CRYPTOLIGATION: An Offbeat Blueprint of Crypto Contract in the Decentralized Administration

Subhalaxmi Chakraborty, Subha Ghosh, Rajarshi Das, and Pritam Kundu

1 Introduction

Information systems and information technology must be able to adapt to social change and technological improvements in order to meet the needs of information and societal use of communication technologies [1]. Blockchain technology is an example of a disruptive technology that is still being developed in response to societal requirements [2–4]. This served as the foundation for the creation of additional technologies, including smart contracts [5]. Smart contract is a system for electronic transactions and can automatically carry out a contract's conditions. Blockchain is essentially a condensed form of payment verification. It aims to enforce a contract made between several parties [5]. Because they are disinter-mediated and generally transparent, smart contracts promise economic efficiency, a decrease in transaction and legal costs, and anonymous transactions [6]. It has many desirable characteristics that make it one of the most in-demand technologies, particularly in the financial industry [7] because it lowers the risk of fraud and non-payment, enhances the quality of financial contracts with a certain level of confidence, and is not influenced by intermediaries [8]. The technology supporting smart contracts has made tremendous strides recently, but there is still little knowledge about how to use them in different businesses.

This research explores and presents an offbeat methodology for creating intelligent contract blueprints. This study also discusses several methods and blueprints. Businesses have tried with and imitated this application of smart contract technology in several different sectors or as prototypes. This paper enhances current theoretical research and offers instances of smart contracts in several industries among other things [9]. This document also offers a thorough explanation of how smart contract

S. Chakraborty · S. Ghosh (✉) · R. Das · P. Kundu
University of Engineering and Management, Kolkata, India
e-mail: subhaghoshssk@gmail.com

J. K. Mandal et al. (eds.), *Proceedings of International Conference on Network Security and Blockchain Technology*, Lecture Notes in Networks and Systems 738,
https://doi.org/10.1007/978-981-99-4433-0_40

implementation works. These techniques allow for the regulation of dynamic identification using information. This document also offers a thorough explanation of how smart contract implementation works. These techniques allow for the regulation of dynamic identification using information. As the body of knowledge expands, blockchain will develop in a few sectors. As this study paper indicates, researchers are seeking to adapt it to a range of contexts, including smart contracts, supply chains, and healthcare (PHI) [10]. The security, privacy, and scalability of blockchain are the current research focuses [11]. Applications for blockchain integration in cognitive computing, IoT, supply chains, and healthcare look promising [12]. An overview of how smart contracts are applicable in the decentralized environment finishes this study work.

But for smart contracts to succeed in the commercial sector, they need to be deployed well. This necessitates a broad framework, including legal protections supplied by technological protection mechanisms and rights management data more often known as Digital Rights Management (DRM), to support the functioning and development of blockchains [11].

2 Related Works

Blockchains are shared and distributed ledgers that provide transactions with persistence and verifiability. An instruction on the blockchain is referred to as a transaction because it is cryptographically signed by the user [13]. The core concepts of blockchain are cryptography, hashing, digital signatures, open source systems, and distributed architectures [14].

As of today, there are 1639 different cryptocurrencies available on the digital currency market. In order to validate transactions, the blockchain uses several proof-of-concept, proof-of-work, and proof-of-stake concepts. A set of rules is included in a smart contract, a piece of software that operates on the blockchain. The development of blockchain technology over the past ten years demonstrates how widely used it is. A smart contract is a computer program that has self-driven, self-auditing, and security properties. Nick Szabo first proposed the concept of a smart contract in 1994 [15]. Heho defines it as a computerized transaction protocol that executes the provisions of a contract. In 2008, the year it first used blockchain data, Satoshi Nakamoto [16] published Bitcoin without a centralized command structure. They created an operation-based smart contract within a permission blockchain and carried it out in a unified and synchronized manner [17]. A consensus approach for the use of smart contracts in applications for digital rights management is put out by Hiroki and Ahmed [17] introduced a Hawk, which offers a novel method for creating cryptographically safe smart contracts. In an Internet of things application, Joshua Ellul and Gordon JPace [18] describe how to establish a smart contract with AlkylVM. The transformational nature of blockchain is suggested by Michael Harte, CTO of Barclays [18]. SCM challenges are characterized by issues with supply chain finance as well as issues with lead-time and throughput. Hurlburt [19] discussed the need

Table 1 Comparison of features with existing approaches and CRYPTOLIGATION

SI No.	Liquidity and token handling	Error handling	Atomicity	Security	Sustainability	Economic challenges
[1]	✘	✔	✘	✔	✘	✘
[4]	✔	✔	✘	✔	✘	✔
[7]	✔	✔	✘	✘	✔	✘
[9]	✔	✘	✔	✔	✘	✔
[11]	✘	✘	✔	✔	✔	✘
[14]	✘	✔	✔	✘	✔	✔
[16]	✘	✔	✘	✔	✘	✘
[18]	✔	✔	✘	✔	✘	✔
[20]	✔	✘	✔	✘	✔	✔
CRYPTOLIGATION	✔	✔	✔	✔	✔	✔

for ethical standards and operational direction in his article from April 2016. Before blockchains are widely used to replace conventional transaction databases, he notes that stringent rules, including acceptable behavioral norms, must be established [20].

However, the use of blockchain technology and smart contracts can offer advantages beyond cost savings and supply chain flow optimization [21]. Since the existing IT systems are interconnected, the suggested framework can make it easier for SMEs to be integrated into cross-organizational business activities by establishing an open and unified IT environment [22]. This would ensure that SMEs participate fairly in supply chains. The study's foundations include expert interviews, surveys, and case studies that took place in the context of international business ventures involving smart contracts. We have described the state of art comparison between existing approaches and proposed approach in Table 1.

3 Proposed Methodology

3.1 CRANQ Analysis

CRANQ which is a low-code integrated development environment has a high degree of reusability and allows component building. Its emphasis on defined data types and ports makes it simple to verify intent. This program is a graphical, user-friendly IDE that enables the compilation and deployment of the smart contract. This third-party application helps to understand how WEB 3.0 and the smart contract work. In a regular code editor, it cannot be understood the flow of code. Therefore, CRANQ makes it extremely simple to track down the smart contract and its working.

The fundamental connection between CRANQ and this research is the desire to create a smart contract with an unconventional design. It allows for the creation of a smart contract's blueprint using a drag-and-drop interface. Additionally, it aids a daily programmer in better time management. CRANQ has a huge collection of nodes that are used to generate smart contracts. Every node in CRANQ can have an output and input gateway. It is also flexible, scalable, and a tricky swift tool for every smart contract developer. This third-party application provides huge advantages of reusability and debugging. This application helps by providing the types of input to use factory deploy image (FDI). Figure 1 shows the process of getting access from CRANQ to use it as IDE in our research work.

With the use of smart contracts, which can be executed on top of blockchains using blockchain technology, parties can codify business rules like those found in legally binding contracts. A smart contract can be viewed as an electronic transaction protocol that permits the digital enforcement of the terms of an underlying legal contract in order to satisfy demands like payment, compliance with legal obligations, and enforcement without the need for a third party. Blockchain is basically a publicly available ledger where participants enter data and certify their acceptance of the transaction via an elliptic curve digital signature algorithm (ECDSA). The equation of an elliptical curve is calculated as

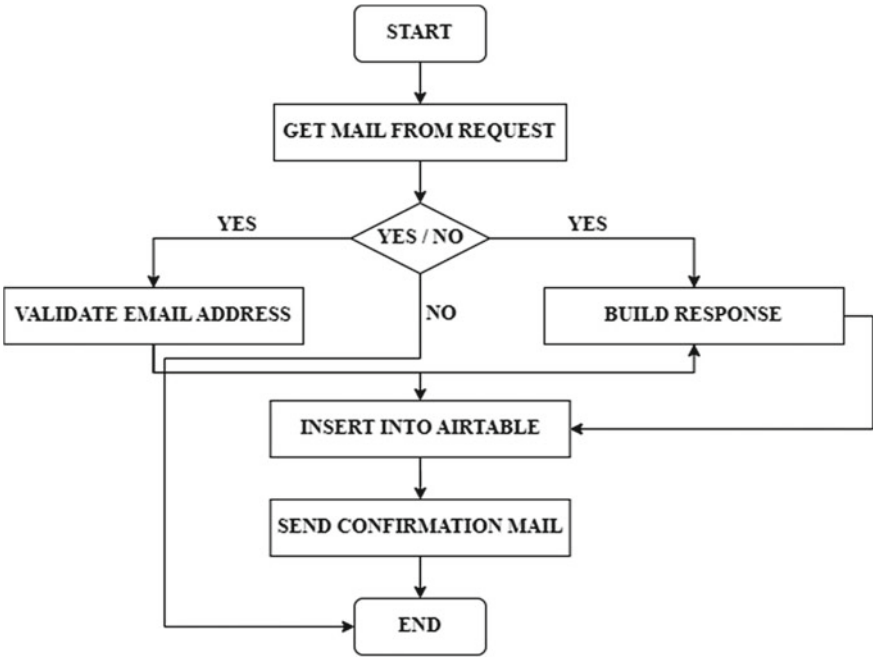

Fig. 1 Flowchart of getting access from CRANQ for generating smart contract

$$y^2 = x^3 + ax + b \tag{1}$$

In Bitcoin and most other implementations, $a = 0$ and $b = 7$, so this is simply

$$y^2 = x^3 + 7 \tag{2}$$

The financial sector has discovered significant applications for smart contract that is executed digitally. Ethereum, Corda, Ripple, Hyper Ledger, and Bitcoin are just a few examples of the blockchain networks where smart contracts are used. Many companies that operate blockchain technologies support smart contracts. To explain the importance of the smart contract, the report is analyzed the difference between traditional security, market-based contracts, and their smart contract counterpart. Blockchain will mature in the number of areas as the body of research grows. A blockchain is a network of computers that must all concur that a transaction has occurred before it is recorded in a chain of computer code, according to a description provided by the financial times in 2016.

The blockchain and smart contract technology are used to bring an offbeat up gradation in the decentralized market. The mathematical solutions are discussed for clarifying the theoretical expression of this ledger's cutting-edge technology. The IDE and its components are described in Sect. (3.1). For generating a smart contract, blockchain technology needs to maintain the various types of protocols. In later parts, these ideas as well as the algorithmic component of creating smart contracts are covered.

3.2 Schematic Algorithmic Structure of CRYPTOLIGATION

'CRYPTOLIGATION' word is discovered from 'crypto' and 'obligation'. 'Obligation' means one type of contract that creates and pays transaction fees to anonymous miners. An offbeat schematic algorithmic structure (Fig. 2) of CRYPTOLIGATION is designed and implemented on a specific scalable platform that can swiftly contribute work pressure in blockchain technology. This structure is an architect by keeping the means of security, scalability, flexibility, and maintenance of smart contracts in blockchain technology.

The starting position of generating a smart contract in CRANQ is used by getting help from the node of 'start' (as shown in Fig. 2). Mostly, the 'start' node is used to kick start the entire program. As per peer reviews, it is already found that there is no solidity language in CRANQ, but CRANQ only supports the nodal architectural view because of its nodal connections. 'Store' (known as 'Config' in Fig. 2) is used for reading the user inputs and passing the information to its neighbor nodes. Suppose somehow the input of the 'store' node is empty, then any attempt to read the content will result in a signal sent out via not found. These (private key, crypto wallet account address, a kind of network, provider URL) are validified by 'store' and then passed to its successive nodes. Therefore, an extra 'store' node is added

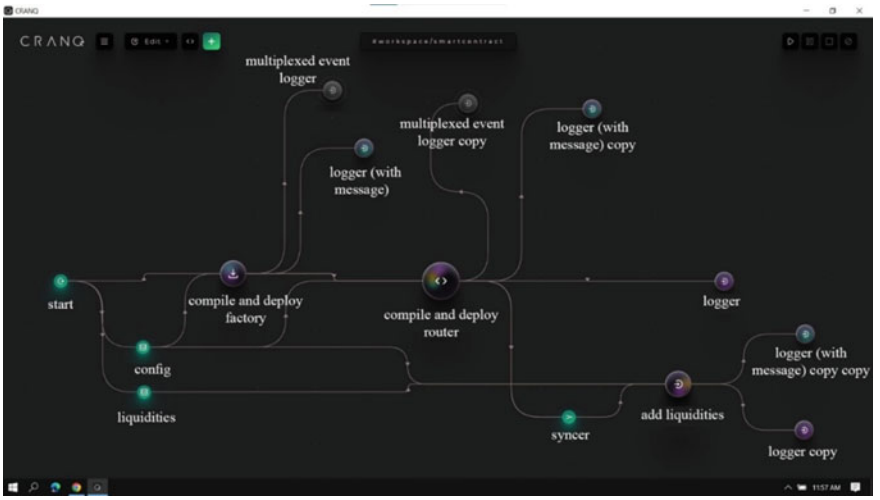

Fig. 2 Offbeat blueprint of smart contract

and named it 'liquidities'. This node is used mainly for getting the token addresses as inputs. Also, exchange rates have been given to this node. The following program code is used to create and connect the nodes to each other in this research work.

```
program Inflation (Output of START and CONFIG).

begin

    SetTimeOut(equals to or greater than start(NULL, 'START'), 0)

#fetch data

    State->data = data
    Outputs->written(NULL, tag)

#read instances (provider URL, private key, account address, network)

    constant value = state->data is not equals to undefined state->data: params-
    >data
    if value is not equals to undefined

        outputs->data(value, tag)

    else

        outputs["NOT FOUND"](NULL, tag)

#read token address and exchange rate

    State->data = data
    Outputs->written(NULL, tag)

end
```

Inside 'compile and deploy factory', a collection of various functioned nodes is gathered. This node helps to provide the authentication based on the inputs which have been collected from its previous node and also compiles smart contract. 'Multiplexed event logger' and 'logger with message' are used for debugging the 'factory' and 'router' nodes. These exception handling nodes are connected for getting green signal from its neighbor connections. The following program code has been provided for better experience.

program Inflation (Output of COMPILE AND DEPLOY FACTORY).

```
begin
#item getter
    constant { } = state
    let bundle = get bundle tag
    if bundle is not present
        bundle = static bundle
        tag and bundle
    let ports = get ports tag
    if ports are not present
        ports = set dynamic fields
        set tag and ports
    bundle[input] = data
    delete input of ports
    if size of ports is equals to 0
        delete bundle tag
        delete ports tag
    read instances()
    factory ABI()
#send factory address
    contract deployment waiter()
#find error and passes signal to a logger node
    factory deployer()
end
```

'Compile and deploy factory' node generates 'factory address' and passes it to 'compile and deploy router'. This 'router' node helps to generating actual smart contract which is implemented in blockchain technology. The following pseudo-program code is attached here.

program Inflation (Output of COMPILE AND DEPLOY ROUTER).

```
begin
```

```
#get network
    get network()
#passes w-eth address
    while (router deployer assign data in chain)
        contract deployment waiter()
            return router address, router ABI
end
```

'Compile and deploy router' provides the router address to 'Syncer' (follows the below program code). 'Syncer' node is bundled up the same types of data or information and passes these for the next phase. These two nodes are different in functionalities. That is why an interface (called as 'application binary interface') is used to communicate in this particular architecture. In Fig. 2, the transaction is completed in 'add liquidities' node. This node is taking the output of 'Syncer' as input and helping to complete the transaction. Other side, this node fetches the data from user end with router contract inputs, and if any error occurs, then the 'logger (with message)' node gives the red signal, otherwise it will be green in whole procedure.

```
program Inflation (Output of SYNCER and LIQUIDITIES)
    begin
    #read router address and ABI
        constant { } = state
        let bundle = get bundle tag
        if bundle is not present
            bundle = static bundle
            set tag and bundle
        let ports = get ports tag
        if ports are not present
            ports = set dynamic fields
            set tag and ports
        bundle[input] = data
        delete input of ports
        if the size of the ports equals to 0
            delete bundle tag
            delete ports tag

#add in liquidities
Address = string,
ABI = {string: any}()
```

```
config()
#calculate deadline
    timestamp(new Date(), tag);
item getter()
#adder
    sum(data->a + data->b, tag);
liquidities adder()
end
```

4 Result and Discussions

There is already widespread agreement that the blockchain's influence will extend well beyond crypto currencies and might have disruptive consequences on the creation of distributed applications. Smart contracts able to support a new type of distributed computing are the primary enabler for this impact. The quantity and variety of smart contract platforms are constantly expanding; this article examines one of them. As a result, the ensuing technical landscape is becoming more complex and heterogeneous. The conceptual foundations of this new landscape are more integrated than one might anticipate from an application standpoint, as this article demonstrates, and smart contracts can in fact be seen as the fundamental building blocks-or services-of a blockchain-based, service-oriented computing paradigm. This proposed approach also demonstrates how we are putting into practice a smart contract approach that values interoperability and reusability as positive characteristics. Several obstacles must be overcome in order to implement the smart contract protocol in blockchain and fully utilize smart contracts. It is startling how little attention has been given to making it possible for developers to reuse contracts that have previously been deployed, especially given that it is generally thought that deploying a new contract is more expensive than just calling a job that has already been done. The idea of resource consumption and the cost of invocations are organically incorporated into smart contracts. While generic contracts that may require transaction processing may result in greater, unpredictable response times, libraries, and data contracts are processed locally inside of each node and have minimal response times. It makes sense that platforms focus on their own technologies these days as their unique selling points. Developing common protocols, interface styles, data formats, and, of course, mechanisms for authentication and certification is a problem. Finally, composition solutions must be designed and implemented in order to abstract away from technical details and give developers the tools and infrastructures they need to increase productivity. Only then will smart contracts' full potential be realized.

Whatever is envisioned is, in essence, a progression from the current technology silos to an abstract, reuse-oriented contract system that can maintain the assurances unique to blockchain technology. Figure 3 demonstrates that this eccentric smart

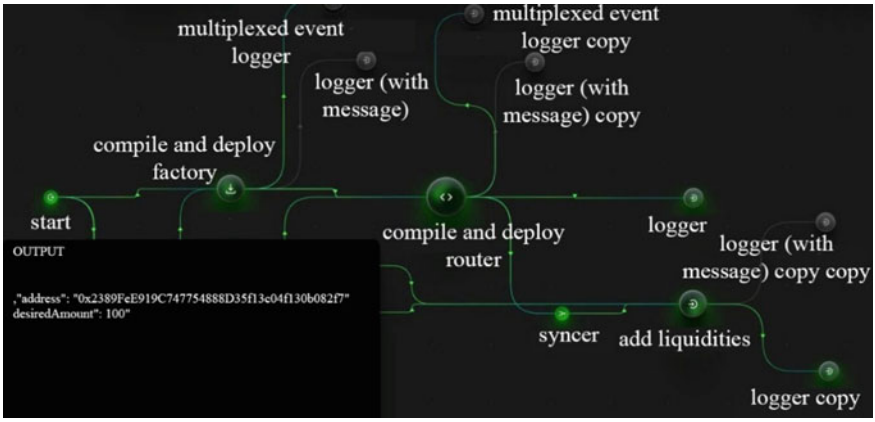

Fig. 3 Execution status of smart contract

Table 2 Comparison with other research papers

SI No	Research papers	Complexity	Transparency	Security
1	Prause, G. (2019). Smart contracts for smart supply chains [6]	O(n^2)	Moderate	Good
2	A triplicate smart contract model using blockchain technology [9]	O(n)	High	Moderate
3	CRYTOLIGATION	O(n)	Good	Best

contract has been successfully run and that all the nodes have received the go-ahead. So, our eccentric blueprint is now good to go and ready to be applied to a crypto currency online application. We have compared the complexity, transparence, and security of the existing approaches [6, 9] with proposed approach in Table 2.

5 Conclusion

The information technology has emerged as a crucial innovation. Here, we have discussed blockchain technology as a catalyst for new applications in the financial and non-financial sectors of healthcare, supply chain, and industrial manufacturing. According to the report, by providing secure trust frameworks, fostering agile value chain production, and fostering closer interaction with technologies like cloud computing and IoT, smart contracts can play a crucial part in revolutionizing the currencies of various sectors and applications. This smart contract's eccentric

design aims to eliminate the necessity for an existing, unreliable third party by self-enforcing contractual requirements like payments and legal duties. In addition to self-organizing and self-optimizing, the study emphasizes how industry 4.0, blockchains, and smart contracts concepts are connected to each other and fit structurally well together. Blockchain has built-in, highly effective cyber security protections that enable instant contracts, engagements, and agreements. This paper explains how, why, and from what angle this smart contract is bringing about a seismic shift in the blockchain industry.

Acknowledgements I would like to convey my keen gratitude to Subhalaxmi Chakraborty, my research supervisor, for giving me the chance to do research and for her helpful advice during this process. Her energy, vision, genuineness, and drive have really motivated me. Additionally, I want to thank Subha Ghosh for his insightful comments and ideas.

References

1. Negara ES, Hidayanto AN, Andryani R, Syaputra R (2021) Survey of smart contract framework and its application. Information 12(7):257
2. Palit SK, Chakraborty M, Chakraborty S (2022) Performance analysis of 5GMAKA: lightweight mutual authentication and key agreement scheme for 5G network. J Supercomput 1–34
3. Palit SK, Chakraborty M, Chakraborty S (2022) MASKA: mutual authentication and session key agreement protocol in global mobility networks. In: Applications of machine intelligence in engineering. CRC Press, pp 309–321
4. Daniel F, Guida L (2019) A service-oriented perspective on blockchain smart contracts. IEEE Internet Comput 23(1):46–53
5. Gupta R, Shukla VK, Rao SS, Anwar S, Sharma P, Bathla R (2020) Enhancing privacy through "smart contract" using blockchain-based dynamic access control. In: 2020 international conference on computation, automation and knowledge management (ICCAKM). IEEE, pp 338–343
6. Prause G (2019) Smart contracts for smart supply chains. IFAC-PapersOnLine 52(13):2501–2506
7. Mohanta BK, Panda SS, Jena D (2018) An overview of smart contract and use cases in blockchain technology. In: 2018 9th international conference on computing, communication and networking technologies (ICCCNT). IEEE, pp 1–4
8. Younus D, Muayad A (2021) Supply chain using smart contract blockchain technology in organizational business. Eur J Res Dev Sustain
9. Eze P, Eziokwu T, Okpara C (2017) A triplicate smart contract model using blockchain technology. Circ Comput Sci-Spec Issue 1–10
10. Azka LI, Firdaus R (2023) Blockchain technology based smart contract model in Indonesia. Blockchain Frontier Technol 2(2):58–63
11. William ADJ, Rajendran S, Pranam P, Berry Y, Sreedharan A, Gul J, Paul A (2023) Blockchain technologies: smart contracts for consumer electronics data sharing and secure payment. Electronics 12(1):208
12. Kumar NM, Chopra SS (2022) Leveraging blockchain and smart contract technologies to overcome circular economy implementation challenges. Sustainability 14(15):9492
13. Lin SY, Zhang L, Li J, Ji LL, Sun Y (2022) A survey of application research based on blockchain smart contract. Wireless Netw 28(2):635–690

14. Verma M (2021) Smart contract model for trust based agriculture using blockchain technology. Int J Res Anal Rev 8(2):354–355
15. Dolgui A, Ivanov D, Potryasaev S, Sokolov B, Ivanova M, Werner F (2020) Blockchain-oriented dynamic modelling of smart contract design and execution in the supply chain. Int J Prod Res 58(7):2184–2199
16. Li X, Jiang P, Chen T, Luo X, Wen Q (2020) A survey on the security of blockchain systems. Futur Gener Comput Syst 107:841–853
17. Golosova J, Romanovs A (2018) The advantages and disadvantages of the blockchain technology. In 2018 IEEE 6th workshop on advances in information, electronic and electrical engineering (AIEEE). IEEE, pp 1–6
18. Halaburda H (2018) Blockchain revolution without the blockchain? Commun ACM 61(7):27–29
19. Meng W, Wang J, Wang X, Liu J, Yu Z, Li J, Chow SS (2018) Position paper on blockchain technology: Smart contract and applications. In: International conference on network and system security. Springer, Cham, pp 474–483
20. Tasatanattakool P, Techapanupreeda C (2018) Blockchain: challenges and applications. In: 2018 international conference on information networking (ICOIN). IEEE, pp 473–475
21. Sharma P, Jindal R, Borah MD (2022) A review of smart contract-based platforms, applications, and challenges. Cluster Comput 1–27
22. Zou W, Lo D, Kochhar PS, Le XBD, Xia X, Feng Y, Xu B (2019) Smart contract development: challenges and opportunities. IEEE Trans Softw Eng 47(10):2084–2106