IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 October 2024, accepted 17 November 2024, date of publication 29 November 2024, date of current version 24 January 2025.

Digital Object Identifier 10.1109/ACCESS.2024.3509036

RESEARCH ARTICLE

Exploring the Integration of Blockchain and Distributed DevOps for Secure, Transparent, and Traceable Software Development

JUNAID NASIR QURESHI[1], MUHAMMAD SHOAIB FAROOQ[1], USMAN ALI[1], ADEL KHELIFI[2], (Senior Member, IEEE), AND ZABIHULLAH ATAL[3]

[1]School of System and Technology, Department of Computer Science, University of Management and Technology, Lahore 54000, Pakistan
[2]Computer Science and Information Technology, Abu Dhabi University, Abu Dhabi, United Arab Emirates
[3]Department of Computer Science, Kardan University, Kabul 1007, Afghanistan

Corresponding author: Zabihullah Atal (z.atal@kardan.edu.af)

ABSTRACT Distributed DevOps is a software development methodology that aims to integrate the work of development and operations teams without being bound by geographical constraints. This methodology excels in enhancing collaboration and speeding software development. However, it does suffer from a lack of security, transparency, and traceability, which can result in project delays, a lack of trust between stakeholders, and even project failure. This paper addresses these issues of Distributed DevOps by implementing Blockchain technology. In this paper, we propose a novel framework that leverages blockchain technology to address the challenges faced by Distributed DevOps. Through performance analysis, we demonstrate the effectiveness of our framework in a real-world scenario, highlighting its ability to improve transparency, traceability, and the security of the DevOps pipeline. Our findings underscore the potential of blockchain-empowered solutions in revolutionizing DevOps practices. Furthermore, this research offers a practical framework for organizations seeking to optimize their development processes by integrating blockchain technology.

INDEX TERMS DevOps, blockchain, smart contracts, distributed DevOps, interplanetary file system, decentralized.

I. INTRODUCTION

Software development has been becoming more complex day by day, and to overcome the complexities and challenges of software development, one of the best approaches is using DevOps. DevOps is a relatively new technique used to give new software updates with maximum reliability and accuracy while doing continuous integration and continuous development (CI/CD) [1]. DevOps combines the Development Team and Operation Team's people, processes, and technology to work in a collaborative environment for delivering software as a service [2]. DevOps eliminates the gap between the Development Team and the Operation Team [3], and because of that, the software update can be released successfully and rapidly. Figure 1 shows a visual representation of the DevOps workflow. On the other hand, Distributed DevOps is an approach to developing software that involves distributed teams working together to build, deploy, and collaboratively maintain the software [4]. In traditional DevOps models, developers and operations teams work together to build and deploy software, but this collaboration often occurs within a single geographical location. In contrast, Distributed DevOps involves geographically dispersed teams that may not be part of the same organization. The potential advantage of distributed DevOps is that it allows organizations to tap into a wider pool of talent and expertise, as they are not limited to a single location. It can also enable organizations to be more agile and responsive to changing market conditions, as they can quickly bring new resources online as needed.

The adoption of DevOps practices in distributed teams offers numerous benefits, but despite being a highly effective approach for software development, it presents challenges

The associate editor coordinating the review of this manuscript and approving it for publication was Thanh Ngoc Dinh.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

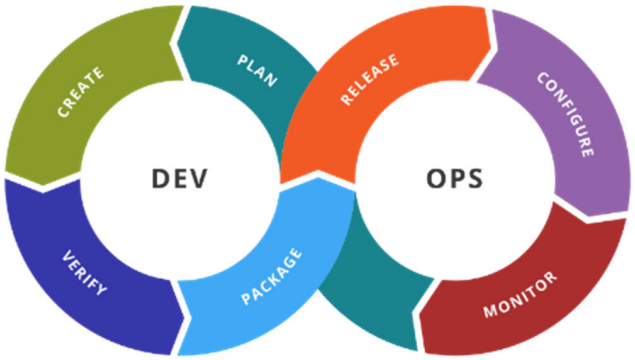


FIGURE 1. DevOps Workflow [5].

in areas such as transparency, trust, security, and traceability. The problem that this study addresses is the need to enhance transparency, trust, security, and traceability within Distributed DevOps. It is critical to establish trust and transparency, particularly when teams are working across different locations and time zones [6], for instance, considering a scenario where teams are located on separate continents. The lack of transparency and trust can result in poor collaboration and communication between team members [7], leading to misunderstandings and misaligned goals. This can also result in delays, wasted effort, and decreased productivity. Security is also paramount to prevent malicious actors from disrupting or compromising the project [8] because the development pipeline is susceptible to cyber threats and malicious attacks [9]. Imagine a situation where some unauthorized person accesses critical project data. This can result in data breaches that can be the reason for financial losses or project failure in the worst-case scenario. Moreover, software traceability also holds significant importance as it allows the traceability of different artefacts, including requirements, design models, and code [10] for example, a software bug surfaces during the testing phase. In the absence of comprehensive traceability, identifying the root cause becomes similar to finding a needle in a haystack. This lack of traceability can result in prolonged debugging cycles, increased costs, and project delays.

Blockchain technology can significantly impact Distributed DevOps and software development as a whole for several reasons. Its decentralized nature, immutability, and distributed ledger can ensure security, trust, and traceability in the software development pipeline. Blockchain can be defined as an interconnected sequence of blocks, where each block consists of transactions, hash, and the previous block's hash, as shown in Figure 2, resulting in decentralized and tamper-proof data [11]. With each block in the chain containing a cryptographic hash to the previous block, it's nearly impossible to alter data without affecting the entire chain. This ensures that the transactions done on the blockchain are safe and tamperproof. Every transaction in the blockchain is recorded in a distributed ledger which is visible to all participants, due to this feature, transparency and traceability are achieved. Blockchain combines the uniqueness and innovation of computing technologies like distributed data storage, decentralized peer-to-peer transactions, intelligent consensus mechanisms, and dynamic encryption algorithms [12]. Since its emergence, blockchain technology has undergone significant advancements, enabling the construction of Smart Contracts that automatically store and execute code on the blockchain. These smart contracts are also excellent in streamlining digital interactions and transactions [13]. To overcome the challenges of trust, transparency, security, and traceability in software development with Distributed DevOps, an efficient blockchain-based framework is essential.

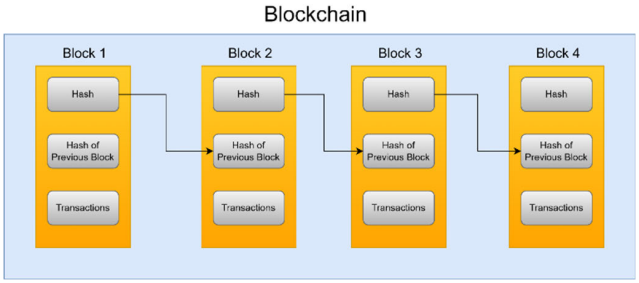


FIGURE 2. Blockchain Representation.

In this paper, we propose a framework that seamlessly integrates Distributed DevOps with blockchain technology, leveraging the benefits of transparency, traceability, and security. The integration is facilitated through the use of the InterPlanetary File System (IPFS), Smart Contracts, and consensus mechanisms. IPFS ensures the secure storage of critical files such as project plans, code, and artefacts, with only their hash values uploaded to the blockchain via Smart Contracts. This not only ensures data integrity but also streamlines the DevOps process. The Smart Contracts automate tasks, reducing the time and effort required for the software development cycle. Robust security measures, including cryptographic hashing and encryption, are implemented to protect against cyber threats. The consensus algorithm is implemented to bring all the nodes in agreement and is used to achieve agreement on a single data value among distributed processes or systems. Additionally, scalability concerns inherent in blockchain are mitigated through the utilization of IPFS. This introduction aims to provide a foundational understanding of our innovative framework.

The novelty of our proposed framework lies in its comprehensive approach to enhancing the security, traceability, and transparency of software development within distributed teams using DevOps. The framework leverages the capabilities of blockchain, including distributed data storage, decentralized peer-to-peer transactions, intelligent consensus mechanisms, and dynamic encryption algorithms. This approach addresses the identified shortcomings of Distributed DevOps, distinguishing this research from previous studies in this specific domain. The strategy also has the

potential to mitigate issues between clients and software development teams. We have also implemented the framework to assess its performance in a real-world scenario. The results show that in our framework, the time of adding a new block in the chain is better as compared to most well-known blockchains e.g., Bitcoin, and Ethereum. The findings of this research highlight the significant potential of Blockchain to revolutionize distributed DevOps by making it more secure, transparent, traceable, and trusty.

The remaining paper organized is as follows: In Section II, the related work has been presented, and in Section III, we have elaborated Preliminaries used in the model. The proposed framework is shown in Section IV, and Section V contains the implementation and performance. Furthermore, in Section VI, the discussion is shown, and finally, Section VII describes the conclusion and future work.

II. RELATED WORK

There has been a significant amount of research done on the intersection of blockchain with DevOps and other software development models. Here are a few examples of research in this area:

Bankar and Shah [14] also presented a paper that intersects blockchain with DevOps. This paper describes how blockchain technology can be used in DevOps for software development. The benefits of this include improved quality and performance, as well as increased security. This is achieved by storing all project artefacts in a decentralized and secure blockchain environment. Although, the proposed solution does not provide a solution when teams are working in a distributed environment.

Akbar et al. [15] have investigated the potential benefits of using blockchain technology in a DevOps paradigm. A framework is proposed which helps merge the characteristics of blockchain with the DevOps paradigm. This provides an effective means for the adoption of blockchain in DevOps while ensuring its advantages over other solutions are maximized. However, the proposed framework does not address any parts related to payment.

Tariq and Colomo-Palacios [16] analyzed the usage and benefits of Blockchain Smart Contracts in Software Engineering. The study also highlighted several challenges that have yet to be addressed by current methodologies. The findings suggest that a greater practical application of this system has the potential to pave the way for future research opportunities. However, this study does not mention any about DevOps or Distributed DevOps.

Faruk et al. [17] examine the impact of existing software engineering processes. They consider the need to adopt new concepts and evolve current software engineering processes for blockchain systems. The study also looks at the role of software project management in the development of blockchain-oriented software. Nevertheless, this was a systematic study and does not present any kind of framework.

Ramakrishna [18] present a novel framework that integrates Blockchain technology into Agile software development processes. This paper explores the potential of Blockchain as a means to address these challenges by enabling robust tracking and traceability mechanisms within Agile software development. Thus, this research is only for agile software development and not DevOps.

Krishnaiah et al. [19] present a survey of how metadata of software development is presented by using blockchain technology. The survey is based on security and privacy for data storage. Our model works for Distributed DevOps software development processes.

Al-Nakeeb et al. [20] have given innovative work to digitize the transformation of information for Business Intelligence and Analytics, which affects the cloud and DevOps in managing the projects. It reflects the idea but differs in comparison to our distributed software development process integration of blockchain for DevOps.

Terzi and Stamelos [21] provide security and data quality management for eHealth systems using blockchain. Our model is generally not just for health care and reflects blockchain with DevOps for a distributed environment.

Qureshi and Farooq [22] provide the working framework for distributed software development based on blockchain. It integrates the blockchain in a distributed software development environment but does not integrate the DevOps.

These are just a few examples of the many research efforts focused on the blockchain with DevOps and other software development models. It is worth noting that while these studies have the potential to bring many benefits to DevOps and other software development processes, they do not present any framework that is suitable for Distributed DevOps where teams are not in the same place or even time zone.

The novelty of our proposed framework is that it could make it simpler for distributed teams to communicate and share data in a secure, traceable, and immutable environment. We have presented a comprehensive strategy for implementing DevOps methods concerning Blockchain to successfully execute projects while reducing conflicts. We have solved the issues of traceability, trust, and transparency of Distributed DevOps so that everyone can work in a trusted and more secure environment. DevOps phases: project initiation, continuous development, continuous integration & delivery, continuous deployment, and continuous monitoring are shown, and how they will work with the Blockchain is also described. Using cryptocurrencies and digital wallets, we have also resolved payment-related issues. In addition, the Interplanetary File System is used to address the scalability issue of the Blockchain.

III. PRELIMINARIES

This section highlights the preliminaries for the proposed framework. The major components that will be used in this framework will be described in this section including, IPFS, Decentralized Applications, Blockchain, Smart Contracts, and Jenkins.

TABLE 1. Comparison of the proposed framework with related work.

Reference	Paper	Blockchain-Based Study	Framework Presented	Distributed DevOps	Payment Solution
[14]	Blockchain-based framework for Software Development using DevOps	Yes	Yes	No	No
[15]	Toward Effective and Efficient DevOps Using Blockchain	Yes	Yes	No	No
[16]	Use of Blockchain Smart Contracts in Software Engineering: A Systematic Mapping	Yes	No	No	No
[17]	Software Engineering Process and Methodology in Blockchain-Oriented Software Development: A Systematic Study	Yes	No	No	No
[22]	Blockchain in Agile Software Development	Yes	No	No	No
[23]	Applying Blockchain to Improve the Integrity of the Software Development Process	Yes	Yes	No	No
[24]	ChainSoft: Collaborative Software Development using Smart Contracts	Yes	Yes	No	Yes
[25]	Is it worth adopting DevOps practices in Global Software Engineering? Possible challenges and benefits	Yes	No	No	No
This Paper	Blockchain-Based Framework for Distributed DevOps	Yes	Yes	Yes	Yes

A. IPFS (INTERPLANETARY FILE SYSTEM)

IPFS (Interplanetary File System) enables decentralized storage and sharing of data in a content-addressable distributed file system [23]. It operates by storing clusters of hashed files in individual nodes of the system [24]. IPFS uses a peer-to-peer network to make it easy for people to share files without having to go through central authorities or servers while also ensuring data resilience and availability. With content-addressed links, it reduces redundancy and accelerates content retrieval, promoting a more efficient and censorship-resistant internet infrastructure.

B. BLOCKCHAIN AND SMART CONTRACTs

Blockchain is a decentralized ledger system that offers a safe, transparent, and permanent record of transactional data between two parties [25]. Through the use of cryptographic techniques, it produces a record of transactions that cannot be altered. This record takes the form of a chain of blocks, each of which references the block that came before it.

Smart Contracts are code snippets designed to execute versatile tasks [26]. They are kept on a blockchain.

They make it possible to execute complicated transactions in a way that is tamper-proof, transparent, and efficient, which could potentially reduce the need for intermediaries. Table 1 shows the proposed framework comparison.

C. DECENTRALIZED APPLICATIONS

DApps are open-source, decentralized applications that can operate without human intervention [27]. These applications are made with smart contracts and have a front and back end that runs on a decentralized peer-to-peer network.

D. JENKINS

Jenkins is a widely used open-source automation tool for CI/CD in software development [28]. Its flexibility, extensibility, and support for pipeline-as-code make it a popular choice among development teams. Its web-based user interfaces, collection of plugins, and other various features make it easy to manage and monitor the build, test, and deployment process, which leads to more efficient and streamlined software development.

IV. PROPOSED FRAMEWORK

In this section, a proposed framework is presented that will make the CI/CD more traceable, secure, and trustworthy even when the teams are scattered in different locations. In this framework, the Blockchain, DApps, Jenkins, Smart Contracts, and IPFS have been used to make DevOps more secure, transparent, traceable, and immutable without disturbing the automation of DevOps. Figure 3. shows the flow chart of the proposed framework.

The main goal of this study is to bring data like source code, files, and artefacts to a distributed ledger so that it can become accessible easily for every stakeholder while still being tamperproof and secure.

A. HIGH-LEVEL ARCHITECTURE

The high-level abstract architecture for the proposed framework is shown in Figure 4. It shows the overall working of the model that will use Blockchain, an Interplanetary File System, and a Smart contract. In architecture, the focus is on data being decentralized without being in control of a central authority.

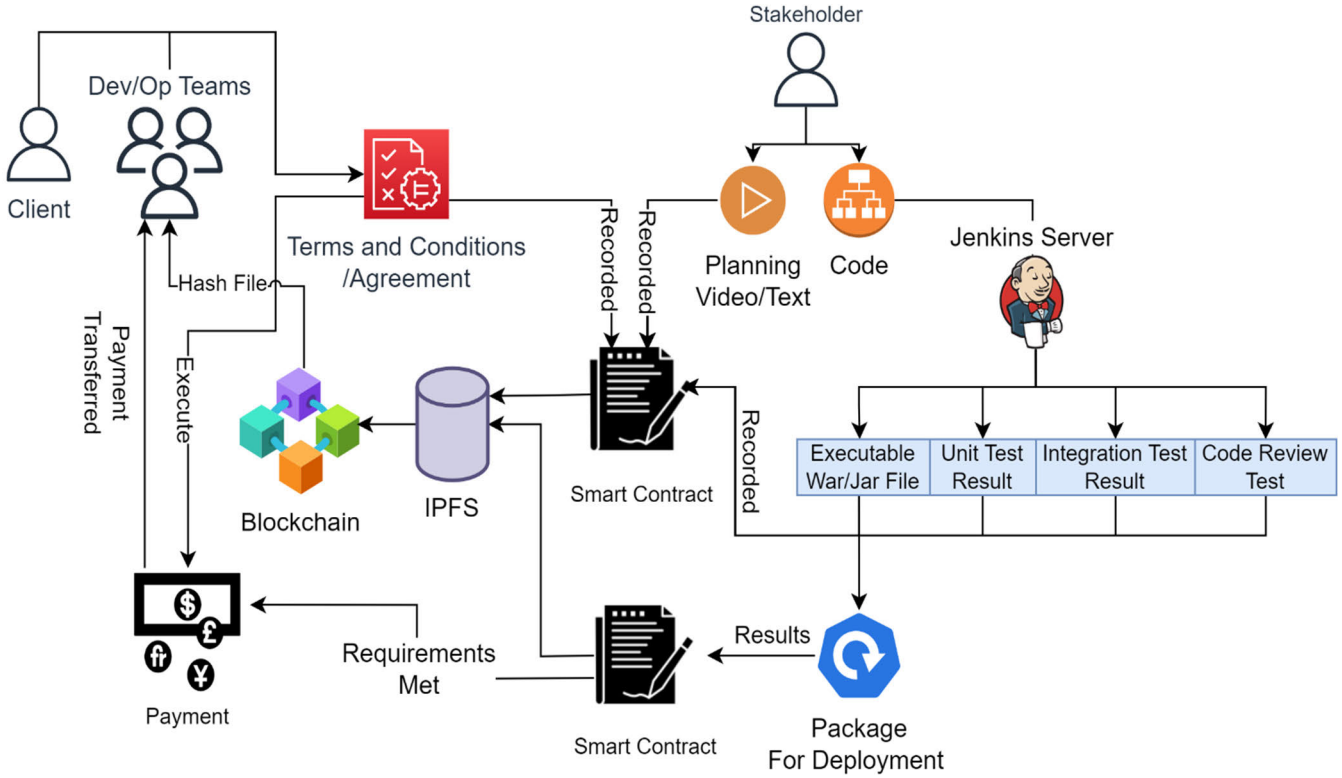


FIGURE 3. Proposed framework process flow.

B. LAYERED ARCHITECTURE

The framework adheres to the blockchain architectural style, and we have presented the 7 layered architecture in Figure 5.

1) PRESENTATION LAYER

The proposed system's Presentation Layer includes a user-facing interface and a decentralized application (DApp). Its primary goal is to connect clients and DevOps teams to the system.

2) APPLICATION LAYER

The metadata relating to transaction records, payments, mockups, prototypes, etc., along with agreements made between DevOps teams and clients in the form of videos, text, and audio, are present within this layer. Digital currencies, including ETH, USDT, BUSD, or BTC, are also in this layer to facilitate transactions after the completion of one successful iteration. The primary function of this layer is to enable seamless communication among stakeholders while serving as an intermediary between the Presentation Layer and the Business Logic Layer.

3) BUSINESS LOGIC LAYER

This layer has all the smart contracts that govern the terms and conditions of interactions within the system. It serves as an active database for these contracts, enabling the acknowledgement, execution, and enforcement of communication rules. This layer plays a critical role in facilitating the functioning of the system by ensuring that all interactions are carried out by established rules and regulations.

4) TRUST LAYER

The Trust Layer of the layered architecture of distributed DevOps is responsible for managing the system's consensus algorithms, such as Proof of Stake or Proof-of-Work. The trust layer plays a critical role in ensuring the security and reliability of the system by implementing robust consensus algorithms and security protocols.

5) TRANSACTION LAYER

The transaction layer is responsible for facilitating the developers and customers by enabling them to trigger transactional smart contracts. This layer also oversees the processes of mining/staking and validating the blocks containing these transactions. The transaction layer plays a critical role in the functioning of the proposed system.

6) HARDWARE/INFRASTRUCTURE LAYER

This layer includes the peer-to-peer network that validates transactions and a distributed storage system that stores and retrieves files on the decentralized storage systems.

7) SECURITY LAYER

The Security Layer is responsible for security measures to protect the network from potential attacks. The security layer

works parallel with the rest of the system, incorporating algorithms and security protocols to safeguard the blockchain network.

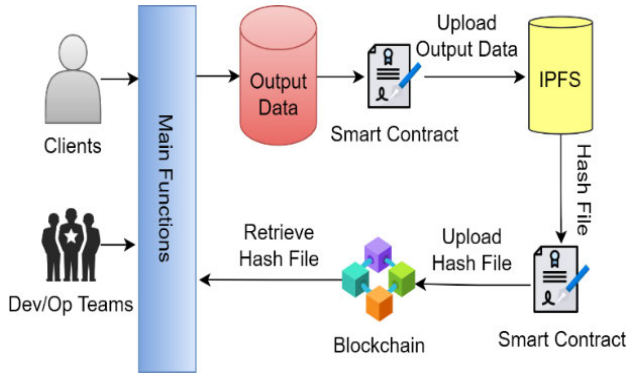


FIGURE 4. Abstract level architecture.

C. DETAILED ARCHITECTURE

This section describes every step of DevOps in detail with the implementation of Blockchain technology. The Detailed framework Architecture layer-wise of the system is shown in Figure 6.

1) PROJECTION INITIATION

The manager or owner can create the project at the project's initiation. After creating a project, the manager/owner can add members to the project using data that include the Name, Username, Email, and Contact Number of developers. After that, all members will be given unique identity materials like a "Key." Members can log in to the project using unique keys to do their tasks. All unique keys, along with members' data and basic project details are uploaded on the blockchain using a smart contract.

The owner or manager will write all basic requirements and terms and conditions at the project's initiation. Table 2 and 3 show the JSON file for the agreement between the client and the organization.

TABLE 2. File for customer agreement.

Customer's Agreement
{
"Total budget": "$1000"
"Project deadline": "30/10/2023"
"Each Iteration Duration": "2 Weeks"
"Requirements": "All project requirements should be completed within budget and time"}

All stakeholders need to accept the terms and conditions set by the other side to reach a consensus. After that, the data can be stored on the blockchain using a smart contract after the consensus is met between the manager/owner, developers, and clients. The complete work of the Project Initiation Phase is shown in Figure 7.

TABLE 3. File for organization's agreement.

Developer's Agreement
{
"Total Payment": "$1000"
"Payment after each Iteration or 2 weeks": "$100"
"Payment deadline": "After each Iteration or 2 weeks"
"Additional Requirement Price": "$100" }

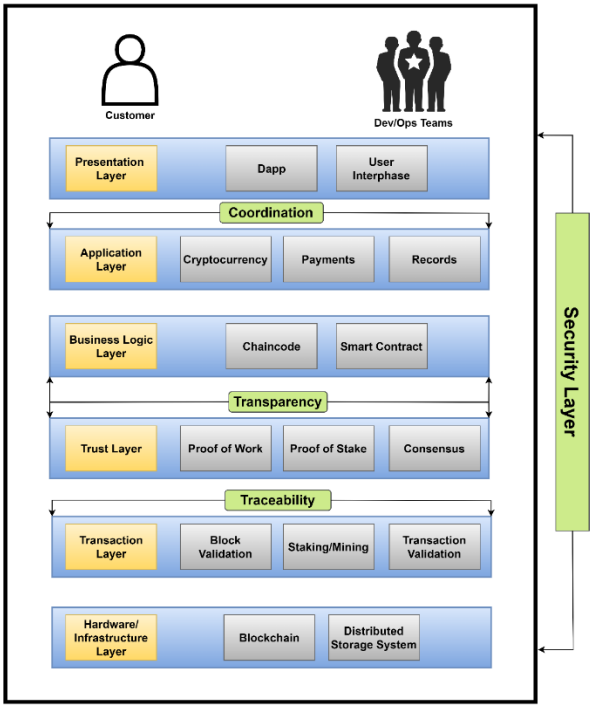


FIGURE 5. Layered architecture for distributed DevOps.

2) CONTINOUS DEVELOPMENT

In the context of DevOps, the continuous development phase refers to the process of continuously planning and coding.

PLANNING: In this phase, the developer, managers, and all other stakeholders directly or indirectly involved in software development make a plan on how to complete the project successfully. Because the teams are located in different locations, the planning can happen via online video calling platforms like Zoom or Google Meetings. All the details of the planning phase, like video conference recordings and notes, can be stored in the IPFS because of Blockchain's scalability issue [29]. The integration of IPFS can be achieved by utilizing the IPFS Desktop App, which has a user-friendly interface for data uploading. After the data is imported/uploaded, the app will generate a CID (Content Identifier), also known as hash data/file which can be used to locate the project file from anywhere in the world. Figure 8 shows the uploaded project on the IPFS Desktop App along

with its generated CID. Subsequently, this returned hash file will be stored on the Blockchain like any other transaction data stored to enhance tamper-proofing and security, ensuring that it remains unalterable, as illustrated in Figure 9. This data is visible to all the stakeholders. In this way, anyone who can access details can see them but not change them. In traditional planning, there is no involvement of a secured and tamper-proof system, which makes the planning phase vulnerable to documents being deleted, manipulated, or tampered with, resulting in confusion and clashes between stakeholders. However, with Blockchain in place, the data can be securely stored.

CODING: Git has been one of the most popular tools for distributed revision control systems [30] and is widely used in DevOps. However, the Interplanetary File System (IPFS) can be used as a decentralized version control system in software development [31]. IPFS provides a content-addressed version control system for managing files. With IPFS's distributed storage, version control, and content addressing features, a decentralized and resilient file management system can be made that operates similarly to Git. However, it is important to note that IPFS cannot fully replicate Git. One such limitation is that IPFS does not support delta storage, meaning entire files rather than just the changes are stored, which can lead to performance inefficiencies in large projects. Additionally, IPFS requires manual management of version metadata and updates via the Interplanetary Naming System (IPNS), which can complicate workflows. While IPFS has the potential to replicate some of Git's functionalities, additional tooling and interfaces are required to fully match Git's features and scalability.

One approach to managing sensitive application property files across different environments is to encrypt the CID (Content Identifier) returned by IPFS before uploading it to the blockchain and share the decryption key with only the concerned person. This ensures both immutability and security, as the encrypted CID makes it difficult for unauthorized users to access sensitive data, which ensures privacy for crucial files while still using the benefits of decentralized storage.

3) CONTINOUS INTEGRATION AND DELIVERY

Continuous Integration (CI) and Continuous Delivery (CD) are the main components of a DevOps workflow, enabling teams to build, test, and deliver software quickly and reliably. In this blockchain-based framework, CI/CD can be further enhanced by leveraging the immutability and transparency of the blockchain.

To implement CI/CD, our framework is using an open-source tool, Jenkins. Jenkins is one of the most popular automation tools for developers [32]. It is used to automate parts of the software development process, such as building, testing, and releasing code changes.

In the proposed framework, whenever developers update the repository, the code goes through the Jenkins server, which performs a series of automated operations, including

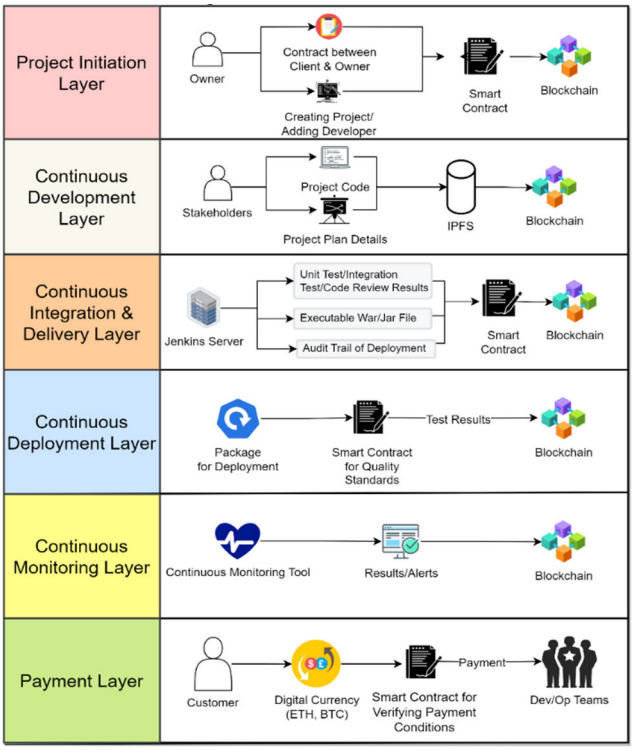


FIGURE 6. Framework architecture layered-wise.

code review, unit testing, integration testing, and building an executable package (e.g., WAR or JAR). Once the package is built, it will be uploaded to a decentralized storage system IPFS. The IPFS hash of the package, along with the results of the code review, unit tests, and integration tests, should be stored on the blockchain. This provides a transparent and immutable record of the file changes, and the testing performed. If there is a code error during code review, unit test, or integration testing, the system generates an alert that notifies all developers to remove the error in the code and update the repo as soon as possible.

To enable continuous delivery, smart contracts are used. A smart contract will be placed that automatically deploys the package to the server or cloud provider when certain conditions are met (e.g., passing all tests). The smart contract can also track the deployment process, ensuring that the package is delivered correctly and providing an audit trail of the delivery.

The information on the executable file that should be stored on the blockchain is shown in Table 4.

After the file is uploaded on the Blockchain, all developers get notified about the file. The complete working of CI/CD is shown in Figure 10.

4) CONTINOUS DEPLOYMENT

Continuous Deployment ensures the automatic release of code changes to the production environment [33].

Smart Contracts play a major role in implementing continuous deployment in our proposed framework. Before

TABLE 4. Information of executable file.

Info	Description
Time	The time when files are uploaded
Date	Date of uploading of file
Name	Name of file
Version	Version of file

deploying the main file, the code has to go through a smart contract specifically created to check quality standards, security requirements, and compliance with regulatory requirements. The smart contract should be already integrated into the CI/CD pipeline to enforce the deployment criteria specified in the contract. This can be done by creating a custom script or plugin that interacts with the contract or by using an existing blockchain integration tool. Once the main file is deployed, the owners, managers, clients, and developers are notified about this change. The status of this implemented functionality changes to ''Deployed''.

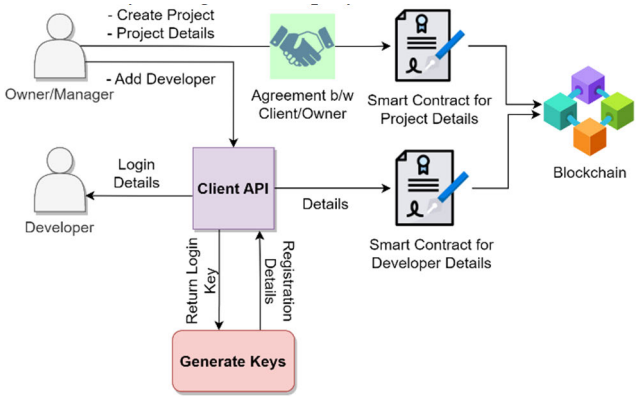


FIGURE 7. Project initialization.

5) CONTINOUS MONITORING

In this phase, all the metrics related to the performance and availability of the software and infrastructure components are recorded on the blockchain with the help of smart contracts. By recording these metrics on the blockchain, they become immutable and secure.

When a problem or alert is detected in the system, it will be recorded on the blockchain with the help of a Smart Contract. In parallel, all the developers who are part of the DevOps team will be notified about it immediately. This notification ensures that the problem can be stored and then resolved as soon as possible, minimizing the impact on end-users.

To monitor the process, any preferred software can be used, such as Nagios, Splunk, Prometheus, etc. However, the software must be compatible with the blockchain framework being used in the distributed DevOps environment. This can be done by creating a custom script or plugin. This ensures that the monitoring data can be recorded on the blockchain and accessed by all the relevant stakeholders securely and transparently.

6) PAYMENTS PHASE

The payment phase is a pivotal component of our framework, seamlessly enabled through the utilization of a Smart Contract. A smart contract can be defined as the algorithmic description of a contractual transaction protocol that is automatically executed based on predefined rules and conditions [34].

The integration of smart contracts with digital wallets can be done by the Web3 API. This API allows DApps to connect to digital wallets, enabling users to interact with smart contracts and participate in decentralized applications. In our framework, the smart contract is triggered either upon the predefined completion of an iteration by the organization or after a specified time interval. Upon successful verification, the smart contract initiates the release of payment, transferring funds from the Customer's Wallet to the Stakeholder's Wallet.

To facilitate this process, every stakeholder involved in the project must possess a Digital Wallet. Payments can be made using various digital currencies such as ETH (Ethereum) or BTC (Bitcoin). Furthermore, stakeholders, including managers, developers, or testers, have the flexibility to convert these digital currencies to their local currencies, such as PKR, USD, or EUR, through cryptocurrency exchanges. The payment phase is illustrated in Figure 11.

FIGURE 8. Data file.

Timely payment by the client is crucial for the uninterrupted progression of the project. The client is obligated to fulfill the payment within the timeframe specified in the smart contract failure to do so results in a holdup of further work. On the contrary, in cases where iteration delays are attributable to developers, penalties are imposed as specified in the smart contract.

The payment requirements, penalties, and other relevant details are encoded in a smart contract, ensuring their immutability. The associated JSON objects for payment, customer's penalty (in case of non-payment), and developer's penalty (in case of iteration delay) are mentioned in Tables ***5***, ***6***, and ***7***, respectively. This integrated approach, governed by a robust smart contract, guarantees transparency, security, and automation in the payment process.

Transactions on blockchains require fees to process. Few steps can be taken to manage the operational costs associated with payments. First, blockchain transaction fees, acquired for storing project details or test results, must be factored into the project budget. These costs can be managed by allocating a portion of the payment to cover such fees. Additionally, smart contract gas fees, which vary depending on network activity, can be mitigated by adopting layer-2 solutions like Polygon. Layer-2 solutions improve transaction processing rates, periods, and fees by minimizing the use of underlying slow and costly blockchains [35]. For decentralized storage via IPFS, only essential data can be stored on the blockchain to optimize costs, while larger files can be handled off-chain. By integrating these strategies, the framework makes sure that operational costs are effectively managed, balancing security and cost efficiency. Table 5 shows the payment file method. Table 6 shows the customer penalty method and Table 7 shows the developer's penalty mechanism.

TABLE 5. File for payment.

Payment
{   "Project Budget": "$1000"   "Payment After 1 Successful Iteration": "$100"   OR   "Payment After 2 Weeks": "$100"   "Payment Due": "Yes" }

TABLE 6. File for customer penalty (in case of no payment).

Customer's Penalty
{   "Overall Project Progress": "50%"   "Payment Due": "Yes"   "Payment Status": "Pending"   "In Case of Non-Payment": "Stop Project's Functions" }

TABLE 7. File for developer's penalty (In case of iteration delay).

Developer's Penalty
{ "Overall Project Progress": "50%"   "Iteration Due Date": "25/05/2023"   "Iteration Status": "Not Complete"   "Penalty": "$5 Deduction" }

D. IMPLEMENTATION CHALLENGES AND LIMITATIONS

While our proposed framework offers innovative solutions to enhance traceability, security, and transparency in distributed DevOps, it's crucial to acknowledge potential challenges and limitations in its implementation. These may include:

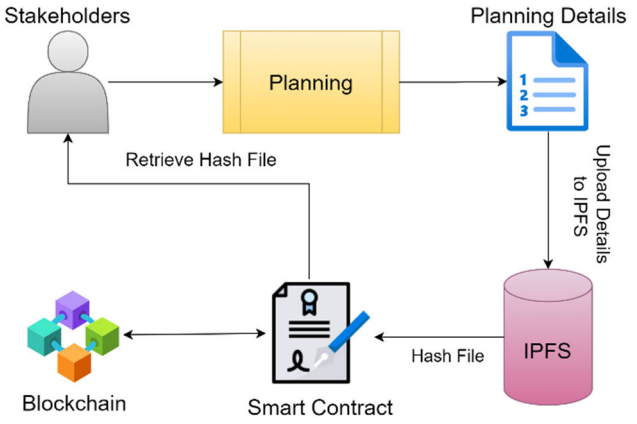


FIGURE 9. Continuous development phase.

1) SCALABILITY CHALLENGES

Despite utilizing IPFS to address the scalability challenge of Blockchain, this framework may still be susceptible to the growing number of transactions and project size. Continuous monitoring and optimization strategies will be vital to ensure sustained scalability, but they can also increase the cost of the overall project.

2) IPFS LIMITATIONS

While IPFS offers functionalities similar to Git, providing decentralized and secure version control, it does not incorporate the complete range of Git's functionalities. To address this, additional tooling and interfaces may be required for full replication of Git's features within the proposed framework.

3) INTEGRATION COMPLEXITY

Integrating DevOps with Blockchain is a relatively new area of research, introducing potential difficulties and, in extreme cases, the possibility of unsuccessful integration. The lack of integration may prevent the framework from fully using the benefits of blockchain technology, thus limiting the scope of improvement. Robust integration strategies and collaboration with experienced professionals may be necessary to overcome this challenge.

4) SMART CONTRACTS

Designing and deploying complex smart contracts that accurately capture the nuances of DevOps processes can be a daunting task. The complex nature of smart contracts increases the risk of errors, making the system challenging to maintain and implement. Implementing rigorous testing protocols and employing standardized smart contract development practices can help mitigate these complexities.

5) USER ADOPTION AND TRAINING

A key implementation challenge for this framework is the introduction of a new system, which requires overcoming a learning curve for team members and clients. Team members and clients may find it challenging to adapt to new

technology; therefore, initiating comprehensive training programs is an essential approach to mitigate this challenge.

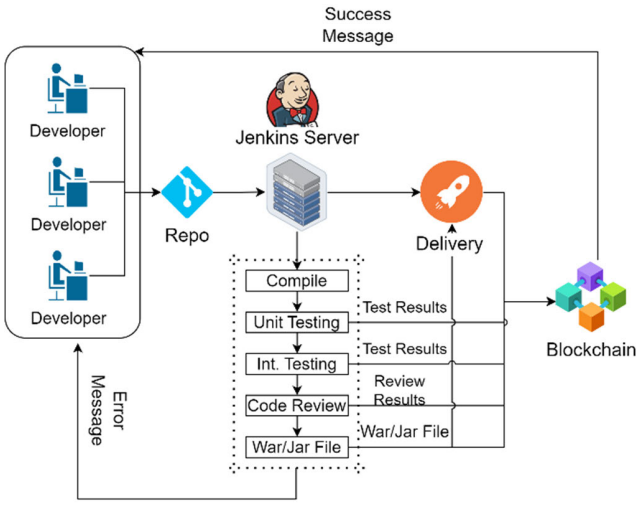


FIGURE 10. Continuous integration and delivery phase.

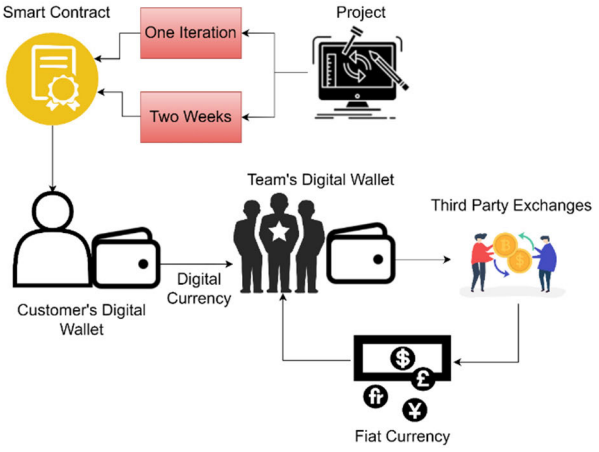


FIGURE 11. Payment process phase.

V. IMPLEMENTATION AND PERFORMANCE

In this section, we have tested the efficiency of the framework to demonstrate its effectiveness in a real-world scenario.

A. PERFORMANCE ASSESSMENT

To test the efficiency and performance of our proposed model, we utilized specific tools to implement and test a blockchain network. We employed Spyder IDE Version 5.4.1 for implementing the blockchain in Python, and Postman Version 10.14.2 was utilized to test the network's performance. Additionally, we have used the Python library Matplotlib for plotting graphs to do graphical representations of our results.

B. PERFORMANCE EVALUATION

In this section, we present the performance results of our model in a real-world scenario. The Postman tool

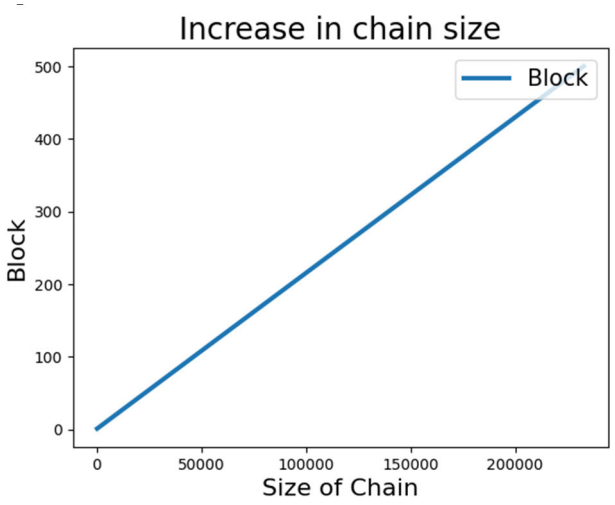


FIGURE 12. Chain size increase.

is responsible for HTTP requests, specifically 'GET' and 'POST', to interact with APIs. Table 8 shows each function's name that is employed to evaluate the framework along with its purpose.

TABLE 8. Functions and their purpose.

Function Name	Purpose
GetChain	To demonstrate the entire Chain.
ConnectNode	To establish connections between nodes.
MineBlock	To successfully mine a block within a blockchain network.
AddTransaction	To include a single transaction within a block.

A total of 500 blocks have been mined by sending HTTP requests. As depicted in Figure 12, the chain size consistently increases with each block mined. On average, the single block size is approximately 465B. Throughout the process, starting from the 1st block and continuing to the 500th block, the chain size has increased from 290B to 232 KB. The size of the blockchain increased with each block primarily due to the addition of new transactions and the block's data structure.

It is essential to highlight that mining each block's latency (measured in milliseconds) appears to be random. There is no visible pattern or significant correlation observed in the latency values. As shown in Figure 13, the latency fluctuates from 12ms to 1359ms as 500 blocks are mined in the blockchain.

This variability in latency suggests that the time taken to mine a block can differ significantly for each instance. Factors such as network congestion, computational resources, and the complexity of the block being mined can contribute to these fluctuations. It is crucial to consider and analyze the latency distribution to gain insights into the overall performance and responsiveness of the blockchain network.

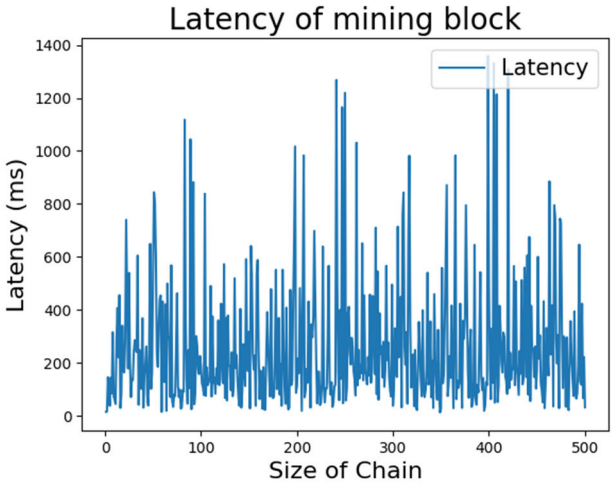


FIGURE 13. Latency in mining the block.

Conclusively, these evaluation results have provided valuable insights into the behaviour of our proposed model in a real-world scenario. The evaluation of our model using specific tools and metrics has allowed us to better understand the efficiency and performance of the blockchain network. Further analysis and optimization can be pursued to enhance the overall reliability and responsiveness of the blockchain system.

C. PRESENTED FRAMEWORK QUALITATIVE AND QUANTITATIVE ANALYSIS

Qualitative

In this subsection, we have presented a comprehensive quantitative comparison between our proposed framework for distributed DevOps and the related work that has been conducted in this domain. We have carefully examined the existing literature and research efforts in the field of blockchain-based software engineering, taking into account the various aspects. By comparing our proposed framework with the related work, we aim to highlight the unique contributions and advantages offered by our approach. Table 1 shows the comparison of our work with related work.

1) BLOCKCHAIN-BASED

The presented framework is based on blockchain, enabling secure, immutable, transparent, and traceable software development.

2) FRAMEWORK

We have introduced a comprehensive framework that addresses the shortcomings of related work. The framework demonstrates the successful execution of each step in DevOps.

3) DISTRIBUTED DEVOPS

This study focuses specifically on distributed DevOps, a relatively new software development technique that involves teams located in different locations. We have successfully addressed the challenges of trust and transparency that are often encountered in distributed settings.

4) PAYMENT SOLUTIONS

This framework also includes a payment solution, ensuring seamless and uninterrupted payment transactions after each process or iteration.

Quantitative

In this subsection, a quantitative analysis has been conducted comparing our extracted results with well-known blockchains such as Bitcoin, Ethereum, Cardano, etc. The results of this quantitative analysis highlight the efficiency of our framework and offer a clear understanding of the framework's advantage in terms of speed. We are utilizing Block Time as a parameter for comparison, where Block Time refers to the duration required to successfully mine a block in the blockchain.

TABLE 9. Block time comparison.

Reference	Blockchain Name	Average Block Time
[36]	Bitcoin	10min
[37]	Ethereum	12s
[38]	Cardano	20s
[39]	Polkadot	6s
This paper	Proposed Framework	0.25s

Table 9 shows the Block Time comparison of our experiment with Bitcoin, Ethereum, Cardano, and Polkadot blockchains, which are well-known blockchains to this date. The average Block Time of 0.25 seconds indicates that our framework outperforms other blockchains.

VI. DISCUSSION

In this section, the performance of our proposed framework is formally discussed. This framework is designed to facilitate the successful development of software projects with Distributed DevOps while maintaining decentralization, security, traceability, transparency, and coordination. The proposed framework addresses potential challenges and problems associated with Distributed DevOps, focusing on security, transparency, and traceability while enhancing coordination and collaboration.

The performance results along with the proposed framework demonstrate that combining Blockchain technology with Distributed DevOps can resolve many problems that may lead to project failures, delays, and financial losses.

The integration of Blockchain with Distributed DevOps offers several benefits, including:

- Increased project decentralization.
- Enhanced collaboration between development teams and operations teams in a secure and transparent environment because the blockchain keeps a record of transactions and is capable of preventing 51% of attacks.
- Elimination of payment-related issues when teams are distributed across different locations.
- Improved client satisfaction is achieved through increased transparency provided by Blockchain, leveraging its ability to track and trace information.
- Improved efficiency with an average time of 0.25 seconds for adding one block on the blockchain which is far better compared to renowned blockchains like Bitcoin, Ethereum, Cardano, and Polkadot.
- Enhanced security for all stakeholders, enabling them to work in a secure environment.
- The utilization of Blockchain technology is a useful tool for tracking work progress and maintaining records.
- Decreased risk of conflicts and errors by implementing Blockchain with distributed DevOps, thereby facilitating smoother collaboration on complex projects.

Overall, our research work proves that implementing Blockchain technology with Distributed DevOps can increase security, transparency, and traceability in software development. Although previous studies have been conducted in this area, none have specifically addressed Distributed DevOps. Our study fills this gap by presenting an efficient approach to software development with CI/CD when teams are located worldwide.

Despite the benefits of utilizing blockchain in the proposed framework, it is important to acknowledge the drawbacks associated with this technology. One such drawback could be technology failure. If any failure occurs, the responsibility will depend upon the specific blockchain implementation. For public blockchains, responsibility is decentralized and shared among participants of a network. In contrast, in private or consortium blockchain, the responsibility lies with the organization or entity managing the network. Also, in case of Smart Contract vulnerabilities, the development team or governance body is accountable. Another drawback is the potential for slower data processing speeds compared to traditional databases. This stems from the decentralized and consensus-based nature of blockchain, which introduces additional steps and computational overhead that can impact data throughput. While DevOps processes can be streamlined overall, the time required for data uploading or loading may increase due to these factors. Another key limitation is the implementation cost of the proposed framework. The complexity of blockchain technology and the associated infrastructure requirements can lead to significant expenses [40], particularly for organizations operating on limited budgets. This cost barrier may hinder the adoption of the framework, especially for companies with resource constraints.

However, the potential benefits, such as increased security, transparency, trust, and traceability, could result in substantial gains over time. Addressing these limitations will require further research and development focused on optimizing blockchain performance and reducing implementation costs. By overcoming these challenges, the proposed framework can be made more scalable and accessible to a wider range of organizations.

Moreover, in conclusion, the findings of this research highlight the significant potential of Blockchain, a major technological advancement, to revolutionize software development.

VII. CONCLUSION AND FUTURE WORK

The current distributed DevOps for software development needs more security, data protection, privacy, transparency, and traceability. These shortcomings are why the operations and development teams need help working together properly in a distributed environment. That is why this proposed model will allow the collaboration of development and operation teams more smoothly. The framework utilizes smart contracts and a decentralized architecture to enable secure and efficient collaboration among distributed teams in the development and operations of software systems. We discussed the benefits of using blockchain technology in DevOps, such as increased transparency, enhanced traceability, improved collaboration, and increased efficiency. We also presented the performance results, which demonstrated the effectiveness of the proposed framework in a real-world scenario.

The proposed framework offers a promising solution for addressing the challenges of Distributed DevOps, and there is a lot of potential for further research and development. For example, one potential research area is to explore the integration of this framework with other emerging technologies, such as Artificial Intelligence and Machine Learning, to automate certain tasks in Distributed DevOps, such as automating testing, improving resource management, or predicting potential risks. Another research direction could involve implementing this framework in different organizations and regions to assess its feasibility in real-world scenarios. Understanding the practical pros and cons through implementation can unlock additional research directions. Overall, the proposed framework provides a promising solution for enhancing trust, traceability, security, and transparency of Distributed DevOps.

Several directions for future work can be pursued based on the proposed framework. One potential area of research is to investigate this framework with other emerging technologies, such as Artificial Intelligence and Machine Learning, to enhance its capabilities and automation. Additionally, it will be interesting to study the real-world adoption and usage of the framework by different organizations and industries. Furthermore, there is also a scope to explore the regulatory compliance aspect of the framework and its adoption in different geographical regions. Inclusively, the proposed framework provides a promising solution for addressing the challenges of Distributed DevOps, and there

is much potential for further research and development in this area.

REFERENCES

[1] L. Leite, C. Rocha, F. Kon, D. Milojicic, and P. Meirelles, "A survey of DevOps concepts and challenges," *ACM Comput. Surv.*, vol. 52, no. 6, pp. 127:1–127:35, Nov. 2019, doi: 10.1145/3359981.

[2] M. Gall and F. Pigni, "Taking DevOps mainstream: A critical review and conceptual framework," *Eur. J. Inf. Syst.*, vol. 31, no. 5, pp. 548–567, Sep. 2022, doi: 10.1080/0960085x.2021.1997100.

[3] E. Diel, S. Marczak, and D. S. Cruzes, "Communication challenges and strategies in distributed DevOps," in *Proc. IEEE 11th Int. Conf. Global Softw. Eng. (ICGSE)*, Aug. 2016, pp. 24–28, doi: 10.1109/ICGSE.2016.28.

[4] Deloitte Luxembourg. *DevOps in a Distributed World and New Ways of Working | Deloitte Luxembourg | Blog*. Accessed: Nov. 21, 2023. [Online]. Available: https://www2.deloitte.com/lu/en/pages/risk/articles/devops-in-a-distributed-world-and-new-ways-of-working.html

[5] (2016). *Kharnagy, English: Illustration Showing Stages in a DevOps Toolchain*. Accessed: Nov. 20, 2023. [Online]. Available: https://commons.wikimedia.org/wiki/File

[6] S. Shrivastava and H. Date, "Distributed agile software development: A review," *J. Comput. Sci. Eng.*, vol. 1, no. 1, pp. 10–17, Jun. 2010.

[7] A. A. Khan and M. Shameem, "Multicriteria decision-making taxonomy for DevOps challenging factors using analytical hierarchy process," *J. Softw., Evol. Process*, vol. 32, no. 10, Oct. 2020, Art. no. e2263, doi: 10.1002/smr.2263.

[8] A. W. Khan, S. Zaib, F. Khan, I. Tarimer, J. T. Seo, and J. Shin, "Analyzing and evaluating critical cyber security challenges faced by vendor organizations in software development: SLR based approach," *IEEE Access*, vol. 10, pp. 65044–65054, 2022, doi: 10.1109/ACCESS.2022.3179822.

[9] M. J. Hossain Faruk, M. Tasnim, H. Shahriar, M. Valero, A. Rahman, and F. Wu, "Investigating novel approaches to defend software supply chain attacks," in *Proc. IEEE Int. Symp. Softw. Rel. Eng. Workshops (ISSREW)*, Oct. 2022, pp. 283–288, doi: 10.1109/ISSREW55968.2022.00081.

[10] S. Maro, J.-P. Steghöfer, P. Bozzelli, and H. Muccini, "TracIMo: A traceability introduction methodology and its evaluation in an agile development team," *Requirements Eng.*, vol. 27, no. 1, pp. 53–81, Mar. 2022, doi: 10.1007/s00766-021-00361-5.

[11] A. G. Gad, D. T. Mosa, L. Abualigah, and A. A. Abohany, "Emerging trends in blockchain technology and applications: A review and outlook," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 9, pp. 6719–6742, Oct. 2022, doi: 10.1016/j.jksuci.2022.03.007.

[12] Y. Lu, "The blockchain: State-of-the-art and research challenges," *J. Ind. Inf. Integr.*, vol. 15, pp. 80–90, Sep. 2019, doi: 10.1016/j.jii.2019.04.002.

[13] A. D. John William, S. Rajendran, P. Pranam, Y. Berry, A. Sreedharan, J. Gul, and A. Paul, "Blockchain technologies: Smart contracts for consumer electronics data sharing and secure payment," *Electronics*, vol. 12, no. 1, p. 208, Jan. 2023, doi: 10.3390/electronics12010208.

[14] S. Bankar and D. Shah, "Blockchain based framework for software development using DevOps," in *Proc. 4th Biennial Int. Conf. Nascent Technol. Eng. (ICNTE)*, Jan. 2021, pp. 1–6, doi: 10.1109/ICNTE51185.2021.9487760.

[15] M. A. Akbar, S. Mahmood, and D. Siemon, "Toward effective and efficient DevOps using blockchain," in *Proc. Int. Conf. Eval. Assessment Softw. Eng.*, New York, NY, USA, Jun. 2022, pp. 421–427, doi: 10.1145/3530019.3531344.

[16] F. Tariq and R. Colomo-Palacios, "Use of blockchain smart contracts in software engineering: A systematic mapping," in *Computational Science and Its Applications-ICCSA* (Lecture Notes in Computer Science), S. Misra, O. Gervasi, B. Murgante, E. Stankova, V. Korkhov, C. Torre, A. M. A. C. Rocha, D. Taniar, B. O. Apduhan, and E. Tarantino, Eds., Cham, Switzerland: Springer, 2019, pp. 327–337, doi: 10.1007/978-3-030-24308-1_27.

[17] M. J. H. Faruk, S. Subramanian, H. Shahriar, M. Valero, X. Li, and M. Tasnim, "Software engineering process and methodology in blockchain-oriented software development: A systematic study," in *Proc. IEEE/ACIS 20th Int. Conf. Softw. Eng. Res., Manage. Appl. (SERA)*, May 2022, pp. 120–127, doi: 10.1109/SERA54885.2022.9806817.

[18] N. S. G. R. Ramakrishna, "Blockchain in agile software development," in *ICT for Competitive Strategies*. Boca Raton, FL, USA: CRC Press, 2020.

[19] P. Nayaka Sheetakallu Krishnaiah, D. L. Narayan, and K. Sutradhar, "A survey on secure metadata of agile software development process using blockchain technology," *Secur. Privacy*, vol. 7, no. 2, Mar. 2024, Art. no. e342, doi: 10.1002/spy2.342.

[20] A. Al-Nakeeb, M. E. Khatib, S. AlHarmoodi, M. Salami, H. Al Shehhi, A. Al Naqbi, M. Al Nuaimi, and H. M. Alzoubi, "Digital transformation and disruptive technologies: Effect of cloud computing and devops on managing projects," in *Technology Innovation for Business Intelligence and Analytics (TIBIA): Techniques and Practices for Business Intelligence Innovation*, H. M. Alzoubi, M. T. Alshurideh, and S. Vasudevan, Eds., Cham, Switzerland: Springer, 2024, pp. 39–62, doi: 10.1007/978-3-031-55221-2_3.

[21] S. Terzi and I. Stamelos, "Architectural solutions for improving transparency, data quality, and security in eHealth systems by designing and adding blockchain modules, while maintaining interoperability: The eHDSI network case," *Health Technol.*, vol. 14, no. 3, pp. 451–462, May 2024, doi: 10.1007/s12553-024-00833-y.

[22] J. N. Qureshi and M. S. Farooq, "ChainAgile: A framework for the improvement of scrum agile distributed software development based on blockchain," *PLoS ONE*, vol. 19, no. 3, Mar. 2024, Art. no. e0299324, doi: 10.1371/journal.pone.0299324.

[23] M. S. Farooq, Z. Kalim, J. N. Qureshi, S. Rasheed, and A. Abid, "A blockchain-based framework for distributed agile software development," *IEEE Access*, vol. 10, pp. 17977–17995, 2022, doi: 10.1109/ACCESS.2022.3146953.

[24] S. Vimal and S. K. Srivatsa, "A new cluster P2P file sharing system based on IPFS and blockchain technology," *J. Ambient Intell. Humanized Comput.*, vol. 10, pp. 1–7, Sep. 2019, doi: 10.1007/s12652-019-01453-5.

[25] A. Gómez, C. Joubert, and J. Cabot, "Blockchain technologies in the design and operation of cyber-physical systems," in *Digital Transformation: Core Technologies and Emerging Topics From a Computer Science Perspective*, B. Vogel-Heuser and M. Wimmer, Eds., Berlin, Germany: Springer, 2023, pp. 223–243, doi: 10.1007/978-3-662-65004-2_9.

[26] G. A. Oliva, A. E. Hassan, and Z. M. Jiang, "An exploratory study of smart contracts in the Ethereum blockchain platform," *Empirical Softw. Eng.*, vol. 25, no. 3, pp. 1864–1904, May 2020, doi: 10.1007/s10664-019-09796-5.

[27] M. Andoni, V. Robu, D. Flynn, S. Abram, D. Geach, D. Jenkins, P. McCallum, and A. Peacock, "Blockchain technology in the energy sector: A systematic review of challenges and opportunities," *Renew. Sustain. Energy Rev.*, vol. 100, pp. 143–174, Feb. 2019, doi: 10.1016/j.rser.2018.10.014.

[28] I. K. Moutsatsos, I. Hossain, C. Agarinis, F. Harbinski, Y. Abraham, L. Dobler, X. Zhang, C. J. Wilson, J. L. Jenkins, N. Holway, J. Tallarico, and C. N. Parker, "Jenkins-CI, an open-source continuous integration system, as a scientific data and image-processing platform," *SLAS Discovery*, vol. 22, no. 3, pp. 238–249, Mar. 2017, doi: 10.1177/1087057116679993.

[29] D. Yang, C. Long, H. Xu, and S. Peng, "A review on scalability of blockchain," in *Proc. 2nd Int. Conf. Blockchain Technol.*, New York, NY, USA, May 2020, pp. 1–6, doi: 10.1145/3390566.3391665.

[30] D. Spinellis, "Git," *IEEE Softw.*, vol. 29, no. 3, pp. 100–101, May 2012, doi: 10.1109/MS.2012.61.

[31] N. Nizamuddin, K. Salah, M. Ajmal Azad, J. Arshad, and M. H. Rehman, "Decentralized document version control using Ethereum blockchain and IPFS," *Comput. Electr. Eng.*, vol. 76, pp. 183–197, Jun. 2019, doi: 10.1016/j.compeleceng.2019.03.014.

[32] J. Hembrink and P.-G. Stenberg. (2013). *Continuous Integration With Jenkins Coaching of Programming Teams (EDA 270)*. Accessed: Nov. 21, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Continuous-Integration-with-Jenkins-Coaching-of-(-)-Hembrink-Stenberg/49ab2ff1a056c2d7ee114a7ba20fa0372863a56e

[33] M. Yashwanth, S. Krishna, and S. K. Gawre, "MLOps for enhancing the accuracy of machine learning models using DevOps, continuous integration, and continuous deployment," *Res. Rep. Comput. Sci.*, vol. 2, no. 3, pp. 97–103, Jun. 2023, doi: 10.37256/rrcs.2320232644.

[34] K. Hu, J. Zhu, Y. Ding, X. Bai, and J. Huang, "Smart contract engineering," *Electronics*, vol. 9, no. 12, p. 2042, Dec. 2020, doi: 10.3390/electronics9122042.

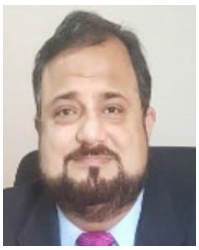
[35] A. Gangwal, H. R. Gangavalli, and A. Thirupathi, "A survey of layer-two blockchain protocols," *J. Netw. Comput. Appl.*, vol. 209, Jan. 2023, Art. no. 103539, doi: 10.1016/j.jnca.2022.103539.

[36] *Vocabulary-Bitcoin*. Accessed: Nov. 21, 2023. [Online]. Available: https://bitcoin.org/en/vocabulary

[37] Ethereum.org. *Blocks*. Accessed: Nov. 21, 2023. [Online]. Available: https://ethereum.org
[38] *Time Handling on Cardano*. Accessed: Nov. 23, 2023. [Online]. Available: https://docs.cardano.org/explore-cardano/time/
[39] *Polkadot Parameters? Polkadot Wiki*. Accessed: Nov. 23, 2023. [Online]. Available: https://wiki.polkadot.network/docs/maintain-polkadot-parameters
[40] K. W. Prewett, G. L. Prescott, and K. Phillips, "Blockchain adoption is inevitable—Barriers and risks remain," *J. Corporate Accounting Finance*, vol. 31, no. 2, pp. 21–28, Apr. 2020, doi: 10.1002/jcaf.22415.

JUNAID NASIR QURESHI received the M.Phil. degree in computer science from the University of Management and Technology, Lahore, where he is currently pursuing the Ph.D. degree.

He is an Assistant Professor with Bahria University, Lahore Campus. He has almost 15 years of experience in industry and academia. He has published many peer-reviewed international journals and conference papers. His research interests include agile software development, databases, blockchain, and education.



MUHAMMAD SHOAIB FAROOQ was an Affiliate Member of George Mason University, USA. He is currently a Professor of artificial intelligence with the University of Management and Technology, Lahore. He possesses more than 28 years of teaching experience in the field of computer science. He has published many peer-reviewed international journals and conference papers. His research interests include the theory of programming languages, big data, the IoT, the Internet of Vehicles, machine learning, blockchain, and education.

USMAN ALI received the M.Phil. degree in software engineering from the University of Management and Technology, Lahore.

He is currently a Lecturer with the University of Management and Technology. He has published many peer-reviewed international journals and conference papers. His research interests include artificial intelligence, machine learning, and deep learning.

ADEL KHELIFI (Senior Member, IEEE) received the Ph.D. degree from the Engineering School of High Technology, Canada, in 2005.

He is currently an Associate Professor with Abu Dhabi University. Previously, he was the Dean of computer information technology with American University in the Emirates, Dubai, United Arab Emirates. He has extensive knowledge and experience. He is driving the open-source software paradigm in the region. He has an impressive career, having worked as a Lecturer with the Engineering School of Technology, Canada; the United Nations MSF, Canada; the Ministry of Citizenship and Immigration, Canada; and the Ministry of Finance, Tunisia. He is a Canadian ISO Member of Software Engineering. He is ABET PEV and is passionate about archaeology.

ZABIHULLAH ATAL received the master's degree in information technology from VU University, Pakistan. He is currently an Assistant Professor with the Computer Science Department, Kardan University, Afghanistan. His research interests include computer networks and information security, the IoT, machine, and neural networks. He is also interested in smart grid applications and technologies, cloud computing, distributed systems, and blockchain.