JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 22, No. 8, pp. 1153-1163, Aug. 2021

Check for updates

블록체인 기반의 신뢰적 거래 플랫폼 개발

최 환 석[1] · 박 민 영[2] · 송 유 빈[2] · 이 우 섭[3*]
[1]한국과학기술원 전기및전자공학부 위촉연구원
[2]국립한밭대학교 정보통신공학과 학사과정, [3*]교수

Development of a blockchain based trusted trading platform

Hoan-Suk Choi[1] · Min-Young Park[2] · You-Bin Song[2] · Woo-Seop Rhee[3*]

[1]Contract Research Scientist, School of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 34051, Korea
[2]Bachelor's degree course, [3*]Professor, Department of Information Communication Engineering, Hanbat University, Daejeon 34158, Korea

[요 약]

공유경제는 잉여자원을 공유하여 수익을 발생시키는 경제 모델로 물품뿐만 아니라 숙박과 같은 각종 인프라의 공유가 이루어지고 있다. 하지만 최근 발생한 코로나 19로 인해 비대면 거래가 증가하면서 전자거래 분쟁이 증가하고 있다. 이를 위해 본 논문에서는 블록체인 기반의 신뢰적 거래 플랫폼을 제안한다. 제안하는 플랫폼은 이더리움 블록체인을 사용하여 거래 정보 및 송금 내역과 같은 중요 정보를 보안성 및 불변성의 특성을 가진 블록에 저장하여 부인방지를 지원하며, 정보의 객관적 보관으로 거래 주체 간 신뢰를 향상시킨다. 동시에 블록체인의 저장 공간 제약 및 트랜잭션 지연 문제를 최소화하기 위해 가변적 데이터를 별도의 서버로 처리한다. 또한 본 논문에서는 신뢰적 거래를 위한 로그인, 구매, 구매확정, 대여, 반납 및 배상 등의 절차를 제안하였으며, 이를 스마트 컨트랙트로 구현하였다. 결과적으로 Web3.js 기반으로 신뢰적 중고거래 서비스 DApp을 개발하였다.

[Abstract]

The sharing economy is an economic model that generates profits by sharing surplus resources. It shares not only goods but also various infrastructures such as accommodation. However the recent outbreak of Covid-19 has led to an increase in non-face-to-face transactions which has led to an increase in e-commerce disputes. To this end, we propose a blockchain based trusted trading platform. The proposed platform uses Ethereum blockchain to store important information such as transaction information and remittance history in blocks with characteristics of security and immutability to support denial prevention, and improves trust between trading entities with objective storage of information. Also, variable data is treated as a separate server to minimize storage space constraints and transaction delay problems in blockchain. This paper proposes procedures such as login, purchase, purchase confirmation, rental, return and compensation for reliable transactions, and implements them as smart contract. As a results, we develop a trusted used product trading service DApp based on Web3.js.

색인어 : 블록체인, 신뢰, 거래 플랫폼, 공유경제, 이더리움

Key word : Block chain, Trust, Trading platform, Sharing economy, Ethereum

http://dx.doi.org/10.9728/dcs.2021.22.8.1153

Received 28 June 2021; **Revised** 30 July 2021
Accepted 30 July 2021

***Corresponding Author; Woo-Seop Rhee**

Tel: +82-42-821-1749
E-mail: wsrhee@hanbat.ac.kr

Ⅰ. 서 론

공유 경제란 한번 생산된 제품을 여럿이 공유하여 사용하는 협업 소비 기반의 경제 모델로, 과잉생산, 시장 포화, 환경오염 등 기존 경제모델에서 발생하는 각종 문제를 해결할 수 있는 대책으로 대두되고 있다[1]. 이는 특정 제품에 한정되지 않고 생산설비 및 서비스와 같이 빌려줄 수 있는 모든 잉여자원을 대상으로 확대되고 있다. 한편 잉여자원을 공유함으로써 소유자는 추가적인 수익을 발생시킬 수 있어 다양한 분야의 공유경제 서비스가 제공되고 있다. Airbnb로 대표되는 숙박 공유는 공유경제의 대표 분야로 꼽힌다. 이는 현재 호텔산업을 위협하는 위치에 올라와 있으며 특히 사회적, 경제적 트랜드에 따라 호텔뿐만 아니라 다양한 협업 공간 제공, 비품, 세무대행, 식당, 헬스장 등의 다양한 서비스를 제공하는 공유 오피스 분야가 급성장하고 있다. 이와 같은 공유 경제모델은 유 무형적인 자원을 손쉽게 대여하여 활용하고 반납함으로써 유연한 조직의 운영이 가능하며, 개인의 입장에서도 다양한 자원을 부담없이 이용할 수 있는 효과적인 방법이 된다.

또한 최근 발생한 코로나 19로 인해, 경제 불황이 장기화 되면서 중고거래, 대여 등을 통한 실용적 소비를 추구하는 사람들이 증가하고 있다[2]. 하지만 사회적 거리두기로 인해 직접 대면이 어려워지면서 비대면 거래의 선호도가 증가하였으며, 이에 따른 사기 위험 또한 함께 증가하고 있다. 한국인터넷진흥원이 발간한 '2020 전자거래 분쟁조정 사례집'에 따르면 2019년 전자거래 분쟁조정 신청 건수는 총 2만845건으로 2018 대비 11% 증가했다[3]. 이러한 중고거래 관련 사기는 대부분 현금 입금을 먼저 유도한 뒤 물건을 보내지 않고 잠적하는 방식이 주를 이룬다. 이를 해결하기 위해 거래당사자 간의 신뢰적 거래를 제공할 방법이 요구된다.

따라서 본 논문에서는 블록체인 기반의 신뢰적 거래 플랫폼을 개발하였다. 블록체인은 체인을 형성하는 참여자가 블록을 유지하며 직접 검증을 수행하는 탈중앙성, 블록 구성 정보가 변경되면 전체 블록의 해시값 변동을 가져오게 되어 특정 노트가 임의로 정보의 변경을 불가능하게 하는 불변성, 블록에 저장된 기록을 누구나 접근 가능한 투명성의 특징을 가지고 있어 신뢰적인 정보 저장 및 관리가 가능하다.

하지만 블록체인을 구성하는 블록은 사용자가 증가함에 따라 블록에 담을 수 있는 데이터 최대 한도를 초과하는 확장성 문제가 발생하며, 저장 용량이 한정적이고, 합의 알고리즘을 통해 모든 참여 노드가 동일한 블록을 유지해야 하므로 블록 추가에 소요되는 처리 시간이 큰 편이다.

본 논문에서 제안하는 플랫폼은 블록체인의 저장 공간 부족 및 블록 추가 시 발생하는 지연을 줄이기 위해 서비스 제공시 처리하는 정보를 가변정보와 불변정보로 구분하여 처리한다. 상품의 설명, 관련 사진 등과 같은 가변정보는 데이터 저장이 쉽고 빠른 데이터베이스를 활용하여 처리하며, 불변 성질을 가지고 지속적으로 보관해야 하는 거래 정보만을 블록체인으로 저장하여 정보의 무결성과 투명성을 제공한다.

이를 위해 본 논문에서는 2장에서 대표적 블록체인 플랫폼을 소개하고 3장에서 제안하는 신뢰적 거래 플랫폼을 소개한다. 4장에서는 제안하는 신뢰적 거래 플랫폼의 구현결과를 보여주며, 5장에서 결론을 맺는다.

Ⅱ. 블록체인 기술 동향

2-1 이더리움(Ethereum)

이더리움[4]은 비탈릭 뷰테린에 의해 고안되어, 2015년 7월 30일에 발표된 퍼블릭 블록체인 플랫폼이다. 비트코인과 달리 프로그래밍이 가능한 블록체인(Programmable Blockchain)으로써, 분산어플리케이션 개발을 위한 플랫폼을 제공한다. 솔리디티(Solidity)라는 튜링완전언어(Turing-complete language)를 사용해 스마트 컨트랙트, 분산 어플리케이션을 작성하고 소유권에 대한 임의의 규칙, 트랜잭션 형식, 상태 변환함수 등을 생성할 수 있다. 또한 이더(ether) 라는 화폐 단위를 이용한다.

이더리움은 Ethash라 불리는 작업증명(PoW; Proof-of-Work) 합의 방식을 이용하며 약 12초당 한 개의 블록이 생성될 수 있도록 설계되었다. 채굴에는 DAG 파일(2차원 배열 데이터)이 이용되며, 이것은 GPU 연산 효율을 높이고 ASIC을 이용한 채굴을 방지한다. 또한 다양한 운영체제를 지원하며 분산 애플리케이션을 제작해 배포할 수 있다. 하지만 느린 처리 속도와 높은 트랜잭션 수수료가 단점으로 지적되고 있다.

2-2 EOS(Everyone's Open Society)

EOS[5]는 이더리움의 느린 처리 속도와 높은 수수료 문제를 해결하기 위한 대안으로 등장했으며, 2017년 5월부터 미국 블록원(Block.one)의 Brendan Blumer 대표이사와 Dan Larimer 기술이사 등이 이더리움을 기반으로 개발하였다. 2018년 6월 이더리움에서 벗어나 자체 메인넷을 오픈했으며 이더리움과 마찬가지로 튜링완전언어로 구성되었으며, 이더리움보다 빠른 속도와 안정성을 보장한다.

트랜잭션 속도 문제를 해결하기 위해 EOS는 전체 토큰 보유자들이 21명의 블록 생성자(BP; Block Producer)를 선출한 후 그들에게 블록체인의 운영을 맡기는 위임지분증명(DPoS; Delegated Proof of Stake)[6] 합의 알고리즘을 사용한다. 선출된 대표 노드만 합의 과정에 참여하므로, 트랜잭션 처리가 빠르다. 이로 인해 이더리움 평균 처리속도 20TPS에 비해, EOS는 평균 3,000TPS의 빠른 트랜잭션 처리가 가능하다.

이더리움의 경우, 노드가 거래 내용을 담은 블록을 생성하는 대가로 가스라는 수수료를 받지만, EOS는 DApp 개발자가 거래 수수료를 지불하여 사용자는 무료로 서비스를 이용할 수 있다. EOS는 EOS 코인을 많이 보유한 개발자에게 트래픽을 보장하기 때문에 많은 EOS 코인을 보유하고 있어야 하며, 초기 런칭 비용이 커질 수 있다.

단점으로 낮은 투표율을 꼽을 수 있다. EOS 투표율은 대략 22% 선에서 오르지 않고 있다. 그 이유는 투표시, 자신의 EOS가 3일 동안 거래 중지되는 스테이킹을 해야 하므로 코인 가격 변동에 따른 위험을 고스란히 감수해야 하기 때문이다.

2-3 클레이튼(Klaytn)

클레이튼(Klaytn)[7]은 ㈜카카오의 자회사인 그라운드엑스가 개발한 퍼블릭 블록체인 플랫폼이다. 클레이튼은 블록체인 노드를 소수의 합의 노드와 불특정 다수의 레인저 노드로 나누어 운영하는 특수한 개념의 하이브리드 구조[8]를 가진다. 비잔틴 장애 허용(BFT; Byzantine Fault Tolerant) 합의 알고리즘을 기반으로 3분의 1이상이 담합하지 않으면 네트워크의 손상 없이 빠르게 블록을 확정하여 속도 문제를 해결한다. 레인저 노드는 퍼블릭 네트워크로 구성되어 읽기 요청을 처리하고, 합의 노드는 쓰기 요청을 처리한다. 합의 노드는 대형 서비스를 제공할 수 있는 기업이 참여하고, 노드가 증가할 경우 속도 저하가 문제가 발생할 수 있어 수십 개로 제한되며 합의, 레인저 노드 모두 클레이튼의 화폐인 클레이로 블록 생성 보상을 받는다.

또한, 허가된 노드만 합의 과정에 참여할 수 있어 신뢰성이 확보되며 퍼블릭 블록체인의 보안과 투명성을 추가할 수 있다.

Ⅲ. 블록체인 기반 신뢰적 거래 플랫폼

제안하는 블록체인 기반 신뢰적 거래 플랫폼은 그림 1과 같이 구성된다.

제안하는 플랫폼은 크게 사용자의 DApp, 실시간 데이터베이스 서버와 이더리움 블록체인 네트워크로 구성된다. 사용자는 DApp을 통해서 자신이 판매/대여 하고자 하는 물품을 등록한다. 이때, 물품정보는 지속적으로 변경 가능하며, 사진 및 세부적인 설명 등의 정보를 포함할 수 있어, 상대적으로 많은 용량을 차지하며, 거래 종료시 계속 보관할 필요가 없다. 반면 거래정보는 거래 종료 후에도 거래 증명을 위해 보관이 필요하며, 이력정보 특성상 수정이 필요 없다.

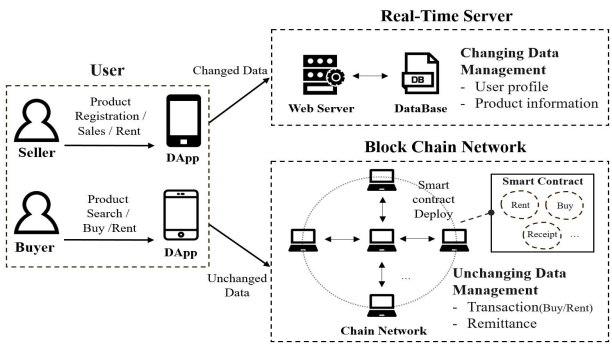

그림 1. 제안하는 블록체인 기반 신뢰적 거래 플랫폼 개요
Fig. 1. Overview of the proposed Block chain based trusted trading platform

따라서 제안하는 신뢰적 거래 플랫폼은 물품정보 등 변경 가능성 있는 자료는 실시간 데이터베이스 서버에 저장하며, 송금정보를 포함한 거래정보는 블록체인 네트워크에 저장하여 불변성 및 무결성을 보장한다.

이를 위해 3-1장에서 본 시스템의 이더리움 기반 개발환경을 소개하며, 3-2장에서 DApp 개발환경을 설명한다. 또한 3-3장에서 분산원장기반 신뢰적인 거래 프로시저를 제안하며, 3-4장에서 이를 위한 스마트컨트렉트 설계 내용을 제시한다.

3-1 이더리움 기반 분산원장 시스템 개발환경

본 장에서는 제안하는 시스템의 개발환경을 설명한다. 이더리움 기반의 개발환경을 구축하기 위해 가나슈(Ganache), 메타마스크(Metamask), 트러플(Truffle)을 활용한다. 먼저 블록을 처리할 이더리움 네트워크를 설치하면, 메타마스크와 가나슈 등을 통해 네트워크 내부의 처리 과정을 확인할 수 있다. Dapp은 Web3.js를 기반으로 블록체인의 다양한 기능을 활용할 수 있다. 인터페이스 연동을 위해서는 Web3와 스마트컨트랙트를 인스턴스화 하고, 인터페이스인 프론트엔드의 사용자 작동을 기반으로 체인에 데이터를 전달하기 위해 Web3 함수를 이용한다. 마지막으로 처리된 결과를 바탕으로 UI를 업데이트 한다.

1) 가나슈 [9]

가나슈(Ganache)란 테스트 목적으로 PC에 설치해서 사용할 수 있는 일종의 간이 블록체인이다. 메인 네트워크와 연결없이 로컬에서 작동하여, 계약을 손쉽게 배포 및 테스트할 수 있으며 계정과 잔액에 대한 정보를 제공한다. geth 또는 parity 같은 클라이언트를 사용하면 각 트랜잭션 실행에 15초씩 걸리기 때문에 테스트가 어렵다. 따라서 가나슈를 활용한다. 또한 테스트를 위해 100이더가 탑재된 10개의 테스트 계정을 제공한다.

본 개발 과정에서는 가나슈 버전 1.1.0.appx – Candy Apple을 사용하였으며 그림 2는 가나슈 UI로 노드에서 채굴한 마지막 블록 넘버인 Current Block, 노드가 트랜잭션을 채굴하기 위한 최소 가스 가격인 Gas Price, 트랜잭션 완료를 위해 필요한 최대 가스인 Gas Limit, 체인 네트워크 식별 ID인 Network Id, 블록 채굴 속도인 Mining status, 메타마스크에서 가나슈 계정이전을 위한 키인 MNEMONIC, 트랜잭션 처리 횟수인 TX count, 계정 생성 순서인 Index, 개인 비밀키 등의 정보를 제공한다.

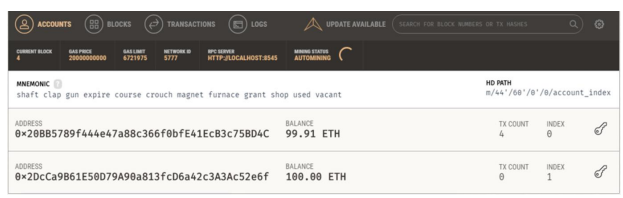

그림 2. 가나슈 사용자 인터페이스
Fig. 2. Ganache user interface

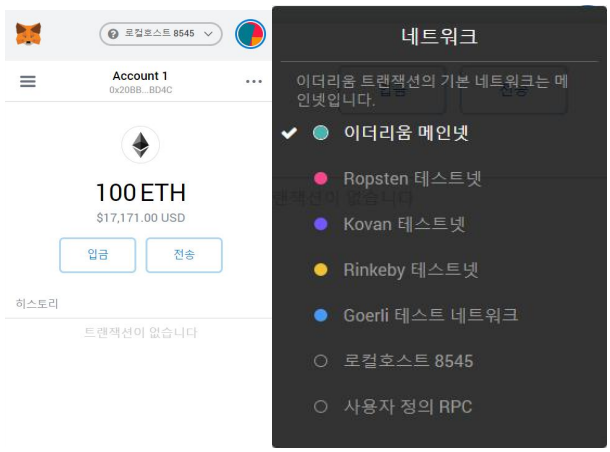

그림 3. 메타마스크
Fig. 3. Metamask

2) 메타마스크 [10]

메타마스크는 크롬이나 파이어폭스의 브라우저 extension으로 제공되는 이더리움 개인 지갑으로, 인퓨라라는 원격 이더리움 노드를 통해 이더리움 네트워크에 접속한다. 앞서 설명한 가나슈의 MNEMONIC을 통해 메타마스크로 계정을 이전할 수 있다. 메타마스크는 메인 네트워크와 테스트 네트워크, Local Host 8545 그리고 Custom RPC에 접속할 수 있다. 테스트 네트워크는 가짜 에더를 사용하지만, 환경은 메인 네트워크와 같다.

3) 트러플[11]

트러플은 스마트 컨트랙트를 컴파일 및 테스트와 배포를 위한 프레임워크로 npm을 통해 설치할 수 있고 node.js 가 먼저 설치되어야 실행 가능하다.

3-2 이더리움 기반 DApp 개발 환경

1) Node.js 기반 백앤드 환경

제안하는 DApp은 Node.js 기반으로 백앤드 환경을 구축하였다. Node.js는 크롬 V8 자바스크립트 엔진으로 빌드된 가볍고 효율적인 비동기 이벤트 주도 자바스크립트 런타임으로써 확장성 있는 네트워크 애플리케이션을 만들 수 있다[12].

DApp은 웹 애플리케이션 형태로 Express 기반으로 개발되었다. Express는 간결하고 유연한 Node.js 기반 웹 애플리케이션 프레임워크이다. 서버의 페이지가 늘어나고 사이트가 커지면 코드가 복잡해지는데, Express는 코드의 양을 줄여주고 유지보수가 쉽다. Express는 요청을 처리할 때 미들웨어를 통해 지정한 동작을 수행할 수 있다. 대표적인 미들웨어는 Morgan, Body-parser, Cors 등이 있으며, 모두 npm으로 설치 가능하다. Morgan은 Express 동작 중 발생하는 메시지를 콘솔에 출력하며, Body-parser는 폼에서 전송되는 POST 값을 사용할 수 있게 해준다. 또한, Cors는 다른 도메인 간의 AJAX 요청을 가능하게 한다. Express의 다른 장점은 라우팅이 편리하다는 것이다.

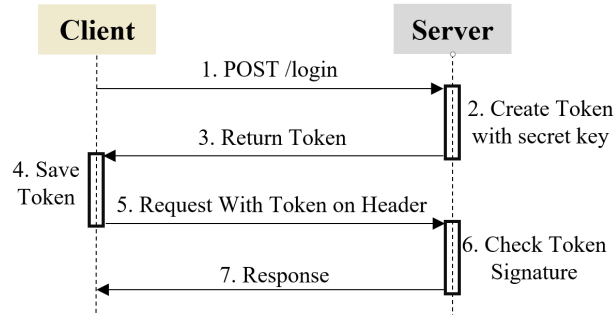

그림 4. 토큰 기반 인증 절차 [14]
Fig. 4. Token-based authentication procedures

라우팅은 요청 주소에 따라 다른 처리를 하는 것으로, Express는 REST API에 따라 처리하며 그 방법이 아주 간단하다. app.get, app.post, app.put, app.delete 등의 메소드를 사용한다.[13]

2) JWT(JSON Web Token) 기반 인증

제안하는 DApp은 토큰 기반의 사용자 인증을 활용한다. 토큰은 클라이언트 측에 저장하기 때문에 완전히 stateless 하며, 서버를 확장하기에 적합하며 쿠키를 사용하지 않아 쿠키관련 취약점이 없고 토큰을 공유하여 다른 서비스에서도 권한을 공유할 수 있다.

JWT는 RFC7519 표준으로 두 개체에서 JSON 객체를 사용하여 가볍고 안전하게 정보를 전달한다. JWT는 C, Java, Python, C++, R, C#, PHP, JavaScript, Ruby, Go, Swift 등 대부분의 주류 프로그래밍 언어에서 지원되며, JWT에서 발급된 토큰은, 토큰 기본정보, 전달 정보, 그리고 토큰 검증을 위한 signature를 포함한다. 또한, JWT는 HTTP 헤더, URL 파라메터 등을 통해 손쉽게 전달될 수 있으며 그림 4와 같이 회원 인증에서 유용하게 사용된다.

1. 유저가 아이디와 비밀번호로 로그인을 한다.
2. 서버는 계정정보를 검증하고, signed 토큰을 발급한다.
3.4. 클라이언트는 전달받은 토큰을 저장하고,
5. 요청을 할 때마다, 해당 토큰을 함께 전달한다.
6. 서버는 토큰을 검증하고,
7. 요청에 응답한다.

서버 측에서는 유저의 로그인 여부를 신경 쓸 필요가 없고, 요청시 동봉된 토큰만 확인하여, 세션 관리가 필요 없다.

3) MySQL 기반 데이터베이스 구축

제안하는 시스템은 자주 변경되며 상대적으로 큰 데이터를 MySQL 데이터베이스를 활용하여 저장, 관리한다. MySQL은 오픈소스기반 관계형 데이터베이스로 다중 사용자와 다중 스레드를 지원하며, C, C++, JAVA, PHP 등 여러 프로그래밍 언어를 위한 다양한 API를 제공한다.

그림 5는 제안하는 시스템을 위한 데이터베이스 구조이다.

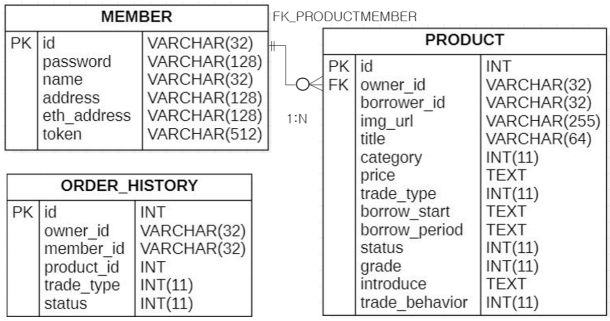

그림 5. 제안하는 공유 마켓 서비스를 위한 데이터베이스
Fig. 5. Proposed DB for sharing market

Id, password, name, address 등 사용자 정보를 저장하는 Member 테이블과 상품의 다양한 정보를 저장하는 Product 테이블의 id를 Key로 연결하였으며, 특정 주문의 상태 변경정보를 관리하기 위한 Order_history 테이블이 존재한다.

3-3 분산원장 기반 신뢰적 거래 절차

본 연구에서는 신뢰적 거래를 위해 물품 구매, 대여, 반납 및 배상 절차를 설계하였다. 이 절차를 기반으로 스마트 컨트랙트가 개발되었다.

1) 회원 가입 및 로그인 절차

그림 6은 제안하는 시스템의 회원가입 및 로그인 절차이다. 일반적인 웹 서비스의 회원가입 절차에 메타마스크를 통한 이더리움 네트워크의 계정정보인 지갑 주소(Wallet address) 획득 절차가 더해진 형태이다. 사용자가 DApp을 통해 회원가입을 요청하면, 먼저 메타마스크를 통해 이더리움 계정정보를 호출한다. 이때 이더리움 계정이 없다면, 메타마스크를 통해 계정을 생성할 수 있다. 계정정보를 획득한 DApp은 이를 포함한 사용자 정보를 웹서버에 전달하여 사용자 회원가입 절차를 진행하며, 해당 정보를 DB에 저장한다.

로그인 절차는 그림6의 하단부와 같이 사용자의 ID, Password를 전달하여 일치 여부를 판단하고, 메타마스크를 통해 획득한 계정정보를 기반으로 블록체인 네트워크의 지갑 주소를 획득한다.

2) 상품정보 업로드 및 갱신 절차

그림 7은 제안하는 시스템의 상품정보 업로드 및 갱신 절차이다. 사용자는 DApp을 통해 상품의 정보를 웹서버에 전송한다. 상품정보는 지속적으로 변경 가능한 정보로 블록체인 네트워크에 저장되지 않는다. 웹서버는 해당 정보를 DB에 기록하며, 상품정보 갱신도 동일한 절차를 거쳐 DB에 갱신된다.

3) 물품 구매 절차

그림 8은 제안하는 시스템의 구매 절차이다. 소비자가 DApp을 통해 구매 요청을 하면 메타마스크를 통해 송금에 관한 서명을 획득한다. DApp은 획득한 서명정보와 구매하려는 제품정보 및 가격정보, 그리고 지갑주소를 전달하며, 송금을 요청한다. 이때 물품 대금은 상품의 수령확인 및 구매확정이 이루어지기 전까지 Locker라고 하는 중간 보관소 역할을 하는 계정에 보관된다. 이와 같은 송금 내역(거래내역)은 이더리움 네트워크의 각 블록에 추가되며, 모든 노드가 확인할 수 있다. 대금의 송금이 완료되면 해당 상품의 상태가 거래중으로 변경되기 위해 웹서버의 DB가 갱신되며, 갱신된 상품의 정보가 소비자와 상품 제공자에게 전달된다.

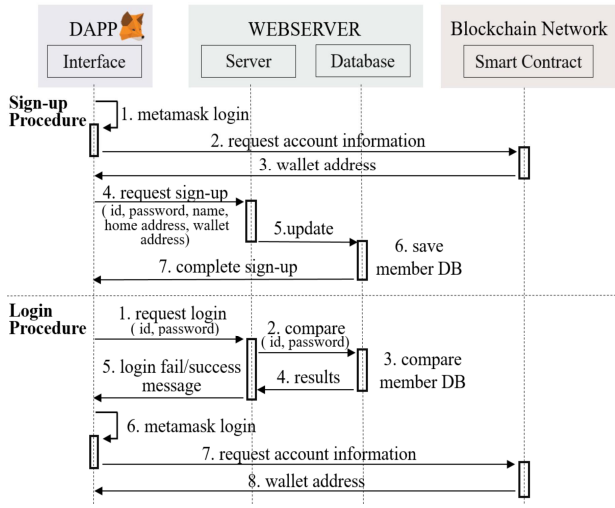

그림 6. 회원가입 및 로그인 절차
Fig. 6. Sign-up & Log-in procedures

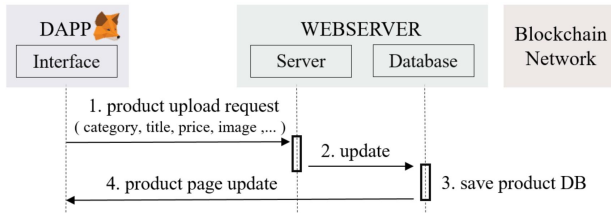

그림 7. 상품정보 업로드/갱신 절차
Fig. 7. Product information upload/update procedures

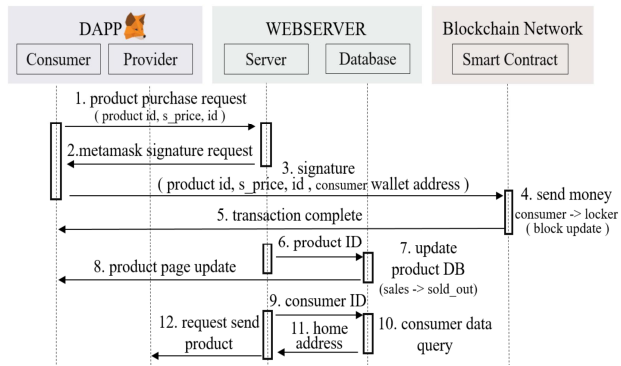

그림 8. 물품구매 절차
Fig. 8. Product purchase procedures

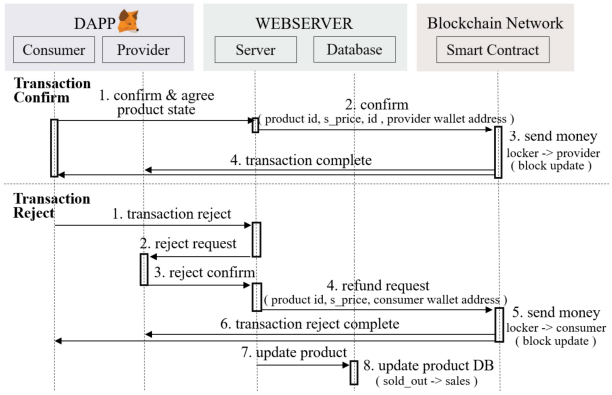

그림 9. 구매확정 및 환불 절차
Fig. 9. Transaction confirm and reject procedures

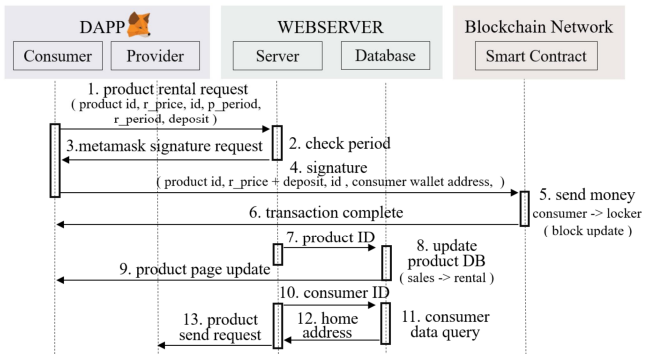

그림 10. 물품대여 절차
Fig. 10. Product rental procedures

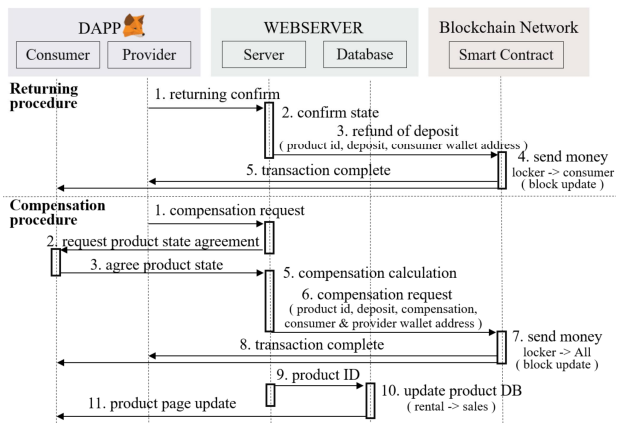

그림 11. 반납 및 배상 절차
Fig. 11. Returning and compensation procedures

4) 물품의 구매확정 및 환불 절차

그림 8에서 설명한 바와 같이 물품 구매 절차가 끝나면 물품 대금은 소비자의 지갑에서 Locker 계정의 지갑으로 송금된다. 이후 그림 9와 같은 구매확정 및 환불절차가 진행된다. 만약 사용자가 해당 물품을 잘 수령하고, 거래를 확정지을 경우, DApp은 거래완료 정보를 웹서버에 보내고, 웹서버는 해당 상품의 상태를 거래완료로 변경한다. 또한 블록체인 네트워크에 Locker 계정에서 물건 제공자 계정으로의 송금을 요청한다. 해당 트랜잭션이 완료되어 블록에 거래 내역이 저장되면, 거래 완료 정보를 판매자와 구매자에게 전달하여, 거래 완료를 알린다.

하지만 해당 상품 구매를 취소하고 환불 절차를 진행하고 싶은 경우, DApp을 통해 거래 거절 메시지를 웹서버로 보낸다. 웹서버는 판매자에게 이를 통보하고, 판매자가 환불에 동의할 경우 해당 상품정보를 갱신하고, 블록체인 네트워크에 Locker 계정에서 구매자 계정으로의 송금을 요청한다. 환불 트랜잭션이 완료되어 블록에 거래내역이 저장되면, 구매확정과 마찬가지로 판매자와 구매자에게 환불절차의 완료를 알린다.

5) 물품 대여 및 반납 절차

그림 10은 제안하는 시스템의 물품 대여 절차이다. 물품의 대여는 대여기간, 보증금 등의 추가 정보가 요구되는 것을 제외하고 구매와 동일하다. 대여 요청 시 상품의 보증금을 포함하여 Locker 계정으로의 송금이 진행되며 물품의 사용이 끝나면 물품 상태의 확인 후 반납과 배상 과정으로 이어진다.

대여의 경우 상품의 반납이 이루어져야 하며, 상품의 손상 상태에 따른 배상이 이루어져야 한다. 제안하는 반납과 배상의 시퀀스 다이어그램은 그림 11과 같다. 대여 상품의 공급자가 상품 수령 후 손상 상태를 확인하고 이에 따라 상태 변화가 있을 경우 배상금 계산이 이루어지며, Locker 계정에 저장되어 있던 보증 금액에서 배상 금액을 차감한 금액을 소비자에게, 배상 금액은 공급자에게 송금되며 거래가 완료된다. 이후 상품의 상태 및 거래 내역 정보의 업데이트가 이루어진다.

3-4 거래 기록 저장을 위한 스마트 컨트랙트 개발

1) 스마트 컨트랙트 개발 환경

스마트 컨트랙트는 계약의 기능을 디지털 명령어로 작성한 디지털 계약 프로토콜이다. 디지털 명령어로 작성되었기에 조건에 따라 계약의 내용이 자동으로 즉각 이행되며, 그 내용은 변경될 수 없도록 한다. 이로 인해 제 3자의 개입 없이도 신뢰할 수 있는 거래가 가능하다. 이더리움은 튜링 완전 스마트 컨트랙트 개발 언어인 솔리디티(Solidity)를 제공한다[15]. 개발자는 솔리디티를 이용하여 자체적인 로직을 포함하는 스마트 컨트랙트를 개발할 수 있으며 계약 진행 후, 계약 당사자 간 부인이 불가능하며 신뢰할 수 있는 트랜잭션을 보관한다[16]. 작성된 스마트 컨트랙트는 컴파일을 통해 바이트 코드로 변환하여 블록체인에 배포된다. 채굴자는 배포된 스마트 컨트랙트 코드를 EVM (Ethereum Virtual Machine)에서 실행한다. 이때, Gas Fee가 계산되면서 블록에 추가되고, 실행 결과가 유효한 경우 State에 반영된다.

스마트 컨트랙트에서 제공하는 함수는 Web3.js 라이브러리를 통해 웹 브라우저에서 접근된다[17]. Web3.js는 이더리움 JSON RPC를 구현한 자바스크립트 API로, 자바스크립트 기반으로 DApp이나 서비스를 구현할 때 활용된다. 내부적으로 HTTP나 IPC를 통해 JSON RPC API를 호출한다.

개발된 컨트랙트의 함수에 대해 여러 유형의 안전한 접근을 지원하기 위해 ABI(Application Binary Interface) 기능을 제공한다. ABI는 컨트랙트의 함수와 매개변수들을 JSON 형식으로 나타낸 리스트로 함수 호출에 필요한 정보를 제공한다[18].

2) 블록 저장 구조 설계 및 스마트 컨트랙트 배포 결과

그림 12는 거래 기록의 저장을 위한 구조체 설계 내용이다. 이는 특정 판매자, 구매자, 거래 유형, 금액 등의 거래 정보를 저장하는 Trade 구조체를 정의하였고 이 구조가 블록에 저장된다. Goblin은 이를 웹상에서 호출하고 활용하기 위한 객체이다. Trades는 블록에 저장한 거래내용을 저장하기 위해 Trade 구조체 형태이며, owner는 해당 스마트 컨트랙트의 생성자 주소를 저장한다. tradeID는 거래정보를 조회하기 위한 ID 값이다. behavior() 함수는 구매 및 대여와 같은 거래행위를 했을 때 송금 처리 및 거래 기록을 남기기 위한 기능을 수행한다. 이때 저장되는 데이터의 위치를 'storage'로 설정하면, 메모리가 아닌 실제 블록에 해당 내용이 기록되며 함수 타입을 Payable로 설정하여 송금이 가능하게 한다. 이때 필요 정보인 판매자, 구매자, 거래액 등의 값은 파라미터로 전달한다. getTrade() 함수는 거래정보를 블록에서 탐색하여 반환하는 함수로, 사용자가 거래정보를 확인할 수 있도록 한다. getTradeID() 함수는 마지막에 처리된 거래의 ID를 반환한다.

테스트를 위하여 스마트 컨트랙트 코드를 컴파일하고 배포할 수 있는 통합 개발 환경인 Remix를 활용하였다. 그림 13은 Remix 환경에서 스마트 컨트랙트의 배포 결과를 확인한 것이다. status 값을 보면 컨트랙트를 배포하는 트랜잭션 실행이 성공했으며 배포자 주소 및 기타 가스 정보 등을 확인할 수 있다.

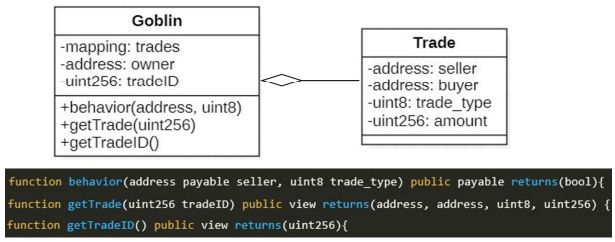

그림 12. 거래 기록 저장을 위한 구조체 구조
Fig. 12. Data structure for storing transaction

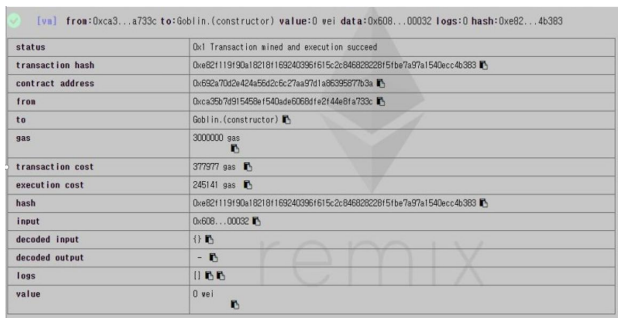

그림 13. 스마트 컨트렉트 배포 결과
Fig. 13. Smart contract deploy result

그림 14. 최근 거래정보 확인 결과
Fig. 14. Last transaction information

getTradeID() 함수를 활용하여 가장 최근에 실행된 트랜잭션의 정보를 확인한다. 그림 14와 같이 판매자, 구매자, 거래 유형, 값을 확인할 수 있다.

Ⅳ. 블록체인 기반 신뢰적 거래 플랫폼 구현 결과

4장에서는 제안하는 신뢰적 거래 플랫폼의 구현 결과를 소개한다. 먼저 해당 서비스의 거래정보를 저장한 체인 네트워크 구성에 대해 4-1에서 소개하며, 4-2는 블록체인 기반 중고거래 서비스의 구현결과를 설명한다.

4-1 이더리움 기반 프라이빗 체인 네트워크 구성

프라이빗 체인은 퍼블릭 체인과 달리 미리 정해진 조직이나 개인들만 참여할 수 있기 때문에 빠른 처리 속도와 블록 생성을 지원할 수 있고, 테스트 넷으로도 활용된다. 본 논문에서 구성한 체인 네트워크는 독립적인 3대의 PC에서 4개의 노드를 생성한 뒤 노드들을 연결하여 구축된다. 먼저 프라이빗 네트워크를 구성하기 위한 노드를 생성한 뒤, 부트 노드에 피어 노드 2개를 연결해 프라이빗 네트워크를 구축한다. 마지막으로 부트 노드를 제외한 피어 노드에서 마이너 노드를 지정한 뒤, 노드 하나를 추가 생성해 연결한다.

1) 제네시스 블록 설정 및 부트스트랩 노드 생성

체인 네트워크를 구축하기 위해서는 먼저 제네시스 블록 설정을 하고, 사용자 정의 Network id와 부트스트랩(부트) 노드가 필요하다. 이때 특정 프라이빗 네트워크를 구성하는 모든 노드의 제네시스 파일과 Network id가 동일해야 하며, 동일한 부트 노드를 통해 연결되어야 한다. 네트워크 상의 피어 노드들을 찾는데 사용되는 노드를 부트 노드라고 하며, 이는 블록체인 정보를 저장하지 않고, 일정 시간 동안 연결된 노드의 목록을 유지하다 새로운 피어 노드에게 노드의 목록을 전달한다. 블록체인은 P2P방식이기 때문에 피어 노드가 네트워크에 연결되면 다른 노드를 탐색하고, 하나 이상의 부트스트랩 노드에 접속하여 연결된 노드의 목록을 받아 해당 노드들과 연결한다. 부트스트랩 노드는 피어 노드가 많은 네트워크에서 사용하며 작은 규모의 경우 Geth 클라이언트가 부트스트랩 노드 역할을 수행한다.

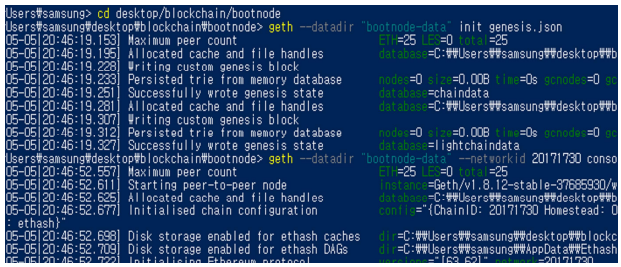

그림 15. 제네시스 블록 생성 결과
Fig. 15. Genesis block creation result

부트 노드 생성 과정은 3단계로 나뉜다. 첫 번째로 제네시스 설정 파일을 작성하고, 두 번째로 제네시스 파일을 이용하여 최초 블록을 생성한 뒤, 사용자가 지정한 Network id에 접속한다. 마지막으로 Account를 생성한 뒤, 초기 Ether를 할당해 주기 위해 제네시스 파일의 alloc부분을 수정하고 제네시스 블록을 다시 생성한다. 그림 15는 부트노드에 제네시스 블록을 생성한 결과이다. 두 번째 단락을 보면, 설정 파일의 ChainId의 값인 20171730을 network id 값으로 세팅하여, 두 번째 노드를 해당 네트워크에 연결한다.

2) 노드 추가 및 정보 전달을 위한 채굴 시작

부트 노드와 2개의 피어 노드를 생성한 후, 부트 노드에 피어 노드를 연결하기 위해 admin.nodeInfo.enode 명령을 사용해 부트 노드의 enode를 확인한 뒤, 피어 노드에서 Geth를 구동시키고 addPeer로 부트 노드와 P2P연결을 한다. 연결 명령은 admin.addPeer(“부트 노드 enode”) 이다. 그림 16은 두번째 노드를 연결한 상황을 나타낸 것이다.

현재 부트 노드와 피어 노드들이 P2P연결이 되어있기 때문에 피어 노드는 부트 노드를 제외한 다른 노드의 존재를 확인할 수 없다. 따라서 연결 노드 목록의 전달을 위해 그림 17과 같이 부트 노드에서 채굴을 수행한다.

그림 18은 채굴 노드로 node3을 지정하고, PC3에서node4를 추가로 생성해 부트 노드에 연결한다. 같은 호스트에서 여러 노드를 구현할 경우 port번호를 달리하고, IPC-RPC를 disable한다. 이처럼 타 노드를 채굴 노드로 선정하면, 부트 노드가 채굴하지 않아도 새로운 노드의 존재를 확인할 수 있다.

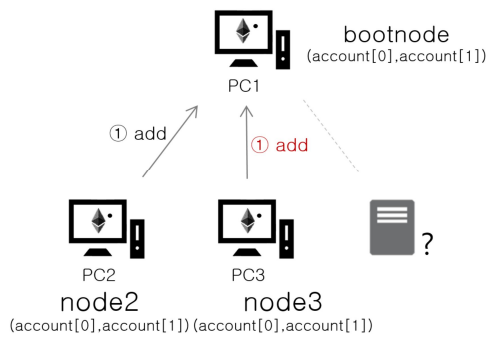

그림 16. 피어 노드와 부트 노드 연결
Fig. 16. Connection for peer and boot node

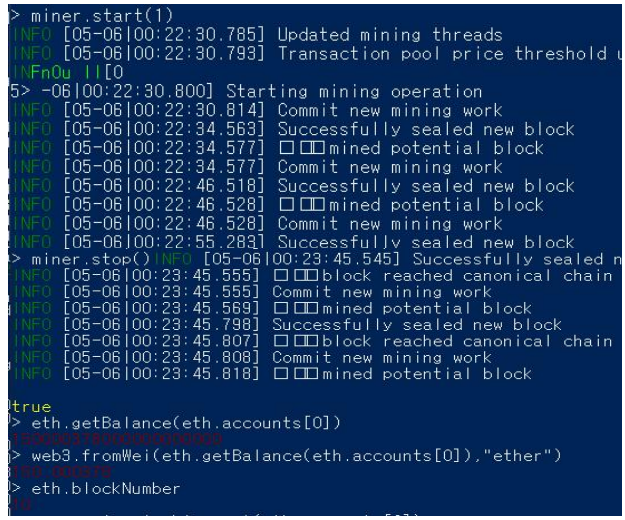

그림 17. 연결된 노드의 정보 전달을 위한 채굴 시작
Fig. 17. Mining work for node information fowarding

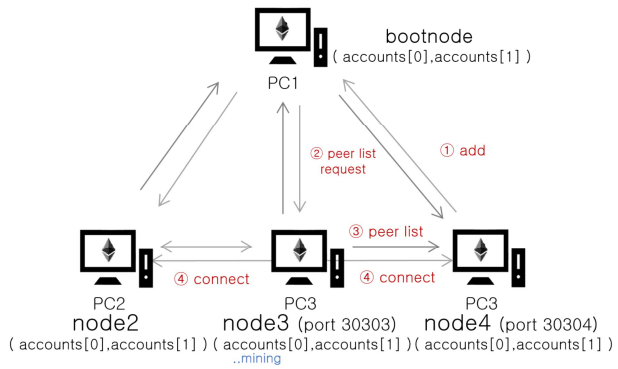

그림 18. 마이닝 노드 지정 및 피어 노드 추가
Fig. 18. Designation mining node and add peer node

4-2 블록체인 기반 중고거래 서비스 구현결과

그림 19는 본 연구에서 구축한 블록체인 기반 거래 마켓의 홈 화면이다. 상단부에는 로그인, 상품 업로드, 마이페이지 등의 메뉴가 위치하며, 아래에는 현재 거래중인 상품이 표시된다.

회원가입을 위해 먼저 이더리움의 계정정보를 획득해야 한다. web3.eth.getCoinbase(), web3.eth.getAccounts()는 체인의 계정 정보를 제공한다. 그림 20은 해당 web3 함수로 메타마스크와 연동하여 계정정보를 획득한 결과로 다양한 계정의 정보를 출력하며 사용자는 간단하게 계정을 선택하여 연동할 수 있다.

그림 21은 제안하는 서비스의 회원가입 페이지로, 메타마스크로 선택된 계정정보가 연동된 것을 확인할 수 있다. 회원정보에 추가로 이더리움 계정의 주소 값이 입력되어 있다.

그림 22는 상품을 선택하고 구매 버튼을 눌렀을 때 실행되는 구매 기능을 보여준다. 특정 상품의 가격이 오른편 메타마스크를 통해 확인되며, 승인했을 경우 거래를 위해 요구되는 수수료가 포함한 가격이 Locker 계정으로 송금된다.

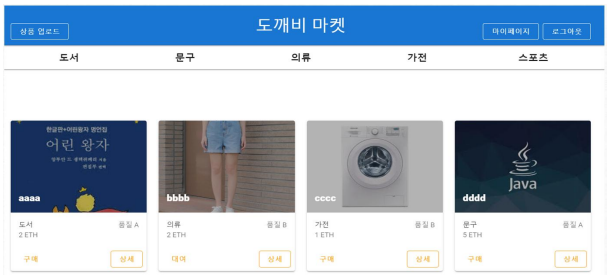

그림 19. 블록체인 기반 거래 마켓 구현 결과
Fig. 19. Implementation result of proposed block-chain based trading market

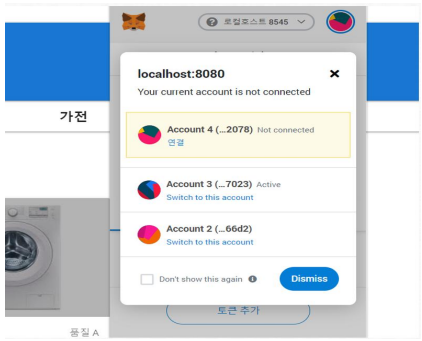

그림 20. 메타마스크를 이용한 계정정보 획득
Fig. 20. Acquiring account data with metamask

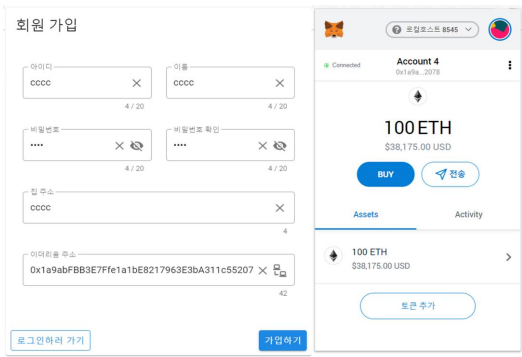

그림 21. 계정정보 연동 결과
Fig. 21. Account data acquisition results

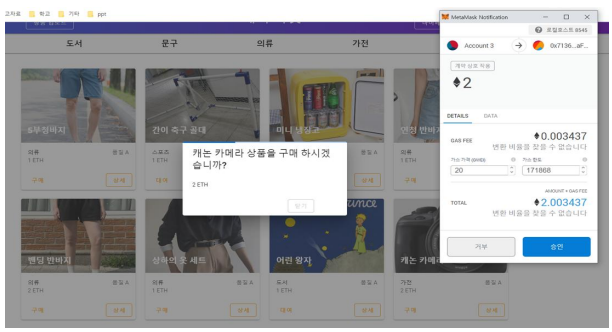

그림 22. 상품 구매 기능
Fig. 22. Product purchase function

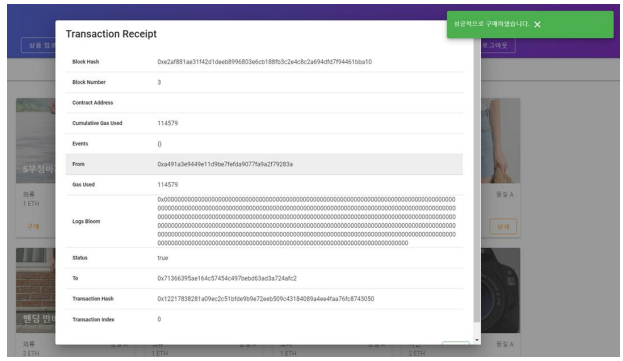

그림 23. 송금 트랜잭션 후 영수증 발급
Fig. 23. Issuance of transaction receipt of remittance

그림 24. 상품 수령 후, 거래 완료 과정
Fig. 24. Transaction completion after product receive

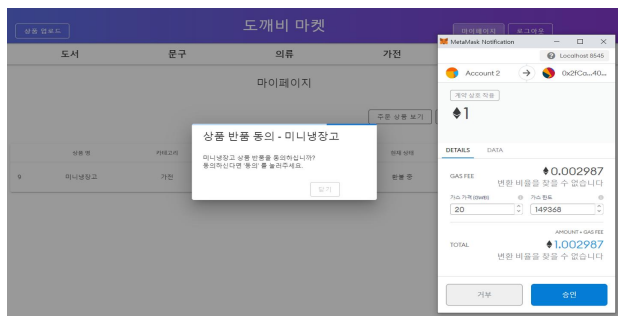

그림 25. 판매자의 반품 동의 과정
Fig. 25. Seller's return agreement phase

그림 23은 송금이 완료된 후 결과를 확인할 수 있는 송금 영수증이 발급된 모습이다. 이 영수증에는 해당 트랜잭션이 추가된 블록의 순서, 해시값, 사용된 가스 값과 전달된 사용자의 해시 값 등을 확인할 수 있다.

송금이 완료되었다면, 판매자는 실제로 제품을 보내고, 구매자는 제품을 받은 후 그림 24와 같이 거래 완료를 위한 제품 상태 동의 과정을 거친다. 이때, Locker 계정에서 판매자에게 구매 금액이 송금된다.

만약 상품 상태가 예상한 바와 달라 구매자가 거래 완료 동의를 하지 않을 경우, 반품과정이 진행된다. 반품이 신청되면, 그림 25와 같이 판매자에게, 반품의 동의를 구하며, 승인할 경우 환불 절차가 수행된다. 메타마스크를 통해 송금이 요청된다. 구매와 마찬가지로 송금 후에는 트랜잭션 영수증이 출력된다.

그림 26. 마이페이지(구매내역) 화면
Fig. 26. Mypage(Purchase list)

그림 27. 상대방 사용자 정보 출력 화면
Fig. 27. Other party information of a certain transaction

그림 26은 특정 사용자가 수행한 거래 내역을 출력하는 화면이다. 자신이 등록한 상품 목록을 출력하고 있으며, 앞서 환불을 진행한 건으로, 상태 값에 따라 환불 완료로 확인된다.

그림 27은 특정 거래의 구매자 정보를 선택했을 때 상대방 사용자의 정보가 출력되는 화면이다. 아이디, 이름, 주소와 이더리움 지갑의 계정 주소가 출력되어 확인할 수 있다.

Ⅴ. 결 론

본 논문에서는 블록체인 기반의 신뢰적 거래 플랫폼을 개발하였다. 블록체인은 체인을 형성하는 참여 노드가 블록을 유지하며 직접 검증을 수행하므로, 탈중앙성, 불변성을 가지므로, 부인방지에 효과적인 신뢰적인 플랫폼 개발에 알맞다. 하지만 체인을 구성하는 블록의 용량 한계와 블록 초과 시 발생하는 지연이 큰 단점을 가지고 있다. 따라서 제안된 블록체인 기반의 신뢰적 거래 플랫폼은 정보를 두 가지로 구분하여 처리하도록 고안되었다. 먼저 상품설명, 관련 사진과 같이 용량이 크고 지속적 변화 가능성이 있는 가변정보는 비교적 처리 속도가 빠르고 간편한 데이터베이스 서버를 활용하며, 거래 및 송금 정보와 같이 상대적으로 정보량은 적으면서, 최대한의 신뢰성을 가져야 하는 불변 정보는 블록체인에 저장하도록 하였다.

본 논문에서는 이를 처리하기 위한 데이터베이스 및 블록의 구조를 설계하고 블록체인 기반 서비스의 회원가입, 로그인, 상품정보 등록/갱신, 물품구매, 구매확정 및 환불, 물품 대여, 반납 등의 처리절차를 제안하였다. 제안된 절차 중 대여 혹은 판매시 물품 대금을 직접 송금하지 않고, 보증금 형태로 Locker 계정에 예치하는 과정을 추가하여 신뢰성을 높였다. 또한, 제안된 플랫폼 구현을 위해, 3대의 독립적인 시스템에서 동작하는 이더리움 기반의 프라이빗 네트워크를 구축하였고, 결과적으로 제안된 서비스 절차를 스마트 컨트랙트로 구현하여 블록체인 기반 중고거래 서비스 DApp을 구현하였다. 구현된 서비스는 프라이빗 이더리움 네트워크에 생성된 계정정보를 기반으로 상품의 구매, 대여, 환불 등의 기능을 제공하며, 판매자/구매자의 요청에 따라 물품 대금을 송금한다. 이러한 구매/송금내역은 블록체인 네트워크에 저장되며, 불변성을 가지므로, 신뢰적인 거래가 가능함을 보였다. 향후 클라우드 기반의 블록체인 네트워크를 통한 IoT 서비스의 신뢰 항상 기법에 관해 연구할 예정이다.

감사의 글

이 논문은 2021년도 정부 (과학기술정보통신부)의 재원으로 정보통신기획 평가원의 지원을 받아 수행된 연구임 (No. 2020-0-00833, 5G 기반 지능형 IoT 트러스트 인에이블러 핵심 기술 연구)

참고문헌

[1] S. H. Kang, “The spread of space-sharing services and further challenges,” *KIIE Industrial Engineering Magazine*, Vol. 26, No. 2, pp. 26-30, June 2019.

[2] J. I. Ryu, “Internet used market transactions are rapidly growing...Set new trends after corona,” *Korea Daily*, [Internet]. Available: http://www.koreadaily.com/news/read.asp?art_id=8411431

[3] D. H. Kang, “Non-face-to-face used product transaction surge due to COVID-19... good to save money, but worried about fraud," *Yonhapnews,* [Internet]. Available: https://www.yna.co.kr/view/AKR20200611140300505

[4] Ethereum, [Internet]. Available: https://ethereum.org/en/

[5] Block.one, “EOS (Everyone's Open Society),” [Internet]. Available: https://eos.io/

[6] Algorand, “Proof of Work, Proof of Stake & Pure Proof of Stake: An Evolution in Distributed Consensus,” [Internet]. Available: https://www.algorand.com/resources/blog/proof-of-stake-vs-pure-proof-of-stake-consensus

[7] Klaytn, “The Ground for All Blockchain Services,” [Internet]. Available: https://www.klaytn.com/

[8] Klaytn, “Klaytn position paper v2.1,” [Internet]. Available: https://www.klaytn.com/developers

[9] Ganache, “Ganache one click blockchain,” [Internet]. Available: https://www.trufflesuite.com/ganache

[10] Metamask, “A crypto walley & gateway to blockchain apps,” [Internet]. Available: https://metamask.io/

[11] Truffle suite, “Smart contracts made sweeter,” [Internet]. Available: https://www.trufflesuite.com/truffle

[12] Poiemaweb, “Modularization and npm,” [Internet]. Available: https://poiemaweb.com/nodejs-npm

[13] ZeroCho.com, “Express Framwork,” [Internet]. Available: https://www.zerocho.com/category/NodeJS/post/5789e8d91c9e1ff02bdedad3
[14] Velopert.log, “Introductionto token-based authentication,” [Internet]. Available: https://velopert.com/2350
[15] K. S. Min, J. W. Lee and B. J. Lee, “Improving Reliability of Smart Contracts and DApps by Applying Property-based and Model-based Test Methods to Different Test Levels,“ *Journal of KIISE,* Vol. 46, No. 8, PP. 763-773, August 2019.
[16] C. J. Kim, “A Static and Dynamic Design Technique of Smart Contract based on Block Chain,” *Journal of the Korea Academia-Industrial cooperation Society,* Vol. 19, No. 6, pp. 110-119, June 2018.
[17] HAECHI AUDIT, “Smart Contract, to properly understand with one article,” [Internet]. Available: https://medium.com/haechi-audit-kr/smart-contract-a-to-z-79ebc04d6c86
[18] Soliditylang.org, “Contract ABI Specification,” [Internet]. Available: https://docs.soliditylang.org/en/v0.5.3/abi-spec.html#contract-abi-specification

최환석(Hoan-Suk Choi)

2009년 : 한밭대학교 멀티미디어공학과 (공학사)
2011년 : 한밭대학교 멀티미디어공학과 (공학석사)
2018년 : 한밭대학교 멀티미디어공학과 (공학박사)

2015년～현 재: 한국 ITU연구위원회 국제표준전문가
2018년～2020년: 한밭대학교 멀티미디어공학과 박사후연구원
2020년～현 재: 한국과학기술원 전기및전자공학부 위촉연구원
2021년～현 재: 주식회사 토브데이터 매니저
※관심분야 : IoT, Social IoT, Semantic Processing, Trust management, Distributed ledger, Trust Chain, 개인정보보호, GDPR

박민영(Min-Young Park)

2021년 : 한밭대학교 정보통신공학과 (공학사)

2018년～2021년: 한밭대학교 정보통신공학과 학부연구생
※관심분야 : Blockchain, Ethereum, Smart contract, IoT, Big data, Data Analysis

송유빈(You-Bin Song)

2021년 : 한밭대학교 정보통신공학과 (공학사)

2018년～2021년: 한밭대학교 정보통신공학과 학부연구생
※관심분야 : Blockchain, Ethereum, Smart contract, IoT, Big data, Data Analysis

이우섭(Woo-Seop Rhee)

1983년 : 홍익대학교 (공학사)
1995년 : 충남대학교 (공학석사)
2003년 : 충남대학교 (공학박사)

1983년～2005년: 한국전자통신연구원 팀장/책임연구원
2005년～현 재: 한밭대학교 정보통신공학과 교수
2006년～현 재: 한국ITU연구위원회 국제표준전문가
2012년～2013년: 프랑스 Institute TelecomSudParis 방문교수
2018년～2019년: 영국 Liverpool John Moores University 방문교수
※관심분야 : Semantic Processing, Trust Management, Trust chain, IoT, Social IoT