Decentralized Blockchain-based platform for collaboration in virtual scientific communities

Lenko V.[1], Kunanets N.[1], Pasichnyk V.[1] , and Shcherbyna Yu[2]

[1]Department of Information Systems and Networks, Institute of Computer Science and Information Technologies, Lviv Polytechnic National University
Mytropolyta Andreja 5, UA 79013 Lviv, Ukraine
e-mail: vs.lenko@gmail.com

[2]Department of Discrete Analysis and Intelligent Systems, Faculty of Applied Mathematics and Informatics, Ivan Franko National University of Lviv
Universytetska 1, UA 79 000 Lviv, Ukraine
e-mail: yshcherbyna@yahoo.com

Received January 10.2019: accepted February 20.2019

Abstract. The paper presents a modern approach to the construction of collaboration platforms in virtual scientific communities, which is based on the ideas of decentralization and crypto security. The risks and disadvantages of existing collaboration solutions are considered, as well as possible ways of their elimination. Research highlights cornerstone technologies of the platform, in particular, peer-to-peer network, the blockchain, distributed hosting, self-sovereign identity and interconnections between them.

Keywords: distributed ledger, decentralized file system, Git, self-sovereign identity, smart contracts

INTRODUCTION

Modern scientific research goes beyond the usual scientific institutions and often requires the involvement of experts from different institutional and geographical locations. A virtual scientific team is a situational group of representatives of various scientific institutions which, through the systematic use of information and communication technologies, virtually unite for joint research and implementation of integrated scientific projects. Effective collaboration within a virtual research team is one of the key factors for a successful implementation of the project. As it mostly happens in a digital manner, there is a strong need for a reliable, private and user-friendly software solution.

The establishment of peer-to-peer networks, improvements in cryptography and hashing, the emergence of Blockchain [1] and InterPlanetary File System (IPFS) [2] have created a solid technological background for a decentralized architecture of collaboration systems. Project collaboration typically involves the cooperation of multiple parties, which might be geographically or institutionally distributed, with different roles and obligations. It is common to track and share project assets state via a version control system, however, their synchronization is still performed through a centralized service. While blockchain offers unique advantages over centralized data stores, it suffers from certain limitations related to the size of supplied data. Collaboration in virtual scientific communities involves sending both short messages and massive data files, therefore there is a need for IPFS technology, which ensures reliable and efficient integration of large files into blockchain. An analysis of the market for the presence of existing software solutions showed the lack of electronic collaboration platforms that would combine the proposed components.

PROBLEM STATEMENT

In the present, applications developed on the principle of client-server architecture [3] (Gmail, Skype, Viber, Basecamp, Web systems) are most often used for communication in virtual scientific groups. The disadvantage of this approach is the need to direct information flows through a server component owned by third parties and at risk of inaccessibility, censorship, loss of privacy, low speed, etc. Existing developments in the field of crypto security [4], distributed databases and networks allow eliminating or minimizing the above-mentioned risks and create new perspectives in the field of virtual scientific communities' collaboration.

While centralization poses a risk to a project, shared collaboration introduces a risk to the project contributors. Typically, dozens of artifacts of different versions are produced during the lifecycle of a project and some of them might possess a unique value. Naturally, project contributors would like to have the technical opportunity to prove authorship of the important artifacts, as well as manage system access privileges and artifacts licensing policies more granularly.

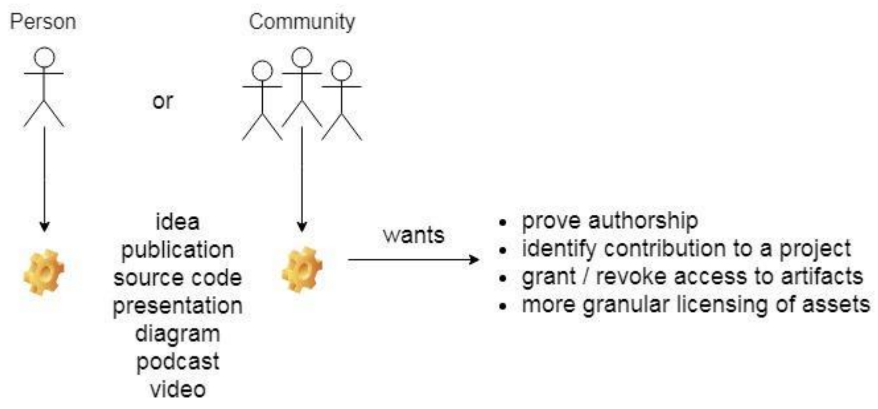
Person
or
Community
idea
publication
source code
presentation
diagram
podcast
video
wants
• prove authorship
• identify contribution to a project
• grant / revoke access to artifacts
• more granular licensing of assets

Fig. 1. Concerns related to virtual collaboration

RELATED WORKS

The rise of peer-to-peer content-based systems and distributed ledger technology laid the foundation for the numerous advancements in the areas of digital security and content sharing. In particular, blockchain technology enhanced with the implementation of a smart contract concept [5], facilitates the development of reliable distributed registries. Content-based addressing in Git [6] or IPFS, backed by a peer-to-peer content distribution network, and the unified naming protocol IPLD [7] creates implementation opportunity for truly independent and decentralized collaboration platforms.

Digital identity concept has been around for a couple of years, while its W3C draft specification has appeared only recently [8]. Among the most notable applications in this field is uPort project [9], which made a significant contribution to the specification of DID concept and implementation of self-sovereign identity (SSI) architecture in Ethereum environment. A well-established infrastructure for decentralized identity management is provided by the Hyperledger Indy project [10], which is an open source, purpose-built distributed leger from the Linux Foundation [11].

BLOCKCHAIN

If there was a requirement to describe a blockchain with a single word, it would be ‘decentralized’. This characteristic is the essential one for a multi-agent trustless environment since it forms the physical basis for trust. Asymmetric cryptography, hashing, consensus protocol, smart contracts – all of them contribute to trust as well, but in a logical manner. It turns out that ‘trust in the specified behavior of a system’ represents a competitive edge of the blockchain technology, which is supposed to disrupt current business models through disintermediation. Reliability of the system is accomplished via transparent decentralization, open source code, digital signatures, consensus protocol, tracking and immutability of the transactions. Having shaped the general mission of a blockchain, it is time to give its definition. According to the Hyperledger project [12], blockchain is a peer-to-peer distributed ledger, forged by a consensus, combined with a system for smart contracts.

Peer-to-peer. The set of all blockchain nodes comprise a peer-to-peer (P2P) network. ‘Client-server’ communication doesn’t exist here since there is no central server. Instead, each node acts both as a ‘client’ (sends requests to the network) and as a ‘server’ (handles requests from the network). To join the network a participant has to download, install and run a node application. On the initial run, a node connects to the ‘bootstrap’ nodes, which are specified in the source code, and acquires the addresses of other peers in the network. Then, peers discovery is typically performed by the means of a distributed hash table (DHT) system [13]. Depending on the domain of a blockchain application, a peer-to-peer network might be public (permissionless) or private (permissioned). In a public network, like Bitcoin or Ethereum, everyone is allowed to join and operate on a blockchain. Availability of peers in a public network and their ‘well-behavedness’ are economically incentivized through the remuneration in cryptocurrency. In a private network, like Hyperledger Fabric, pre-verification of the nodes is required. Since the participating parties tend to know each other and commit to their responsibilities in the form of an agreement, there is typically no need for the use of a cryptocurrency in a private blockchain.

Distributed ledger. The blockchain is a type of distributed ledger where the blocks of data are stored in a linear chain. ‘Distributed ledger’ term denotes a consensus of replicated, shared and synchronized digital data geographically spread across multiple sites, countries, or institutions [14]. So, how does it work? In general, every node stores a data model that captures the current state of a ledger. The history of states is stored as a chain of blocks in a database file. Once a new block is mined, the mining node propagates it to the P2P network and other nodes append it to the block with the highest index in their local data model. This process is called replication. Having thousands of mining nodes in the network, it is quite possible that two blocks are generated approximately at the same time at different locations and are assigned the same index number. Since it takes some time to propagate the block through the P2P network, various nodes will end up with the different versions of the latest block. This blockchain state is called a fork. A fork is resolved with a 'fork selection rule'. The idea is to wait until block gets a number of confirmations, where each confirmation is a new block appended to its chain. Then depending on the algorithm, the 'heaviest' or 'longest' chain is selected as the main chain. The blocks which reside in a defeated chain are disregarded and their transactions return back to transactions pool. In Ethereum the orphaned block can still be included into the main chain as 'ommer' block during the process of mining of seven successor blocks [15].

Consensus protocol. Blockchain can be modeled as a state transition system, where each transaction is an event that changes the state of a machine. The role of a consensus protocol is to ensure that the new state is valid. This is accomplished via the set of rules which validate the correctness of transaction (signature, account balance, hash), well-formedness of a block (proof of work, timestamp, hash) and fork selection. In addition, to prevent the abuse of a network various kinds of consensus algorithms are utilized: proof of work (PoW) [16], proof of stake (PoS), proof of elapsed time (PoET), etc. Proof of work algorithms consume a lot of computational resources and introduce an issue of network scalability, therefore most modern public blockchains are switching to the proof of stake algorithms. The idea behind the PoS consists of depositing an amount of cryptocurrency to a smart contract address and selection of node for block validation process based on the size of a stake. If a node is suspected in the malicious activity, it may lose its stake. This approach imposes better security, reduced risk of centralization, energy efficiency and network scalability. In general, the main concern of the consensus protocols is security. They constantly evolve and are designed to protect against the well-known and potential vectors of attack which include: tampering, double spending, DDoS attack, bribing attack, Sybil attack, ASIC hardware, factorization (Quantum computing), etc.

Smart contracts. Distributed ledger technology (DLT) consists of a distributed ledger and system for smart contracts. A smart contract is an executable computer program that is deployed to a blockchain via transaction and represents a separate account with its own address and state. Once a smart contract is broadcasted to the network, its execution can be triggered by sending a transaction with appropriate data to smart contract's address. The role of a blockchain is to provide a trusted environment for running the program. So what behavior can be programmed? It depends on the platform and language. Ethereum supports a nearly Turing-complete language Solidity which allows encoding any computable function. The word 'nearly' reflects the notion of 'gas', which is used to prevent the halting problem in Ethereum Virtual Machine [17]. Ethereum smart contract has two phases of execution: initialization and message-based call. The first one is activated during the deployment of a contract and aims to set up a valid environment for its execution, while the latter is triggered upon the transaction to the contact's address. In both cases, a contract may update its state, create new smart contracts, send a transaction to other accounts – do any computations that are specified in its code. A very famous example of a smart contract is 'DAO' [18], which was hacked through the exploit in a 'fallback' function. This case reveals a strong need for reliable correspondence between the specification and implementation of a program, thus creating the niche for formal verification.

Restrictions. Blockchain offers unique advantages over centralized databases, but it also suffers from certain limitations that have to be taken into account. Since communication in the virtual science team involves sending both chunks of data and massive files, there is a need for a technology that would ensure reliable and efficient integration of large files into Blockchain. For the sake of this InterPlanetary File System is the right choice. IPFS is a distributed file system, in which each file and its blocks are associated with the unique cryptographic hash-code that is calculated based on the underlying content. Such structuring prevents the duplicates on the network and further allows you to do the versioning of any file. The integration of IPFS with Blockchain consists of writing the hash code into a dedicated smart contract registry.

Private blockchain. Public or permissionless blockchains like Ethereum serve as a good initial foundation for the development of DApps, as they encode the principles of decentralization, immutability, and transparency into their protocol. Anyone who is willing to participate in consensus can join the network, run full node, make transactions, audit blockchain, leave the network, etc. The price for this degree of freedom is paid with a low transaction throughput (TPS), lack of privacy and transaction fees. While it is possible to mitigate most of these issues by switching a consensus protocol, referencing hash of sensitive data, public key encryption of transaction payload, minimization of traceability with off-chain operations, still immutability, validity, accessibility, and visibility of data in distributed ledger significantly depends on the network of anonymous peers.

Private or permissioned blockchains like Hyperledger Fabric tackle the task of distributed consensus in a different manner: anyone who is willing to participate in blockchain has to be authenticated and authorized for intended activities. Since each peer, its privileges and responsibilities are easily identifiable and verifiable; the risks of peer misbehavior or network unavailability are significantly reduced, thus eliminating the need for incentivization and transaction fees. In addition, a group of authorized peers, which participate in a designated activity, may reach the consensus over its own transactions internally, without the disclosure of any details to the rest of a network, therefore ensuring privacy and high performance in reaching the consensus. Described attractiveness of private blockchains comes at a price of quasi-central authority presence and requires the establishment of a dedicated operational infrastructure.

SYSTEM ARCHITECTURE

Specification and implementation of the promising idea in the form of a project is a continuous process, which typically produces dozens of digital assets of different versions from multiple contributors. The issue of digital assets versions management is a well-known task in the field of software engineering with already established industry standards like Git, a distributed version control system for tracking changes in files and coordinating work in a team. Once the idea, code or paper (preferably modular LaTeX document) is digitally encoded, it is committed to local Git repository and later pushed to a centralized repository like Github or Bitbucket [19]. While author credentials are attached to each commit, the history of Git repository can be rewritten with ‘git rebase’ command, thus introducing a risk for the original author.

Refusing to ignore authorship risk and taking into account potential non-disclosure requirements of a project, it is proposed to timestamp commit hash in a blockchain transaction to a dedicated smart contract registry [Fig. 2]. Since Git uses Merkle-tree [20] data structure for hashing the content of commit object, and the hash of commit uniquely identifies its content without revealing any confidential information, it is safe to include commit hash into public blockchain transaction. For technical convenience, it is advised to create a post-commit ‘Git hook’ in a local repository, supplied with the required tokens for signing (Private key) and publishing (Infura token [21]) a transaction to Ethereum smart contract. As each transaction in the Ethereum main network is paid, an author might choose to avoid timestamping commits with low importance. Finally, when a transaction with a commit hash code is successfully mined and confirmed with at least twelve blocks [22], the commit might be pushed to a centralized Git repository. To prove the authorship of a specific commit, an author should maintain his private key from blockchain account safely.

Proposed architecture successfully mitigates the authorship risk, yet it doesn’t solve the issue of centralization. An emergence of InterPlanetary Linked Data (IPLD) protocol and its tight connect to IPFS content-based network allows truly distributed hosting of Git repository. Dedicated IPLD codec for Git [23] is designed to transform Git objects into IPLD graph, which can be later traversed using IPFS commands. Each node in the graph is labeled with a self-describing content-addressed identifier (CID) that can be used for referencing corresponding Git object. Updated system architecture [Fig. 3] envisions blockchain timestamping of commit object CID instead of commit hash and storing of commit IPLD graph into IPFS network. Since IPFS is a public network, the satisfaction of non-disclosure requirements might require the establishment of a private IPFS cluster with whitelisted nodes.

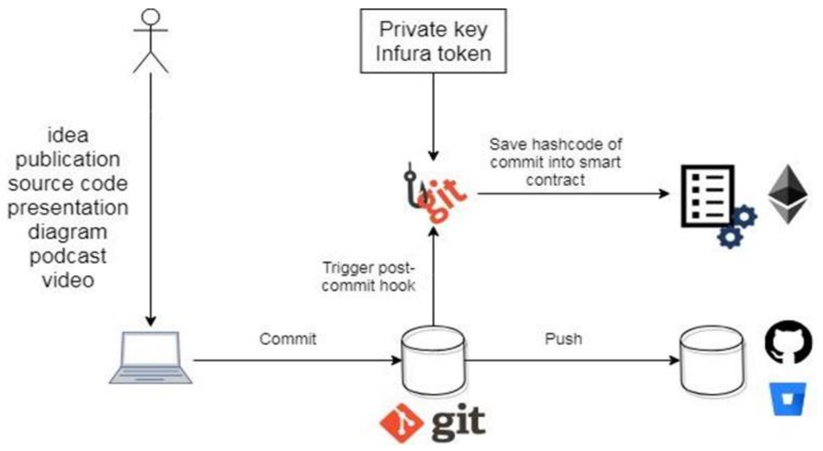

Fig. 2. Blockchain timestamped commits with the centralized repository

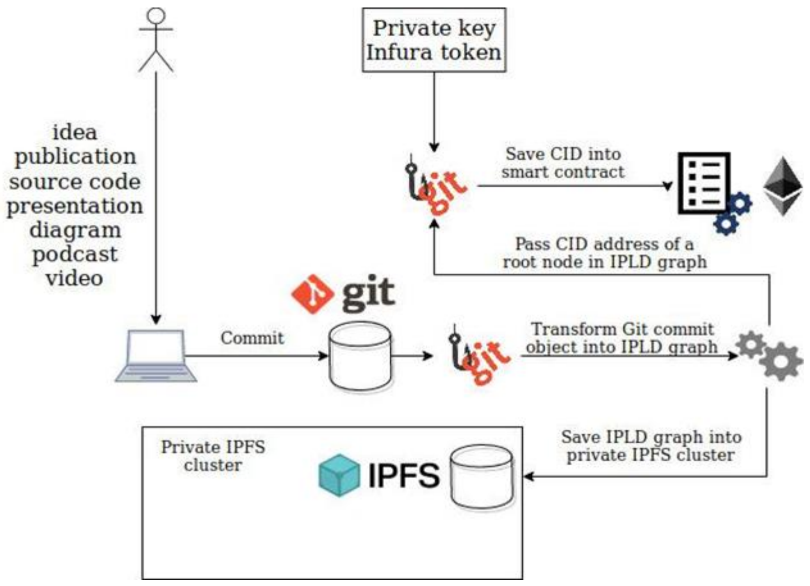
Private key
Infura token
idea
publication
source code
presentation
diagram
podcast
video
Save CID into smart contract
Pass CID address of a root node in IPLD graph
git
Commit
Transform Git commit object into IPLD graph
Private IPFS cluster
IPFS
Save IPLD graph into private IPFS cluster

Fig. 3. Decentralization of Git repository on IPFS

One more enhancement comes from the field of system authorization and relies on the concept of self-sovereign identity [24]. Centralized storage and maintenance of the list of peer nodes, which are granted access to private IPFS cluster, can be substituted with a distributed authorization mechanism backed by decentralized identifiers (DIDs). System authorization requires the presence of well-established DID resolver and properly configured authorization service endpoint. Authorization flow typically starts with DID authentication, where DID has to sign a random challenge with its own private key, for the sake of identity ownership confirmation [Fig. 4]. If the authentication handshake is successful, the DID might request a Verifiable Claim [25] document with the endpoint signature that will grant access to service after being countersigned by the recipient's private key. From a technical perspective, granting of authentication and authorization privileges requires registration of project coordinator DID as a delegate in project DID. Research of the existing platforms for decentralized identities management and collaboration has revealed an open source, purpose-built distributed ledger project Hyperledger Indy [10].

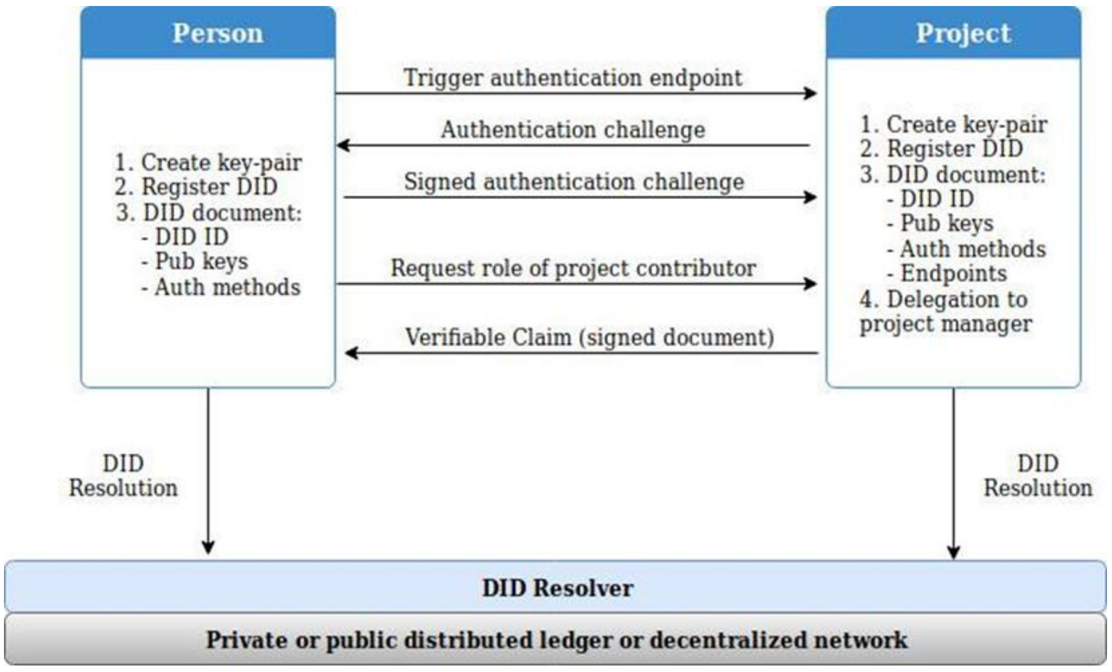

Fig. 4. System authorization using decentralized identifiers

CONCLUSIONS

This article presents a modern approach to the design of collaboration platforms within virtual scientific communities based on the ideas of crypto security and decentralized storage. The technological stack of proposed collaboration platform consists of a distributed version control system Git used for assets versions tracking, Ethereum blockchain with a dedicated smart contract for commits timestamping, distributed content-addressed file system IPFS for assets sharing, IPLD data model for encoding Git objects, and Hyperledger Indy distributed ledger for decentralized identities management. Presented figures specify an integration of the aforementioned technological components into a holistic software system.

Further research consists in the extension of the proposed system architecture with knowledge management module, which provides type-theoretical encoding and semi-automated inference in ontologies [26], and incorporation of peer-to-peer chat module like Orbit [27]. Implementation of the proposed architecture should mostly rely on open source solutions.

REFERENCES

1. **Werbach K. 2018.** The Blockchain and the New Architecture of Trust. The MIT Press, Cambridge, MA.
2. **Benet J. 2014.** IPFS - Content Addressed, Versioned, P2P File System. https://github. com/ipfs/papers/raw/master/ipfs-cap2pfs/ipfs-p2p-file-system.pdf.
3. **Leibnitz K., Hoßfeld T., Wakamiya N., Murata M. 2007.** Peer-to-peer vs. client/server: reliability and efficiency of a content distribution service. Lecture Notes in Computer Science Managing Traffic Performance in Converged Networks, 4516, 1161–1172.
4. **Bertoni G., Daemen J., Peeters M., Van Assche G. 2012.** Permutation-Based Encryption, Authentication and Authenticated Encryption. Directions in Authenticated Ciphers, 159–170.
5. **Shabo N. 1997.** Smart Contracts: Formalizing and Securing Relationships on Public Networks. First Monday, 2(9).
6. **Chacon S., Straub B. 2014.** Pro Git (2nd edition). Apress, New York, NY.
7. InterPlanetary Linked Data (IPLD) – The data model of the content-addressable web. https://ipld.io.
8. Data Model and Syntaxes for Decentralized Identifiers (DIDs). https://w3c-ccg.github.io/did-spec.
9. uPort – Open Identity System for the Decentralized Web. https://www.uport.me.
10. Hyperledger Indy – distributed ledger for decentralized identity. https://www.hyper ledger.org/projects/hyperledger-indy.
11. The Linux Foundation – a non-profit technology consortium. https://www.linux foundation.org.
12. Hyperledger – Open Source Blockchain Technologies. https://www.hyperledger.org.
13. **Maymounkov P., Mazières D. 2002.** Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. Peer-to-Peer Systems Lecture Notes in Computer Science, 53–65.
14. UK Government Chief Scientific Adviser. Distributed Ledger Technology: beyond block chain. https://assets.publishing.service.gov.uk/governmen t/uploads/system/uplo ads/attachmeat_data/file/492972/gs-16-1-distributed-ledger-technology.pdf.
15. **Ritz F., Zugenmaier A. 2018**. The Impact of Uncle Rewards on Selfish Mining in Ethereum. https://arxiv.org/abs/1805.08832.
16. **Jakobsson M., Juels A. 1999.** Proofs of Work and Bread Pudding Protocols. Secure Information Networks, 258–272.
17. Ethereum: A Secure Decentralised Generalised Transaction Ledger. https://ethereum. github.io/yellowpaper/paper.pdf.
18. The DAO: A Million Dollar Lesson in Blockchain Governance. https://digi.lib. ttu.ee/i? 9460.
19. Bitbucket: Web-based version control repository hosting service. https://bitbucket.org.
20. **Niaz M., Saake G. 2015.** Merkle hash tree based techniques for data integrity of outsourced data. GvD, 66–71.
21. Infura: Scalable Blockchain Infrastructure. https://infura.io.
22. **Buterin V. 2015.** On Slow and Fast Block Times. https://blog.ethereum.org/2015/09/ 14/on-slow-and-fast-block-times.
23. Git ipld format. https://github.com/ipfs/go-ipld-git.
24. **Der U., Jähnichen S., Sürmeli J. 2017**. Self-sovereign Identity − Opportunities and Challenges for the Digital Revolution. https://arxiv.org/abs/1712.01767.
25. Verifiable Credentials Data Model. https://w3c.github.io/vc-data-model.
26. **Lenko V., Pasichnyk V., Kunanets N., Shcherbyna Y. 2018.** Knowledge Representation and Formal Reasoning in Ontologies with Coq. Advances in Computer Science for Engineering and Education, 759–770.
27. Orbit: Distributed, Serverless, Peer-to-Peer Chat Application on IPFS. https://orbit. chat.