Journal of Convergence Technologies and Smart Application Vol.1, No.1 (2020), pp.15-22
http//dx.doi.org/10.46410/jctsa.2020.1.1.03
Received: June 22, 2020 Reviewed: August 9, 2020 Accepted: September 4, 2020

A Study on Building a Blockchain Development Environment

Jin-whan Kim[1]

[1] School of Computer Engineering, Youngsan University, 288 Junam-ro, Yangsan city, Gyeongnam 50510, Republic of Korea; kjw@ysu.ac.kr

Abstract: Blockchain technology is emerging as a key infrastructure technology that will lead the fourth industrial revolution, and blockchain promises to bring about revolutionary growth in almost all fields. It is also expected to be a very important infrastructure technology that can spur innovation in many fields of human society such as politics, economy, society, and culture. This paper provides an overview of the development environment for developing and distributing the blockchain platform. The paper also covers main terms and related tools for decentralized applications (DApp) of blockchain technology, which have been recently gaining much attention.

Keywords: *Blockchain, IDE, Development Environment Tool, Bitcoin, Ethereum, Smart Contract*

1. Introduction

Blockchain technology stores data blocks to be managed on a distributed peer-to-peer (P2P) database. Chains are formed between these blocks so that they cannot be arbitrarily modified. Blockchain is a distributed computing-based trust network technology where any changes can be viewed. In other words, blockchain is a distributed ledger technology (DLT) in which transaction information is recorded on the computers of the participants' peer-to-peer (P2P) network rather than on the centralized servers of an institution. The main characteristics of blockchain are distribution, immutability, reliability through an agreement process in an untrusted environment, security, economics, speed, decentralization, disintermediation, safety, transparency, efficiency, and scalability [1,2,3].

In this paper, we will look into the development environment and related tools for developing DApps on a blockchain platform. Currently, there are various development environments and methods, so the experience may confuse newcomers to the field. Accordingly, we intend to contribute to the further development of the blockchain industry and its ecosystem by organizing key specifics.

2. Blockchain's Terminology

The blockchain platform allows contracts and transactions to be made even when the associated parties do not know each other. Blockchain enables all types of transactions that handle important business logic (e.g. contracts, information records, transactions, asset transfers by verifying mutual identities and establishing trust without relying on brokers (e.g. banks, credit rating agencies, governments). In a blockchain, the blocks that include the transaction history are stored and managed in the form of a chain in which all past transactions are linked together with cryptographic technology [4,5].

2.1. Main Terms

2.1.1. P2P (Peer to Peer) Network

A P2P network is a distributed computer network. This method employs a large number of computers that communicate directly with each other on a one-to-one basis. Here key information is

distributed, stored, and managed on all computers connected through the network. Thus, a P2P network is a system with high reliability because, if even several computers are disconnected during a hacking attack, the entire system is not affected.

2.1.2. Hash Function Encryption

A hash value is a technique that always generates the same hash value when the input data is the same. Hash values are generated by a 'hash function' that maps to a fixed-size data for arbitrary data and its length. It is a key technology used to detect breakage and manipulation of data at a very high speed.

2.1.3. Smart Contract and Chain Code

A smart contract is a term used in Ethereum. The contents previously negotiated on by certain parties are registered as a programmed electronic contract, and the contents of the contract are automatically executed when the conditions of the contract are met. In other words, a smart contract is a method to automatically process a specific transaction according to an agreed-upon method. Smart contracts are developed using Solidity or Viper programming language, but in Hyperledger blockchains, “smart contracts” are called “chain codes” and are developed using Go, Java, or Javascript languages.

2.1.4. Mainnet and Testnet

Mainnet is the term used to describe a fully developed and deployed blockchain protocol. This means that cryptocurrency transactions are being broadcasted, verified, and recorded on a distributed ledger technology.

In contrast to Mainnet networks, the term Testnet describes the blockchain protocol or network that is not operating at full capacity. Testnet is used by programmers and developers to test and troubleshoot all the aspects and features of a blockchain network before certifying that the system is secure and ready for the Mainnet launch. Typical Ethereum-based Testnets include Ropsten and Kovan.

2.1.5. Faucet

On Testnet, Faucet is used to issue the cryptocurrency Ether for testing purposes at no cost. Ether in Testnet is only used for development and testing purposes, so it has no inherent value and is not handled by any exchange.

2.1.6. Coins and Tokens

The cryptocurrency issued by the blockchain system on Mainnet is called a coin. In contrast, tokens appear when the Mainnet’s blockchain system is borrowed and a separate cryptocurrency is issued. In general, when token use becomes active, a separate Mainnet is created and promoted to issuing coins.

2.1.7. Consensus Algorithm

The process by which all nodes in a distributed system determine the same result is called consensus. This is an algorithm that determines the ownership and the creation order of blocks to prevent errors and ensure integrity in the blockchain system. A consensus algorithm is a method for correctly forming consensus among a large number of unspecified participants. The types of consensus algorithms include Proof of Work (PoW), Proof of Stake (PoS), Delegated Proof of Stake (DPoS), Proof of Importance (PoI), and PBFT (Practical Byzantine Fault Tolerance), to name a few.

2.1.8. Fork

Forks occur when a group of nodes in a network adopts a different set of consensus rules. A soft fork occurs when the new rule is a subset of the old rule. In this case, the task of updating to the new rule is the role of the miner. A hard fork, however, occurs when the new rule is not a subset of the existing rule. In this case, all nodes must update, but the rejecting node branches from the updating node. Ethereum Classic was created in this way.

2.1.9. Gas

Gas is a network fee paid when running an application on Ethereum.

To guarantee the operating environment for a complete Turing application on a blockchain system, it is necessary to prevent the execution of malicious programs (e.g., infinite repetitive programs). Gas is used to prevent these problems by charging numerous fees that hinder these malicious programs.

2.1.10. ERC-20 Token

ERC-20 is a standard token specification established by the Ethereum blockchain network. ERC stands for Ethereum Request for Comments, and the number 20 means the 20th proposal of EIPs, which is the Ethereum Improvement Proposals. Usage of the token name, symbol, total supply and remittance, balance, and other tasks as specified in detail on the website below.
https://eips.ethereum.org/EIPS/eip-20

2.1.11. DApp (Decentralization Application)

This is an abbreviation of decentralized applications. A DApp executes a smart contract while operating continuously without a specific administrator. For this reason, it is also called an autonomous distributed application.

2.1.12. Oracle Problems

Oracle is the name given to a problem that occurs when external data is brought into the blockchain. When developing applications, it is sometimes necessary to use various application programming interfaces (APIs). In general, smart contracts on a blockchain cannot obtain external information to the blockchain. The act of solving this issue is known as the Oracle problem. It is impossible to access any external information source to inquire about any information in the smart contract. However, this method is a means to bypass the issue, and it enables the transfer of information requested by the smart contract from a trusted agent or data provider.

Service providers such as BlockoneIQ, Chainlink, Link, Blocksense, and Oraclize provide Oracle services.

- BlockOneIQ (https://blockoneiq.thomsonreuters.com)
- Chainlink (http://www.initc3.org)

2.1.13. OpenZeppelin

Written in Solidity, this is a library of modular, reusable, and secure smart contracts for the Ethereum network. This library provides templates for developing ERC20-compatible tokens, ICOs, and ownership, among other things. The site below provides detailed information about OpenZeppelin.

- https://openzeppelin.com

3. Building a Blockchain Development Environment

In this section, we will examine the development environment for Bitcoin and Ethereum which are widely known as blockchain platforms.

3.1. Bitcoin

The Bitcoin Core program can be installed by accessing the following website:
https://bitcoin.org/en/download

Software versions are available for multiple operating systems, including MS Windows, Mac OS, and Ubuntu (Linux). After modifying the configuration file of bitcoin.conf, individuals can run bitcoind.exe or bitcoin-qt.exe to start the server program. By executing the client program bitcoin-cli.exe, the program can perform tasks such as querying and analyzing blockchain data [6,7]. Additional information can be found through the sites below.

- Bitcoin developer website: https://bitcoin.org/en/development
- Starter project: https://bitcoin.org/en/development#start-project
- Developer community: https://bitcoin.org/en/development#dev-communities
- Bitcoin source code: https://github.com/bitcoin/bitcoin.git

Bitcoin uses the SHA-256 hashing algorithm in the proof-of-work (PoW) consensus algorithm. The miner uses the nonce and timestamp fields to generate new input and hash the block header repeatedly with SHA-256. A miner who succeeds in generating a targeted hashing value according to the mining difficulty then creates a block and propagates it to the blockchain network. The miner will include a coin-based transaction in the block and receives a new bitcoin in reward for his hash power. To improve the inefficiency (e.g. high-performance computer, ASIC chip, GPU, excessive power consumption) of the proof-of-work algorithm (PoW) used by Bitcoin, several modified cryptocurrencies were designed, and hundreds of cryptocurrencies such as Litecoin and Monero were also developed.

3.2. Ethereum

Ethereum is a distributed computing platform for implementing smart contract functions based on blockchain technology. It can be developed using Geth (Go-ethereum), an Ethereum client developed in the Go language. The Ethereum node can perform network synchronization in three nodes: Light, Full, and Archive (or Super) node.

Light nodes synchronize block headers but do not process transactions or maintain state trees. Light clients are useful for users who only want to send and receive ether by using only their digital wallets. From the developer's point of view, a full node is required because the light client alone is not sufficient to satisfy the development purpose. The full node maintains a local snapshot of the blockchain state tree, downloads the entire block, executes block transactions on the blockchain local copy, and participates in the consensus process. The full node is the backbone of the Ethereum network and provides network information to other peers, but a light node only fetches information from the network. The archive node also called a full archive or supernode, maintains a copy of all state transitions that have occurred on the blockchain as well as the current snapshot of the state tree since the genesis block [8,9,10,11].

3.2.1. Geth

Geth is widely used as a tool for developing Ethereum client applications, and it can perform most of the tasks required in Ethereum such as smart contract creation and execution, Ether transfer, account creation, and mining.

Interested parties may download the installation files for operating systems such as Linux, MacOS, and Windows from the following site. Figure 1 shows a screen of the Geth download site.
Geth download site: https://geth.ethereum.org/downloads/
Geth document: https://github.com/ethereum/go-ethereum/wiki/geth

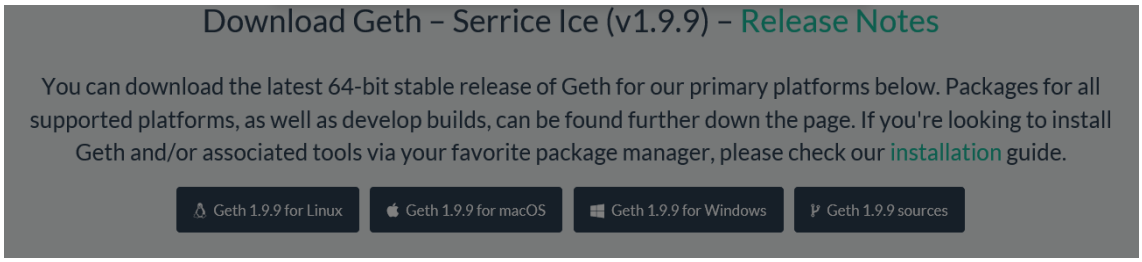

Figure 1. Geth download

3.2.2. Metamask

Metamask is an electronic wallet for cryptocurrency and an extension app designed as a plugin for the Google Chrome web browser. Users can execute both smart contracts and transactions on the browser without the Ethereum full node. It is also possible to send and receive Ether by creating an electronic wallet and user account. Software is available to download from https://metamask.io, and it is compatible with Testnets like Ropsten as well as Mainnet.

3.2.3. Remix - Solidity IDE

The compilation, testing, debugging, and deployment can be performed using a web-based IDE (Integrated Development Environment) called Remix. Alongside Solidity, it is also a programming language for developing Ethereum smart contracts. Developers can create, compile, and deploy Solidity smart contracts on the blockchain using the website https://remix.ethereum.org.

3.2.4. Truffle

Truffle is a framework for developing smart contracts and DApps on the Ethereum platform. It is available for download at https://trufflframework.com. Truffle is useful for integrated development management because it has the compiling, linking, and distribution functions required for smart contracts and DApp development. Figure 2 shows Truffle's home page.

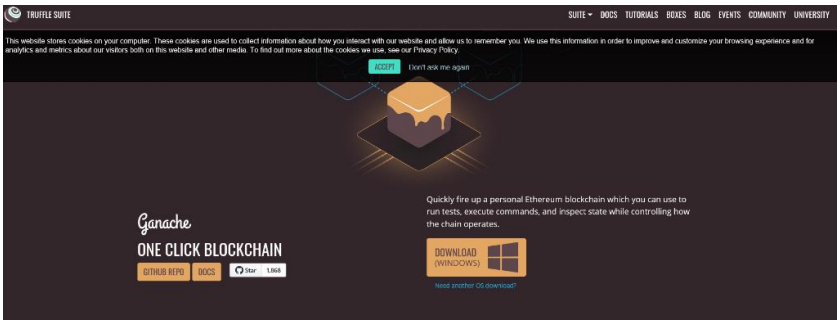

Figure 2. Truffle download screen

3.2.5. Ganache

Ganache is a personal blockchain that can be installed on a PC for the testing and development of blockchain applications. It can be operated locally without a network connection. It can also easily deploy and test smart contracts. Installing the ganache-cli or GUI versions comes with 10 accounts containing virtual 100 ETH coins for testing. It is available to download at https://trufflframework.com/ganache.

3.2.6. A configuration of dApp development environment

The most popular dApp development environments are Ethereum-based dApp development kits. To guarantee the dApps' compatibility, it needs a middleware concatenating blockchain core for the

common user environment, e.g., the web. Seth (Sawtooth-Ethereum) [12] for the Hyperledger Sawtooth platform [13] is a good example of providing Ethereum Solidity compatibility. Seth middleware provides software compatibility for dApp developers. Also, software libraries, e.g., web3.js or node.js, should be supported to provide the dApp developers with a convenient development environment. Figure 3 shows an example of dApp development environment for IoT service [14].

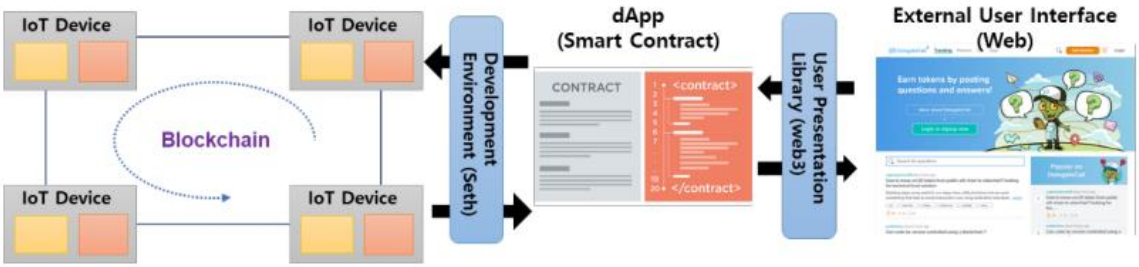

Figure 3. An example of dApp development environment for IoT applications

4. Conclusion

In this paper, we have examined the development environment, related tools, and main terms used when developing DApps on a blockchain platform. Currently, there are a variety of development environments and methods, so beginners who are new to the blockchain may experience confusion. This paper has contributed to the further development of the blockchain industry and ecosystem by highlight key specifics. The hope is that even beginners and students in computer-related majors can access more easily and participate in the development of a blockchain platform or DApp. Additionally, it will also be necessary to upgrade and stabilize the blockchain system through the cooperation and efforts of users across multiple fields and sectors. Further development through the application of new business models and fusion with existing systems should also be sought. Finally, continued research on the interconnectedness among various blockchain platforms, international standardization, and legal and institutional issues will be continued to promptly resolve current issues.

Acknowledgments

This work was supported by the Youngsan University Research Fund of 2020.

References

[1] Jin-whan Kim, “Blockchain Technology and Its Applications: Case Studies”, Journal of System and Management Sciences, Vol. 10, No. 1, pp. 83-93, (2020).

[2] Lin, I. C. & Liao, T. C, “A survey of blockchain security issues and challenges”, international journal of network security, Vol. 19, No. 5, pp. 653-659, (2017).

[3] IRS Global, “Blockchain related global market and business trends”, Market Report, 2019.

[4] Nomura Research Institute, “Survey on blockchain technology and related services”, FY2015 Report, 2016.

[5] Kim, E, “A study for the innovativeness of blockchain”, The Journal of Society for e-Business Studies, Vol. 23, No.3, pp. 173-187, (2018).

[6] Bitcoin Developer Reference, https://bitcoin.org/en/developer-guide, Bitcoin Project 2009-2020.

[7] Nakamoto, S, “Bitcoin: A peer-to-peer electronic cash system”, http://bitcoin.org/bitcoin.pdf, 2008.

[8] Andreas M. Antonopoulos, Gavin Wood Ph.D., “Mastering Ethereum: Building Smart Contracts and DApps”, 2019.

[9] Bellaj Badr, Richard Horrocks, Xun (Brian) Wu, “Blockchain By Example: A developer's guide to creating decentralized applications using Bitcoin, Ethereum, and Hyperledger”, 2018.

[10] Ethereum white paper, https://github.com/ethereum/wiki/wiki/White-Paper

[11] Ethereum yellow paper, https://ethereum.github.io/yellowpaper/paper.pdf

[12] Linux Foundation, Hyperledger Sawtooth Seth. Article (CrossRef Link)

[13] Linux Foundation, Hyperledger Sawtooth Project. Article (CrossRef Link)

[14] Seungcheol Lee, Jaehyun Lee, Sengphil Hong, and Jae-Hoon Kim, “Lightweight End-to-End Blockchain for IoT Applications”, KSII Transactions on Internet and Information Systems, Vol. 14, No. 8, pp. 3224-3242, (2020).

