AlphaWallet區塊鏈門票測試報告

2018俄羅斯世界杯ERC875門票

AlphaWallet

二零一八年八月

作者簡介

AlphaWallet 張中南

AlphaWallet(Stormbird Pte. Ltd.)聯合創始人兼CEO。中國籍澳大利亞居民。超過10年的票務市場商務經驗，在亞太各國（中國，澳大利亞，新加坡，日本，韓國，馬來西亞）管理跨國團隊和企業超過7年。成功幫助360Experience（Ticketbis 3 個業務部門之一）進入亞太市場，從零開始到2016年被eBay以1.65億美金整體（Ticketbis）收購。連續創業者，在澳大利亞，中國北京，中國香港和新加坡創辦過5個公司。技術愛好者，2014年開始接觸區塊鏈，曾嘗試用區塊鏈技術重構熱門活動門票市場，持有少量比特幣和以太幣。詠春和拳擊愛好者。

SHANKAI 盛开体育 鄭來

現任盛開體育副總裁，2010/2014/2018 三屆世界杯官方款待計劃中國區負責人，曾經就職中國中旅集團負責公民出境業務，中信集團國際金融控股集團有限公司負責旅遊，航空和體育板塊的投資業務。

SHANKAI 盛开体育 高東東

盛開體育IT經理。曾負責運營多場國際大賽票務管理（2014-2018世界杯、2016年歐洲杯、歐冠、巴薩俱樂部等）。2016年接觸區塊鏈，並從2017年開始與 AlphaWallet(Stormbird Pte. Ltd.)緊密合作，以國際頂級賽事門票為基礎嘗試新型互聯網技術，一直致力於通過票務通道打開國際賽事的中國大市場，推動中國體育事業的快速發展。

目錄

背景介紹

ALPHAWALLET

AlphaWallet (Stormbird Pte. Ltd.) 是一家區塊鏈創業公司，主要專注在Layer 2，Offchain的區塊鏈協議開發，還有消費者終端應用平台的開發，從應用的角度入手提升區塊鏈的可用性，性能和隱私。AlphaWallet手機應用是面向普通消費者的以太坊智能合約調用工具以及協議運行平台。ERC 875是服務於真實商用案例的不可替代性通證（non-fungible token）標準，開發人員和企業可以非常容易的用ERC 875 Token來指代物理或數字世界內的人/事/物/權，並實現高效的原子化交易。

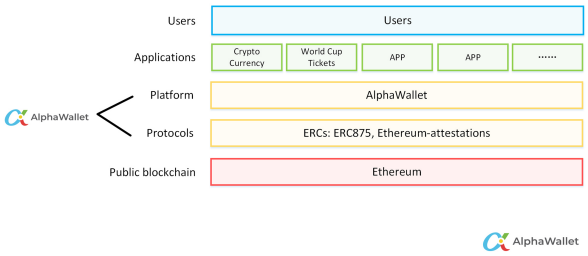


圖1 AlphaWallet平台結構

盛開體育

北京盛開國際旅行社有限公司，是一家快速向前發展的年輕體育營銷公司，為國內外的投資者提供體育商業運營方案。我們對於本土市場有著深入的瞭解掌握，並有著特殊的國內、外國際人脈網絡關係。我們擁有豐富的從事大型國際體育賽事的經驗，充分瞭解國際體育賽事參與者的需求。充分利用體育作為交流平台幫助中國品牌不斷成長，我們將世界頂級賽事引進中國並通過數字媒體技術讓中國的體育愛好者可以近距離接觸到更多的精彩體育項目。

一級市場

活動主辦方監管下的官方授權的渠道組成的市場如主辦方自己的官方銷售網站，大麥，永樂等官方授權票務平台和其他官方授權銷售機構。

二級市場

非主辦方官方授權的銷售渠道組成的市場包括個人專業票販子，P2P市場平台等。

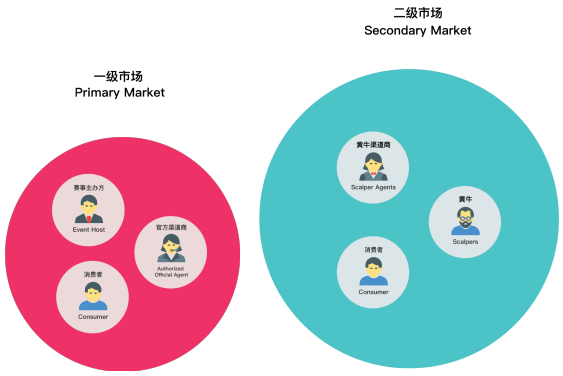


圖2 一級市場和二級市場劃分

區塊鏈技術

區塊鏈是一個分布式賬本技術，能夠保證「賬本」內數據所代表的信息不容易被篡改，進而實現相應信息代表的事物所有權的唯一性。同時公有鏈因為是一個公開賬本，沒有准入門檻，所以能實現所有權的點對點自由流通。隨著以太坊的興起，在「記賬」的同時還能支持智能合約實現各種流通邏輯。

現階段區塊鏈技術還處在非常早期階段，在這個階段，除實驗性質的項目外，真正適合落地商業化的應用場景非常少。現階段適合放上區塊鏈的數據需要滿足以下標準：

1. **數據所代表的信息是有高價值的**。如錢，重要的權益等等。

2. **數據所代表的信息在使用（體現價值）的時候需要被證明**。如使用支付寶支付商家的時候，需要支付寶/銀行/清算系統來證明「資金」的流轉。

3. **相應信息所代表的事物所有權，有流通轉讓變更的使用場景**。如各種兌換券等。

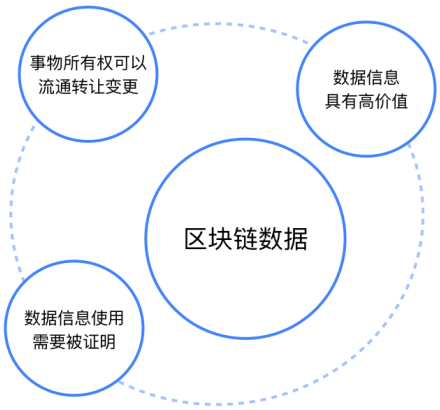


圖3 區塊鏈數據需要滿足的標準

現階段，如果數據不能同時滿足1、2、3，那麼有比區塊鏈好很多的其他技術來滿足相應需求。最典型的反例如身份信息，符合1和2，但是基本不存在轉讓所有權的使用場景，使用現有的成熟的認證（Attestation）技術可以更好的滿足使用需求。又例如各種存證類應用等，區塊鏈是「賬本」不是數據庫，不是用來存儲各種所有權不需要轉讓的「證據」的。

當前票務市場的問題和區塊鏈能帶來的變化

二級市場及票務造假

二級市場一直以來都是讓各方既愛又恨的存在，在大部分熱門項目上，二級市場無論參與人數還是資金規模都比一級市場大很多。

受影響方	存在的問題
主辦方	很難從二級市場獲得應得的好處 難以獲取市場數據但仍需要支付高額手續費 二級市場負面信息影響主辦方品牌和聲譽
消費者	二級市場無監管（大量黃牛票、假票，以及存在囤積行為） 導致消費者利益無法得到保障
官方授權的銷售渠道	黃牛搶購囤積門票對官方授權銷售渠道造成負面影響
二級市場銷售渠道	缺乏官方授權，信任缺失 回款週期長 支付高額手續費給提供擔保的市場（擔保市場需要承擔巨額風險）

表1 當前票務市場存在的問題

當前票務市場的問題

對於主辦方　二級市場加大了活動的傳播範圍是很好。但同時因為二級市場完全不受主辦方的監管，所以主辦方沒有辦法更進一步的從二級市場拿到應得的好處。沒有辦法拿錢，也沒有辦法向贊助商證明二級市場的傳播規模到底有多大。一級市場經常被大型渠道商壟斷，主辦方拿不到數據還要支付高額手續費。各種二級市場負面信息，如假票，天價票等等影響主辦方品牌和聲譽。

對於消費者　二級市場的存在滿足了部分消費者的需求，同時因為二級市場完全無監管，消費者的利益沒有辦法得到保障，各種天價票和假票充斥著二級市場，各種囤積行為導致消費者買不到票，場地內卻還有空座。

對於官方授權的銷售渠道　二級市場很多時候是他們處理庫存的必須渠道，但同時對於熱門活動，大

量黃牛搶購囤積門票，也對官方授權的銷售渠道造成非常不好的影響。

對於二級市場銷售渠道　二級市場銷售渠道為整個票務市場注入了更多的流動性，同時為消費者提供了更高額便利性。但是二級市場的賣家因為沒有官方授權背書，所以需要為了得到消費者的信任而投入大量成本，超長的回款週期，支付高額的手續費給提供擔保的市場如Viagogo，Stubhub，淘寶等。提供擔保的市場同樣承擔著巨額的擔保風險。

區塊鏈能夠帶來的變化

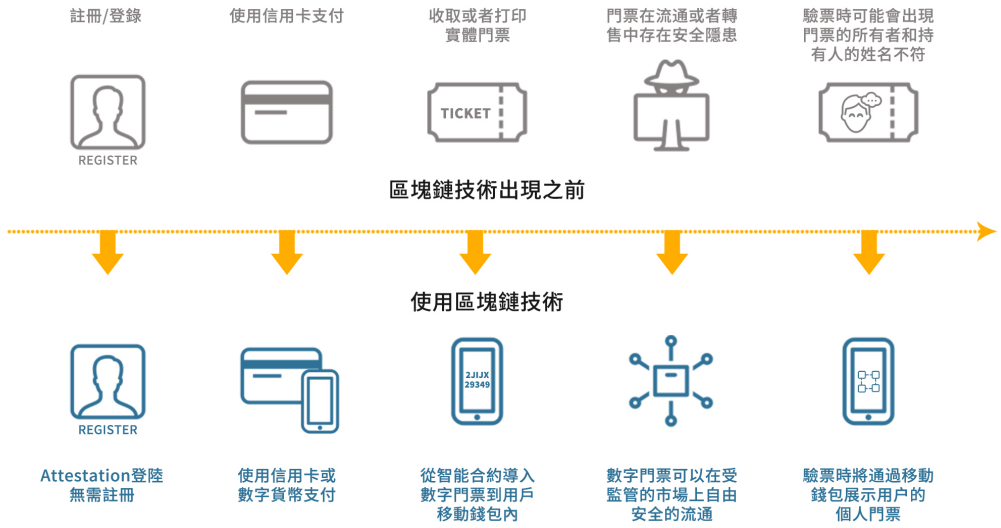


圖4 區塊鏈給消費者帶來的便利

對於主辦方

- 可在在智能合約內定義各種規則來實現對一二級市場的監管。比如票A只允許轉售5次，轉售價格不得超過100，每次轉售最高利潤不得超過10%，不超過5%的部分FIFA不動，超過5%的部分返給FIFA裡面的50%等；
- 可以直接採集到二級市場規模的具體數字，門票流通的全部過程可監可控；
- 可以直接從每一筆二級市場的交易中獲利，按規定的比例自動返回主辦方錢包；
- 可以直接連接到每一個持票用戶，知道每一個用戶的錢包地址。

對於消費者

監管下的二級市場會更加健康，並且保持自由提供足夠流動性。

- 票價變低；

- 不會再買到假票；
- 隨時能買到票；
- 享受數字化帶來的便利；
- 一些額外好處，比如持票在其他地方消費打折等。

對於官方授權的市場平台

- 各種銷售平台市場沒機會再壟斷數據和資金流了。數據代表的信息所有權歸主辦方和消費者，主辦方在消費者允許的情況下掌控全部數據。資金流在貨款對付的原子化交易下，不再經過銷售平台市場，直達主辦方或者其他門票銷售方。
- 銷售平台市場的功能簡化到：市場宣傳和撮合交易，不再提供結算和數據整理分析。

對於官方售票機構

- 可以更容易的拓展自己的銷售渠道，因為區塊鏈門票的流通渠道沒有准入許可，任何機構都可以加入到系統幫忙售票，不再需要不用系統之間的集成；
- 之前依靠用戶數據部分的收入降低；
- 之前依靠收單結算部分的收入降低。

對於二級市場平台方

- 類似於官方授權市場，二級市場平台方也沒機會再壟斷數據和資金流了；
- 票款對付的原子化交易徹底替代掉了二級市場平台方的擔保功能，同時迫使二級市場平台方大幅降低收費比率（現階段每一個交易15-30%的交易額用來支付二級市場平台方的服務費）。

對於二級市場內售票機構

- 不再需要中介平台方提供信用擔保，大幅擴大銷售面，降低交易門檻，大幅提高客戶數量；
- 票款對付的原子化加一，及時收賬；
- 受主辦方監管，所以無法再賺取單張超高額暴利；
- 總的來看，變得薄利多銷。

支付欺詐

支付欺詐是所有涉及收單結算業務的公司不可避免的一個大問題，大量的信用卡盜刷，假信用卡，信用卡爭議等等造成商家巨額損失。對於活動門票行業來說，這個需要兩部分同時支持：區塊鏈門票加上區塊鏈貨幣（加密貨幣）。只有兩者同時使用的情況下，才能實現原子化的貨款對付，徹底杜絕支付欺詐。

活動安全KYC

這部分並不是區塊鏈門票帶來的直接好處，而是說數字化門票可以帶來的好處，區塊鏈是數字化門票的最好解決方案。這裡主要會用到認證技術（Attestation），這個技術最好落地位置是在用戶端的客戶端（加密錢包）上面。客戶端通過自我生成或者導入第三方認證機構提供的Attestation（如用戶自己通過手機短信認證手機號碼，或如政府提供的生份證，護照等），智能合約和網站內置校驗Attestation的方法。智能合約和網站在不同用戶提交不同的Attestation時，給予不用的反饋。最終實現類似於：

- 任何不在黑名單上的用戶都可以持有門票A；
- 只有中國用戶可以持有門票B；
- 只有出生在1958年之前的用戶可以持有門票C；
- 只有用戶XXX可以持有門票D；
- 只有用戶XXX可以持有門票D，而且在需要檢票入場的時候，需要用戶XXX的生物特徵來解鎖用戶加密錢包內的門票D。

通過把門票綁定部分用戶信息，可以實現各種級別的KYC。通過嚴格的KYC可以大幅提高活動安全性。

監管下的自由流通

在公有鏈上開發智能合約發行token

- 公有鏈幫助token確權，能讓token實現無准入許可的自由流通。
- 智能合約能提前制定每一個token的流通/使用規則。

兩者結合實現監管下的自由流通，下面來具體說一說：

在智能合約內，token可以代表各種事/物/權，可以是數字世界內的事/物/權，也可以是物理世界內的事/物/權。為了方便理解，可以把token按照代表的事/物/權屬性，分成兩種類型：

- **token物理 - token代表物理世界內的事/物/權**
- **token數字 - token代表數字世界內的事/物/權**

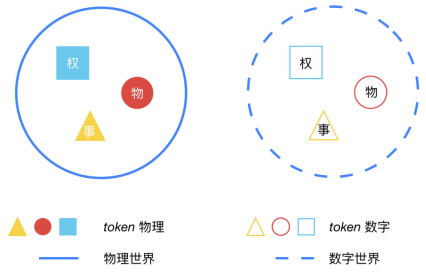


圖5 token物理和token數字

token的流通和「使用」方式的四種方式：

1. token數字-在數字世界內流通-在數字世界內交易/使用
2. token數字-在數字世界內流通-在物理世界內交易/使用
3. token物理-在數字世界內流通-在物理世界內交易/使用
4. token物理-在數字世界內流通-在數字世界內交易/使用

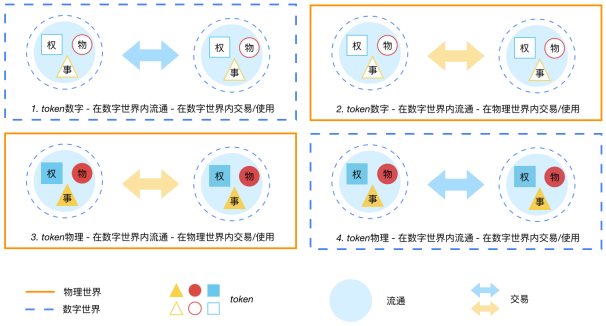


圖6 token的流通和使用方式

token數字-在數字世界內流通-在數字世界內交易/使用，這個大家見的多了，如各種公鏈的消耗型代幣，如ETH等等，用數字貨幣購買數字資產等等，各種數字貨幣之間交易等等。

token數字-在數字世界內流通-在物理世界內交易/使用，這個也很多，也是最熱門的，如各種以替代貨幣為出發點的公鏈，如BTC等，使用token作為數字貨幣購買物理世界內的事/物/權等。

token物理-在數字世界內流通-在數字世界內交易/使用，使用比較少見，因為物理世界的事/物/權不容易在數字世界內使用，另外這類型的token現在很少。

token物理-在數字世界內流通-在物理世界內交易/使用，大部分連接真實世界的應用場景，如門票應用的都是「3.token物理-在數字世界內流通-在物理世界內交易/使用」。

每當token需要和物理世界連接的時候都少不了中心信任機構，他們是把物理世界裡面的事/物/權token化的網關，也是「使用」token實現物理事/物/權時的網關。這個中心信任機構要有足夠的信用度來支撐他發行的token所代表的事/物/權，來完成網關功能：

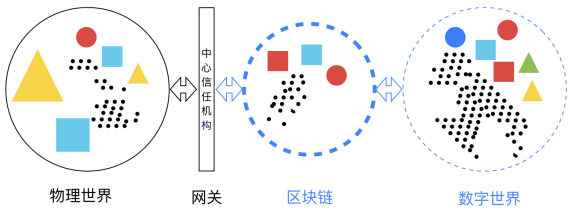


圖7 中心信任機構：網關

- 比如包子鋪能發行代表他一個一個包子的token，他的信用度可以讓所有人相信，在用戶「使用」這個token的時候可以得到一個包子。
- 比如房產所可以用token代表房屋的所有權，他的信用度可以讓所有人相信，我擁有這個token就等同於擁有這個房屋的所有權。
- 比如房東和房屋中介一起可以用token代表自己房屋1-3月的使用權，他們的信用度可以讓所有人相信，我擁有這個token就可以在1-3月使用這個房子。
- 比如國家可以發行替代貨幣的token，他的信用度可以讓所有人相信，我「擁有和使用」這個token和「擁有和使用」現有貨幣是一樣。

物理世界內的事/物/權在公有鏈上面token化，就會為事/物/權帶來自由流通的屬性。這種流通屬性的產生原因是區塊鏈確權，原本需要第三方機構確權的事情被一個公開的共享賬本替代了，帶來的好處就是，任何流通渠道都可以自由的加入token的流通，原本需要和第三方簽合同，打通數據庫，現在不用了。與此同時，中心信任機構可以在發行token的智能合約內設置各種規則，來實現每個token自帶規則的在公有鏈上自由流通，從而實現監管下的自由流通。

2018俄羅斯世界杯ERC 875門票測試結果

技術方案

2018俄羅斯世界杯ERC 875門票技術方案	
公有鏈	以太坊
智能合約標準	ERC 875 NFT（Non-fungible Token）標準
合約地址	0xA66A3F08068174e8F005112A8b2c7A507a822335
合約代碼	基於ERC 875 開發 https://github.com/alpha-wallet/contracts/blob/master/TicketingContract.sol
主辦方檢票端	Usher Mobile App 近期開源，代碼請參考https://github.com/alpha-wallet
用戶客戶端	AlphaWallet 近期開源，代碼請參考https://github.com/alpha-wallet
主辦方數據整理工具	Akio的解決方案

表2 2018俄羅斯世界杯ERC 875門票技術方案

公有鏈：以太坊

選用以太坊是因為：

1. 世界杯門票對流動性有著極高的要求，需要使用的公鏈是最廣泛被人接受的；
2. 世界杯是全球頂級賽事，測試過程不能有任何閃失，以太坊是支持智能合約的公鏈內，最成熟穩定的；
3. 世界杯門票價值比較高，對比來看，以太坊的gas費用影響不大；
1. 世界杯門票都是提前銷售，就算二級市場內也很少有最後一分鐘確認交易的需求，所以現有以太坊的性能可以支撐。

智能合約標準：ERC875 NFT（Non-fungible Token）標準

比較現有兩個NFT標準ERC721和ERC875後，選用ERC875的原因是：

1. ERC875能簡單的實現原子化交易，結合AlphaWallet基於ERC875開發出來的MagicLink功能，能夠實現：
   a) 主辦方通過任意渠道發送一個鏈接（MagicLink）給用戶用戶通過MagicLink就能直接導入1張或多張世界杯門票。不需要用戶持有以太幣。

a) 用戶可以自己使用AlphaWallet創建MagicLink來導出自己的門票，發送MagicLink給接收方用戶，接收方點擊鏈接，盛開體育支付gas，接收方倒入門票。免費轉讓時，發送方和接收方都不需要支付gas。

b) 用戶可以使用AlphaWallet創建MagicLink來出售他們的門票，發送MagicLink給買家或者把MagicLink發佈在各種市場內，買家點擊MagicLink，在AlphaWallet倒入頁面內檢查門票信息和價格，點擊確認購買，原子化交易發生，買家支付票價和gas費用收到門票，同時賣家收到錢門票消失。

2. ERC875能夠一次打包轉讓多個NFTs：門票應用場景內有很多時候需要同時轉讓多張門票，少到2張，多到2萬張都有。ERC875能讓我們時一次轉讓任意數量的門票。

合約地址：

0xA66A3F08068174e8F005112A8b2c7A507a822335

合約代碼： 基於ERC875標準開發

https://github.com/alpha-wallet/contracts/blob/master/TicketingContract.sol

合約代碼

```
//mainnet: 0xA66A3F08068174e8F005112A8b2c7A507a822335

// ["0x474d542b330000000000000000000000020b5b23d4704d415249524e04050001",
// "0x474d542b330000000000000000000000020b5b23d4704d415249524e04050002",
// "0x474d542b330000000000000000000000010a5b291a70504f4c53454e0f050001",
// "0x474d542b330000000000000000000000010a5b291a70504f4c53454e0f050002",
// "0x474d542b330000000000000000000000020b5b2944a052555345475911050001",
// "0x474d542b330000000000000000000000020b5b2944a052555345475911050002",
// "0x474d542b330000000000000000000000006055b3fae205735335735343a050001",
// "0x474d542b330000000000000000000000006055b3fae205735335735343a050002",
// "0x474d542b330000000000000000000000020b5b4a01e04c36314c36323f050001",
// "0x474d542b330000000000000000000000020b5b4a01e04c36314c36323f050002",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050001",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050002",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050003",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050004",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050005",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050006",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050007",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050008",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050009",
// "0x474d542b330000000000000000000001075b2282f05255534b53410105000a",
// "0x474d542b330000000000000000000001075b2282f05255534b53410105000b",
// "0x474d542b330000000000000000000001075b2282f05255534b53410105000c",
// "0x474d542b330000000000000000000001075b2282f05255534b53410105000d",
// "0x474d542b330000000000000000000001075b2282f05255534b53410105000e",
// "0x474d542b330000000000000000000001075b2282f05255534b53410105000f",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050010",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050011",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050012",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050013",
// "0x474d542b330000000000000000000001075b2282f05255534b534101050014"],
// "FIFA WC2018",
// "SHANKAI",
// "0x0D590124d2fAaBbbdFa5561ccBf778914a50BCca",
// "0xFE6d4bC2De2D0b0E6FE47f08A28Ed52F9d052A02",
// "0x2e558Bc42E2e37aB638daebA5CD1062e5b9923De"


pragma solidity ^0.4.17;
contract TicketPro
{
    mapping(address => bytes32[]) inventory;
    uint16 ticketIndex = 0; //to track mapping in tickets
    address organiser;
    address paymaster;
    uint numOfTransfers = 0;
    string public name;
    string public symbol;
    uint8 public constant decimals = 0; //no decimals as tickets cannot be split

    event Transfer(address indexed _to, uint16[] _indices);
    event TransferFrom(address indexed _from, address indexed _to, uint16[]
_indices);
    event Trade(address indexed seller, uint16[] ticketIndices, uint8 v, bytes32 r,
bytes32 s);
    event PassTo(uint16[] ticketIndices, uint8 v, bytes32 r, bytes32 s, address
indexed recipient);

    modifier organiserOnly()
    {
        if(msg.sender != organiser) revert();
        else _;
    }

    modifier payMasterOnly()
    {
        if(msg.sender != paymaster) revert();
        else _;
    }
```

```
    function() public { revert(); } //should not send any ether directly

    constructor (
        bytes32[] tickets,
        string nameOfContract,
        string symbolForContract,
        address organiserAddr,
        address paymasterAddr,
        address recipientAddr) public
    {
        name = nameOfContract;
        symbol = symbolForContract;
        organiser = organiserAddr;
        paymaster = paymasterAddr;
        inventory[recipientAddr] = tickets;
    }

    function getDecimals() public pure returns(uint)
    {
        return decimals;

    // example: 0, [3, 4], 27,
"0x9CAF1C785074F5948310CD1AA44CE2EFDA0AB19C308307610D7BA2C74604AE98",
"0x23D8D97AB44A2389043ECB3C1FB29C40EC702282DB6EE1D2B2204F8954E4B451"
    // price is encoded in the server and the msg.value is added to the message digest,
    // if the message digest is thus invalid then either the price or something else in the message is
invalid
    function trade(uint256 expiry,
                   uint16[] ticketIndices,
                   uint8 v,
                   bytes32 r,
                   bytes32 s) public payable
    {
        //checks expiry timestamp,
        //if fake timestamp is added then message verification will fail
        require(expiry > block.timestamp || expiry == 0);

        bytes32 message = encodeMessage(msg.value, expiry, ticketIndices);
        address seller = ecrecover(message, v, r, s);

        for(uint i = 0; i < ticketIndices.length; i++)
        { // transfer each individual tickets in the ask order
            uint16 index = ticketIndices[i];
            assert(inventory[seller][index] != bytes32(0)); // 0 means ticket gone.
            inventory[msg.sender].push(inventory[seller][index]);
            // 0 means ticket gone.
            delete inventory[seller][index];
        }
        seller.transfer(msg.value);

        emit Trade(seller, ticketIndices, v, r, s);
    }

    function loadNewTickets(bytes32[] tickets) public organiserOnly
    {
        for(uint i = 0; i < tickets.length; i++)
        {
            inventory[organiser].push(tickets[i]);
        }
    }

    function passTo(uint256 expiry,
                    uint16[] ticketIndices,
                    uint8 v,
                    bytes32 r,
                    bytes32 s,
                    address recipient) public payMasterOnly
    {
        require(expiry > block.timestamp || expiry == 0);
        bytes32 message = encodeMessage(0, expiry, ticketIndices);
        address giver = ecrecover(message, v, r, s);
        for(uint i = 0; i < ticketIndices.length; i++)
        {
            uint16 index = ticketIndices[i];
            //needs to use revert as all changes should be reversed
            //if the user doesn't hold all the tickets
            assert(inventory[giver][index] != bytes32(0));
            bytes32 ticket = inventory[giver][index];
            inventory[recipient].push(ticket);
            delete inventory[giver][index];
        }

        emit PassTo(ticketIndices, v, r, s, recipient);
    }

    //must also sign in the contractAddress
    function encodeMessage(uint value, uint expiry, uint16[] ticketIndices)
        internal view returns (bytes32)
    {
        bytes memory message = new bytes(84 + ticketIndices.length * 2);
        address contractAddress = getContractAddress();
        for (uint i = 0; i < 32; i++)
        {   // convert bytes32 to bytes[32]
            // this adds the price to the message
            message[i] = byte(bytes32(value << (8 * i)));
        }

        for (i = 0; i < 32; i++)
        {
            message[i + 32] = byte(bytes32(expiry << (8 * i)));
        }

        for(i = 0; i < 20; i++)
        {
            message[64 + i] = byte(bytes20(bytes20(contractAddress) << (8 * i)));
        }

        for (i = 0; i < ticketIndices.length; i++)
        {
            // convert int[] to bytes
            message[84 + i * 2 ] = byte(ticketIndices[i] >> 8);
            message[84 + i * 2 + 1] = byte(ticketIndices[i]);
        }

        return keccak256(message);
    }

    function name() public view returns(string)
    {
        return name;
    }

    function symbol() public view returns(string)
    {
        return symbol;
    }

    function getAmountTransferred() public view returns (uint)
    {
        return numOfTransfers;
    }

    function balanceOf(address _owner) public view returns (bytes32[])
    {
        return inventory[_owner];
    }

    function myBalance() public view returns(bytes32[]){
        return inventory[msg.sender];
    }

    function transfer(address _to, uint16[] ticketIndices) public
    {
        for(uint i = 0; i < ticketIndices.length; i++)
        {
            uint index = uint(ticketIndices[i]);
            assert(inventory[msg.sender][index] != bytes32(0));
            //pushes each element with ordering
            inventory[_to].push(inventory[msg.sender][index]);
            delete inventory[msg.sender][index];
        }
        emit Transfer(_to, ticketIndices);
    }

    function transferFrom(address _from, address _to, uint16[] ticketIndices)
        organiserOnly public
    {
        for(uint i = 0; i < ticketIndices.length; i++)
        {
            uint index = uint(ticketIndices[i]);
            assert(inventory[_from][index] != bytes32(0));
            //pushes each element with ordering
            inventory[_to].push(inventory[msg.sender][index]);
            delete inventory[_from][index];
        }

        emit TransferFrom(_from, _to, ticketIndices);
    }

    function endContract() public organiserOnly
    {
        selfdestruct(organiser);
    }

    function isStormBirdContract() public pure returns (bool)
    {
        return true;
    }

    function getContractAddress() public view returns(address)
    {
        return this;
    }

}
```

用戶客戶端：AlphaWallet

近期開源，代碼請參考https://github.com/alpha-wallet

- AlphaWallet是現在少數支持ERC875的加密錢包之一
- AlphaWallet支持自定義XML文檔，能夠實現世界杯門票內容的完整顯示如：門票級別，場次，隊伍，時間，日期，城市，場館等。https://github.com/alpha-wallet/contracts/blob/master/TicketingContract.xml

```
<attribute-type id="locality" oid="2.5.4.7" syntax="1.3.6.1.4.1.1466.115.121.1.15">
    <name lang="en">City</name>
    <name lang="zh">城市</name>
    <name lang="es">Ciudad</name>
    <name lang="ru">город</name>
    <origin as="mapping"
bitmask="0000000000000000000000000000000000FF0000000000000000000000000000">
      <option key="1">
        <value lang="ru">Москва́</value>
        <value lang="en">Moscow</value>
        <value lang="zh">莫斯科</value>
        <value lang="es">Moscú</value>
      </option>
```

- AlphaWallet為自定義XML文檔提供4個不同安全級別的簽名，在自定義的同時保證安全性。

```
<ds:SignatureValue>
kporx19FbGkNJaJAF4hMmpT6riH/lv2wdJdoMOlj8JVva5KIBc4eY0TW3tGz6sH7QwtAkDtb+n92
41/ebewwRVGHqYQ6lsNiS1AXq14e3nnSQkBEB4IwJ/4svy57e2TqQsPgiYfyf0kNpvQnhlUbHX6z
zSg6McerMl4Tlo7sE/+gUw==
</ds:SignatureValue>
```

- AlphaWallet可以在平台內直接運行盛開體育的門票應用，無需跳轉其他應用，也不使用用戶體驗極差的DApp瀏覽器。世界杯門票token在需要檢票的時候，可以轉化為動態線下二維碼（每30秒鐘變一次，不需要用戶端在線）

主辦方檢票端：Usher Mobile App

近期開源，代碼請參考https://github.com/alpha-wallet

檢票時Usher應用掃描用戶端二維碼，讀取區塊鏈上信息和二維碼內信息進行校驗，掃描通過後，用戶門票（ERC875 Token）被銷毀，防止二次利用。

主辦方數據整理工具：主辦方選擇了Akio的解決方案，應該是市場上面現存的唯一解決方案

https://akiolabs.com/product/nft/

Akio is a data platform for Ethereum that provides insight into the activity of any smart contract. With it, you can transform the raw data, join it with other off-chain sources, or create charts to view core metrics that your stakeholders care about.

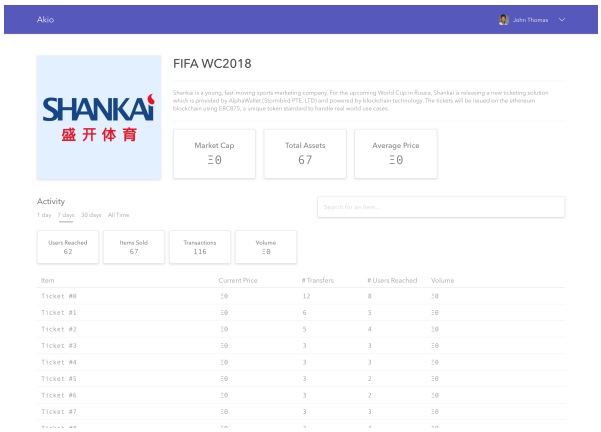


圖8 盛開合約內全部門票信息

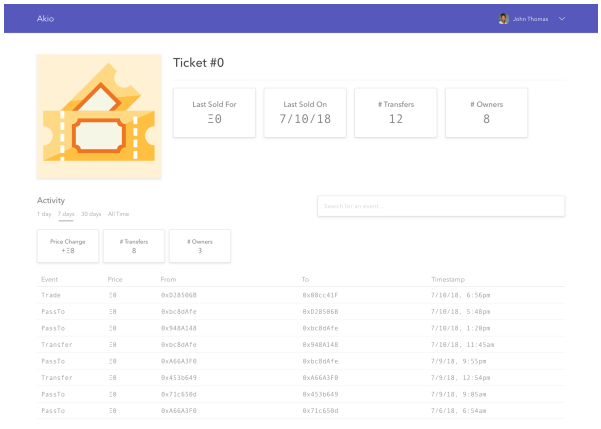
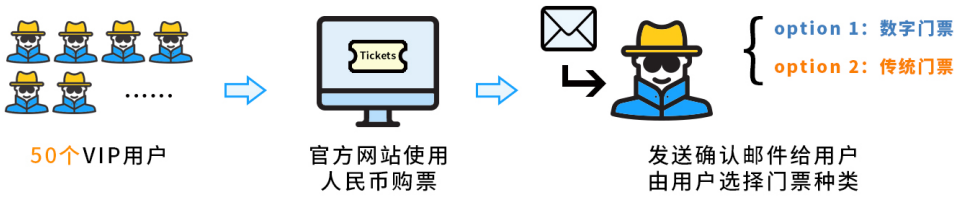


圖9 單張門票轉手信息

測試內容



我們抽取了50個開幕式VIP用戶，進行了測試。這些用戶在盛開體育官方售票網站，使用人民幣購票之後，收到了盛開體育發給他們的確認郵件，在郵件內提示用戶可以選擇數字門票解決方案或者傳統方式，數字門票部分含有一個MagicLink以及相應的使用說明，用戶可以自由選擇是否把他們的門票兌換卷轉化為區塊鏈門票並導入AlphaWallet。如果轉化成區塊鏈門票，原有門票兌換卷自動作廢。用戶可以在開賽前到盛開體育在俄羅斯各處設立的門票兌換櫃台，兌換最終的紙質門票。

測試目的和結果

1. 用戶對全數字門票方案的接受程度

最終有**28人**選擇區塊鏈門票，並順利使用區塊鏈門票兌換到了紙質門票，轉化率高於預期，我們有理由相信，如果強制用戶只能使用數字門票，用戶也不會有意見。

2. 用戶在使用過程中是否有我們沒有預料到的問題發生（用戶端APP）

一些國家出於保護消費者遠離數字貨幣欺詐的原因採取了一刀切的政策，不允許區塊鏈應用上架相應的國家的app市場。

3. 主辦方在使用過程中是否有我們沒有預料到的問題發生（主辦方端APP）

很多用戶不希望自己的區塊鏈門票被銷毀掉，希望可以留下來做紀念。基於本次測試只有28人使用區塊鏈門票，最後盛開體育同意用戶保留門票，沒有把用過的區塊鏈門票銷毀，而是選擇了額外記錄28人使用信息的方式來防止雙花。

4. 以太坊不斷變化的使用環境下，對用戶體驗造成的影響（如表3所示）

用戶體驗	造成的影響	
	對用戶	盛開體育
門票導入	**影響比較大** 門票只有在以太坊確認交易後才會顯示。以太坊擁堵情況下，交易確認最長延遲達14小時。	**有一定影響** 用戶有可能會不斷打電話給盛開，要求確認是否已經成功導入門票。
門票使用	**基本沒有影響** 動態二維碼可以離線使用。影響僅限於，被銷毀的票什麼時候從用戶端消失。	**基本沒有影響** 有可能有用戶，嘗試在門票銷毀前重復使用，但不會成功。
門票轉讓	**影響比較大** 門票只有在以太坊確認交易後才會顯示。以太坊擁堵情況下，交易確認最長延遲達14小時。	**有一定影響** 用戶有可能會不斷打電話給盛開，要求確認是否已經成功轉讓。
門票出售	**有一定影響** 用戶需要學習瞭解什麼是以太幣，為什麼出售後收到的是以太幣，有了基礎知識後，對於交易確認延遲也會有一定理解。建議用戶要交易的話，最好留出富裕時間。	**基本沒有影響** 因為買賣雙方都對以太坊有了一定瞭解

表3 以太坊不斷變化的使用環境下對用戶體驗造成的影響

AlphaWallet

官方網站 https://www.awallet.io

聯繫郵箱 info@awallet.io

關注我們

https://www.facebook.com/AlphaWallet/

https://www.linkedin.com/company/awallet/

https://twitter.com/Alpha_wallet

https://www.reddit.com/r/AlphaWallet/

https://github.com/alpha-wallet