區塊鏈應用實例

報告人 湯耀翔

區塊鏈是什麼

- 區塊鏈是一種透過共識算法實現去中心化的技術
- 區塊鏈可以用於有價值的商品上如:房子、汽車等
- 區塊鏈提供了在鏈上成員可以即時檢閱、共享資訊
- 區塊鏈的特性與技術增加了客戶的信任度。

共識機制

共識演算法是區塊鏈的核心制度之一，由於節點之間是分散且平行的，因此必須設計一套制度，來維護系統的運作順序與公平性，統一區塊鏈的版本。

- 工作量證明(Proof of Work, PoW)
- 權益證明(Proof of Stake, PoS)
- 權威證明(Proof of Authority, PoA)

創世區塊(Genesis Block)

區塊鏈上的第一個區塊，創建者可以自己設定創世區塊的所有值與參數，只要沒有人跟你設定相同的參數，那你的區塊鏈就是獨一無二的。

```
{
  "config": {
        "chainId":1997,
        "homesteadBlock": 0,
        "eip150Block": 0,
        "eip155Block": 0,
        "eip158Block": 0
    },
  "alloc"      : {},
  "coinbase"   : "0x0000000000000000000000000000000000000000",
  "difficulty" : "0x20000",
  "extraData"  : "0x74686520726f73657320696e20686572206861e642c2074686520666c61766f7220696e206d696e65",
  "gasLimit"   : "0x44C747",
  "nonce"      : "0x00000000c000ff59",
  "mixhash"    : "0x0000000000000000000000000000000000000000000000000000000000000000",
  "parentHash" : "0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp"  : "0x599DA33A"
}
```

創世區塊(Genesis Block)

參數的解釋

Aa Name	Description
chainID	區塊鏈的識別ID
homesteadBlock	這是以太坊的第二個主要版本，第一個是 Frontier，這個值設為"0"表示目前正在使用 Homestead 版本
eip155Block	eip 是 ethereum improvement proposal 的縮寫，私有鏈不會因為這些提議分岔，因此設為"0"
eip158Block	eip 是 ethereum improvement proposal 的縮寫，私有鏈不會因為這些提議分岔，因此設為"0"
mixhash	與 nonce 配合用於挖礦，由上一個區塊的一部分產生的雜湊
nonce	一個 64 位隨機數，用於挖礦
difficulties	挖礦難度參數
alloc	預置帳號與該帳號內金額
coinbase	預設為第一個建立帳號的礦工
timestamp	設置創世區塊的時間戳
parentHash	上一個區塊的 Hash 值，因為是創世區塊所以為 0
extraData	附加信息
gasLimit	交易所使用到的 gas 總量限制，用來限制區塊能包含的交易信息總和，因為我們是私有鏈，所以填最大。

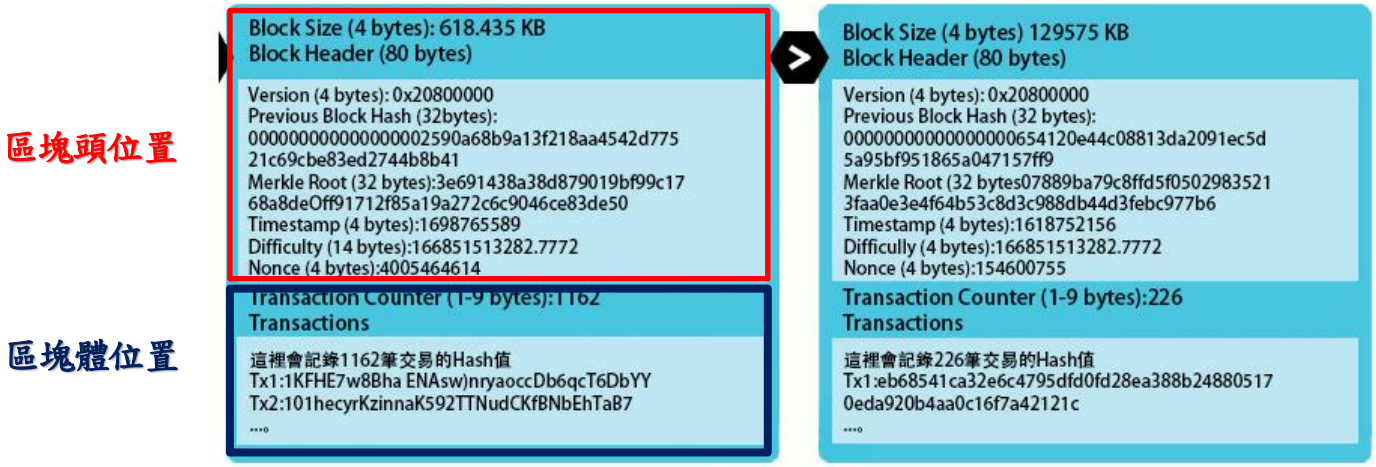
資料來源: https://medium.com/samumu-clan/%E7%94%A8-geth-%E6%9E%B6%E8%A8%AD%E7%A7%81%E6%9C%89%E9%8F%8841a2baa0efd8

區塊鏈中的區塊

一個完整的區塊中所包含的內容會有兩個部分

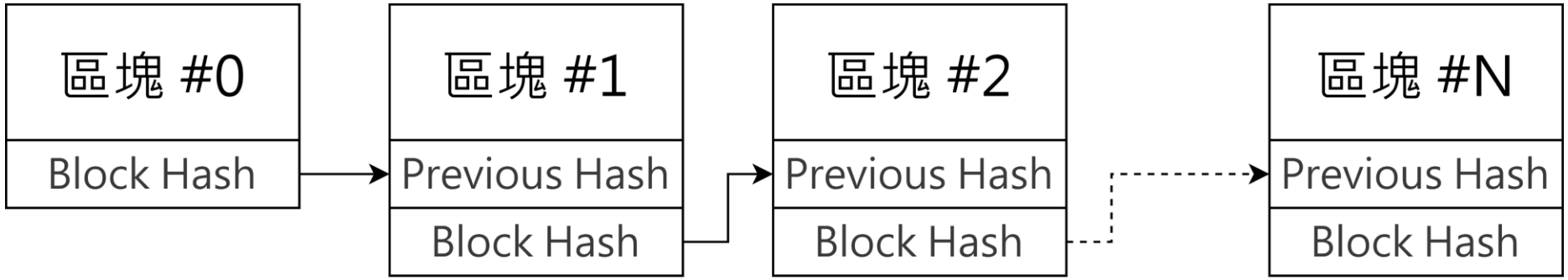
1) 區塊頭（Block Header）

2) 區塊體（Block Body）



區塊+鏈

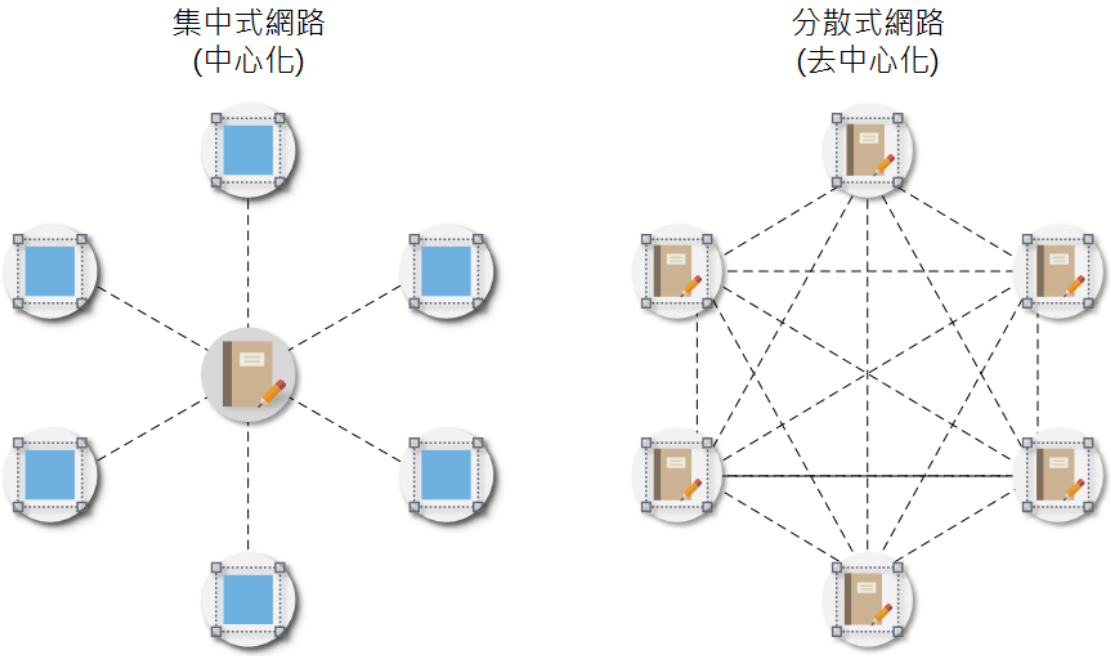
每個區塊中透過區塊頭裡面的Previous Hash來形成一個鏈結串連每一個區塊形成區塊鏈。



區塊鏈技術與特色

- **去中心化的分散式電子帳本**

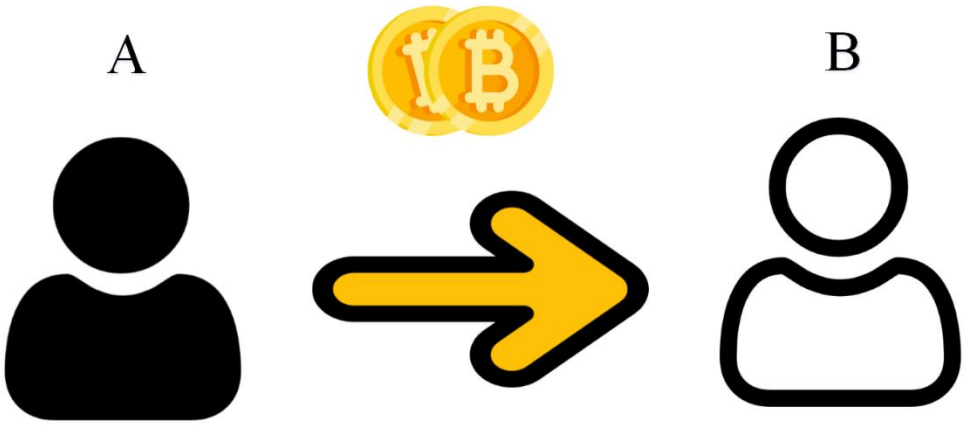
帳本是由一群節點(電腦)共同維護，而不是由一個類似銀行的中心機構來掌管，以此達到去中心化的概念。而每一個節點都會有一本相同的帳本，記錄著過去所有交易的內容。



區塊鏈技術與特色

- 所有節點都以匿名方式共同驗證整份帳簿

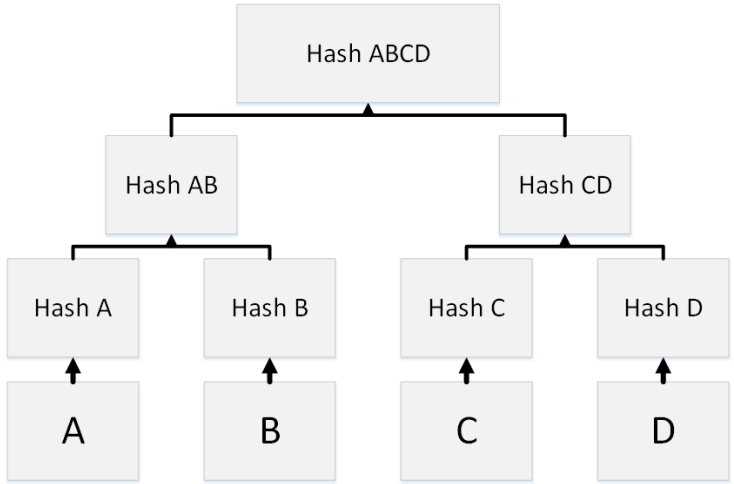
節點會使用一組「英文+數字」的代碼作為名稱，我們可以稱作為地址(Address)
可以不需要使用真實名字便可進行交易。



區塊鏈技術與特色

- 每份歷史記錄都不易被篡改

區塊鏈中的每一筆資料一旦寫入就不能再更改，只要資料被驗證完後就會永久的寫入區塊中，其中的技術是透過 Hash Function 對其內容進行加密動作。



以太坊 - 智能合約

智能合約是一個運行於區塊鏈上的一串代碼，在以太坊的網路中有許多節點，這些節點除了挖礦外，也要負責運行以太坊上的智能合約，這項工作運行在每一個礦工的以太坊虛擬機(Ethereum Virtual Machine, EVM)上。

其智能合約經過編譯後會產生Bytecode與ABI兩個部分

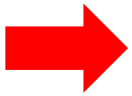
1) ABI: 在部署智能合約的時候，需要藉由ABI當作媒介傳遞Data

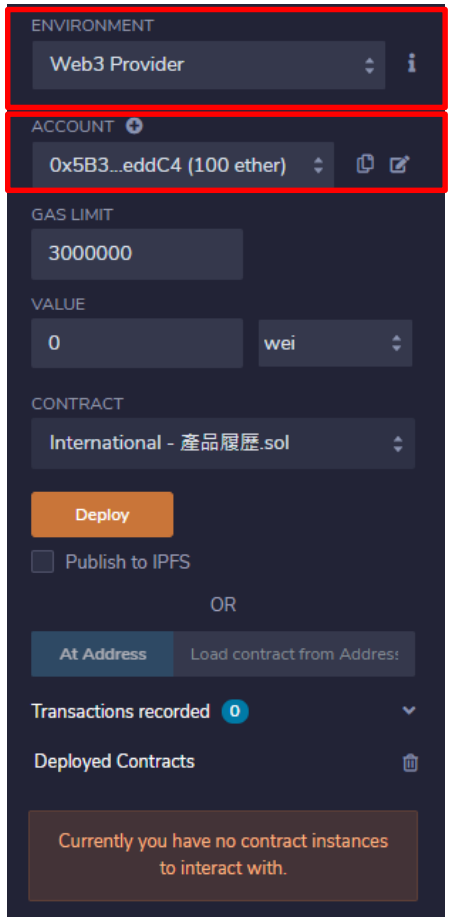
2) Bytecode: 讓虛擬機可以運行的程式碼

以太坊 - 智能合約

智能合約編譯完成後，可以連接自己所建立的私有鏈，並且完成部署。

```
pragma solidity ^0.4.25;
pragma experimental ABIEncoderV2;
contract International {
    enum Station { Fishingboat,Wholesale,Processingplant,LogisticsCenter,Massmerchandiser }
    //訂單位置
    struct transportpath {
        uint index;//訂單順序
        Station state;//
    }

    //transportpath[] public Path;
    mapping (uint => transportpath) public Path;
    //漁產品履歷
    struct data{
        string transName;//魚的名稱
        string[5] reason;//特殊原因； string[5]預設5站
        uint[5] transTime;
        uint weight;//重量
        string Location;//產地
        //string enterLocationtime;//進站時間
        //string LeaveLocationtime;//出站時間
        Station state;
    }
    mapping (uint => data) public datas;
    uint[] internal fishList;

    //event A1(data a1);
    event LogFish( uint _number, string transName, string reason, uint weight, string Location);
    event LogShipping(uint _number, Station state, uint time);

    function get_datas(uint _number) public view returns(string[5],uint[5]) {
        return (datas[_number].reason, datas[_number].transTime);
    }
    function goto_Fishingboat(uint _number, string _transName, string _reason, uint _weight, string _Location) public{
        transportpath storage entry = Path[_number];
        if(entry.index > 0){
            return;
        }else{
            fishList.push(_number);
            entry.index = fishList.length;
            entry.state = Station.Fishingboat;

            datas[_number].transName = _transName;
            datas[_number].reason[uint(Station.Fishingboat)] = _reason;
            datas[_number].transTime[uint(Station.Fishingboat)] = now;
            datas[_number].weight = _weight;
            datas[_number].Location = _Location;
            //LogFish(_number,_transName,now,_weight,_Location);
        }
    }
```

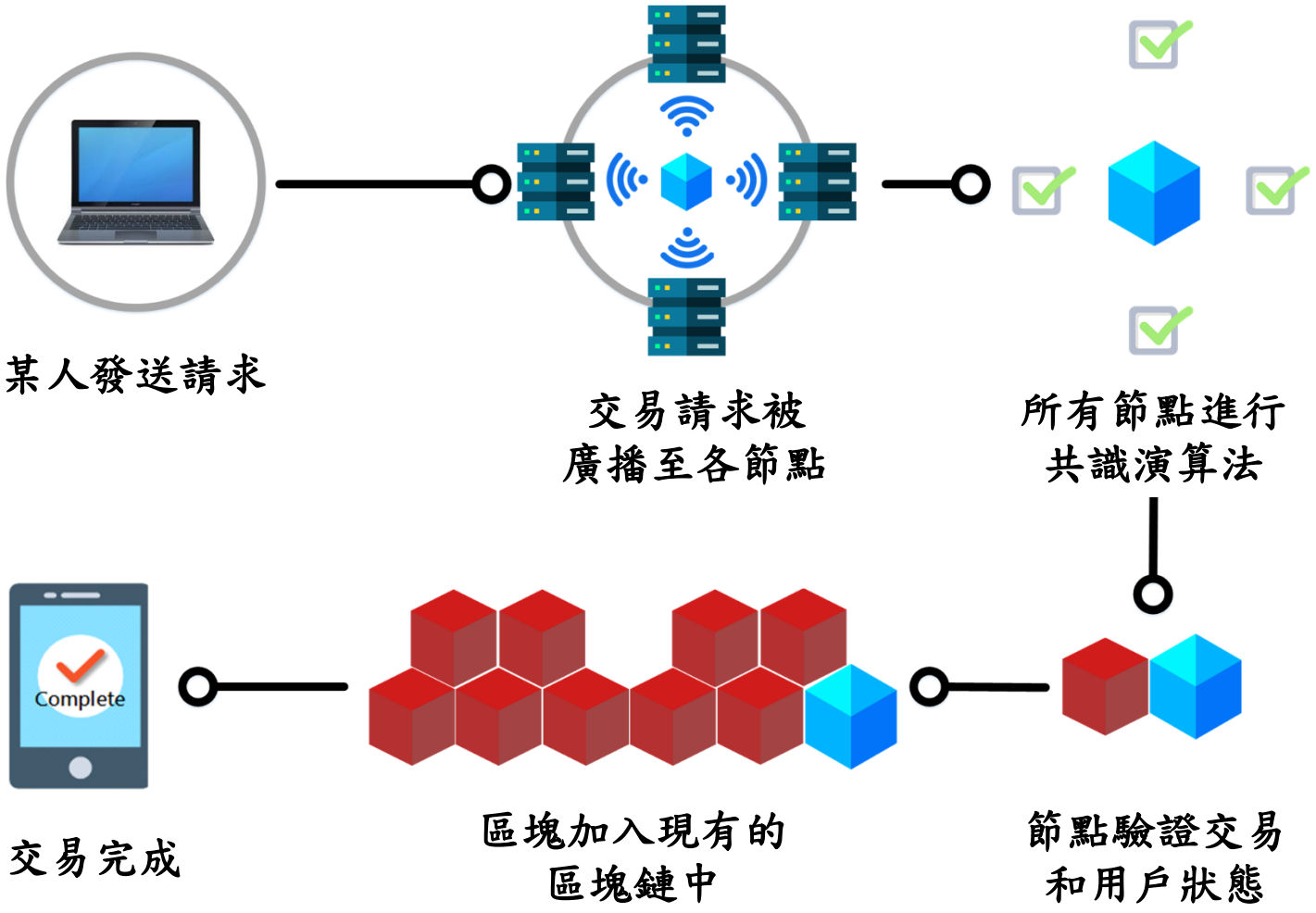




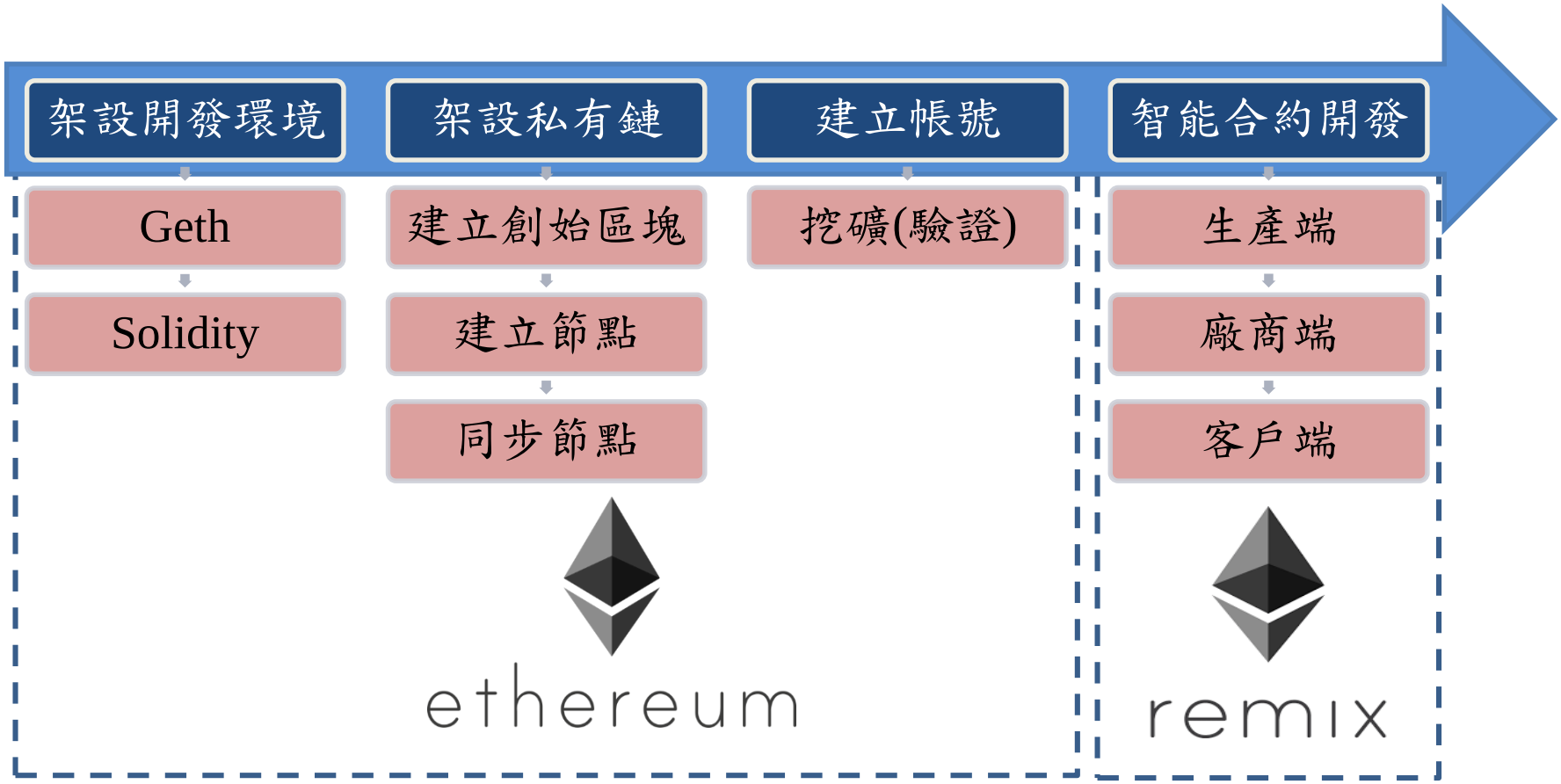
本機端

部署的帳戶

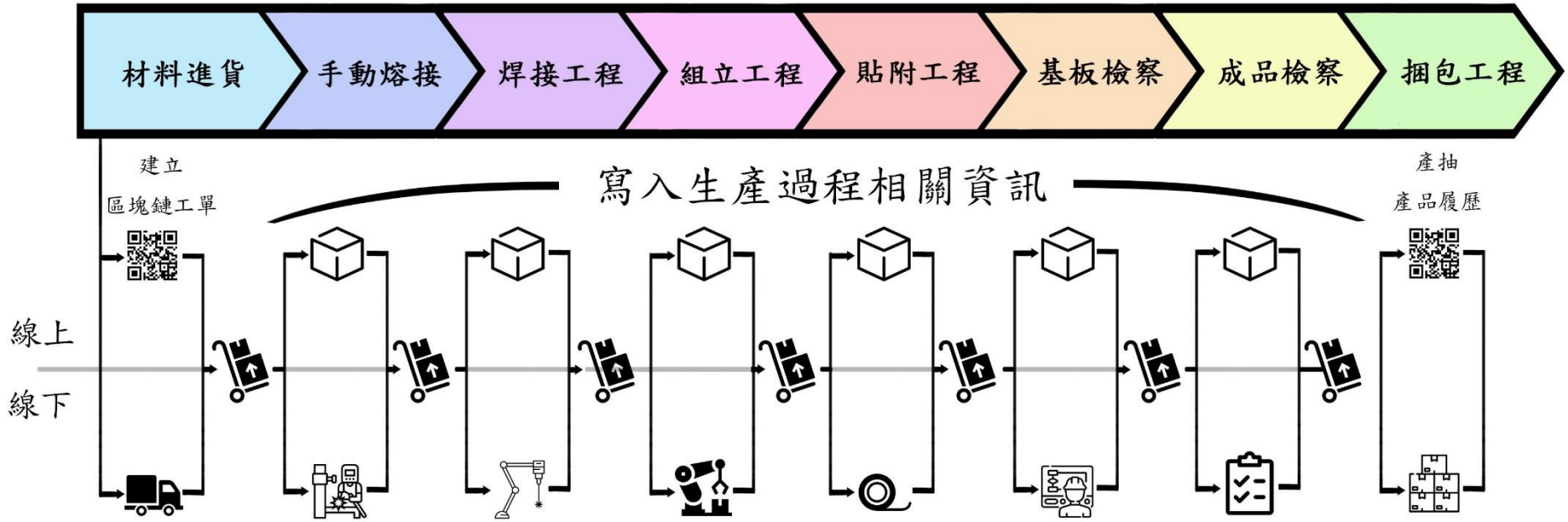
區塊鏈運作流程


某人發送請求
交易請求被
廣播至各節點
所有節點進行
共識演算法
節點驗證交易
和用戶狀態
區塊加入現有的
區塊鏈中
交易完成
Complete

區塊鏈的開發流程


架設開發環境
架設私有鏈
建立帳號
智能合約開發
Geth
Solidity
建立創始區塊
建立節點
同步節點
挖礦(驗證)
生產端
廠商端
客戶端
ethereum
remix

廠內生產線流程置入區塊鏈示意圖

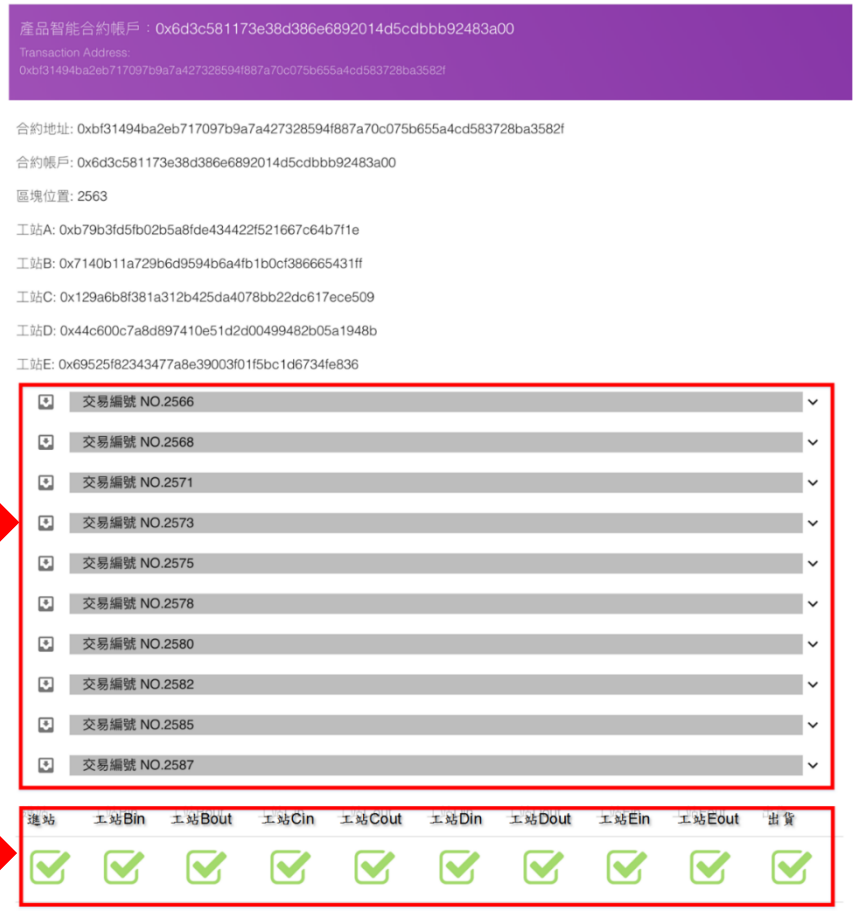


作業員紀錄之生產資訊

紀錄生產訊息識別編號

生產線狀態

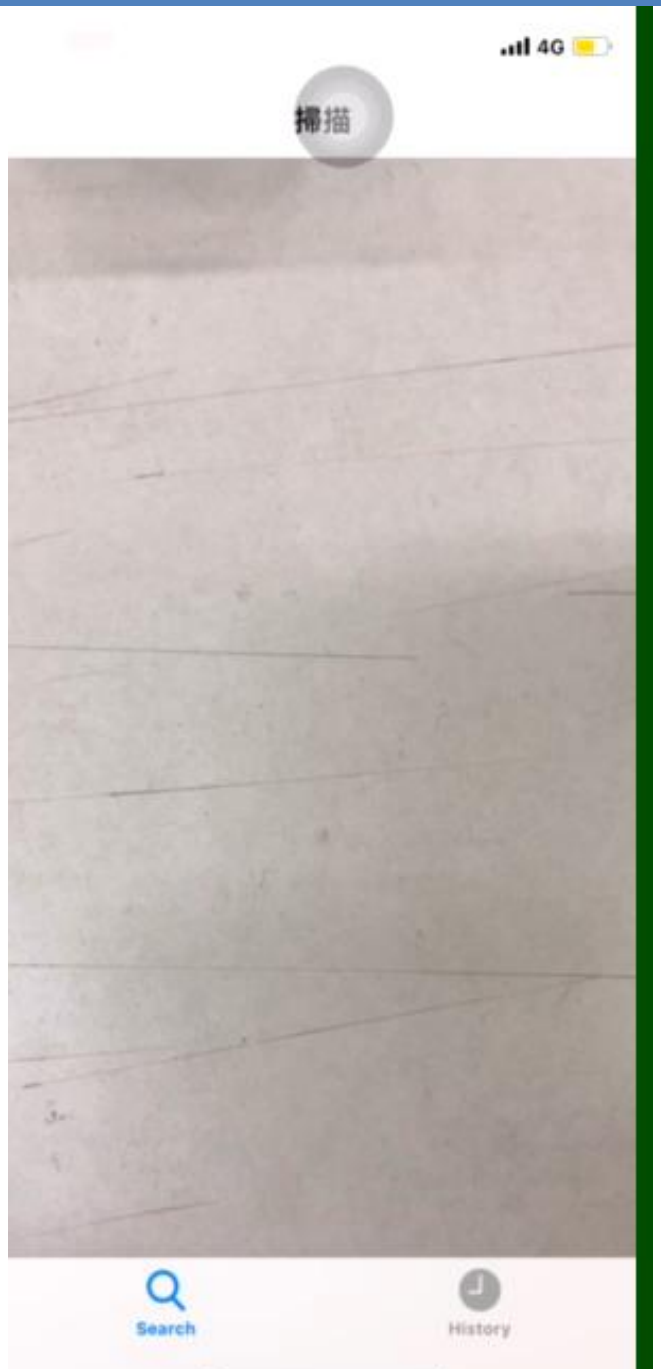
產品生產履歷



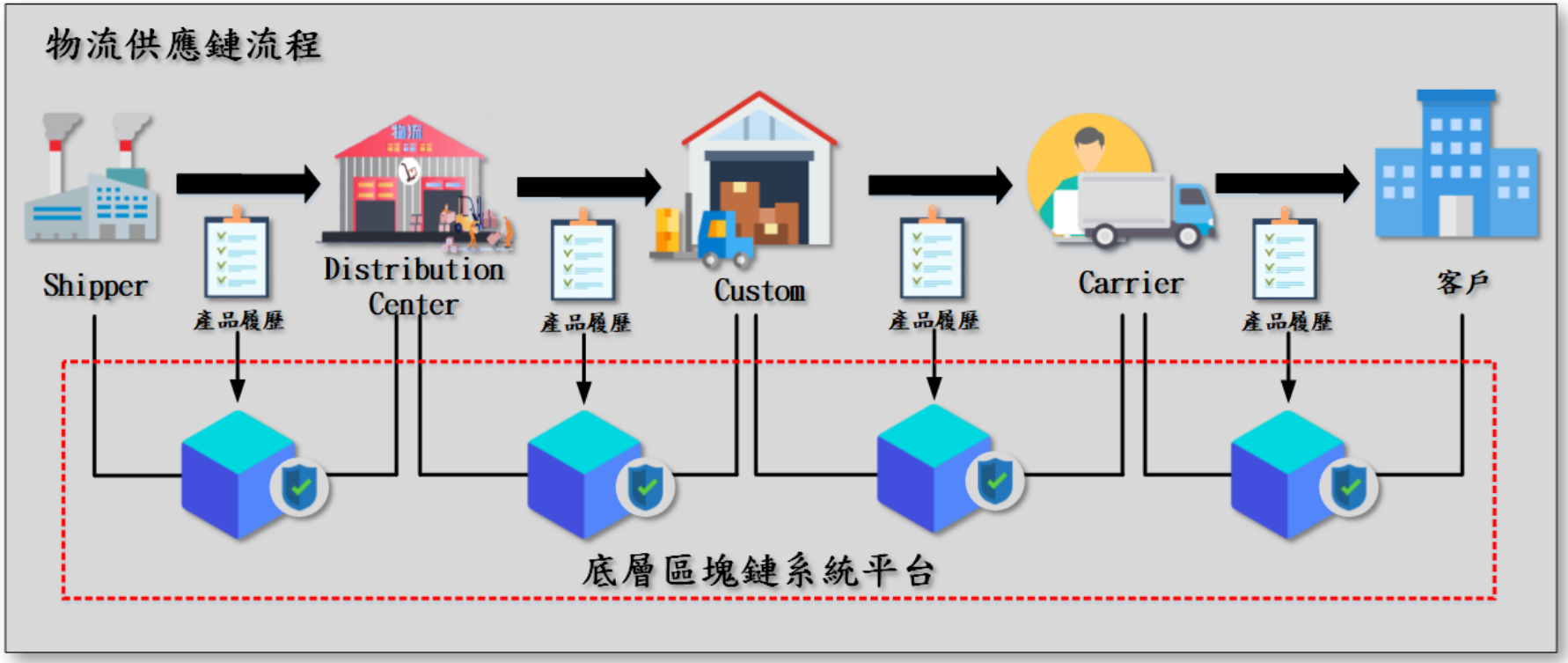
產線紀錄訊息

生產線狀態

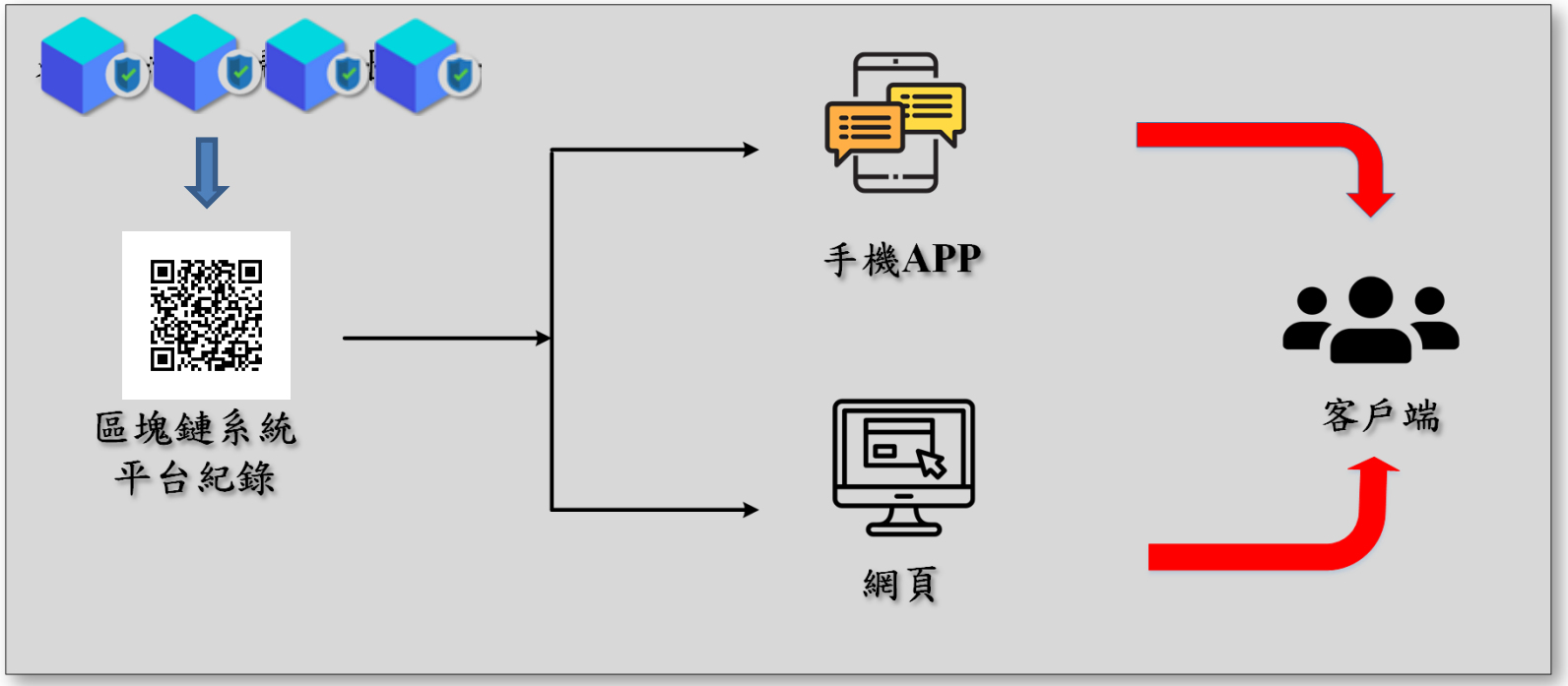
產品生產履歷實測



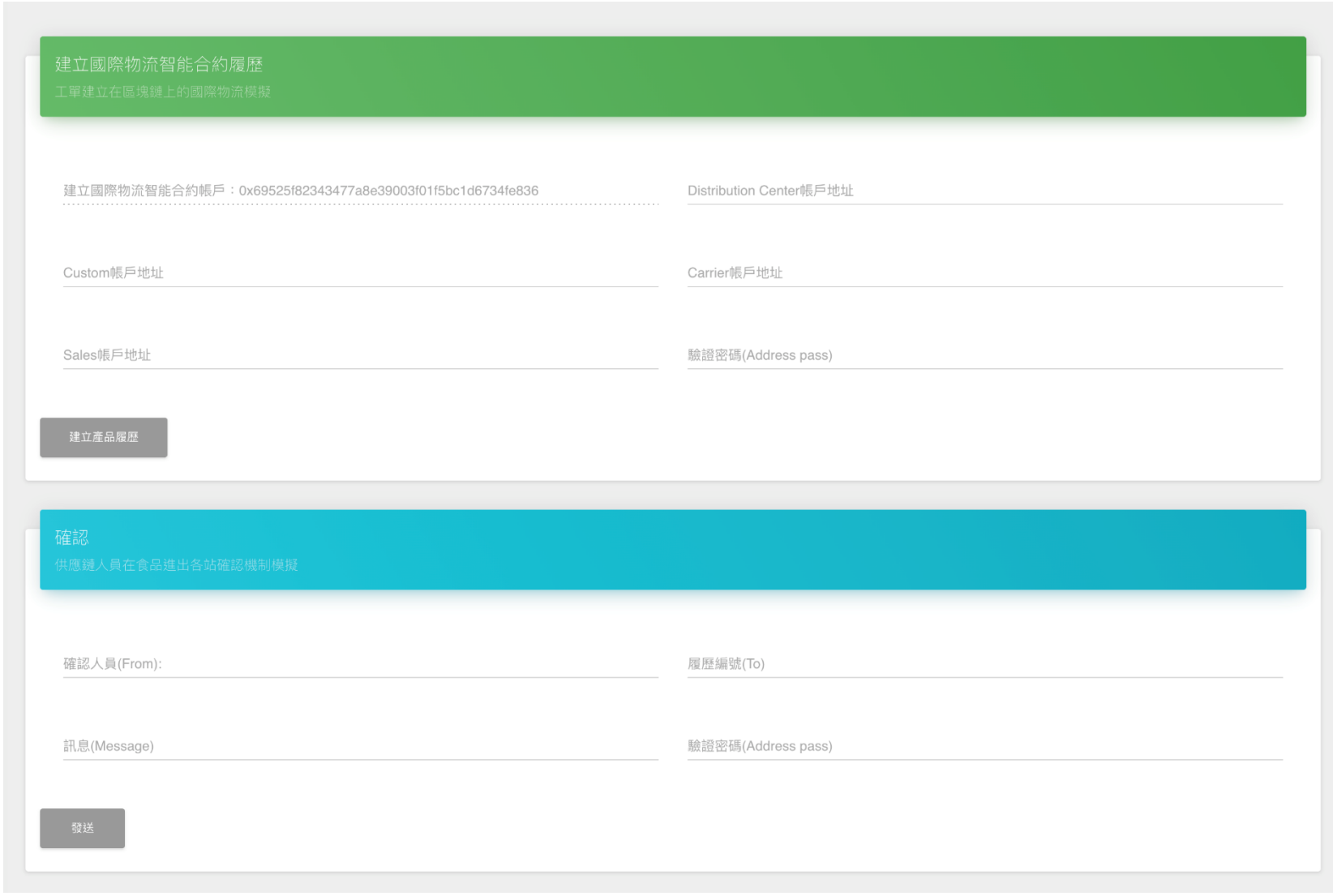
國際物流導入區塊鏈示意圖



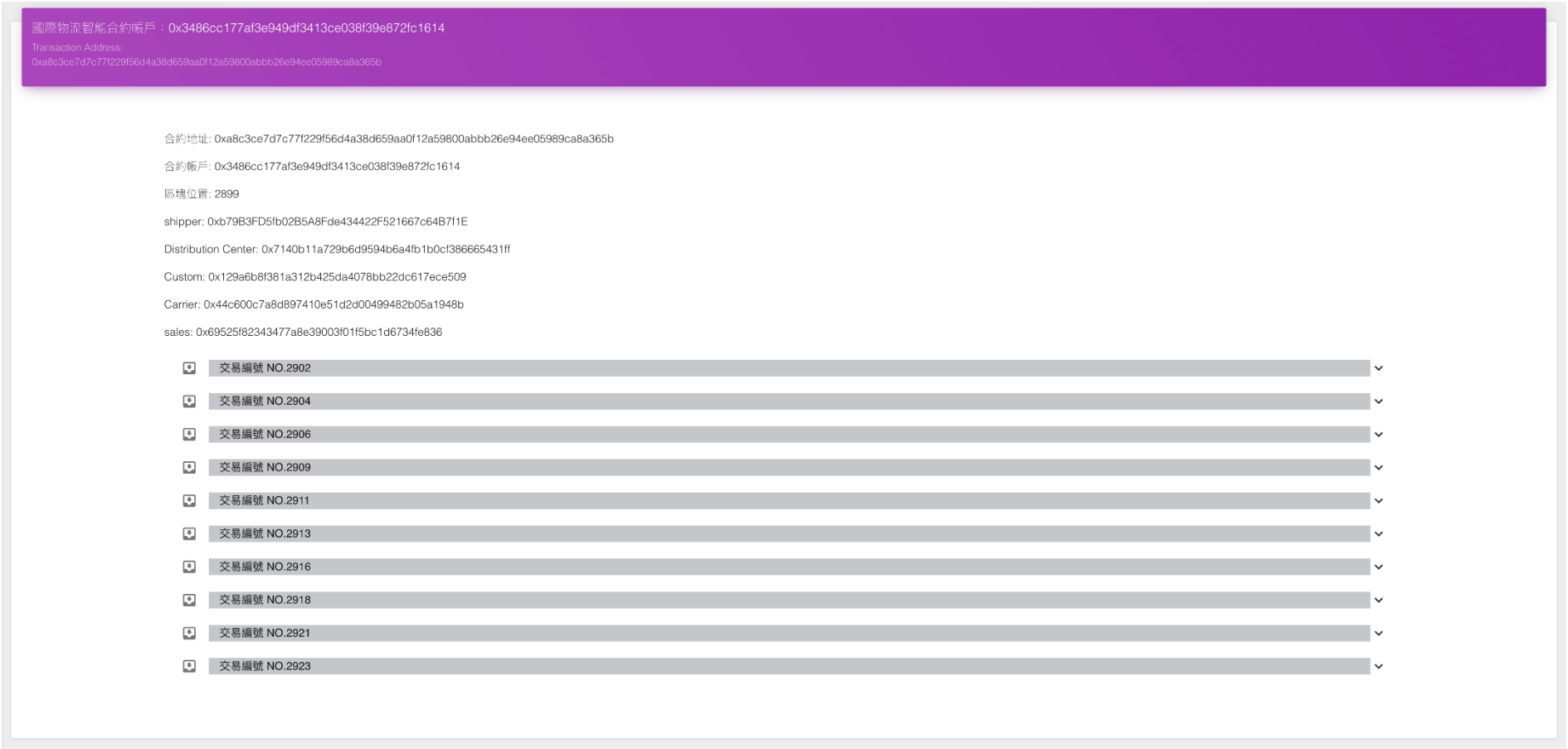
區塊鏈系統透過使用者介面將資訊傳遞給客戶端示意圖



國際物流供單建立


建立國際物流智能合約履歷
工單建立在區塊鏈上的國際物流模擬
建立國際物流智能合約帳戶：0x69525f82343477a8e39003f01f5bc1d6734fe836
Distribution Center帳戶地址
Custom帳戶地址
Carrier帳戶地址
Sales帳戶地址
驗證密碼(Address pass)
建立產品履歷
確認
供應鏈人員在食品進出各站確認機制模擬
確認人員(From):
履歷編號(To)
訊息(Message)
驗證密碼(Address pass)
發送

國際物流資訊紀錄


國際物流智能合約帳戶：0x3486cc177af3e949df3413ce038f39e872fc1614
Transaction Address:
0xa8c3ce7d7c77f229f56d4a38d659aa0f12a59800abbb26e94ee05989ca8a365b
合約地址: 0xa8c3ce7d7c77f229f56d4a38d659aa0f12a59800abbb26e94ee05989ca8a365b
合約帳戶: 0x3486cc177af3e949df3413ce038f39e872fc1614
區塊位置: 2899
shipper: 0xb79B3FD5fb02B5A8Fde434422F521667c64B7f1E
Distribution Center: 0x7140b11a729b6d9594b6a4fb1b0cf386665431ff
Custom: 0x129a6b8f381a312b425da4078bb22dc617ece509
Carrier: 0x44c600c7a8d897410e51d2d00499482b05a1948b
sales: 0x69525f82343477a8e39003f01f5bc1d6734fe836
交易編號 NO.2902
交易編號 NO.2904
交易編號 NO.2906
交易編號 NO.2909
交易編號 NO.2911
交易編號 NO.2913
交易編號 NO.2916
交易編號 NO.2918
交易編號 NO.2921
交易編號 NO.2923

國際物流資訊紀錄

每個站別紀錄訊息並寫入區塊鏈中

把資訊寫入QR code中

實測

- 將產銷履歷訊息寫入json server 再利用QR code的方式呈現，藉由我們自己開發的APP來讀取資料。