Journal of KIIT. Vol. 18, No. 5, pp. 109-119, May 31, 2020. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2020.18.5.109

블록체인과 오픈뱅킹 API를 이용한 부동산 임대차 거래 시스템

손민성*, 김희열**

A Real Estate Lease Transaction System Using Blockchain and Open Banking API

Minsung Son*, Heeyoul Kim**

이 논문은 2018년도 정부(과학기술정보통신)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No.2018R1C1B6002903), 본 연구는 2019년 경기대학교 대학원 연구원장학생 장학금 지원에 의하여 수행되었음.

요 약

블록체인의 등장과 발전에 따라 제 3자의 보증 없이 신뢰할 수 없는 사용자 간의 거래가 가능해 졌다. 이에 따라 부동산 거래와 블록체인을 연계하여 거래를 최소화 하고 편의성과 안정성을 향상시키고자 하는 연구가 진행되고 있다. 하지만 거래의 핵심이 되는 금융과의 연계와 금융 시스템과 블록체인 시스템 간의 오라클 문제에 대한 논의는 부족하다. 따라서 우리는 오픈뱅킹 API를 통해 블록체인과 금융과의 연계 방법에 대하여 제안한다. 또한, 하이퍼레저 패브릭의 결정론적 블록 검증 과정을 이용해 오픈뱅킹 API로 부터의 이체 확인 메시지에 대한 확정성을 얻음으로써 오라클 문제를 해결한다. 이를 통해 블록체인을 이용한 종이 없는 계약을 가능하게 하며 계약의 내용뿐만 아니라 금융 거래의 내역을 블록체인에 저장하여 거래의 견고성을 높일 수 있다. 또한 사용자 친화적인 UX를 제공함으로써 거래의 편의성을 증진시킬 수 있다.

Abstract

With the advent and development of blockchain, transactions between untrusted users have become possible without third party guarantees. Accordingly, research is underway to minimize transactions and improve convenience and stability by linking real estate transactions with blockchain. However, there is a lack of discussion about the connection between finance, which is the core of the transaction, and Oracle issues between the financial system and the blockchain system. Therefore, we propose a method of linking blockchain and finance through an open banking API. It also solves the Oracle problem by obtaining determinism of transfer confirmation messages from the OpenBanking API using the Hyperledger Fabric's deterministic block verification process. Through this, it is possible to make a paperless contract using the blockchain, and it is possible to increase the robustness of the transaction by storing the details of the contract as well as the details of the financial transaction on the blockchain. In addition, by providing user-friendly UX, it is possible to enhance the convenience of transactions.

Keywords
blockchain, openbanking, real estate, hyperledger fabric

* 경기대학교 컴퓨터과학과 석사과정
- ORCID: https://orcid.org/0000-0002-1022-7669
** 경기대학교 컴퓨터과학과 교수(교신저자)
- ORCID: https://orcid.org/0000-0001-8776-5185

· Received: Mar. 17, 2020, Revised: May 15, 2020, Accepted: May 18, 2020
· Corresponding Author: Heeyoul Kim
Dept. of Computer Science, Kyonggi University, 94-6 Iuidong, Yeongtonggu, Suwon, Gyeonggi, 443-760, Korea,
Tel.: +82-31-249-9675, Email: heeyoul.kim@kgu.ac.kr

Ⅰ. 서 론

한국에서 가계자산 중 부동산이 차지하는 비율은 약 70%이며 매해 꾸준히 증가하는 추세이다[1]. 이에 따라 정부는 안전하고 합리적인 부동산 거래, 부당하지 않은 계약을 위한 주택, 상가 건물 등 다양한 유형에 따른 표준 계약서를 배포하고 등기부등본, 토지대장을 민간에서 확인할 수 있도록 공개하여 사기 피해를 줄이고자 노력하고 있다. 한국에서 기존의 임대차 계약에서는 공인 중개사를 통해 위임한 매물에 대한 특약 사항을 구체적으로 합의한 후 보증금과 계약금, 그리고 중도금을 정한다. 합의된 계약 내용은 집주인과 임차인이 각각 한 장씩 가질 수 있도록 계약서 두 장을 작성하고 이를 서명한 후 나눠 갖는다. 계약서를 작성한 이후에는 임차인이 계약을 이행하기 위해 집주인에게 약속된 보증금과 계약금을 송금을 해야 하며 이 과정은 월세와 같이 계약 종류에 따라 반복될 수 있다. 하지만 이러한 기존의 부동산 거래 시스템은 많은 사람들이 부동산 계약의 많은 종이서류와 어려운 프로세스로 인해 계약을 진행하는데 어려움을 겪고 있다.

부동산 거래를 위한 어플리케이션은 존재하고 있다. 하지만 이러한 어플리케이션은 거래자와 판매자의 중간자 역할만을 제공하며 실제로 거래를 진행하기 위해서는 대면을 통한 거래가 필요하다. 실제로 시중에 상용화 되어있는 부동산 거래 어플리케이션을 확인해 본 결과 직접적인 구매는 불가능하며 판매자에게 문자 또는 통화를 연결해 주는 기능만을 제공하고 있다. 이는 결국 사용자와 판매자의 중계자로서의 역할을 가능하지만 기존의 부동산 거래 프로세스를 따라야 한다는 문제점을 가지고 있으며, 부동산 거래에 어려움을 느끼는 사용자들에게 편리함을 줄 수 없다. 또한 계약 또한 종이 기반으로 이루어지기 때문에 종이 없는 계약을 수행할 수 없다는 문제점이 존재한다.

위의 문제점을 해결하기 위해 우리는 블록체인 기반의 부동산 거래 시스템을 제안한다. 제안하는 모델은 블록체인의 스마트 컨트랙트를 이용하여 종이 없는 계약(Paperless contract)을 실현하고 사용자의 편의성을 증진시키며 모든 거래내역 및 계약서의 무결성이 보증됨에 따라 계약의 부인방지, 허위계약 등 다양한 거래상의 문제점을 해결할 수 있다. 또한 계약자 간에 계약서를 나눠 갖는 문서 작업 과정을 종이 없는 디지털 거래로 진행하면 문서 분실의 위험성을 없앨 수 있으며 더욱 효율적인 계약이 가능하다. 우리는 기존의 연구와는 다르게 한국의 은행권 공동 공개데이터인 오픈뱅킹 API를 사용하여 단순히 계약서만이 아닌 계약에 관련된 모든 거래내역을 블록체인에 저장하여 거래 내역 확인 및 부인방지를 통해 사기 피해를 최소화 할 것이다.

Ⅱ. 배경 지식

2.1 블록체인

블록체인 기술은 2009년 Nakamoto Satoshi에 의해 처음 보여진 기술[2] 이며 저장된 데이터의 무결성, 영속성, 투명성, 부인방지를 만족시켜 준다. 이러한 블록체인 기술의 특징은 [3][4]와 같이 여러 분야에서 다양한 활용을 가능하게 하였다.

블록체인은 퍼블릭 블록체인과 프라이빗 블록체인으로 나눌 수 있다. 퍼블릭 블록체인은 만들어진 블록에 대해 PoW(Proof of Work), PoS(Proof of Stake)와 같은 합의 알고리즘을 이용하여 네트워크 참여자들 사이의 블록에 대한 Safty를 만족시킨다. 또한 네트워크 참여자에 대한 제한을 두지 않기 때문에 높은 확장성을 가진다. 하지만 이 방법은 낮은 TPS(Transaction Per Second)를 갖는 문제점이 있어 실제 서비스에서 사용하기 어려움을 겪고 있다. 이에 반하여 프라이빗 블록체인은 네트워크 참여자를 신뢰할 수 있는 노드들의 묶음으로 제한하며 PBFT[5], IBFT[6], RAFT[7]와 같은 합의 알고리즘을 사용해 높은 TPS를 갖지만 낮은 확장성을 갖는 문제점을 가지고 있다.

2.2 스마트 컨트랙트

스마트 컨트랙트[8]는 1997년 Nick Szabo에 의해 발의된 개념으로 암호화 및 기타 보안 매커니즘을 사용하여 컴퓨터를 이용한 완전한 계약을 가능하게

하는 내용을 담고 있다. 이 개념은 당시에는 각광받지 못하다가 2015년 블록체인 시스템에 스마트 컨트랙트 개념을 추가한 Vitalik Buterin의 이더리움이 등장하면서 각광받기 시작했다. 블록체인에서의 스마트 컨트랙트는 블록체인 네트워크의 스마트 컨트랙트의 소스와 스마트 컨트랙트를 구동하기 위한 파라메터, 그에 의해 변하는 상태에 대한 내용이 모두 기록되고 변조가 불가능 하기 때문에 그 의의를 갖게 된다. 이를 통해 블록체인 네트워크의 사용자 사이의 신뢰가 존재하지 않더라도 제 3자의 보증 없이 계약을 진행할 수 있게 한다.

2.3 하이퍼레저 페브릭와 오더링 시스템

하이퍼레저 페브릭[9]은 IBM의 주도 하에 개발이 이루어지고 있는 오픈소스 기반 프라이빗 블록체인 플랫폼이다. 하이퍼레저 페브릭에서는 오더링 방식의 합의를 진행하고 있으며 이는 퍼블릭 블록체인에서 사용하는 기존의 합의 방식과 같이 확률론적 접근을 통한 합의를 진행하지 않고 결정론적 합의 모델을 갖는다. 이는 곧 트랜잭션의 정렬과 빠른 TPS를 가능하게 한다. 하이퍼레저 페브릭의 오더링 시스템이 트랜잭션을 검증하는 방식은 그림 1과 같다.

어플리케이션이 발생시킨 트랜잭션은 오더링 시스템에 전송되며 일정 시간 혹은 일정 개수의 트랜잭션이 모이면 오더링 시스템은 모인 트랜잭션을 이용하여 블록을 생성하고 이를 endorser라고 불리는 검증자들에게 전송한다.

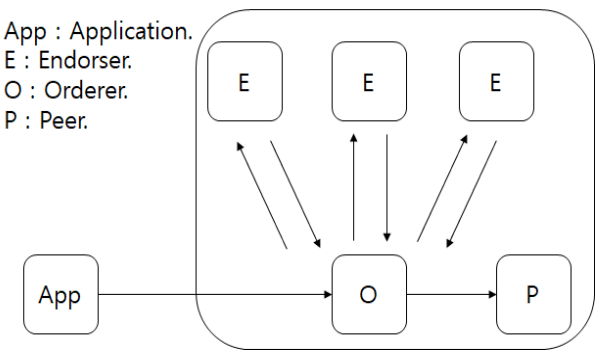

그림 1. 오더링 시스템의 트랜잭션 검증 방식
Fig. 1. Transaction verification method of ordering system

검증자는 전달받은 블록의 트랜잭션을 실행시키며 자신의 원장의 상태 변화값을 담아 오더링 시스템에게 되돌려 준다. 오더링 시스템은 이 데이터가 모두 같은지 확인하여 트랜잭션을 통한 상태의 변화가 결정적으로 이루어질 수 있도록 한다.

2.4 오픈뱅킹 API

오픈뱅킹[10]이란 핀테크 기업이 금융 서비스를 편리하게 개발할 수 있도록 은행의 금융서비스를 표준화된 형태로 제공하는 인프라를 의미한다. 금융기관 대신 고객의 금융정보를 조회하거나 대신 이체를 진행할 수 있게 하는 등의 서비스를 제공하고 있다. 한국에서는 2019년 10월부터 시범 운영을 시작하였으면 같은 해 12월부터 정식으로 서비스를 시작하였다. 우리는 제안하는 모델에서 오픈뱅킹 API를 사용하여 결제를 가능하게 할 것이며 그 증거 데이터를 얻어 올 것 이다.

2.5 기존의 블록체인 기반 부동산 거래 시스템

블록체인과 부동산 거래를 연계하고자 하는 연구는 계속되어 왔다(e.g. [11][12]). 하지만 이러한 연구는 한국의 부동산 및 금융시스템에 초점을 두고 있지 않으며 그 방법을 명확하게 서술하지 않고 있는 문제점을 가지고 있다. [13]는 블록체인과 스마트 컨트랙트에 대한 간략한 설명과 부동산 거래에서의 블록체인 사용 예시를 보여주고 있다. 이더리움 네트워크를 기반으로 하고 있으며 이더리움 계정 즉 EOA(Externally Owned Account)를 이용하여 임대인 및 임차인을 네트워크에 참여할 수 있도록 한다.

또한, 임대인과 임차인 사이의 거래를 스마트 컨트랙트를 이용하여 진행할 수 있도록 하였다. 이를 가능하게 하기 위해 계약서에 임대인과 임차인의 서명을 동시에 서명하는 방식을 통해 계약서 내용에 대한 합의를 도출했으며 블록체인의 무결성과 부인방지의 특성을 이용하여 이 계약을 확정시켰다.

하지만 이 논문은 프로세스 임대 계약의 '지불'에 대한 방법과 증명을 위한 명확한 방법을 제시하

지 않았으며 단지 '여러 가지 방법을 통한 지불을 한다.' 라고 명시하고 있다. 따라서 이 모델을 통해서는 정상적으로 계약서대로의 지불이 이루어졌는지 확인할 수 있는 방법이 존재하지 않는다.

III. 제안 모델

본 절에서는 제안하는 모델의 계층 모델과 구조 및 전체적인 동작 방식을 살펴볼 것 이다. 제안하는 모델은 스마트 컨트랙트를 통한 거래의 성사뿐만 아니라 오픈뱅킹 API를 사용하여 거래간의 비용 지불 또한 자동적으로 수행하게 한다. 이 내역들은 블록체인 네트워크에 기록되며 이를 통해 거래의 부인 방지 및 사기 피해의 최소화를 가능하게 한다.

우리 모델의 최종적인 목표는 아래와 같다.

- 종이 없는 계약 : 제안하는 모델은 기존 임대차 계약에서의 불필요한 종이 문서를 최소화 하여 거래의 복잡성을 줄이고 사용자의 편의성을 증진시키고자 한다. 따라서 전체적인 계약 흐름에 있어서 기존의 계약 방식보다 종이 문서의 이용량이 적어야 한다.
- 지불된 비용의 증거 : 제안하는 모델은 오픈뱅킹 API를 이용하여 지불을 가능하게 하며 그 증거를 블록체인에 저장하여 거래를 증명하고자 한다. 따라서 제안하는 모델은 블록체인에 올라간 지불의 증거가 믿을 수 있는 방법을 통해 업로드된 것인지 확인할 수 있어야 한다.

3.1 계층 모델

제안하는 모델의 계층구조는 그림 2와 같다. 블록체인 플랫폼은 하이퍼레저 페브릭으로 하며 합의 알고리즘은 오더링 시스템을 사용할 것이다. 거래를 위해 하이퍼레저 패브릭의 스마트 컨트랙트 기능을 이용하며 결재와 결제의 증거를 가져오기 위해 오픈뱅킹 API를 사용할 것이다. 또한 제안하는 모델을 사용하는 여러가지 어플리케이션들이 만들어 질 수 있다.

3.2 시스템 모델

제안하는 모델은 하이퍼레저 페브릭 기반의 블록체인 네트워크로 이루어져 있다. 이때 거래를 원하는 유저는 네트워크 가입 절차를 통해 Fabric CA에서 발급받은 개인키를 기반으로 하여 임차인 및 임대인이 될 수 있으며 자신임을 증명하게 된다.

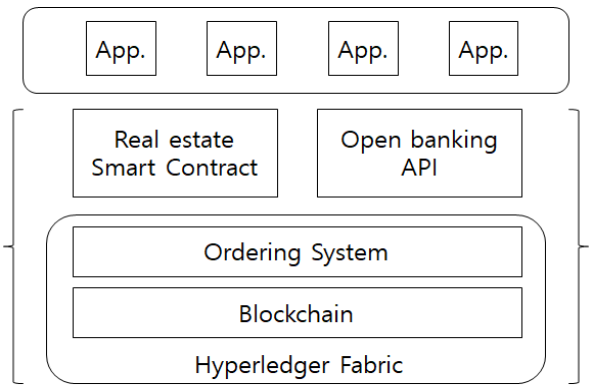

그림 2. 제안 모델의 계층모델
Fig. 2. Hierarchical model of the proposed model

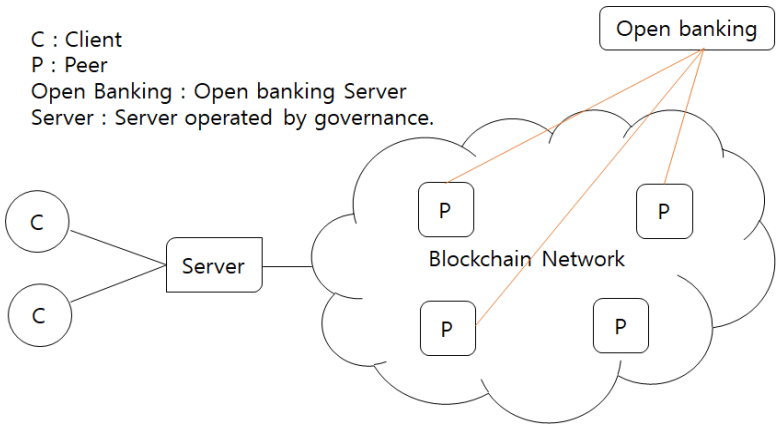

그림 3. 시스템 모델
Fig. 3. System model

블록체인 네트워크를 이루는 피어는 국토교통부, 한국 부동산 개발 협회, 공정거래 위원회 등 신뢰 있는 기관들이 될 수 있으며, 이 기관들을 모아 거버넌스 형태로 블록체인 네트워크를 운영하도록 한다. 또한 오픈뱅킹 API는 금융거래가 진행될 때에만 사용되기 때문에 언제나 피어와 연결되어 있을 필요가 없다.

현재 오픈뱅킹 API는 REST API 방식을 취하고 있기때문에 HTTP를 사용하여 통신한다. 따라서 통신이 필요할 때 적절한 메소드를 사용하여 통신을 하도록 한다. 또한 사용자의 시스템 사용 편의성을 위해 모델은 어플리케이션을 지원해야 한다. 이는 웹 또는 모바일 어플리케이션으로 지원하며 거버넌스가 운영하는 중앙 서버를 통해 한다.

3.3 사용자 등록

제안하는 모델을 사용하고자 하는 사용자는 거버넌스가 제공하는 어플리케이션을 통해 사용자 등록을 진행하여야 하며 오픈뱅킹에도 가입하여야 한다. 오픈뱅킹 가입절차는 오픈뱅킹의 절차를 그대로 따르며 제안하는 모델의 사용자 등록 흐름은 다음과 같다.

사용자는 어플리케이션을 통해 회원가입 요청을 보낸다. 이때 어플리케이션에서 사용자에게 요청할 수 있는 데이터는 실명, 생년원일, 주소 와 같은 데이터들이 있으며 SMS 인증 방식이나 ARS 인증방식을 통해 사용자의 실명 인증 절차를 진행하게 한다. 모든 인증절차가 완료되면 Fabric CA를 통해 새로운 인증서를 생성하고 사용자에게 전송한다. 이를 통해 사용자는 본 시스템의 사용 권한을 얻을 수 있게 된다. 이때, 하이퍼레저 패브릭은 x.501 인증서 표준을 사용한다. 따라서 제3기관에서 발행하는 공인인증서를 본 시스템에서 그대로 사용할 수도 있다. 사용자가 공인인증서를 통한 가입을 시도하면 어플리케이션에서는 공인인증서 발급기관에 공인인증서의 유효 여부를 파악한다. 공인인증서가 정상이라고 판별되면 공인인증서의 공개키를 Fabric CA에 등록하고 사용자는 기존의 공인인증서를 본 모델에서 사용할 수 있게 된다.

3.4 매물 등록

사용자는 자신의 부동산에 대한 계약을 진행하기 위해 매물 등록 절차를 진행할 수 있다. 매물의 등록은 어플리케이션을 통해 진행되며 매물의 실사, 가격, 거래 내용을 입력하며 거버넌스가 운영하는 서버와 블록체인 네트워크에 저장된다. 이때 블록체인 네트워크에 저장되는 데이터는 매물의 실사를 제외한 데이터이다.

3.5 계약

본 절에서는 제안하는 모델의 계약에 대한 내용을 전반적으로 다룬다. 이때 사용하는 표기법은 표 1과 같다.

표 1. 표기법
Table 1. Notation

C	Lease agreement data
S	Seller
B	Buyer
pub_{α}	α's public key
$h()$	cryptographic hash function
T	Timestamp
σ_{α}	Signature of α
d_S	$[h(C \|\| pub_B)]_{\sigma_S}$
d_B	$[h(C \|\| pub_S)]_{\sigma_B}$
$OTRD$	Openbanking transfer response data
$OCRD$	Openbanking transfer check response data
tv	Transfer verification data from $OCRD$

3.5.1 계약 생성

구매자가 매물을 구매하고자 판매자에게 거래 요청을 보내면 판매자는 C, T, pub_B, d_S를 임대차 계약을 위한 스마트 컨트랙트로 전송한다. 데이터를 받은 스마트 컨트랙트는 전송받은 데이터를 기반으로 Agreement 구조체를 생성하고 블록체인에 저장한다. 이로써 매물 등록자와 매물 구매자간의 계약이 생성된다.

Agreement 구조체는 계약에 관련된 데이터들의 집합을 의미하며 C, pub_B, d_S, d_B, tv로 이루어져 있

다. d_S는 판매자가 계약서 C에 대해 구매자와의 계약을 동의했다는 것을 의미하며 d_B는 구매자가 계약서 C에 대해 판매자와의 계약을 동의했다는 것을 의미한다. 표 2에서는 아직 구매자가 계약에 동의하지 않고 입금 프로세스를 진행하지 않았으므로 d_B, tv를 null로 채워 넣는다.

표 2. Create Agreement 알고리즘
Table 2. Algorithm of create agreement

function name : CreateAgreement
Input : C, T, pub_B, d_S Output : 32bytes array *//hash value*
Algorithm
if (!verify(d_S)) return c = Agreement{$C, pub_B, d_S, null, null$} *//create a Agreement struct* h = $h(C,T)$ *//contract's primary key* putState(h, c) *//put in to blockchain ledger* return h

3.5.2 계약 체결

부동산 거래의 경우 현물거래가 필요하다. 따라서 기존의 블록체인 기반 부동산 거래 시스템은 블록체인에 계약서의 내용만 저장 하고 계약금 이체를 따로 진행하는 경우가 많다. 이는 곧 스마트 컨트랙트가 구매자와 판매자 사이의 현물거래가 이루어 졌음에 대한 증거는 제공하지 못한다는 것을 의미한다. 하지만 본 모델에서는 오픈뱅킹 API를 사용하여 거래금의 이체 및 확인을 자동화 시킬 것이다.

또한, 그 증거를 블록체인에 저장하여 거래가 정상적으로 진행됐음을 증명할 것이다. 우리는 계약 생성 절에서 판매자가 판매를 위한 데이터를 스마트 컨트랙트에 전송하는 것을 확인했다. 이 스마트 컨트랙트를 통해 매물 구매자는 전송된 계약서의 내용을 확인한 뒤 거래에 동의 할 수 있다. 동의를 하게 되면 서버는 사용자를 결제 페이지로 리다이렉트 시킨다. 결제 페이지로 리다이렉트된 사용자는 오픈뱅킹의 입금 이체 프로세스에 따라 입금을 진행하고 서버는 그 응답으로 그림 4와 같은 구조를 가지는 데이터를 받는다.


```
{
    "api_tran_id": "fce59fac-e567-48a6-a301
      -34893f83ce50",
    "rsp_code": "A0000",
    "rsp_message": "",
    "api_tran_dtm": "20200227232411089",
    "wd_bank_code_std": "097",
    "wd_bank_code_sub": "0970001",
    "wd_bank_name": "오픈은행",
    "wd_account_num_masked": "1234567890123***",
    "wd_print_content": "홍길동캐시백",
    "wd_account_holder_name": "김오픈",
    "res_cnt": "1",
    "res_list": [
        {
            "tran_no": "1",
            "bank_tran_id": "T991596400U27232409A",
            "bank_tran_date": "20190201",
            "bank_code_tran": "097",
            "bank_rsp_code": "000",
            "bank_rsp_message": "",
            "fintech_use_num":
              "199159640057870609850416",
            "account_alias": "테스트계좌",
            "bank_code_std": "088",
            "bank_code_sub": "0970001",
            "bank_name": "신한은행",
            "account_num_masked": "01043382***",
            "print_content": "오픈서비스캐시백",
            "account_holder_name": "손민성",
            "tran_amt": "500",
            "cms_num": ""
        }
    ]
}
```


그림 4. 입금 이체 응답 데이터
Fig. 4. Deposit transfer response data

이때 우리는 차후에 이체 결과를 조회하기 위해 위 데이터 중 bank_trand_id, bank_tran_date, tran_amt를 필요로 하며 이를 $OTRD$라고 정의한다.

위 결제 과정이 완료되면 $OTRD$와 계약 생성 과정에서 생성된 계약서의 기본키인 $h(C,T)$, 블록체인을 통해 받은 계약서의 데이터를 해쉬화 한 $h(C)$, 그 서명 데이터 $[h(C)]_{\sigma_B}$를 스마트 컨트랙트로 전송한다. 스마트 컨트랙트는 $h(C,T)$를 통해 Contract 구조체를 블록체인으로부터 불러오고 그 내부에 저장된 pub_B를 이용하여 $[h(C)]_{\sigma_B}$를 검증한다. 이는 구매자가 계약서 내용에 동의하였음을 의미한다. 그런 뒤 $OTRD$와 C를 이용하여 계약서와 이체된 금액이 일치하는지 확인한다. 마지막으로 $OTRD$를 이용하여 오픈뱅킹 API를 통해 이체가 실제로 진행되었는지에 대한 검증을 진행한다. 이 프로세스는 거래 검증 절에서 자세히 설명한다.

표 3. SignAgreement 알고리즘
Table 3. Algorithm of SignAgreement

function name : SignAgreement
Input : $OTRD, h(C, T), h(C), [h(C)]_{\sigma_B}$ Output : boolean
Algorithm

c = getState($h(C, T)$)
if(h(c.C) != $h(C)$) return false
if(signVerify($[h(C)]_{\sigma_B}$, c.pub_B) return false
if(amountVerify(c.C, $OTRD$) return false
 	
//Using OpenBanking API
tv = transferVerify($OTRD$)
c = Agreement{$C, pub_{B,} d_S$, $[h(C)]_{\sigma_B}$,tv}
 	
putState($h(C, T)$, c)
return true

3.5.3 결제 검증

우리는 결제가 정상적으로 진행되었는지를 확인하기 위한 프로세스를 진행하며 이 프로세스는 하이퍼레저 페브릭의 오더링 시스템을 기반으로 한다. 또한 전체적인 검증의 흐름은 그림 5와 같다.

결제 체결 절에서 우리는 $OTRD$를 스마트 컨트렉트로 전송하며 표 3의 알고리즘을 통해 오픈뱅킹 API와 $OTRD$를 이용해 이체 검증을 한다는 것을 확인하였다. 또한 우리는 2.3절에서 하이퍼레저 페브릭의 트랜잭션 검증 방식을 확인했다. 하이퍼레저 페브릭의 트랜잭션 검증 방식에 따르면 트랜잭션을 검증하는 모든 검증자 노드들의 state가 일치하지 않으면 에러를 발생시킨다. 즉 SignContract를 실행시키는 트랜잭션을 검증할 시 모든 검증자 노드에서 해당 알고리즘이 실행된다는 것을 의미하며 이는 검증자 노드의 수만큼 오픈뱅킹 API를 통해 이체 검증을 하게 되는 것을 의미한다. 이체 검증 쿼리는 각 검증자 노드로부터 따로 전송되며 그 응답 데이터도 검증자 노드들이 각자 받게 된다. 응답 데이터의 구조는 그림 6과 같다.

그림 6에서 볼 수 있듯이 응답 데이터는 이체 이력에 대한 데이터를 포함하고 있다. 우리는 이 데이터 중 req_list 데이터를 이체의 증거로써 남길 것이며 이 데이터를 $OCRD$라고 정의한다. 오픈뱅킹 서버에 이체 요청을 전송한 뒤 실제 이체가 발생하기 까지 지연이 있을 수 있다. 또한 각 검증자 노드가 개별적인 이체 확인 요청을 전송하기 때문에 검증자 노드 중 일부는 이체 결과 조회 데이터를 받지 못할 수도 있다.

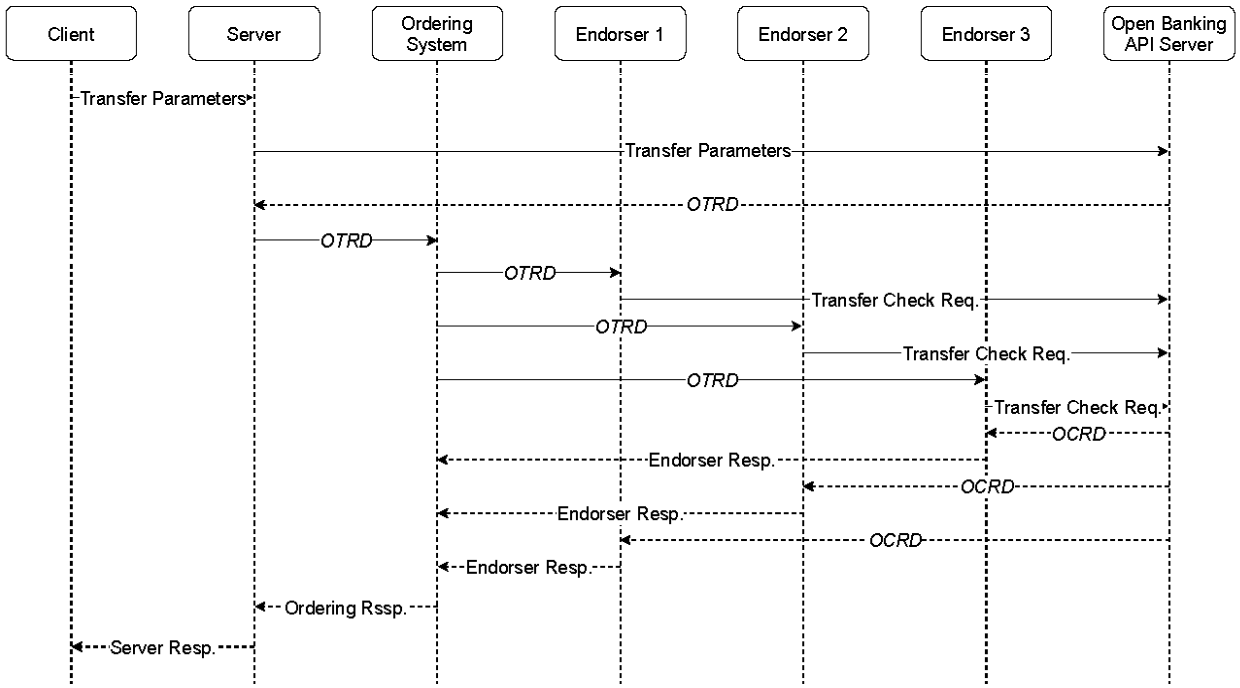

그림 5. 전체적인 검증 흐름
Fig. 5. Overall verification flow

```
{
    "api_tran_id": "7da86fc9-19b9-4627-900a-6073d723310b",
    "rsp_code": "A0000",
    "rsp_message": "",
    "api_tran_dtm": "20200221011234296",
    "res_cnt": "1",
    "res_list": [
        {
            "tran_no": "1",
            "bank_tran_id": "T991596400U20123223A",
            "bank_tran_date": "20200220",
            "bank_code_tran": "097",
            "bank_rsp_code": "000",
            "bank_rsp_message": "",
            "wd_bank_code_std": "097",
            "wd_bank_code_sub": "0970001",
            "wd_bank_name": "오픈은행",
            "wd_fintech_use_num": "",
            "wd_account_num_masked": "",
            "wd_print_content": "출금계좌인자샘플",
            "wd_account_holder_name": "",
            "dps_bank_code_std": "097",
            "dps_bank_code_sub": "0970001",
            "dps_bank_name": "오픈은행",
            "dps_fintech_use_num": "",
            "dps_account_num_masked": "",
            "dps_print_content": "입금계좌인자샘플",
            "dps_account_holder_name": "",
            "tran_amt": "500"
        }
    ]
}
```


그림 6. 이체 결과 조회 데이터
Fig. 6. Transfer result inquiry data

검증자 노드는 *OCRD*를 블록체인에 추가하여 state를 변경시키고 그 결과값을 오더링 시스템에게 다시 전송한다. 이때 위에 서술한 문제로 인하여 각 검증자에게라도 다른 데이터를 받게 된다면 결정론적이지 않게 됨을 의미하기에 해당 트랜잭션은 실패하게 된다. 이때 만일 해당 트랜잭션이 실패했다면, 오픈뱅킹 API 명세서에 따라 일정 시간이 지난 후 해당 트랜잭션을 다시 시도한다.

만일 트랜잭션이 성공한다면 계약서에 대한 구매자와 판매자의 서명과 오픈뱅킹 API로부터 전송받은 입금에 대한 증거가 블록체인 네트워크에 저장되었으므로 결제 프로세스가 종료된다.

Ⅳ. 구현 및 실험

본 절에서는 제안하는 모델의 구현을 통한 PoC를 진행한다. 구현 환경은 표 4와 같다.

하이퍼레저 페브릭은 1.4.6버전을 이용하였다. 블록체인 네트워크는 국토 교통부, 공정거래 위원회가 거버넌스가 된다고 가정하여 구축하였으며 이때 각각을 Org.1, Org2라고 부르도록 한다.

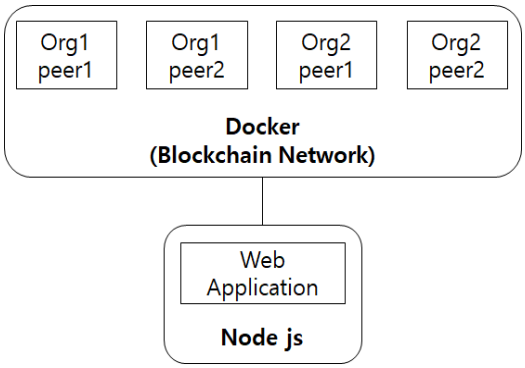

그림 7. 실험 환경 구성도
Fig. 7. Diagram of the experimental environment

표 4. 실험 환경
Table 4. Experiment environment

Component	Environment	Remark
Server	Windows10	Node Js
Hyperledger fabric	Ubuntu 16.04	v.1.4.6
Chaincode	Ubuntu 16.04	Go lang
Open banking API	Open banking server	Testnet

도커를 이용하여 네트워크를 구현하였으며 서버는 Node js를 이용하여 구현하였다. 또한, 체인코드는 go언어를 이용하여 구현하였다. 오픈뱅킹 서버는 실 서버를 사용하기에 제약이 있기 때문에 Testnet을 사용하여 실험을 진행하였다.

4.1 계약서 작성

본 구현에서 계약서는 실제 임대차 계약서를 html로 컨버팅하여 사용하였다. 필요한 부분은 판매자가 직접 채워 넣을 수 있으며 해당 내용들은 블록체인에 저장된다. 구매자가 해당 계약서를 확인하기 위해 서버에 요청하면 서버는 블록체인에서 해당 데이터를 받아와 컨버팅 된 임대차 계약서에 내용을 채워 넣고 구매자의 화면에 보여준다.

4.2 입금 확인

입금 확인 프로세스는 하이퍼레저 패브릭의 오더링 방식에 의거하여 작동한다. 구현에서 네트워크는 2개의 Org.으로 이루어졌기 때문에 그림 9를 보면 2개의 Org.에 체인코드가 설치된 것을 확인할 수 있으며 검증자 노드 또한 2개로 지정하였다.

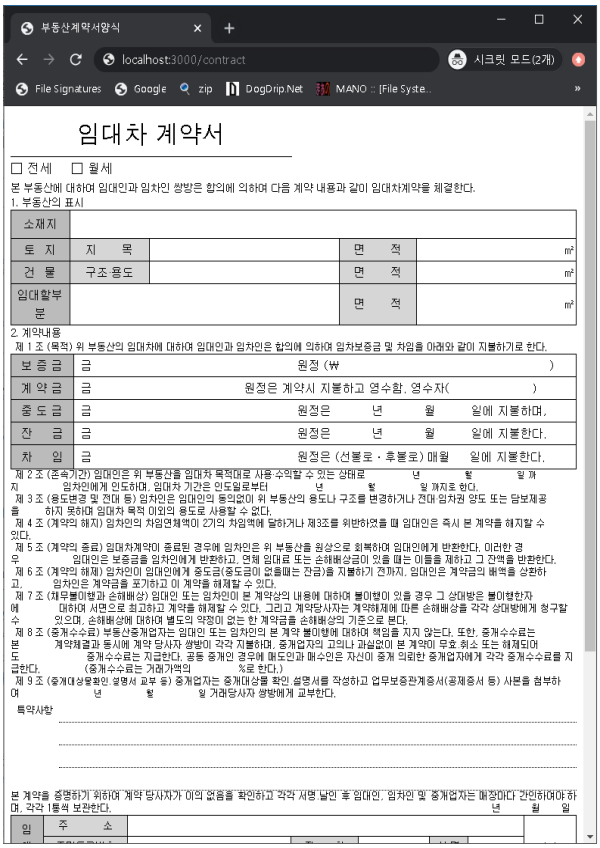

그림 8. 웹으로 컨버팅된 임대차 계약서
Fig. 8. Web-converted lease agreement

```
sms2831@islab-ProLiant-DL120-G7:~/2020/fabric-samples2/chaincode/fabcar/go$ docker ps
CONTAINER ID        IMAGE
PORTS                    NAMES
357deaa53ab7        dev-peer0.org2.example.com-mycc3-1.0-40cbd3a57fc842823c7b49280ae30cb4
                         dev-peer0.org2.example.com-mycc3-1.0
b170cb990db9        dev-peer0.org1.example.com-mycc3-1.0-d2balebd4d51fa69ec5354d3dbe4a00d
                         dev-peer0.org1.example.com-mycc3-1.0
```

그림 9. 인스턴스화된 체인코드
Fig. 9. Instantiated chaincode

우리는 앞에서 오더링 시스템에 의해 검증자 노드의 수만큼 오픈뱅킹 서버에 입금 확인 요청을 진행할 것 이라고 말하였다. 이를 검증하기 위해 체인코드 소스에 로그를 남기도록 하였다.

그림 10과 그림 11을 보면 각 Org.에서 실행시킨 체인코드에 대한 로그를 보여주고 있다. 제일 처음 입금 확인 요청을 진행하기 위한 *OTRD*를 보여주고 그에 대한 전체 응답 데이터와 응답데이터에서 추출한 *OCRD*(res_list)를 확인할 수 있다. 전체 응답 데이터에서 api_tran_id 는 각 요청마다 고유하게 매겨지는 식별자이다. 이때 그림 10과 그림 11의 식별자가 다른 것으로 보아 각 검증자 노드마다 다른 요청을 수행한 것을 확인할 수 있다.

```
sms2831@islab-ProLiant-DL120-G7:~$ docker logs 35
open banking request

"org_bank_tran_id": "T991596400U20123223A",
"org_bank_tran_date": "20200220",
"org_tran_amt": "500"
----------
open banking response

{"api_tran_id":"a938363f-5c05-426a-87d1-9ecf77ef8c9e","rsp_code":"A0000","rsp_message":"","api_tran_dtm":"20200305131435704","res_cnt":"1","res_list":[{"tran_no":"1","bank_tran_id":"T991596400U20123223A","bank_tran_date":"20200220","bank_code_tran":"097","bank_rsp_code":"000","bank_rsp_message":"","wd_bank_code_std":"097","wd_bank_code_sub":"0970001","wd_bank_name":"오픈은행","wd_fintech_use_num":"","wd_account_num_masked":"","wd_print_content":"출금계좌인자샘플","wd_account_holder_name":"","dps_bank_code_std":"097","dps_bank_code_sub":"0970001","dps_bank_name":"오픈은행","dps_fintech_use_num":"","dps_account_num_masked":"","dps_print_content":"입금계좌인자샘플","dps_account_holder_name":"","tran_amt":"500"}]}
----------
res_list

{1 T991596400U20123223A 20200220 097 000  097 0970001 오픈은행  출금계좌인자샘플  097  오픈은행  입금계좌인자샘플  500}
```

그림 10. Org.1의 체인코드 로그
Fig. 10. Chaincode logs of Org.1

```
sms2831@islab-ProLiant-DL120-G7:~$ docker logs b1
open banking request

"org_bank_tran_id": "T991596400U20123223A",
"org_bank_tran_date": "20200220",
"org_tran_amt": "500"
----------
open banking response

{"api_tran_id":"1cc9ae87-51a7-4dbd-9c37-05441bc63286","rsp_code":"A0000","rsp_message":"","api_tran_dtm":"20200305131408994","res_cnt":"1","res_list":[{"tran_no":"1","bank_tran_id":"T991596400U20123223A","bank_tran_date":"20200220","bank_code_tran":"097","bank_rsp_code":"000","bank_rsp_message":"","wd_bank_code_std":"097","wd_bank_code_sub":"0970001","wd_bank_name":"오픈은행","wd_fintech_use_num":"","wd_account_num_masked":"","wd_print_content":"출금계좌인자샘플","wd_account_holder_name":"","dps_bank_code_std":"097","dps_bank_code_sub":"0970001","dps_bank_name":"오픈은행","dps_fintech_use_num":"","dps_account_num_masked":"","dps_print_content":"입금계좌인자샘플","dps_account_holder_name":"","tran_amt":"500"}]}
----------
res_list

{1 T991596400U20123223A 20200220 097 000  097 0970001 오픈은행  출금계좌인자샘플  097  오픈은행  입금계좌인자샘플  500}
```

그림 11. Org2.의 체인코드 로그
Fig. 11. Chaincode logs of Org2

다음은 오픈뱅킹 서버로부터 서로 다른 값(입금 실패 등)이 오는 경우를 확인해 볼 것이다. 하지만 우리는 오픈뱅킹 테스트 넷을 사용하고 있기 때문에 이를 직접적으로 확인하기에는 어려움이 있다. 따라서 이를 확인하기 위해 의도적으로 체인코드를 비결정적으로 만들어 체인코드가 제대로 수행되는지 확인해 볼 것이다.

테스트 코드는 그림 12와 같다. 렛저의 state를 변경시키는 값을 랜덤하게 주어지게 함으로써 각 검증자 노드마다 서로 다른 state를 가지도록 하였다.

그림 12의 체인코드를 실행시킨 결과는 그림 13과 같다. could not assemble transaction: ProposalResponsePayloads do not match 라는 에러 메시지와 함께 실패하는 것을 볼 수 있다. 즉 만일 입금과정에 문제가 생겨 오픈뱅킹 서버로부터 잘못된 응답이 하나라도 온다면, 비 결정적인 결과가 도출됨으로써 거래에 실패하게 된다.

```
func test(stub shim.ChaincodeStubInterface, args []string) (string, error) {
        stub.PutState("test2", []byte(strconv.Itoa(rand.Intn(100))))
        return "", nil
}
```

그림 12. 비 결정적 체인코드
Fig. 12. Non-deterministic chaincode

```
Error: could not assemble transaction: ProposalResponsePayloa
ds do not match - proposal response: version:1 response:<stat
us:200 > payload:"\n \354\261\216x\216\377\215-0\275'\221\235
```

그림 13. 비 결정적 체인코드 응답
Fig. 13. Response of non-deterministic chaincode

V. 토 론

국토교통부에서는 블록체인 기술을 사용하지 않은 인터넷 부동산 거래 서비스인 '부동산 거래 관리 시스템'을 제공하고 있다. 따라서 우리는 부동산 거래시스템에서 블록체인 기술 도입에 대한 타당성에 대하여 논할 필요가 있다.

기존의 부동산 거래 관리 시스템은 공인 인증 전자서명, 부인방지 기술을 통해 부동산 거래의 타당성을 제공하고 있다. 또한, 부동산 거래의 데이터를 보관하기 위해 과학기술 정보 통신부 장관으로부터 지정받은 법인 또는 국가 기관인 공인전자문서센터를 운영하여 전자문서 보관의 효율성과 안전성을 보장하고 있다. 하지만 이러한 저장소는 부동산 거래 데이터의 무결성 및 영속성을 법적 기준을 통해 이를 해결하고 있다. 따라서 사용자는 부동산 거래 데이터의 보관에 대한 기술적인 무결성 및 영속성을 지속적으로 보장받기 충분하지 않다. 하지만 우리는 부동산 거래를 위해 블록체인을 이용한 시스템을 제안하였다. 이를 통해 사용자는 부동산 거래 데이터의 무결성 및 영속성을 현실적이고 효율적으로 보장받을 수 있으며 이를 통해 거래의 신뢰성을 더욱 향상시킬 수 있다. 따라서 부동산 거래 시스템에 대한 블록체인의 적용은 타당하다고 할 수 있다.

또한, 우리는 기존 시스템과 제안한 시스템과의 비교를 통해 제안모델의 우수성을 입증할 필요가 있다. 표 5는 기존 시스템과 제안모델의 비교 결과를 보여주고 있다.

제안모델과 [11]은 블록체인 기술을 기반으로 동작하기 때문에 거래 정보에 대한 무결성과 영속성을 기술적으로 보장받을 수 있다. 하지만 국토교통부의 경우는 무결성과 영속성을 법적으로 보장하고 있기 때문에 데이터 저장소에 문제가 생길 경우 사용자는 자신의 데이터의 영속성 및 무결성을 보장받을 수 없게 된다.

표 5. 기존 시스템과의 비교
Table 5. Comparison with existing systems

Model / Offer	MOLIT	[11]	Proposed model
Structure	Server.	Blockchain.	Server+ Blockchain
Integrity	Legal.	Technical	Technical
Permanence	Legal.	Technical.	Technical
Transfer verification	Technical.	No.	Technical

[11]의 경우 부동산 거래 시 발생하는 금융 거래에 대한 방안에 대해 명확하게 제시하고 있지 않다. 국토교통부의 경우 이러한 기능을 기술적으로 제공하고 있으나 서버 클라이언트 기반의 시스템에서 동작하고 있기 때문에 금융 거래 정보에 대한 무결성과 영속성을 완벽하게 보장하고 있지 않다. 하지만 제안모델은 오픈뱅킹 API를 통한 금융 거래 프로세스를 제공하고 금융 거래 정보를 스마트 컨트랙트 에서 검증하고 블록체인에 저장한다. 따라서 국토교통부와 다르게 금융 거래 정보에 대한 기술적인 무결성과 영속성 또한 보장받을 수 있다. 따라서 제안모델은 기존 시스템보다 우수하다고 볼 수 있다.

VI. 결 론

우리는 기존 부동산 거래의 어려움과 종이문서 최소화를 위해 블록체인을 이용한 부동산 거래 시스템을 제안했다. 기존의 블록체인을 이용한 부동산 거래 시스템과는 다르게 블록체인에 이체에 대한 증거를 남으로써 실제 금전적인 거래가 이루어 졌는지에 대한 확인 또한 가능하게 했다. 이를 가능하게 하기 위해 하이퍼레저 패브릭의 오더링 시스템을 이용하였으며 실제로 이를 통한 이체 검증이 가능한지에 대한 실험을 진행하였다. 이를 통해 이 시스템을 사용하는 사용자들은 부동산 거래에 대한 편의성이 증대될 것이며 이체 거래를 블록체인에

저장하기에 거래를 더 명확하게 할 수 있을 것이라고 예상한다.

References

[1] Bank of Korea and Statistical Office of Korea, "2018 National Balance Sheet", Jul. 2019.

[2] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", http://bitcoin.org/bitcoin.pdf, 2009.

[3] Jinsu Kim, jaeyoung Cho, and Namje Park, "Block Chain Based CCTV Image Forgery· Modulation Verification Mechanism", Journal of KIIT, Vol 17, No. 8, pp. 107-114, Aug. 2019.

[4] Donghyeok Lee and Namje Park, "CCTV Video Privacy Protection Scheme Based on Edge Blockchain", Journal of KIIT, Vol. 17, No. 10, pp. 101-113, Oct. 2019.

[5] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance", OSDI: Symposium on Operating Systems Design and Implementation, USENIX Association, Co-sponsored by IEEE TCOS and ACM SIGOPS, New Orleans, USA, pp. 173-186, Feb. 1999.

[6] https://github.com/ethereum/EIPs/issues/650. [accessed: Mar. 16, 2020]

[7] Diego Ongaro and John Ousterhout, "In Search of an Understandable Consensus Algorithm", 2014 USENIX Annual Technical Conference, Philadelphia, PA. USA, pp. 305-319, Jun. 2014.

[8] Nick Szabo, "Formalizing and securing relationships on public networks", First Monday, Vol. 2, No. 9, Sep. 1997. https://doi.org/10.5210/fm.v2i9.548.

[9] Elli Androulaki et al., "Hyperledger fabric: A distributed operating system for permissioned blockchains", EuroSys '18: Proceedings of the Thirteenth EuroSys Conference, Porto Portugal, Article No. 30, pp. 1-15, Apr. 2018.

[10] https://www.open-platform.or.kr/apt/content/openplatform?location=openPlatform1. [accessed: Mar. 16, 2020]

[11] Ioannis Karamitsos, Maria Papadaki, and Nedaa Baker Al Barghuthi, "Design of the Blockchain Smart Contract: A Use Case for Real Estate", Scientific Research An Academic Publicher, Journal of Information Security, Vol. 9, No. 3, pp. 177-190, Jul. 2018.

[12] A. Spielman, "Blockchain: Digitally Rebuilding the Real Estate Industr", Massachusetts Institute of Technology, Cambrigde, MA. 2016.

저자소개

손 민 성 (Minsung Son)

2019년 3월 : 경기대학교 컴퓨터과학과(공학사)
2019년 3월 ~ 현재 : 경기대학교 컴퓨터과학과 석사과정(공학석사).
관심분야 : 정보보호, 블록체인.

김 희 열 (Heeyoul Kim)

2000년 : 한국과학기술원 전산학과(공학사).
2002년 : 한국과학기술원 전산학과(공학석사).
2007년 : 한국과학기술원 전산학과(공학박사).
2009년 ~ 현재 : 경기대학교 컴퓨터공학부 부교수
관심분야 : 블록체인, 보안.