Global Journal of Computer Science and Technology: C
Software & Data Engineering
Volume 18 Issue 4 Version 1.0 Year 2018
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

# Application of Ethereum Smart Contracts in Purpose of Generating New Cryptocurrencies

By Jovan Toroman

*Abstract-* As Bitcoin's popularity increases so does the familiarity with cryptocurrencies in general. Ethereum is one of the most popular platforms in the cryptocurrency world. Development of a cryptocurrency using programming language Solidity and Ethereum platform is presented in this paper. Following the development, possible utilization of this cryptocurrency will be discussed.

*Keywords:* *crypto currency, security, systemstructure, ethereum.*

*GJCST-C Classification:* *E.m*


*Strictly as per the compliance and regulations of:*

# Application of Ethereum Smart Contracts in Purpose of Generating New Cryptocurrencies

Jovan Toroman

*Abstract-* As Bitcoin's popularity increases so does the familiarity with cryptocurrencies in general. Ethereum is one of the most popular platforms in the cryptocurrency world. Development of a cryptocurrency using programming language Solidity and Ethereum platform is presented in this paper. Following the development, possible utilization of this cryptocurrency will be discussed.

*Keywords: crypto currency, security, systemstructure, ethereum.*

## I. Introduction

Modern human civilization is based on the trade process. By the means of trading, people from the original communities managed to obtain the necessary resources they could not produce or procure in any other way, while at the same time exchanging their products and resources for other resources [1]. Later, currencies were introduced, so that they could be traded more easily, and exchanged for various necessary resources.

With the emergence of currencies, the first centralized entities that take control and distribution of currencies also appear, thereby enabling or disabling participants in the free market [2]. In this way, the absolute ownership of money belongs to small groups of people, which are mostly its richest owners who determinerules of the game and participation in systems where certain currencies are used.

One solution to this problem of lack of freedom in trade mediation, as well as the control of total capital by a third entity was found in cryptocurrencies. Cryptocurrencies as a concept allow for creation of the representation of material goods by relying on the cryptographic integrity of use, regardless of the field of application or usage of money. With their use, it is possible to have a distributed and decentralized system of trade with money that has an equal value among all the entities participating in the exchange.

In this paper, Chapter 2 gives an overview of the cryptocurrencies. Chapter 3 describes the Ethereum platform. Chapter 4 explains the notion of smart contracts, while in Chapter 5 we deal with the creation of cryptocurrencies using the Ethereum platform. Chapter 6 gives the conclusions of the work outlined in the previous chapters.

*Author: Fakultet Organizacionih Nauka. e-mail: toromanj@gmail.com*

## II. Cryptocurrencies

Cryptocurrencies were designed with the same purpose as the standard currencies - to enable people to exchange goods more easily. However, what distinguishes the existing cryptocurrencies from the standard currencies is the way they are issued, as well as how financial transactions are performed. Instead of the standard situation where the central bank issues and affects the value of money, with cryptocurrencies this process is completely decentralized. The way in which this is done is precisely the usage of cryptographic methods.

Cryptocurrencies as a concept tookthe attention of the publicin January 2009, with the emergence of Bitcoin - the first cryptocurrency that is at the same time a decentralized payment system [3]. Designed by, until that moment unknown, Satoshi Nakamoto, the main purpose of Bitcoin was to create a fully decentralized currency, where mediation of financial institutions is not necessary for its exchange [4].

Given that there is no centralized financial institution, such as a central bank or money mint, how are cryptocurrencies created and how is their value determined? In the case of Bitcoin, like the vast majority of other cryptocurrencies, the mining process creates cryptocurrencies that are then distributed to "miners" by a particular algorithm, most often using the Proof of Work algorithm [5]. Mining represents the process of recording and confirming transactions within the decentralized network into the chain of blocks-blockchain. Blockchain represents a distributed database representing the history of all recorded transactions made using Bitcoin [6]. The constituent element of the blockchain is a block - the currently active part of a database that records all or most of the recent transactions [7]. Each block contains a reference to the previous block in the chain, all the way to the initial block - genesis block. In the case of Bitcoin, as a reference cryptocurrency, each block also contains a "puzzle" that must be solved so that the block can be marked as completed. It is in the form of conditions that a cryptographic hash function must satisfy for a certain value. The first value that permits the fulfillment of the puzzle is the solution of the block and the computer (or groups of computers) that participated in its calculation is rewarded for the spent computer resources. The award is presented in the form of Bitcoin, which is then

distributed among the participants in mining. Initially, 50 Bitcoins were awarded for solving each block, but this amount is reduced after every 210,000 blocks resolved so that Bitcoin's inflow would be proportional to their value and the difficulty of puzzles added to each new block [8].

## III. Ethereum

Ethereum is a distributed open source platform that uses blockchain to enable usage of smart contracts. By applying virtualization techniques, it is possible to execute various commands like on a distributed nodal network. Ether denomination is used as the propelling agent of these commands (transactions). In this way, it prevents accumulation of irrational tasks in the nodes of the network.

Due to the collapse of The DAO organization, the Ethereum has collapsed into two chains. The chain with a smaller number of users got the new name Ethereum Classic, while the popular bridle retained the old name. This happened because of a security breach in one of the applications, where attackers managed to return the Ether over recursive calls several times before updating the balance of the smart contract itself. Unfortunately, it turned out that it is not possible to easily undo new transactions. Due to disagreement over whether the ether should be returned to the original owners, at the expense of some of the basic principles of functioning of the platform, or other option-continuation without changes, the aforementioned division occurred.

## IV. Smart Contracts

Smart contracts are one of the features of the cryptocurrencies and the decentralized payment system Ethereum. They allow that on a particular Ethereum address, there is a computer program along with it's associated data (its state) which enables automatic cryptocurrency management [9]. Changes made by calling this program are recorded and registered in Ethereum blockchain. Smart deals are presented with Turing's complete machine, but there are opinion currents which suggest that it is not the best solution in the case of smart contracts [10].

Each Ethereum smart contract is executed on the Ethereum Virtual Machine (EVM), so there is a degree of isolation of smart contracts from the operating system where it is executed. In addition, there is a degree of isolation between other smart contracts that are executed on the same EVM instance. The Ethereum address of a smart contract, unlike the Ethereum user address that represents its public key, is determined using the public key of the creator, the time when the contract was created and the number of transactions that the owner of the smart contract has done [11]. Based on this, we can conclude that smart contracts have been made by their creators and owners.

When performing each transaction using smart contracts, each transaction is charged by a certain amount of "gas" resource - which at the time of writing this work is 0.00000002 ether or 22312210827 wei. The gas is a value derived from the ethereum used to charge the use of the Ethereum Transaction Processing Network and is set by the transaction maker so that it is possible that persons who mine transactions refuse to execute transaction due to an unfavorable gas price [12].

Ethereum smart contracts are written using the Solidity programming language that supports concepts from object-oriented languages, such as inheritance and static typing [13]. Because of the possibilities it provides and because it is based on already existing programming languages, Solidity allows you to create a large number of different smart contracts.

```
pragma solidity ^0.4.0;
contract SimpleStorage
    {uint storedData;
    function set(uintx) {
      storedData = x;
   }
function get() constant returns (uint) {
      return storedData;
  }
}
```

*Figure 1:* An example of a smart contract that allows you to enter and read the stored value at the smart contract address.

As it is possible to observe from the above example, there are minimal barriers to the development of smart contracts using the Solidity programming language, judging by the fact that it is based on the syntax of different programming languages, such as C and Java.

## V. Usage of Solidity and Ethereum Platform for Creating a Cryptocurrency

As we defined in the introduction, cryptocurrencies as a concept allow for creation of representations of material goods, relying on the cryptographic integrity of use, independentfrom the field of application or usage of money.

The cryptocurrency we will create will be different in contrast to Bitcoin or Ethereum. Namely, we will have an administrator of the cryptocurrency who will represent it's owner, who can also create new instances of cryptocurrencies. The crypto administrator will be a

person with a specific Ethereum address. It is possible that the crypto administrator is an address belonging to a smart contract that can automatically, if it is so constructed, create and cancel the amount of cryptocurrency in relation to another external factor (such as aligning with a course of another currency or even the movement of another cryptocurrency). In addition, it is important to note that, for the purposes of the case, we will not use Proof of Work [14], Proof of Stake [15] or Proof of Burn [16].

Launching a smart contract, and therefore cryptocurrency, is done using Ethereum Wallet [17]. To begin the development of cryptocurrency, it is necessary to have a minimal set of functionalities: the collection of all addresses with the amounts of our cryptocurrency, the initial amount of cryptocurrency, the address where the initial amount of cryptocurrency is located (in our case crypto administrator), and a function that allows the transfer of cryptocurrencies between different addresses (which may belong to other people or smart contracts, and are linked to the Ethereum network).

```
contract MyToken {
   /* Creating an array with all
addresses and balances */
   mapping (address => uint256)
   public balanceOf;
  /* Initializes a contract with the
 initial amount of cryptocurrency allocated to the creator
of smart contract */
   function MyToken(
    uint256 initialSupply
    ) {
   balanceOf[msg.sender] =
initialSupply;
//   Give the creator all the
//   units of the cryptocurrency
    }
/*Cryptocurrency transfer function*/
function transfer(address _to,
uint256 _value) {
       if (balanceOf[msg.sender] < _value)
throw;
//   Check that the sender has enough
//   cryptocurrency units
     if (balanceOf[_to] + _value
<balanceOf[_to]) throw;
//   Checking the quantity overflow
//   balanceOf[msg.sender] -= _value;
//   Subtract from the sender
        balanceOf[_to] += _value;
//   Add to recipient
     }
}
```

*Figure 2:* Example code for the smallest functional smart agreement that creates a new cryptocurrency

With the example code above, after loading it as a smart contract within Ethereum Wallet, we have the ability to distribute cryptocurrencies to other interested parties. As already stated, in order for the smart contract to become active and in order for it to be correct, it is necessary to spend a certain amount of gas, which represents a kind of fee paid to the Miners of the Ethereum network in order to confirm the creation of smart contract and transactions belonging to it.

Adding cryptocurrency administrators is done by inheriting a smart contract that defines the behavior of the cryptocurrency administrator. A description of the behavior that needs to be implemented in a smart contract for cryptocurrency administrator is contained within the Solidity documentation [18].

```
contract owned {
   address public owner;
   /* the cryptocurrency owner's address which was set
when originaly setting up the smart contract */
  /* function to check
cryptocurrency ownership */
   function owned() { owner = msg.sender;
}
/* checking if there is only one
 owner */
    modifier onlyOwner {
       if (msg.sender != owner) throw;
        _;
}
/* function to transfer
cryptocurrency ownership*/
    function transferOwnership(address
newOwner) onlyOwner {
      owner = newOwner;
   }
}
```

*Figure 3:* Example of a contract used to define the cryptocurrency administrator

The inheritance of the contract is done using a keyword "is" in the definition of a contract from which we

want to inherit it, so in our case, the definition of the MyToken contract is as follows:

```
contract MyToken is owned
```

Managing the available amount of cryptocurrency is possible by adding one variable and one function. The variable monitors the total number of available amount of cryptocurrency, while the function modifies the variable, thus altering the total amount of available cryptocurrency, taken that only the cryptocurrency administrator has the ability to change the cryptocurrency in circulation.

It is necessary to define the totalSupply variable that will represent the total available amount of cryptocurrency in circulation:

```
uint256 public totalSupply;
```

It is also necessary to add the starting amount of available cryptocurrency to the smart contract constructor, so that, when creating a contract, we would have an insight into how many cryptocurreny units are available.

```
totalSupply = initialSupply;
```

The function that deals with the change in the available amount of cryptocurrency in circulation is as follows:

```
function mintToken(address target, uint256 mintedAmount) onlyOwner {
    balanceOf[target] += mintedAmount;
    totalSupply += mintedAmount;
    Transfer(0, owner, mintedAmount);
    Transfer(owner, target, mintedAmount);
}
```

This function uses the previously defined Transfer function to perform the transmission of the modified circulation of cryptocurrency to the cryptocurrency administrator address.

## VI. Conclusion

The application of the concepts outlined in this paper opens numerousinteresting possibilities. Creating different cryptocurrencies is simple with this kind of infrastructure, and on the otherside, there is great freedom in terms of the amount of currency in circulation as well as the way transactions are performed.

Such a concept is applicable in different scenarios.One of the most interesting is the game industry.In fact, computer games are one of the first examples of virtual currencies. It is often the case thata game allows you to purchase different items and functionalities. A virtual currency is a simple way of transferring value from one game to another, as well as it's transformation into other goods. This is especially interesting for smaller development teams that can have significant benefits by merging into an ecosystem with other teams using a common cryptocurrency.

## References Références Referencias


1. G. Davies, A History of Money from Ancient Times to the Present Day, rev. ed. Cardiff: University of Wales Press, 1996. 716p. ISBN 0 7083 1351 5.
2. C. Vantieghem, Monetary practices in Ancient Egypt, Available at http://www.nbbmuseum.be/en/2012/05/nederlands-geldgebruik-in-het-oude-egypte.htm
3. J. Davis, The Crypto-Currency, 2011, Available at http://www.newyorker.com/magazine/2011/10/10/the-crypto-currency
4. S. Nakamoto, Bitcoin - A Peer-to-Peer Electronic Cash System, available at https://bitcoin.org/bitcoin.pdf
5. PProof of work description, available at https://en.bitcoin.it/wiki/Proof_of_work
6. Blockchain description, http://www.investopedia.com/terms/b/blockchain.asp
7. Solidity transactions description, available at http://solidity.readthedocs.io/en/develop/introduction-to-smart-contracts.html#transactions
8. Bitcoin block description, available at https://en.bitcoin.it/wiki/Block
9. Smart contracts definition, available at http://solidity.readthedocs.io/en/develop/introduction-to-smart-contracts.html
10. Turing smart contracts description, available at http://www.coindesk.com/turing-complete-smart-contracts/
11. Official Solidity documentation - Intorduction to smart contracts, available at http://solidity.readthedocs.io/en/develop/introduction-to-smart-contracts.html#accounts
12. Ethereum gas description, available at https://www.cryptocompare.com/coins/guides/what-is-the-gas-in-ethereum/
13. Solidity, official documentation, available at https://solidity.readthedocs.io/en/develop/
14. Bitcoin - proof of work description, available at https://en.bitcoin.it/wiki/Proof_of_work
15. Bitcoin – proof of stake description, available at https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ
16. Bitcoin – proof of burn description, available at https://en.bitcoin.it/wiki/Proof_of_burn
17. Ethereum implementation, available at https://github.com/ethereum/mist
18. Solidity implementation, available at https://solidity.readthedocs.io/en/develop/