이더리움 스마트 컨트랙트 상태 모니터링 시스템의 설계 및 구현†

(Design and Implementation of Ethereum Smart Contract State Monitoring System)

홍 준 기 ‡ 김 순 태 § 류 덕 산 ¶
(Joongi Hong) (Suntae Kim) (Duksan Ryu)

요 약 블록체인 생태계에는 다양한 이해관계자들이 존재한다. 이더리움이 등장한 이후 스마트 컨트랙트를 활용한 거래가 많이 이루어지게 되었고 개발자뿐만 아니라 투자자, 은행, 기업, 일반 사용자 등등 더욱 폭넓은 이해관계자들이 참여하고 활동하고 있다. 하지만 다양한 이해관계자가 스마트 컨트랙트의 상태를 확인하기 위해서는 어렵고 복잡한 과정을 거쳐야 한다는 문제점을 가지고 있다. 상태 확인이 어렵게 된다면 스마트 컨트랙트에 대한 신뢰도가 낮아지게 되어 활용도가 낮아질 것이다. 개발자에게 있어서도 상태 확인이 어렵고 복잡하다면 자신이 개발한 스마트 컨트랙트의 테스팅과 디버깅을 하는데 어려움을 겪어 높은 품질을 제공하기 힘들 것이다. 본 연구에서는 다양한 이해관계자와 개발자들이 스마트 컨트랙트의 상태를 쉽고 지속적으로 확인할 수 있으며 히스토리 데이터를 활용하여 분석할 수 있도록하는 이더리움 스마트 컨트랙트 상태 모니터링 시스템의 설계 및 구현 방법을 제안한다.

키워드 : 이더리움, 스마트 컨트랙트, 상태, 이해관계자, 모니터링, 개발자

Abstract There are various stakeholders in the blockchain ecosystem. Since the emergence of Ethereum, many transactions have been made using smart contracts, and a wider range of stakeholders are participating, including not only developers, but also investors, banks, companies, and general users. However, various stakeholders have a problem in that it is difficult and complicated to check the state of smart contracts. If it becomes difficult to check the state, the reliability of the smart contract will be lowered and the utilization will be lowered. Also, if the state check is difficult and complicated for the developer, it will be difficult to provide high quality due to the difficulty of testing and debugging the smart contract developed by the developer. In this research, we propose a design and implementation method of the Ethereum Smart Contract State Monitoring System that enables various stakeholders and developers to easily and continuously check the state of smart contracts and analyze them using historical data.

Key words : Ethereum, Smart Contract, State, Stakeholder, Monitoring, Developer

1. 서 론

블록체인(Blockchain)은 2009년 가상화폐인 비트코인(Bitcoin)의 탄생과 함께 세상에 나타났다[1]. 2015년에는 단순히 가상화폐로의 기능으로만 활용되는 것을 넘어서 스마트 컨트랙트(Smart Contract)라는 개념을 도입한 이더리움(Ethereum)이 등장한다[2]. 이더리움은 스마트 컨트랙트를 기반으로 다양한 거래를 가능케 하였으며, 가상화폐를 기반으로 한 비즈니스가 가능하기에 개발자뿐만 아니라 투자자와 은행, 기업, 일반 사용자 등 다양한 이해관계자가 참여하고 활동하고 있다. 다양한 이해관계자가 참여하고 있는 프로젝트로는 대표적으로 DAO(Decentralized autonomous organization) 가 있다.

스마트 컨트랙트를 활용하여 거래하기 위해서 이해관계자들은 작성한 내용이 요구사항대로 처리가 되는지 확인하길 원할 것이며, 개발자 또한 자신이 작성한 로직이 설계한 대로 동작하는지 검증하고 싶어 할 것이다. 하지만 스마트 컨트랙트의 상태를 확인하기 위해서는 프로그래밍 지식이 필요하여 개발자 이외에는 어려울 수 있고, 복잡하고 번거로운 과정을 거쳐야 한다. 상태 확인을 위해서는 먼저

† 이 논문은 정부(정보통신기술진흥센터)의 재원으로 대학ICT연구센터육성지원사업의 지원을 받아 수행된 연구임(No. IITP-2018-2017-0-01628).
이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2019R1G1A1005047)
‡ 학생회원 : 전북대학교 소프트웨어공학과
rlwns012@jbnu.ac.kr
§ 종신회원 : 전북대학교 소프트웨어공학과 교수
stkim@jbnu.ac.kr
¶ 종신회원 : 전북대학교 소프트웨어공학과 교수
duksan.ryu@jbnu.ac.kr
논문접수 : 2019년 11월 4일
심사완료 : 2019년 12월 1일

스마트 컨트랙트 언어인 Solidity와 EVM(Ethereum Virtual Machine)의 메모리 구조를 이해하고 있어야 하며, 스마트 컨트랙트에서 선언한 상태의 타입과 순서에 따라 EVM의 메모리 구조에 맞게 값을 가져오고 변환하는 작업을 거쳐야 한다. 이와 같은 불편함은 스마트 컨트랙트의 상태에 대한 투명성을 낮춰 이해관계자들의 신뢰도를 떨어뜨리며, 개발자에게는 자신이 개발한 로직이 정상적으로 동작하는지 검증하기 어렵게 한다.

이더리움 네트워크의 스마트 컨트랙트에 대한 정보를 제공하는 방법으로 여러 모니터링 웹어플이케이션이 존재하고 있다. 가장 대표적으로 이더리움에서 자체적으로 제공하는 이더스캔(Etherescan)이 존재하며, 시각적인 형태로 정보를 제공하는 알비오(Alvio), Solidity IDE인 리믹스(Remix)도 존재한다. 하지만 스마트 컨트랙트의 상태를 상세하게 보여주는 기능을 하고 있지 않다.

개발자에게 스마트 컨트랙트에 대한 검증을 돕기 위한 연구로는 SmartInspect이 있다. 이더리움에 배포된 스마트 컨트랙트의 상태를 소스 코드의 변수명과 타입에 맞게 값을 변환해주어 소스 코드의 내용대로 파악할 수 있게 도움을 준다. 하지만 SmartInspect은 특정 한순간의 상태만을 제공해주기 때문에 지속적인 파악은 힘들다는 단점을 가지고 있다.

본 연구에서는 이해관계자들이 스마트 컨트랙트의 상태 확인이 어렵고 복잡하여 발생하는 문제를 해결하고자 블록 생성 시마다 자동으로 처리되고 소스 코드에서 작성한 내용대로 상태 확인이 가능한 모니터링 설계 및 구현 방법을 제안한다. 지속적인 상태 변화 파악을 위하여 모니터링 방법을 선택하였으며, SmartInspect[4]의 상태 변환 과정을 기반으로 하여 동작한다. 이해관계자들에게 스마트 컨트랙트의 검증과 확인을 위해서 과거의 상태 변화 히스토리를 남겨두고 이전 상태를 확인할 수 있도록 지원한다. 다시 말하자면 이해관계자들은 본 연구에서 제안하는 기법을 통해 별다른 과정 없이 스마트 컨트랙트의 상태 정보를 확인 가능하며, 지속해서 변화하는 상태 정보를 보고 요구사항에 맞는 로직이 수행되었는지 파악할 수 있도록 지원받는다.

이더리움 스마트 컨트랙트의 상태는 이해관계자들에게 올바른 거래 수행을 위해 매우 중요한 정보이다. 본 연구에서 제안하는 기법을 통해 중요한 상태 정보를 파악하기 쉽고 용이하게 하였으며, 순간적인 상태가 아닌 지속적으로 변화하는 상태를 보여줌으로써 진행한 거래를 바로 확인할 수 있게 하였다. 스마트 컨트랙트 상태의 히스토리를 저장하여 오류 또는 결함이 발생하였을 때 해당 문제에 추적이 가능하고 개발자에게는 코드의 디버깅 요소가 어떤 곳인지 확인할 수 있도록 하였다. 이는 현재 이더리움에 배포된 스마트 컨트랙트의 검증을 위한 도구가 부족한 상황에서 디버깅 도구로서의 역할을 수행할 수 있다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 배경 지식과 관련 연구에 대해 알아보고 3장에서는 스마트 컨트랙트 상태 확인을 위한 모니터링 기법에 관해 이야기한다. 4장에서는 결론 및 향후 연구에 관해 이야기하고 마무리한다.

2. 배경 지식 및 관련 연구

2.1 배경 지식

본 연구에서는 생소할 수 있는 블록체인과 스마트 컨트랙트에 대해 다룬다. 이해를 돕기 위해서 블록체인과 스마트 컨트랙트에 대한 개념을 간략하게 설명한다.

2.1.1 *블록체인(Blockchain):* 블록체인은 탈중앙화된 분산 시스템이다. 탈중앙화란 기능 수행을 위해 하나의 중앙기관이 존재하지 않고 다수의 참여자가 함께 수행해 나가는 것을 의미한다. 기본적으로 블록체인 네트워크는 P2P 네트워크로 구성이 되며, 합의 알고리즘을 활용하여 공통된 과정을 거쳐 블록을 생성한다. 해시의 활용은 블록체인에서 핵심적이며 해시와 함께 다수의 참여자가 같은 데이터를 통해 같은 처리를 한다는 점이 결합하여 데이터의 위변조를 방지한다.

블록체인에는 비트코인과 이더리움 등이 가장 대표적인 어플리케이션이며, 비트코인은 블록체인의 탄생을 알리고 현재 가장 인기 있는 가상화폐이다. 이더리움은 2015년에 등장하였으며, 스마트 컨트랙트의 적용으로 블록체인의 커다란 변화를 이끌었다. 위변조될 수 있는 스마트 컨트랙트를 블록체인에 배포함으로써 작성 내용을 무결하게 유지할 수 있게 되었으며, 블록체인의 기능 확장을 가능하게 했다.

블록체인은 많은 사람의 관심을 끌었으며, 특히 이더리움은 스마트 컨트랙트를 가능하게 하여, 단순하게 개발자들뿐만 아니라 은행, 투자자, 기업, 일반 사용자들이 참여하여 투자와 펀드 등 다양한 활동을 이어가고 있다. 다양한 이해관계자가 참여하는 대표적인 예로 DAO(Decentralized autonomous organization) 프로젝트가 존재하고 있다.

2.1.2 *스마트 컨트랙트(Smart Contract):* 현실의 계약을 프로그래밍 코드를 활용하여 작성한다는 개념이며, 조건이 충족되면 계약서에 있는 내용대로 자동으로 처리된다[3]. 하지만 디지털 문서 같은 경우 위변조가 쉬우므로 당시에는 활용되기에는 한계점이 존재하였다. 이더리움에서는 블록체인에 스마트 컨트랙트 배포하여 활용할 수 있는 개념을 제안하였고, 블록체인에 배포되어 저장된 데이터는 위변조가 어렵다는 점을 활용하여 한번 작성되어 배포되었다면 스마트 컨트랙트는 무결성을 보장받을 수 있다.

이더리움에서 작성되는 스마트 컨트랙트는 Solidity 언어를 활용하며, EVM 위에서 동작한다. Solidity는 Javascript와 Python 등에 영향을 받은 언어이며, 정수형 데이터를 취급하지 않고 Address와 mapping 등 독자적으로 제공하는 데이터 타입이 존재한다. EVM은 JVM과 비슷한 개념이며, 블록체인의 특성에 맞는 제약이 존재한다.

```
pragma solidity ^0.4.16;

contract TestContract {
    int64 age;
    int64 num;
    int64 num2;

    bytes32 bytedata1;
    bytes32 bytedata2;

    string name;
    address chairperson;
    uint256 time;

    function TestContract() public {
        chairperson = msg.sender;
        age = 100;
        num = 200;
        num2 = 300;

        bytedata1 = "a";
        bytedata2 = "b";
        bytedata1 = "c";
        name = "Hong";

        time = now;
    }
```

그림 1 Solidity를 활용한 TestContract 스마트 컨트랙트 예제 코드

[그림 1]은 Solidity 언어를 활용하여 작성된 TestContract 라는 스마트 컨트랙트의 예제 코드이다. 스마트 컨트랙트에서 상태란 전역변수로 선언되어 이더리움 상에서 지속적으로 유지되는 변수를 의미한다. 소스코드 확인해 볼 수 있듯이 int64 age, int64 num, byte32 bytedata1, string name 등이 전역으로 선언되어 있으며, 이 변수들을 상태로 정의한다. 해당 스마트 컨트랙트는 생성자를 활용하여 블록체인에 배포됨과 동시에 상태 값이 초기화가 되도록 작성해두었다. 해당 예제 코드에서는 상태를 파악하는 것에 초점을 맞추기 위해 상태 이외의 함수, 이벤트 등의 정보는 생략하였다.

2.2 관련 연구

이더리움에 대한 정보를 파악하고 개발자들의 테스팅과 디버깅을 위한 도구들을 알아보고 현재 진행되고 있는 연구에 대해 알아보도록 한다.

2.2.1. *이더스캔(Etherescan):* 이더리움은 블록의 생성, 트랜잭션의 발생, 스마트 컨트랙트 활용 정보 등 블록체인의 현황 파악을 위해 이더스캔(Etherescan)을 기본적으로 제공하고 있다. 이더스캔에서는 이해관계자들이 스마트 컨트랙트를 사용했다는 트랜잭션 정보를 확인해 볼 수 있다. 하지만 이해관계자에 의해 발생한 트랜잭션이 변화시킨 스마트 컨트랙트의 상태 변화에 대한 정보는 인코딩된 형태로 제공되어 이해하기 어려우며 전체의 상태를 파악할 수 없다.

2.2.2 *리믹스(Remix):* 리믹스는 이더리움 스마트 컨트랙트 개발을 위한 Web 기반의 IDE다. 리믹스에서는 개발뿐만 아니라 이미 배포된 스마트 컨트랙트의 정보도 확인해 볼 수 있다. 하지만 상태를 확인하기 위해서는 Solidity에서 상태가 public으로 선언되어 있어야 하거나 개발자가 get 함수를 만들어야만 확인할 수 있다. get 함수로 만들어 제공할 경우 사용자는 이더리움의 수수료 개념인 gas를 지불해야 하므로 지속해서 사용해야 한다면 적절하지 못한 방법일 것이다. 또한, OPCODE의 형태로 디버깅 도구로 제공하지만, 해당 지식을 이해하고 있지 않다면 이해하기 어려운 내용이다.

2.2.3 *SmartInspect:* SmartInspect는 개발자가 스마트 컨트랙트의 디버깅이 어렵다는 점을 지적하며 Solidity 소스 코드에서 선언한 상태 변수를 기반으로 하여 상태 값을 확인하는 방법을 제안하였다[4]. 이더리움 상에 스마트 컨트랙트의 상태 정보는 인코딩(Encoding)되어 저장된다. 따라서 실제 소스 코드와 받아온 데이터를 직접 연관 지어 볼 수 없으며, 데이터 변환과정을 거쳐야 최종적으로 사용자가 원하는 형태로 볼 수 있게 된다. *SmartInspect* 는 해당 과정 해결을 위해 mirror-based architecture를 설계하고 XML, JSON 등 파일의 형태로 만들어 쉽게 확인 가능한 것을 보여주었다. 하지만 *SmartInspect* 는 순간적인 상태만을 제공하며, 상태의 변화를 자동으로 파악하고 히스토리를 쌓는 기능이 없기에 잦은 변화가 있는 블록체인 생태계에는 적합하지 않은 형태이다.

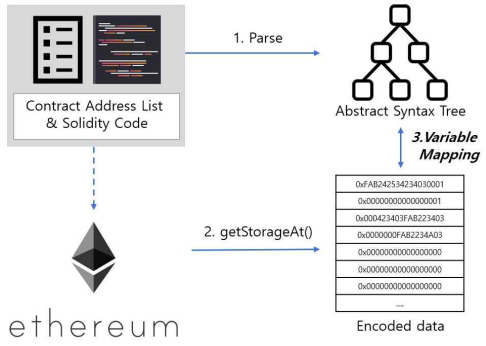

그림 2 SmartInspect에서 제안한 스마트 컨트랙트 상태 확인 방법

SmartInspect는 [그림 2]와 같은 형태로 소스 코드와 상태 값을 매핑한다. 사용자로부터 Solidity 소스 코드와 Address를 입력받고 소스 코드는 Parser를 활용하여 상태 변수를 추출하고 Address를 활용하여 이더리움으로부터 상태 값을 가져온다. EVM의 메모리 구조와 상태 변수의 타입 및 순서를 고려하여 매핑을 진행하고 실제 값을 매핑하는 방식이다. 해당 방법은 본 연구의 기반으로 활용된다.

3. 이더리움 스마트 컨트랙트 상태 모니터링 시스템의 설계 및 구현

이 장에서는 본 연구의 접근 방법인 이더리움 스마트 컨트랙트 상태 모니터링 시스템의 설계 및 구현에 관해 이야기한다. 상태에 대한 모니터링에 대한 이해를 돕기 위해 2.1.2 장에서 제시한 [그림 1]의 Solidity 스마트 컨트랙트 예제를 활용하여 설명을 이어나간다.

본 연구에서는 이해관계자와 개발자에게 상태 정보를 전달하기 위해 적합한 형태가 모니터링 시스템이라고 판단하였다. 이더리움에는 수많은 트랜잭션이 발생하고 트랜잭션에 의해 스마트 콘트랙트의 상태는 실시간으로 변화하고 있다. 지속해서 변화하는 데이터를 프로파일링하고 분석, 진단, 디버깅해야 하는데 모니터링 시스템은 이와 같은 기능을 수행하기 위해 발전해왔다 [5].

모니터링 시스템은 Observe와 Analyze로 두 개의 부분으로 구성된다. 하나는 소프트웨어에 대한 정보를 파악하고 해당 정보를 지속해서 관찰 기능을 수행하는 Observe 부분과 Observe에서 파악한 정보를 기반으로 도메인별로 필요한 분석을 진행하고 분석한 내용에 따라 대처를 가능케 하는 Analyze 부분이 있다. 본 연구에서는 Observe 부분에 초점을 두었다.

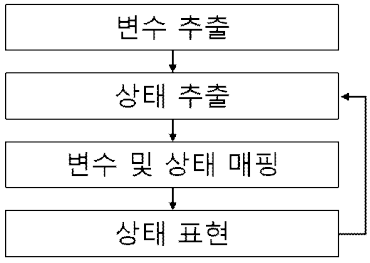

그림 3 모니터링 개요

[그림 3]에서는 모니터링을 위한 단계의 개요를 나타내고 있다. 단계는 변수 추출, 상태 추출, 변수 및 상태 매핑, 상태 표현 등 4단계로 구성이 되어있으며, 모니터링의 기본적인 구성인 Observe와 Analyze 중 Observe 부분에 해당하는 단계이다. 본 연구에서는 스마트 컨트랙트의 상태를 파악하기 위한 모니터링 기법에 관해 설명하고 있으며 블록체인에 배포된 코드는 변경이 불가능하므로 문제가 발생하여도 대처할 수 없다는 특성이 있다. 따라서 모니터링 중 프로파일링에 초점을 두어 연구를 진행하였다. 아래에서는 단계별로 수행되는 상세 내용을 서술한다.

첫 번째로 변수 추출 단계에서는 사용자로부터 Solidity source code와 Address를 입력받으며, 입력받은 Source code를 Antlr4[6]를 활용하여 만든 Solidity Parser로 파싱을 진행한다. 파싱을 통해 상태로 선언된 변수들을 가려내고 변수들의 타입과 식별자 이름을 JSON의 형태로 Token 화한다. [그림 4]는 Solidity Parser를 활용하여 만들어낸 Token JSON이다.

```
[{"type":"int64","name":"age","size":64},
{"type":"int64","name":"num","size":64},
{"type":"int64","name":"num2","size":64},
{"type":"bytes32","name":"bytedata1","size":32},
{"type":"bytes32","name":"bytedata2","size":32},
{"type":"string","name":"name","size":256},
{"type":"address","name":"chairperson","size":160},
{"type":"uint256 ","name":"time","size":256}]
```

그림 4 Variable Token JSON

Token에는 상태의 타입, 식별자 이름, 타입의 크기로 구성이 된다. [그림 1]의 예제에서 선언된 age, num, num2 등등의 상태들이 Token JSON의 형태로 변환된 것을 확인할 수 있다. Token JSON은 스마트 컨트랙트별로 단 하나만이 생성되며, 등록과 동시에 DB에 Source code와 Address, Token JSON이 함께 저장된다. 데이터는 블록이 생성될 때마다 다시 불러와 다음 과정을 수행하기 위해 저장되어 관리된다.

두 번째로 상태 추출 단계를 진행한다. 이더리움 네트워크로부터 Web3의 getStorageAt() 함수를 활용하여 스마트 컨트랙트의 상태를 추출한다. EVM의 메모리 구조에 맞춰 저장된 16진수의 값을 얻어 오는 과정이다. [그림 5]와 같이 이더리움으로부터 인코딩된 데이터(Encoded data)를 불러오는 과정이다. 이렇게 추출된 16진수의 상태 정보는 사용자들에게 당장 쓸 수 있는 형태로 제공되는 것이 아니며 값을 파악하기 어렵다. 또한, 상태의 정확한 값을 알기 위해서는 상태의 순서와 타입에 맞게 변환하는 과정이 필요하다.

그림 5 Web3를 활용하여 이더리움 네트워크에 배포된 스마트 컨트랙트의 상태를 추출한다.

세 번째 과정은 변수 및 상태 매핑 작업이다. 이더리움 네트워크에서 가져온 인코딩된 데이터는 Source code에서 선언한 변수명과 1:1로 연결되지 못한다. [5] 에서 제안하는 방법을 활용하여 [그림 7]와 같이 실제 값과 선언된 상태가 연결될 수 있도록 매핑하는 작업을 진행한다. EVM은 여러 개의 Slot을 가지고 있으며, 한 개의 Slot은 256bit로 구성된다. 데이터 타입별로 사이즈가 다르며, 사이즈에 맞춰 어떤 상태가 어디에 저장되어 있는지를 가려내야 한다. 또한, 모든 변수가 사이즈에 맞춰 저장되는 것이 아니므로 이에 따른 처리를 진행하여야 한다. 예제의 int64 age, int64 num, int64 num2 등등이 메모리

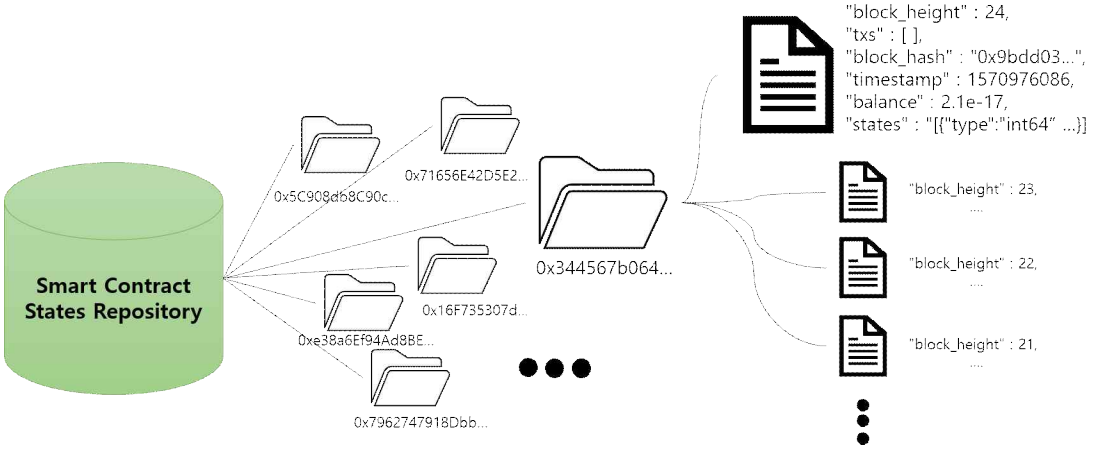

그림 6 스마트 컨트랙트별 상태 저장 방법

구조에 맞게 매핑이 진행된다.

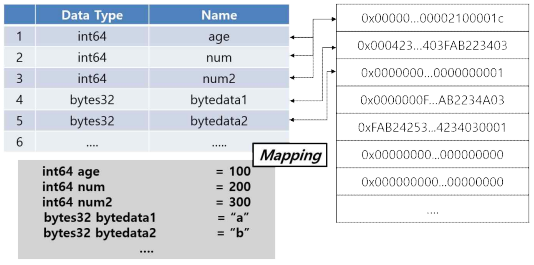

그림 7 EVM 메모리는 한 Slot이 256bit로 구성이 되어있으며, 여러 개의 Slot으로 구성되어 있다. 상태는 타입별로 다양한 크기를 가지고 있으며 Slot에는 크기에 맞게 여러 개의 상태가 들어갈 수 있게 설계되어 있다.

마지막으로 사용자에게 스마트 컨트랙트의 상태를 출력하는 단계이다. 출력되는 정보는 앞의 과정을 거쳐 만들어낸 매핑된 상태 정보와 함께 현재 생성된 블록의 블록 번호, 스마트 컨트랙트를 활용한 트랜잭션, 보유 이더량을 추가하여 하나의 데이터로 구성한다. 사용자에게 해당 상태 정보는 과거의 정보도 함께 보여주게 된다. 과거의 정보는 Smart Contract States Repository에 스마트 컨트랙트 별로 관리된다. 상태 정보를 출력한 후 블록체인으로부터 모니터링을 위한 프로토타입은 블록이 생성될 때까지 대기한다. [그림 2]에서 확인할 수 있듯이 블록이 생성될 경우 두 번째 과정인 상태 추출과정부터 다시 진행한다. 이 과정은 한 번으로 끝나지 않고 이더리움 네트워크에 블록이 쌓이는 한 무한히 반복하며 스마트 컨트랙트의 상태 변화의 지속적인 모니터링을 제공한다.

블록이 생성될 때마다 스마트 컨트랙트 별로 상태 정보를 DB에 저장하게 된다. 데이터는 [그림 6]와 같은 구조로 저장되고 사용자에 의해 스마트 컨트랙트가 등록된 순간부터 저장한다. 해당 저장소는 Solidity source code와는 별개의 저장소이며, 스마트 컨트랙트 하나당 블록이 쌓인 수 만큼 상태를 저장하게 된다. 상태 데이터 구조에 관해 설명하자면 우선 상태는 블록이 생성될 때마다 모니터링이 되기 때문에 구분하기 위한 기준으로 블록의 번호를 선정하였다. 더불어 블록의 해시를 함께 저장한다. 트랜잭션의 경우 블록에 담겨 있던 모든 트랜잭션을 저장하는 것이 아닌 스마트 컨트랙트를 활용한 트랜잭션만을 저장하게 된다. 저장되는 내용은 트랜잭션의 해시값이 들어가게 된다. 스마트 컨트랙트의 상태 중 하나인 이더량도 함께 저장하며, 모니터링 시간을 추가하여 구성하였다. 마지막으로 앞의 작업을 수행하여 얻어낸 매핑된 상태를 JSON의 형태로 담아 저장한다. 상태 정보에는 Source code에 선언된 모든 상태가 표현되며 실제 값이 어떤 값인지도 볼 수 있다.

[그림 8]는 최종적으로 사용자가 확인 가능한 텍스트 기반의 스마트 컨트랙트 상태이다. 상태의 데이터 구조는 앞에서 언급했듯이 이더량, 블록 해시, 블록 번호, 상태 정보, 스마트 컨트랙트와 관련된 트랜잭션 해시, 타임스탬프로 구성이 되어있는 것을 확인할 수 있다. 또한, 상태 정보를 들여다보면 Source code에서 상태로 선언되었던 변수의 이름과 타입을 확인할 수 있고 블록이 생성된 시점에 유지하고 있는 실제값을 realValue에서 확인할 수 있다.

그림 8 스마트 컨트랙트 상태 확인 텍스트 기반 모니터링 기법 출력 결과물

4. 결론 및 향후 연구

이더리움 생태계에는 다양한 이해관계자가 존재하며, 스마트 컨트랙트를 활용하여 다양한 계약과 거래를 진행한다. 하지만 모든 이해관계자가 스마트 컨트랙트가 요구사항대로 동작하고 있는지 파악하기 위해서는 프로그래밍 지식이 필요하며, 데이터 변환을 위한 복잡한 과정을 거쳐야 한다. 본 연구에서는 이더리움의 스마트 컨트랙트에 대한 이해가 부족하더라도 현재 상태가 어떤 값이고 어떻게 변화하고 있는지 파악할 수 있는 모니터링 시스템 설계 및 구현 방법에 관해 이야기하였다. 더불어 실제로 데이터가 어떻게 보이는지에 대한 텍스트 형태의 예시를 보여주었다. 본 연구를 통해 우리는 이해관계자들의 스마트 컨트랙트에 대한 이해도가 증가할 것으로 기대하고 있으며, 상태에 대한 투명성이 증가하여 당사자 간에 신뢰를 높여 계약을 진행할 수 있게 지원 도구로서의 역할을 할 수 있으리라 기대하고 있다.

향후 계획으로는 모니터링 시스템에서 데이터에 대한 표현을 텍스트로만 보여주는 것을 그래프와 타임라인 등을 활용하여 사용자들이 더욱 쉽게 인식할 수 있는 연구를 진행할 것이다. 또한, 다양한 이해관계자들은 거버넌스를 형성할 수 있고 거버넌스에 필요한 다양한 정보를 제공할 수 있는 모니터링 기능을 연구하고 확장할 계획이다. 관찰된 데이터를 분석하여 악의적인 사용과 이상 현상이 발생할 수 있는 점을 경고하고 파악하기 위한 기능을 연구할 예정이다.

참 고 문 헌

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system.” 2009. [Online]. Available: bitcoin.org
[2] Buterin, Vitalik. "Ethereum white paper." GitHub repository (2013): 22-23.
[3] Szabo, Nick. "Smart contracts." Unpublished manuscript (1994).
[4] Bragagnolo, Santiago, et al. "SmartInspect: solidity smart contract inspector." 2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE). IEEE, 2018.
[5] Delgado, Nelly, Ann Q. Gates, and Steve Roach. "A taxonomy and catalog of runtime software-fault monitoring tools." IEEE Transactions on software Engineering 30.12 (2004): 859-872..
[6] Parr, Terence. The definitive ANTLR 4 reference. Pragmatic Bookshelf, 2013.

홍 준 기
2019년 전북대학교 소프트웨어공학과 졸업(학사) 2019년~현재 전북대학교 석사재학

김 순 태
2007년 서강대학교 컴퓨터공학과 졸업(석사) 2010년 서강대학교 컴퓨터공학과 졸업(박사) 2014년~현재 전북대학교 소프트웨어공학과 부교수

류 덕 산
2012년 카이스트 및 카네기멜론대학교 소프트웨어공학 복수학위 졸업(석사). 2016년 카이스트 전산학부 졸업(박사). 2018년~현재 전북대학교 소프트웨어공학과 조교수