僅為分別花錢和賺取加密貨幣的客戶和商家，同時也能協助府金融監督單位審計加密貨幣交易籌集稅收。此外，加密貨幣交易實驗的初步結果，使用比特幣測試幣及於章節 4.1 所設計的 BRTMS 數據庫，我們以著名的比特幣加密貨幣錢包實作的 Java 應用程序和 Android 應用程序客戶端。

此外，修改其在 Android 系統上的開放原始碼應用程式，使得本論文闡述之概念能夠實際在區塊鏈上運行，以期未來加密貨幣不僅能維持目前的便利性，還可以讓使用者不用擔心交易後找不到賣家，使一般民眾能更安心使用加密貨幣作為日常生活的行動支付管道。本文提出的 BRTMS 架構還包括以下特色如下：

1. 向建議的 BRTMS 進行商業註冊需經政府批准。
2. 所有出售的商品將受到政府審查。
3. 消費者仍然匿名以確保個人信息的隱私。
4. 消費者的交易記錄不能被刪除。
5. 如果消費者有關交易上訴的問題，他們需要提交交易發票或證明付款人或收款人地址的訪問權。
6. 所有交易記錄均公開透明。
7. 原始交易數據採用區塊鏈技術記錄，具有高度的可靠性，分散和未篡改的數據。
8. 政府可以檢查建議的 BRTMS 系統的交易記錄，以快速有效的方式審查稅務信息。

針對消費者、商家以及政府，採用提議的 BRTMS 有以下優點：

1. 消費者：交易信息公開透明，保護消費者的權益。由於交易是可信的，並有明確的時間戳。當消費者需要上交他們的消費者權利時，他們可以從提出的系統中獲得更有效和可信的證據。
2. 商家：企業可以根據所有數字化交易信息為自己的業務目的進行統計和計算。這可以減少手動操作計算結果中的錯誤。統計數據甚至可以與商家的庫存管理相結合，使貨物和資金達到更加便利地進行統計，進一步改善業務的會計準確性和人工成本。
3. 政府：在解決交易糾紛的過程中，可以提供更多可信的證據供參考。數字交易收據也可以解決紙本收據丟失或破損的問題。考慮到稅收問題，政府可以審查具有高信譽的商業交易細節作為稅收計算程序，以定制稅標準，減少許多稅收糾紛。

参考文献

- [1] Charlie Lee. *Litecoin Official website*, 2011. <https://litecoin.org>.
- [2] Billy Markus. *DogeCoin*, 2013-12. <http://dogecoin.com>.
- [3] Daniel Kraft. *Namecoin*, 2011-04. <https://namecoin.org>.
- [4] Sunny King. *Primecoin*, 2013-07. <http://primecoin.io/>.
- [5] Constantine Kryvomaz. *Ethereum Classic*, 2015-07. <https://ethereumclassic.github.io/>.
- [6] Ethereum Foundation Vitalik Buterin. *Ethereum*, 2015-07. <https://ethereum.org>.
- [7] *solidity*. <https://solidity.readthedocs.io/en/develop/>.
- [8] Gavin Wood. “*Ethereum: A secure decentralised generalised transaction ledger*”. *Ethereum Project Yellow Paper*, 2014, 151: 1–32.
- [9] Rainer Böhme, Nicolas Christin, Benjamin Edelman et al. “*Bitcoin: Economics, technology, and governance*”. *Journal of Economic Perspectives*, 2015, 29(2): 213–38.
- [10] *Cryptocurrency Market Capitalizations*. <https://coinmarketcap.com/all/views/all/>.
- [11] John Gregor Fraser and Ahmed Bouridane. *Have the security flaws surrounding BITCOIN effected the currency's value?*, 2017: 50–55.
- [12] Kyle Torpey. “*You Really Should Run a Bitcoin Full Node: Here's Why*”. *Bitcoin Magazine*, 2017.
- [13] 蔡怡杼. 银行员监守自盗手法有这些, 2017-10. <https://www.nownews.com/news/20171029/2633984>.
- [14] CM Adams and SE Tavares. *The use of bent sequences to achieve higher-order strict avalanche criterion in S-box design* [techreport], 1990.
- [15] *Bitcoin Improvement Proposals*. <https://github.com/bitcoin/bips/blob/master/README.mediawiki>.
- [16] *Western Union*. <https://www.westernunion.com/us/en/home.html>.
- [17] *PayPal*. <https://www.paypal.com>.
- [18] *Digibyte*. <https://www.digibyte.io>.
- [19] blockchain.info. *Blockchain Size*, 2018. <https://blockchain.info/charts/blocks-size?timespan=all>.
- [20] J Göbel and AE Krzesinski. “*Increased block size and Bitcoin blockchain dynamics*”. In: *Telecommunication Networks and Applications Conference (ITNAC), 2017 27th International*, 2017: 1–6.
- [21] Shen Noether and Sarang Noether. “*Monero is not that mysterious*”. *Technical report*, 2014.
- [22] Emmanuel Bresson, Jacques Stern and Michael Szydlo. “*Threshold ring signatures and applications to ad-hoc groups*”. In: *Annual International Cryptology Conference*, 2002: 465–480.
- [23] Ming Zhong. “*A faster single-term divisible electronic cash: ZCash*”. *Electronic Commerce Research and Applications*, 2002, 1(3-4): 331–338.

- [24] Uriel Feige, Amos Fiat and Adi Shamir. “Zero-Knowledge Proofs of Identity”. *J. Cryptology*, **1988**, 1(2): 77–94. <https://doi.org/10.1007/BF02351717>.
- [25] Thibault de Balthasar and Julio Hernandez-Castro. “An Analysis of Bitcoin Laundry Services”. In: *Nordic Conference on Secure IT Systems*, **2017**: 297–312.
- [26] Ayush Singh Panwar. “Asymmetric Key Cryptography”. *Browser Download This Paper*, **2014**.
- [27] Taher ElGamal. “A public key cryptosystem and a signature scheme based on discrete logarithms”. *IEEE transactions on information theory*, **1985**, 31(4): 469–472.
- [28] Andrew Miller and Joseph J LaViola Jr. “Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin”. Available on line: <http://nakamotoinstitute.org/research/anonymous-byzantine-consensus>, **2014**.
- [29] Don Johnson, Alfred Menezes and Scott Vanstone. “The elliptic curve digital signature algorithm (ECDSA)”. *International Journal of Information Security*, **2001**, 1(1): 36–63.
- [30] Dmitry Khovratovich, Christian Rechberger and Alexandra Savelieva; ed. by Anne Canteaut. “Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family”. In: *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*. Springer, **2012**: 244–263. https://doi.org/10.1007/978-3-642-34047-5_15.
- [31] Florian Mendel, Norbert Pramstaller, Christian Rechberger *et al.*; ed. by Sokratis K. Katsikas, Javier Lopez, Michael Backes *et al.* “On the Collision Resistance of RIPEMD-160”. In: *Information Security, 9th International Conference, ISC 2006, Samos Island, Greece, August 30 - September 2, 2006, Proceedings*. Springer, **2006**: 101–116. https://doi.org/10.1007/11836810_8.
- [32] The Bitcoin Core developers. *Base58*, **2009**. <https://github.com/bitcoin/bitcoin/blob/master/src/base58.cpp>.
- [33] *The Large Bitcoin Collider*. <https://lbc.cryptoguru.org>.
- [34] Tim Dierks. “The transport layer security (TLS) protocol version 1.2”. **2008**.
- [35] Android Developers Blog. *Some SecureRandom Thoughts*, 2013-08. <https://android-developers.googleblog.com/2013/08/some-securerandom-thoughts.html>.
- [36] bitcoin.org. *Android Security Vulnerability*, 2013-08. <https://bitcoin.org/en/alert/2013-08-11-android>.
- [37] BurtW. *Bad signatures leading to 55.82152538 BTC theft*, 2013-08. <https://bitcointalk.org/index.php?topic=271486.0>.
- [38] Lars R Knudsen, Vincent Rijmen, Ronald L Rivest *et al.* “On the design and security of RC2”. In: *International Workshop on Fast Software Encryption*, **1998**: 206–221.
- [39] Ron Rivest. “Rc4”. *Applied Cryptography by B. Schneier*, John Wiley and Sons, New York, **1996**.
- [40] Ronald L Rivest. “The RC5 encryption algorithm”. In: *International Workshop on Fast Software Encryption*, **1994**: 86–96.
- [41] RL Rivest, MJB Robshaw, R Sidney *et al.* *The RC6 block cipher. v1. 1, August 20, 1998*, **2016**.
- [42] Data Encryption Standard. “Data encryption standard”. *Federal Information Processing Standards Publication*, **1999**.

- [43] American Bankers Association *et al.* “*Triple Data Encryption Algorithm Modes of Operation*”. *ANSI X9*: 52–1998.
- [44] Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, **2013**.
- [45] Ronald L Rivest, Adi Shamir and Leonard M Adleman. *Cryptographic communications system and method*. Google Patents, 1983-9 20.
- [46] Neal Koblitz. “*Elliptic curve cryptosystems*”. *Mathematics of computation*, **1987**, 48(177): 203–209.
- [47] Nicholas Jansma and Brandon Arrendondo. “*Performance comparison of elliptic curve and rsa digital signatures*”. *nicj.net/files*, **2004**.
- [48] Léo Ducas, Alain Durmus, Tancrede Lepoint *et al.* “*Lattice signatures and bimodal Gaussians*”. In: *Advances in Cryptology–CRYPTO 2013*. Springer, **2013**: 40–56.
- [49] Scott Vanstone. “*Deployments of Elliptic Curve Cryptography*”. In: *the 9th Workshop on Elliptic Curve Cryptography (ECC)*, **2005**.
- [50] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*, **2008**.
- [51] Christian Decker and Roger Wattenhofer. “*Information propagation in the bitcoin network*”. In: *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*, **2013**: 1–10.
- [52] Ghassan O Karame, Elli Androulaki and Srdjan Capkun. “*Double-spending fast payments in bitcoin*”. In: *Proceedings of the 2012 ACM conference on Computer and communications security*, **2012**: 906–917.
- [53] Addy Yeow. *Global Bitcoin Node Distribution*. <https://bitnodes.earn.com/>.
- [54] *Bitcoin Testnet*. <https://en.bitcoin.it/wiki/Testnet>.
- [55] Hiroki Kuzuno and Christian Karam. “*Blockchain explorer: An analytical process and investigation environment for bitcoin*”. In: *Electronic Crime Research (eCrime), 2017 APWG Symposium on*, **2017**: 9–16.
- [56] Po-Wei Chen, Bo-Sian Jiang and Chia-Hui Wang. “*Blockchain-based payment collection supervision system using pervasive Bitcoin digital wallet*”. In: *Wireless and Mobile Computing, Networking and Communications (WiMob)*, **2017**: 139–146.

致謝

原本就對比特幣區塊鏈技術深感興趣的我，來到了北京大學攻讀工程碩士學位。在這段期間深怕著會因為科系的關係而影響到了我的研究方向，在導師雙選了劉京老師，老師相當支持我做自己的研究，後來也見到了李傑教授，大力鼓勵著我繼續往科研的方向前進，老師的宏亮的聲音、霸氣的指導深植我心。段莉華老師也相當支持我做學術研究，因為段老師也一度的前往北京大學的校本部探討密碼學的研究。

在台灣實習的我來到了台灣最高學術研究機構中央研究院資訊科學所繼續展開我的科研路，同時也延續著之前與銘傳大學王家輝老師合作的科技部計畫“比特幣監督收銀系統”，因為有著計畫的補助也使得在求學的路上獲得更充足的預算，也因為計劃上的補助，使我能夠順利地前往義大利羅馬參加 IEEE WiMob 會議發表論文“Blockchain-based payment collection supervision system using pervasive Bitcoin digital wallet.”^[56]，參加了台灣最大的計算機會議 TANET 發表論文“匿名加密貨幣與實名商家交易的有效行動支付監督平台之建置與實作-以比特幣為例”，也得到了 TANET 會議的最佳論文獎，也要對與我合作的最佳夥伴江柏憲同學，我們共創了大學時期專題研究的第一名，這次我們也一舉奪下了 TANET 的最佳論文，相信都在我們的人生道路中寫下了嶄新的一頁。感謝李開暉教授願意教導我並讓我在其旗下做科學研究，同時給予我最大的資源與協助，並總是指引我研究方向。

除了在諸位教授的諄諄教誨下使我有機會完成這篇論文外，也要誠摯的感謝於二零一四年帶我認識比特幣的啟蒙老師楊哲豪先生，沒有他的教導無法成就至今已多達三萬四千人的比特幣中文社群之社群，也不會給予我有這樣的機會了解比特幣的運作原理，通過所學與社群間交流種種架構出我現在的區塊鏈產業概念，成為我在區塊鏈科技產業發展之道路最重要的基石。

北京大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名： 日期： 年 月 日

学位论文使用授权说明

(必须装订在提交学校图书馆的印刷本)

本人完全了解北京大学关于收集、保存、使用学位论文的规定，即：

- 按照学校要求提交学位论文的印刷本和电子版本；
- 学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务，在校园网上提供服务；
- 学校可以采用影印、缩印、数字化或其它复制手段保存论文；
- 因某种特殊原因需要延迟发布学位论文电子版，授权学校在一年/两年/三年以后在校园网上全文发布。

(保密论文在解密后遵守此规定)

论文作者签名： 导师签名： 日期： 年 月 日