블록체인 기반 학교 재정장부 자동 관리 시스템

*신승민 , *서상민 , *송동훈 , *최효선 , *Grzegorz Rypesc , *고석주(교신저자)
**이광열
*경북대학교 컴퓨터학부
**㈜업라이프
E-mail : steve6238@naver.com, sangmin95@gmail.com, thdehdgns@gmail.com, gytjs9611@gmail.com, g.rypesc@wp.pl, sjkoh@knu.ac.kr, qazwsdcrv@naver.com

An Automated Management System for A School Account Book Based on Blockchain Technology

*Seung-min Shin , *Sang-min Seo , *Dong-hoon Song , *Hyo-sun Choi , *Grzegorz Rypesc, *Seok-Joo Koh, **Gwang-Yeol Lee

*School of Computer Science and Engineering, Kyungpook National University
**Uplife Company

요 약

기존의 중앙 집중적 학교 재정장부 관리 시스템은 보안에 취약하고 투명하지 않아 공금 횡령, 장부 조작 등의 문제 가능성이 크다. 이는 학생회와 학생들 사이에 갈등을 일으키고 큰 사회적 문제가 되었다. 이에 본 논문에서는 블록체인 기술을 활용하여 학교 재정장부를 안전하고 투명하게 관리하는 방법을 제시한다. 더 나아가 자동화 기술을 도입해 장부를 기록하는 과정을 단순화하고자 한다.

I. 서론

학생들은 등록금과 더불어 학생회비를 내면서 학교를 다니지만, 그 학생회비가 어떻게 쓰이는지 잘 알지 못한다. 기존의 재정장부 관리 시스템으로는 실시간으로 사용된 내역을 볼 수 없고, 학생회비가 올바르게 사용되었는지 알지 못하는 문제점이 있다. 이로 인해 수많은 대학가에서는 학생회비 횡령 사건이 발생하여 투명한 재정관리 시스템에 대한 연구가 필요한 실정이다.

따라서 본 논문에서는 'Solidity'를 활용한 블록체인 기반의 자동 재정장부 관리 시스템을 제안한다. 기존 시스템과는 달리, 재정 장부가 자동으로 업데이트되고 그 데이터를 블록에 저장하기 때문에, 학생들이 직접 실시간으로 사용내역을 확인할 수 있으며, 데이터의 위변조가 어려워진다.

II. 관련연구

2.1 Smart Contract

Smart Contract란, 블록체인 기술을 기반으로 하여 설정된 계약 조건을 만족시키면 자동으로 계약이 실행되고, 중개자 없이 P2P로 쉽고 편리하게 계약을 체결하는 기술이다. 개발자가 계약의 조건과 내용을 코드로 작성하기 때문에 이는 금융거래, 부동산 계약 등 다양한 형태의 계약에 활용이 가능하다.

이러한 Smart Contract를 구현하기 위해서, 'Solidity'라는 프로그래밍 언어를 사용한다. Solidity는 컨트랙트 기반의 정적타입 언어로, EVM(Ethereum Virtual Machine) 상에서 작동하는 Smart Contract의 개발을 위해 설계되었다. EVM에서 작동 가능한 바이트코드로 컴파일되며, 개발자는 Solidity를 통해 Smart Contract에 담아 Dapp(Decentralized application)을 구현할 수 있다.

III. 기존 시스템

우선, 기존의 학생회비 관리 시스템은 다음과 같다. 매년 재정부장은 자신의 명의로 된 계좌를 개설한 후 총회에서 회비 사용 내역을 엑셀 파일과 통장 사본을 공개한다. 이와 같은 시스템은 매년 재정부장이 계좌를 새로 개설해야 하고, 인계 과정 또한 번거롭다는 문제점이 있다. 무엇보다도 가장 큰 문제는 실시간으로 회비가 사용된 내역을 볼 수 없고, 실제로 사용된 금액이 올바르게 사용되었는지 확인할 방법이

없기 때문에 부정부패가 발생하기도 한다는 것이다.

IV. 개선 시스템

문제점이 많은 기존 시스템을 개선해 새로운 시스템을 고안해보았다. 개선된 방법은 다음과 같다. 학생회는 학과 명의로 된 통장을 개설하고 E-Mail 결제 알림 서비스를 신청한다. 그러면 사용내역을 은행으로부터 실시간으로 수신 받을 수 있다.

메일에서 데이터를 추출해 자동으로 블록에 저장하고 검증 후 체인에 연결한다. 학생들은 웹을 통해 거래내역을 실시간으로 열람할 수 있다. 또한 학생회는 데이터베이스에 사용내역을 검증할 수 있는 사진 및 문서들을 저장하여 웹서버에 올린다.

학생들은 그 자료들을 열람할 수 있고 의문점이 들면 이의 제기를 할 수 있다.

4.1 제안 시스템 설계

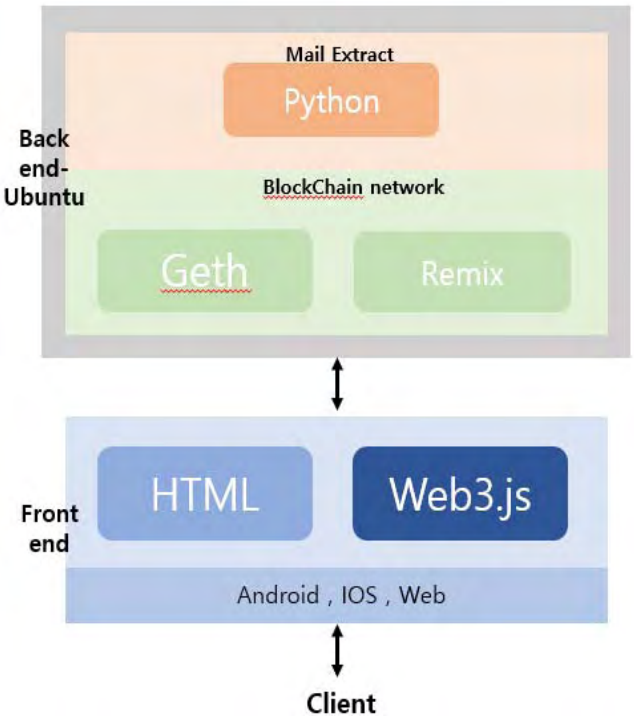
(그림 1)는 제안한 시스템의 구조를 나타내는 개요도이다. 이 시스템은 크게 메일 추출, 블록체인 생성, 웹서버로 구성된다. 학생회 계좌에서 거래가 발생하면 E-Mail로 거래내역 알림이 전송된다. Ubuntu Server에서는 은행에서 메일이 수신될 때마다 실시간으로, python을 이용하여 E-Mail로 수신받은 html 파일 속 거래 정보를 추출한다. 추출된 데이터로 Solidity로 블록을 생성한다.

(그림 2) 그 후 검증 과정을 거쳐서 블록체인에 연결하여 업로드 한다. 이러한 개선된 시스템은 자동 입출금 관리가 가능해지고, 시스템의 유저들은 회비가 사용되는 내역을 실시간으로 확인할 수 있다. 이로 인해, 기존 시스템에 비해 훨씬 효율적인 재정관리가 가능해지고, 투명성을 제고할 수 있다.

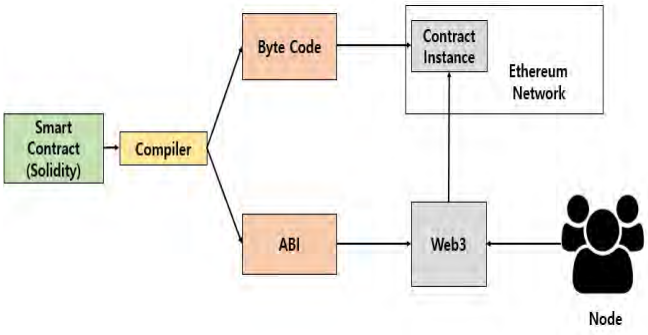
4.2 solidity를 활용한 스마트 컨트랙트 흐름

Solidity로 작성된 스마트 컨트랙트(Smart Contract)를 컴파일하면 바이트 코드(Byte Code)와 ABI(Application Binary Interface)를 얻을 수 있다. ABI로부터 스마트 컨트랙트 객체를 생성한 후 인스턴스를 생성한다. 스마트 컨트랙트를 하나의 트랜잭션처럼 생성하는데 이 때 트랜잭션의 송신자와 바이트 코드를 포함시킨다.

해당 트랜잭션이 포함된 블록을 채굴하게 되면 스마트 컨트랙트 주소가 생성된다. 이 주소를 이용하여 Node(사용자)가 Web app을 통해 스마트 컨트랙트에 접근할 수 있다.



(그림 1) 시스템 개요도



(그림 2) 블록체인 내부 과정

4.3 solidity를 활용한 입출금내역저장 스마트 계약 예시

```
pragma solidity ^0.4.8;
contract Trade{
   address public owner;
    struct trade{ /* 입출금 내역을 담을 구조체 */
    uint Trade_Num; //거래번호
    string Trade_Date; //거래일자
    uint Trade_Cost; //거래금액
    uint Sum_Cost;// 남은잔고
    string Trade_Place; // 거래장소
    string Type; //거래유형(입금or출금or취소)
    }
    trade[] public TradeList;
```

(그림 3)입출금내역 데이터

```
trade[] public TradeList;
function Trade()//계약 유포자를 owner 로 지정
{
    owner = msg.sender;
}
function setValue(string _Date, uint _Cost ,uint _sumCost,
string _Place,string _what)returns(uint)  {
        if(msg.sender != owner) // 계약 유포자(관리자) 가아니면 set불가
    {
        throw;
    }
    TradeList.length += 1;//가변 배열 이기 때문에 길이 1증가
    uint keyIndex = TradeList.length -1;
    //인덱스는 자동으로 받아오기 때문에 관리자여도 수정 불가

    TradeList[keyIndex].Trade_Num=keyIndex;
    TradeList[keyIndex].Trade_Date=_Date;
    TradeList[keyIndex].Trade_Cost=_Cost;
    TradeList[keyIndex].Sum_Cost=_sumCost;
    TradeList[keyIndex].Trade_Place=_Place;
    TradeList[keyIndex].Type=_what;
    return keyIndex;
}
```

(그림 4)입출금 내역 저장

계약 유포자(관리자) 가 아니면 입출금 기록을 작성할 수 없으며 관리자 역시 Index가 자동으로 지정되기 때문에 함부로 수정할 수 없다.

```
function getValue(uint _key)constant
returns (uint,string ,uint,uint,string,string)
{ //거래내역에 관한 기록을 가져옴
return (TradeList[_key].Trade_Num,TradeList[_key].Trade_Date
,TradeList[_key].Trade_Cost,TradeList[_key].Sum_Cost,
TradeList[_key].Trade_Place,TradeList[_key].Type);
}
```

(그림 5)거래내역 가져오기

index 번호를 이용해 거래내역 기록을 가져올 수 있다.

```
> contract.setValue.sendTransaction("2018-07-06",1000000,55000000,"Daiso","withd
raw",{from : eth.accounts[0],gas:1100000})

"0xe26ee4746f2b67edc4fd8c83f809827f3050482550e50441346e90bb24860802"
> contract.setValue.sendTransaction("2018-07-07",5000000,6000000,"Harry","Deposi
t",{from : eth.accounts[0],gas:1100000})
"0xe99732bc7c138690b1d1b4c264ad312dee05980cc786132ca5e626d30c4d994a"
```

(그림 6)블록에 거래내역 저장 예시

```
> contract.getValue(1)
[1, "2018-07-06", 1000000, 55000000, "Daiso", "withdraw"]
> contract.getValue(2)
[2, "2018-07-07", 5000000, 6000000, "Harry", "Deposit"]
>
```

(그림 7)Geth 블록에서 거래내역 불러오기 예시

V. 결론

기존의 중앙화된 재정 장부 시스템은 조작의 위험이 크고 보안에 취약하다. 따라서 불투명하고 신뢰받지 못하며 많은 사회적 문제를 일으킨다.

이 논문에서 블록체인 기술을 이용해 조작이 불가능하고 안전한 재정장부 시스템을 구축함으로써, 투명성과 신뢰성을 제고하고 각종 학생회비 횡령, 장부 조작 사건들을 뿌리 뽑을 수 있을 것이라고 기대한다. 또한, 학생회에서도 학생회비를 합리적으로 사용할 것이라고 기대한다. 더불어 장부 관리를 자동화하여 번거로운 데이터 입력과정을 단순화했다. 학생들이 학생회비를 주도적으로 납부하고 재정장부에도 많은 관심을 가질 것이라고 기대한다.

Acknowledgement

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW중심대학사업의 연구결과로 수행되었음 (2015-0-00912)

참고문헌

[1] 아카하네 요시하루, 아이케이 마나부. “블록체인 구조와 이론”. 파주:위키북스

[2] 와타나베 아츠시, 마츠모토 유타, 니시무라 요시카즈, 시미즈 토시야. “블록체인 애플리케이션 개발 실전 입문”. 파주:위키북스

[3] Michael Crosby, Nachiappan, Pradan Pattanayak, Sanjeev Verma, Vignesh Kalyanaraman. "Blockchain technology : Beyond bitcoin". Applied Innovation Review(2016), 2

[4] Mougayar, William. 《The Business Blockchain: Promise, Practice, and Application of the Next Internet Technology》

[5] Ethereum, https://www.ethereum.org/

[6] Ronny Hans, Hendrik Zuber, Amr Rizk, Ralf Steinmetz. "Blockchain and Smart Contracts: Disruptive Technologies for the Insurance Market".

[7] Solidity, https://github.com/ethereum/solidity