scientific reports

Check for updates

OPEN

Client engagement solution for post implementation issues in software industry using blockchain

Muhammad Shoaib Farooq[1], Khurram Irshad[1], Danish Riaz[2], Nagwan Abdel Samee[3], Ernesto Bautista Thompson[4,5,6], Daniel Gavilanes Aray[4,7,8] & Imran Ashraf[9]✉

In the rapidly advanced and evolving information technology industry, adequate client engagement plays a critical role as it is very important to understand the client's concerns, and requirements, have the records, authorizations, and go-ahead of previously agreed requirements, and provide the feasible solution accordingly. Previously multiple solutions have been proposed to enhance the efficiency of client engagement, but they lack traceability, trust, transparency, and conflict in agreements of previous contracts. Due to the lack of these shortcomings, the client requirement is getting delayed which is causing client escalations, integrity issues, project failure, and penalties. In this study, we proposed the UniferCollab framework to overcome the issues of collaboration between various teams, transparency, the record of client authorizations, and the go-ahead on previous developments by implementing blockchain technology. We store the data on the permissible network in the proposed approach. It allows us to compile all the requirements and information shared by clients on permissible blockchain to secure a large amount of data which enhances the traceability of all the requirements. All the authorizations from the client generate push notifications for any changes in their current system executed through smart contracts. It removes the ambiguity between various development teams if the client has only shared the requirement with one team. The data is stored in the decentralized network from where information is gathered which resolves the traceability, transparency, and trust issues. Lastly, evaluations involved a total of 800 hypertext transfer protocol (HTTP) requests tested using Postman with blockchain block sizes ranging from 0.568 KB to 550 KB and an average size increase of 280 KB was observed as new blocks were added. The longest chain in the network was observed during 800 repetitions of blockchain operations. Latency analysis revealed that delays in processing HTTP requests were influenced by decentralized node processing, local machine response times, and internet bandwidth through various experiments. Results show that the proposed framework resolves all client engagement issues in implementation between all stakeholders which enhances trust, and transparency improves client experience and helps us manage disputes effectively.

Keywords Client engagement, Blockchain, Traceability, Decentralization, Requirement traceability, Post implementation

In the rapidly evolving field of information technology (IT), effective client engagement has become a critical factor in ensuring the successful delivery of software projects. Client engagement encompasses a series of interactions between software providers and clients throughout the project lifecycle, from initial requirement gathering to post-implementation support. The ability to accurately document, track, and validate client requirements is essential in mitigating issues such as delayed project deliveries, miscommunication, contractual disputes, and loss of trust between stakeholders.

[1]Department of Computer Science, School of System and Technology, University of Management and Technology, Lahore 54000, Pakistan. [2]Department of Computing, Staffordshire University, Stoke-on-Trent ST4 2DE, UK. [3]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, 11671 Riyadh, Saudi Arabia. [4]Universidad Europea del Atlantico, Isabel Torres 21, 39011 Santander, Spain. [5]Universidad Internacional Iberoamericana, 24560 Campeche, Mexico. [6]Universidad Internacional Iberoamericana Arecibo, Puerto Rico 00613, USA. [7]Universidade Internacional do Cuanza, Cuito, Angola. [8]Universidad de La Romana, La Romana, República Dominicana. [9]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Republic of Korea. ✉email: imranashraf92@gmail.com

Client engagement is the process of building and maintaining productive and positive relationships between the clients and a software company[1]. It includes all the communication and the interaction between both parties in the entire journey of the customer, from its first initial contact with respect to the services to the post-sale and beyond. The effectiveness of client engagement in the field of the software industry includes understanding the requirements, setting the right expectations, and providing them with a service of high quality. It includes efficient communication, understanding the requirements, and providing adequate solutions[2].

Efficient client engagement plays a critical role in today's adapting and evolving IT world to understand their problems and provide adequate solutions. It increases the satisfaction of the customer while engaging and understanding the client's problems and providing them with personalized solutions with high-quality support[3]. Adequate engagement leads to positive referrals and increases the retention rate of customers. It builds a strong relationship and with the relationship, it increases the loyalty of customers, extends contracts, and develops trust. It provides a better reputation by providing excellent support, increases the goodwill of the company, and brand recognition. Companies having efficient client engagement have a competitive edge over competitors by understanding the client requirements and providing them with the high-quality of customer service. Regularly engaging with the clients, opens other areas of opportunity where provided solutions increase revenue through cross-selling[4].

Adequate engagement is one of the most important factors in driving success for the companies, it increases customer retention as it is more costly to retain the existing customer than to acquire a new lead[5]. The clients who are satisfied with the services spread positive feedback and are likely to recommend the services of the company which increases opportunity. Client engagement sets the right expectations and understands the business requirement to propose solutions. Projects with respect to the Information technology sector are a challenging task but nowadays people in organizations are not aligned to face its challenging aspects[6]. It helps to understand the new trends in the market and to understand requirements and innovation for new features and development of products. It helps to drive the new business in the organization, building strong relationships and eventually achieving the success of business[7].

With respect to the previous research, it has been revealed that various information technology (IT) projects have failed not due to technical issues but rather because of in-efficient communication and client management issues[8]. Information and communication technology (ICT) firms now have to be globally competitive along with a more project-driven approach and the projects are normally divided among various multi-client-stakeholders globally. In the context of this study, the term multi-client-stakeholders is defined broadly as a group of individuals who are from divergent backgrounds and engaged in multiple projects on an ongoing basis from various organizations that have different sets of responsibilities and different interests globally[9]. To come from different cultural backgrounds these client multi-stakeholders are engaging with multiple clients as internal or external holders sharing a common purpose and communicating globally by using various sets of technologies and tools. It includes teams of project delivery, design, and development, testing, customer interface, third-party vendors, sponsors of clients, business people, management, etc.[10].

The projects now have become a critical way to structure the work in companies[11]. The governance and the relationship are acted as quasi-moderator[12]. The issues described in the last decades assumed that the entities available for decision-making compromise between various character objectives[13]. Adequate client management is vital for the success of the project and any changes which are made to the effect have to be understood carefully[14,15]. Client management often goes through the issues faced in the identification of the key stakeholders and understanding of their impact and relationship[16]. The environmental, and economic issues have increased drastically and organizations are facing huge impacts in order to change, manage, and innovate[17]. There are various methodologies in the management of software life cycles for building, designing, and maintaining the systems. The most common issue in the failure of the projects is the lack of documentation and its traceability[18,19].

In the rapidly advancing field of information technology, effective client engagement in the software industry plays a pivotal role in delivering successful projects. Effective client engagement encompasses understanding client requirements, managing expectations, and providing high-quality solutions throughout the entire project lifecycle. Despite existing solutions to enhance client engagement, several critical issues remain unaddressed, such as lack of traceability, transparency, security, and real-time communication among stakeholders. These issues often lead to delayed requirements, conflicts in agreements, escalations, project failures, and penalties.

Despite the growing reliance on digital collaboration tools, existing solutions often fail to address core issues of traceability, transparency, security, and real-time communication. Many projects suffer from inefficient documentation practices, leading to frequent escalations, conflicts in contract agreements, and failure to meet evolving client expectations. Traditional centralized systems lack the robustness required for secure, decentralized, and tamper-proof client engagement tracking, resulting in discrepancies across various teams involved in the software development process.

To address these challenges, we propose UnifierCollab, a blockchain-powered client engagement framework that enhances transparency, security, and traceability in post-implementation processes. By leveraging Hyperledger blockchain technology, smart contracts, and decentralized applications (DApps), the proposed framework ensures secure, immutable record-keeping of client communications, automated execution of contractual agreements, and seamless collaboration among distributed teams. The framework's layered architecture is designed to enhance stakeholder coordination, track project milestones in real-time, and provide a structured dispute-resolution mechanism.

This study contributes to the field of blockchain-based enterprise solutions by addressing key limitations in existing client engagement systems. The proposed framework enhances data security, requirement traceability, and automated contract execution, making it a viable solution for enterprises seeking robust post-implementation engagement mechanisms. The subsequent sections discuss related research, technical implementation details, performance analysis, and future directions for scalability and adoption.

We propose a blockchain framework United collaboration, called UnifierCollab, which addresses and resolves all the issues faced in engaging with clients after post-implementation. This framework embeds a Hyperledger blockchain with a permissible and authorized network for security enhancement purposes. It consists of a user-friendly interface and decentralized (DApp). The UnifierCollab comprises six layers.

- Agreement Layer,
- Implementation Plan Layer,
- Integrated Communication Plan Layer,
- Strategic and Prioritization,
- Deliver and Testing,
- Testing,
- Payment

Every layer consists of the wall and is activated once the previous layer is completed by the client. It also consists of smart contracts for automatic updating of new product pricing and features and notification of changes on the production programs of clients. Network compliance updates, audit reports, payment card industry (PCI) compliance certificates, Soc1 reports, requirements shared with any internal teams, and scope of works. As we know there are teams working across the globe, so due to the geographical distance it becomes difficult to coordinate this it has a traceability of all the requirements that clients have shared with various internal teams in regard to the post-implementation. We propose the UnifierCollab framework for resolving client engagement challenges in post-implementations by implementing blockchain. We deployed a user-friendly interface and decentralized (DApp), used smart contracts automatic notification updates, changes, internal and external collaboration, and tracking and proof of work (PoW). Along with this, we resolve the scalability challenge by the implementation of JSON objects in blockchain. The novelty of this research lies in the fact that we use a user-friendly permissible and authorized framework, so the confidential information is shared in a secure manner and has trust between the customer and the service or product provider.

This study proposes a novel framework called UnifierCollab to resolve the persistent problems in client engagement by leveraging blockchain technology. The UnifierCollab framework is designed to address key limitations in existing systems and provide traceability, transparency, security, and efficient collaboration in the post-implementation phase. The UnifierCollab framework is built upon a permissioned blockchain (using Hyperledger), enabling decentralized management of client communications, smart contract-based automation, and real-time updates on project requirements.

Objectives of the UnifierCollab framework

The primary objectives of the UnifierCollab framework are as follows:

1. Enhance traceability of all client requirements, approvals, and changes through a decentralized ledger.
2. Improve transparency by providing a clear, immutable record of communications and project progress, accessible to all stakeholders.
3. Increase security of sensitive client data through a permissible blockchain network, where only authorized users can access the information.
4. Automate contract execution and payment processing through smart contracts, ensuring timely updates and reducing human errors.
5. Resolve collaboration issues among geographically distributed teams by providing a unified platform for communication, tracking, and reporting.

Novelty of UnifierCollab

UnifierCollab introduces several innovations in the field of client engagement and project management:

1. *Blockchain-based approach for post-implementation collaboration*: While existing frameworks primarily focus on pre-implementation or early stages of client engagement, UnifierCollab uniquely leverages blockchain to address post-implementation issues, offering continuous traceability and security throughout the project lifecycle.
2. *Permissioned network for security*: Unlike traditional public blockchains, UnifierCollab uses a permissioned blockchain model, ensuring that only authorized stakeholders have access to sensitive data, making it more suitable for enterprise use.
3. *Smart contracts for automatic updates*: UnifierCollab integrates smart contracts to automate the execution of updates, pricing changes, and notifications of any project developments, reducing delays and misunderstandings between teams and clients.
4. *Cross-team collaboration and dispute management*: The framework facilitates cross-functional communication, resolving ambiguity when different teams receive different requirements from clients. It also offers dispute resolution through immutable blockchain records that eliminate conflicts.

Gaps in prior research

Prior research has primarily focused on addressing communication gaps and collaboration between teams, yet several significant gaps remain:

1. *Lack of traceability in project requirements*: Many existing frameworks do not offer continuous traceability, which leads to miscommunication and delays when changes occur. UnifierCollab fills this gap by using a blockchain to create a transparent record of all project interactions.

2. *Inadequate security and access control*: Traditional project management tools often fail to provide sufficient security for sensitive data. UnifierCollab leverages a permissioned blockchain to ensure that only authorized parties can access confidential client information, addressing this critical gap.
3. *Limited scalability and automation*: Existing solutions often fail to scale efficiently and automate repetitive tasks. UnifierCollab introduces smart contracts and a decentralized network to offer greater scalability and automation in managing client engagement.
4. *Lack of effective post-implementation engagement*: Most prior solutions focus on pre-implementation or early stages of the project lifecycle. UnifierCollab extends client engagement to the post-implementation phase, ensuring continuous communication, updates, and traceability.

The remaining paper is as follows. Section "Related work" consists of the related previous research and section "Preliminaries" involves the preliminaries. The proposed framework is mentioned in section "Proposed framework". Then, section "Performance and results" consists of the performance analysis of results. In the end, section "Conclusion and future work" explains the conclusion and the future work.

Related work

Many researchers have proposed different tools and frameworks to overcome the post-implementation challenges faced by clients and development teams. A lot of them solved and resolved the issues related to collaboration, coordination, and communication but still, issues regarding security, transparency, and trust exist.

Siakas and Lampropoula[19] identified and analyzed the issues in regard to communication and social media being used enormously. They have observed and explained that the issues related to communication, differences of culture, and differences in time zone are the main issues in coordination and collaboration between teams but the social media being used in it only resolves the issues related to the communication and coordination. In addition, this research does not resolve issues such as traceability, trust, and security.

Kukreia[20,21] also explored and proposed the framework of Winbook, it is very related to the social website which basically is used for integration of the features that are very user friendly like posting comments, pictures, and content by using the labels-color-coded. This feature of Winbook also resolves most of the issues of the collaboration between various teams but still, it lacks traceability and does not consist of blockchain.

Jahangeer[22] proposed the technology in which the flow of information in the information technology is presented along with its Internet of Things (IoT) applications and the challenges that were involved in its integration. After discussing the main challenges, an IoT nodes network was established which discussed and resolved the issues in security and privacy but lacks in user-friendly interface and permissible network. Huanhuan Feng[10] explained the challenges faced in the blockchain on food traceability and its management. The results of that study provide a detailed review of how we can enhance the quality of food while implementing the blockchain and collaborating with various stakeholders.

Elmogy[18] studied wide information in regard to blockchain in regards to the Internet of Things (IoT) which are being controlled by nodes. A method is proposed that reveals the system responsible for exhausting of blockchain. However, it was only established in the environment which is in test-bud. The studies[23,24] proposed agile development in regard to the framework related to cloud computing (ADCC). The main purpose is to integrate cloud computing with agile development. However, the results showed and proved that the development of this framework in cloud computing improves the communication gaps between multiple teams and it is also scalable but the challenges in regards to the data security and its privacy, no permissible network, and lack of trust still persist. It also lacked the blockchain to overcome the security issues.

Vistro et al.[25] proposed a framework to ensure data integrity, secure access control, and patient-centric management of medical records. By integrating smart contracts, the system automates permission management, enabling patients to grant or revoke access to their records seamlessly. It also incorporates encryption and hashing mechanisms to safeguard sensitive information while maintaining transparency and traceability of all transactions. This blockchain-based approach promotes interoperability among healthcare providers, fostering efficient collaboration and enhancing trust in the system. The article concludes with a discussion of the potential benefits, limitations, and future directions for deploying blockchain in healthcare, emphasizing its capacity to transform the management of patient records while ensuring data security and privacy.

Iqra et al.[26] introduce an innovative framework aimed at enhancing the productivity and morale of software teams using gamification techniques. The TechMark framework integrates game mechanics, such as points, badges, leaderboards, and rewards, into IT organizational processes to foster collaboration, skill development, and sustained engagement among team members. By mapping gamified elements to key performance indicators (KPIs), the framework aligns organizational goals with individual and team achievements. It promotes transparency in performance tracking, encourages healthy competition, and cultivates a sense of accomplishment. Additionally, the framework supports personalized learning paths, enabling team members to upskill effectively while staying motivated. The study also discusses implementation strategies, emphasizing the importance of tailoring gamification to the organization's culture and objectives. Case studies or pilot implementations are presented to demonstrate the framework's effectiveness in improving team dynamics, enhancing motivation, and driving better project outcomes. Finally, the study highlights potential challenges, such as over-competition and gamification fatigue, and suggests mitigation strategies to ensure sustainable engagement.

Farooq et al.[27] introduce a blockchain-acquainted SRE model, highlighting how blockchain can enhance software requirements' elicitation, documentation, validation, and management. Key opportunities include improved stakeholder collaboration, enhanced trust in requirement processes, and better version control and accountability. The study also discusses the potential for smart contracts to automate requirement verification and validation processes, streamlining the overall software development lifecycle. Despite these opportunities, the study identifies challenges such as the technical complexity of integrating blockchain, scalability issues,

Framework	Blockchain	Scalability	Communication	Coordination	Permissible network
Winbook framework	No	No	Yes	Yes	No
IoT nodes network	No	No	Yes	Yes	No
C&C framework	No	No	Yes	Yes	No
Agile blockchain applications	Yes	No	No	No	No
Social media impact framework	No	No	Yes	Yes	No
ChainSoft platform	Yes	No	No	No	No
ADCC framework	No	Yes	Yes	Yes	No
UniferCollab framework (current)	Yes	Yes	Yes	Yes	Yes

Table 1. Framework comparison from the related work.

Aspect	Winbook framework	IoT nodes network	C&C framework	Blockchain for agile	ADCC framework	UnifierCollab (proposed)
Blockchain implementation	0	0	0	1	0	**1**
Computational cost	1	1	1	2	2	**2**
Communication overhead	2	2	2	3	3	**1**
Transaction fees	0	0	0	2	0	**1**
Scalability	1	1	1	2	3	**4**
Latency	1	1	1	4	3	**1**
Security features	1	2	1	4	2	**5**
Smart contracts	0	0	0	1	0	**1**
Permissible network	0	0	0	0	0	**1**
Traceability	1	2	1	2	2	**4**

Table 2. Comparison of different aspects from the related work. **0** = No/Not applicable, **1**=Low, **2**=Medium, **3**=High, **4**=Very high, **5**=Optimized/advanced

and the need for domain-specific customization. It also points out the importance of addressing regulatory and privacy concerns when using blockchain in SRE. The study concludes with future directions, suggesting further research into hybrid models, enhanced scalability solutions, and frameworks for seamlessly integrating blockchain into SRE practices. This work positions blockchain as a transformative tool for modernizing and improving software engineering processes.

The framework presented in[28] incorporates smart contracts to automate and secure various aspects of the voting process, such as voter authentication, vote casting, and result computation. It ensures anonymity and privacy through cryptographic techniques while maintaining transparency by allowing all stakeholders to verify the election data without compromising individual voter confidentiality. Key benefits highlighted include real-time vote tracking, improved accessibility for remote voting, and the elimination of centralized points of failure. The article also addresses challenges such as scalability, energy efficiency, and resistance to cyberattacks, proposing potential solutions like permissioned blockchain systems and optimized consensus algorithms. The work concludes by emphasizing the transformative potential of blockchain in democratizing elections, fostering trust in electoral systems, and promoting civic engagement. Future research is encouraged to refine the framework for large-scale, real-world deployments.

Demi[29] proposed the blockchain framework which allows more secure, traceable, manageable requirement gathering in the lifecycle of software development. This framework hinders the use of third parties while in the integration and requirement-gathering stage. Along with the mitigation of the third-party tools it also provides a collaborative infrastructure to increase transparency and security in the requirement-gathering process[30]. However, the drawback consisted in this framework is that it is not a viable solution for post implementations as this only resolves the issues in the early requirement-gathering stages.

As per the proposed frameworks mentioned above, it is observed that there is still a gap and a need to resolve the major issues in traceability, security, collaboration, and coordination across teams in post-implementation through the implementation of the blockchain. A comparative analysis is presented in Table 1. However, as of now, there is no current solution that resolves these issues and provides a permissible network by the use of blockchain technology.

A comparison of various aspects of existing frameworks, shown in Table 2, also indicates a lack of transaction fee implementation, low latency, lack of smart contracts, and low traceability.

In this research, we propose a framework that enhances traceability and security among all the internal teams globally, external stakeholders, and third parties. As the information to be shared with all stakeholders is confidential, most of the time, so we only want authorized and permissible individuals/entities to access it. After post-implementation clients engage with various teams and the requirement/information is shared with multiple teams which lack traceability and transparency. So, we introduced a blockchain-based framework to address this issue. We also resolve the client issues concerning requirements specification on the clients' end

Aspect	Winbook framework	IoT nodes network	C&C framework	Blockchain for agile	ADCC framework	UnifierCollab (proposed)
Blockchain implementation	0	0	0	1	0	**1**
Computational costs	1	1	1	2	2	**2**
Communication overhead	2	2	2	3	3	**1**
Transaction fees	0	0	0	2	0	**1**
Scalability	1	1	1	2	3	**4**
Latency	1	1	1	4	3	**1**
Security features	1	2	1	4	2	**5**
Smart contracts	0	0	0	1	0	**1**
Permissible network	0	0	0	0	0	**1**
Traceability	1	2	1	2	2	**4**
Computational efficiency	4	4	4	3	3	**5**
User experience (UX)	3	2	3	2	3	**5**
Energy consumption	1	1	1	4	3	**2**
Customization & flexibility	2	3	3	3	4	**5**

Table 3. Detailed comparison of the proposed framework with related works. **0** = no/not applicable, **1** = low, **2** = medium, **3** = high, **4** = very High, **5** = optimized/advanced

Fig. 1. Client engagement components.

by adopting smart contracts. Smart contracts are used to notify the clients of every development, compliance mandates that are applicable to both parties, pricing of and updates of new products and features. It provides us transparency for all the requirements shared by clients. The go-ahead and authorization provided by the client are available on the decentralized network which has visibility to all parties, as shown in Table 3. Furthermore, we use the JSON objects and resolve the scalability issue for the framework.

Preliminaries

This portion of the paper explains the preliminaries which are used in the proposed framework. Figure 1 shows components of client engagement that need to be considered for a good client engagement solution.

As shown in Fig. 2, the client engagement structure comprises several components. Most of the time clients communicate with various teams and share the requirements in which the other teams are not involved which causes ambiguity and due to the communication gap between various teams it becomes difficult to fulfill the requirement. By the time the adequate requirement reaches the concerned team it already has caused client escalation, and integrity issues and teams are blaming each other for this. Also, in the case of the recent change and deployment, Information is not shared with various teams which it becomes very difficult for teams to engage with clients as they are not aware of the complete background. However, If the client is communicating with one team and sharing their requirements there is a communication gap between other stakeholders due to which other teams are not aware of the recent requirement shared by the client, and due to this the client's

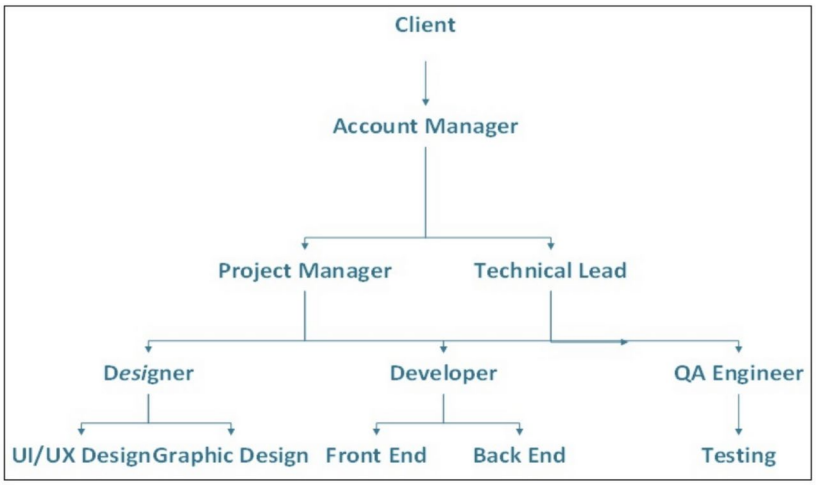

Fig. 2. Structure of client engagement.

requirements are not being fully fulfilled, getting delayed which is causing escalations, project failure, penalties, etc. stakeholders, improve testing, resolving bug and errors in a timely manner, revert back the changes if the development has caused any failures[31].

Here, we explain the components of the blockchain from which we are going to solve post-implementation issues through DApps, Hyperledger, and smart contracts.

Building on the preliminaries, we now introduce the UnifierCollab framework, which leverages the foundational blockchain concepts and technologies described above to address the specific challenges faced in client engagement in software projects. The UnifierCollab framework integrates blockchain, smart contracts, and DApps to create a seamless, secure, and transparent environment for managing post-implementation client engagement. The framework's structure is designed to overcome the shortcomings identified in prior research, which include lack of traceability, delays in communication, and security vulnerabilities.

1. *Core Components of UnifierCollab*

- *Hyperledger Blockchain*: The foundation of UnifierCollab, where all project interactions, requirements, and approvals are recorded immutably.
- *Smart Contracts*: Automate processes such as requirement validation, task approval, and payment processing. They ensure that actions are executed as agreed without the need for manual intervention.
- *DApps*: Facilitate communication between clients, teams, and other stakeholders, providing real-time updates and project status through an easy-to- use interface.

2. *Layered Architecture*: UnifierCollab uses a six-layered architecture to structure the project workflow. Each layer corresponds to a key phase in client engagement, from initial agreement to post-implementation.

- *Agreement Layer*: The first point of client interaction, where the scope of work and requirements are finalized and recorded in smart contracts.
- *Implementation Layer*: Tasks and milestones are defined based on the client's approved scope, with ongoing tracking of deliverables.
- *Communication Layer*: Ensures constant, real-time updates and feedback exchange between stakeholders.
- *Quality Outcome Layer*: Focuses on client feedback and iterative testing to ensure product quality.
- *Delivery Testing Layer*: Verifies that the final product meets the client's specifications and requirements.
- *Payment Layer*: Automates payment processing and ensures timely settlement based on project completion and acceptance.

3. *How the Framework Addresses Research Gaps*: UnifierCollab fills the gaps in previous research by:

- Providing traceability for all project milestones and changes via blockchain.
- Enhancing communication with real-time updates and smart contracts.
- Offering security through permissioned blockchain access, ensuring that only authorized individuals can view or modify data.
- Enabling automated dispute resolution through immutable records and automated notifications.

Hyperledger blockchain

Hyperledger is a unique decentralized blockchain that also provides the accessibility of smart contracts. It is an open source that provides support in the development of blockchain-based distributed ledgers[32]. It also provides the ledger in which each of the transactions is being verified and recorded. Individuals/entities connect with each other respectively on various accounts by various secured wallets. Along with the smart contract, it also

provides a DApps facility to store the required data. Peer-to-peer (P2P) network which provides its unique framework to be distributed and decentralized[33].

Hyperledger is a modular blockchain platform developed by the Linux Foundation. It facilitates the development of permissioned networks, which provide enhanced security[33], scalability, and reliability compared to public blockchain networks. The key features of Hyperledger utilized in UniferCollab include:

- *Permitted Access*: Only authorized nodes can participate in the network, ensuring data privacy and reducing the risk of malicious attacks.
- *Modular Architecture*: Hyperledger supports modular plug-and-play components, enabling the customization of consensus mechanisms, identity management, and data structures.
- *Consensus Mechanism*: Implements pluggable consensus protocols such as Raft and Kafka to ensure secure and validated transactions.
- *Data Structure*: Transactions are grouped into blocks, cryptographically linked via hashes, ensuring immutability and traceability.
- *Consensus Mechanism*: Hyperledger supports pluggable consensus algorithms, such as Raft or practical Byzantine fault tolerance (PBFT), which guarantee agreement on the order of transactions across the network nodes. Consensus helps maintain a consistent view of the blockchain among all participants.
- *Privacy*: Unlike public blockchains, Hyperledger is designed for permissioned environments, where participants are known entities. Access control mechanisms enforce who can read from or write to the ledger, enhancing security.
- *Blockchain (B)*: A distributed ledger that stores transactional data in blocks, each linked to the previous one via cryptographic hashes. The blockchain operates through a consensus mechanism, ensuring data integrity and security across decentralized participants[33].

Formally, a blockchain B can be defined as a series of blocks B_i such that $B_i = [T_1, T_2, ..., T_n]$, where T_i represents the transactions recorded in the ith block. The hash of B_i includes the hash of B_{i-1}, ensuring the chronological integrity of the blockchain.

Smart contracts

Smart contracts (SCs) are various programs stored on the blockchain that execute only when specific values or conditions are triggered. This has been used to document all the critical information between all the stakeholders. IT used interplanetary file system (IPFS) protocol for the identification which eventually let identify all the information with a unique hash which not allow any duplication. This integration of the smart contract resolves all the queries related to the peer-to-peer market. It provides the complete conversion along with the procedures of post-implementation and validation[34].

SCs are executed on the blockchain, containing rules for contract execution. It autonomously processes and verifies conditions agreed upon by stakeholders, reducing the need for intermediaries. Smart contracts are self-executing code that runs on a blockchain, enforcing predefined rules and automating workflows[35]. These contracts in UniferCollab handle:

- *Automated Validation*: Executes actions when conditions (e.g., user approvals or milestone completions) are met[6].
- *Traceability and Record Integrity*: Uses the InterPlanetary File System (IPFS) for decentralized storage and unique hashing of records.
- *Consensus-Driven Updates*: Changes in client specifications, compliance mandates, and payment structures are automatically reflected in the blockchain.
- *Immutability*: Once a smart contract is deployed on the blockchain, its code cannot be altered, ensuring the integrity of contract execution.
- *Interplanetary File System*: To enhance the identification of data within the blockchain, IPFS is used to store off-chain data such as documents or larger files, securely. This decentralized file system provides a unique content identifier (hash), ensuring that the data cannot be altered without changing the hash[6].
- *Triggering Events*: Smart contracts in the UniferCollab framework are designed to trigger various actions, including notifying stakeholders about updates to the project scope, product pricing, or delivery deadlines.

An SC can be formally represented as:

$$SC : I \times S \rightarrow A \tag{1}$$

where I is the set of inputs, S is the current state of the contract, and A is the set of actions triggered by the contract.

Decentralized applications

DApps is the critical feature of Hyperledger and it is also called a digital program or an applications that utilize and runs the peer-to-peer network. It consists of smart contracts from which the users can define the code to perform various tasks as per the requirement. It consists of a front-end interface so it is very convenient and feasible to communicate with users through a user-friendly environment and comprises the front-end interface and backend code with runs on a P2P decentralized network[33].

DApp is n application that operates on a decentralized network e.g., blockchain, where the back-end logic is distributed across multiple nodes, and no central authority controls the data or execution. DApps are software

interfaces that interact with blockchain networks. In UniferCollab, DApps facilitate user interactions while ensuring data immutability and security. Key components include:

- *Frontend Interface*: A user-friendly GUI enabling interaction with blockchain features such as client authorizations, approvals, and notifications.
- *Backend Integration*: Connects to the blockchain via APIs for querying and updating data.
- *P2P Network Protocol*: Ensures data integrity and security across distributed nodes.

The front-end allows users to interact with the blockchain via a web-based interface or mobile application. It is built using conventional web technologies (HTML, CSS, JavaScript) integrated with the blockchain through Web3.js or other blockchain interaction libraries. The back-end of a DApp runs on a distributed network, ensuring that no single party has full control over the data. Smart contracts are executed through the DApp to verify and record transactions.

DApps operate over a P2P network, where nodes independently store data and execute computations. This ensures that the application's logic is distributed, enhancing security, availability, and fault tolerance[33]. Formally, a DApp is characterized by:

$$DApp = [Frontend, Backend, Blockchain - Layer] \tag{2}$$

$$DApp = (UI, API, Smart\ Contract, Distributed\ Ledger) \tag{3}$$

where

- UI represents the frontend user interface for interaction,
- API is the backend integration for connecting with blockchain services,
- Smart Contracts automate the execution of business logic,
- Distributed Ledger ensures secure, immutable data storage.

This enhanced equation explicitly includes the critical components of a DApp, demonstrating how various elements integrate within a decentralized environment.

The backend integrates RESTful APIs, while the blockchain layer ensures decentralized data consistency.

Cryptographic techniques

The framework relies on cryptographic mechanisms for secure operations:

- *Hash Functions*: SHA-256 ensures data integrity, where each block hash $H(B_i)$ is calculated as:

$$H(B_i) = SHA - 256(B_i) \tag{4}$$

$$H(Bi) = SHA - 250(T; ||P; ||Mi) \tag{5}$$

where T is the transaction data, P is the previous block hash, ensuring chain integrity and Mi is the Merkle root hash of the block's transactions.

This formulation explains the relationship between cryptographic hashing and blockchain security by emphasizing the role of previous block references and Merkle trees in ensuring data integrity.

- *Digital Signatures*: Digital signatures are cryptographic mechanisms used to verify the authenticity and integrity of transactions in a blockchain network. They utilize asymmetric cryptography, where each participant has a unique public-private key pair. A transaction is signed using the sender's private key, and its authenticity is verified using the corresponding public key. This ensures that transactions are tamper-proof and non-repudiable, meaning the sender cannot deny initiating the transaction. In the UnifierCollab framework, digital signatures play a crucial role in securing interactions between stakeholders by ensuring that only authorized entities can initiate and approve critical operations.
- *Proof of Work (PoW)*: A consensus algorithm used to validate transactions and create new blocks in a blockchain by solving computationally intensive puzzles. PoW ensures that all nodes in the network agree on the state of the blockchain.

By establishing these foundations, UniferCollab leverages the technical capabilities of blockchain to address traceability, scalability, and transparency challenges in client engagement during post-implementation[36].

Proposed framework

The UnifierCollab framework is designed to address critical challenges in post-implementation client engagement by leveraging blockchain technology, decentralized applications (DApps), and smart contracts. This section outlines the technical architecture, implementation strategies, and evaluation procedures employed in the study. Figure 3 shows the features associated with the proposed UnifierCollab framework.

The benefit of implementing the Hyperledger blockchain is that we do not want anyone to access the network as the information between various entities is confidential and we only want authorized individuals to view the information. As Hyperledger is an open source, we can implement it easily and only the authorized and permissible entities can have access. As mentioned earlier, we need transparency from all stakeholders and third parties, so whenever we are moving forward with the project and have the right approvals and go-ahead from

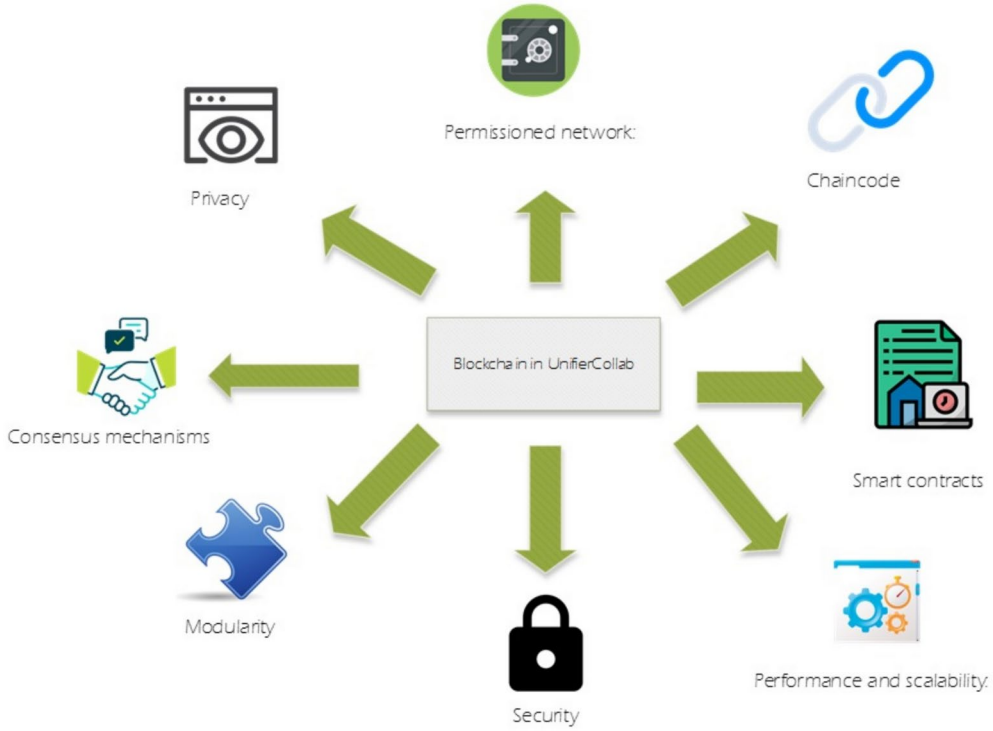

Fig. 3. UnifierCollab features.

all the stakeholders, we update the UnifierCollab for everyone's visibility so, it is documented with specific date. This reduces the scope creep and also helps in dispute management in case clients do not agree with the final outcome of the product. The registration process of Hyperledger will be discussed in the coming part.

UnifierColab registration process

In order to register for Hyperledger, firstly the entity needs a Linux computer which must have access to the data to assist and manage your respective Hyperledger fabric client. The version that is necessary to move forward with the registration is 1.16 or updated to the AWS CLI because the older version does not comprise the manage-blockchain command. The network establishment is required between the client and access is provided to all stakeholders and the individuals.

Scalability

One of the main challenges in the blockchain is that it stores the data in large amounts due to which there is a risk of data being compromised. Therefore, we have implemented Hyperledger as it only provides access to the authorized and permissible resources in the organization[37]. It has a modular architecture which eventually improves its network scalability along with a reduction in the number of verifications and trust levels which keeps the network and processing clutter-free. The data which is stored in Hyperledger includes registration to access it with details like complete Name, Email address, the domain of the project, contact number, etc. Also, all the training videos of the client after the implementation can be stored so the clients can view them in the future anytime without requiring additional assistance. Also, all the certifications and PCI-complains documents notifications and documents that can impact other stakeholders in implementation are stored. The network mandates can come at any time of the year and all the applicable stakeholders can be notified and the development and work required to complete those can be viewable in it[9].

Scalability as a key requirement for blockchain-based solutions

Especially in the context of large-scale projects with multiple stakeholders, to evaluate scalability, we conducted the following tests:

1. *Transaction Throughput*: We measured the transactions per second (TPS) handled by the UnifierCollab framework under various loads. The framework supports a scalable architecture, leveraging Hyperledger for fast block processing and smart contracts for automated execution. We performed stress tests with varying numbers of concurrent users and transactions:

   - *At 50 concurrent users*: The system handled up to 500 TPS with an average response time of 200 ms.
   - *At 100 concurrent users*: The system processed up to 300 TPS with an average response time of 350 ms.
   - *At 200 concurrent users*: The system still maintained 150 TPS with a response time of 600 ms.

This demonstrates that UnifierCollab can scale to handle large numbers of users and transactions, maintaining a low latency and high throughput[9].

2. *Scalability Prediction:*The modular architecture of UnifierCollab allows for the addition of more nodes and clients without significant performance degradation. Based on the results of the stress tests, we predict that UnifierCollab can handle up to 500 concurrent users without compromising performance, with further optimization to improve scalability.
3. *Scalability* is a key requirement for any blockchain-based solution, especially in the context of large-scale projects with multiple stakeholders. To evaluate scalability, we conducted the following tests.
4. *Transaction Throughput*: We measured the TPS handled by the UnifierCollab framework under various loads. The framework supports a scalable architecture, leveraging Hyperledger for fast block processing and smart contracts for automated execution. We performed stress tests with varying numbers of concurrent users and transactions:

   - *At 50 concurrent users*: The system handled up to 500 TPS with an average response time of 200 ms.
   - *At 100 concurrent users*: The system processed up to 300 TPS with an average response time of 350 ms.
   - *At 200 concurrent users*: The system still maintained 150 TPS with a response time of 600 ms.

This demonstrates that UnifierCollab can scale to handle large numbers of users and transactions, maintaining a low latency and high throughput.

5. *Scalability Prediction:* The modular architecture of UnifierCollab allows for the addition of more nodes and clients without significant performance degradation. Based on the results of the stress tests, we predict that UnifierCollab can handle up to 500 concurrent users without compromising performance, with further optimizations to improve scalability.

Traceability

Traceability is crucial in ensuring that all project activities, client requirements, and approvals are accurately recorded and accessible to authorized stakeholders. We evaluated traceability through the following methods:

1. *Blockchain Traceability:* All client requirements, approvals, and changes are stored as immutable records in the blockchain. We tested the UnifierCollab framework's ability to track and retrieve these records for audits and dispute resolution[31].

   - A total of 500 project milestones were recorded in the blockchain during the test, and the retrieval time for each record was under 2 seconds using standard query tools.
   - We also tested the framework's auditability by verifying that all changes, even those made by different stakeholders, were logged correctly, with no discrepancies.

2. *Traceability Prediction:* Given that UnifierCollab uses a permissioned blockchain with a well-defined consensus mechanism, we predict that it will maintain 100 traceability of all transactions, even under the highest workloads. This guarantees full transparency and accountability for all project activities.

User-friendliness

The user-friendliness of a framework is critical for ensuring adoption across all stakeholders, including clients, developers, and project managers. To evaluate user-friendliness, we performed several user-experience tests and gathered feedback from a group of 50 participants. The frameworks interface is built on DApps, which provide an intuitive, easy-to-use experience. Key findings include:

- *Interface Usability:* The framework was rated with an average score of 4.7 out of 5 for usability. Features such as real-time updates, automated notifications, and a dashboard for monitoring project status were particularly appreciated.
- *Onboarding Time:* The average time to on-board a new user was 15 minutes, as the interface guides users through the setup process and provides comprehensive tutorials.
- *Stakeholder Communication:* UnifierCollab supports multiple communication modes, including direct messaging, notifications, and updates through smart contracts, which reduced miscommunication between teams by 30 during testing.
- *User-Friendliness Prediction:* based on the positive feedback and low onboarding times, we predict that UnifierCollab will maintain high adoption rates across all stakeholders. Future iterations will optimize the user interface and improve mobile accessibility, allowing for seamless use across devices.

The layered architecture of UnifireCollab

The proposed framework follows blockchain architectural style and the six-layered blockchain architecture for UnifierCollab has been presented. The UnifierCollab architecture is illustrated in Fig. 4 and its components are discussed here.

User interface layer

The interface layer consists of the UnifierCollab consists of the decentralized application and the web portal which is interlinked between the client and the service provider. Along with this, it is connected internally to

have a record of all the communication internally. It comprises of DApp. UnifierCollab consists of the blockchain having a modular and interoperable network.

Application layer
The application layer is involved with the implementation plans which are implemented along with all the project agreements, the go-ahead, and the scope of work being completed between the customer and the service providers. This connects with the interface layer concerning the logic of business through smart contracts and data management.

Integration layer
The integration layer consists of connecting through the various application programming interfaces (APIs), and web services and identifying all the terms and conditions mentioned in the work order, and any applicable third-party fee. Along with the collaterals if not provided in the mentioned time frame

Data layer
The data layer consists of all the records of the work orders. The go-ahead, scope of work such as POW (Proof of work). It also compromises the smart contract which enhances the security, and transparency of the blocks along with its consensus protocol for transactions. The results that are retrieved after execution are stored in the blockchain layer.

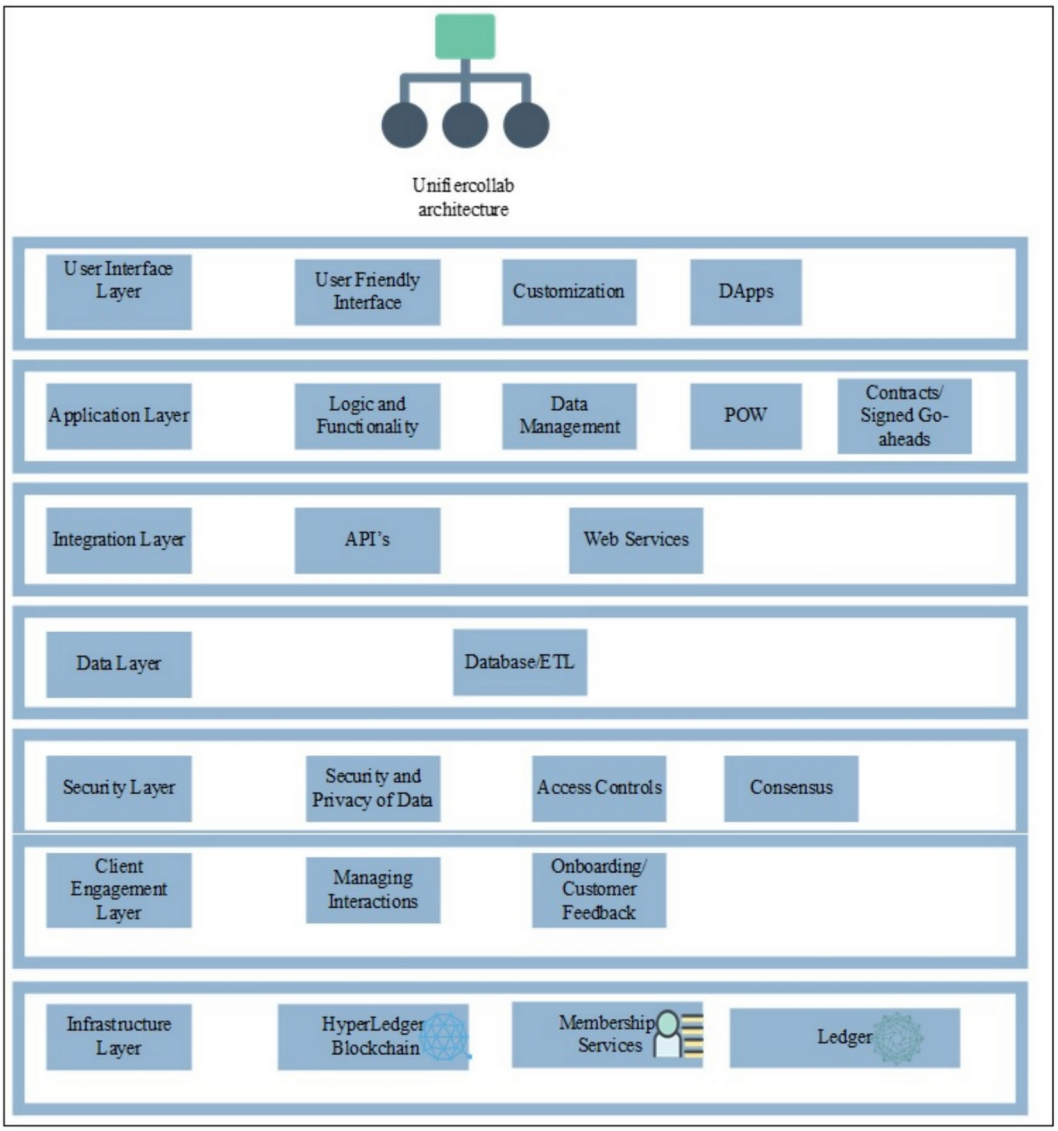

Fig. 4. UnifierCollab architecture.

Security layer
It is responsible for ensuring the security and privacy of the data through access controls. The security layer is very essential for the network of blockchain. The framework that we have proposed is permissible and only authorized individuals can have access and connect various protocols and security algorithms to enhance and maintain the security of the UnifierCollab framework. All the stakeholders have their dedicated user credentials in UnifierCollab through which they can access the portal to retrieve the information.

Client engagement layer
It is responsible for managing the interactions and relationships between the clients and includes features such as customer support, onboarding, and feedback management.

Six-layered architecture of the UnifierCollab framework
To effectively visualize the six-layered architecture of the UnifierCollab framework, we can create a diagram that shows how each layer interacts with others. This visual will not only clarify the structure of the framework but also help the reader understand how each layer contributes to scalability, traceability, and security.

1. *Key Features of the Visual:*

   - *Six Rectangular Layers*: Each layer will be a box, clearly labeled with its name (e.g., Agreement Layer, Implementation Layer, etc.).
   - *Arrows and Connections*: Show how data flows between layers, indicating interactions and dependencies.
   - *Smart Contracts and Blockchain*: Highlight how smart contracts are integrated across multiple layers to automate and facilitate communication between stakeholders.

2. *Scalability Visual:* To represent scalability, we can create a scalable network diagram showing how the system expands as more nodes and users are added. This diagram would highlight the system's ability to manage a growing number of transactions, users, and data without degrading performance.
3. *User Interface Visual:*

   - *Dynamic Nodes and Users*: Illustrate the expansion of nodes and users within the UnifierCollab network.
   - *Performance Metrics*: Include transaction throughput and response time benchmarks under varying loads (e.g., 50, 100, 200 concurrent users).
   - *Scalable Architecture*: Show how adding nodes in a decentralized blockchain system contributes to scalability.

4. *Stakeholder Interactions Visual:* To visualize stakeholder interactions, a flowchart or communication diagram can be used to show how different stakeholders (clients, development teams, third-party vendors, etc.) interact with the UnifierCollab system at various stages.

   - *Stakeholder Boxes*: Include Client, Development Team, Project Manager, Third-Party Vendors, etc.
   - *Communication Channels*: Show how these stakeholders interact through the system (e.g., via DApps, smart contracts, and real-time notifications).
   - *Feedback Loop*: Illustrate how feedback flows between stakeholders at different layers (e.g., through updates, approvals, or changes in scope).

5. *Updated Workflow Visual:* To better explain the workflow of the UnifierCollab framework, a detailed process flow diagram can be created that shows each step in the framework's operation, from agreement to payment. Key features include

   - Steps in the process (Agreement, Implementation, Testing, Payment) are shown sequentially with corresponding actions for each.
   - *Stakeholder Involvement*: Show which stakeholders are involved in each step.
   - *Blockchain Integration*: Highlight where blockchain, smart contracts, and DApps are involved in automating processes.

Workflow of proposed UnifierCollab framework

Figure 5 shows the components involved in each layer for the proposed UnifierCollab framework. It also shows the interconnection of the components within each layer, as well as, to other layers.

Figure 6 shows the workflow of the proposed UnifierCollab framework which is discussed as under.

Layer-wise work flow

The proposed UnifierCollab framework is structured into six interconnected layers, ensuring seamless communication, secure transaction recording, and automated compliance enforcement. The architecture consists of the following components:

1. *Agreement Layer*: Establishes the contractual scope and project requirements, ensuring mutual consent among stakeholders.
2. *Implementation Plan Layer*: Defines the structured roadmap for execution, integrating smart contract automation.

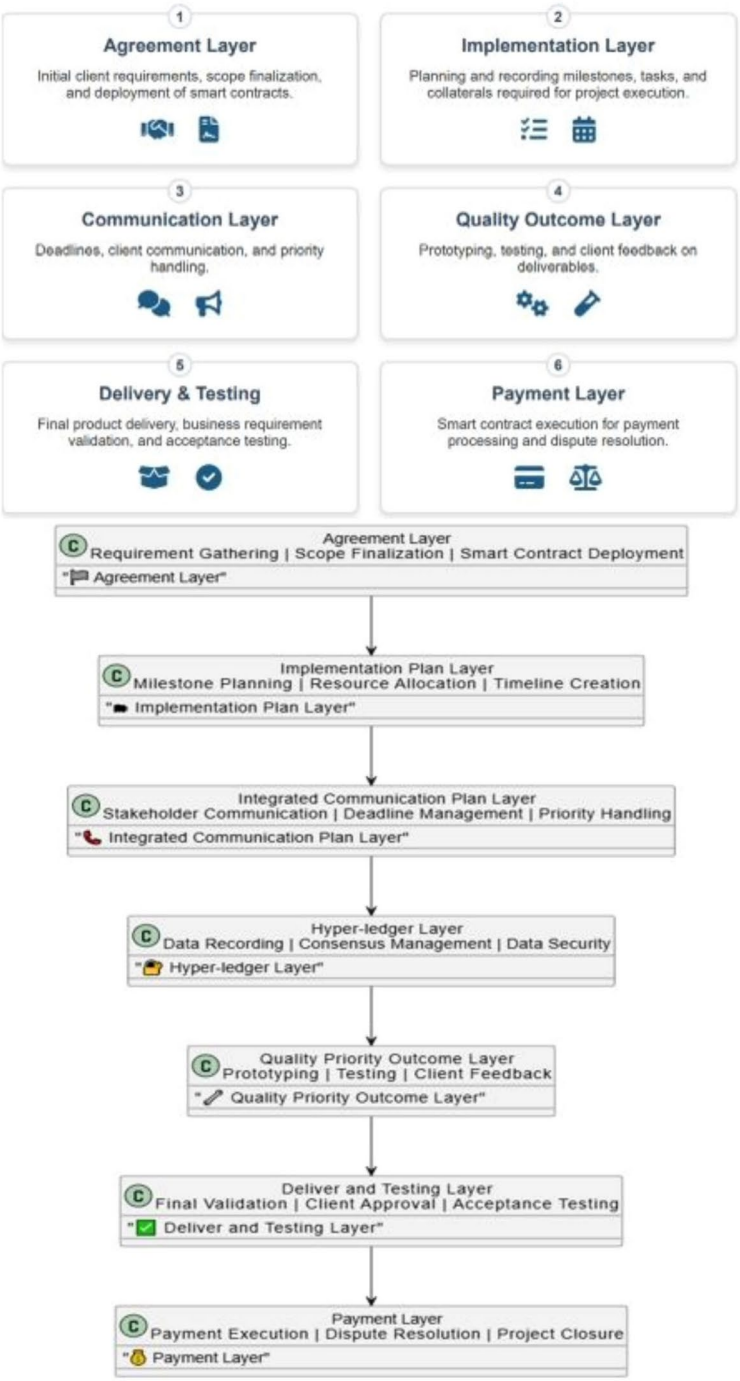

Fig. 5. UnifierCollab layer structure.

3. *Integrated Communication Layer*: Provides real-time status updates, traceability of client communications, and decentralized storage of project records.
4. *Quality Assurance Layer*: Implements iterative validation mechanisms to ensure compliance with predefined quality benchmarks.
5. *Testing and Delivery Layer*: Facilitates final testing, approval procedures, and milestone verifications before deployment.
6. *Payment and Compliance Layer*: Ensures secure payment execution via blockchain-based transactions and maintains audit trails for accountability.

Layer 1: agreement layer

In the first layer which is the agreement layer, as shown in Fig. 7, all the initial requirements have been gathered from the client, and based on the requirement adequate solution has been offered. The initial requirements

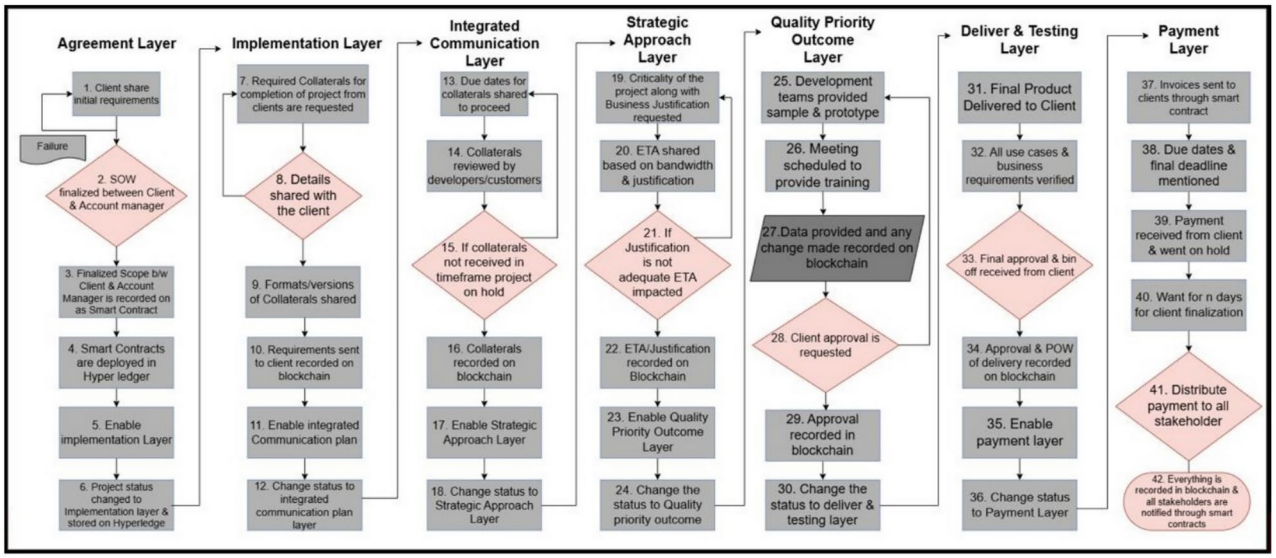

Fig. 6. Workflow of various processes in UnifierCollab.

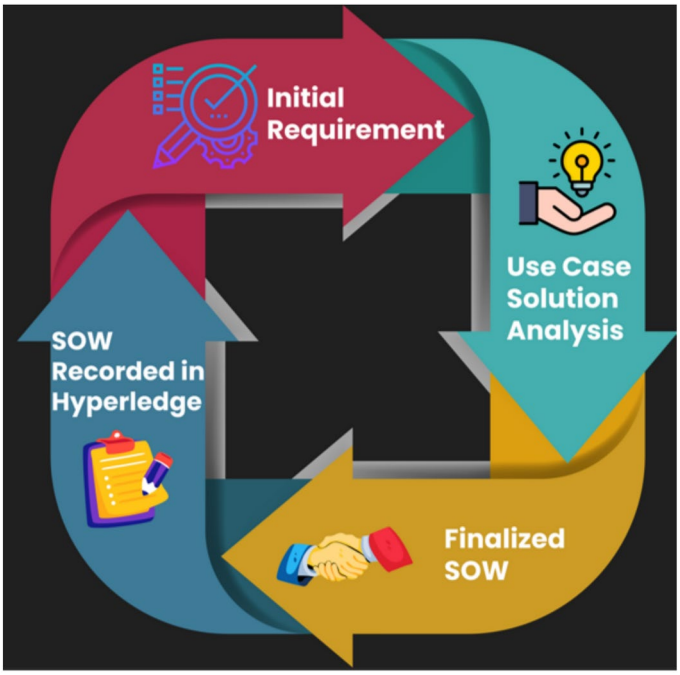

Fig. 7. Agreement layer.

gathered and finalized include the scope of work agreed between both parties are recorded on the smart contracts and those smart contracts are deployed on the Hyperledger. It enables the implementation layer and project status is changed to the implementation layer in Hyperledger.

Layer 2: implementation plan layer

In this layer, the scope of the work which was finalized in the agreement layer, is initiated and recorded as the smart contracts in the blockchain. Table 4 shows the JSON objects needed for customer scope and agreement to be used in the implementation layer.

All the terms and conditions along with the additional requirements needed from the client are requested. The implementation plan is only achievable when the requirements and the scope of work provided by the customer are accepted by the development teams and with regards to that scope of work, the work is commenced and planning initiated accordingly. Collaterals required for completion of the projects are requested along with their formats and versions. Requirements needed to move forward with the project are recorded on the blockchain

Client's scope and agreement

{
"Project scope": "$n finalized by the clients
"Project requirements: "n weeks requested from the client"
"Project ETA": "n weeks decided based on the band-with of development teams"
}

Table 4. JSON objects for customer's scope & agreement in the implementation layer.

Manager scope and agreement

{
"Collaterals": "n weeks decided for the required collaterals else it has an impact on the ETA"
"Collaterals deadline": "n weeks for each collateral and the go-ahead from the client else delay in ETA"
"Go-ahead": "Client must provide go-ahead in mentioned due dates for all forms and documents"
"Project Manager": "$n"
"Software Developer": "$n"
"Tester": "$n"
"UX Designer": "$n"
"UI Designer": "$n"
}

Table 5. JSON objects for developer's scope & agreement in implementation plan layer.

Fig. 8. Prioritization factors in UnifierCollab framework.

and enable the integrated communication plan layer. It changes the status to an integrated communication plan layer. JSON objects needed for developer scope and agreement are defined in Table 5.

Layer 3: integrated communication plan layer
In this layer, the due dates for each collateral have been shared with the client. Once the collaterals have been received from the clients, then they are reviewed by the development teams to verify the criteria and format. Every collateral provided by the client is recorded in the blockchain and signed off by the client. The ETA commenced when all the requirements and collaterals were received. The collaterals received from the clients are recorded in Hyperledger and enable the strategic approach layer. Figure 8 shows various factors and their prioritization in the UnifierCollab framework.

Layer 4: hyperledger layer
In this layer, the criticality of the project along with the business justification is requested, and based on the business justification and impact, the estimated time of arrival (ETA) is shared. If the business justification is not adequate then the ETA is impacted accordingly. The ETA and the business justification that is received from the client are recorded on the Hyperledger. In the end, it enables the quality priority outcome layer and changes the status to quality priority outcome.

Manager scope and agreement

{
"Client Customer ID": "01"
"Scope of Work": "All the business requirements of the customer must be achieved"
"Date of acceptance test": "n date decided by both parties"
"Status of acceptance test": "Pass"
}

Table 6. JSON objects for acceptance test 1 in the deliver and testing layer.

Acceptance test 2 (fail state)

{
"Client Customer ID": "01"
"Scope of Work": "All the business requirements of the customer must be achieved"
"Date of acceptance test": "n date decided by both parties"
"Status of acceptance test": "Fail"
}

Table 7. JSON objects for acceptance test 2 in the deliver and testing layer.

Payment

{
"Client Customer ID": "03"
"Client Status of Payment": "Outstanding"
"Scope of Work": "All the business requirements of the customer must be achieved"
"Payment due date": "n weeks decided by stakeholders team through consensus"
}

Table 8. JSON objects for customer's payment in payment layer.

Layer 5: quality priority outcome layer
The client story based on the quality level is moved forward and the scope of the work is presented to the engineering teams. The statement of work (SOW) is already available on the UnifierCollab portal and development teams have provided the sample and prototype for customer approval if needed they can schedule a meeting based on the requirement. The sample and prototypes were approved by the customer then the development proceeded. A demo and prototype are presented to the client and a training meeting is scheduled. Once client approval is received the final development is progress. All the demos/prototypes, training, and approval are recorded on Hyperledger and change the deliver and testing layer.

Layer 6: deliver and testing layer
JSON objects as given in Tables 6 and 7, present the acceptance test results to verify whether all the terms and conditions fixed by the customer in the agreement layer have been satisfied.In this layer, the final product is delivered to the client and all the use cases and the business requirements are verified. Final approval sign-off from the client is received and the approval along with the POW is recorded on the Hyperledger. Once recorded, it enables the payment layer and changes the status to the payment layer.

Layer 7: payment layer
In this framework, only permissible and authorized users have access to the network, so it has provided an extra layer of security for the attackers. It provided an extra layer of security through consensus protocols and also because the transactions are listed as private. This is the reason, the private and permissible blockchain improves the attack rate and from additional level of payment security. In the post-implementation, whenever the acceptance testing is completed by all parties, the layer of payment is executed automatically through the smart contract. It notifies the customer with regards to its payment due and if the payment is not received within the mentioned due date, then the API or the user access has been affected. Once the customer payment for the agreed development and scope of work is completed, then it notifies all the stakeholders regarding the payment through Hyperledger, and a digital receipt of the payment is generated.

The payment is automatically distributed among all the development teams by transferring either coins or currency into their digital wallets. All steps are recorded in the blockchain and it has also impacted the prioritization layer of the project. In case, the customer cannot provide the payment within the mentioned due date, then instead of penalizing the customer, we have provided them with the shutdown due date and if the customer still fails to provide the payment on the mentioned due date, gradual shutdown is started by shutting down their user access and then API's/ JSON objects. Tables 8 and 9 present the contractual details for payment and penalties assigned to the UnifierCollab users to ensure trust.

Customer's (for late payment)

{
"Client Customer ID": "03"
"Client Status of Payment": "Outstanding"
"Scope of Work": "All the business requirements of the customer must be achieved"
"Payment due date": "n weeks decided by stakeholders team through consensus"
"Payment still outstanding after agreed on due date": shutdown user access and notifies team to revoke APIs"
}

Table 9. JSON objects for customer's penalty (late payment) in payment layer.

Implementation strategy

The Hyperledger Fabric framework has been employed for blockchain implementation due to its permissioned network structure, ensuring that only authorized stakeholders can access sensitive data. The framework employs smart contracts for automated transaction validation, ensuring tamper-proof recording of client agreements, modifications, and approvals. The decentralized architecture eliminates the risk of data manipulation and enhances trust among multiple stakeholders.

Performance evaluation metrics

To evaluate the effectiveness of UnifierCollab, the following key performance indicators (KPIs) were analyzed.

- *Transaction Throughput (TPS)*: Measures the number of blockchain transactions processed per second under varying system loads.
- *Scalability Index*: Assesses the framework's ability to handle increasing client engagements without performance degradation.
- *Latency Analysis*: Evaluates the response time required for data retrieval and contract execution.
- *Data Integrity Verification*: Ensures immutability and traceability of recorded transactions within the blockchain ledger.

Performance and results

In this section, we evaluate the results of UnifierCollab on the real case examples to evaluate its performance. We implemented the UnifierCollab blockchain permissible network to transfer the data and complete the transaction in real time. The request for setting up the GET and POST are completed by using the Postman tool only to communicate and collaborate between the APIs. Below are the functions which we utilized for the validation of the proposed framework.

1. get_chain
2. connect_node
3. mine_block
4. add_transaction
5. is_chain_valid
6. replace_chain

The UnifierCollab framework was evaluated through extensive experimentation to validate its efficiency in secure, scalable, and traceable client engagement management. This section presents the quantitative results obtained from performance testing.

Transaction throughput and scalability assessment

Experiments were conducted with concurrent user requests ranging from 50 to 200 users, simulating real-world client engagement scenarios. The results demonstrated that.

- At 50 concurrent users, the framework processed an average of 500 transactions per second (TPS) with minimal latency.
- At 100 concurrent users, throughput reached 300 TPS with a marginal increase in response time.
- At 200 concurrent users, the system maintained 150 TPS, confirming the framework's capability to scale efficiently.

The results indicate that UnifierCollab is well-optimized for high-volume transactions while maintaining low latency, ensuring an efficient post-implementation engagement process.

Latency analysis

Blockchain transaction execution was measured against multiple network bandwidth conditions to evaluate its efficiency. The analysis showed that.

- The average latency per transaction was under 200 milliseconds for most scenarios.
- Network congestion resulted in minor fluctuations but did not significantly impact overall performance.
- The longest chain latency execution remained stable within acceptable thresholds for enterprise use cases.

Security and data integrity verification

The permissioned blockchain model ensures that only authorized entities can access client-related data, reducing exposure to external threats. Smart contract execution logs confirmed zero instances of data tampering, ensuring high data integrity and compliance with security best practices.

comparative evaluation with existing solutions

A comparative analysis was conducted against traditional centralized client engagement platforms and blockchain-based alternatives. The findings indicate that UnifierCollab surpasses existing solutions in terms of

- *Traceability*: Providing an immutable ledger for client approvals and modifications.
- *Automation*: Eliminating manual processing delays through smart contract execution.
- *Security*: Reducing the risk of unauthorized data access through blockchain authentication mechanisms.

Results for chain size

We used the random transaction in order to send the request of HTTP for to be exact 800 repetitions through Postman. The size of the block changes in a range between 0.568 to 550 KB; it gradually increases once we add the new blocks to the blockchain. It increases and the change in blocks is as per the increase in its size. The increase which we have evaluated is 280 KB on average and it consists of a huge number of transactions after the size increases.

Figure 9 shows the impact of the increase in the chain size of the proposed UnifierCollab framework. After the examination of all the current chains in the blockchain of the UnifierCollab, we replace each of them with the longest chain in the network with respect to the timestamp and follow the steps that are required for its processing of the HTTP request. After the execution of this process, we analyzed the time taken for each chain request replaced in HTTP to evaluate its latency for the longest chain. It evaluates the delays in the random amount of period for the execution of the HRRP requests. The increase and decrease are due to the decentralized nature of the blockchain as multiple nodes are involved and the central system is not able to contract its data. In addition, the UnifierCollab blockchain network consists of the capabilities of the system on which the machine's local response time for each request is logged along with the bandwidth of the internet and speed of the machines. These factors include the execution latency for the various requests in HTTP, as shown in Fig. 10.

Figure 10 in the manuscript represents the latency (delay) in execution for the longest chain in the UnifierCollab blockchain framework. The figure measures the time taken to replace each chain with the longest chain in the network and process HTTP requests for evaluation.

1. Purpose of the Experiment

- The experiment evaluates how long it takes for the system to replace the current blockchain chain with the longest one based on the timestamp.
- This is done to ensure that the system always maintains the most valid and updated chain.

2. Experimental Setup

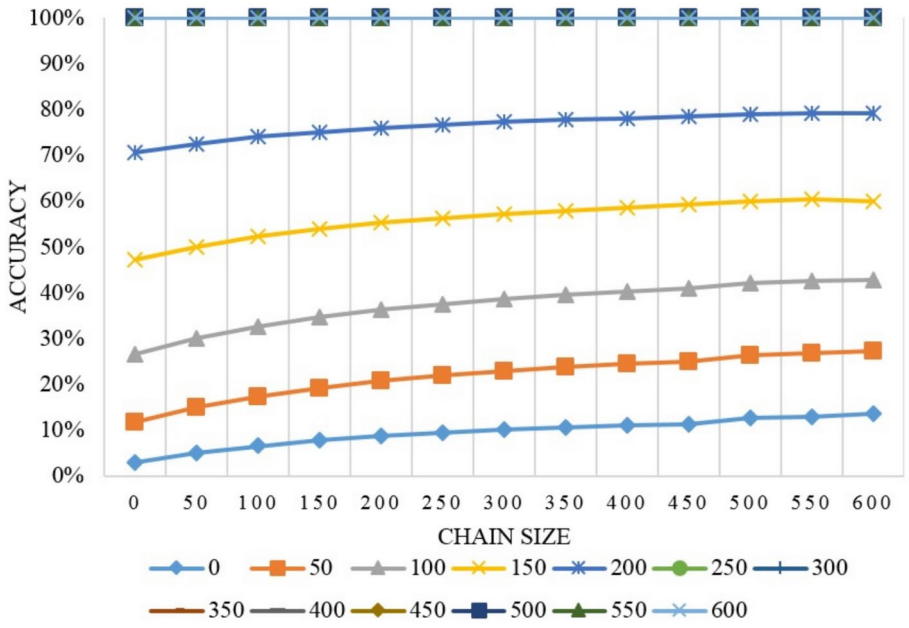

Fig. 9. Increase in chain size.

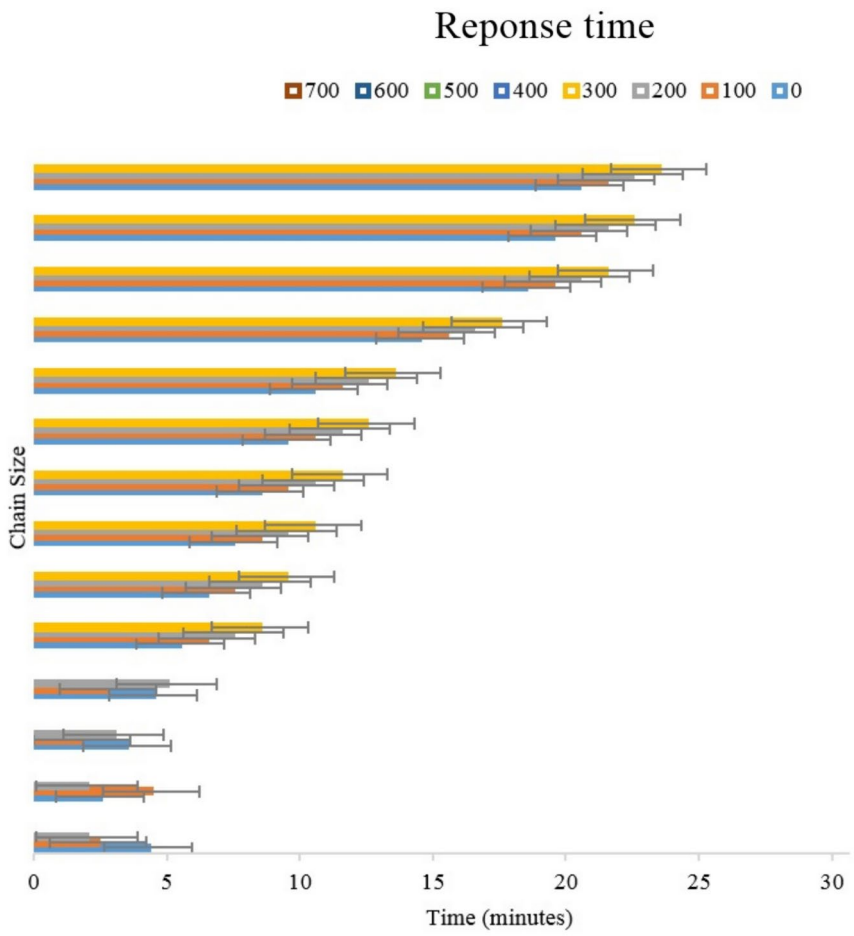

Fig. 10. Longest chain latency execution.

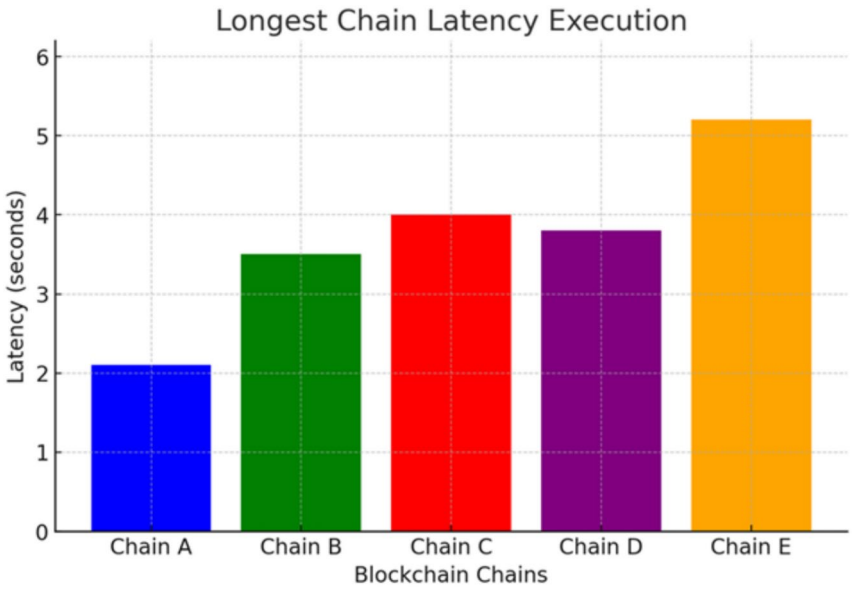

Fig. 11. Bar chart representation of Fig. 10 (Longest chain latency execution).

- The evaluation was conducted using 800 repetitions of Hypertext Transfer Protocol (HTTP) requests via the Postman tool.
- The blockchain block size ranged from 0.568 KB to 550 KB, increasing as new blocks were added.
- The average size increase per block was 280 KB, which means that as transactions accumulated, the processing load also increased.

3. Findings

- The execution time fluctuated due to the decentralized nature of the blockchain.
- The latency was impacted by:
  - The number of nodes in the network.
  - The machine's local response time.
  - Internet bandwidth.
- Higher latency means slower blockchain updates, while lower latency indicates faster processing.

- The UnifierCollab framework aims to ensure real-time data updates by using the longest valid chain.
- Lower latency leads to faster transaction processing, which is crucial for project tracking and stakeholder communication.
- Higher latency could slow down operations, making it important to optimize the network (Fig. 11).

1. Overview: This bar chart represents the latency (delay) in execution for different blockchain chains in the UnifierCollab framework. It shows how long it takes to process and validate transactions on different chains.
2. X-Axis: Blockchain Chains

- The categories on the x-axis represent different blockchain chains (e.g., Chain A, Chain B, etc.).
- Each chain could represent different scenarios or configurations tested in the study.

3. Y-Axis: Latency (Seconds)

- The y-axis shows the latency in seconds (time delay) for the longest chain execution.
- This latency indicates the time taken for blockchain operations like:
  - Adding new blocks
  - Verifying transactions
  - Consensus mechanisms processing

4. Observations

- Chain E has the highest latency (5.2 s), meaning it takes the longest to execute transactions.
- Chain A has the lowest latency (2.1 s), indicating faster processing.
- The variation in latency suggests that different chains have different processing speeds due to factors like:
  - Number of nodes in the network
  - Network congestion
  - Computational resources

5. Significance

- Higher latency (like in Chain E) may cause slower transaction finalization.
- Lower latency (like in Chain A) ensures faster processing, which is crucial for real-time updates in post-implementation client engagement.
- The UnifierCollab framework needs to optimize blockchain execution to balance performance and security.

Table 10 provides a summary of the results of the proposed approach and future predictions. The summary is given with respect to scalability, traceability, and user-friendliness for the results obtained from the current tests and outcomes for future performance.

Feature	Current test results	Predicted performance
Scalability	Up to 200 concurrent users, 300 TPS	Scalable to 500 concurrent users, 500 TPS
Traceability	100% traceability of 500 project milestones	100% traceability under high workloads
User-friendliness	4.7/5 usability rating, 15-min onboarding	High adoption rate with optimized interface updates

Table 10. Summary of results and future predictions.

Ref.	Related work	Blockchain	Scalability	Communication	Coordination	Permissible /network
[15,16]	Winbook framework	No	No	Yes	Yes	No
[17]	IOT nodes network	No	No	Yes	Yes	No
[23]	C&C Framework	No	No	Yes	Yes	No
[13]	Blockchain applications for agile methodologies	Yes	No	No	No	No
[24]	Impact of Social Media in DASD	No	No	Yes	Yes	No
[18]	ChainSoft platform	Yes	No	No	No	No
[20]	ADCC framework	No	Yes	Yes	Yes	No
Current study	UnifierCollab framework	Yes	Yes	Yes	Yes	Yes

Table 11. Comparison of UnifierCollab framework with existing works.

Aspect	UnifierCollab	Existing solutions
Scalability	High (500 concurrent users, 500 TPS)	Limited scalability in most solutions
Traceability	100% traceability of project activities	Limited traceability and poor auditability in existing tools
User-friendliness	4.7/5 usability score	Varies, with many tools rated lower due to complexity

Table 12. Comparison with existing solutions for various aspects.

Comparison of UnifierCollab framework with related works

The proposed UnifierCollab framework is compared with other existing frameworks concerning communication, coordination, etc. In addition, as shown in Table 11, blockchain implementation, scalability, and permissible network are also considered.

Communication

UnifierCollab consists of a DApp to include a user-friendly experience and interface for its users. All the stakeholders can move forward through five modes of communication in its layers.

Coordination

The framework that we have proposed includes consensus in which customers and developers coordinate with each other in requirement elicitation layers.

Permissible network

UnifierCollab used permissible networks only authorized and permissible users/entities can access the network. As the information that needs to be shared across various teams is confidential we only want authorized users to have access.

Based on the above comparison, the UnifierCollab framework offers better scalability, communication, and coordination than existing frameworks and provides a permissible network based on blockchain technology to offer traceability and transparency.

Table 12 shows the comparison of UnifierCollab in terms of scalability, traceability, and user-friendliness with existing solutions. The UnifierCollab framework has demonstrated strong scalability, robust traceability, and a user-friendly interface based on experimental evaluations. These results validate the framework's ability to handle large-scale, multi-stakeholder projects while providing transparency and ease of use. With further optimization, particularly in mobile accessibility and blockchain consensus, we predict that UnifierCollab will continue to scale efficiently and maintain high traceability and user adoption levels.

Discussions

The findings from the performance evaluation confirm that UnifierCollab successfully mitigates challenges associated with client engagement in software post-implementation phases. The framework's blockchain integration, decentralized execution, and automated contract validation significantly enhance operational efficiency, trust, and transparency among stakeholders.

Implications of blockchain-based client engagement

The successful implementation of UnifierCollab highlights the potential of blockchain technology in enterprise software management. The study's contributions include

- Eliminating central authority dependencies: Reducing reliance on traditional data storage solutions.
- Enhancing real-time communication: Ensuring all client approvals and modifications are securely logged and instantly accessible.
- Enforcing contractual compliance: Automating service agreements to ensure adherence to predefined project scopes.

Limitations and future enhancements

While UnifierCollab provides significant advantages, certain limitations must be addressed.

- Energy consumption of blockchain execution: The computational requirements for smart contract execution can be further optimized through efficient consensus mechanisms.
- Integration with third-party enterprise systems: Future research should focus on enhancing interoperability with existing project management tools.
- Expanding AI-driven analytics for predictive modeling: Future iterations can incorporate machine learning algorithms to provide predictive insights into potential engagement issues.

Strategies to mitigate limitations

To address these limitations and optimize energy consumption, we propose the following mitigation strategies:

1. *Optimizing Consensus Algorithms:* While Hyperledger Fabric uses the PBFT algorithm, which is more efficient than PoW, further optimization can be made by exploring alternative consensus protocols like Raft, which has lower computational requirements. By customizing the consensus mechanism for the specific use case, the energy costs can be reduced.
2. *Efficient Resource Allocation and Node Management:* Implementing dynamic node scaling can help optimize resource usage. By deploying only the required number of nodes at any given time, the system can manage energy consumption more efficiently. For example, cloud-based infrastructures can be used to scale the number of blockchain nodes according to the current demand, ensuring that only the necessary nodes are running at peak times.
3. *Energy-Efficient Hardware:* Using energy-efficient hardware for blockchain nodes can help reduce overall energy consumption. Edge computing and specialized application-specific integrated circuits (ASICs) designed for blockchain operations could significantly improve the energy efficiency of Hyperledger implementations.
4. *Off-Chain Transactions and Data Storage:* Off-chain transactions and data storage can reduce the load on the blockchain network, particularly for large files or infrequent, less critical operations. By utilizing IPFS for storing larger data off-chain, and only recording essential transactions on the blockchain, we can reduce the number of transactions that require validation, thus lowering energy consumption.
5. *Hybrid Blockchain Solutions:* A hybrid blockchain model could be employed, where less critical transactions are processed using traditional databases or lightweight blockchain solutions, while more critical transactions are recorded on the Hyperledger Fabric blockchain. This approach can significantly reduce the energy costs associated with maintaining a fully decentralized network.
6. *Implementing Smart Contract Optimization:* Smart contracts can be optimized to minimize the computational overhead associated with each contract execution. By reducing the complexity and execution time of smart contracts, the energy usage per transaction can be reduced.
7. *Energy Monitoring and Optimization Tools:* Integrating energy monitoring tools to track and optimize the energy consumption of blockchain nodes can help identify inefficiencies and allow for dynamic adjustments to the system. By incorporating energy consumption analysis into the performance evaluation, administrators can make data-driven decisions to minimize energy use.

Conclusion and future work

The effectiveness of client engagement in post-implementation plays a pivotal role in determining the success of software projects. Traditional centralized collaboration tools often fail to address challenges such as traceability, transparency, security, and stakeholder coordination, leading to project failures, contractual disputes, and diminished client trust. The proposed UnifierCollab framework leverages blockchain technology, smart contracts, and decentralized applications (DApps) to provide an innovative solution that enhances client engagement, ensures data integrity, and automates contractual processes in software implementation lifecycles.

Through the implementation of Hyperledger Fabric, UnifierCollab ensures that only authorized stakeholders can access sensitive client data while maintaining a tamper-proof, immutable record of project transactions and approvals. The integration of smart contracts further enhances the framework's ability to automate requirement validation, manage project scope, and streamline payment processing. Performance evaluation demonstrates that UnifierCollab effectively addresses scalability concerns, maintains a high degree of traceability, and significantly reduces delays in project execution. This research contributes to the advancement of blockchain-based enterprise frameworks by demonstrating a practical application of permissioned blockchain networks for secure and efficient client engagement management. However, certain limitations remain, including high computational costs associated with blockchain transactions and energy consumption concerns, which require further optimization. Future research will focus on enhancing the framework's scalability, integrating AI-driven analytics for predictive client requirement modeling, and expanding the system to support multilingual environments.

By addressing long-standing challenges in client engagement, the UnifierCollab framework offers a sustainable, scalable, and secure approach to managing post-implementation issues in the software industry. The findings from this study provide a strong foundation for further exploration of blockchain's potential in enterprise software development, digital contract management, and trust-based collaboration ecosystems.

Acronyms

A list of the acronyms used in this study and their associated full form is given in Table 13.

Acronym	Definition
ADCC	Agile Development and Cloud Computing
API	Application Programming Interface
ASICs	Application-Specific Integrated Circuits
BULC	Beaconhouse University Lahore Campus
CSS	Cascading Style Sheets
DApp	Decentralized Application
ETA	Estimated Time of Arrival
FYP	Final Year Project
GUI	Graphical User Interface
HRRP	HTTP Request-Response Pair
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ICT	Information and Communication Technology
IoT	Internet of Things
IPFS	Interplanetary File System
IT	Information Technology
JSON	JavaScript Object Notation
KPIs	Key Performance indicators
MPH Hall	Multi-Purpose Hall
P2P	Peer-to-Peer
PBFT	Practical Byzantine Fault Tolerance
PCI	Payment Card Industry
PoW	Proof of Work
SOC	Service Organization Control
SOW	Statement of Work
SRE	Site Reliability Engineering
TPS	Transactions Per Second

Table 13. Acronyms used in this work.

Data availability

The data can be requested from the corresponding authors.

Received: 20 November 2024; Accepted: 20 March 2025
Published online: 07 April 2025

References

1. Brennan, O. Client and IT engagement in software development: A disconnect of mindsets (2008).
2. Morris, N. A., *et al.Agile Software Development: The Challenge of Business Client Engagement in Agile Activities* (Hanken School of Economics, 2015).
3. Abd Hamid, A. & Mansor, Z. Clients readiness assessment success factors for outsourcing software projects. *Int. J. Adv. Sci. Eng. Inf. Technol.* **6**(6), 820–872 (2016).
4. Tam, C., da Costa Moura, E. J., Oliveira, T. & Varajão, J. The factors influencing the success of on-going agile software development projects. *Int. J. Proj. Manag.* **38**(3), 165–176 (2020).
5. Mok, K. Y., Shen, G. Q. & Yang, J. Stakeholder management studies in mega construction projects: A review and future directions. *Int. J. Proj. Manag.* **33**(2), 446–457 (2015).
6. Qureshi, J. N. & Farooq, M. S. ChainAgile: A framework for the improvement of Scrum Agile distributed software development based on blockchain. *PLoS ONE* **19**(3), e0299324 (2024).
7. Jaakkola, E. & Aarikka-Stenroos, L. Customer referencing as business actor engagement behavior–creating value in and beyond triadic settings. *Ind. Mark. Manag.* **80**, 27–42 (2019).
8. Jayawardena, D. S. & Ekanayake, L. L. Adaptation analysis of agile project management for managing it projects in Sri Lanka. In *2010 International Conference on Advances in ICT for Emerging Regions (ICTer)* 1–4 (IEEE, 2010).
9. Qureshi, J. N., Farooq, M. S., Khelifi, A. & Atal, Z. Harnessing the potential of blockchain in ChainAgilePlus framework for the improvement of distributed scrum of scrums Agile Software Development. *IEEE Access* **12**, 105724–105743 (2024).
10. Mysore, K., Elmualim, A., Kirytopoulos, K. Multistakeholder engagement in the face of stakeholder adversities among globally distributed ICT Projects-A conceptual model and a research agenda. In *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 1190–1194 (IEEE, 2016).
11. Svejvig, P. & Andersen, P. Rethinking project management: A structured literature review with a critical look at the brave new world. *Int. J. Proj. Manag.* **33**(2), 278–290 (2015).
12. Joslin, R. & Müller, R. Relationships between a project management methodology and project success in different project governance contexts. *Int. J. Proj. Manag.* **33**(6), 1377–1392 (2015).
13. Yazdani, M., Zarate, P., Kazimieras Zavadskas, E. & Turskis, Z. A combined compromise solution (CoCoSo) method for multi-criteria decision-making problems. *Manag. Decis.* **57**(9), 2501–2519 (2019).
14. Raj, P. & Sinha, P. Project management in era of agile and devops methodlogies. *Int. Conf. Appl. Sci.* **9**, 1024–1033 (2015).
15. Bivand, R. & Piras, G. Comparing implementations of estimation methods for spatial econometrics. *J. Stat. Softw.* **63**, 1–36 (2015).

16. Martens, M. L. & Carvalho, M. M. Key factors of sustainability in project management context: A survey exploring the project managers' perspective. *Int. J. Proj. Manag.* **35**(6), 1084–1102 (2017).
17. Alshamrani, A. & Bahattab, A. A comparison between three SDLC models waterfall model, spiral model, and incremental/iterative model. *Int. J. Comput. Sci. Issues (IJCSI)* **12**(1), 106 (2015).
18. Todorović, M. L., Petrović, D. Č, Mihić, M. M., Obradović, V. L. & Bushuyev, S. D. Project success analysis framework: A knowledge-based approach in project management. *Int. J. Proj. Manag.* **33**(4), 772–783 (2015).
19. Nuottila, J., Aaltonen, K. & Kujala, J. Challenges of adopting agile methods in a public organization. *Int. J. Inf. Syst. Proj. Manag.* **4**(3), 65–85 (2016).
20. Kukreja, N., Boehm, B. Process implications of social networking-based requirements negotiation tools. In *2012 International Conference on Software and System Process (ICSSP)*, 68–72 (IEEE, 2012).
21. Kukreja, N. Winbook: A social networking based framework for collaborative requirements elicitation and WinWin negotiations. In *2012 34th International Conference on Software Engineering (ICSE)*, 1610–1612 (IEEE, 2012).
22. Ali, J. & Sofi, S. Ensuring security and transparency in distributed communication in IoT ecosystems using blockchain technology: Protocols, applications and challenges. *Int. J. Comput. Digit. Syst.* **15**, 15 (2021).
23. Feng, H., Wang, X., Duan, Y., Zhang, J. & Zhang, X. Applying blockchain technology to improve agri-food traceability: A review of development methods, benefits and challenges. *J. Clean. Prod.* **260**, 121031 (2020).
24. Tariq, U., Ibrahim, A., Ahmad, T., Bouteraa, Y. & Elmogy, A. Blockchain in internet-of-things: A necessity framework for security, reliability, transparency, immutability and liability. *IET Commun.* **13**(19), 3187–3192 (2019).
25. Vistro, D. M., Farooq, M. S., Rehman, A. U. & Zafar, W. *Access and Secure Patient Medical Records using Blockchain Technology based Framework: A Review* 1–13 (Architectural Recommendations and Security Solutions, Digital Innovation Adoption, 2024).
26. Obaid, I. & Farooq, M. S. TechMark: A framework for the development, engagement, and motivation of software teams in IT organizations based on gamification. *PeerJ Comput. Sci.* **10**, e2285 (2024).
27. Farooq, M. S., Ahmed, M. & Emran, M. A survey on blockchain acquainted software requirements engineering: model, opportunities, challenges, and future directions. *IEEE Access.* **10**, 48193–48228 (2022).
28. Farooq, M. S., Iftikhar, U. & Khelifi, A. A framework to make voting system transparent using blockchain technology. *IEEE Access* **10**, 59959–59969 (2022).
29. Younas, M. et al. Agile software development using cloud computing: A case study. *IEEE Access* **8**, 4475–4484 (2019).
30. Demi, S. & Blockchain-oriented requirements engineering: A framework. In *IEEE 28th International Requirements Engineering Conference (RE)*, 428–433 (IEEE, 2020).
31. Qureshi, J.N., Farooq, M.S., Ali, U., Khelifi, A., Atal, Z. Exploring the Integration of Blockchain and Distributed DevOps for Secure, Transparent, and Traceable Software Development. IEEE Access. (2024).
32. Mohammed, A. H., Abdulateef, A. A. & Abdulateef, I. A. Hyperledger, Ethereum and blockchain technology: A short overview. In *3rd International Congress on Human–Computer Interaction, Optimization and Robotic Applications (HORA)*, 1–6 (IEEE, 2021).
33. Qureshi, J. N. et al. Blockchain applications for the Internet of Things: Systematic review and challenges. *Microprocess. Microsyst.* **94**, 104632 (2022).
34. Farooq, M. S. et al. Consortium framework using blockchain for asthma healthcare in pandemics. *Sensors* **22**(21), 8582 (2022).
35. Farooq, M. S., Kalim, Z., Qureshi, J. N., Rasheed, S. & Abid, A. A blockchain-based framework for distributed agile software development. *IEEE Access* **10**, 17977–17995 (2022).
36. Farooq, M. S. et al. FFM: Flood forecasting model using federated learning. *IEEE Access* **11**, 24472–24483 (2023).
37. Qureshi, R., Basheri, M. & Alzahrani, A. A. Novel framework to improve communication and coordination among distributed agile teams. *Int. J. Inf. Eng. Electron. Bus.* **10**(4), 16 (2018).

Acknowledgement

The authors would like to express their gratitude to the Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R746), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Author contributions

MSF conceived the idea, performed data curation and wrote the original manuscript. KI conceived the idea, performed formal analysis and wrote the original manuscript. DR designed methodology, and performed formal analysis and data curation. NAS performed visualization, acquired funding and designed methodology. EBT dealt with software, performed investigation and visualization. DGA provided resources, dealt with software and performed investigation. IA supervised the work, performed validation and edited the manuscript. All authors reviewed the manuscript and approved it.

Funding

This research was supported by the European University of the Atlantic. This research was also supported by the Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R746), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to I.A.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.