공학전문석사 학위 연구보고서

블록체인을 기반으로 한 Smart Contract 활용 연구

A Study on the Application of Smart Contract based on Block Chain

2019 년 2 월

서울대학교 공학전문대학원

응용공학과 응용공학전공

윤 근 수

블록체인을 기반으로 한
Smart Contract 활용 연구

A Study on the Application of
Smart Contract based on Block Chain

지도 교수 문 봉 기

이 프로젝트 리포트를 공학전문석사 학위
연구보고서로 제출함
2019 년 2 월

서울대학교 공학전문대학원
응용공학과 응용공학전공
윤 근 수

윤근수의 공학전문석사 학위 연구보고서를 인준함
2019 년 2 월

위 원 장 곽 우 영 (인)

위 원 문 봉 기 (인)

위 원 구 윤 모 (인)

국문초록

인터넷 네트워크를 통한 거래는 대부분 기업의 통제를 통해 이루어지고 있고 본질적으로 분쟁 발생 시 판매자를 향해 편향된다. 구매자의 사기, 거래 수수료 및 통화 변환 수수료의 일정 비율은 구매비용이 증가되는 것을 피할 수 없게 만드는 원인이 된다. 비트코인과 같은 기존의 암호 화폐 거래는 상인들의 문제를 해결해 주지만, 상인들이 그들의 의무를 이행하지 않는다면 그것은 구매자들에게 아무런 보호도 제공하지 않는다. 본 프로젝트에 의한 구매자 보호가 있는 온라인 거래 방법은 기업 판매에 국한되지 않는다. 누구든지 그들의 공동체를 위한 "판매자"가 될 수 있고, 수수료를 부과하고 그들의 위임 서비스를 제공할 수 있다. 자유 시장 경쟁을 도입함으로써, 위임 수수료를 낮추고 온라인에 서 거래를 하는 사람들에게 이익을 제공할 수 있기를 기대한다.

중계자가 필요 없는 개인간의 직거래 방식을 구현하기 위해 프로그래밍이 가능한 Ethereum 블록체인을 사용하였다. 프론트 앤드는 HTML5와 JAVA Script 를 사용하였으며 백 앤드로는 Web3.js 를 이용하여 Solidity 언어를 통해 Ethereum 네트워크와 연동하였다. 현재는 Ethereum 과 Solidity 가 가장 많이 사용되고 있지만 새로운 언어가 계속해서 등장하고 빠르게 트랜드가 바뀌는 것이 블록체인의 특징이다. 따라서, 보고서는 주로 의사코드(Pseudocode)를 사용하여 기술하여 다른 블록체인과 언어에서 응용가능성을 제시하고 Ethereum 네트워크를 통해 실제 공유경제 활용시스템을 구현하여 동작성을 입증하였다.

주요어 : Blockchain(블록체인), Ethereum(이더리움), Solidity (솔리디티), Sharing Economy(공유경제), Smart Contract(스마트컨트랙), Bitcoin(비트코인)

학 번 : 2017-21117

목 차

표 목차

코드 목차

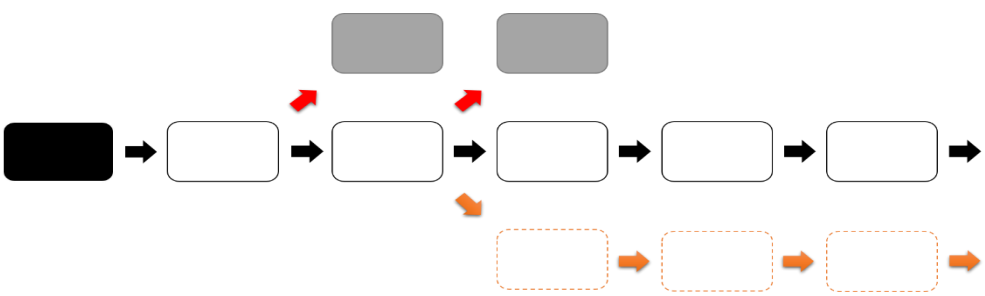
그림 목차

제 1 장 서 론

제 1 절 연구 배경

지금까지의 디지털 환경에서는 생성된 데이터를 쉽게 조작하고 삭제할 수 있었기 때문에 디지털화된 데이터에 가치를 부여하기가 쉽지 않았고 신뢰에 대한 보증을 제공할 수 없었다. 그러나, 블록체인 기술을 이용하게 되면 디지털 자료에 가치를 부여 할 수 있으며 이를 바탕으로 본 보고서의 주제인 공유 경제 영역에의 적용을 통해 그 가능성과 역할을 확대해 나갈 수 있다. 블록체인은 사토시 나카모토라는 가공의 확인되지 않은 인물 또는 그룹에 의해 2009 년 논문 [1]을 통해 비트코인으로 구현되어 등장한 이후 현재는 금융 분야를 넘어 다양한 분야에 응용하고자 하는 노력이 활발하게 이루어지고 있다. [2]

먼저 블록체인의 작동 방식을 간단히 살펴보면, <그림 1>에서처럼 모든 블록은 이전 블록을 근거로 생성되며 최초의 블록에 연계되어 이전 블록의 정보를 가진 체인을 이루게 된다. 회색 블록은 블록 생성 경쟁에서 실패하여 블록체인에서 제외되며 흰색 블록은 해시 파워 경쟁에서 승리하여 블록체인의 일부로 남게 되어 데이터는 공유 데이터 (Shared Data)가 된다. 점선으로 나타낸 블록과 같이 기존의 블록체인 에서 이탈하여 새로운 블록체인을 형성하는 것을 하드포크라고 하며 이전에 생성된 블록들은 변조나 삭제가 불가능해 그 가치의 유효성을 증명하는 역할을 한다. [3]



<그림 1> 블록체인 도식

제 2 절 연구 동향

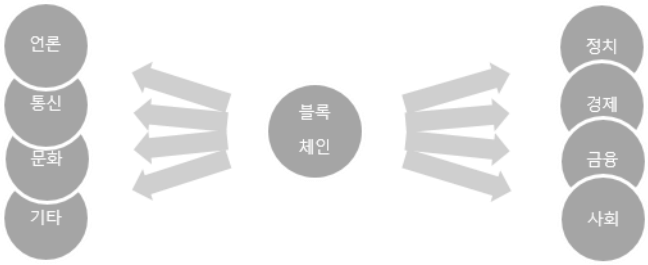
블록체인은 2009 년 암호 화폐인 비트코인의 엔진으로 사용되어 제 3 의 신뢰할 수 있는 권한이나 중앙 서버의 필요 없이 이중 지출 문제를 해결하는 최초의 디지털 화폐가 되었다. 블록체인은 증가하는 데이터 자료의 목록이며, 암호화 방식을 사용하여 연결된다. 각 블록에는 이전 블록의 암호화 해시 값과 시간 및 전송 데이터 기록이 포함되어 있다.

최초 생성된 제네시스 블록의 정보를 SHA256 알고리즘으로 암호화한 해시 값을 포함하여 거래 기록들과 함께 다음 블록이 생성되고 이것이 반복되어 후속 블록이 생성되기 때문에 원천적으로 변조와 위조가 불가능하여 중개자 개입 없이 신뢰할 수 있는 거래가 가능하다.

	장점	단점
익명성	개인정보 비제공, 높은 익명성	불법 거래 결제, 비자금 조성, 탈세
투명성	거래 내용 공개, 거래 양성화	완벽한 익명성 불가, 트래킹 가능
확장성	공개된 소스, 구축비용 절감	실물 경제와의 괴리
P2P	제 3 자의 비개입, 수수료 절감	문제 발생시 책임소재 불분명

<표 1> 블록체인의 장단점

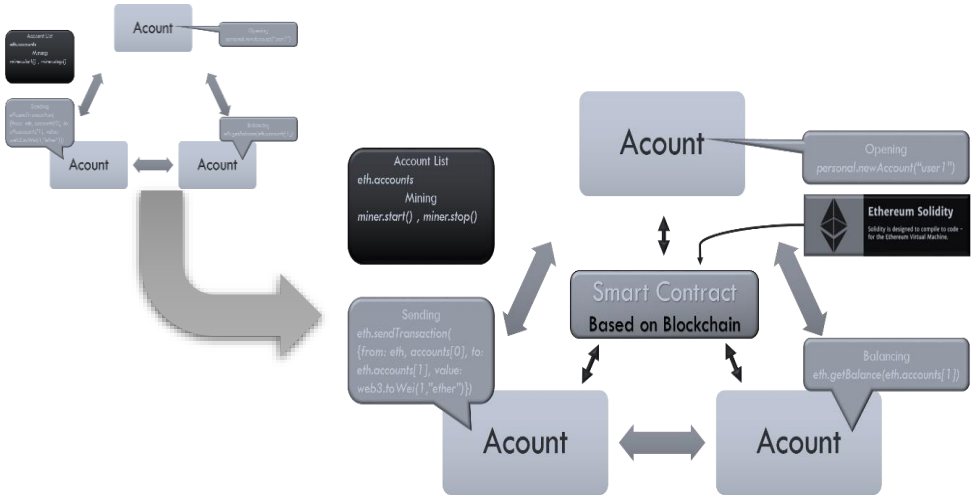
현재 블록체인은 정치 [4], 경제, 금융, 사회, 언론, 문화, 통신 등 모든 분야에서 다양한 계층의 인재들이 블록체인 기술을 적용하려는 노력을 경주하고 있다. [5]



<그림 2> 블록체인 응용분야

제 3 절 연구 범위

지금까지는 블록체인 기술을 가상 화폐 거래소를 통해 주로 개인들이 판매와 구입을 통해 그 가격만 키워가고 있고 그 속도가 각계각층의 우려를 낳을 만큼 급격히 증가하고 있다. 따라서, Ethereum 블록체인 상에 Solidity 언어로 작성된 Smart Contract 을 통해 탈 중앙화된 블록체인을 활용하여 공유 경제를 비롯한 다양한 분야에 응용 가능한 인프라 구축을 본 프로젝트의 목표로 한다.



<그림 3> Smart Contract 적용 예시

이용자의 블록체인 접근 용이성을 위해 사용이 익숙한 웹 브라우저 만으로 복잡한 추가 명령어 습득 없이 HTML(Hyper Text Markup Lan-

guage), CSS(Cascading Style Sheets)와 같은 정적인 요소와 Javas-cript 를 이용한 동적인 처리방식을 사용하여 접근성과 용이성을 확보한다. 웹 환경에서 직접 Ethereum 네트워크에 접근하기 위해 공개된 Web3.js 를 사용하며 연결은 API 의 일종인 INFURA 를 적용한다. 원활한 개발 환경 조성을 위해 Ethereum Ropsten 네트워크 상에서 전개했고 이것은 메인 네트워크와 동일한 테스트 네트워크 환경이기 때문에 Smart Contract 주소와 일부 변수 수정만으로 메인 네트워크에서도 활용 가능하다.

제 2 장 기반 기술

제 1 절 Blockchain

블록체인은 크게 퍼블릭 블록체인과 프라이빗 블록체인으로 나뉜다. 유일한 차이는 네트워크에 참여할 수 있는 사람, 합의 프로토콜 실행 및 공유된 원장 유지와 관련이 있다. [6]퍼블릭 블록체인 네트워크는 완전히 개방되었으며 누구든지 네트워크에 참여하고 참여할 수 있다. 네트워크는 일반적으로 더 많은 참여자들이 네트워크에 참여하도록 장려하는 인센티브 메커니즘을 가지고 있다. 불특정 다수의 참여자들이 처리하는 트랜잭션 정보 검증은 익명성과 보안성이 우수하지만 블록 생성 시간과 비용이 크다는 한계를 가지고 있다. [7]

이러한 이유로 프라이빗 블록체인이 등장하게 되었다. 이것은 중앙관리 기관이나 단체가 블록체인을 관리하는 블록체인의 유형이라고 할 수 있다. 별도의 네트워크 망 또는 접근 승인으로 참여자를 제한하게 되어 퍼블릭 블록체인에 비해 빠른 속도와 저렴한 비용으로 운영이 가능하다. 실질적으로는 블록체인 기술을 사용하지만 중앙 관리자에 의해 트랜잭션 정보가 생성되고 관리되어 안정적인 네트워크 유지가 가능하다. 중앙관리자의 관리로 인해 블록체인의 특성인 신뢰성과 익명성에 대한 보증은 이루어지기 힘들지만 기술적인 안정성과 신속한 유지보수가 가능하다. 이것은 블록체인의 중요한 기본 철학인 탈 중앙화 정신에 어긋나기 때문에 블록체인으로 볼 수 없다는 의견도 많다. [8]

그래서 등장하게 된 것이 프라이빗 블록체인이 변형된 형태인 컨소시엄 블록체인이다. 이것은 특정 개인이나 단체만 참여 가능한 프라이빗 블록체인과 비슷하지만 여러 집단의 협의체로 노드를 구성하기 때문에 신뢰성과 일정 수준의 익명성 보장을 제공한다. [9]

현재 가장 많은 이용자를 확보하고 있는 비트코인은 오늘날 생산되는 가장 큰 퍼블릭 블록체인 네트워크 중 하나이며 본 프로젝트는 프로그래밍이 가능한 Ethereum 네트워크를 이용한다.

	PUBLIC BLOCKCHAIN	PRIVATE BLOCKCHAIN
접근 권한	참여자 모두 가능	사전 접근 승인 필요
운영 관리	참여자 모두 가능	제한된 참여
블록 생성	참여자 모두 가능	사전 승인 기관
처리 속도	느림	빠름
예	Bitcoin, Ethereum, Tron, EOS 등	Loopchain, Fabric, Ripple, 하이퍼렛저 등

<표 2> Public 블록체인과 Private 블록체인 비교

제 2 절 Bitcoin

비트코인은 전자 현금의 한 형태인 암호통화이다. 중앙 은행 또는 단일 관리자가 없는 분산형 디지털 통화로, 중개자의 필요 없이 개인간에 비트코인 네트워크를 통해 사용자 간 전송이 가능하다. [10] 거래 기록은 암호화를 통해 네트워크 노드에 의해 확인되고 블록체인이라고 하는 공용 분산 시스템에 기록된다. 비트코인은 사토시 나카모토라는 가명을 사용하는 개인 또는 사람들에 의해 발명되었고 2009 년에 오픈소스 소프트웨어로 출시되었다. 비트코인은 채굴이라고 알려진 과정에 대한 보상으로 만들어진다. 다른 통화, 제품 및 서비스와 교환할 수 있다. 채굴은 컴퓨터 처리 능력을 이용한 기록 보관 서비스이다.

채굴은 새로 생성되는 트랜잭션을 블록으로 반복적으로 그룹화하여 블록체인을 일관성 있고 완전하며 변경할 수 없도록 유지하며 네트워크 상에 전개되어 모든 참여자가 거래 내역을 확인 할 수 있다. [11] 작업증명이라는 채굴 과정을 통해 각각의 블록에는 이전에 전개된 블록의 SHA-256 알고리즘으로 암호화된 해시 값이 포함되어 거래 내역의 해킹을 위해선 해당 블록 이후의 모든 블록과 전체 네트워크 노드를 변경해야 하기 때문에 위조와 변조를 원천 봉쇄한다.

현재 거래되고 있는 모든 비트코인은 채굴을 통해 생성되었다. 비트코인 네트워크는 블록 생성에 대한 보상을 21,000 블록(약 4 년)마다 반으로 줄이도록 설계되어 있다. 결국 보상으로 생성되는 비트코인은 사라지게되어 2 천 1 백만개가 채굴된 뒤에는 거래 수수료만으로 보상이 이루어지게 된다. 이러한 희소성을 통해 비트코인은 가치를 키워나가고 있으며 다수의 참여자들의 자발적인 채굴 참여와 대중의 관심을 이끌어 내어 기축통화로의 위치를 공고히 하고있다.

제 3 절 Ethereum

Ethereum 은 스마트 계약(스크립팅) 기능을 갖춘 공개 소스, 공개 블록체인 기반의 분산 컴퓨팅 플랫폼 및 운영 체제이다. 2013 년 말 암호연구자이자 프로그래머인 비탈릭 부테린에 의해 제안되었다. 공식적인 개발은 2014 년 초 스위스 회사인 Ethereum Swiss GmbH 를 통해 시작되었고 2014 년 7 월부터 8 월까지 온라인 클라우드 세일에 의해 판매가 시작되었으며 첫 시스템 가동 일은 2015 년 7 월 30 일로 7 천 2 백만 개의 동전을 발행하였다.

Ethereum 네트워크의 공식 출시 전 코드명 Olympic 은 마지막 시제품이다. 출시 이후에도 Ethereum 은 수차례 계획된 프로토콜 업그레이드를 수행하였고 이는 플랫폼의 기본 기능 및 인센티브 구조에 영향을 미치는 중요한 변화였으며 소프트 포크를 사용하여 수행된다. Homestead 는 처음으로 안정적이라고 알려졌고 거래 처리, 가스 가격, 보안

개선사항을 포함하였다. 2017 년 10 월 16 일 발효된 Metropolis 는 EVM 의 복잡성을 줄이고 Smart Contract 개발자들에게 더 많은 유연성을 제공하기 위한 업그레이드가 진행되었다. Ethereum 2.0 으로 불리고 있는 세레니티(Serenity)는 새로운 합의 알고리즘인 캐스퍼(Casper)와 데이터를 분할 저장하는 샤딩(Sharding)을 대표적인 구성요소로 제시한다. Ethereum 은 작업증명(PoW: Proof of Work)합의 알고리즘은 전기를 대량으로 소비하고 대형 채굴업자들에게 권력이 집중된다는 문제점이 제시되어 왔다. [12] 따라서, Ethereum 은 PoW 에서 지분증명(PoS:Proof of Stake)방식으로 전환을 모색하고 있으며 개발중인 캐스퍼는 Ethereum 의 독자적인 PoS 합의 알고리즘이다. 또 다른 문제점으로는 느린 트랜잭션 속도와 확장성 부족이고 이것의 해결 방법으로 연구중인 프로젝트가 샤딩이다. 샤딩은 데이터를 분할해 각 노드들에 할당한 뒤 처리하게는 분산형 저장기술이다. [13]

2015. 5	Olympic	0 단계
2015. 7	Frontier	1 단계
2016. 3	Homestead	2 단계
2017. 10	Metropolis	3 단계
예정	Serenity	4 단계 (Ver 2)

<표 3> 이더리움 개발 단계

Ethereum 은 비트코인과 몇 가지 다른 커다란 특징을 가진다. 첫째, 비트코인은 블록 생성 시간이 약 10 분인데 반해 10 초에서 15 초 가량으로 짧고 둘째, 비트코인 거래는 트랜잭션 크기(바이트)를 기준으로 정해지는 반면 Ethereum 의 트랜잭션 비용은 계산상의 복잡성, 대역폭 사용 및 저장공간에 따라 달라진다. 가장 큰 특징은 비트코인은 프로그래밍이 불가능하지만 Ethereum 은 Solidity 를 필두로 하여 개발자들이 네

트워크상에 개발한 프로그램을 업로드하여 운영이 가능하기 때문에 블록체인 2.0 으로 불리고 있다.

제 4 절 Altcoin

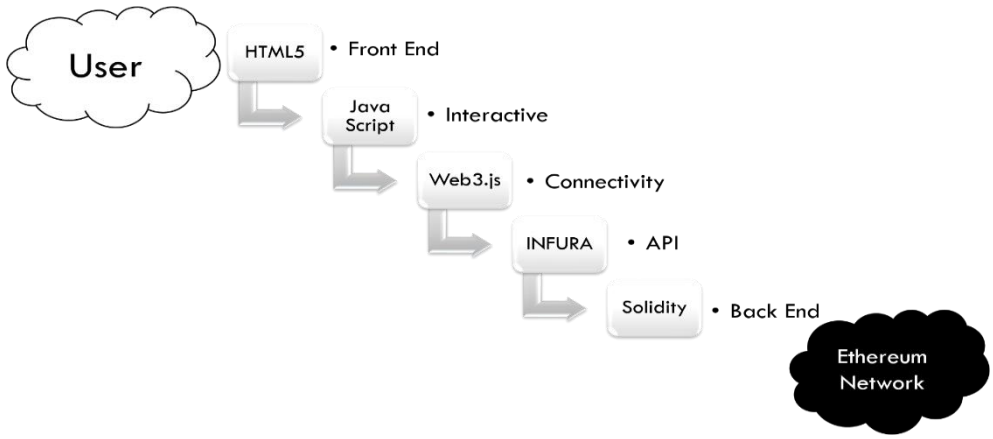
알트 코인은 대안을 뜻하는 Alternative 와 Coin 의 합성어이며 비트코인 이후 등장한 모든 코인을 지칭한다. 프로그래밍이 가능한 2 세대 블록체인이라 불리는 알트코인의 하나인 Ethereum 의 괄목할만한 성공 이후 다양한 기능을 추가한 에이다, 트론, 이오스, 대시, 모네로, 제트캐시, 퀀텀 등 3 세대라 불리우는 블록체인 코인이 등장하여 현재는 수많은 알트 코인을 볼 수 있다. 많은 알트코인이 비트코인 소스코드를 기반으로 만들어지기 때문에 채굴 방식의 블록 생성 프로세스와 그에 따른 보상 방식을 제공하며 불특정 다수의 참여를 유도하여 네트워크를 유지한다.

이 밖에도 비트코인 캐시, 비트코인 골드, 비트코인 실버, 비트코인 다이아몬드, 비트코인 SV 등 사토시의 오리지널 비트코인에서 하드포크로 분리된 형태의 알트코인도 활발한 거래가 이루어지고 있는 것을 볼 수 있다. 대표적인 하드포크 알트코인인 비트코인 캐시에 대해 자세히 살펴보면, 2017 년 8 월 1 일 10 시 16 분에 생성된 478558 블록을 기점으로 채굴장과 거래소를 운영하는 ViaBTC 에 의해 비트코인과 비트코인 캐시 둘로 분리 되었다. 기존의 1MB 블록 크기에서 2~8MB 까지 유동적으로 크기를 확장할 수 있기 때문에 블록에 기록할 수 있는 거래 기록을 늘릴 수 있어 기존의 비트코인보다 낮은 수수료를 지불하게 되었지만 이것은 채굴자들에게 더 큰 권력을 부여하게 되는 단점도 제공하게 된다. 다른 하드포크된 알트코인들도 마찬가지로 비트코인 캐시의 경우 같이 일부 기능의 확장이나 추가 등의 개선 작업으로 새로운 블록체인을 구성하였다. 여기서 주지할 만한 것은 이러한 하드포크로인해 비트코인의 시장 지배력과 가치의 저하보다는 오히려 강화되는 양상을 발견할 수 있다는 점이다. [14]

	BITCOIN	ETHEREUM	ALT COIN
분류	1 세대	2 세대	3 세대
개발자	사토시 나카모토	비탈릭 부테린	다양함
최초발행	2009 년	2015 년	다양함
조직형태	비영리기구	비영리재단	다양함
채굴방식	작업증명 [15]	지분증명(예정)	다양함
블록크기	1 MB	Dynamic	다양함
생성주기	약 10 분	약 15 초	다양함
전송시간	약 7 tx/sec	약 25 tx/sec	다양함
SMART CONTRACT	미지원	지원	다양함

<표 4> 블록체인 세대별 비교

제 3 장 선행 기술



<그림 4> 기술 적용 도식

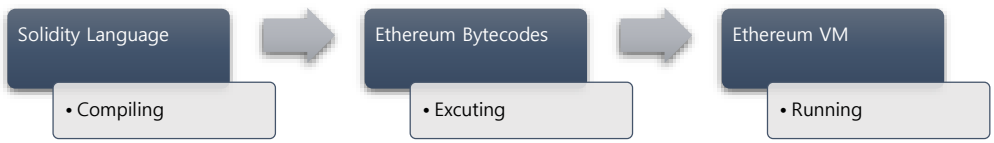
제 1 절 Solidity

Solidity 는 2014 년 8 월 가빈 우드에 의해 처음 제안되었고 후에 크리스티안 리트위스너가 이끄는 Ethereum 프로젝트 팀에 의해 개발되었다. 이 언어는 Ethereum 가상 머신(EVM)을 대상으로 설계된 여러가지 언어들 LLL, Viper, Mutan 중 하나이다.

현재 Solidity 는 Ethereum 네트워크 상에서 가장 많이 사용되는 언어이며, Monax 및 Tend 를 사용하는 하이퍼더 버로우 블록체인 등 Ethereum 과 경쟁하는 플랫폼에서 실행되는 다른 프라이빗 블록체인에서도 사용된다. [16]

Solidity 는 EVM(Ethereum Virtual Machine)에서 동작하는 Smart Contract 을 개발하기 위해 사용되는 프로그래밍 언어로 Solidity 언어로 작성된 코드는 EVM 에서 실행 가능한 바이트코드로 컴파일 된다. 개발자는 이를 통해 애플리케이션을 작성할 수 있으며 트랜젝션에 대한 신뢰할 수 있는 기록을 Ethereum 네트워크 상에 남길 수 있다. Serpent 와 Mutan 과 같은 다른 언어와 비교하여 Solidity 는 기존 웹 개발자에게 익

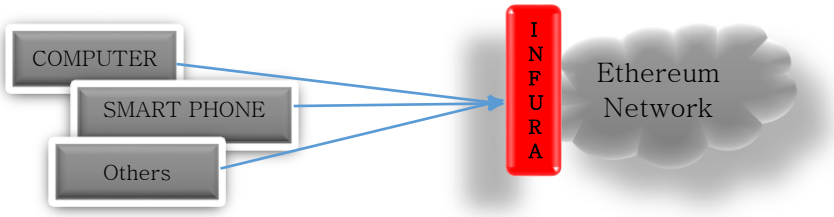
숙해지도록 설계되었으며 계층적 매핑과 구조를 포함한 복잡한 구성원 변수가 지원되었고 단일 Smart Contract 상에서 복수의 함수 기능을 사용 가능하도록 지원하는 ABI(Application Binary Interface) 기능도 도입되었다.



<그림 5> Solidity 개념도

제 2 절 INFURA

Ethereum 블록체인 네트워크와 상호 작용하는 응용 프로그램을 작성 시 개발자가 직면하게되는 큰 어려움은 코드가 Ethereum 블록 체인에 참여하는 전체 노드와 연결 할 필요가 있다는 것이다. 애플리케이션을 작성하고자하는 개발자에게 Ethereum 네트워크와의 상시 연결을 위해 로컬에 시스템을 설치하고 최신 상태로 유지하는 것은 다소 어렵다.



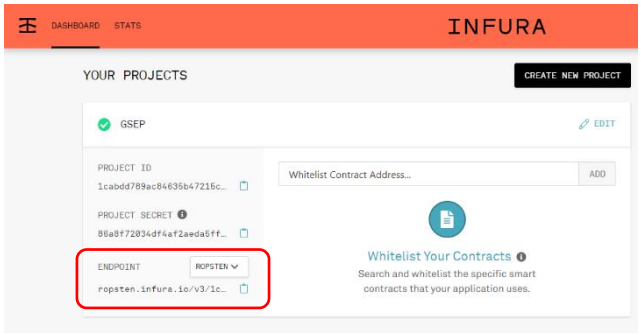
<그림 6> INFURA 적용 개념도

INFURA 의 역할은 안정적이며, 확장 가능한 Ethereum 노드를 무료로 제공하는 것이다. 그렇게 함으로써, 개발자가 자신의 인프라를 유지하는 부담을 없애고 애플리케이션 작성하는 데 최선을 다할 수 있다.

Infura.io 사이트를 통해 무료로 발급받은 토큰을 사용하여 아래의 코드로 사용하여 브라우저에서 직접 Ethereum 네트워크에 접근하여 계좌 생성, 거래 실행, 데이터 읽고 쓰기 등의 작업을 Web3.js 라이브러리를 사용해서 명령어를 직접 실행할 수 있다.

```
// Setup web3
var web3 = new Web3(new Web3.providers.HttpProvider("https://ropsten.infura.io/TOKEN"))
```

<코드 1> Web3.js 로 Ropsten Test Network 와 연결

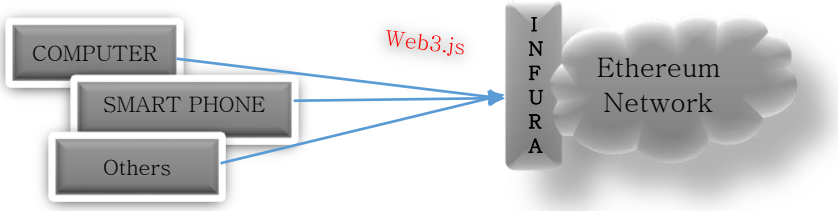


(https://infura.io/dashboard)

<그림 7> INFURA 사용 정보

제 3 절 Web3.js

Web3.js 를 사용하면 Ethereum 블록체인 네트워크와 상호 작용하는 애플리케이션을 개발할 수 있다. 이더를 한 계정에서 다른 계정으로 보내거나 데이터를 읽고 쓰기, Smart Contract 만들기 등과 같은 작업을 웹 상에서 수행할 수 있는 일종의 라이브러리 모음이다.



<그림 8> Web3.js 적용 개념도

Ethereum 은 모든 데이터와 코드의 복사본을 블록체인에 저장하는 피어투피어 네트워크이기 때문에 Web3.js 를 사용하여 Ethereum 노드에 데이터를 읽고 쓸 수 있다. 마치 JSON API 와 함께 jQuery 를 사용해서 웹서버에 데이터를 읽고 쓰는 것과 같다. [17]

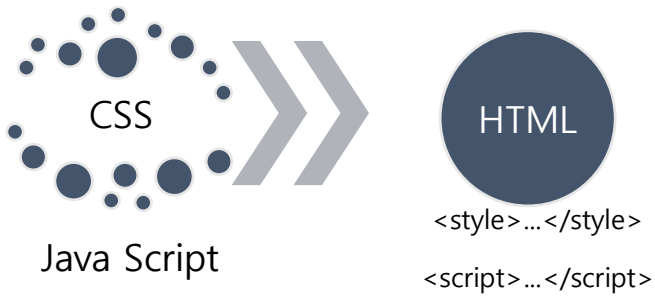
```
// New Account
var new = web3.eth.accounts.create();
// Get Balance
var balance = web3.eth.balance(ADDRESS);
// Make Transaction
web3.eth.sendTransaction({from:ADDRESS,to:ADDRESS, value:web3.toWei(1," ether" )});
```

<코드 2> Web3.js Library 사용 예시

제 4 절 Java Script

HTML 및 CSS 와 함께 Java Script 는 World Wide Web 의 세 가지 핵심 기술 중 하나이다. Java Script 는 인터랙티브 웹페이지를 활성화하므로 웹 어플리케이션의 필수적인 부분이다. 대부분의 웹 사이트에서는 이 기능을 사용하고 있으며, 모든 주요 웹 브라우저에는 이 기능을 실행할 수 있는 전용 Java Script 엔진이 있다. [18]

언어 이름, 구문 및 각 표준 라이브러리를 포함하여 Java Script 와 Java 언어 간의 외견상 유사성이 크지만, 두 가지 언어가 구별되고 설계가 크게 다르다. JavaScript 는 Self 및 Scheme 과 같은 프로그래밍 언어에 영향을 받았으며 따라서 Java 와는 다른 문법체계를 가지고 있다.



<그림 9> Java Script 와 HTML 적용 개념도

HTML 로 구현한 텍스트 상자의 속성값을 자바스크립트로 읽어오는 방식을 코드로 아래의 예시 코드를 통해 볼 수 있다. 본 프로젝트에는 자바스크립트를 통해 읽어온 값을 Web3.js 라이브러리를 통해 Ethereum 블록체인 상에 INFURA API 를 거쳐 업로드 하게 된다. [19]

```
// Java Script
<script type="text/javascript">
        var id = $('#test).val();
</script>

// HTML
<html>
        <body>
                <input type="text" id=test name=test value=50/>
        </body>
</html>
```

<코드 3> Java Script 와 HTML 연동 예시

제 5 절 HTML5

HTML5는 주요 업데이트 및 W3C 권장 사항으로 2008년 1월 22일에 공개되어 2014 년 10 월에 공개되었다. 목표는 최신 멀티미디어 및 기타 새로운 기능에 대한 지원으로 언어를 개선하는 것이다. 상호운용 가능한 구현을 장려하기 위한 세부 프로세싱 모델이 포함되어 있으며, 문서에 제공되는 마크업을 확장, 개선 및 합리화하고 복잡한 웹 응용프로그램에 대한 마크업 및 응용프로그램 프로그래밍 인터페이스(API)를 도입한다. 이와 동일한 이유로 HTML5 는 저전력 장치를 염두에 두고 설계된 기능을 포함하고 있기 때문에 교차 플랫폼 모바일 애플리케이션에도 적합하다. [20]

기존의 HTML4 버전과는 달리 많은 새로운 구문적 특징들이 포함되어 있다. 멀티미디어 및 그래픽 콘텐츠를 기본적으로 포함 처리하기 위해

새로운 <video>, <audio>, <canvas>요소들이 추가되었으며, 확장 가능한 벡터 그래픽 콘텐츠 및 수학적 기능을 지원한다. 이외에도 문서의 의미적 내용을 풍부하게 하기 위해 <header>, <section>, <article>, <main>, <footer>, <aside>, <nav>, <figure> 와 같은 새로운 페이지 요소들이 추가되었고 <a />, <cite>, <menu>와 같은 특성들은 변경 또는 재정의 되었다.

```
<head>
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

<코드 4> 모바일 브라우저 Viewport 메타 태그 사용 예시

본 프로젝트에서는 모바일이 아닌 일반 컴퓨터용 크롬 브라우저 위에서 MetaMask(https://metamask.io) 지갑 사용으로 개발 범위를 제한하였으나 모바일에서도 <코드 4>와 같은 약간의 코드 추가만으로 아이폰, 아이패드, 안드로이드폰, 안드로이드패드 등에서 응용이 가능하다.

<meta> “viewport” 요소는 페이지의 크기를 제어하는 방법에 대한 정의이며 “width=device-width”는 해당 기기의 화면 너비를 따르도록 장치에 따라 다르게 정의하고 initial-scale 은 브라우저에 로드 될 때 초기 줌 레벨을 설정하여 기기의 화면 크기 마다 다르게 보여진다.

제 6 절 IPFS

IPFS(Interplanetary File System)는 컨텐츠 주소 지정이 가능하고 피어 투 피어 방법을 만들어 분산 파일 시스템에서 하이퍼미디어를 저장하고 공유하도록 설계된 프로토콜 및 네트워크이다. IPFS 는 처음에는 후안 베네에 의해 설계되었으며, 현재는 커뮤니티의 도움을 받아 개발, 진행되고 있는 오픈 소스 프로젝트이다. IPFS 는 모든 컴퓨팅 장치를 동

일한 파일 시스템과 연결하려는 피어 투 피어 분산 파일 시스템이다. 어떤 면에서 IPFS는 월드 와이드 웹과 비슷하지만, IPFS는 하나의 Git 저장소 내에서 개체를 교환하는 하나의 토렌트 집단으로 보일 수 있다. 즉, IPFS 는 콘텐츠 주소 지정 하이퍼링크를 통해 처리량이 높은 콘텐츠 주소 지정 블록 스토리지 모델을 제공한다. [21]

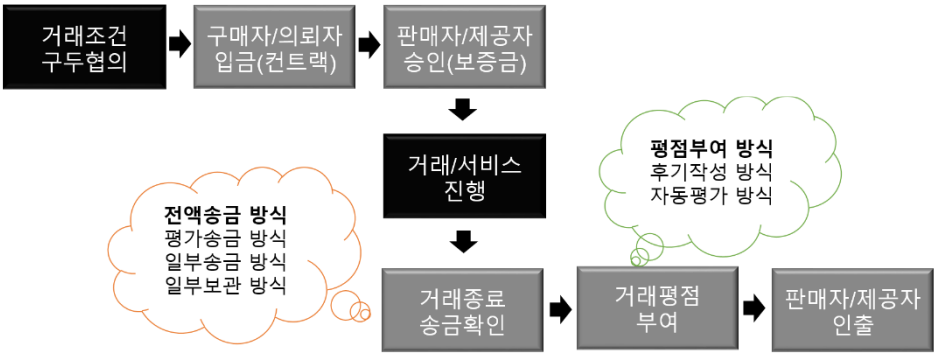
파일 시스템은 FUSE 및 HTTP 를 통해 다양한 방법으로 액세스할 수 있고 로컬 파일을 IPFS 파일 시스템에 추가하여 전 세계에 제공할 수 있다. 파일은 해시로 식별되므로 캐싱에 편리하고 그것들은 토렌트 기반의 프로토콜을 사용하여 배포된다. 콘텐츠를 보는 다른 사용자는 네트워크의 다른 사용자에게 콘텐츠를 제공하는 데 도움이 된다. IPFS 에는 IPNS 라는 네임 서비스가 있으며, PKI 를 기반으로 하는 글로벌 네임스페이스가 있다. 이와 비슷한 종류의 탈 중앙 스토리지로는 Swarm, SIA, STORJ, Filecoin, STOKIT 등이 있다. [22]

<그림 10> IPFS 개념도

본 프로젝트의 범위에 IPFS 와 같은 탈중앙 스토리지 기술을 직접 포함하지는 않았지만 평가 기록 모듈에 대한 부가 서비스로 사진이나 동영상 후기 등을 블록체인에 업로드하기 위한 가능성과 방법의 하나로 살펴보았다.

제 4 장 공유경제 시스템 설계

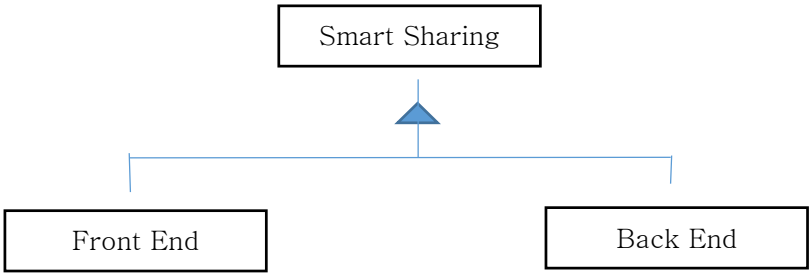
제 1 절 모형 설계



<그림 11> 설계 도식

인터넷이 연결된 브라우저를 통해 블록체인상에 쉽게 접근할 수 있으며 제목, 세부내용, 옵션 설정 등을 완료한 이후 모든 거래 내역과 승인 과정은 Solidity 언어를 통해 Ethereum 네트워크 데이터베이스 상에 실시간 업로드 되어 완료된다.

가장 중요한 점은 첫째, 유저의 지불액을 Ethereum 네트워크상에 보관하여 상호 신뢰를 제공하는 것이며 둘째, 쌍방간의 거래 종료 후 상호 동의한 계약 조건에 따른 결제금액 설정과 상대방에 대한 평가를 기록하여 추후 발생 가능한 분쟁의 소지를 제거하고 신용도의 유지를 그 목적으로 한다.



<그림 12> System Decomposion

Front End 는 시스템의 제일 앞에 위치하여 외부 입력을 받아들이는 부분이다. 이와는 반대로 Ethereum 과 Solidity 와 같이 시스템의 뒤 또는 안에서 시스템을 지원하는 부분을 Back End 로 구성한다.

제 2 절 의사 코드

4.2.1 거래 생성

물품 거래의 경우 판매자와 구매자 또는 용역 거래의 경우 제공자와 의뢰자가 거래에 대한 세부 협의를 전화, 메신저, SMS, 이메일 등을 통해 마친 후 의뢰자와 구매자는 거래 생성 모듈을 통해 Ethereum 블록체인 네트워크 상의 Smart Contract 에 거래를 생성하게 된다. 블록체인에 대한 세부 지식이 전혀 없이도 사용 가능하도록 브라우저에서 보여지는 HTML 을 이용해서 입금금액, 수신인 주소, 보증금 설정액 등을 입력하여 거래 트랜젝션을 생성하고 MetaMesk 등의 지갑을 통해 협의된 금액을 입금하여 거래 생성을 완료한다. 입력 후에는 자동으로 Java Script 를 이용해서 Web3.js 를 통해서 블록체인에 기록된다.

<그림 13> 1. 거래 생성

```
1: Function makeTransaction

2:        If amount != 0 then
3:                amount= msg.value,
4:                collateral = collateral,
```

```
5:              …… (Variables)
6:              senderCanWithdraw = true,
7:              receiverCanWithdraw = false,
8:              complete = false
9:              Push to Contract
10:     End If

11: End Function
```

<코드 5> 의사 코드 – 1. 거래 생성

4.2.2 거래 승인

구매자 또는 의뢰자가 사전 협의 후 생성한 거래 트랜젝션에 판매자 또는 제공자는 설정된 보증금을 전송하여 상호간 거래의 신뢰도를 높이게 된다.

입금된 보증금과 구매자 또는 의뢰자가 거래 생성 시 입금한 거래금액은 거래 종료 후 원만한 거래 완료 시 보증금과 거래금액은 입금한 판매자 또는 제공자가 돌려 받게 되고 문제 발생 시에는 동의 하에 각각의 입금자가 돌려받게 된다.

<그림 14> 2. 거래 승인

```
1: Function acceptTransaction

2:      If transaction = accepted then
```

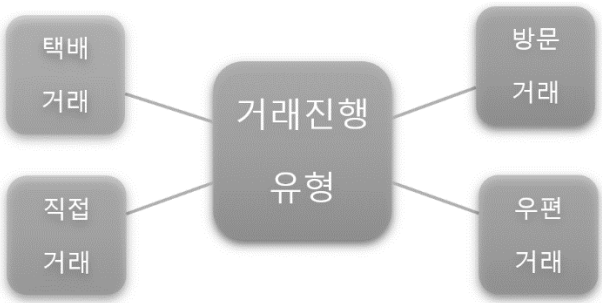
```
3:              accepted = true;
4:              …… (Variables)
5:              senderCanWithdraw = false
6:              Push to Contract
7:      End If

8: End Function
```

<코드 6> 의사 코드 – 2. 거래 승인

4.2.3 거래 진행

구매자와 의뢰자의 거래 생성과 입금과 판매자와 제공자의 보증금 입금을 마친 뒤에는 물품 거래의 경우 직접, 우편, 택배 등의 방식으로 이루어지고 용역 거래의 경우는 상호 협의 된 장소로 방문하여 거래가 이루어지게 된다.

<그림 15> 3. 거래 진행



<그림 16> 거래 유형

4.2.4 거래 종료

정상적인 거래를 마친 뒤에는 구매자 또는 의뢰자가 거래 종료 승인 단계를 거쳐 Smart Contract 에 보관 중인 이더가 판매자 또는 제공자의 계좌로 전송이 이루어진다. 일단 승인이 이루어진 후에는 철회가 불가능하기 때문에 신중한 결정이 필요하며 원만한 거래가 이루어지지 못한 경우에는 환불 절차로 진입하게 된다.(4.2.7 환불 참조) 거래 종료 시 현 프로젝트에서는 기존에 협의한 금액의 100% 전액을 수신인이 수령하도록 되어 있지만 거래에 대해 완전히 만족한 경우에는 모든 금액을 수령할 수 있고 경우에 따라서 만족하지 못하는 경우 90%, 80% 등 일부만 전송하도록 루틴을 수정하게 되면 또 다른 형태의 거래 평가가 이루어질 수 있다.

<그림 17> 4. 거래 종료

```
1: Function finalizeTransaction

2:      receiverCanWithdraw = true
3:      senderCanWithdraw = false
4:      Push to Contract

5: End Function
```

<코드 7> 의사 코드 – 4. 거래 종료

4.2.5 평점 부여

거래 종료 시 평점 부여를 통해 거래 상대방에 대한 평가를 블록체인 상에 기록하여 일종의 신용 기록을 남기게 된다. 평점은 1 에서 5 단계로 부여하며 거래 생성 전에 신용 조회를 통해 상대방의 신용을 사전에 확인하여 5 점 만점을 기준으로 평점을 조회 할 수 있고 평가한 인원도 확인 할 수 있어 거래에 대한 신뢰도 향상을 그 목적으로 한다. 평점의 경우 1 에서 5 단위를 각각 매우 불만족, 불만족, 보통, 만족, 매우 만족의 5 개로 구분하였으나 좀 더 세밀한 평가를 위해 10 개의 단위로 조정할 수 있으며 평점 방식 이외에도 Ethereum 네트워크상에 직접 대용량의 데이터를 올리기는 어렵기 때문에 IPFS (Interplanetary File System)를 이용하여 사진 후기나 동영상 후기 등을 올려 다양한 평가 방식의 도입도 가능하다.

상대방의 Ethereum 주소로 조회가 이루어지기 때문에 개인정보 노출이나 범죄의 위험으로부터 회피가 가능하다. 상대방에 대한 평가를 강제할 수는 없기 때문에 원하지 않는 경우 평가 부분은 제외한다.

<그림 18> 5. 평점 부여

```
1: Function ratingStore

2:      If rating <= 5 && rating >= 0 then
3:              RatingStore store = rating
4:              AddressStore store = address
```

5: End If

6: End Function

<코드 8> 의사 코드 – 5. 평점 부여

4.2.6 거래 인출

마지막 단계인 거래 인출은 판매자 또는 제공자가 Smart Contract 주소에서 자신의 Ethereum 주소로 상대방으로부터 입금된 금액을 전송 받는 것을 말한다. 이렇게 전송 받은 금액은 일반 거래소를 통해 현금화하거나 개인 지갑에 보관할 수 있다. 동시에 거래 승인 시 Smart Contract 으로 입금한 보증금도 함께 돌려받게 된다.

<그림 19> 6. 거래 인출

1: Function receiverWithdrawal

2: transfer amount to receiver

3: transfer collateral to receiver

4: End Function

<코드 9> 의사 코드 – 6. 거래 인출

4.2.7 환불

원만한 거래가 이루어지지 못한 경우 환불 절차에 돌입한다. 구매자 또는 의뢰인은 거래 생성 시에 Smart Contract 으로 입금한 금액을 돌려받게 되고 판매자 또는 제공자는 거래 승인 시 입금한 보증금을 각각 자신들의 Ethereum 주소로 돌려 받게 된다.

<그림 20> 7. 환불

```
1: Function refundTransaction

2:        receiverCanWithdraw = false
3:        senderCanWithdraw = true

4: End Function
```

<코드 10> 의사 코드 – 7. 환불

4.2.8 평점 조회

최초 단계인 거래 생성 전에 대상자의 거래 내역에 대한 신용도를 조회 할 수 있다. 이미 이루어진 거래에 대한 평가를 통해 먼 거리에 있는 대상과도 심리적으로 안정적인 거래를 진행할 수 있으며 평점 부여에 대한 최소한의 부담을 통해 원만한 거래를 유도하게 된다.

<그림 21> 8. 평점 조회

1: Function rate

2: search address

3: call back

4: End Function

<코드 11> 의사 코드 – 8. 평점 조회

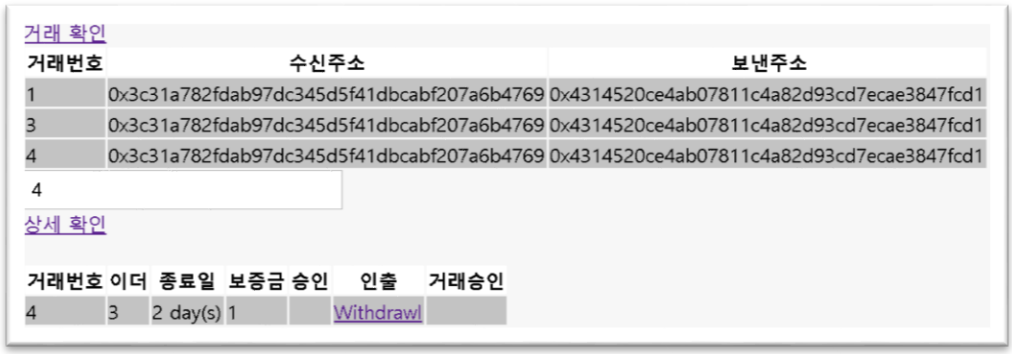
제 5 장 시스템 운영 및 검증

제 1 절 적용 시나리오

5.1.1 거래 생성

<그림 22> 적용 시나리오 – 거래 생성 1

의뢰자 또는 구매자는 거래 생성 페이지를 통해 거래가 협의된 거래 대상의 Ethereum 주소, 이더, 종료일, 보증금을 기입하고 거래를 생성한다. 생성된 거래는 고유 번호가 부여되며 상호간의 실시간 확인과 검증이 가능하다.



<그림 23> 적용 시나리오 – 거래 생성 2

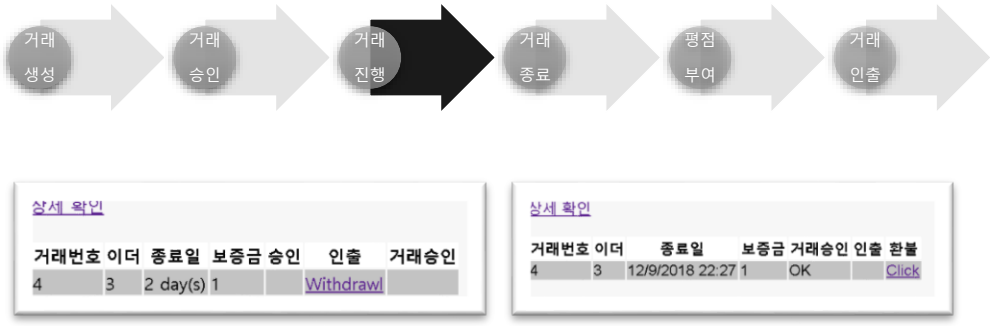
상기 거래 화면을 통해 의뢰자 또는 구매자는 현재 상태를 확인 할 수 있고 거래 상대방이 거래 승인을 하기 전에는 거래가 이루어지기 전이기 때문에 인출이 가능하다.

5.1.2 거래 승인

<그림 24> 적용 시나리오 – 거래 승인

판매자 또는 제공자는 본인의 Ethereum 주소로 생성된 거래를 확인하고 보증금을 Smart Contract 에 전송하여 거래를 승인한다. 거래 승인을 마치고 상세 확인을 하면 거래 승인은 OK 로 상태가 전환되고 아직 거래가 이루어지 지 않은 상태이므로 환불은 가능한 상태로 변경된다.

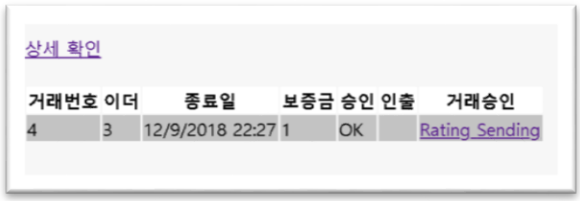
5.1.3 거래 진행



<그림 25> 적용 시나리오 – 거래 진행

Smart Contract 에 상호간의 입금을 확인하게 되면 직접, 우편, 택배, 방문 등의 사전 협의된 방식으로 거래가 이루어진다.

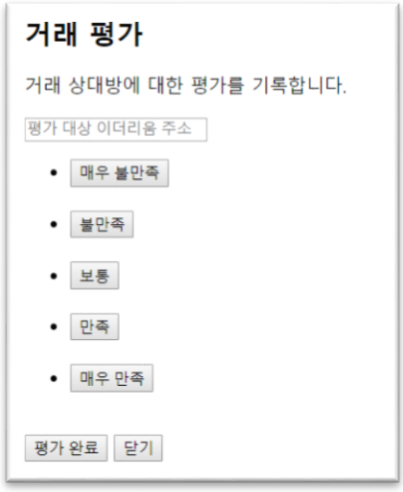
5.1.4 거래 종료



<그림 26> 적용 시나리오 – 거래 종료

거래가 종료된 뒤 “Sending” 을 누르면 Smart Contract 에 보관 되어 있는 이더를 상대방이 자신의 Ethereum 주소로 전송 할 수 있게 된다. 평점 부여는 선택사항으로 평가 없이 종료도 가능하나 신뢰 형성을 위해 평가를 권장한다.

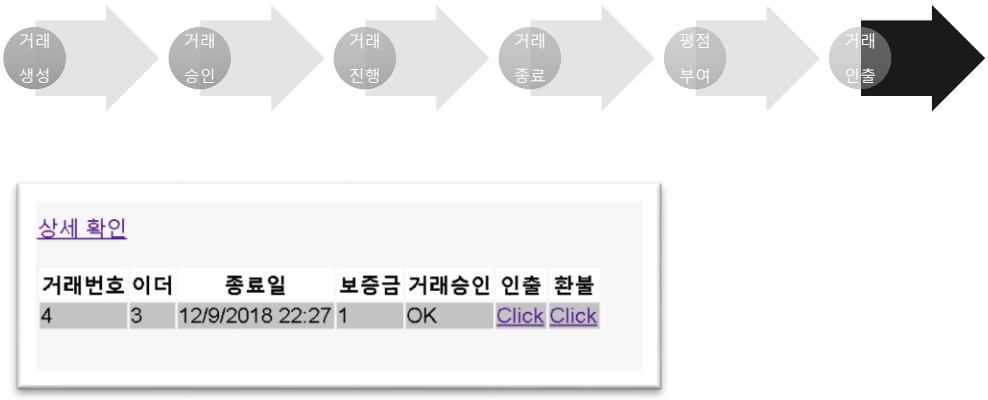
5.1.5 평점 부여



<그림 27> 적용 시나리오 – 평점 부여

원만한 거래를 마치게 되면 의뢰자 또는 구매자는 “Rating”을 이용하여 거래한 상대방에 대한 평가를 하게 된다. 평점 부여를 원하지 않으면 평가 없이 “Sending” 을 이용하여 Smart Contract 에 보관되어 있는 이더를 상대방이 자신의 Ethereum 주소로 전송할 수 있도록 승인된다.

5.1.6 거래 인출



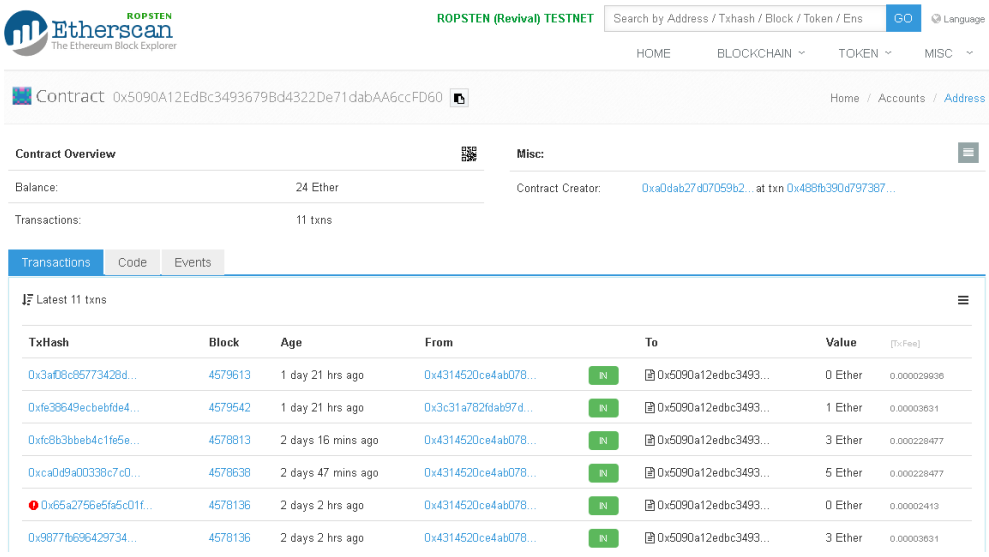
<그림 28> 적용 시나리오 – 거래 인출

판매자는 인출 메뉴를 통해 이더를 자신의 Ethereum 주소로 받아 거래를 완전히 종료한다. 원만한 거래가 이루어지지 않았을 경우 환불 메뉴를 통해 환불을 할 수 있다.

5.1.7 평점 조회

<그림 29> 적용 시나리오 – 평점 조회

최초 거래 생성 전 상대방의 신용을 점검하여 거래에 대한 신뢰도를 높이게 되며 거래를 마친 후에는 직접 평가가 가능하고 누구나 조회가 가능하다는 점을 상대방도 인지하고 있기 때문에 원활한 거래를 유도할 수 있다.



<그림 30> 거래 내역 확인 (Etherscan)

제 2 절 적용 연구

5.2.1 시나리오 A – 차량 공유 서비스

공유 경제는 세상을 바꾸기 시작했지만 일부 장벽들은 자동차 공유 산업과 같은 분야에서 그것을 확장시키는 것을 막고 있다. 세계는 자동차 소유에 대한 불편함이 증가하고 자동차 공유에 대한 더 높은 수요를 경험하고 있지만, 상호 신뢰에 대한 문제는 이것을 확장하는데 있어서 또 하나의 장벽이 된다.

블록체인 기술을 적용하면 이러한 문제를 극복할 수 있다. 최종 거래 결과를 통해 고객이 서로 신뢰하고 더 안전한 경험을 할 수 있게 될 것이다. 이러한 자동차 공유의 증가는 도로에 있는 자동차 수를 현저히 줄이는데 도움이 될 수 있고 이것은 더 많은 공간을 확보하고 이를 통해 지구 환경 보호에도 이바지 할 수 있다.

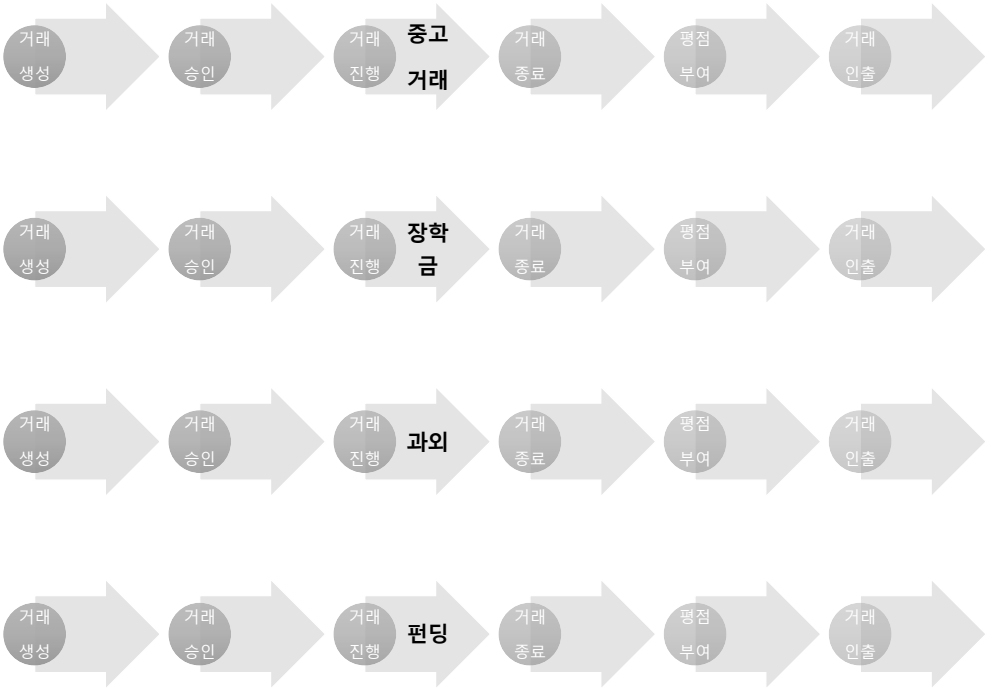
<그림 31> 적용 연구 – 시나리오 A

5.2.2 시나리오 B – 방 공유 서비스

방 공유 서비스는 웹사이트 및 모바일 앱을 통해 온라인 마켓 플레이스를 운영한다. 회원은 이러한 서비스를 제공하는 업체를 통해 숙박, 주로 홈스테이 또는 관광을 주선하거나 제공 받게 된다. 업체는 건물이나 방을 직접 소유하지 않으며 이벤트를 주최하지 않고 예약 과정에서 커미션을 얻고 있다. 블록 체인은 이러한 중앙 집중식 교환 시스템 대신 자동화된 네트워크를 통해 사용자들이 직접 거래를 운영하게 되어 거래 비용과 불필요 한 수수료를 낮추고 이익을 최대화 할 수 있게 된다.

<그림 32> 적용 연구 – 시나리오 B

5.2.3 시나리오 C – 기타 거래 유형



<그림 33> 적용 연구 – 시나리오 C

5.2.4 예시 코드

본 시나리오에 적용된 예시 코드는 부록 1 Solidity 예시 코드, 부록 2 Java Script 예시 코드, 부록 3 HTML 예시 코드로 발췌하여 첨부하였다.

제 6 장 결 론

방 공유, 차량 공유, 동영상 공유 등 오늘날의 거대 인터넷 기업들의 운영 방식을 보면, 사용자들이 직간접적으로 기여하는 것에 의존하는 하나의 공통적 가치를 가지고 있다는 것을 알 수 있다. 지난 수십년 동안 이어져온 지배적인 위치를 가진 대형 사업자들이 수동적인 소비자를 대상으로 서비스를 제공하는 책임을 지고 있는 전통적인 중앙집권적 형태에서 점차 벗어나고 있으며 최근에는 점점 더 분산되어 탈 중앙화를 지향하는 새로운 모델로 나아가고 있다. 대규모 운영자들은 훨씬 더 활동적인 소비자 그룹에 서비스를 제공하기 위해 여러 사람의 자원을 통합해야 한다. 이러한 변화는 물리적 사무실, 자산 또는 심지어 직원을 필요로 하지 않는 새로운 세대에 출현한 것이다. 그러나 이 모델의 커다란 문제점은 대부분의 경우, 참여자들이 창출한 가치가 가치 생산에 기여한 모든 사람들 사이에서 균등하게 재분배되지 않는다는 것이다. 대부분의 이익은 플랫폼을 운영하는 중개자들에 의해 침유된다.

이러한 패러다임의 전환을 가져오고 있는 새로운 기술인 블록체인은 중개자의 필요 없이 안전하고 분산적인 방식으로 가치의 교환을 촉진한다. 블록체인 기술의 가장 혁신적인 측면은 소프트웨어를 안전하고 분산적으로 운영할 수 있다는 것이다. 블록체인을 사용하면 응용소프트웨어를 더 이상 중앙 집중식 서버에 구축할 필요가 없이 어떤 개인도 자력으로 운영할 수 있는 피어 투 피어 네트워크에서 실행할 수 있다. 제삼자, 즉 중개인의 도움 없이 Blockchain 기술은 궁극적으로 개인이 공통의 활동을 조정하고, 서로 직접 상호작용하며, 보다 안전하고 분산적인 방식으로 자신을 통치하는 수단이다. 네트워크를 관리하고 어떤 콘텐츠를 표시할지 규정하기 위해, 이러한 플랫폼은 분산 방식으로 실행되며 쉽게 참여자들은 거래를 네트워크에 게시하기 위해 마이크로 수수료를 지불해야 하며, 이것은 네트워크를 유지하고 운영하는 데 기여하는 사람들에게

수수료로 지불되어 가치의 재분배와 불특정 다수의 참여를 독려하는 유인 책으로 적절히 사용된다.

이와 마찬가지로, 블록체인을 기반으로 한 본 프로젝트는 어떤 중개업체나 중개자 없이 독립적으로 운영된다. 이 플랫폼은 블록체인 기술에 의존하여 구매자와 판매자가 중앙집중화된 중개인을 거치지 않고도 서로 직접적으로 상호작용할 수 있도록 한다. 플랫폼에 제품이나 용역을 자유롭게 등록할 수 있으며, 네트워크에 연결된 모든 사용자가 이를 볼 수 있다. 구매자가 해당 제품의 가격 및 거래 조건에 동의하면, Ethereum 블록체인에 에스크로 트랜잭션이 생성된다. 구매자가 Smart Contract 으로 대금을 송금하면, 판매자는 상품을 배송하고, 구매자는 상품을 수령한 후 판매자는 계정에서 자금을 인출한다. 판매자와 구매자 간의 거래 종료 후 거래에 대한 평가를 기록하여 거래 당사자들이 더 많은 거래를 성실하게 할수록, 그들은 신용을 얻을 수 있고 커뮤니티에 더 큰 영향력을 미칠 수 있게 된다. 본 프로젝트의 공유 경제 모델은 중앙 관리자가 없는 Smart Contract 을 활용하기 때문에 최소화된 수수료만으로 서비스 운영이 가능하다. 또한, DApp(Decentralized Application)형태로 서비스가 제공되기 때문에 DoS 공격 (Denial of Service attack, 서비스 거부 공격), DDoS 공격(Distributed Denial of Service attack, 분산 서비스 거부 공격) 등으로부터 안전을 도모할 수 있으며 데이터 위변조 공격으로부터 안정성도 확보할 수 있다. [23]

블록체인 기술은 조직의 형태를 갖추지 않고도 분산화된 새로운 형태의 조직의 출현을 촉진하고 이사, CEO 또는 기타 계층 구조가 없는 이러한 조직은 모든 개인이 수평적인 관계에서 공동으로 관리한다. 블록체인 기술은 불특정 다수가 참여하는 훨씬 더 협력적인 형태를 지원할 수 있고 중간 사업자가 없기 때문에, 이러한 플랫폼 내에서 생산된 가치는 가치 창출에 기여한 사람들 사이에서 더 균등하게 재분배될 수 있다.

이러한 "협력주의"의 새로운 기회와 함께, 우리는 진정한 공유나 협력적 경제로 나아갈 수 있고, 이는 소수의 거대 중개업자들이 통제하지 않지만, 개개인에 의해 스스로 통제될 수 있음을 의미한다. 인터넷이 등장

한 초기에는 개인과 기업 경쟁하게 될 것으로 예상하는 많은 전문가들을 볼 수 있었다. 그러나 시간이 흐르면서, 큰 기업들이 형성되고 디지털 환경을 지배하면서 초기 인터넷 시대의 꿈과 희망은 대부분 사라졌다. 이제 이러한 희망을 다시 꿈꿀 수 있는 새로운 기회를 얻었다. 블록체인 기술을 통해 하향식 계층 조직의 모델을 분산형 상향식 협업 시스템으로 대체할 수 있다. 이러한 변화는 우선 부의 분배 방식을 바꾸어 사람들이 공동의 이익을 창출하는 방향으로 협력할 수 있게 하는 동시에 모든 사람들이 그들의 노력과 기여에 대해 적절히 보상받을 수 있게 할 것이다.

우리 사회가 진정한 공유 경제라는 개념을 정말로 소중하게 생각하고, 개인이 자신의 노력에 대해 공정한 보상을 받을 수 있게 된다면 우리 모두가 창의적이고 자발적으로 기술에 종사하고 실험하여 제공 할 수 있는 새로운 기회를 탐구 할 수 있을 것이고 본 프로젝트는 실제 시스템 구현을 통해 동작성을 입증하였다.

참고 문헌

[1] S. Nakamoto, “Bitcoin: A peer-to-peer delectronic cash system,” 2009. [온라인]. Available: http://www.bitcoin.org/bitcoin.pdf.

[2] M. A. Cusumano, The Bitcoin Ecosystem, 제 57, Communications of the ACM, 2014, pp. 22-24.

[3] 김자봉, 비트코인 거래 메커니즘의 분석과 시사점, 2014(11), pp. 1-82.

[4] JC. A. Brito, BITCOIN A PRIMER FOR POLICYMAKERS, 2013, pp. 3-12.

[5] 김태오, 가상화폐의 이용현황과 시사점: Bitcoin 과 Linden Dollar 를 중심으로", 2013, pp. 33-64.

[6] D. A. Nair, The bitcoin innovation, crypto currencies and the Leviathan., 2018, pp. 1-19.

[7] G. Hogben, Security issues and recommendations for online social networks, 2007, p. 1–36.

[8] T. Kim, On the transaction cost of Bitcoin. Finance Research Letters, 2017, pp. 300-305.

[9] S. Manski, Building the blockchain world: Technological commonwealth or just more of the same?, 2017, pp. 511-522.

[10] D. Descôteaux, Bitcoin More Than a Currency, a potential for innovation.

[11] 김진화, NEXT MONEY 비트코인, 서울, 2013.

[12] N. Shi, A new proof-of-work mechanism for bitcoin. Financial Innovation, 2016.

[13] G. Wood, Ethereum: A secure decentralized generalised transaction ledger, Ethereum Project Yellow Paper 151, 2014.

[14] G. Selgin, Synthetic commodity money, Journal of Financial Stability, 2015, pp. 92-99.

[15] V. Marko, The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication, Springer International Publishing, 2015.

[16] R. Modi, Solidity Programming Essentials, 2018.

[17] F. C. Maldonado, Introduction to Blockchain and Ethereum, Packt Publishing , 2018.

[18] D. Flanagan, JacaScript:The Definitive Guide:Activate Your Web Pages, O'Reilly, pp. 4-8.

[19] D. Goodman, Dynamic HTML, O'Reilly, pp. 107-110.

[20] J. Duckett, HTML and CSS: Design and Build Websites, John Wiley & Sons, 2011.

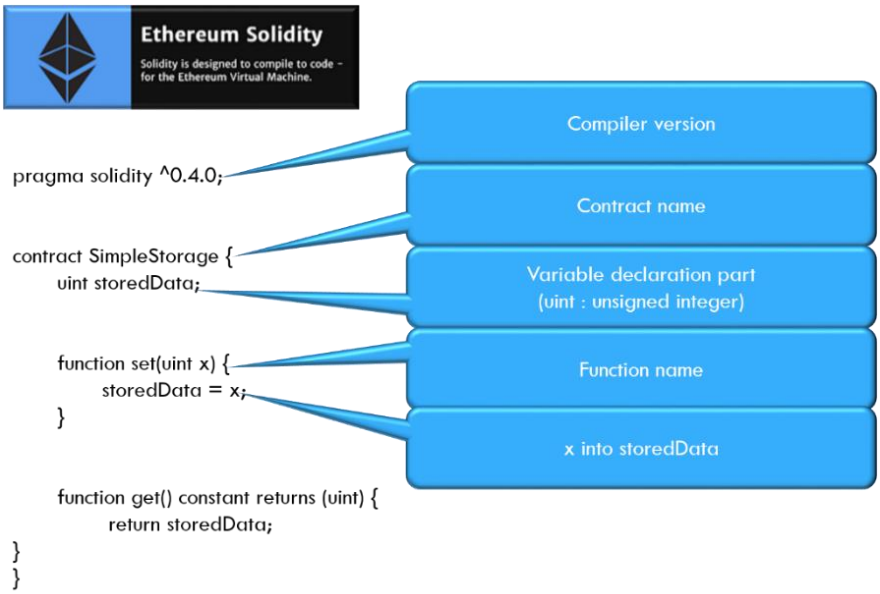
[21] L. Wang, Measuring large-scale distributed systems: case of bittorrent mainline dht. In Peer-to-Peer Computing (P2P), 213, p. 1–10.

[22] B. Cohen, Incentives build robustness in bittorrent, In Workshop on Economics of Peer-to-Peer systems, 2003, p. 68–72.

[23] H. Subramanian, Decentralized Blockchain - Based Electronic Marketplaces, Communications of the ACM, 2018, pp. 78-84.

[24] Hayden Covington, Blockchain and Bitcoin, International Journal of Cyber, pp. 27-37.

부 록 1 : Solidity 예시 코드



<그림 34> Solidity 언어 기본 구조

```
pragma solidity ^0.4.12;
contract SharingEco {
  struct transaction {
    uint ether;
    uint coll;
    address senderaddress;
    address receiveraddress;
    uint tid;
    uint tdate;
    bool taccept;
    bool senderCW;
    bool receiverCW;
    bool tok;
  }
```

```
transaction[] public transactions;
uint numt = 0;
mapping (address => uint[]) tsent;
mapping (address => uint[]) trec;
address owner;
modifier isReceiver(uint tid){
  require(transactions[tid].receiver == msg.sender);
  _;
}
modifier isSender(uint tid){
  require(transactions[tid].sender == msg.sender);
  _;
}
modifier isNotComplete(uint tid){
  require(!transactions[tid].complete);
  _;
}
function SharingEco() {
  owner = msg.sender;
}
function tmake(address receiveraddress,
uint tdate,
uint tcoll)
payable {
  require(msg.value != 0);
  transactions.push(transaction({
      tether: msg.value,
      tcoll: collateral,
      tsender: msg.sender,
```

```
        treceiver: receiver,
        tid: numTransactions,
        tdate: deadline,
        taccept: false,
        cwsender: true,
        cwreceiver: false,
      tcomplete: false}));
   tsent[msg.sender].push(tnum);
   trec[receiver].push(tnum);
   tnum++;
}
function taccept(uint tid) payable isReceiver(tid)
isNotComplete(tid) {
   require(!transactions[tid].taccept && msg.value ==
   transactions[tid].tcoll);
   transactions[tid].taccept = true;
   transactions[tid].cwsender = false;
   transactions[tid].tdate =
   now + (transactions[tid].tdate * 1 days);
}
function wreceiver(uint tid) isReceiver(tid) isNotComplete(tid) {
   require(transactions[tid].cwreceiver ||
   ((now > transactions[tid].tdate) && (transactions[tid].taccept)));
   uint tcoll = transactions[tid].tcoll;
   uint fee = transactions[tid].tether / 500;
   uint tether = transactions[tid].tether - fee;
   transactions[tid].tcomplete = true;
   msg.sender.transfer(tether);
   owner.transfer(fee);
```

```
        transactions[tid].receiver.transfer(tcoll);
    }
    function wsender(uint tid) isSender(tid) isNotComplete(tid) {
        require(transactions[tid].cwsender);
        uint tether = transactions[tid].tether;
        transactions[tid].tcomplete = true;
        if (transactions[tid].taccept) {
            uint tcoll = transactions[tid].tcoll;
            transactions[tid].receiver.transfer(tcoll);
        }
        msg.sender.transfer(tether);
        }
    function tfinish(uint tid) isSender(tid) isNotComplete(tid) {
        require(!transactions[tid].cwreceiver);
        transactions[tid].cwreceiver = true;
        transactions[tid].cwsender = false;
    }
    function trefund(uint tid) isReceiver(tid) isNotComplete(tid) {
        require(transactions[tid].taccept);
        transactions[tid].cwreceiver = false;
        transactions[tid].cwsender = true;
    }
function stdget(uint tid) isSender(tid) constant returns(
            uint tether,
            uint tdate,
            bool taccept,
            bool wsender,
            uint tcoll,
            bool tcomplete) {
```

```
 return(
          transactions[id].tether,
          transactions[id].tdate,
          transactions[id].accept,
          transactions[id].cwsender,
          transactions[id].tcoll,
          transactions[id].complete
  );
  }
  function rtdget(uint tid) isReceiver(tid)
  constant returns(
          uint tether,
          uint tdate,
          bool taccept,
          bool cwreceiver,
          uint tcoll,
          bool tcomplete) {
          bool twithdraw = false;
     if (transactions[tid].cwreceiver || now >
transactions[tid].tdate) {
        withdraw = true;
     }
     return(
          transactions[tid].tether,
          transactions[tid].tdate,
          transactions[id].accept,
          withdraw,
          transactions[tid].tcoll,
          transactions[tid].complete);
```

```
}
function stget() constant returns(uint[] ids,
   address[] addresses) {
   uint length = 0;
   for (uint p = 0; p < tsent[msg.sender].length; p++){
      var z = transactions[tsent[msg.sender][p]];
      if (!z.tcomplete) {
         length++;
      }
   }
   uint[] memory returnIds = new uint[](length);
   address[] memory returnAddresses = new address[](length);
   uint num = 0;
   for (uint i = 0; i < tsent[msg.sender].length; i++) {
      var x = transactions[tsent[msg.sender][i]];
      if (!x.complete) {
         returnIds[num] = x.tid;
         returnAddresses[num] = x.receiver;
         num++;
      }
   }
   return(returnIds, returnAddresses);
}
function rtget() constant returns(uint[] ids,
 address[] addresses) {
uint length = 0;
for (uint p = 0; p < tsent[msg.sender].length; p++) {
   var z = transactions[tsent[msg.sender][p]];
      if (!z.tcomplete) {
```

```
            length++;
          }
        }
        uint[] memory returnIds = new uint[](length);
        address[] memory returnAddresses = new address[](length);
        uint num = 0;
        for (uint i = 0; i < trec[msg.sender].length; i++) {
          var x = transactions[trec[msg.sender][i]];
          if (!x.complete) {
          returnIds[num] = x.id;
          returnAddresses[num] = x.receiver;
          num++;
          }
        }
        return(returnIds, returnAddresses);
    }
    function () {
        revert();
    }
}
```

부 록 2: Java Script 예시 코드

```
window.tmake = function(){
   $("#transMsg").hide();
   let receiver = $("#receiver").val();
   let tethert = $("#tether").val();
   let tdate = $("#tdate").val();
   let tcoll = $("#tcoll").val();
   Sharingeco.deployed().then(function(contractInstance){
contractInstance.tmake(receiver, tdate, web3.toWei(tcoll,
'ether'), {value: web3.toWei(tether, 'ether'), from:
web3.eth.accounts[0]}).then(function(v){
         $("#transMsg").show();
         console.log(v);
      });
   });
}
window.stcheck = function(){
   $("#tableSent").show();
   $("#sentId").show();
   $("#checkSentId").show();
   $("#recId").hide();
   $("#checkRecId").hide();
   $("#tableRec").hide();
   $("#tableRecTrans").hide();
   $("#sendMsg").hide();
   $("#recMsg").hide();
   $("#tableInfo").html("");
```

```
  Sharingeco.deployed().then(function(contractInstance){
     contractInstance.stget.call({from:web3.eth.accounts[0]}). then(
 function(v){
       let ids = v[0];
       let addresses = v[1];
       for(let i=0;i<ids.length;i++){
         $("#tableInfo").append("<tr><td>"    +    ids[i].toString()    +
"</td><td>"
        + addresses[i] + "</td></tr>");
       };
     });
  });
}
window.rtcheck = function(){
  $("#tableRec").show();
  $("#recId").show();
  $("#checkRecId").show();
  $("#tableSent").hide();
  $("#tableSentTrans").hide();
  $("#sentId").hide();
  $("#checkSentId").hide();
  $("#sendMsg").hide();
  $("#recMsg").hide();
  $("#tableInfoR").html("");
  Sharingeco.deployed().then(function(contractInstance){
     contractInstance.rtget.call({from:
web3.eth.accounts[0]}).then(function(v){
       let ids = v[0];
       let addresses = v[1];
```

```
        for(let i=0;i<ids.length;i++){
          $("#tableInfoR").append("<tr><td>" + ids[i] + "</td><td>"
         + addresses[i] + "</td></tr>");
        };
      });
   });
}
window.stcheck = function(){
   let tid = $("#sentId").val();
   Sharingeco.deployed().then(function(contractInstance){
      contractInstance.stdget.call(id,
      {from: web3.eth.accounts[0]}).then(function(v){
        let tether = parseFloat(v[0].toString());
        if(tether == 0 || v[5]){
          $("#tableSentTrans").hide();
          $("#sendMsg").text("None").show();
          return;
        }

        $("#tableSentTrans").show();
        $("#sendMsg").hide();
        let tdate = parseFloat(v[1].toString());
        $("#tableId").text(id);
        $("#tableAmount").text(web3.fromWei(tether,"ether"));
        if(v[2]){
          $("#tableAccepted").text("승인");
              $("#tableFinish").html("<a href='#' onclick='comment()'
class='btn btn-warning'>Rating</a><a href='#' onclick='finish()'
          class='btn btn-default'>Finalizing</a>");
```

```
            let tdate = new Date(tdate * 1000);
            let dateString = "";
            dateString += (date.getMonth() + 1) + "/";
            dateString += date.getDate() + "/";
            dateString += date.getFullYear() + " ";
            dateString += date.getHours() + ":";
            dateString += date.getMinutes();
            $("#tableDeadline").text(dateString);
            $("#tableDispute").html("<a href='#' onclick='dispute()'
class='btn
            btn-default'>Dispute</a><br><input type='number' id='add-
Dispute'
            placeholder='Days' style='width:75px'>");
        }
        else{
            $("#tableAccepted").text("미승인");
            $("#tableDeadline").text(deadline + " day(s)");
            $("#tableFinalize").text("미승인");
            $("#tableDispute").text("미승인");
        }
        if(v[3]){
            $("#tableWithdrawal").html("<a href='#'
            onclick='wsender()' class='btn btn-default'>인출</a>");
        }
        else{
            $("#tableWithdrawal").text("인출 불가");
        }
        let tcoll = parseFloat(v[4].toString());
        $("#tableCollateral").text(web3.fromWei(tcoll, 'ether'));
```

```
  });
});
}
window.comment = function(){
window.open("comment.html", "_myblank", "width=900, height=700,
toolbar=no, menubar=no, scrollbars=no, resizable=no" );
}
window.rtcheck = function(){
  let id = $("#recId").val();
  Sharingeco.deployed().then(function(contractInstance){
    contractInstance.rtdget.call(tid, {from:
    web3.eth.accounts[0]}).then(function(v){
      $("#tableIdR").text(tid);
      let tether = parseFloat(v[0].toString());
      if(tether == 0 || v[5]){
        $("#tableRecTrans").hide();
        $("#recMsg").text("None").show();
        return;
      }
      $("#tableRecTrans").show();
      $("recMsg").hide();
      let tdate = parseFloat(v[1].toString());
      $("#tableAmountR").text(web3.fromWei(tehter,"ether"));
      if(v[2]){
        $("#tableAcceptedR").text("승인");
        let tdate = new Date(tdate * 1000);
        let dateString = "";
        dateString += (date.getMonth() + 1) + "/";
        dateString += date.getDate() + "/";
```

```
            dateString += date.getFullYear() + " ";
            dateString += date.getHours() + ":";
            dateString += date.getMinutes();
            $("#tableDeadlineR").text(dateString);
            $("#refund").html("<a href='#' onclick='refund()' class='btn
          btn-default'>환불</a>");
            $("#refundH").show();
            $("#refund").show();
          }
          else{
            $("#tableAcceptedR").html("<a href='#'
          onclick='taccept()' class='btn btn-default'>승인</a>");
            $("#tableDeadlineR").text(tdate + " day(s)");
            $("#refundH").hide();
            $("#refund").hide();
          }
          if(v[3]){
            $("#tableWithdrawalR").html("<a href='#' onclick=
            'recWithdrawal()'
          class='btn btn-default'>인출</a>")
          }
          else{
            $("#tableWithdrawalR").text("인출 불가");
          }
          let tcoll = parseFloat(v[4].toString());
          $("#tableCollateralR").text(web3.fromWei(tcoll, 'ether'));
      });
});
}
```

```
window.taccept = function(){
   let tid = $("#tableIdR").text();
   let tcoll = $("#tableCollateralR").text();
   Sharingeco.deployed().then(function(contractInstance){
      contractInstance.taccept(id, {gas: 140000, value:
      web3.toWei(tcoll, 'ether'), from: web3.eth.accounts[0]}).
      then(function(){
      });
   });
}
window.wrec = function(){
   let tid = $("#tableIdR").text();
   Sharingeco.deployed().then(function(contractInstance){
      contractInstance.wreceiver(tid, {gas: 140000, from:
 web3.eth.accounts[0]}).then(function(v){
      });
   });
}
window.wsender = function(){
   let tid = $("#tableId").text();
   Sharingeco.deployed().then(function(contractInstance){
      contractInstance.wsender(tid, {gas: 140000,
      from: web3.eth.accounts[0]}).then(function(v){
      });
   });
}
window.tfinish = function(){
   let tid = $("#tableId").text();
   Sharingeco.deployed().then(function(contractInstance){
```

```
        contractInstance.tfinish(id, {gas: 140000,
        from: web3.eth.accounts[0]}).then(function(v){
        });
    });
}
window.trefund = function(){
    let tid = $("#tableIdR").text();
    Sharingeco.deployed().then(function(contractInstance){
        contractInstance.trefund(tid, {gas:140000, from:
        web3.eth.accounts[0]}).then(function(v){
        });
    });
}
```

부 록 3 : HTML 예시 코드

Transaction.html

```
<html>
<head>
  …
</head>
<body class="container" >
  <h1 id="title">공전원 프로젝트</h1>
        <div class="row">
    <div class="row" id="make">
      <h2>거래 생성</h2><br>
      <input type="text" id="readdress"
       placeholder="수신인 주소" style="width:500px"><br>
      <input type="number" id="ether"
       placeholder="이더" style="width:500px"><br>
      <input type="number" id="date"
       placeholder="종료일" style="width:500px"><br>
      <input type="number" id="collmount"
       placeholder="보증금" style="width:500px"><br>
        <a href="#" onclick="checkCredit()" class="btn btn-danger">
      신용도 확인</a>
 <a href="#" onclick="makeTransaction()" class="btn btn-danger">
        거래 개시</a><br>
  <span id="tmessage" style="display: none"><h2>종료</h2></span>
    </div>
</body>
</html>
```

CheckS.html

```
<html>
<head>
  …
</head>
<body class="container" >
<div class="row" id="stran">
        <h2 style="width:500px;" id="setran">보낸 거래 확인</h2><br>
        <a href="#" onclick="checkST()"  id="checkS">거래 확인</a>
        <table id="tsent" style="display: none">
  <thead><tr><th>거래 번호</th><th>수신자 주소</th></tr></thead>
          <tbody id="tinfo"></tbody></table>
        <input type="text" id="sid" placeholder="거래 번호"><br>
        <a href="#" onclick="checkST()" id="checksid"
         style="display: none">거래 확인</a>
        <table class="table id="tablest" style="display: none">
          <thead>
         <th>거래번호</th><th>금액</th><th>기한</th><th>보증금</th>
      <th>거래승인</th><th>인출</th><th>최종승인</th><th>연기</th>
          <thead>
          <tbody><tr><td id="tid"></th><td id="tether"></td>
              <td id="tdate"></td><td id="tcoll"></td>
              <td id="taccepted"></td><td id="twithdrawal"></td>
              <td id="tfinalize"></td><td id="tdis-
pute"></td></tr></tbody>
        </table>
  </div>
  </body>
</html>
```

CheckR.html

```
<html>
<head>
…
</head>
<body class="container" >
     <h3 id="smsg" style="display:none"></h3>
     </div><div class="row">
       <h2 id="rec">수신 거래 확인</h2><br>
       <a href="#" onclick="checkrt()">수신 거래 확인</a>
       <table id="tabler" style="display: none">
  <thead><tr><th>거래 번호</th><th>수신자 주소</th></tr></thead>
     <tbody id="tableir"> </tbody></table>
       <h3 id="recm" style="display:none"></h3>
       <input type="text" id="recid" placeholder="거래 번호"><br>
 <a href="#" onclick="checkrt()" id="checkrid" style="display: none">
         거래 확인</a><br>
         <table id="tablert" style="display: none">
         <thead>
        <th>거래번호</th><th>금액</th><th>기한</th><th>보증금</th>
           <th>승인</th><th>인출</th><th id="reh">환불</th>
         </thead><tbody>
           <tr><td id="tidr"></td><td id="tr"></td><td id="tdr"></td>
             <td id="tcir"></td><td id="tar"></td><td id="twr"></td>
             <td id="ref"></td></tr>
         </tbody> </table> </div></div>
  </body>
</html>
```

Check.html

```
<html>
    <head>
        …
    </head>
    <body>
        <center>
        <header>
        <h1 class="title"> 공전원 프로젝트 </h1>
        <h2 class="subtitle">All Things are Possible</h2>
    </header></center>
         <div id="app"></div>
         <div id="modal-container"></div>
        <section class="section" id="lookup">
 <div class="container"><h2 class="title is-2">신용 조회</h2>
   <p class="subtitle is-5">주소에 대한 신용도를 확인합니다.</p>
<form id="lookup-form" onsubmit="submitLookup(this);
return false;"> <div class="field has-addons"><p class="control">
<input class="input" name="address" type="text" placeholder="조회
대상 이더리움 주소"></p><p class="control">
          <button id="lookup-button" type="submit" class="button
         is-info has-icons-left">
           <i class="fa fa-search"></i>조회</button>
        <button align="center" class="button is-info has-icons-left"
onclick="closeWindow(); return false;"> 닫기 </button>
   </p></div></form><p id="lookup-score" class="hide"></p></div>
              </section><section class="section" id="rate"></section>
</html>
```

Rating.html

```
<html>
    <head>
        …
    </head>
    <body><center><header>
<h1 class="title"> GSEP Project </h1>
<h2 class="subtitle">All Things are Possible</h2>
    </header></center>
        <div id="app"></div> <div id="modal-container"></div>
<section class="section" id="rate">
<div class="overlay">
<p>일부 브라우저에서는 사용할 수 없습니다. </p></div>
<div class="container"><h2 class="title is-2">거래 평가</h2>
<p class="subtitle is-5">거래 상대방에 대한 평가를 기록합니다.</p>
<!--<div class="field has-addons rating-gradient">
<p class="control"><input class="input" id="rating-address"
type="text" placeholder="평가할 상대방 이더리움 주소"></p>
<p class="control"><button onclick="selectRating(1)">
매우 불만족</button></p>
<p class="control"><button onclick="selectRating(2)">
불만족</button></p>
<p class="control"><button onclick="selectRating(3)">
보통</button></p>
<p class="control"><button onclick="selectRating(4)">
만족</button></p>
<p class="control"><button onclick="selectRating(5)">
매우 만족</button></p></div>
```

```
<p class="control submit"><button class="button is-info"
onclick="submitRating(); return false;">평가 완료</button>
<button class="button is-info" onclick="closeWindow(); return false;">
닫기 </button></p>
 <input type="hidden" id="rating-value" value="0"></div>
 <p id="rating-results" class="hide">
 </p><p id="rating-errors" class="hide"></p>
</div>  </section>
</body>
</html>
```

Abstract

A Study on the Application of Smart Contract based on Block Chain

Keunsoo Yoon
Department of Engineering Practice
Graduate School of Engineering Practice
Seoul National University

Transactions over the internet are mostly governed by companies and they are inherently biased towards sellers when dispute arises. A certain percentage of buyer fraud, transaction fees, and currency conversion fees are unavoidable, contributing to high prices.

Existing payment by cryptocurrency such as Bitcoin solves the issue for merchants; however, it provides no protection for buyers if merchants fail to fulfil their obligations.

Through this project, online transaction methods with escrow protection are not limited to companies. Anyone can be the "Seller" for their community, charging a fee and offering their escrow services. By introducing free market competition, I hope it drives lower escrow fee and drives better service level for anyone who transacts online.

Programmable Ethereum Blockchain was used to implement a direct transaction between individuals that did not require a relay. The front-end part is coded using HTML5 and JAVA Script, the back-end part is coded using Solidity Language and Web3.js.

Although it is currently used the most, new languages continue to emerge and the fast-changing trend is the hallmark of the blockchain industry. Therefore, the report was mainly described using Pseudocode and the project can be used as another blockchain language when applied with examples of transforming into Solidity.

Keywords : Blockchain, Ethereum, Sharing Economy, Smart Contract, Bitcoin

Student Number : 2017-21117