applied sciences

Article

Blockchain-Based Decentralized Architecture for Software Version Control

Muhammad Hammad [1], Jawaid Iqbal [2], Ch Anwar ul Hassan [3], Saddam Hussain [4,*], Syed Sajid Ullah [5,*], Mueen Uddin [6], Urooj Ali Malik [7], Maha Abdelhaq [8] and Raed Alsaqour [9]

1 Department of Software Engineering, Capital University of Science & Technology, Islamabad 44000, Pakistan
2 Faculty of Computing, Riphah International University, Islamabad 44000, Pakistan
3 Department of Creative Technologies, Air University, Islamabad 44000, Pakistan
4 School of Digital Science, Universiti Brunei Darussalam, Jalan Tungku Link, Gadong BE1410, Brunei
5 Department of Information and Communication Technology, University of Agder (UiA), N-4898 Grimstad, Norway
6 College of Computing and IT, University of Doha for Science and Technology, Doha 24449, Qatar
7 Military College of Signals, National University of Science & Technology Rawalpindi, Islamabad 46000, Pakistan
8 Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
9 Department of Information Technology, College of Computing and Informatics, Saudi Electronic University, Riyadh 93499, Saudi Arabia
* Correspondence: saddamicup1993@gmail.com (S.H.); syed.s.ullah@uia.no (S.S.U.)

Abstract: Version control is an important component of configuration management, and most enterprise-level software uses different tools and technologies to manage the software version control such as CVS, Subversion, or Perforce. Following the success of bitcoin, the first practical application of blockchain, it is being implemented in other fields such as healthcare, supply chains, financial management, real estate, electoral systems, and so on. Blockchain's core features include decentralization, immutability, and interminability. Most version control repositories are centralized and can be modified by external sources, implying that they are in danger of being corrupted or controlled. In this study, we present the BDA-SCV architecture for implementing a version control system in blockchain technology. Our proposed approach would replace the necessity for a centralized system, with a decentralized approach implemented in the blockchain using distributed file storage, for which we will use the InterPlanetary File System (IPFS), which is a distributed file system. The proof of authority (PoA) consensus algorithm will be used to approve the developer communicating modifications to the private blockchain network; the authority will only provide permission and will not be able to add, edit, or delete code files. For each change, a ledger block will be created with a reference to the file stored in the distributed repository. A block cannot be manipulated once it has been created. Smart contracts will be used to register developers, create blocks, and manage the repository. The suggested model is implemented using the Hyperledger Fabric network, and the developer and authorizer ends are built into the dotnet web application.

Keywords: version control; blockchain; change management; IPFS; smart contract

check for updates

Citation: Hammad, M.; Iqbal, J.; Hassan, C.A.u.; Hussain, S.; Ullah, S.S.; Uddin, M.; Malik, U.A.; Abdelhaq, M.; Alsaqour, R. Blockchain-Based Decentralized Architecture for Software Version Control. *Appl. Sci.* **2023**, *13*, 3066. https://doi.org/10.3390/app13053066

Academic Editors: Hui Li, Shancang Li and Konstantinos Demertzis

Received: 25 November 2022
Revised: 9 January 2023
Accepted: 23 February 2023
Published: 27 February 2023

1. Introduction

With the introduction of bitcoin, a cryptocurrency, back in 2008 [1], the era of blockchain began. The core innovation behind bitcoin was blockchain, a technology based on immutability, decentralization, and interminability. After the achievement of bitcoin, many cryptocurrencies were introduced including Ethereum, Ripple, Solana, Dot, etc., and with them were developed new platforms for developing blockchain-based systems, with Ethereum and Hyperledger being the main toolkits [2]. Following the introduction of

Ethereum, blockchain applications expanded beyond bitcoin, and smart contracts phenomena came into existence. A smart contract is a small software program that exists in the blockchain and is activated when specific criteria are satisfied or when contacted by an external source [3]. Decentralized applications were first launched using smart contracts, and then we progressed toward developing complete systems on the blockchain, which included ERPs, supply chain management systems, e-voting systems, etc. [4,5]. Blockchain essentially creates a distributed record or database that is shared across all workers/group members in the network. There is currently no demand for outsider verification, as the system is now safer and completely decentralized. [6]. The external exchange authority end is completed by utilizing cryptography's capacity to provide dependable responses for the blockchain's members. Mainly, if using the proof of authority consensus algorithm, this carefully marks, reviews, and authorizes any modifications (if required) to the blockchain record, and a copy of the record is retained, making trades safe, completely decentralized, time-stamped, and tamper-proof. [6]. Usually, the copy is kept in multiple locations like a peer-to-peer network but now many different blockchain architectures have been introduced. The major three types of blockchains are private, public, and consortium blockchains, each having different architecture and subtypes. [7].

Blockchain innovation has been used in numerous enterprises like healthcare, financial systems, logistics, document management, and accounting [7]. The blockchain mining hub executes, confirms, and stores information in blocks. Blockchain has become one of the most advertised innovations nowadays; however, putting away enormous archives is still pricey as blockchain restricts the document, meaning a large amount of data cannot be stored in a blockchain. The vital need for storing enormous size records has been met by utilizing decentralized file storage frameworks like the InterPlanetary File System (IPFS), file coin, Storj, OrbitDB, Skeps, SWARM, and Sia [8–10]. Blockchain has also been merged with other fields like artificial intelligence [10] and cloud computing [11,12]. Regional Chain and Edge Computing [13], Inventory networks, supply chain management [14], Decentralized storage scheme Blockchain has a vigorous and decentralized foundation, so it is applied to deal with issues connected with trust, effectiveness, security, and information sharing [15]. In our proposed approach with BDA-SCV we have used IPFS as a decentralized file storage system and record details are kept in blockchain, both linked with the file reference being stored in block data. The blockchain is incapable of storing massive volumes of information yet it is demonstrated successful when it stores hashes of documents and only important information in the chain, rather than the actual record being linked by the hash of the next block. Fundamentally, a hash against the document is created each time a record is transferred to the IPFS, and this hash is used in the smart agreement/contract that is used to get to the document [16]. The hash value is changed each time, so any time changes are made in the data of the document it is considered as a new document. In this way, the integrity of information is confirmed.

Software configuration management is a field of software engineering that is specifically for managing change during and after the development of a software system [17]. The fundamental concepts in software configuration management are making baselines and versioning different lower-level components known as configuration items. Change can arise from any place or part, e.g., it can arise from the development team or it can arise from the user or any stakeholder as a new requirement. These changes are versioned and stored in a repository. The changes that are not controlled properly can cause chaos in software. Software configuration management helps in erasing confusion that can be caused after a change and it records every piece of information about the change as it is made. The earlier the change, the easier it is to manage. Software configuration management is a central part of software quality assurance [18]. Software configuration management is for controlling change during the software product life [19]. The integration of the SCM process in the development process is shown in Figure 1 [20]. We tend to implement a version control system that is a subpart of software configuration management in blockchain technology. Figure 1 shows the SCM process.

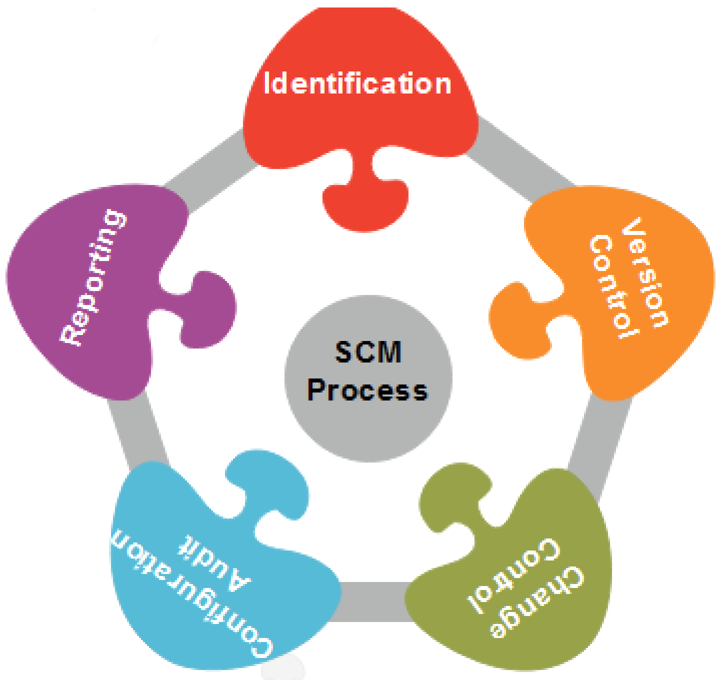


Figure 1. Software configuration management process.

In traditional software configuration management tools, the version control is usually managed by a single entity that can also change data or place files directly to the project, which can be devastating and can lead to software failure. In existing systems that are used for version control purposes, the majority of current version control systems are centralized, meaning they have a single repository and are also managed by the authority that can manipulate or temper the repository's data. The disadvantages are that clients do not have unlimited oversight of the report or file, and that documents can be erased, controlled, or altered, which can cause havoc in the development process [21].

In the software development process, changes are done frequently and need to be looked after in a managed way. Figure 2 shows a traditional version control architecture, which includes procedures to 'update' the localized document's new version and 'commit' procedures to upgrade the document from the working copy onto the central repository. The data will then be pushed and pulled in and out of the central repository. This shows a very simple representation of a version control system that becomes very complicated when there are hundreds of files and many people working on the project [22]. However, the main database is primarily centralized, with administrators and a central authority in charge. Before a file is 'committed' to the repository, the verification requires the confirmation of one of the authorities in case of a modification. By using the blockchain-based version control system, we are proposing a decentralized, distributed blockchain-based version control system, because, in existing distributed systems, this centralized notarization and verification management causes trust issues between different stakeholders of the project [23]. The concept of a distributed system is the decentralized storage of the data to maintain transparency and backup of the data. In a blockchain-based system, the data is stored in a distributed location to maintain the transparency of the data and achieve data integrity. However, in a centralized system, all the records are stored in a central location and, therefore, for a version control system, it is difficult to maintain data integrity. Blockchain technology variants and consensus selection are also important factors. While proposing the system, it is understood that the selection of a blockchain variant is important to maintain efficacy and the consensus algorithm. If the private blockchain variant is selected, then we face the issue of centralized security and data privacy being compromised somewhere. Moreover, the centralized solution is the same as the earlier version control systems that are already working. We adopted the Hyperledger Fabric network to develop the decentralized blockchain to maintain data integrity and proposed an efficient blockchain-based version control system.

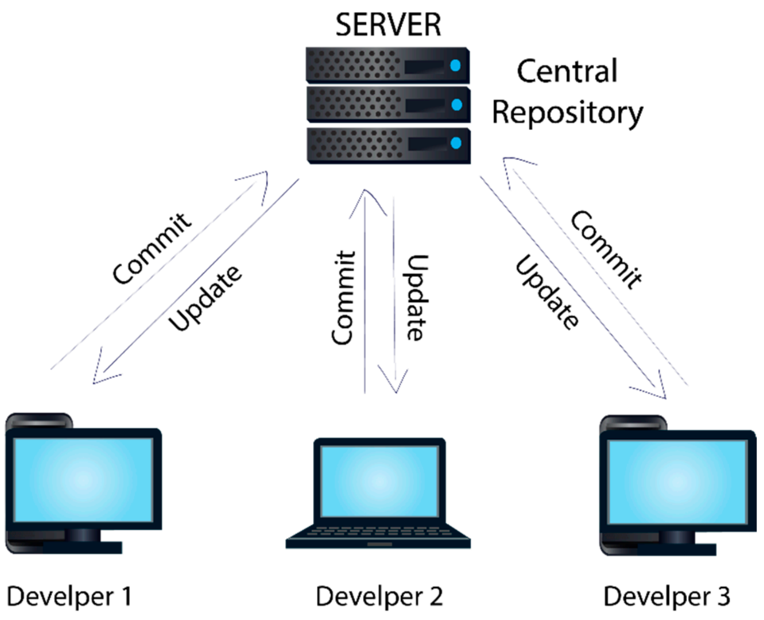


Figure 2. Tradition version control.

One of the areas where blockchain technology can help is document sharing and version control. To improve security and scalability, a decentralized version control solution without the need for a single authority and central repository is required. The document version approval process typically involves a large number of approvers, each of whom has a log that must be updated depending on the signatures received from all approvers across the distributed network. As a result, all parties involved in the registration process must keep a consistent record, which is not supported by current document version control systems [24]. In this paper, we propose BDA-SCV, a blockchain-based decentralized architecture for software version control, on top of which all SCM processes can be implemented. We have used smart contracts to create a blockchain-based system for version control for code files, which diminishes the need for a trusted third-party authentication between the developers and the approvers. The four major components of our proposed approach are a decentralized repository, blockchain, smart contract, and consensus algorithm. For a distributed repository, we have considered the IPFS distributed file system. We have used a proof of authority consensus algorithm, where the authority is responsible for adding development to the specific project. The Hyperledger toolset is used for implementation and the Hyperledger Fabric Client is deployed on a cloud environment, to which the developers will be connected, and the Hyperledger network is deployed where the ledger and IPFS system are deployed.

This paper’s main contributions can be summarized as follows:

1. Review of work done previously on the version control system and blockchain;
2. Proposal of a decentralized blockchain-based architecture for software version control;
3. Implementation of the proposed approach using Hyperledger Fabric on a cloud platform.

The rest of the paper is structured as follows. First, we discuss the background study for the research, followed by related work in which we explain the different related papers. Then, that we propose our approach to BDA-SCV and present implementation, ending the article with future work and conclusions.

2. Background

The blockchain is a distributed digital ledger (a digital record of transactions or data kept in several locations on a computer system) that is immutable (meaning that a transaction or file stored cannot be updated). The notion or protocol that makes the blockchain system work is known as blockchain technology. Cryptocurrencies like bitcoin are made feasible by blockchain technology, much as email is made possible by the internet. It has various applications outside of cryptocurrencies. Carrying out a blockchain from the beginning includes managing a progression of components that are not instinctive from the start. The hubs that collaborate should be synchronized at the network level, handle the approvals of consensus strategies, manage the web customer with which the client collaborates with the framework, and more. If a framework administrator has to control all of this, they can be overwhelmed by the intricacy of administrations that cooperate to keep the network working. In this sense, BaaS suppliers have arisen, which permit businesses to utilize predefined answers to produce, construct, and host the vital components in the cloud for the activity of a blockchain [25]. The service administrator manages the administrations of the provider, which permits the chairman to exploit the important assets, for example, dealing with the character/identity of the clients, adding security components, and heightening client honors, alongside security for the capacity of information in the cloud. First, we will discuss blockchain technology.

2.1. Blockchain Technology

A blockchain is defined as a "distributed database that maintains a continuously expanding list of ordered records, known as blocks", which are "connected using encryption". A cryptographic hash of the preceding block, a timestamp, and transaction data are all included in each block. A blockchain is a decentralized and distributed, and a public digital ledger is used to record transactions across many computers in such a way that the record cannot be changed retrospectively without affecting all following blocks and the network's consensus [6]. Figure 3 shows the structure of the blockchain ledger.

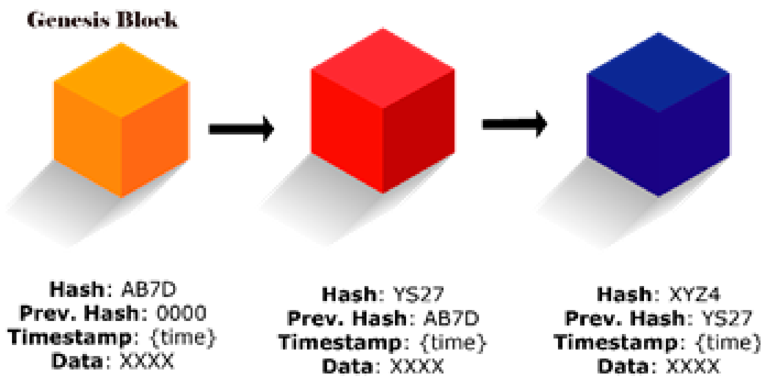


Figure 3. Blocks data security and linking in blockchain technology.

2.2. Software Configuration Management and Software Version Control

As we want to implement a version control system in blockchain technology, we should know how the software configuration process works. We will look at the basic elements of software configuration management. The baseline, configuration item, versions, and configuration control board are the four core elements of SCM. Software configuration management is a whole process in which changes are managed in software. There are different elements in the software configuration management process, which are discussed below. The version is the major component of software configuration management and version control is a process to manage multiple versions in a software process.

- Configuration Item (CI): in software configuration management, configuration items are single lower-level items that can be changed. A configuration item can be anything from a software project including a project planning document, project proposal, pert charts, code, requirement and design document, and test cases [4]. Configuration items are versioned as they change to distinguish them from other configuration items. Their dependence is also looked at.
- Version: CIs in software configuration management are versioned according to the different rules for distinguishing them from other CIs [26]. Our focus will be to manage version control in decentralized blockchain technology.
- Version Control: version control allow you to manage multiple revisions of the same unit of information, e.g., document, source file, or any document including code files in a project. The system has multiple users to control and share files. VCS allows an archive of the versions by which subsequent versions of source-controlled items can be stored, and historical information about these versions is also stored [23]. The major thing we achieve by version control is a collaboration by which people can share data and collaborate easily with other team members [22].
- Baseline: a baseline is a basic concept in software configuration management that help control a change. A baseline is a stable version of a code and it is known as the basis for further development [27]. It can comprise a single or more than one strong configuration item. Figure 4 shows baselines before the official release [27].

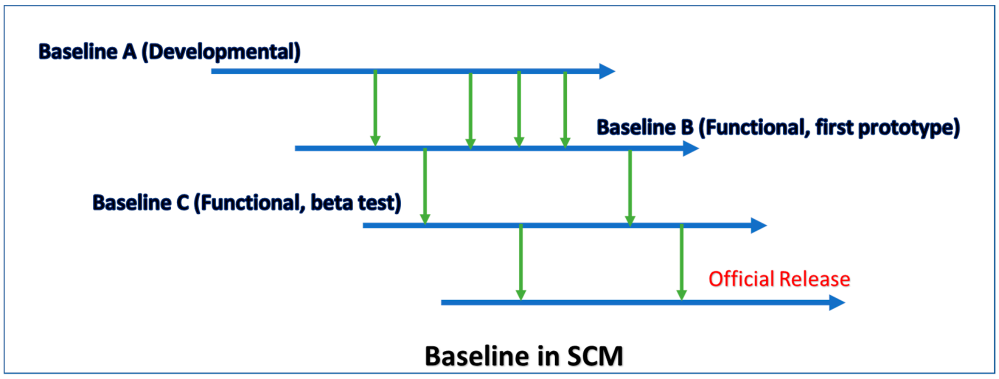

Baseline A (Developmental)
Baseline B (Functional, first prototype)
Baseline C (Functional, beta test)
Official Release
Baseline in SCM

Figure 4. Baselines in SCM [27].

- Release: this is the full and final version of a software system or the system that is implemented.
- Configuration Control Board: a configuration control board is a group of people that are responsible for making a change in a software system. The proposal of change is sent to the configuration control board, which looks at both the negative and positive consequences of a change and on basis of the effect of the change they either approve or disapprove a change. A configuration control board can also give the solution for a change. In smaller projects, a configuration control board made up of testers, programmers, as well as requirement engineers, work informally, while in larger projects a formal configuration control board is in charge of tracking, controlling, and monitoring change [28].

2.3. *Proof of Authority Consensus*

Proof of authority (PoA) is a consensus mechanism for a blockchain platform. Furthermore, permissionless (bitcoin, Ethereum) and permissioned (bitcoin, Ethereum) consensus processes can be found on blockchain platforms (Apla, Ethereum Private). All nodes are pre-authenticated on a permissioned blockchain. This advantage enables the adoption of consensus types that, in addition to other advantages, give a high transaction rate. PoA is a new type of consensus algorithm that is both fast and reliable. In PoA, nodes that have demonstrated their authority to generate new blocks are granted the right to do so.

A node must pass a preliminary authentication to get this power and the ability to create new blocks. If we further look into the advantage of PoA, then we can see that it can help to defend against attacks [29]. If we compare PoA with other commonly used consensus algorithms like proof of stake (PoS) and proof of work (PoW) then, for our problem, PoA is the best choice as it provides security and is energy efficient.

To disrupt the operation of a particular network node and make it unavailable, an attacker sends a high number of transactions and blocks to that node.

- Block generation rights can only be given to nodes that can withstand DoS assaults since network nodes are pre-authenticated;
- A validating node can be removed from the list of validating nodes if it is unavailable for a while.

2.4. Blockchain and Version Control System Structure

If we look at blockchain structure and file structure in a version control system, there is a similarity between both technologies. There is a sheer need for immutability in the version control system as a direct change to file content can cause issues in development and cause a delay in the development of software. Figure 5 shows the blockchain-based version control architecture. The version control system follows a sequential structure in which we have a main trunk or branch that contains the main file, and then we have sub-branches that are being merged into main branch. Blockchain also has a sequence of blocks, and version control systems can naturally be implemented in the blockchain.

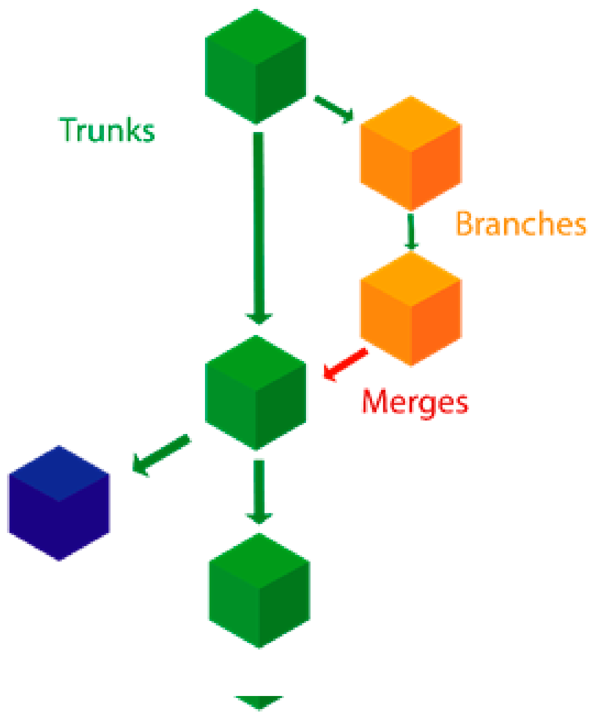


Figure 5. Blockchain-based version control system.

The development cost of the blockchain system is usually high compared with the traditional system, and it requires more computational power to compute the desired hashes by finding and adding the nonce value. However, blockchain technology is flexible with other technologies. We can develop the front-end using C++, Python, and dotnet by developing the blockchain in Ethereum or using the Hyperledger platform, which provides the opportunity for adoption. We can simply write the smart contract and deploy it using these platforms, which reduces costs and allows for efficient systems to be developed in accordance with the marked trends.

3. Related Work

Here we look at the work that has been done by many writers in relation to the usage of blockchain technology for version control, as well as how controlled data exchange of digital documents might be handled.

The authors of one paper [30] present a method and framework for document sharing and version control based on blockchain technology for multi-user collaboration and change tracking in a safe, secure, and decentralized manner without the requirement for a centralized third party or trusted entity. This is a decent approach based on governing and regulating document version control functions among the document's creators and developers, as well as its validators, utilizing Ethereum smart contracts. In addition, the proposed method takes advantage of the benefits of storing documents on an IPFS-based decentralized file system (InterPlanetary File System). Interactions between different users, such as developers and testers, are automated by the suggested approach. Smart contracts are created using the Solidity programming language and the capabilities of smart contracts are tested using the Remix IDE (integrated development environment). The approach was tested and verified. This is the basis for our proposed blockchain-based decentralized architecture for version control.

If we look into the usage/implementation of blockchain in the field of healthcare, we can see that different authors have worked to prove the successful usage of blockchain for decentralized and secure transfer of data among multi-users. Kumar et al. [31] planned off-chain circulated capacity for clinical patient information where the hash of the documents was put away on a consortium blockchain, saving the trustworthiness of the information and security of the patients. The insurance of data by maintaining the honesty of the information was studied by Agyekum et al. [32], where the IPFS and blockchain were consolidated to produce a design zeroed in on ensuring the uprightness of copyright and information security.

One paper [24] presented an approach for a decentralized package version control using blockchain. Blockchain and distributed consensus are popular among cryptocurrencies because no one owns the data or transactions but the coin users. Open-source repositories are similar in that the data belong to the users. The idea of having a repository for software programs in a blockchain with distributed consensus is described in this study, which is also supported by the idea of the demonstrated proof-of-download. In another paper [33], Bell et al. discussed about combining smart contract and blockchain technology; such a system may ensure end-to-end integrity of scientific data and outcomes while also facilitating collaborative research.

The authors of one study [34] used an experimental setting to examine and explain the impact of an access-controlled IPFS. The authors focused on how we could increase file storage space in blockchain, because it is inefficient to store huge files on the blockchain. Because of the block size constraints, files must be divided and reassembled off-chain. This article primarily addressed the need for blockchain applications to transfer larger files, particularly those that include sensitive data. The authors addressed the design and implementation of aclIPFS, which uses Ethereum smart contracts to manage the access control list, for this purpose. A use case for aclIPFS has been given, as well as tested, to demonstrate the system's impact in comparison with the original IPFS.

Other authors [35] concentrated on computerized systems and software that have evolved in the business and allow for the optimization and planning of manufacturing, storing, shipping, selling, and distributing operations. As a proof of concept, the authors created a decentralized and permitted blockchain technology for stockpiling and retrieving global task-scheduling schemes using the Hyperledger Fabric v2 architecture and IPFS off-chain storage, which is enabled by a Hyperledger Fabric v2 framework. Vue.js was used to construct the web interface for this system, which was deployed in Chainstack. This paper is a unique contribution that adds an extra degree of security to scheduling systems. The authors described the newest edition of Hyperledger Fabric, which allows for the establishment of effective tiers of approval and regulations for access to information while maintaining notable levels of security and privacy, preventing the system being deceived or scammed by cybercriminals.

The author of Records Keeper [36] has proposed a non-proprietary, workable blockchain ecosystem with elevated cryptography and blockchain technology that may be used for managing records and document protection. The goal of proposing this high-level solution is to provide a platform that allows for more secure data transport and authorization. With better security, document sharing between groups of users through a decentralized storage network becomes easy. Conventional database platforms like MySQL as well as Oracle do not have a framework for creating immutable records that cannot be tampered with. This paradigm governs document version control, but there is no practical implementation.

If we look at the use of blockchain in the field of management, we can see that the Swedish government [37] is using a management system based on blockchain technology to register and record land papers and land titles in order to record and manage the real estate field. National Lands of Sweden is developing a proof-of-concept model in collaboration with ChromaWay, a blockchain firm. This system will be used to check how blockchain can reduce the possibility of human mistake or errors. The goal of this strategy is to make a system that is more reliable and secure for transmitting papers/data across parties and amending existing documents. This technology that is being proposed can also be used in a smart contract-based system to authenticate the user’s identity.

Query performance is important in considering acquiring data from external sources through blockchain. Zhang et al. propose a middleware system [38] for query performance in blockchain that can save storage while also storing files for blockchain. The current blockchain cannot provide storage version control and multi-person collaboration, and this study presents a solution. Another paper [39] presents an improved P2P file system using IPFS and blockchain. The paper also solves the performance issue in file transfer; a zig-zag-based model is proposed to improve the block storage model.

In recent years, blockchain and artificial intelligence (AI) have received the most research attention. Blockchain is a network of participants sharing a distributed ledger of trustworthy digital records. Blockchain technology has the potential to revolutionize many industries, including international payment, secure data sharing and marketing, and supply chain management. On the other hand, AI is used to create machines that can perform tasks that require intelligence. Blockchain enables AI to scale to provide more actionable insights, manage data usage and model sharing, and create a trustworthy and transparent data economy by providing access to large volumes of data from within and outside the organization [40]. Data are the input for various artificial intelligence (AI) algorithms to mine valuable features, but data on the Internet is scattered everywhere and controlled by various stakeholders who do not trust each other, and data usage in complex cyberspace is difficult to authorize or validate. As a result, enabling data sharing in cyberspace for real big data and real powerful AI is extremely difficult [41]. Artificial intelligence (AI) and blockchain have recently emerged as two of the most popular and disruptive technologies. Blockchain technology has the potential to automate cryptocurrency payments and provide decentralized, secure, and trusted access to a shared ledger of data, transactions, and logs. With smart contracts, blockchain can also govern interactions among participants without the need for an intermediary or a trusted third party. Artificial intelligence, on the other hand, provides intelligence and decision-making capabilities for machines that are similar to humans [42]. “AI and Blockchain as New Triggers in the Education Arena”, promotes dialogue in the promising field of AI and blockchain in the educational sector as new triggers in the development of new approaches to academic management, learning contexts, and AI- and blockchain-based technology applications and tools devoted to education [43].

Blockchain is a cutting-edge technology that has transformed the way culture interacts and trades. It could be defined as a chain of blocks in a fully decentralized network that contains data with digital signatures. The author of [44] examines the use of blockchain in a variety of fields, including finance, healthcare, information systems, wireless communications, the Internet of Things, smart grids, governmental services, security, and smart cities. Smart cities can provide the best services to improve citizens' daily lives in healthcare, mass transit, energy usage, and education. The authors of [45] present cutting-edge blockchain technology to address smart city security issues. Although blockchain offers a valuable solution in a number of fields, by maintaining privacy, security, anonymity, decentralization, and immutability, it also provides valuable benefits to a number of fields and subjects, including the Internet of Things, energy, finance, medical services, and government, which stand to benefit disparately from its implementation [46] Blockchain technology has a wide range of applications. It is also suitable for automated software testing. In [47], an automated testing method for DApps that works in two stages is proposed. To begin, random events are used to derive an abstract relationship between browser-side events as well as blockchain-side contracts. Second, a procedure yields a set of test cases based on inferred relations and arranges the test cases according to a read–write graph. Paper [48] introduces blockchain-oriented software (BoS) testing, particularly at the smart contract level. Concentrating on dynamic testing, it begins by providing a classification of 20 studies based on smart contract code accessibility. Second, it provides an insight into each identified work, emphasizing its benefits and limitations. In this paper, we propose the blockchain-based version control system to promote the education arena more and make the work scalable and stable.

Table 1 shows the list of papers previously published that we have discussed above in detail where version control is merged or implemented with blockchain technology. The majority of research involving large data files makes use of IPFS for file storage with blockchain addresses, one of the main reasons we would use IPFS to implement our proposed system. Integration of blockchain with version control has received limited attention, and the majority of studies have been presented at conferences.

Table 1. Reviewed studies.

Paper	Published in (Journal/Conference)	Method	Platform	Implemented /Proposed	Key Terms
N. Nizamuddin et al. [30]	Journal—Computers & Electrical Engineering	VCS in blockchain	Ethereum	Proposed and implemented	IPFS, Smart contracts
Randhir Kumar et al. [31]	Conference—COMSNETS	Solution for patient data	-	Proposed	IPFS, Medical Research
Kwame Opuni-Boachie Obour Agyekum et al. [32]	Conference—ICIG 2019	Content protection with blockchain	Fabric Alliance Blockchain	Proposed and implemented	Fabric, IPFS, Digital Media
Felipe Zimmerle da N. Costa and Ruy J. Guerra B. de Queiroz [24]	cryptography and security	VCS in blockchain	Custom	Proposed and implemented	Version control, Capivara
Mathis Steichen et al. [34]	Conference—IEEE SmartData	Modified IPFS	Custom	Proposed and implemented	IPFS
Dongcheng Li et al. [35]	Conference—QRS-C	Storage & task scheduling in IPFS & Blockchain	Hyper ledger fabric	Proposed and implemented	IPFS, task scheduling, storage
Zhaiyo Zhang et al. [38]	Conference—ICSP 2021	Query performance for version control using blockchain	Custom	Proposed and implemented	Query performance, version control space
Yongle Chen et al. [39]	Conference—BIGDATA	Throughput improvement	Custom	Proposed and analyzed	P2P, IPFS

4. Proposed Approach

In this section, we explain our proposed approach that uses blockchain for version control, which is an important component of software configuration management. We are using a repository that uses an immutability protocol for distributed file sharing, mainly IPFS. The Inter Planetary File System (IPFS) is a peer-to-peer network and distributed file system technology for data storage [49]. The main quality of IPFS is that it is distributed, and a traditional version control system works on a centralized approach. So, IPFS fits perfectly with our approach of a blockchain-based distributed solution for version control; each block will contain data about one document being uploaded by a developer. Private blockchain will be used with a proof of consensus algorithm. The authority will be responsible for adding developers to the private blockchain.

Following are the major components of our proposed approach.

1. File Repository;
2. Smart Contract;
3. Blockchain;
4. Consensus Algorithm.

The proposed architecture BDA-SCV is modeled in Figure 6. Next, we discuss each component in detail.

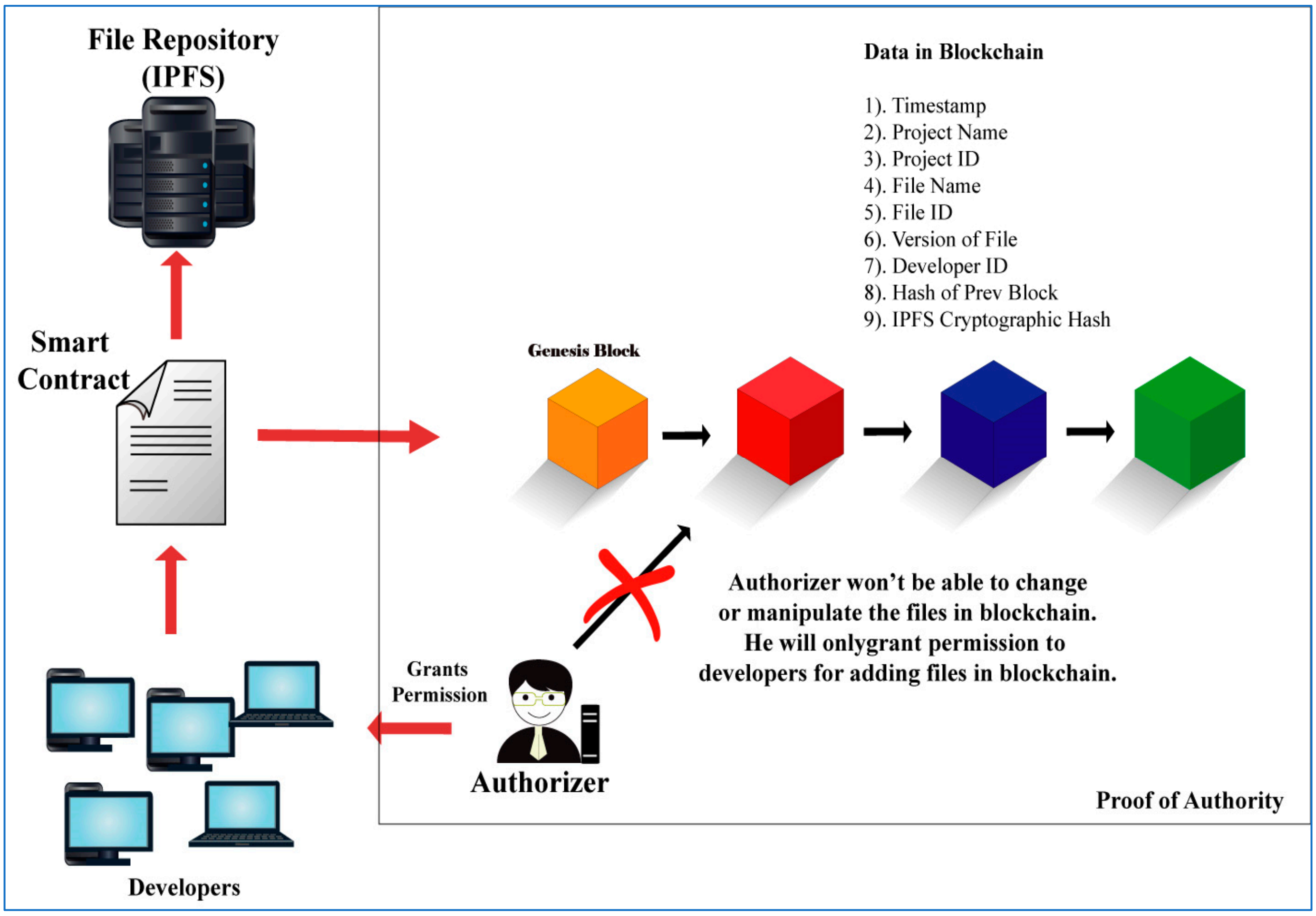


Figure 6. Proposed BDA-SCV Architecture.

4.1. File Repository

As, we have proposed a distributed solution to the centralized approach for version control, we need a distributed repository. IPFS has been proposed for this. IPFS is a decentralized file sharing network that recognizes files based on their content. It is a distributed file system protocol and peer-to-peer network for data storage, and we will be using it for our approach. It employs a distributed hash table (DHT) to determine location of files and nodes integration. In addition, it is not possible and inefficient to store complete code files in blockchain because blockchain can only store limited amounts of data, so we will also give a reference of the file in blockchain that will be stored in the IPFS system. With the reference we will also store the developer ID with which the file and version is associated. A unique cryptographic hash will be generated for each file.

4.2. Smart Contract

Smart contracts, which are stored on a blockchain, are essentially programs that run when specific conditions are met. They are typically used to automate the execution of a contract so that all parties can be certain of the outcome immediately, without the need for any intermediaries or wasted time. They can also automate a workflow by starting the next phase once certain conditions are met [50]. Smart contracts will be used to create new blocks, register users, and provide developers with permission to use a certain project version control blockchain. Everything will be regulated by the system's smart contracts. Uploading documents, which creates a new block for each entry, will be the most important smart contract in our system.

4.3. Blocks Contain the Data

With our distributed repository, we have a self-constructing blockchain with blocks containing data about the project being developed, its version, timestamp and reference to the file stored in IPFS. Following data will be stored in each block.

a. Hash of previous block;
b. Time stamp;
c. Project name;
d. Project ID;
e. File ID;
f. Version of change;
g. Previous version;
h. File name (saved on IPFS);
i. Developer ID.

The blockchain will be constructed based on the hash of the previous block. The first block as we know in blockchain is known as the genesis block. The genesis block in our case will contain some basic information about the project being developed, and the next blocks will contain data specifically related to the changes made. For calculating the hash, Hyperledger Fabric uses the SHA256 algorithm that is also used for hash calculation in bitcoin blockchain [1]. SHA256 is an encryption algorithm that returns a string of fixed length when passed with any amount of data [51]. Our input to the encryption algorithm will be all the above-mentioned data stored in a single block, then the hash of the block will be generated and that hash will be stored in the next block, making a blockchain just like in the link list in which we have pointers working together to make a list. Figure 7 illustrates how our blockchain will work.

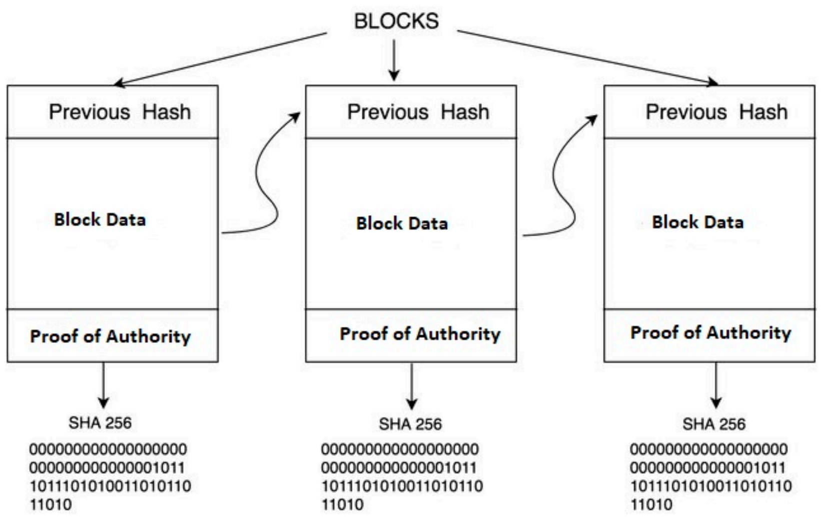


Figure 7. Blockchain structure and consensus algorithms.

Two essential blockchain qualities are immutability and distributed structure, and because the ledger is immutable you can always rely on it to be correct. Its decentralized nature protects it against network threats. On the ledger, each transaction or record is stored in a "block". Blocks on the bitcoin blockchain, for example, typically contain more than 500 bitcoin transactions, so on same structure our block will contain a limited amount of data. A block's data is dependent on and related to the data in the preceding block, forming a chain of transactions across time.

4.4. Consensus Algorithm

For our approach we will be using proof of authority (PoA) consensus algorithm. In PoA-based networks, transactions and blocks are validated by approved accounts, known as validators. Validators run software allowing them to put transactions in blocks. The process is automated, so validators do not have to keep an eye on their computers all the time. It does, however, necessitate keeping the computer (the authority node) secure. Individuals who earn the right to become validators through PoA have an incentive to keep the position they have earned.

5. Implementation

In this section we present the implementation of the model proposed in the above section. Our proposed model is implementing the version control system in blockchain technology and it consists of multiple components, i.e., distributed repository, smart contract, private blockchain, and consensus algorithm. Figure 8 shows the implementation model based on the proposed approach. The implementation is performed on a Hyperledger toolset. Hyperledger are a group of open-source tools for blockchain development, which notably consist of Hyperledger Besu, Fabric and Sawtooth [52]. We use Hyperledger Fabric because it is now the most common tool for developing private blockchain and writing smart contracts. Hyperledger Fabric is a plug-and-play blockchain architecture that serves as a foundation for constructing blockchain-based goods, solutions, and applications for private organizations. Hyperledger Fabric was founded by Digital Asset and IBM and has since evolved into a collaborative cross-industry effort hosted by the Linux Foundation. Fabric was the first Hyperledger project to pass through the "incubation" stage and enter the "active" stage in March 2017 [53].

We have divided our implementation into three major modules, with one consisting of developers who will be responsible for uploading document. Next, we have a Hyperledger Fabric Client and Fabric network. The developers will be connected to the Fabric Client and it will be responsible for communication with the network. The Fabric Client will be hosted on a cloud environment, so that anyone from any place can use our distributed version control system. Authorizers who will be connected directly with the network will be responsible for enrolling developers in the Hyperledger network. Authorizers will not be able to make any change to the blockchain data or files in IPFS and will only have authority to enroll developers into using the version control blockchain using smart contacts. All the programming is done through smart contracts, which are written using JavaScript in the Hyperledger network.

For deployment of our application, we require a platform. For the purpose we have used Docker. It is a platform for creating containers on which applications can be deployed and could easily be integrated in cloud environment [54]. We have deployed both Hyperledger client and server on different containers on Docker and accessed that from a third source, which will act as the developer. We have used the InterPlanetary File System (IPFS) for storing the file versions, and only reference to the file and cryptographic hash against the file will be stored in blockchain as storing large amount of data will not be possible in blockchain. The cryptographic hash will be unique against each file and for change a new file will be stored to the blockchain against the file ID but with a new version.

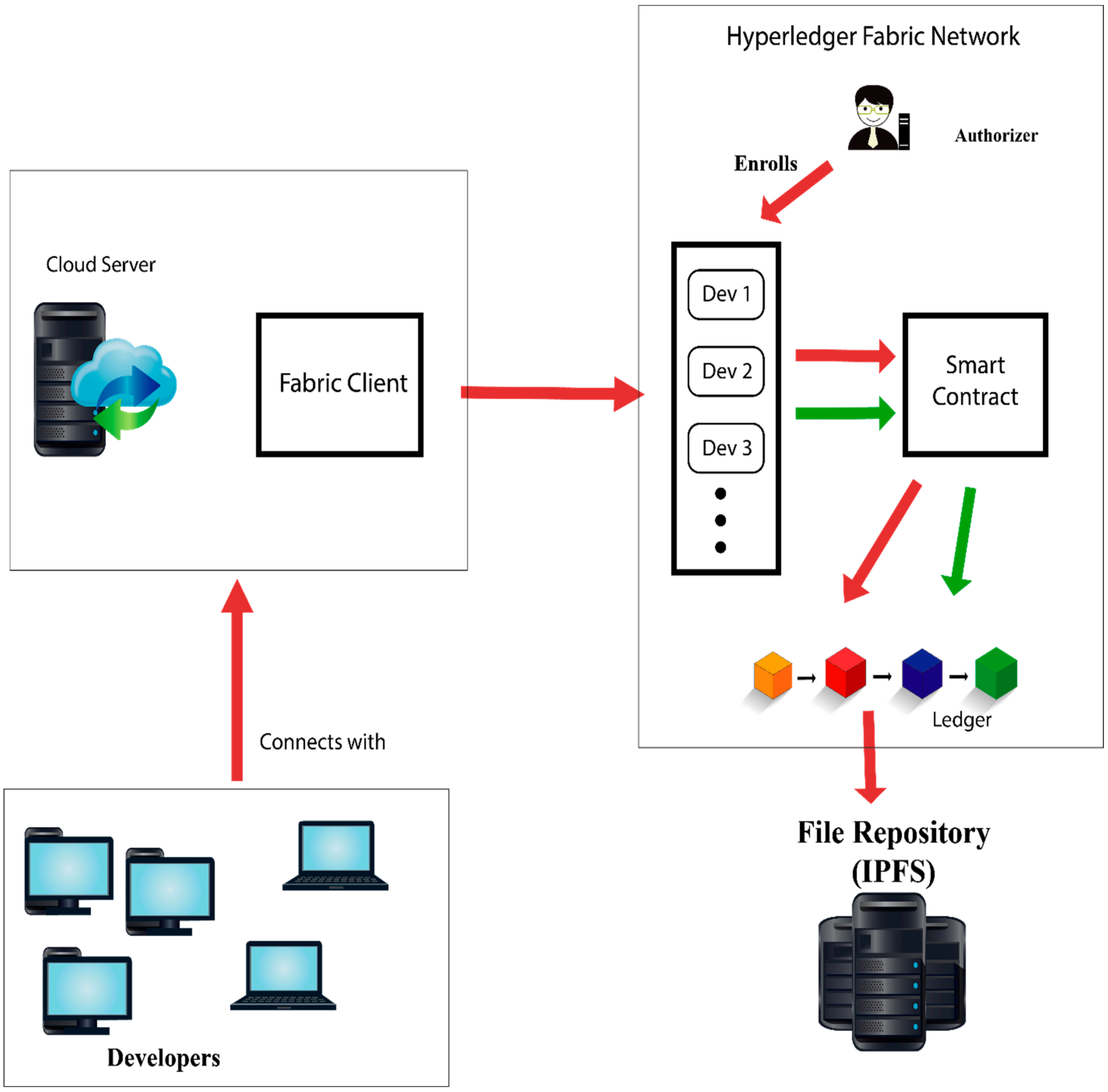


Figure 8. Implementation model for BDA-SCV.

So, the flow of application will be:

1. Developer will connect to Fabric Client;
2. Client will send all requests to Fabric network;
3. Fabric network will be responsible for making all the changes in to ledger;
4. Each block in ledger will be associated to a file placed in distributed repository (IPFS);
5. Cryptographic hash generated for each file in IPFS will be stored in the ledger.

Figure 9 shows the flow of a document being uploaded from the developer's end to IPFS. First of all, the developer will be connected to the Hyperledger Fabric network. The first request will be sent to the Fabric network from where it will be sent to the smart contract implemented in the blockchain network that contains the ledger. First, the ledger will be updated, and then the document will be uploaded to the IPFS repository. A confirmation email will be issued to the developer confirming the successful upload of the file and ledger entry. The developer will then provide the file ID if he wants to add content to the same file.

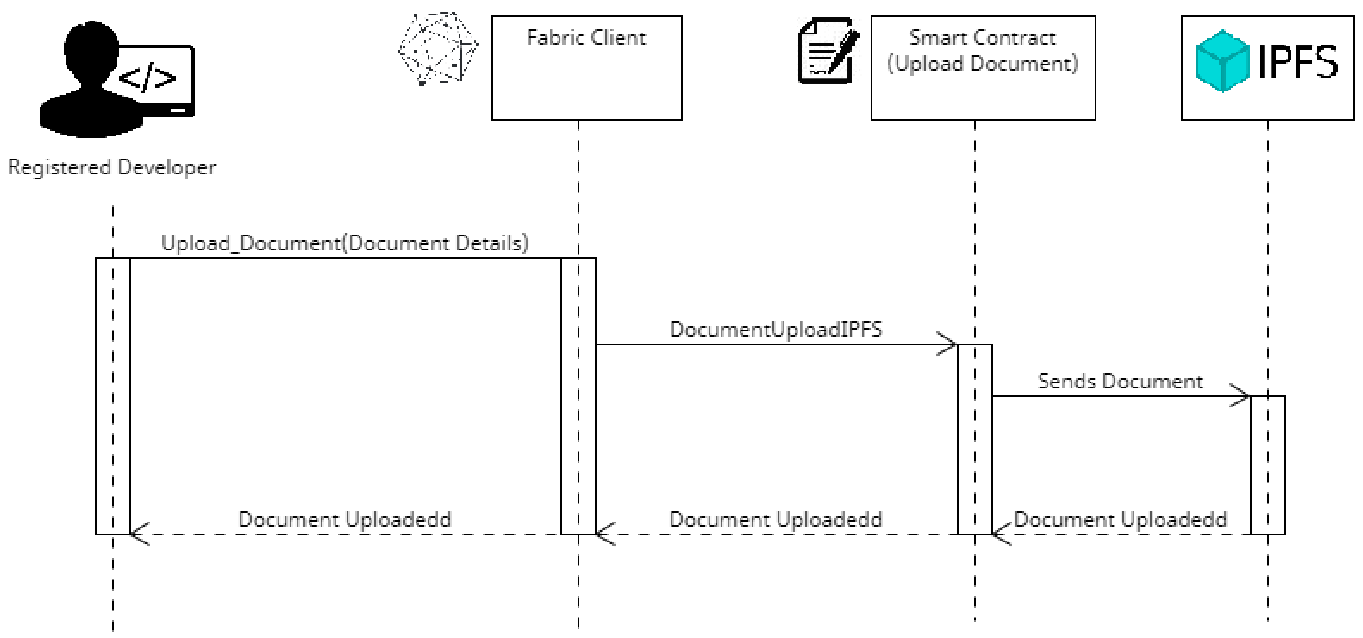


Figure 9. Upload Document SD.

The flow of a new developer joining the project is shown in Figure 10. The authorizer will be responsible for adding new developers to the project and registering them. Figures 11 and 12 also shows the code for this SD. At this time, the developer will be assigned to only one project.

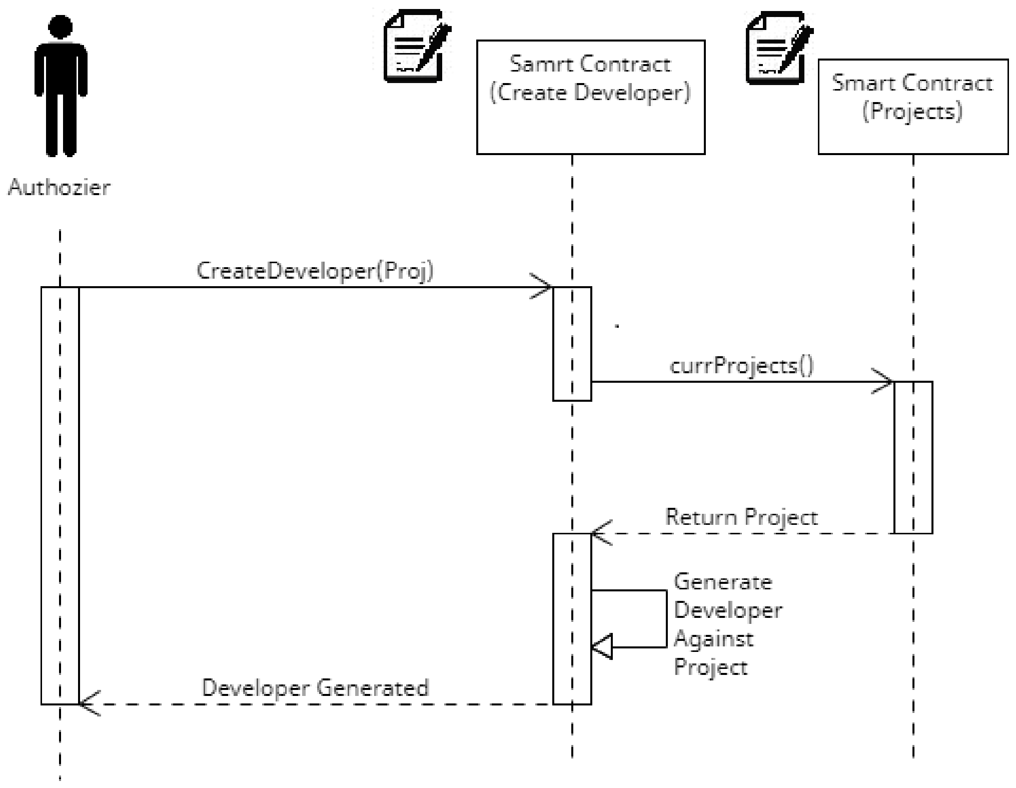
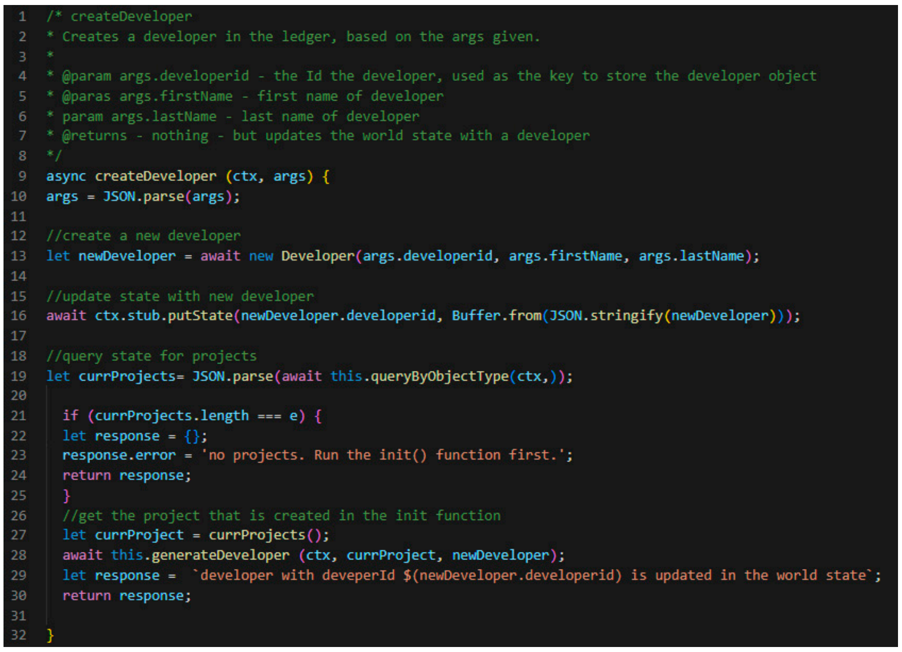

Authozier
Samrt Contract (Create Developer)
Smart Contract (Projects)
CreateDeveloper(Proj)
currProjects()
Return Project
Generate Developer Against Project
Developer Generated

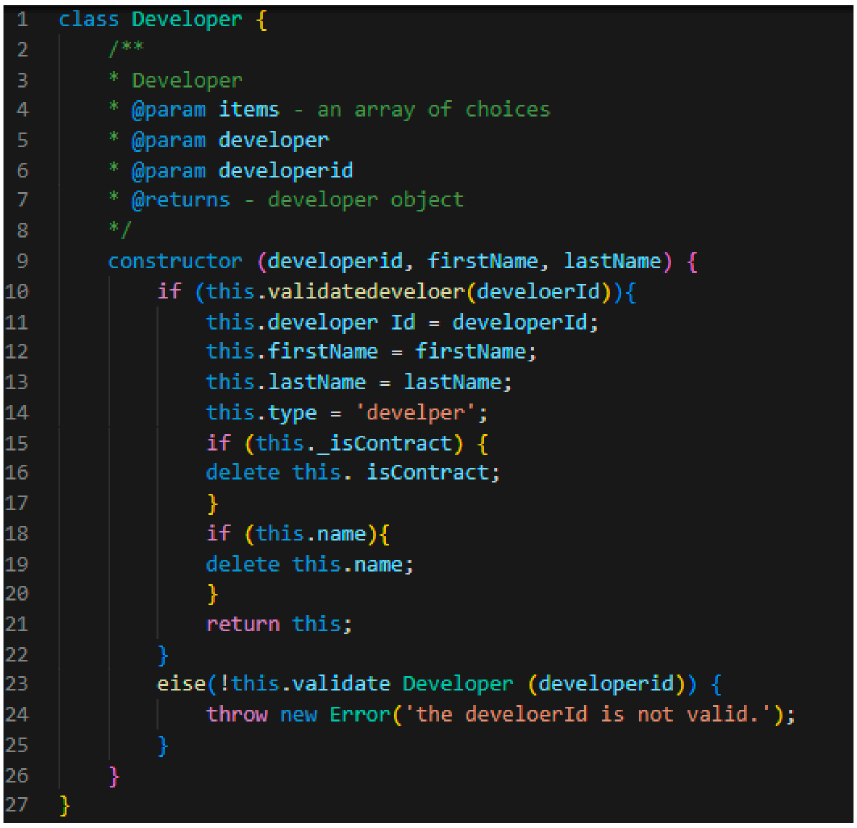
Figure 10. SD for creating developer.

Figures 11 and 12 show smart contracts code for creating a new developer in ledger. The developer ID is used as the key to store the developer object in ledger, and the developer's first name and last name are added with the developer ID. Once this is done, the developer is associated with a project; in the case of our implementation we have considered only one project.



```
/* createDeveloper
* Creates a developer in the ledger, based on the args given.
*
* @param args.developerid - the Id the developer, used as the key to store the developer object
* @paras args.firstName - first name of developer
* param args.lastName - last name of developer
* @returns - nothing - but updates the world state with a developer
*/
async createDeveloper (ctx, args) {
args = JSON.parse(args);

//create a new developer
let newDeveloper = await new Developer(args.developerid, args.firstName, args.lastName);

//update state with new developer
await ctx.stub.putState(newDeveloper.developerid, Buffer.from(JSON.stringify(newDeveloper)));

//query state for projects
let currProjects= JSON.parse(await this.queryByObjectType(ctx,));

  if (currProjects.length === e) {
  let response = {};
  response.error = 'no projects. Run the init() function first.';
  return response;
  }
  //get the project that is created in the init function
  let currProject = currProjects();
  await this.generateDeveloper (ctx, currProject, newDeveloper);
  let response =  `developer with deveperId $(newDeveloper.developerid) is updated in the world state`;
  return response;

}
```

Figure 11. Smart contract for creating a developer.

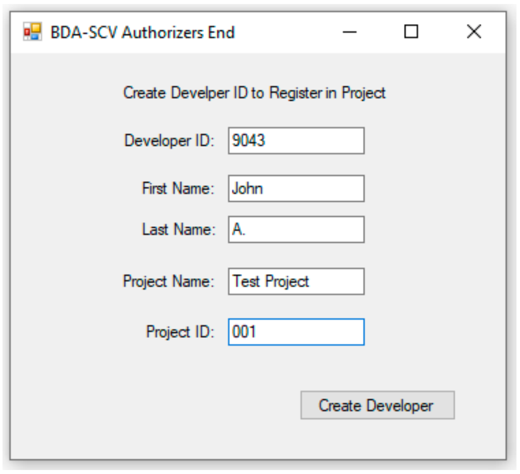


```
class Developer {
    /**
    * Developer
    * @param items - an array of choices
    * @param developer
    * @param developerid
    * @returns - developer object
    */
    constructor (developerid, firstName, lastName) {
        if (this.validatedeveloer(develoerId)){
            this.developer Id = developerId;
            this.firstName = firstName;
            this.lastName = lastName;
            this.type = 'develper';
            if (this._isContract) {
            delete this. isContract;
            }
            if (this.name){
            delete this.name;
            }
            return this;
        }
        eise(!this.validate Developer (developerid)) {
            throw new Error('the develoerId is not valid.');
        }
    }
}
```

Figure 12. Smart contract developer class.

Figure 12 shows the code for creating and validating a new developer on ledger, and after creation of a new developer on ledger the object will be sent back to create the developer method, as earlier shown in Figure 8, where they will be associated with a project. This method is managed by the authorizer, who is responsible for adding and deleting developers on specific project.

We have created the developer end on the dotnet framework from which they can upload files to the Hyperledger network using a restful API call. Figure 13 shows the authorizer end view from which they will register the developer in the ledger. The details that they need to enter are developer ID, first name, last name, project name, and project ID. The smart contracts shown in Figures 11 and 12 will be executed against the view in Figure 13.


BDA-SCV Authorizers End

Create Develper ID to Register in Project

Developer ID: 9043

First Name: John

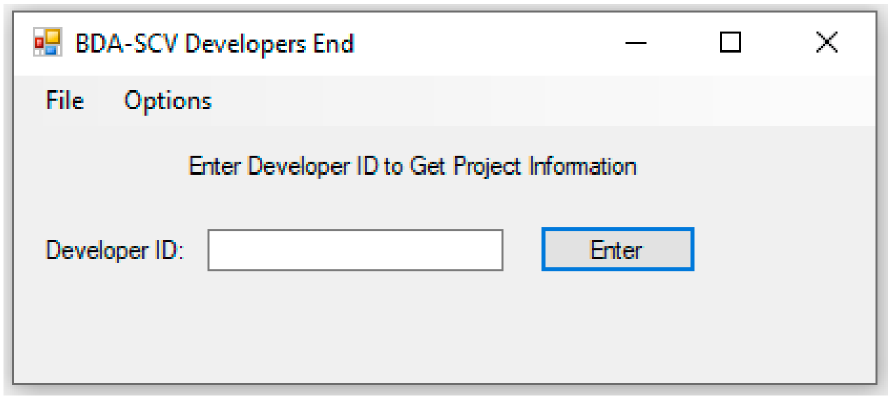
Last Name: A.

Project Name: Test Project

Project ID: 001

Figure 13. BDA-SCV authorizer's end view.

Figure 14 shows the developer end from where they will enter the system to upload files to the network, which will end up in IPFS with a cryptographic hash being stored in the ledger.

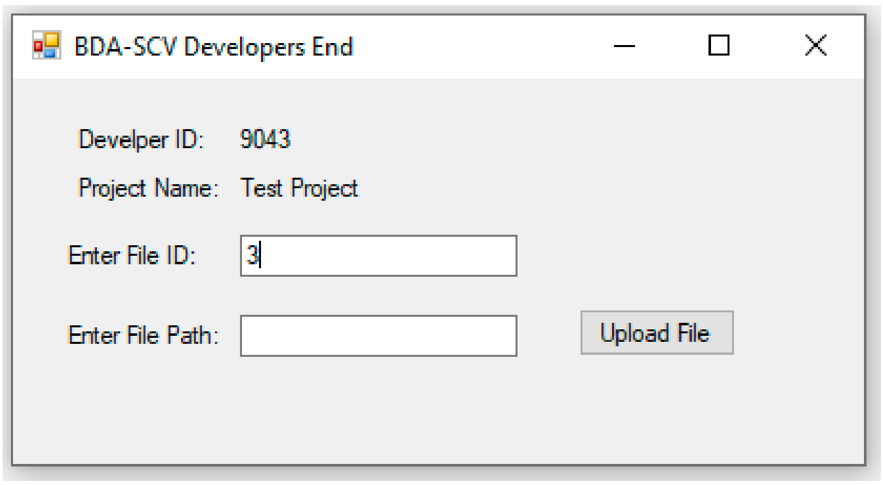

BDA-SCV Developers End

Enter Developer ID to Get Project Information

Developer ID:

Figure 14. BDA-SCV developer's end view.

The file upload view for the developer is shown in Figure 15. The developer will enter the file id and browse the file to add to the network. The developer will enter the file ID that will be stored in the ledger. The version of the file will be automatically created by the smart contract to the ledger.


BDA-SCV Developers End

Develper ID: 9043

Project Name: Test Project

Enter File ID: 3

Enter File Path:

Figure 15. BDA-SCV file upload GUI.

The authenticator will be responsible for creating the project in blockchain and then adding the developers to the specific project. The developer will upload document that will be created as the unique block on the blockchain, with the specific file being stored on IPFS with reference in the block data.

6. Conclusions

Version control is an important part of the software development process. There is a huge amount of version control software available but most of it follows a centralized approach where a single entity can modify or tamper with the files stored. Blockchain is immutable and composed of indestructible blocks connected together by a cryptographic algorithm. Both blockchain and version control systems follow the same pattern for storing block data and versioned files. In this paper we have implemented a version control system on blockchain technology, with immutability. We have also included a decentralized approach by using IPFS storage for the file repository. The file versions will be stored in IPFS, with the file reference being stored in blockchain data. It is not possible to store large amount of data in blockchain so we have considered storing it on a decentralized repository. Smart contracts have been written for authorizing developers for using blockchain, for developer registration, block creation, and repository management. The consensus algorithm used in this approach is the proof of authority (PoA). The implementation is performed on Hyperledger Fabric consisting of a cloud-based Fabric Client and Hyperledger Fabric network, which will have a smart contract and ledger. Developers will be enrolled by being authorized into to the Fabric network. In the future, implementation of the proposed approach using state-of-the-art tools and implementation of other SCM processes over the same architecture will be carried out for empirical analysis. Multiple variants can be created if we look at the current VCS tools; each one has a different functionality and you can choose a VCS based on your team size and budget. This implementation can be performed using the presented architecture for analyzing and trying to propose a more reliable version control system.

Author Contributions: Writing—original draft, M.H., J.I., C.A.u.H., S.H., S.S.U., M.U., U.A.M., M.A. and R.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project Number (PNURSP2023R97), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: Not Applicable.

Acknowledgments: Princess Nourah bint Abdulrahman University Researchers Supporting Project Number (PNURSP2023R97), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. Available online: www.bitcoin.org (accessed on 20 October 2022).
2. Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A.; Enyeart, D.; Ferris, C.; Laventman, G.; Manevich, Y.; et al. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference, Porto, Portugal, 23–26 April 2018; pp. 1–15.
3. Khan, S.; Naheed, S.; Loukil, F.; Ghedira-Guegan, C.; Benkhelifa, E.; Bani-Hani, A. Blockchain smart contracts: Applications, challenges, and future trends. *Peer-Peer Netw. Appl.* **2021**, *14*, 2901–2925. [CrossRef] [PubMed]
4. ul Hassan, C.A.; Hammad, M.; Iqbal, J.; Hussain, S.; Ullah, S.S.; Al Salman, H.; Mosleh, M.A.A.; Arif, M. A Liquid Democracy Enabled Blockchain-Based Electronic Voting System. *Sci. Program.* **2022**, *2022*, 1383007. [CrossRef]
5. Rainer, B.; Christin, N.; Edelman, B.; Moore, T. Bitcoin: Economics, technology, and governance. *J. Econ. Perspect.* **2015**, *29*, 213–238.
6. Xu, M.; Chen, X.; Kou, G. A systematic review of blockchain. *Financ. Innov.* **2019**, *5*, 1–14. [CrossRef]
7. Casino, F.; Dasaklis, T.K.; Patsakis, C. A systematic literature review of blockchain-based applications: Current status, classification and open issues. *Telemat. Inform.* **2019**, *36*, 55–81. [CrossRef]
8. Gürpinar, T.; Guadiana, G.; Ioannidis, P.A.; Straub, N.; Henke, M. The Current State of Blockchain Applications in Supply Chain Management. In Proceedings of the 2021 the 3rd International Conference on Blockchain Technology, Shanghai, China, 26–28 March 2021; pp. 168–175.
9. Tandon, A.; Dhir, A.; Islam, A.N.; Mäntymäki, M. Blockchain in healthcare: A systematic literature review, synthesizing framework and future research agenda. *Comput. Ind.* **2020**, *122*, 103290. [CrossRef]
10. Salah, K.; Rehman, M.H.U.; Nizamuddin, N.; Al-Fuqaha, A. Blockchain for AI: Review and open research challenges. *IEEE Access* **2019**, *7*, 10127–10149. [CrossRef]
11. Sohrabi, N.; Yi, X.; Tari, Z.; Khalil, I. BACC: Blockchain-based access control for cloud data. In Proceedings of the Australasian Computer Science Week Multiconference, Melbourne, Australia, 4–6 February 2020; pp. 1–10.
12. Sohrabi, N.; Tari, Z. On the scalability of blockchain systems. In Proceedings of the 2020 IEEE International Conference on Cloud Engineering (IC2E), Sydney, Australia, 21–24 April 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 124–133.
13. de Figueiredo, S.; Madhusudan, A.; Reniers, V.; Nikova, S.; Preneel, B. Exploring the storj network: A security analysis. In Proceedings of the 36th Annual ACM Symposium on Applied Computing, Virtual, 22–26 March 2021; pp. 257–264.
14. Luo, X. Design and Implementation of Decentralized Swarm Intelligence E-Commerce Model Based on Regional Chain and Edge Computing. *Complexity* **2021**, *2021*, 5595002. [CrossRef]
15. Zhu, Y.; Lv, C.; Zeng, Z.; Wang, J.; Pei, B. Blockchain-based decentralized storage scheme. In *Journal of Physics: Conference Series*; IOP Publishing: Bristol, UK, 2019; Volume 1237, p. 042008.
16. Zheng, Q.; Li, Y.; Chen, P.; Dong, X. An innovative IPFS-based storage model for blockchain. In Proceedings of the 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI), Santiago, Chile, 3–6 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 704–708.
17. Serrano, J.P.; Pereira, R.F. Improvement of it infrastructure management by using configuration management and maturity models: A systematic literature review and a critical analysis. *Organizacija* **2020**, *53*, 3–19. [CrossRef]
18. Software Quality Assurance. Available online: http://www.computer.org (accessed on 24 November 2022).
19. Siegmund, N.; Ruckel, N.; Siegmund, J. Dimensions of software configuration: On the configuration context in modern software development. In Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual, 8–13 November 2020; pp. 338–349.
20. Javapoint. Software Configuration Management Process. Available online: https://www.javatpoint.com/scm-process (accessed on 31 March 2022).
21. Koc, A.; Tansel, A.U. A survey of version control systems. *ICEME* **2011**, *2011*, 1–6.
22. Deepa, N.; Prabadevi, B.; Krithika, L.B.; Deepa, B. An analysis on version control systems. In Proceedings of the 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 24–25 February 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–9.
23. Zolkifli, N.N.; Ngah, A.; Deraman, A. Version control system: A review. *Procedia Comput. Sci.* **2018**, *135*, 408–415. [CrossRef]
24. da N. Costa, F.Z.; de Queiroz, R.J.G.B. Capivara: A decentralized package version control using Blockchain. *arXiv* **2019**, arXiv:1907.12960.
25. Mahmoud, Q.H.; Lescisin, M.; AlTaei, M. Research challenges and opportunities in blockchain and cryptocurrencies. *Internet Technol. Lett.* **2019**, *2*, e93. [CrossRef]

26. Pei, S.; Chen, D. The implementing of software configuration management based on CMM. In Proceedings of the 2009 5th International Conference on Wireless Communications, Networking and Mobile Computing, Beijing, China, 24–26 September 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 1–4.
27. Tao, X. Software configuration management of change control study based on baseline. In Proceedings of the 2010 International Conference on Intelligent Control and Information Processing, Dalian, China, 12–15 August 2010; IEEE: Piscataway, NJ, USA, 2010; pp. 93–96.
28. Mohd, F.S.S.; Bannerman, P.L.; Staples, M. Software configuration management in global software development: A systematic map. In Proceedings of the 2010 Asia Pacific Software Engineering Conference, Washington, DC, USA, 3 November–30 December 2010; IEEE: Piscataway, NJ, USA, 2010; pp. 404–413.
29. Adelin, M.M.; Manolache, S.; Tapus, N. Decision Making using the Blockchain Proof of Authority Consensus. *Procedia Comput. Sci.* **2022**, *199*, 580–588.
30. Nishara, N.; Salah, K.; Azad, M.A.; Arshad, J.; Rehman, M.H. Decentralized document version control using ethereum blockchain and IPFS. *Comput. Electr. Eng.* **2019**, *76*, 183–197.
31. Randhir, K.; Marchang, N.; Tripathi, R. Distributed off-chain storage of patient diagnostic reports in healthcare system using IPFS and blockchain. In Proceedings of the 2020 International Conference on COMmunication Systems & NETworkS (COMSNETS), Bangalore, India, 7–11 January 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–5.
32. Agyekum, K.O.; Xia, Q.; Liu, Y.; Pu, H.; Cobblah, C.N.; Kusi, G.A.; Yang, H.; Gao, J. Digital media copyright and content protection using IPFS and blockchain. In *Image and Graphics: 10th International Conference, ICIG 2019, Beijing, China, August 23–25, 2019*; Springer: Cham, Switzerland, 2019; pp. 266–277.
33. Jonathan, B.; LaToza, T.D.; Baldmitsi, F.; Stavrou, A. Advancing open science with version control and blockchains. In Proceedings of the 2017 IEEE/ACM 12th International Workshop on Software Engineering for Science (SE4Science), Buenos Aires, Argentina, 22 May 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 13–14.
34. Mathis, S.; Fiz, B.; Norvill, R.; Shbair, W.; State, R. Blockchain-based, decentralized access control for IPFS. In Proceedings of the 2018 Ieee International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1499–1506.
35. Dongcheng, L.; Wong, W.E.; Zhao, M.; Hou, Q. Secure storage and access for task-scheduling schemes on consortium blockchain and interplanetary file system. In Proceedings of the 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C), Macau, China, 11–14 December 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 153–159.
36. Records Keeper. Available online: https://www.recordskeeper.com/ (accessed on 24 November 2022).
37. Juliet, M.; Young, A.; Verhulst, S. Addressing transaction costs through blockchain and identity in swedish land transfers. In *Blockchain Technologies for Social Change*; GovLab: New York, NY, USA, 2018.
38. Zhang, Z.; Zhong, Y.; Yu, X. Blockchain storage middleware based on external database. In Proceedings of the 2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP), Xi'an, China, 9–11 April 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1301–1304.
39. Chen, Y.; Li, H.; Li, K.; Zhang, J. An improved P2P file system scheme based on IPFS and Blockchain. In Proceedings of the 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 2652–2657.
40. Ala, E.; Amintoosi, H.; Seno, A.H.; Dehghantanha, A.; Parizi, R.M. A systematic literature review of integration of blockchain and artificial intelligence. *Blockchain Cybersecur. Trust. Priv.* **2020**, 147–160.
41. Wang, K.; Dong, J.; Wang, Y.; Yin, H. Securing data with blockchain and AI. *IEEE Access* **2019**, *7*, 77981–77989. [CrossRef]
42. Al-Breiki, H.; Rehman, M.H.U.; Salah, K.; Svetinovic, D. Trustworthy blockchain oracles: Review, comparison, and open research challenges. *IEEE Access* **2020**, *8*, 85675–85685. [CrossRef]
43. Sousa, M.J.; Dal Mas, F.; Gonçalves, S.P.; Calandra, D. AI and Blockchain as New Triggers in the Education Arena. *Eur. J. Investig. Health Psychol. Educ.* **2022**, *12*, 445–447. [CrossRef]
44. Krichen, M.; Ammi, M.; Mihoub, A.; Almutiq, M. Blockchain for modern applications: A survey. *Sensors* **2022**, *22*, 5274. [CrossRef] [PubMed]
45. Bhushan, B.; Khamparia, A.; Sagayam, K.M.; Sharma, S.K.; Ahad, M.A.; Debnath, N.C. Blockchain for smart cities: A review of architectures, integration trends and future research directions. *Sustain. Cities Soc.* **2020**, *61*, 102360. [CrossRef]
46. Abou Jaoude, J.; Saade, R.G. Blockchain applications–usage in different domains. *IEEE Access* **2019**, *7*, 45360–45381. [CrossRef]
47. Gao, J.; Liu, H.; Li, Y.; Liu, C.; Yang, Z.; Li, Q.; Chen, Z. Towards automated testing of blockchain-based decentralized applications. In Proceedings of the 2019 IEEE/ACM 27th International Conference on Program Comprehension (ICPC), Montreal, QC, Canada, 25–26 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 294–299.
48. Lahami, M.; Maâlej, A.J.; Krichen, M.; Hammami, M.A. A Comprehensive Review of Testing Blockchain Oriented Software. Proceedings of 17th International Conference on Evaluation of Novel Approaches to Software Engineering ENASE, Online Streaming, 25–26 April 2022; pp. 355–362.
49. Amritha, P.P. A Blockchain and IPFS based framework for secure Research record keeping. *Int. J. Pure Appl. Math.* **2018**, *119*, 1437–1442.

50. Wang, S.; Yuan, Y.; Wang, X.; Li, J.; Qin, R.; Wang, F.-Y. An overview of smart contract: Architecture, applications, and future trends. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Suzhou, China, 26–30 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 108–113.
51. Lawrence, S.A.; Ganadhas, C.S. The evaluation report of sha-256 crypt analysis hash function. In Proceedings of the 2009 International Conference on Communication Software and Networks, Chengdu, China, 20–22 February 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 588–592.
52. Nasir, Q.; Qasse, I.A.; Talib, M.A.; Nassif, A.B. Performance analysis of hyperledger fabric platforms. *Secur. Commun. Netw.* **2018**, *2018*, 3976093. [CrossRef]
53. Amazon. Hyperledger Fabric. Available online: https://aws.amazon.com/blockchain/what-is-hyperledger-fabric/ (accessed on 27 October 2022).
54. Bashari, R.B.; Bhatti, H.J.; Ahmadi, M. An introduction to docker and analysis of its performance. *Int. J. Comput. Sci. Netw. Secur.* **2017**, *17*, 228.