Fabchain: Managing Audit-able 3D Print Job over Blockchain

Ryosuke Abe
Graduate School of
Media and Governance
Keio University
Kanagawa, Japan
chike@sfc.wide.ad.jp

Shigeya Suzuki
Graduate School of
Media and Governance
Keio University
Kanagawa, Japan
shigeya@sfc.wide.ad.jp

Kenji Saito
Graduate School of
Business and Finance
Waseda University
Tokyo, Japan
ks91@aoni.waseda.jp

Hiroya Tanaka
Faculty of Environment
and Information Studies
Keio University
Kanagawa, Japan
htanaka@sfc.keio.ac.jp

Osamu Nakamura
Faculty of Environment
and Information Studies
Keio University
Kanagawa, Japan
osamu@sfc.wide.ad.jp

Jun Murai
Keio University
Tokyo Japan
junsec@sfc.wide.ad.jp

Abstract—**Improvements in fabrication devices such as 3D printers are becoming possible for personal fabrication to freely fabricate any products. To clarify who is liable for the product, the fabricator should keep the fabrication history in an immutable and sustainably accessible manner. In this paper, we propose a new scheme, "Fabchain," that can record the fabrication history in such a manner. By utilizing a scheme that employs a blockchain as an audit-able communication channel, Fabchain manages print jobs for the fabricator's 3D printer over the blockchain, while maintaining a history of a print job. We implemented Fabchain on Ethereum and evaluated the performance for recording a print job. Our results demonstrate that Fabchain can complete communication of a print job sequence in less than 1 minute on the Ethereum test network. We conclude that Fabchain can manage a print job in a reasonable duration for 3D printing, while satisfying the requirements for immutability and sustainability.**

Index Terms—**Digital Fabrication, Fabrication History, Blockchain**

I. INTRODUCTION

During the past decade, digital fabrication devices such as 3D printers have improved such that they can fabricate increasingly complex and delicate products. As a result, many individuals now fabricate anything, possibly becoming to fabricate a house in the future [1]–[4]. This style of fabrication is called "personal fabrication [5]–[7]."

As individual participation in personal fabrication increases, it is necessary to provide sustainable access to fabrication history to clarify who is liable for the product. For example, if a 3D printed toy injures a child, the fabricator and the printed design data must be identified to ascertain the cause. The Japanese Product Liability Act states that consumers can pursue compensation for damages from the fabricator of the product by proving that the design has a defect [8]. Thus, it is necessary to record and maintain a history that identifies

the fabricator, the design data, and the other fabrication environment data to ascertain the cause of the failure.

However, even if 3D printers record print job records as a type of history, the integrity of this history cannot be guaranteed without ensuring that no one has rewritten the history. For example, even if a fabricator stored the history on a securely operated database, the database operator could still rewrite history. Product liability acts in many countries define the period that the fabricator should be liable (e.g., in Japan, for ten years). Thus, for recording and maintaining the history, it is a requirement that the history is immutable, and sustainably accessible while the fabricator is liable for products.

In this paper, we propose "Fabchain," a new scheme that manages a 3D print job over a blockchain, thus recording 3D print job records as history on the blockchain. Blockchain is a distributed ledger that records publicly immutable and highly available records in a peer-to-peer (P2P) network without a trusted third party. Fabchain constructs the history in an immutable, and sustainably accessible manner by owing a scheme that uses a blockchain as an audit-able communication channel [9]. We implemented a Fabchain prototype on the top of Ethereum, an application platform based on a blockchain. To evaluate the performance of our implementation, we measured the time required for completing a print job. As a security analysis, we discuss the security of Fabchain and compared it with a case using a public cloud service.

The remainder of this paper is organized as follows; In Sec. II, we describe the fabrication process and issues related to recording history. In Sec. III, we describe "Blockchain," which is the key technology in this paper. In Sec. IV, we describe our proposed scheme "Fabchain," and how the scheme satisfies the requirements of recording the history of print

jobs. In Sec. V, we describe the details of our implementation of Fabchain. In Sec. VI, we evaluate the performance by measuring the time required to complete a print job. In Sec. VII, we discuss the security of our proposed scheme and compare it with a case using a public cloud service. In Sec. VIII, we discuss the open issues of Fabchain. In Sec. IX, we describe the related works of Fabchain. In Sec. X, we conclude the paper.

II. PERSONAL FABRICATION AND FABRICATION HISTORY

First, we describe the model for personal fabrication processes that is our focus and the necessity of using history to clarify who is liable for product failures. Second, we discuss the requirements for recording and maintaining the history.

A. Fabrication Process of Personal Fabrication

Fig. 1 shows one of the use cases of personal fabrication. There are three actors:

- **Fabricator**, which fabricates products with a 3D printer.
- **E-store**, which lists and sells products fabricated by fabricators.
- **Consumer**, who buys products.

Initially, the fabricator lists their product in the e-store. At this time, the fabricator has not yet fabricated the product. When the fabricator receives an order, the fabricator fabricates the ordered products on-demand.
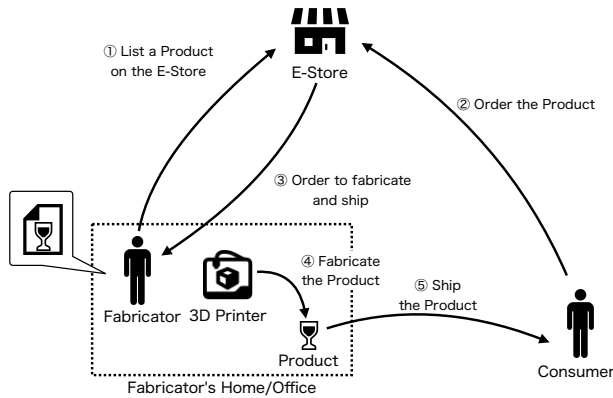

Fig. 1. Use case of Personal Fabrication; First, the fabricator lists their product in the e-store (①). When the consumer orders the fabricator's product (②), the e-store requests the fabricator to fabricate and ship the product (③). The fabricator then fabricates the product with the fabricator's 3D printer (④) and ships it to the consumer (⑤)

In this use case, the fabrication history is created in the fabrication process. In the process of fabrication, there are two actors:

- **Designer**, who creates a 3D model and publishes it on a 3D model publishing service.
- **Fabricator**, who owns a 3D printer and fabricates a product.

The designer and the fabricator can be individuals, sole proprietorships, or small companies. The designer and the fabricator own a PC to create and modify 3D models with Computer-Aided Design (CAD) software. The designer can upload and publish 3D models that they have created to 3D model publishing services, such as Thingiverse[1]. The fabricator owns three devices:

- **3D printer**, which prints a product from a 3D model.
- **Print server**, which manages print jobs and controls the 3D printer.
- **Print client**, which is a PC for modifying 3D models and makes printing requests to the print server.

Fig. 2 shows one of typical fabrication processes in person fabrication. In this process, the fabricator's devices (3D printer, print server, or print client) records the fabrication history in the devices' storage.
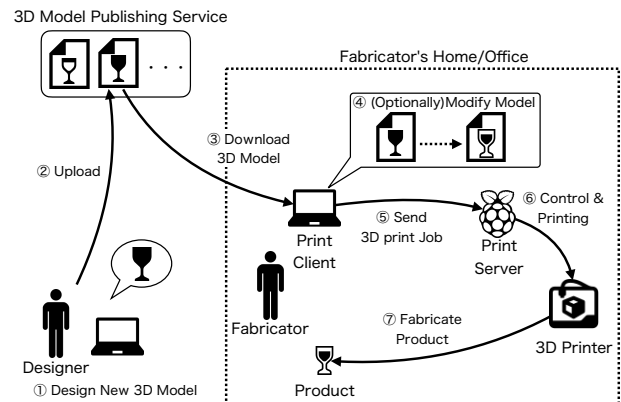

Fig. 2. A Personal Fabrication Process with Optional Modification of the Design; First, a designer creates a new 3D model (①) and publishes it on a 3D model publishing service (②). A fabricator then downloads the 3D model on the client (③) and optionally modifies it to suit their use (④). The fabricator then sends a print job, including the 3D model, from the client to their print server (⑤). The print server controls the 3D printer (⑥) and fabricates a product from the 3D model (⑦).

B. Requirements to Clarify Liability

Clarifying liability requires a fabrication history that identifies the fabricator and the printed 3D model. Hence, how to record and maintain that history is a significant issue. Previous studies have employed various approaches to managing the fabrication history of products, including attaching a product serial identifier, embedding an RFID tag into the 3D printed product, or embedding an identifier as internal structure [10], [11]. In this paper, we focus on a scheme for recording and maintaining fabrication history specified by the product serial identifier.

To utilize fabrication history as evidence of product liability, the fabrication history must be *immutable* and *sustainably accessible*. First, the history should be *immutable* to prevent

---

[1]https://www.thingiverse.com/

anyone from rewriting said history, which would falsify claims of liability. For example, a fabricator might delete the history of their product and then claim that they did not fabricate the product. Alternatively, to fraudulently blame the fabricator or the designer, the consumer might falsify the history so that the consumer to pretend that the 3D model or the fabricating process has defects.

Next, the history should be *sustainably accessible*, that is, everyone should be able to verify the product liability. When the consumer pursues product liability, the consumer should retrieve the history of the product and be able to verify whether the design has a defect. According to the Japanese Product Liability Act, we estimate that the fabricator should keep history accessible for ten years [8].

III. BLOCKCHAIN TECHNOLOGY

In this section, we describe the "Blockchain," which is the key technology in this paper.

A. Blockchain Overview

Blockchain technology is a distributed ledger that records publicly immutable records in a P2P network without a trusted third party. In a blockchain system, a node records data in a unit called a "Block." A block consists of the cryptographic hash value of the prior block and a series of units of data, called "Transactions." Each node stores all the blocks in the node's local storage.

When a user records new data into the blockchain, the user creates a transaction and broadcasts it to the blockchain network. Each node in the network verifies the transaction. Some nodes create a new block from verified transactions and broadcast the new block. In the block creation process, the nodes perform a cryptographic protocol called "Proof-of-Work (PoW)," wherein the nodes demonstrate to other nodes that they have performed a certain amount of computational work in a specified interval of time [12]. Each node verifies the new block, and if verification succeeds, nodes store the block in local storage as a part of the blockchain.

When there are contradicting blocks, i.e., some blocks include the same hash value of the prior block with different transactions, each node chooses a block that spent the most computational power during the block creation process. Because each block includes the previous block's hash value, if an attacker seeks to rewrite a historical block in a blockchain, the attacker will need computational power to create all blocks from the rewritten block to the latest block. Because it is generally difficult to conduct such attacks, we can treat blockchain as an immutable ledger.

Because a node with the entire blockchain can independently verify blocks and transactions, the blockchain achieves an immutable ledger without a trusted third party. Blockchains are also highly available because all nodes in the blockchain network replicate and maintain the blockchain. Those characteristics of the blockchain are considered useful in applications, in addition to cryptocurrency [13]–[15].

B. Ethereum

Ethereum is a blockchain-based application platform on which users can run programs called "smart contracts [16]." An Ethereum node has the blockchain and an Ethereum Virtual Machine (EVM), which includes an interpreter and the World State consisting of contract states for each smart contract. When the node records a transaction that includes a smart contract into the blockchain, the EVM creates a contract state and executes the constructor of the smart contract (we refer to this condition as "a smart contract is deployed."). The transaction includes the digital signature of the issuer of the transaction. The issuer is identified by the "Ethereum Address," which is the hash value of its public key. After creating the contract state, the smart contract is identified by the smart contract pointer called the "Contract Address."

To update the contract state, a user broadcasts a transaction that includes the contract address and input values for a function in the smart contract. When the node records a block with the transaction into the blockchain, the EVM executes the function and updates the contract state. With this scheme, because the node records each transition of the contract state in an immutable and highly available manner, the contract state is also immutable and highly available. When simply reading the contract state, no transaction is required.

With Ethereum, a transaction for deploying and executing a smart contract requires execution fees. The fee amount for the EVM opcodes is defined in the Ethereum Yellow paper and is hardcoded in the EVM implementation [16].

C. Blockchain as Audit-able Communication Channel

Suzuki and Murai proposed to use blockchain as an audit-able communication channel [9]. In their scheme, a client creates a transaction that includes a message to a server. The server retrieves the transaction from the blockchain and obtains the message. With the scheme, the history of the communication is recorded in an immutable, highly available, and verifiable manner. Past studies utilized this scheme in IoT (Internet of things) applications and human-robot interactions [17], [18].

IV. FABCHAIN: MANAGING AUDIT-ABLE 3D PRINT JOB OVER BLOCKCHAIN

In this paper, by utilizing the scheme described in Sec. III-C, we propose "Fabchain," a scheme that manages 3D print jobs over a blockchain.

A. Proposed Scheme

Fig. 3 shows the overview of Fabchain. A fabricator posts a request transaction that includes a 3D printing request (① and ②). After recording the request transaction into the blockchain, a print server that controls a 3D printer retrieves the transaction from the blockchain and obtains the request (③). After printing (④), the print server posts a response transaction that includes fabricating results (⑤). With the scheme, the status and data of the print job are managed over the blockchain. For example, if there is only the request on the blockchain, the print job has not been completed.
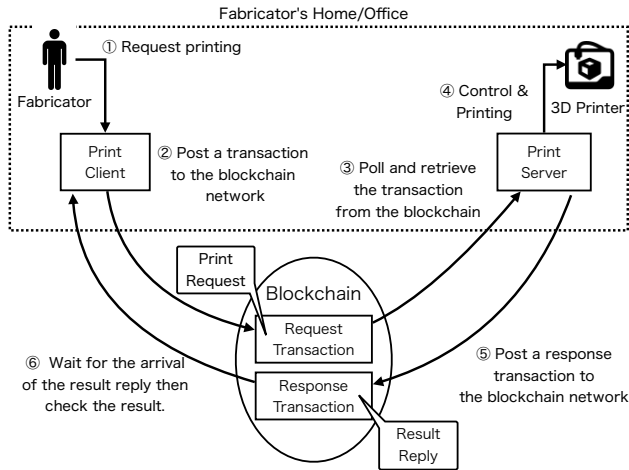
Fig. 3. Fabchain Overview

B. Satisfying Requirements

In this section, we describe how Fabchain satisfies each requirement mentioned in Sec. II-B. In this paper, we discuss Fabchain as implemented on top of Ethereum.

1) Immutable: The fabrication history within Fabchain is immutable according to the immutability of the blockchain. The immutability of blockchain is described in the Bitcoin white paper [19]. It can be adopted to Ethereum because Ethereum also has immutability based on the PoW. The white paper describes that the more blocks that are stacked on the block that is the target of rewriting, the more immutable is the block. In Ethereum, the interval for the block creation is approximately 12 seconds. When fabricating with a 3D printer, the fabrication process will typically takes a few hour. Therefore, while fabricating, the history would become immutable by stacking some blocks on the block that includes the request transaction.

2) Sustainably Accessible: With Fabchain, the history is public because the Ethereum main network is a public network. We can obtain the history on the blockchain by accessing the blockchain from any online Ethereum nodes. Therefore, history is accessible as long as one of the nodes remains in the Ethereum network.

V. IMPLEMENTATION

We implemented a proof-of-concept version of Fabchain on top of Ethereum. We defined the objective of the implementation as the ability to identify the printed 3D model. Thus, we implemented Fabchain so that it records the hash value of a printed 3D model. Methods to trace the original designer of the printed 3D model are out-of-scope for the current implementation.

This implementation consists of the followings components:

- **Print client**, which creates a request transaction that includes a print request.
- **Smart contract**, which manages the status of print jobs and provides interfaces for accessing history in the contract state.

- **Print server**, which retrieves print requests, creates a response transaction that includes the results of the print request, and controls a 3D printer.

We implemented the print client and the print server as command-line applications in Python, and the smart contract in Solidity [20]. Fig. 4 shows an overview of the processing of a print job in our implementation. The client and the server interact with the smart contract via possibly different Ethereum nodes. The Ethereum nodes participate in an Ethereum network that the smart contract has been deployed. The client and server have corresponding private keys that the fabricator generated in advance. The client and the server are identified by their Ethereum Addresses.
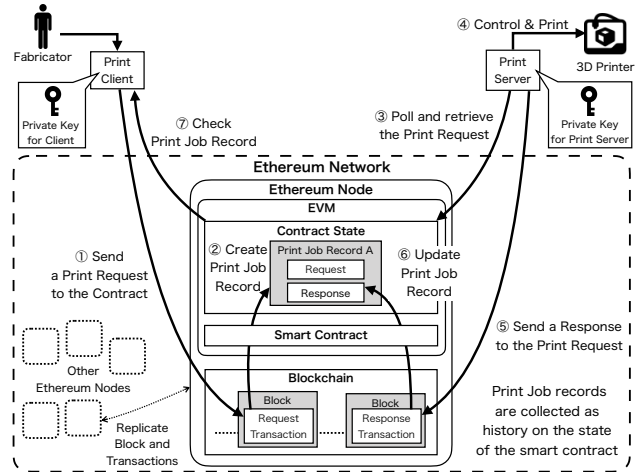

Fig. 4. Implementation Overview; This figure shows the process for the product fabricated by Print Job A. The smart contract collects historical print job records as a history and provides that history so that anyone can read.

The overview of the process in our implementation is as follows. When the client sends a request transaction (①), the smart contract creates a new print job record (②). The print job record is identified by the Print Job ID, which is the request's hash. Anyone then can retrieve the print job records using the Print Job ID. The server polls the contract state to retrieve new print job records (③). When the server finds a new print job record, the server verifies and adds the request to the print queue. The server waits for the print to finish (④), then sends a response transaction (⑤). The contract then records the response to the print job record (⑥). During the entire process, the client waits for the update of the print job record by polling. The client finally receives the result in the print job record and does post-print processes accordingly (⑦).

A. Assumptions

We state the following assumptions for our implementation. First, we assume that consumers can obtain a Print Job ID as a serial identifier that identifies the product by technology-agnostic schemes. For example, each product may contain a serial identifier embedded in an RFID tag.

Second, we assume that entities use Ethereum Addresses as identifiers of each entity, however, how to authenticate the

Parameter	Type	Description
Request	Request	Fabrication Request described in Table II
PrintDate	bytes32	Timestamp when the response transaction is generated
Approved	bool	Request is approved or not approved by the server
Printed	bool	The print job is printed or not printed by the server

TABLE II
FIELDS IN REQUEST RECORD

Parameter	Type	Description
From	Address	Ethereum Address of fabricator
Printer	Address	Ethereum Address of 3D printer
ModelHash	Bytes32	Hash value of requested 3D model data
Date	Bytes32	Timestamp when the request is generated

owner of the Ethereum Address is out of the scope of the paper.

Third, we assume to use an external data repository that stores data that is difficult to store on the Ethereum blockchain. In Ethereum, it is difficult to manage large data, such as 3D model data, due to the size limitations of the data type of Solidity [20]. It is a typical scheme that stores large data as a value on a Key-Value Store (KVS) and stores the key to a blockchain [21], [22]. By using a hash value of the data as the key, the user can verify whether the retrieved data is the data that is identified by that key. In this paper, we assume that the data is highly available by storing on a highly available KVS such as IPFS [2]. To simplify our implementation, we implemented a simple KVS to store and retrieve data using the hash value and utilized this implementation [3].

B. Data Structures

In this section, we describe the data structures used in the contract state: Print Job record, and Request record. The fields in the Print Job record are shown in Table I. Print Job record includes the Request record, the printing date, and flags that show the condition of completion of approval and print.

The fields in the Request record are shown in Table II. The Request record includes the hash values of the 3D model to retrieve those data from the data repository.

C. Print Job Sequence

In this section, we describe the print job sequence. Table I and Table II show the data structures in the contract state: the Print Job record and the Request record. Fig. 5 shows the sequence of the process. The process is separated into four phases:

2https://ipfs.io/
3https://github.com/chike0905/simple-hashstorage

---

1) **Request Phase:** the client creates a new Print Job record on the contract state.
2) **Approve Phase:** the server approves the Request in the Print Job record.
3) **Print Phase:** the server prints the print request using the connected 3D printer.
4) **Response Phase:** the server responds to the Request in the Print Job record.

The client performs only the Request phase and, after the request phase, keeps polling the print job status in the contract state. The server keeps polling the contract state to find a new Print Job to the server.

VI. PERFORMANCE ANALYSIS

We evaluated the performance of Fabchain by measuring the time required for each phase described in Sec. V-C. We experimented on an Ethereum test network. We then estimated the performance on the main network with parameters in a previous study, which measured the performance of a smart contract [23].

A. Estimation of Performance

Before the experiment, we clarified parameters that affect the performance and estimate the performance. In this paper, we defined the duration for a transaction calling a contract function as D_{tx}. In Ethereum, D_{tx} depends on the duration from when the transaction is submitted to when the transaction is recorded in the blockchain. Hence, the D_{tx} is defined by the following parameters:

- D_{block}: The duration between each block creation event
- $N_{\mathrm{UntilIncluded}}$: The number of blocks between the latest block when a transaction is submitted and the block that includes the transaction.

Because the elapsed time since the last block is determined when a transaction is submitted, we can estimate the value of D_{tx} would be as follows.

$$D_{\mathrm{tx}} \leq D_{\mathrm{block}} \cdot N_{\mathrm{UntilIncluded}} \tag{1}$$

We estimated the duration for a phase that includes one transaction process, as defined by (1).

B. Experimental Setup

For the experiment, we prepared four virtual machines as a client, a server, a data repository, and an Ethereum node. For the experimental setup, we connected the client and the server connects to the same Ethereum node because the client and the server are in the fabricator's home according to the use case mentioned in Sec. II-A. Table III shows the configuration of the four virtual machines. We used an Ethereum node implementation known as "Geth [24]." We experimented with our implementation on a node that participates in an Ethereum test network called "Ropsten." The configuration for Ropsten is the same as the main network: the network is public, and D_{block} is adjusted as an average of 12 seconds. The difference between the Ropsten network and the main network is that the cryptocurrency on Ropsten has no value, and participants of
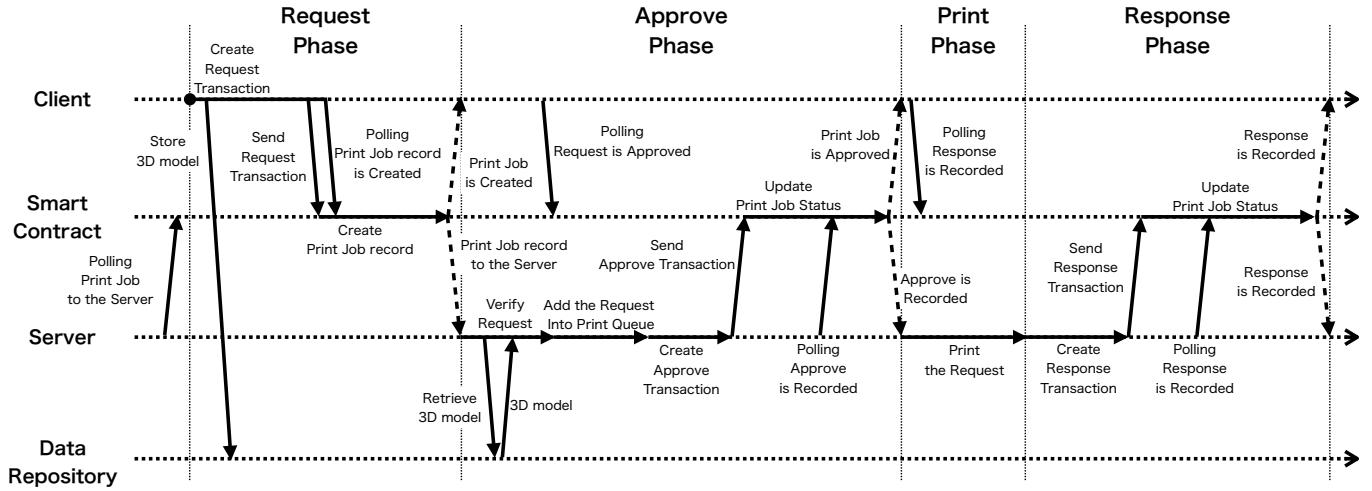
Fig. 5. Sequence of the Print Job Process

Item	Configuration
Operating System	Debian 10
VM Config	4GB memory, 4 vCPUs
Hypervisor	ESXi 6.5
Hypervisor Hardware	Fujitsu rx200 s6 (48GB Memory, 24CPUs)

Ropsten are fewer than in the main network. Hence, we can experiment on Ropsten as an emulation of the main network. The client and the server communicate with the Geth node via the RPC API. We processed the experiment remotely via the console and obtained processing logs on each of the virtual machines. We measured the print job sequences 100 times.

C. Results

First, we analyzed the log of Geth to verify D_{tx}. Fig. 6 shows the experimental results. In the experiment on Ropsten, it is difficult to measure when a node in Ropsten creates a block. Thus, we instead take the time when our node receives a block as the time when the block is created. In the experiment on Ropsten, D_{block} was 9.17 seconds, D_{tx} is was 15.90 seconds, and $N_{\mathrm{UntilIncluded}}$ was 1 blocks on average. The result was slightly different from our estimation because there was overheads for propagating transactions and blocks from nodes.

Next, we measured the duration of each phase. We measured the request phase from logs on the client and other phases from logs on the server. We also measured the duration from the client starts the request phase to the client can detect the request is approved ("request-approve"), and from the client starts to the client detects the response is recorded ("All Phase"). In the measurement, we eliminated the print phase because the duration for printing is affected by the 3D printer and 3D model.

Fig. 7 shows the results. The resulting duration of request-approve was approximately 38.93 seconds, and the result
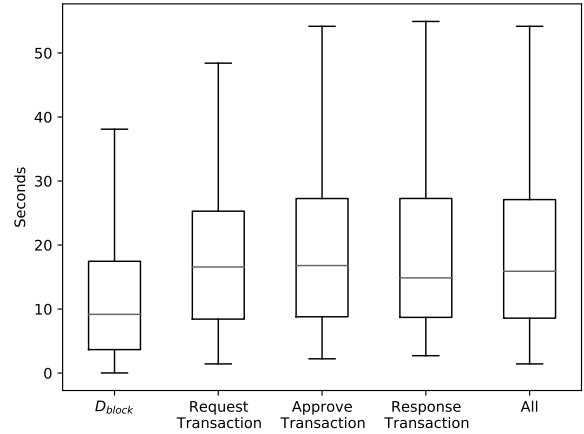

Fig. 6. Duration from submitting transaction

of the request phase was approximately 14.66 seconds. The distribution of those results were in accord with the distribution of D_{tx}. Thus, we concluded that our estimation is reasonable on Ropsten.

D. Discussion: Estimation on Ethereum Main Network

In the Ethereum main network, the D_{block} is 12 seconds[4]. Spain et al. measured the latency of transactions on the Ethereum main network [23]. They concluded that over half of all transactions issued at depth i in the blockchain were included by the block at depth $i + 2$. Therefore, we can estimate the $N_{UntilIncluded}$ is approximately two blocks from their result. With those parameters, the duration for request-approved would be 48 seconds. The duration for the response phase would be 24 seconds. Even if we consider the overhead for the propagation of transactions and blocks, we concluded

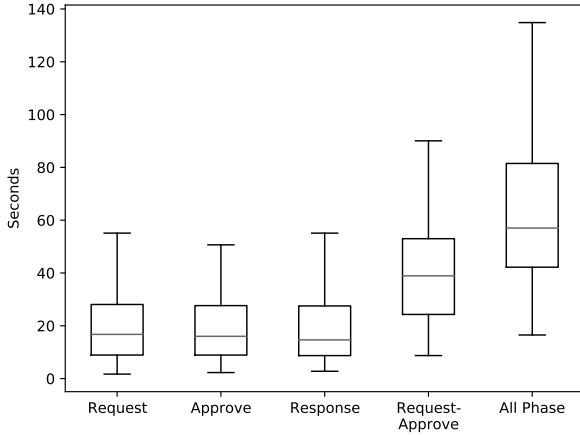[4]https://etherscan.io/chart/blocktime

Fig. 7. Duration of Each Phase

that this duration does not present a problem because 3D printing takes significantly more time, even when printing small objects.

VII. SECURITY ANALYSIS

In this section, we compare Fabchain and a case in which a fabricator constructs a history publishing service, from a security perspective. Namely, we address how to satisfy the requirements mentioned in Sec. II-B.

For comparing each scheme, we defined three setups to operate Fabchain or a history publishing service.

- **Setup 1** A fabricator operates the client and the server of Fabchain and an Ethereum node as a gateway to the Ethereum network.
- **Setup 2** A fabricator operates the client and the server of Fabchain and utilizes an Ethereum node that other entities, such as Infura operate [5].
- **Setup 3** A fabricator operates a history publishing service on a cloud computing service, such as Google Cloud Platform [25].

1) Security of Fabchain: In Fabchain, the immutability of the history is provided by the underlying blockchain. If one of the operators of the Ethereum nodes tries to fake history on a blockchain in their node, other nodes do not accept the faked history. The fabricator and consumers do not need to trust any operator of a specific Ethereum node.

From the perspective of sustainable accessibility, history by Fabchain is highly available. In December 2021, several entities are operating approximately 2000 Ethereum nodes all over the world [6]. It is not realistic that all the nodes suddenly leave the network. Hence, history would be sustainably accessible semi-permanently.

[5]https://infura.io/
[6]https://etherscan.io/nodetracker

2) Security of history publishing service: In the case of a history publishing service, it is difficult to prove that the history is immutable. For example, when the fabricator is malicious and makes it so that the history publishing service provides a faked history, the consumer cannot verify whether the provided history is faked or not because there is no proof.

In another aspect, the availability of the history publishing service depends on the operator of the instances on the public computing service. To keep history sustainably accessible, a specific operator should keep operating the history publishing service. When the fabricator operates the history publishing service for its products, the history is lost if the fabricator stops working as a fabricator and operating the instance.

3) Security Discussion on Each Setup: In Setup 1, the fabricator's Ethereum node broadcasts transactions of Fabchain to other Ethereum network participants. The fabricator's Ethereum node can independently verify whether the transaction is recorded in the blockchain. Hence, Setup 1 can satisfy the requirements securely.

In Setup 2, there is a risk that the gateway node is malicious. For example, the gateway node can deceive whether the transaction is recorded. However, there are several lightweight node schemes, such as Simple Payment Verification, that can verify whether a transaction is included in the blockchain [26]. Combining those lightweight node schemes, Setup 2 can also satisfy the requirements while mitigating the risk.

In Setup 3, it is difficult to satisfy the requirements because there is no proof of immutability. Considering the operation of the history publishing service, we conclude there is an advantage for Fabchain.

VIII. OPEN ISSUES

In this section, we describe three open issues in Fabchain. The first issue involves managing the identity of each entity. The second issue involves the volatility of the cost for Fabchain. The third pertains to the applicability of Fabchain.

A. Identity Management

In our implementation, we utilized an Ethereum Address as an identifier of a fabricator. To pursue a product's liability, it is necessary to identify the fabricator from the print job record. Therefore, it is a requirements of future research for Fabchain to develop methods to identify the owner of Ethereum Address. One method would be to embed the public key certificate into the Print Job record. By verifying the public key certificate and the Ethereum Address, the method can verify that the Print Job record is issued with the private key owned by the entity described in the public key certificate.

B. Volatility of Cost

Because the price of ether is highly volatile, the cost for a print job with Fabchain may be high. While working on this research from 2020 March to 2021 December, the cost for Fabchain became 20,000 times higher than when we first estimated the cost. The issue is caused by the design of Ethereum, wherein a user pays a fee for executing a

smart contract. Blockchain-based applications that do not relate to cryptocurrency share this issue [27]. Re-designing a blockchain as an application platform without cryptocurrency is one possible solution for this problem.

C. Applicability of Fabchain

In this section, we discuss the applicability of Fabchain. First, fabrication history is useful for evaluating fabrication processes. Various factors, such as temperature and 3D model structures, can cause digital fabrication devices to fail when creating products. Thus, the detailed results are important to analyze the reasons for the failure. Hence, we need to determine what data should be recorded in the history.

IX. RELATED WORKS

Several studies have applied blockchain technology to tasks in industries [28], [29]. With the widening adoption of "Industry 4.0," which involves collaborating with several parties in the industrial process via the Internet, previous studies have applied blockchain to guarantee trust within parties [30]–[33]. Makerchain, one of the previous studies, is a blockchain-based system that records multi-party activity in Industry 4.0 [34]. In other fields, several studies have applied blockchain to maintain traceability [35], [36]. Those studies focused on the integrity of the information from another party. Fabchain focuses on a single party's internal process to record each manufacturing process in a single party. With the combination of Fabchain and Makerchain, we might verify that a party has manufactured a subset of a product and the party's manufacturing process.

3D printing technology is one of the most significant technologies associated with Industry 4.0. Within the field of 3D printing, several studies have applied blockchain technology to guarantee the copyrights of 3D models [37]–[39]. In Industry 4.0, Fabchain could be the underlying scheme for those use cases.

Other studies have utilized blockchain technology in supply chain management and the detection of counterfeit products [40]–[42]. Combined with the ideas proposed in the studies described above, Fabchain would ensure transparency in the fabrication and distribution processes.

X. CONCLUSION

In this paper, we proposed and implemented Fabchain, which manages a 3D print job over a blockchain. To clarify who is liable for the product, a fabricator should keep fabrication history in an immutable, and sustainably accessible manner without a trusted third party. Fabchain can produce an immutable and publicly accessible fabrication history, due to its scheme that utilizes blockchain as an audit-able communication channel. We implemented a prototype of Fabchain and performed experiments on the Ropsten network. We concluded that our prototype can manage communication of a print job sequence within less than one minute so that Fabchain was reasonable for 3D printing. In our evaluation scenario, Fabchain can satisfy the requirements of immutability and

accessibility. Finally, we described some open issues about the identification of entities and volatility of cost, and applicability of Fabchain. In future work, we plan to implement applications which enable everyone to fabricate products with liability.

REFERENCES

[1] P. Blikstein, "Digital fabrication and 'making'in education: The democratization of invention," *FabLabs: Of machines, makers and inventors*, vol. 4, no. 1, pp. 1–21, 2013.

[2] N. Gershenfeld, "How to make almost anything: The digital fabrication revolution," *Foreign Aff.*, vol. 91, p. 43, 2012.

[3] B. Kolarevic, "Digital fabrication: Manufacturing architecture in the information age," in *Proc. the Twenty First Annual Conference of the Association for Computer-Aided Design in Architecture*, 2001, pp. 268–278.

[4] J. G. Sanjayan and B. Nematollahi, "Chapter 1 - 3d concrete printing for construction applications," in *3D Concrete Printing Technology*. Butterworth-Heinemann, 2019, pp. 1–11.

[5] N. Gershenfeld, *Fab: the coming revolution on your desktop–from personal computers to personal fabrication*. Basic Books, 2008.

[6] C. Mota, "The rise of personal fabrication," in *Proc. the 8th ACM Conference on Creativity and Cognition*, 2011, pp. 279–288.

[7] D. Chen *et al.*, "Direct digital manufacturing: definition, evolution, and sustainability implications," *Journal of Cleaner Production*, vol. 107, pp. 615–625, 2015.

[8] M. of Justice Japan, "Product liability act," [Accessed December 14, 2021]. [Online]. Available: http://www.japaneselawtranslation.go.jp/law/detail/?id=86

[9] S. Suzuki and J. Murai, "Blockchain as an audit-able communication channel," in *Proc. 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, 2017, pp. 516–522.

[10] K. Fujiyoshi *et al.*, "Rfid 3d printing objects that connote information," in *Proc. 30th International Conference on Digital Printing Technologies and Digital Fabrication*, vol. 2014, no. 1, 2014, pp. 316–319.

[11] K. D. D. Willis and A. D. Wilson, "Infrastructs: Fabricating information inside physical objects for imaging in the terahertz region," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 138:1–138:10, Jul. 2013.

[12] M. Jakobsson and A. Juels, "Proofs of work and bread pudding protocols(extended abstract)," in *In Proc. Secure Information Networks: Communications and Multimedia Security IFIP TC6/TC11 Joint Working Conference on Communications and Multimedia Security (CMS'99)*, 1999, pp. 258–272.

[13] A. Azaria *et al.*, "Medrec: Using blockchain for medical data access and permission management," in *Proc. 2016 2nd International Conference on Open and Big Data (OBD)*, 2016, pp. 25–30.

[14] M. A. Khan and K. Salah, "Iot security: Review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018.

[15] "Namecoin," [Accessed December 14, 2021]. [Online]. Available: https://www.namecoin.org/

[16] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," 2014, [Accessed December 14, 2021]. [Online]. Available: https://gavwood.com/paper.pdf

[17] H. Haddadi *et al.*, "Siotome: An edge-isp collaborative architecture for iot security," in *Proc. International Workshop on Security and Privacy for the Internet-of-Things (IoTSec)*, 2018, pp. 42–45.

[18] E. C. Ferrer *et al.*, "Robochain: A secure data-sharing framework for human-robot interaction," 2018. [Online]. Available: https://arxiv.org/abs/1802.04480

[19] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, [Accessed December 14, 2021]. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[20] Ethereum, "Solidity documentation v0.6.3," [Accessed December 14, 2021]. [Online]. Available: https://solidity.readthedocs.io/en/v0.6.3/

[21] E. Nyaletey *et al.*, "Blockipfs - blockchain-enabled interplanetary file system for forensic and trusted data traceability," in *Proc. 2019 IEEE International Conference on Blockchain (Blockchain)*, 2019, pp. 18–25.

[22] R. Kumar *et al.*, "Distributed off-chain storage of patient diagnostic reports in healthcare system using ipfs and blockchain," in *Proc. 2020 International Conference on COMmunication Systems NETworkS (COMSNETS)*, 2020, pp. 1–5.

[23] M. Spain, S. Foley, and V. Gramoli, "The Impact of Ethereum Through-put and Fees on Transaction Latency During ICOs," in *International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2019)*, vol. 71, 2020, pp. 9:1–9:15.

[24] Ethereum, "Geth - official go implementation of the ethereum protocol," [Accessed December 14, 2021]. [Online]. Available: https://github.com/ethereum/go-ethereum

[25] Google, "Google cloud platform," [Accessed December 14, 2021]. [Online]. Available: https://cloud.google.com/

[26] Bitcoin.org, "Simplified payment verification (spv)," 2018, [Accessed December 14, 2021]. [Online]. Available: https://bitcoin.org/en/glossary/simplified-payment-verification

[27] K. Shudo, R. Kanda, and K. Saito, "Towards application portability on blockchains," in *Proc. 2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, 2018, pp. 51–55.

[28] J. Al-Jaroodi and N. Mohamed, "Industrial applications of blockchain," in *Proc. 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, 2019, pp. 0550–0555.

[29] V. Dedeoglu *et al.*, "A journey in applying blockchain for cyberphysical systems," 2019. [Online]. Available: https://arxiv.org/abs/1912.01606

[30] N. Mohamed and J. Al-Jaroodi, "Applying blockchain in industry 4.0 applications," in *Proc. 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, 2019, pp. 0852–0858.

[31] J. Liu *et al.*, "A framework of credit assurance mechanism for manufacturing services under social manufacturing context," in *Proc. 2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, 2017, pp. 36–40.

[32] A. Angrish *et al.*, "A case study for blockchain in manufacturing:"fabrec": a prototype for peer-to-peer network of manufacturing nodes," *Procedia Manufacturing*, vol. 26, pp. 1180–1192, 2018.

[33] H. Schaffers, "The relevance of blockchain for collaborative networked organizations," in *Working Conference on Virtual Enterprises*. Springer, 2018, pp. 3–17.

[34] J. Leng *et al.*, "Makerchain: A blockchain with chemical signature for self-organizing process in social manufacturing," *Journal of Cleaner Production*, vol. 234, pp. 767–778, 2019.

[35] Z. Wang, T. Wang, H. Hu, J. Gong, X. Ren, and Q. Xiao, "Blockchain-based framework for improving supply chain traceability and information sharing in precast construction," *Automation in Construction*, vol. 111, p. 103063, 2020.

[36] K. Salah, N. Nizamuddin, R. Jayaraman, and M. Omar, "Blockchain-based soybean traceability in agricultural supply chain," *IEEE Access*, vol. 7, pp. 73 295–73 305, 2019.

[37] M. Holland *et al.*, "Intellectual property protection of 3d print supply chain with blockchain technology," in *Proc. 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, 2018, pp. 1–8.

[38] S. Belikovetsky, O. Leiba, A. Shabtai, and Y. Elovici, "3d marketplace: Distributed attestation of 3d designs on blockchain," 2019. [Online]. Available: https://arxiv.org/abs/1908.06921

[39] M. McConaghy *et al.*, "Visibility and digital art: Blockchain as an ownership layer on the internet," *Strategic Change*, vol. 26, no. 5, pp. 461–470, 2017.

[40] K. Toyoda *et al.*, "A novel blockchain-based product ownership management system (poms) for anti-counterfeits in the post supply chain," *IEEE Access*, vol. 5, pp. 17 465–17 477, 2017.

[41] Z. C. Kennedy *et al.*, "Enhanced anti-counterfeiting measures for additive manufacturing: coupling lanthanide nanomaterial chemical signatures with blockchain technology," *Journal of Materials Chemistry C*, vol. 5, no. 37, pp. 9570–9578, 2017.

[42] E. A. Maroun *et al.*, "Adoption of blockchain technology in supply chain transparency: Australian manufacturer case study," in *Proc. the 10th European Decision Sciences Institute (EDSI)*, 2019.