A Blockchain-based Secure Storage and Access Control Scheme for Supply Chain Finance

Dun Li[1,2*†], Dezhi Han[1*†], NoelCrespi[2], RobertoMinerva[2] and Kuan-Ching Li[3†]

[1*]College of Information Engineering, Shanghai Maritime University, Shanghai, 201306, China.
[2]Samovar, Telecom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France, Palaiseau, 91764, France.
[3] Dept. of Computer Science and Information Engineering, Providence University, Taichung City, 43301, Taiwan.

*Corresponding author(s). E-mail(s): lidunshmtu@outlook.com; dzhan@shmtu.edu.cn;
Contributing authors: noel.crespi@mines-telecom.fr; roberto.minerva@telecom-sudparis.eu; kuancli@pu.edu.tw;
†These authors contributed equally to this work.

Abstract

Supply Chain Finance (SCF) provides credit for Small and Medium-sized Enterprises (SMEs) with low credit lines and small financing scales. The resulting financial credit data and related business transaction data are highly confidential and private. However, traditional SCF management schemes use third-party platforms and centralized designs that cannot achieve highly reliable secure storage and fine-grained access control. To address such a need, we propose Fabric-SCF, designing and implementing a Blockchain-based secure storage system by utilizing distributed consensus to realize data security, traceability, and immutability. The Attribute-Based Access Control (ABAC) model is deployed for access control, also utilizing smart contracts to define system processes and access policies to ensure the system's efficient operation. To verify the performance of Fabric-SCF, two sets of simulation experiments are designed its effectiveness. Experimental results show that

Fabric-SCF achieves dynamic and fine-grained access control while maintaining high throughput in a simulated real-world operating scenario.

Keywords: Blockchain, Smart Contract, Supply Chain Finance, ABAC, Hyperledger Fabric

1 Introduction

Supply Chain Finance (**SCF**) enables enterprises of different sizes to have better opportunities for joint development. Under the situation of economic globalization, Small and Medium-sized Enterprises (**SMEs**) have gradually become an essential driving force for national economic development [1]. SMEs need sufficient cash flow for their operations, but capital inflows and expenditures for supply chain operations occur at different times, so access to financing services is often required to cover temporary shortfalls in capital flow. However, traditional financial services treat SMEs as isolated entities [2], ignoring their business potential in the supply chain, making them face long-term financing difficulties[3]. As a matter of the fact, SCF plays a significant role in serving the entity economy and SMEs, by combining capital operations with the supply chain and establishing a sound transaction structure [4]. Furthermore, SCF has entered the information and intelligent systems stage [5]. Through the construction of enterprise information systems, the difficulties of information asymmetry among supply chain stakeholders have been alleviated, the cost of information circulation has been reduced, and convenient and efficient financial services can be provided [6].

Nevertheless, SCF involves significant amounts of digital data. The flow of production, information, and capital is often seen as isolated across functions and parties. In traditional SCF management solutions, the initiation of business processes depends on sequential input and manual validation. The high cost and complexity of these solutions, their security flaws, and time-consuming processing have caused the development of SCF to encounter bottlenecks [7]. In addition, in the financing business of SCF, the credit transmission capacity of core enterprises is limited. SMEs also face problems such as irregular financial statements and inadequate management systems. Therefore, it is difficult for financial institutions to control risks through the information provided effectively. Besides, the de-globalisation of the economy and the intense political conflicts (such as the recent Russia-Ukraine war) due to different political ideologies have made it even more difficult to adapt the traditional SCF model to the technological requirements of the new era [8]. Thus, it is urgent to add effective privacy protection mechanisms and access control mechanisms to the SCF management plan to help financial institutions obtain information efficiently and securely [9].

Interestingly, Blockchain technology does not only solve the current issues of information asymmetry, lack of visibility in the transaction process, but also possibly joint fraud in the core business model. As the underlying technological framework for cryptocurrencies, Blockchain is decentralized and immutable [10–12]. Blockchain is a transparent, secure, decentralized system based on an immutable model, seen as a solution to change the rules in the traditional supply chain industry. Currently, Blockchain projects can be divided into four main categories: cryptocurrency, platform, application, and asset tokenization. Mainstream Blockchain platforms have been widely used in healthcare [13–18], smart cities [19–21], energy networks [22–25], Internet of Things (IoT) [26–30], Internet of Vehicles (IoV) [31–37], and education[38–41]. Due to such, a Blockchain-based SCF solution is urgently needed and feasible.

With the emergence of the smart contract-enabled Blockchain projects such as Ethereum and Hyperledger, the Blockchain has entered the 2.0 era, where Smart contracts are programmable and Turing-complete [42], added that each transaction can be initiated automatically according to the rules set by the code. The introduction of smart contracts is expected to change the existing SCF's transaction mode and reconstruct the breakthrough technology of such a society from the underlying infrastructure. Moreover, in a realistic access control strategy, it is necessary to approve or deny users' requests based on the environmental conditions implemented by the current policy. In the open network environment of Blockchain, the Attribute-Based Access Control (**ABAC**) model is suitable and efficient as a flexible and fine-grained access control method [43]. This model determines whether the data requester has the correct attributes to assess the data requester's access control rights to private data resources.

To date, scholars have given extensive attention to risk pricing, profit enhancement, and business models for SCF. However, relatively few attempts have focused on applying Blockchain to SCF management systems for optimization and improvement [44, 45]. In this context, we propose a blockchain-based SCF management system that digitizes the business processes in SCF, optimizes them by deploying smart contracts, and stores them in the Blockchain in the form of data streams, thereby ensuring privacy, immutability, and traceability. We also introduce the ABAC model for access control to provide efficient and secure access to this system. Notably, the programmable smart contract design ensures the novelty of this solution. In summary, the motivation of this paper is to use the blockchain-based model design to ensure that the fine-grained access control and strong privacy protection in the SCF model are guaranteed under the condition of large-scale access.

Specifically, the main contributions of this article can be summarized as follows.

- Blockchain is applied to SCF management systems, enabling decentralized management, secure storage, and multi-party maintenance of financing projects with the support of distributed consensus and authentication mechanisms.
- Based on the basic framework of Blockchain, we design an auxiliary architecture based on ABAC to achieve fine-grained access control.

- Smart contracts are applied in the design to define multi-tier data structures, access policies, and system workflow to ensure high-speed data storage, retrieval, query, and fine-grained access control.
- Preliminary design and implementation of Fabric-SCF is complete, shown, and proven through simulation experiments to be efficient and promising performance.

The remaining of this article is organized as follows. The related works are presented in Section 2, followed by the relevant preliminaries in Section 3. Next, we present the system design, assumptions, and implementation details in Section 4, then performance and validation analysis of the proposed design based on experimental results in Section 5. Finally, the concluding remarks and future work are depicted in Section 6.

2 Related Work

In this section, we first survey the related works on the SCF model in Subsection 2.1, the application of Blockchain in SCF scenarios in Subsection 2.2, then followed by smart contract in SCF scenarios in Subsection 2.3.

2.1 SCF Model

Supply Chain Finance (SCF) has long been recognized as a significant intersection between the fields of trade finance and supply chain management. Empirical results show that, a well-structured SCF solution can significantly enhance the efficiency of capital flows for SMEs [46–55].

A number of recent researches have proposed building new SCF models that improve the efficiency of data flow and privacy protection mechanisms. To hedge financial risk, traditional SCF relies on credit ratings [56]. In [57, 58], a multi-agent simulation technology instead of absolute rationality is used to design an SCF simulation model based on Simon's bounded rationality. Lu *et al.* [59] evaluated the bank's fair pricing rate under the guarantee model and found that the financial model of third-party guarantees may not always be applicable. Xiao *et al.* [60] and Liao *et al.* [61] tried to apply entropy weight, the analytic hierarchy process, a multi-expert decision model, and other methods to sort suppliers to improve model efficiency. In [62], Big Data analytics is applied to build business processing platforms for SCF and structure business data, while Jia *et al.* [63] summarized that Sustainable Supply Chain Finance (SSCF) can significantly enhance the sustainability and development performance of SMEs. And, the works [64–68] confirms this view. Emtehani *et al.* [69] proposed an effective SCF decision framework from an optimization business perspective that assists in operational financing decisions and improving financial efficiency from an algorithmic perspective. Later, Sang [70] proposed a risk assessment strategy based on genetic algorithm and neural network methods to reduce the probability of profit damages to the fund providers in SCF. Yin *et al.* [71] proposed an edge-intelligent SCF model based on B2B platforms and has obvious efficiency advantages in responding to the changing trends of the parameters related to the SCF model of a B2B platform. Abbasi *et al.* [72] designed an IoT-based SCF model and demonstrated that the credit risk measurement model is credible.

The above models incorporate many new techniques in operations research, machine learning, neural networks, IoT, and edge computing toward improvements on the efficiency of SCF business processes and enhancement of data security. However, they all rely on third-party platforms and thus cannot offer adequate security.

2.2 Applications of Blockchain in SCF

The applications of Blockchain technology first originated from cryptocurrencies owing the underlying database and maintenance technology. Blockchain technology has broad industrial application prospects thanks to its decentralization, consensus mechanisms, and unalterability.

In [73, 74], researchers reviewed the feasibility of applying Blockchain technology in SCF and confirmed that the technical characteristics of Blockchain could effectively promote the development of SCF. More specifically, Du *et al.* [45] proposed a Blockchain-based business management platform for SCF that solves the distrust problem among supply chain participants. Liu *et al.* [75] proposed a hybrid chain model combining a public chain-based consistency algorithm and a federated chain-based consistency algorithm to process each account's transactions in parallel, serving the SCF data management needs of engineering projects and enabling more efficient transaction authenticity auditing, risk assessment, and credit delivery for core enterprises. Li *et al.* [76] proposed a solution for SCF and illustrated the operation process of three SCF models on a Blockchain platform. Shi *et al.* [77] discussed the applications of blockchain in SCF by large multinational companies such as Amazon, Alibaba and Microsoft Xbox. Wag *et al.* [78] proposed a model that combines blockchain technology and SCF gaming, using Blockchain to connect banks, other financial institutions, and core companies. Chen *et al.* [79] proposed BCAutoSCF, a Blockchain-powered SCF platform dedicated to serving the automotive retail industry, enabling reliable, transparent, and traceable business inquiries and data access. Similarly, in [80–90], Blockchain technology was applied to enhance the privacy of business management processes for SCF.

2.3 Applications of Smart Contracts in SCF

Notably, smart contracts have Turing-complete programmability and can be used to deploy the specific implementation of SCF-related business. Kim *et al.* [91] proposed an Ethereum-based Blockchain platform and transformed business processes into smart contracts to ensure data traceability. Also based on Etherium, Huertas *et al.* [92] proposed a smart contract-based ecosystem allowing SMEs to implement SCF solutions called Eximchain quickly. Such a system uses a consensus protocol and a secondary voting-based governance model to provide practical, time-limited security guarantees. References [93–95] also proposed similar SCF solutions based on smart contracts deployed in Etherium. Ma *et al.* [96] proposed a Hyperledger Fabric-based solution that adapts to a wide variety of scenarios and complex business processes using segmented permissions, privacy protection mechanisms, among others. Additional financial models based on smart contracts are discussed in [97].

In summary, these models attempt to build decentralized Blockchain-based models from a privacy-preserving perspective, using smart contracts to implement business processes in SCF [98]. Nevertheless, most of these models have been tested on public chains, and there is a lack of industrial level platforms.

3 Preliminaries

This section provides details of techniques and technologies related to the proposed research work. Subsection 3.1 describes Hyperledger Fabric, and followed by the main business processes for SCF in Section 3.2. Table. 1 lists the notations used in this article.

Table 1 Key Notations

Notation	Description
CE	Core enterprise
FI_i	Financial institution i
FE_i	Financing enterprise i
Sp_i, Dt_j	Supplier i, Distributor j
CF_i	Credit facility i
Col_i	Collateral i
$P_i{-}_j$	Productons $i - j$
ARD_i	Accounts receivable document i
PC_i	Prepayment Contract i
FO_i	Financing order i
FP_i	Financing Project i
CA	Certificate Authority
Ct_i	Certificate i
Cf_i	Config file of i
$pubk_i$	The public key of i
$prik_i$	The private key of i
σ_i	The anonymous credential of entity i
$RC_{(x)}$	The storage credential of content x
$CT_{(x)}$	Encrypted content for x
$HV_{(x)}$	The hash value of x
$ST_{(x)}$	The encrypted content of x

3.1 Hyperledger Fabric

Hyperledger Fabric is one of the most significant projects in the Blockchain industry, consisting of a set of open-source tools and multiple sub-projects. In detail, it only permits authorized members to participate in data maintenance, and members recognize each other. The principal components of Hyperledger Fabric are the nodes, transactions, chain code, channels, consensus, and ledgers.

3.1.1 Nodes

Hyperledger Fabric contains four types of nodes: *CA*, *Peer*, *Orderer*, and *Client* as shown in Fig. 1.

CA can generate or cancel member identity certificates it receives in the registration application from the Client and returns the registration password for the user to log in to obtain the identity certificate.

Peer is the physical carrier of the ledger and the chaincode. It can be divided into *Endorsement*, *Leader*, and *Comitter* components, which are responsible for data verification, transaction distribution and storing respectively.

Orderer receives the transactions sent by *Peer* and sorts the transactions based on the consensus algorithm.

Client is used to interact with *Peers* to realize the operation of adding, deleting, modifying, and checking blockchain networks.

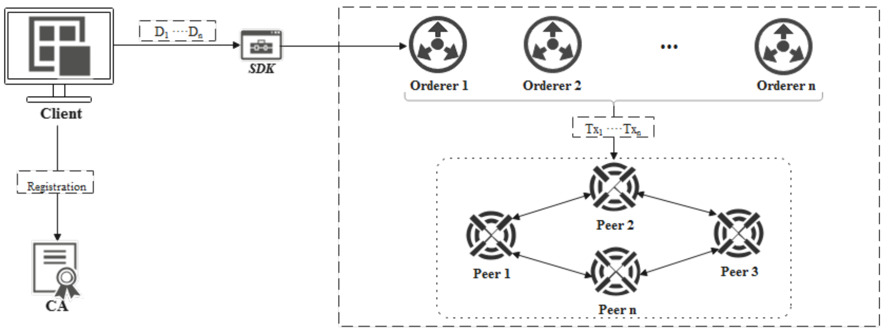


Fig. 1 Nodes Network.

3.1.2 Transactions

The application can only write data D_i by initiating a transaction Tx_i. Fabric Software Development Kit (**SDK**) [99–102] provides the corresponding interface. The application calls the *SDK* interface and then submits the transaction proposal through transaction management.

3.1.3 Chaincode

This chaincode is similar to other smart contracts in the blockchain network, such as Etherium. It is a decentralized transaction program that executes the verification code. The code service on the chain uses *Docker* to store the code on the chain without relying on a specific virtual machine or computer language [103].

3.1.4 Channel

In Fabric, the channels isolate the Blockchain data of different organizations, and the nodes of organizations in different channels cannot be directly accessed across channels.

3.1.5 Consensus

Hyperledger Fabric consensus mechanism, currently including Solo and Kafka. A consensus cluster consists of multiple *Orderers* and a Kafka cluster. The *Orderer* does not communicate directly and just communicates with the Kafka cluster. All messages received from *Peers* are sorted and generated as a block at this node, and then consensus is achieved using the Kafka sorting function.

3.1.6 Ledger

The Fabric ledger consists of the state of the world and the Blockchain together. Blockchain data is stored on *Committer* using a file system. Data is transmitted, packaged, and stored in this ledger in the form of transactions. The verified Tx_i will be submitted to an *Orderer* to be packaged into a block, and permanently stored by a *Peer*.

3.2 Supply Chain Finance

SCF uses self-repaying trade finances based on the core enterprise. Compared to traditional credit business, SCF focuses more on the real trade background between the upstream and downstream of the supply chain. A typical SCF framework consists of a core company, suppliers, distributors, and financial institutions, and an example is shown in Fig.2.

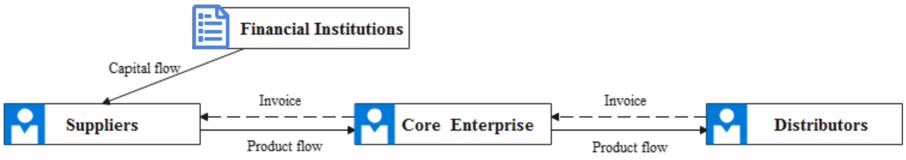


Fig. 2 An example of SCF framework.

Based on different financing directions and mortgage targets, SCF businesses have three main types: accounts receivable financing, inventory financing, and prepayment financing.

3.2.1 Accounts Receivable Financing

The accounts receivable financing model is a financing business provided by a firm based on accounts receivable arising from a real trade between a seller and a buyer, which is repaid by the contracted accounts receivable. Specifically, the supplier Sp_i first enters into a transaction with the core enterprise CE_i, which is downstream in the supply chain as $Sp \xrightarrow{P_i - j} CE$, and the core enterprise CE_i issues an accounts receivable document to the supplier Sp_i as $CE \xrightarrow{ARD} Sp$. Then, the supplier Sp_i assigns the receivable document ARD_i to the financial institution FI_i, and the downstream supplier in the supply chain makes a payment commitment to the financial institution FI_i as $Sp \xrightarrow{ARD} FI$. The financial institution FI_i provides a credit facility CF_i to the supplier Sp_i as $FI \xrightarrow{CF} Sp$.

3.2.2 Inventory Financing

Inventory financing uses the productions $P_i{-}_j$ in the trade process as the collateral Col_i for financing. This issue occurs when the supply of Sp_i has a large inventory or slow inventory turnover, which leads to high pressure on capital turnover and enterprises using existing products. $P_i{-}_j$ get cash in advance as $Sp \xrightarrow{P_i - j} FI$. The financial institution FI_i can issue the corresponding credit facility CF_i based on the evaluation of the third-party regulatory company as $FI \xrightarrow{CF} Sp$. If the SME loses the ability to repay during the repayment period, the financial institution FI_i can find the core enterprise CE, sign a purchase agreement with the core enterprise CE as $FI \xrightarrow{P_i - j} CE$, and complete the repayment as $CE \xrightarrow{Cash} FI$.

3.2.3 Prepayment Financing

Based on inventory financing, prepayment financing has been developed. The buyer Br_i negotiates full payment to the seller Sr_i on the condition that a percentage of the deposit is paid. The seller Sr_i ships the productions $P_i{-}_j$ following the purchase and sale contract also the cooperation agreement, and the productions $P_i{-}_j$ arrive with a pledge as security Col_i for the financial institution Fi_i. In the specific process, the core enterprise CE_i provides a purchase and sale contract PC_i to the distributor Dt_i. Next, the distributor Dt_i acts as a financing enterprise FE_i to retrieve credit facility as $CE \xrightarrow{P_i - j} FI$ and $Fi_i \xrightarrow{CF} Dt_i$. Distributor Dt_i submits the delivery deposit to the financial institution Fi_i in installments, and the financial institution Fi_i then informs the core enterprise in installments to deliver the productions $P_i{-}_j$ to the distributor Dt_i.

4 System Model and Design

In this section, we show the architectural design of Fabric-SCF and corresponding details. The designed system architecture is presented in Subsection 4.1, while the defined data structure in Subsection 4.2. Next, details of the smart contract design are depicted in Subsection 4.3, and lastly, the workflow of Fabric-SCF in Section 4.4.

4.1 System Architecture

The proposed Fabric-SCF is composed of a consortium Blockchain, with user, admin, and user access modules forming the underlying structure, as shown in Fig. 3. **Blockchain** is a distributed network composed of trusted nodes for data synchronization and storage, in which authentication is realized through digital certificates to ensure the security and integrity of data in the system.

User contains business participants, which logically include a core enterprise CE, one or more supplier SP_i, and one or more Distributors Dt_j.

Admin is responsible for managing access to the blockchain system. After entering the Blockchain system via the validation of CA, it needs to develop access policies, distribute user access to data, and maintain access policies.

User Access is a collection of access commands. In this network, Fabric-SCF allows all parties involved in SCF to be authorized to send data flows, access data, and perform $add()$, $query()$, and $update()$ functions.

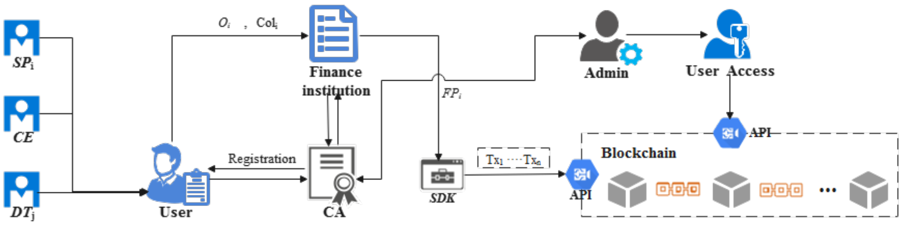


Fig. 3 Architecture of Fabric-SCF.

Three types of business correspond to different stages of the supply chain. All the credit facility CF_i is provided by financial institutions FI_i, based on the stage of each business with the core enterprise CE. Fabric-SCF system utilizes Blockchain as a database and is divided into a supply-side and a distribution-side in business processes, with the core enterprise as the center. Sellers Sr_i and buyer Br_j can be unified to supplier Sp_i and distributor Dt_j. For financing business, supplier Sp_i and distributor Dt_j can be grouped as financing companies FE_i or FE_j as users. The financial institution FI_i uses inventory $P_i{-}_j$, accounts receivable documents ARD_i or prepayment contracts PC_i as collateral Col_i and the financing order FO_i to lend credit facility CF_i to the financing enterprise FE_i based on the business transactions between the financing enterprise FE_i and the core enterprise CE. All users and financial institutions FI need to be authorized by CA, and all data access requests need to go through $Admin$. Smart contracts control the data flow, and the corresponding data is stored in the ledger in the Blockchain, thus forming a closed loop of data exchange.

4.2 Data Structure

Blockchain can be summarized as a distributed state machine. To ensure that the world state can be updated quickly when the transaction is executed, the design and implementation of the world state scheme should consider the fast search and efficient update of the state data. To ensure system efficiency and reduce system latency, nine data structures are designed in this research, and the index relationship is shown in Fig. 4.

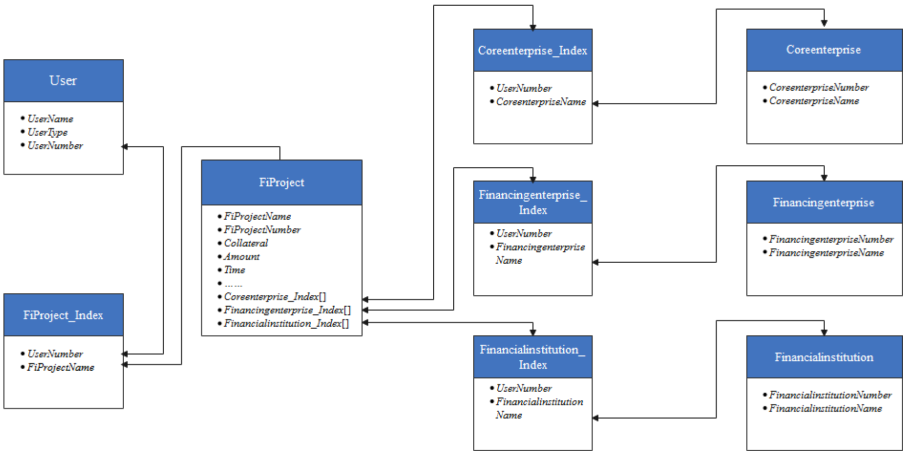


Fig. 4 Relationship between data structures.

Defining the data structures in this way facilitates the design of uniform data access operations and reduces the system's complexity. The multiplicity of mappings between data structures is shown in Eq. 1.

$$\begin{cases} User \to FC\_Index & = 1:n \\ FC\_Index \to FiProject & = 1:1 \\ FC \to CE\_Index & = 1:1 \\ FC \to FE\_Index & = 1:1 \\ FC \to Fi\_Index & = 1:1 \\ CE\_Index \to CE & = 1:1 \\ FE\_Index \to FE & = 1:1 \\ FI\_Index \to FI & = 1:1 \end{cases} \tag{1}$$

4.2.1 User

The data structure of $User$ corresponds to the system role of $User$, and it mainly contains three parts. $UserName$ directly reflects the user's name, and $UserType$ corresponds to the main participants of the system process SP_i, Dt_j, CE. In addition, $UserNumber$ is calculated by Eq. 2.

$$UserNumber_i = HV(UserName_i, pubKey_i) \tag{2}$$

4.2.2 FiProject

As the core data block, it contains many attributes. *FiProject* has a one-to-one correspondence with *FiProject_Index*, and the core segments are as follows.

i) *Collateral*: The collateral Col_i of the financing enterprise FE_i to the financial institution FI_i when handling the financing business includes inventory P_i-_j, accounts receivable document ARD_i, and prepayment Pm_i as Eq. 3.

$$Col_i = \begin{cases} P_i-_j \\ ARD_i \\ Pm_i \end{cases} \tag{3}$$

ii) *Amout*: The interest rate and amount of the credit facility CF_i specified in the financing project.

iii) *Time*: The timeline of the credit facility CF_i in the financing project FP_i.

iv) *Index*: The index of the core enterprise CE, Financing enterprise FI_i and financial institution FI_i related to the financing project FP_i.

4.3 Smart Contract Design

Smart contracts are the key to realizing this system's business logic and access control. To focus on maximizing data privacy and integrity while ensuring the system's efficiency, the smart contract designed in this research is from the perspective of user management, financing project management, and access control.

4.3.1 User Management Contract

In the proposed system, user management mainly involves creation and query. To ensure the uniqueness of the identities of participants from multiple organizations, each user's identity is stored in the state database of the Blockchain, as shown in Fig. 5.

i) *createUser*(): *User* registration is the first step for to participate in the system, and *createUser*() needs to generate a *UserNumber* based on the *UserName* and *UserType* as a unique identifier.

ii) *queryUser*(): This function can query the *user* information based on *UserName* and *UserNumber*. In addition, it is also needed to check if the user already exists before a member is registered.

iii) *checkUser*(): If *user* intends to be authorized by an access policy, it needs to pass on verification first; that is, to have the *Ct*, *SingnID*, *Pubk*, *Prik* and *UserType* checked.

```
{
"userName": "user1",
"userNumber": "1a462ae9c1350c24ca8e389090946c7ed517667231c7b6ae22d0fed39ca55dd1",
"userType": "CE"
}
```

Fig. 5 An example of member data.

4.3.2 Financing Project Management Contract

According to user requests, related financing project data access operations mainly include $addFiproject()$, $queryFiproject()$, $updateFiproject()$, and $deleteFiproject()$. Moreover, $checkFiproject()$ and $checkUdvInfo()$ modules are also defined to ensuring the legitimacy and reliability. These operations correspond to different entities, different authorizations, and different processes.

i) $addFiproject()$: When a user submits a request to add a contract, $checkFiproject()$ will be triggered first to check the reasonableness of the contract information. If the contract information is deemed reasonable, the contract will be published, so the contract and all operations specified are packaged and written to the Blockchain ledger. The adding and judgment logic are shown in **Algorithm** 1.

Algorithm 1 addFiProject()

```
Require: FiProject(FP)
Ensure: Ok or Error
APIstub ChaincodeStub ← Invoke();
if CheckFiProject(FP) == False then
    return Error;
end if
FPid ← Sha256(FP.FiProjectName + FPFiProjectNumber);
if QueryFiProject(FPid) != Null then
    return Error;
end if
err ← APIstub.PutState(Id, FP);
if err != null then
    return Error;
end if
return Ok;
```

ii) $queryFiproject()$: The query function is ubiquitous and all users can implement queries for details related to the authorized contract data. Both the financing project name $FiProjectName$ and financing project number $FiProjectNumber$ can be used as the basis for queries.

iii) $updateFiproject()$: This function updates the order information. In exceptional cases, the contract details such as contract term, interest rate due, among others, need to be changed. Similar to $addFiproject()$, the update process first triggers $checkUdvInfo()$ to check the reasonableness of the updated information and posts the verified update information, updating the ledger last.

iv) $deleteFiproject()$: Deleting a contract usually happens by mistake or when a contract is re-entered. When the user issues a request to delete an order, $checkFiproject()$ is first triggered to check the reasonableness of the contract information, so then $queryFiproject()$ is called to check the status of the order.

v) $checkFiproject()$: When the user submits $addFiproject()$, this method is used to check whether the key fields of the financing project information are reasonable.

vi) $checkUdvInfo()$: This method is used to verify the legitimacy of the update related to $updateFiproject()$.

4.3.3 Access control contract

The access control management contract is used to verify that a user's data access request satisfies the access control policy set by the administrator and consists mainly of $Auth()$, $GetAttrs()$, $CheckPolicy()$ and $CheckAccess()$.

i) $Auth()$: The public key distributed to the user by $Admin$ can be used to encrypt the request. This method is used to authenticate the user request and further determine the user's identity.

ii) $GetAttrs()$: The attribute fields in the user's request are resolved by calling the $GetAttrs()$ module after verifying the user's identity. The attribute fields contained in the request are the subject attribute and the object attribute $\{S, O\}$. After further determining the environment attribute E, the attribute combination can be obtained as $\{S, O, E\}$.

iii) $CheckAccess()$: This method is the core on the implementation of access control management. First, the S, O attribute is obtained via the $GetAttrs()$ module, and then the $QueryPolicy()$ module is called to obtain the corresponding ABACPolicy. If the method returns null, then no policy supports the request and such a request is invalid. Though, the request is verified by validating the policy that matches, and if the attributes E and A in the policy are satisfied, the request passes the verification. The pseudo-code of the $CheckAccess()$ is shown in **Algorithm** 2.

.

4.3.4 Policy Contract

The PC provides the following methods to operate ABACPolicy.

i) $AddPolicy()$: The CA needs to run the $CheckPolicy()$ module before calling this method to pass it to add the policy, and the policy can only be written to the SDB and the Blockchain after the policy is legal. The details are shown in **Algorithm** 3.

ii) $DeletePolicy()$: This operation is called in two ways. Firstly, the administrator could actively call on it to delete an ABACPolicy. Secondly, if a policy

Algorithm 2 CheckAccess()

```
Require: ABAC_Request
Ensure: OK or Error
 1: $\{S_u, O_u, E_u\} \leftarrow$ GetAttrs(ABAC_Request);
 2: Policy =$\{Policy_1,...Policy_n\} \leftarrow QueryPolicy(S_u, O_u)$;
 3: if Policy == Null then
 4:     return Error();
 5: end if
 6: for Policy in $\{Policy_1,...Policy_n\}$ do
 7:     $\{. . . , P_p, E_p\} \leftarrow$ Policy
 8:     if Value(Pp) == 1 && Eu $\cap$ Ep !=Null then
 9:         return OK;
10:     end if
11: end for
12: return Error();
```

Algorithm 3 AddPolicy()

```
Require: Policy
Ensure: OK or Error
 1: APIstub ChaincodeStub $\leftarrow$ Invoke();
 2: if CheckPolicy(Policy) == False then;
 3:     return Error;
 4: end if
 5: Policy$_{id} \leftarrow$ Hash(Policy.S + Policy.O);
 6: if PolicyContract.QueryPolicy(Policy$_{id}$) != Null then
 7:     return Error;
 8: end if
 9: err $\leftarrow$ APIstub.PutState(Policy$_{id}$, Policy);
10: if err! = null then;
11:     return Error;
12: end if
13: return OK;
```

is found to have expired during the execution of the *CheckAccess*() module, then this method will be called automatically to delete the useless policy.

iii) *UpdatePolicy*(): This module can be called in two ways. Firstly, the administrator can actively call this module to delete an ABACPolicy, and it occurs when the administrator needs to modify an ABACPolicy. A modified record must be written into the SDB and the Blockchain when calling this method. After completing the policy update, the module executes the *addPolicy*() module and enters the modified policy into the Blockchain.

iv) *QueryPolicy*(): All the strategies are stored in the Status Database CouchDB, and the administrator can query the details of the ABACPolicy by attributes S or O.

4.4 Workflow

The system workflow consists of five parts. 1) Blockchain network initialization. 2) Access policy deployment. 3) User registration. 4) Financial project management. 5) User access assignment. We present the details of the five parts in Fig 6.

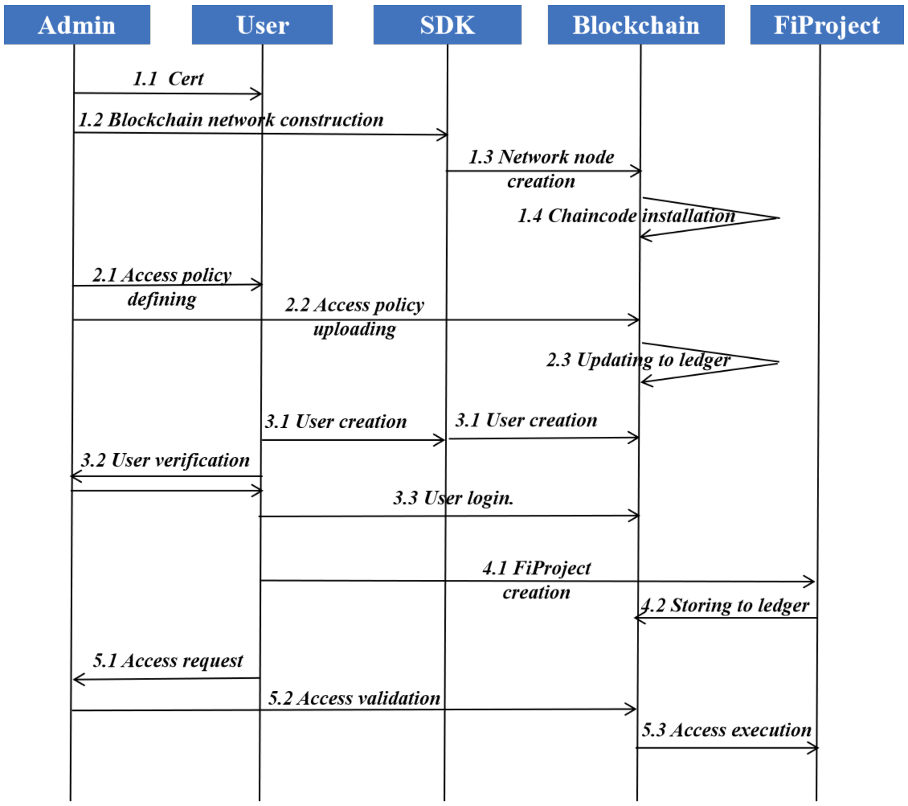


Fig. 6 Workflow

5 Performance Evaluation and Analysis

This section describes the experimental procedure to evaluate the functionality and presents the proposed Fabric-SCF's performance and comparison results. We begin in Section 5.1 with a description of the environment and experiments' parameter setup, followed by the system implementation in Section 5.2. The performance of the proposed system and experimental results are discussed in Section 5.3.

5.1 System environment and configuration

The experimentation processing for this work is conducted in a single machine environment, configured as outlined below.

5.1.1 Network architecture

The scheme consists of seven network nodes, as shown in Table 2.

Table 2 Number of network nodes

Node name	Description	Number
Fabric-couchdb	Database node	4
Fabric-ca	CA node	2
Fabric-peer	peer node	4
Fabric-orderer	orderer node	1
Fabric-tools	cli node	1
PSC	policy smart contract node	4
ASC	access smart contract node	4

5.1.2 Chaincode deployment

In Fabric-SCF, the attribute delimitation, access control, and system operations are implemented by the chaincode, which setup consists of installation, instantiation, and upgrade blocks.

i): Installation: As soon as the Blockchain network's initialization is completed, the installation of the chaincode starts. Chaincode installation is performed through Hyperledger client nodes that install the chaincode into each peer node in turn.

ii): Instantiation: After the installation of the chaincode is completed, any peer node is designated to instantiate the installed chaincode.

iii): Upgrade: Before a chaincode can be updated, a new chaincode must be installed, i.e., the chaincode update is only valid on the peer node where the new chaincode has been installed.

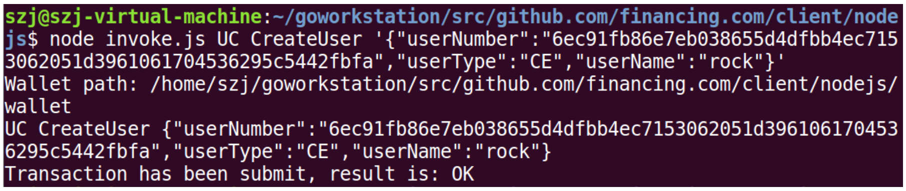
5.2 System implementation

It is depicted in this section the operational effects of three smart contracts for a clearer understanding. Three operation results: User management contract, Finance management contract, and Access control contract are presented next.

5.2.1 User management contract

As stated in section 4.3.1, this part of the method focuses on the creation, query, and deletion of each participant SP_i, Dt_j, CE in the system. The steps are detailed below.

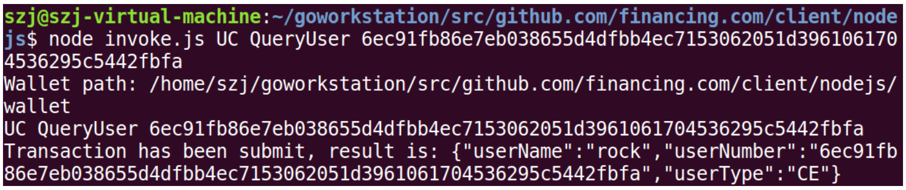
$CreateUser()$: This module is called when adding a user, and automatically generates a user number based on the user name and type, as shown in Fig. 7.



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js UC CreateUser '{"userNumber":"6ec91fb86e7eb038655d4dfbb4ec715
3062051d3961061704536295c5442fbfa","userType":"CE","userName":"rock"}'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
UC CreateUser {"userNumber":"6ec91fb86e7eb038655d4dfbb4ec7153062051d396106170453
6295c5442fbfa","userType":"CE","userName":"rock"}
Transaction has been submit, result is: OK
```

Fig. 7 System response to calling the $CreateUser()$ module.

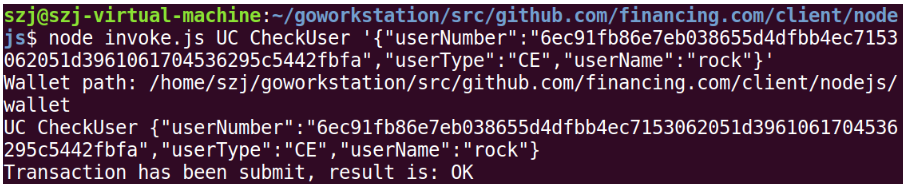
$QueryUser()$: To query the details of a user, calling this module will query the user's information based on the user number, as shown in Fig. 8.



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js UC QueryUser 6ec91fb86e7eb038655d4dfbb4ec7153062051d396106170
4536295c5442fbfa
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
UC QueryUser 6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa
Transaction has been submit, result is: {"userName":"rock","userNumber":"6ec91fb
86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa","userType":"CE"}
```

Fig. 8 System response to calling the $QueryUser()$ module.

$CheckUser()$: This module checks a user's identity, including its certificate, private key and identity type, and other information. It returns OK if the check passes; otherwise, it returns Error, as illustrated in Fig. 9.



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js UC CheckUser '{"userNumber":"6ec91fb86e7eb038655d4dfbb4ec7153
062051d3961061704536295c5442fbfa","userType":"CE","userName":"rock"}'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
UC CheckUser {"userNumber":"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536
295c5442fbfa","userType":"CE","userName":"rock"}
Transaction has been submit, result is: OK
```

Fig. 9 System response to calling the $CheckUser()$ module.

5.2.2 Financing project management contract

As introduced in section 4.3.2, this part of the chaincode mainly realizes the functions of adding, querying, updating, and deleting financial projects.

$AddFiProject()$: This module is called to write project information to the SDB when project information needs to be added, as shown in Fig. 10.

$QueryFiProject()$: This module is called to view information about a particular project according to its project number, thus enabling the retrieval of detailed information about a financial project, as shown in Fig. 11.

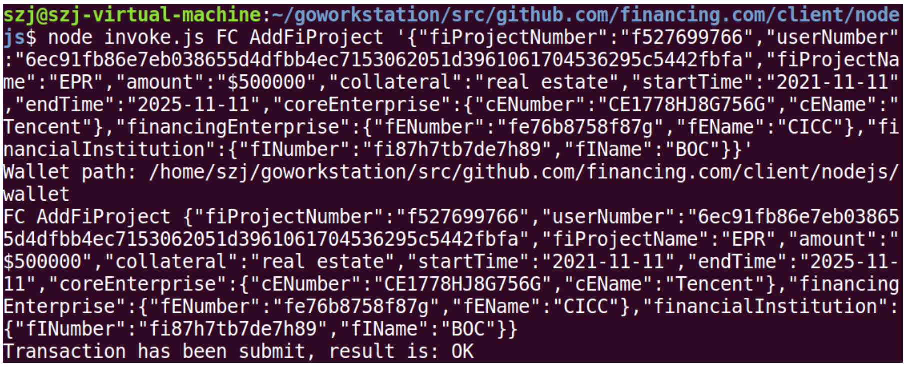

```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js FC AddFiProject '{"fiProjectNumber":"f527699766","userNumber"
:"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa","fiProjectNa
me":"EPR","amount":"$500000","collateral":"real estate","startTime":"2021-11-11"
,"endTime":"2025-11-11","coreEnterprise":{"cENumber":"CE1778HJ8G756G","cEName":"
Tencent"},"financingEnterprise":{"fENumber":"fe76b8758f87g","fEName":"CICC"},"fi
nancialInstitution":{"fINumber":"fi87h7tb7de7h89","fIName":"BOC"}}'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
FC AddFiProject {"fiProjectNumber":"f527699766","userNumber":"6ec91fb86e7eb03865
5d4dfbb4ec7153062051d3961061704536295c5442fbfa","fiProjectName":"EPR","amount":"
$500000","collateral":"real estate","startTime":"2021-11-11","endTime":"2025-11-
11","coreEnterprise":{"cENumber":"CE1778HJ8G756G","cEName":"Tencent"},"financing
Enterprise":{"fENumber":"fe76b8758f87g","fEName":"CICC"},"financialInstitution":
{"fINumber":"fi87h7tb7de7h89","fIName":"BOC"}}
Transaction has been submit, result is: OK
```

Fig. 10 System response to calling the *AddFiProject*() module.

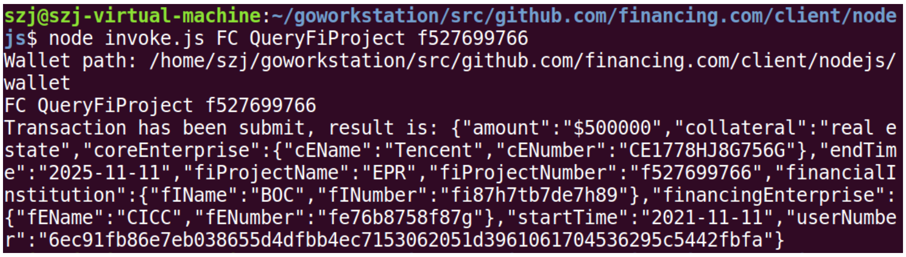

```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js FC QueryFiProject f527699766
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
FC QueryFiProject f527699766
Transaction has been submit, result is: {"amount":"$500000","collateral":"real e
state","coreEnterprise":{"cEName":"Tencent","cENumber":"CE1778HJ8G756G"},"endTim
e":"2025-11-11","fiProjectName":"EPR","fiProjectNumber":"f527699766","financialI
nstitution":{"fIName":"BOC","fINumber":"fi87h7tb7de7h89"},"financingEnterprise":
{"fEName":"CICC","fENumber":"fe76b8758f87g"},"startTime":"2021-11-11","userNumbe
r":"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa"}
```

Fig. 11 System response to calling the *QueryFiProject*() module.

UpdateFiProject(): This approach is called when a project needs to be updated to modify the financial information. Before this module can be executed, the *CheckUdvInfo*() module must be performed to determine if the update permission has been granted, as shown in Fig. 12.

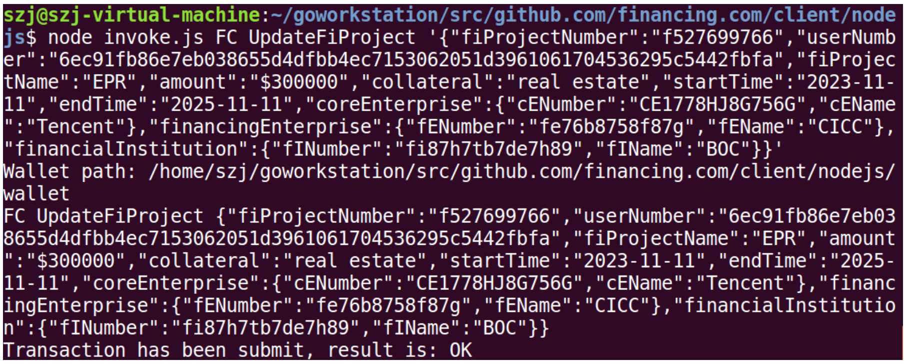
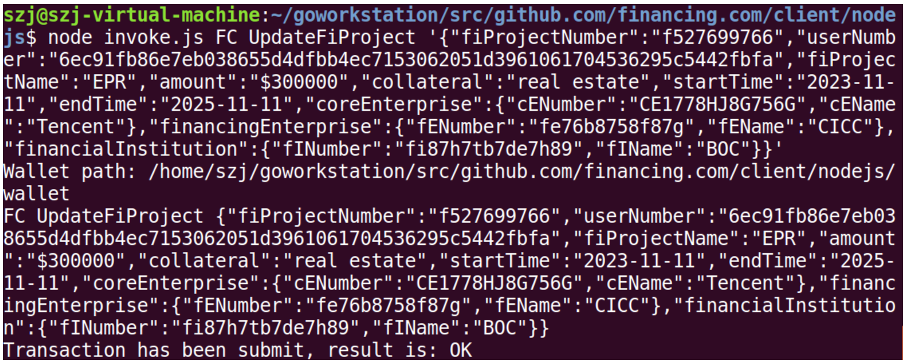

```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js FC UpdateFiProject '{"fiProjectNumber":"f527699766","userNumb
er":"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa","fiProjec
tName":"EPR","amount":"$300000","collateral":"real estate","startTime":"2023-11-
11","endTime":"2025-11-11","coreEnterprise":{"cENumber":"CE1778HJ8G756G","cEName
":"Tencent"},"financingEnterprise":{"fENumber":"fe76b8758f87g","fEName":"CICC"},
"financialInstitution":{"fINumber":"fi87h7tb7de7h89","fIName":"BOC"}}'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
FC UpdateFiProject {"fiProjectNumber":"f527699766","userNumber":"6ec91fb86e7eb03
8655d4dfbb4ec7153062051d3961061704536295c5442fbfa","fiProjectName":"EPR","amount
":"$300000","collateral":"real estate","startTime":"2023-11-11","endTime":"2025-
11-11","coreEnterprise":{"cENumber":"CE1778HJ8G756G","cEName":"Tencent"},"financ
ingEnterprise":{"fENumber":"fe76b8758f87g","fEName":"CICC"},"financialInstitutio
n":{"fINumber":"fi87h7tb7de7h89","fIName":"BOC"}}
Transaction has been submit, result is: OK
```

Fig. 12 System response to calling the *UpdateFiProject*() module.

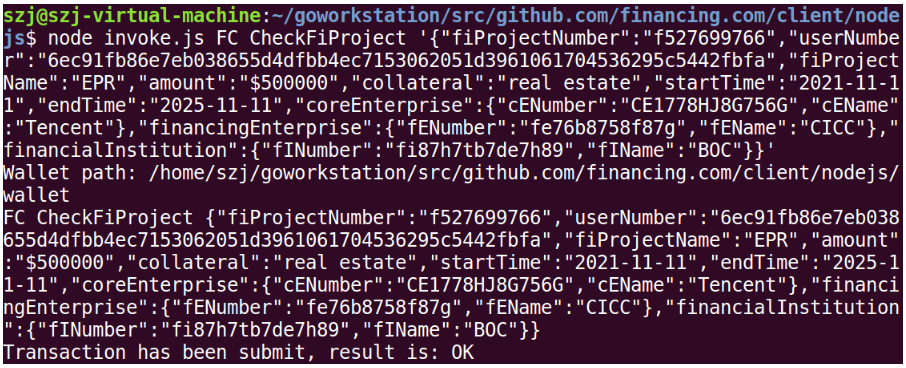
DeleteFiProject(): This module is executed if the system needs to delete project information from the SDB as shown in Fig. 13.



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js FC UpdateFiProject '{"fiProjectNumber":"f527699766","userNumb
er":"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa","fiProjec
tName":"EPR","amount":"$300000","collateral":"real estate","startTime":"2023-11-
11","endTime":"2025-11-11","coreEnterprise":{"cENumber":"CE1778HJ8G756G","cEName
":"Tencent"},"financingEnterprise":{"fENumber":"fe76b8758f87g","fEName":"CICC"},
"financialInstitution":{"fINumber":"fi87h7tb7de7h89","fIName":"BOC"}}'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
FC UpdateFiProject {"fiProjectNumber":"f527699766","userNumber":"6ec91fb86e7eb03
8655d4dfbb4ec7153062051d3961061704536295c5442fbfa","fiProjectName":"EPR","amount
":"$300000","collateral":"real estate","startTime":"2023-11-11","endTime":"2025-
11-11","coreEnterprise":{"cENumber":"CE1778HJ8G756G","cEName":"Tencent"},"financ
ingEnterprise":{"fENumber":"fe76b8758f87g","fEName":"CICC"},"financialInstitutio
n":{"fINumber":"fi87h7tb7de7h89","fIName":"BOC"}}
Transaction has been submit, result is: OK
```

Fig. 13 System response to calling the *DeleteFiProject*() module.

CheckFiProject(): This module is used to verify the legitimacy of a financial project and to write a complete project message to the SDB once it has been verified, as shown in Fig. 14.



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js FC CheckFiProject '{"fiProjectNumber":"f527699766","userNumbe
r":"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa","fiProject
Name":"EPR","amount":"$500000","collateral":"real estate","startTime":"2021-11-1
1","endTime":"2025-11-11","coreEnterprise":{"cENumber":"CE1778HJ8G756G","cEName"
:"Tencent"},"financingEnterprise":{"fENumber":"fe76b8758f87g","fEName":"CICC"},"
financialInstitution":{"fINumber":"fi87h7tb7de7h89","fIName":"BOC"}}'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
FC CheckFiProject {"fiProjectNumber":"f527699766","userNumber":"6ec91fb86e7eb038
655d4dfbb4ec7153062051d3961061704536295c5442fbfa","fiProjectName":"EPR","amount"
:"$500000","collateral":"real estate","startTime":"2021-11-11","endTime":"2025-1
1-11","coreEnterprise":{"cENumber":"CE1778HJ8G756G","cEName":"Tencent"},"financi
ngEnterprise":{"fENumber":"fe76b8758f87g","fEName":"CICC"},"financialInstitution
":{"fINumber":"fi87h7tb7de7h89","fIName":"BOC"}}
Transaction has been submit, result is: OK
```

Fig. 14 System response to calling the *CheckFiProject*() module.

5.2.3 Access control contract

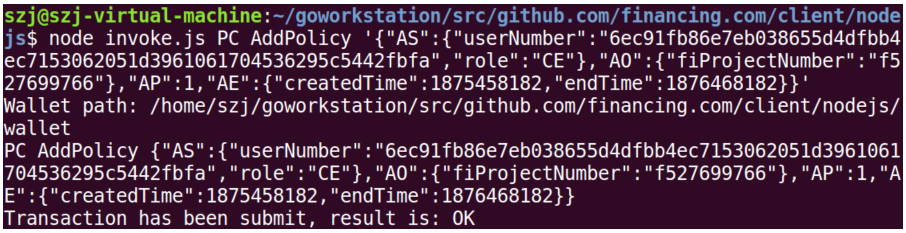
As presented in section 4.3.3, this part of the chaincode is mainly used to implement auditable access control for participants.

AddPolicy(): This module is used to store the specified policy in the status database after it has been agreed upon between the administrator and the user, as shown in Fig. 15.

QueryPolicy(): This module is used to determine if a user has access rights, found by querying the user's number as shown in Fig. 16.

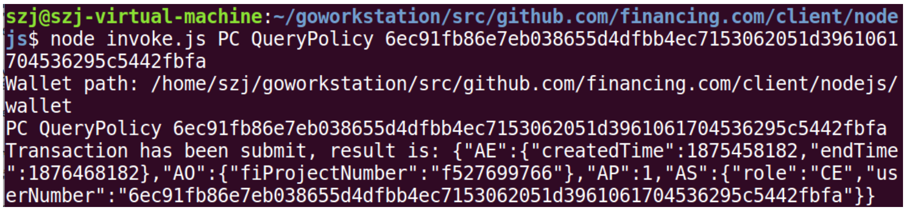
UpdatePolicy(): This module is called upon to update the policy after the access requirements have changed, as shown in Fig. 17.

DeletePolicy(): The module called upon to remove a policy if it is no longer valid or if a policy must be removed, as illustrated in Fig. 18.



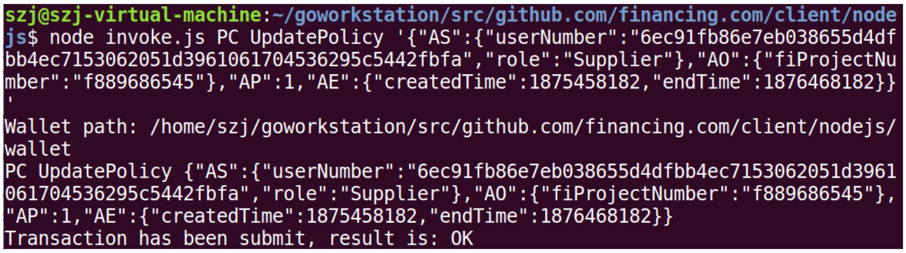
```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js PC AddPolicy '{"AS":{"userNumber":"6ec91fb86e7eb038655d4dfbb4
ec7153062051d3961061704536295c5442fbfa","role":"CE"},"AO":{"fiProjectNumber":"f5
27699766"},"AP":1,"AE":{"createdTime":1875458182,"endTime":1876468182}}'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
PC AddPolicy {"AS":{"userNumber":"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061
704536295c5442fbfa","role":"CE"},"AO":{"fiProjectNumber":"f527699766"},"AP":1,"A
E":{"createdTime":1875458182,"endTime":1876468182}}
Transaction has been submit, result is: OK
```

Fig. 15 System response to calling the *AddPolicy*() module.



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js PC QueryPolicy 6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061
704536295c5442fbfa
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
PC QueryPolicy 6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa
Transaction has been submit, result is: {"AE":{"createdTime":1875458182,"endTime
":1876468182},"AO":{"fiProjectNumber":"f527699766"},"AP":1,"AS":{"role":"CE","us
erNumber":"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa"}}
```

Fig. 16 System response to calling the *QueryPolicy*() module.



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js PC UpdatePolicy '{"AS":{"userNumber":"6ec91fb86e7eb038655d4df
bb4ec7153062051d3961061704536295c5442fbfa","role":"Supplier"},"AO":{"fiProjectNu
mber":"f889686545"},"AP":1,"AE":{"createdTime":1875458182,"endTime":1876468182}}
'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
PC UpdatePolicy {"AS":{"userNumber":"6ec91fb86e7eb038655d4dfbb4ec7153062051d3961
061704536295c5442fbfa","role":"Supplier"},"AO":{"fiProjectNumber":"f889686545"},
"AP":1,"AE":{"createdTime":1875458182,"endTime":1876468182}}
Transaction has been submit, result is: OK
```

Fig. 17 System response to calling the *UpdatePolicy*() module.

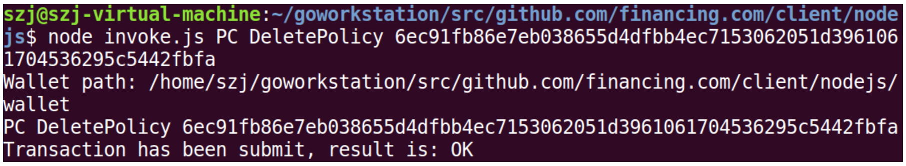
CheckAccess(): After receiving a request from the user, the *CheckAccess*() module in AC will be called automatically to verify whether the request is valid or not, as shown in Fig. 19.

5.3 Performance tests and experimental results

To validate the performance of the proposed Fabric-SCF, we designed two sets of simulation experiments to simulate the concurrency, transaction time, and throughput of the system under real-life usage conditions.

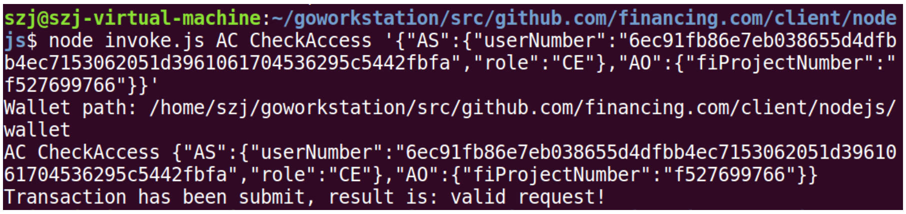
5.3.1 Execution time of the contract

Following the system usage logic, we first tested the system response time of the user management contract, the financing project management contract, and the access control contract separately by module, as shown in Fig. 20. Experimental results show that with the increase of block size, the contract execution time shows an orderly growth trend and that the growth is controllable. Obviously, the execution time for the *add*() and *update*() contracts



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js PC DeletePolicy 6ec91fb86e7eb038655d4dfbb4ec7153062051d396106
1704536295c5442fbfa
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
PC DeletePolicy 6ec91fb86e7eb038655d4dfbb4ec7153062051d3961061704536295c5442fbfa
Transaction has been submit, result is: OK
```

Fig. 18 System response to calling the *DeletePolicy*() module.



```
szj@szj-virtual-machine:~/goworkstation/src/github.com/financing.com/client/node
js$ node invoke.js AC CheckAccess '{"AS":{"userNumber":"6ec91fb86e7eb038655d4dfb
b4ec7153062051d3961061704536295c5442fbfa","role":"CE"},"AO":{"fiProjectNumber":"
f527699766"}}'
Wallet path: /home/szj/goworkstation/src/github.com/financing.com/client/nodejs/
wallet
AC CheckAccess {"AS":{"userNumber":"6ec91fb86e7eb038655d4dfbb4ec7153062051d39610
61704536295c5442fbfa","role":"CE"},"AO":{"fiProjectNumber":"f527699766"}}
Transaction has been submit, result is: valid request!
```

Fig. 19 System response to calling the *CheckAccess*() module.

are higher than those for *query*() and *delete*() contracts due to the number of data write operations involved, however the difference is small and reasonably manageable.

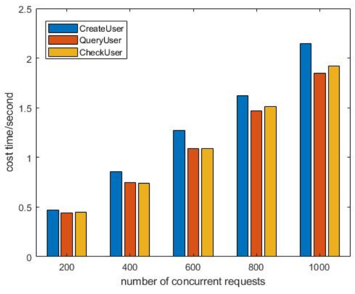
5.3.2 TPS of the contract

In the second group of experiments, we compared the TPS of Fabric-SCF under different concurrent numbers under real simulated conditions. Under the Solo and Kafka consensus mechanism, the throughput of the proposed system can be maintained at a relatively smooth level, and none of them exceeds the threshold of 550, as shown in Fig. 21. In conclusion, the Fabric-SCF system maintains high TPS with relatively consistent performance output under intensive requests.

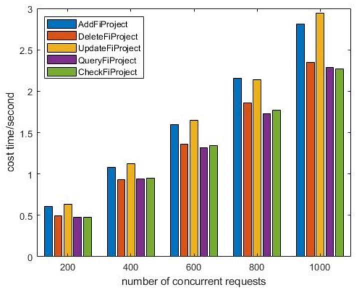
6 Conclusions and Future Work

In this article, we propose a blockchain-based SCF management system to enable secure data storage and auditable access control. To accommodate the changing SCF business logic and increasing privacy requirements, the proposed system enables cross-organizational data management and access while ensuring secure storage, efficient access, and access traceability of privacy-sensitive data. With the combination of consensus mechanisms and smart contracts, financial project data in SCF can be securely stored in the Blockchain. Meanwhile, the access policy of the proposed system supports programmability and thus can be modified according to different business processes. Thanks to the ABAC access control mechanism, participating parties can have auditable access, provided they are fully authorized. Finally, simulation experiments confirm that Fabric-SCF can guarantee system stability and TPS at larger business volumes.

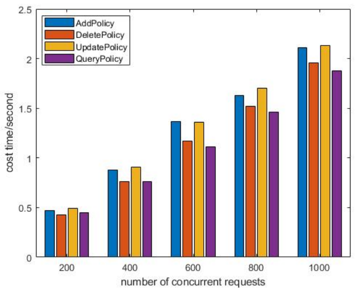
As future work, we intend to investigate the following issues:



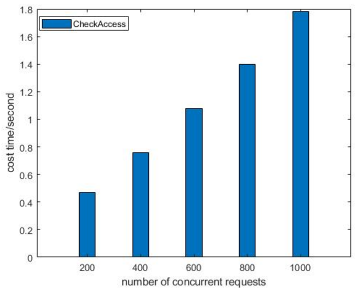
(a) User management contract



(b) Financing project management contract



(c) Access control contract



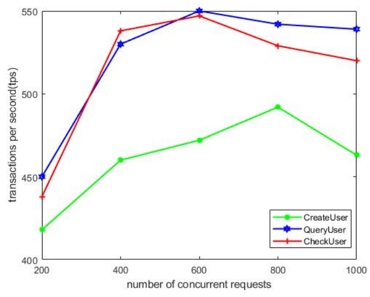
(d) Check access contract

Fig. 20 Execution time under different block sizes.

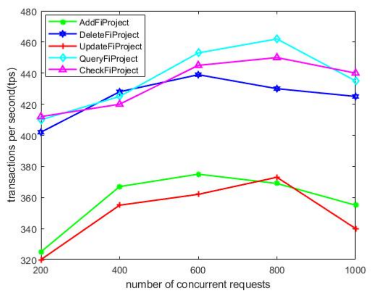
1. As for now, the business logic of SCF is relatively simple. In the future, we will consider introducing more complex business processes to experience the system's universality in the financial field.
2. Using large-scale test environments such as changing stand-alone environments to multi-machine environments or testing a real dataset to simulate usage scenarios more realistically.
3. Combining the data storage on the chain with the data storage mode under the chain to optimize data storage while streamlining the design.
4. Evaluations on the deployment of Fabric-SCF in a broader range of platforms, such as Ethereum, to verify the throughput and execution time together with fuel consumption.

Funding

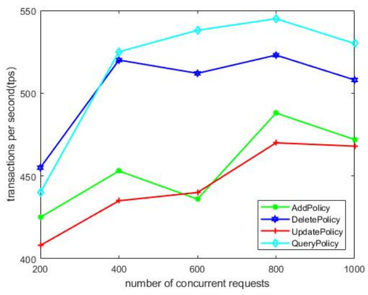
This research is supported by the National Natural Science Foundation of China under Grant 61873160, Grant 61672338, the Natural Science Foundation of Shanghai under Grant 21ZR1426500 and Top-notch innovative program of Shanghai Maritime University under Grant 2021YBR007.



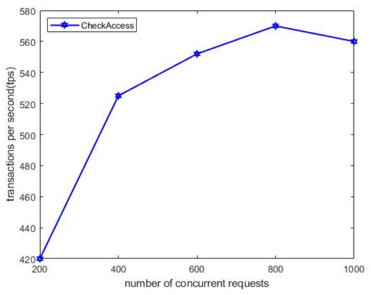
(a) User management contract



(b) Financing project management contract



(c) Access control contract



(d) Check access contract

Fig. 21 Execution time under different block sizes.

Declarations

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Availability of data and materials

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

References

[1] Zulqurnain, A., Bi, G., Mehreen, A.: Does supply chain finance improve smes performance? the moderating role of trade digitization. Bus. Process. Manag. J. **26**, 150–167 (2020)

[2] Yue, H., Xu, J., Wang, J., Wu, D., Niu, B.: A general introduction and overview of supply chain finance. (2019)

[3] Somjai, S., Vasuvanich, S., Laosillapacharoen, K., Suteerachai, B.: Governing role of trade digitalization in global supply chain finance, negotiation and smes performance. International Journal of Supply Chain Management **8**, 660–672 (2019)

[4] Gelsomino, L., Mangiaracina, R., Perego, A., Tumino, A.: Supply chain finance: a literature review. International Journal of Physical Distribution & Logistics Management **46**, 348–366 (2016)

[5] Caniato, F., Henke, M., Zsidisin, G.: Supply chain finance: Historical foundations, current research, future developments. Journal of Purchasing and Supply Management **25**, 99–104 (2019)

[6] Guo, S., Liu, N.: Influences of supply chain finance on the mass customization program: risk attitudes and cash flow shortage. Int. Trans. Oper. Res. **27**, 2396–2421 (2020)

[7] Omran, Y., Henke, M., Heines, R., Hofmann, E.: Blockchain-driven supply chain finance: Towards a conceptual framework from a buyer perspective. (2017)

[8] Su, C., Lyu, W., Liu, Y.: The relationship between digital rmb and digital economy in china. arXiv preprint arXiv:2205.14517 (2022)

[9] Li, D., Han, D., Liu, H.: Fabric-chain & chain: A blockchain-based electronic document system for supply chain finance. In: BlockSys (2020)

[10] Zheng, Z., Xie, S., Dai, H., Chen, X., Wang, H.: An overview of blockchain technology: Architecture, consensus, and future trends. 2017 IEEE International Congress on Big Data (BigData Congress), 557–564 (2017)

[11] Li, X., Jiang, P., Chen, T., Luo, X., Wen, Q.: A survey on the security of blockchain systems. Future Gener. Comput. Syst. **107**, 841–853 (2020)

[12] Li, D., Han, D., Weng, T.-H., Zheng, Z., Li, H., Liu, H., Castiglione, A., Li, K.-C.: Blockchain for federated learning toward secure distributed machine learning systems: a systemic survey. Soft Computing, 1–18 (2021)

[13] Omar, A., Bhuiyan, M.Z.A., Basu, A., Kiyomoto, S., Rahman, M.S.: Privacy-friendly platform for healthcare data in cloud based on blockchain environment. Future Gener. Comput. Syst. **95**, 511–521 (2019)

[14] Tanwar, S., Parekh, K., Evans, R.: Blockchain-based electronic healthcare record system for healthcare 4.0 applications. J. Inf. Secur. Appl.

50 (2020)

[15] Khatoon, A.: A blockchain-based smart contract system for healthcare management. Electronics **9**, 94 (2020)

[16] Dimitrov, D.: Blockchain applications for healthcare data management. Healthcare Informatics Research **25**, 51–56 (2019)

[17] Bhattacharya, P., Tanwar, S., Bodke, U., Tyagi, S., Kumar, N.: Bindaas: Blockchain-based deep-learning as-a-service in healthcare 4.0 applications. IEEE Transactions on Network Science and Engineering, 1–1 (2020)

[18] Sun, Z., Han, D., Li, D., Wang, X., Chang, C.-C., Wu, Z.: A blockchain-based secure storage scheme for medical information. EURASIP Journal on Wireless Communications and Networking **2022**(1), 1–25 (2022)

[19] Bhushan, B., Khamparia, A., Sagayam, K., Sharma, S., Ahad, M., Debnath, N.: Blockchain for smart cities: A review of architectures, integration trends and future research directions. Sustainable Cities and Society **61**, 102360 (2020)

[20] Sharma, P., Park, J.H.: Blockchain based hybrid network architecture for the smart city. Future Gener. Comput. Syst. **86**, 650–655 (2018)

[21] Aujla, G., Singh, M., Bose, A., Kumar, N., Han, G., Buyya, R.: Blocksdn: Blockchain-as-a-service for software defined networking in smart city applications. IEEE Network **34**, 83–91 (2020)

[22] Pieroni, A., Scarpato, N., Nunzio, L., Fallucchi, F., Raso, M.: Smarter city: Smart energy grid based on blockchain technology. International Journal on Advanced Science, Engineering and Information Technology **8**, 298–306 (2018)

[23] Munsing, E., Mather, J., Moura, S.: Blockchains for decentralized optimization of energy resources in microgrid networks. 2017 IEEE Conference on Control Technology and Applications (CCTA), 2164–2171 (2017)

[24] Li, D., Han, D., Zhang, X., Zhang, L.: Panoramic image mosaic technology based on sift algorithm in power monitoring. 2019 6th International Conference on Systems and Informatics (ICSAI), 1329–1333 (2019)

[25] Wang, S., Taha, A.F., Wang, J., Kvaternik, K., Hahn, A.: Energy crowdsourcing and peer-to-peer energy trading in blockchain-enabled smart grids. IEEE Transactions on Systems, Man, and Cybernetics: Systems **49**, 1612–1623 (2019)

[26] Dorri, A., Kanhere, S., Jurdak, R., Gauravaram, P.: Lsb: A lightweight scalable blockchain for iot security and privacy. J. Parallel Distributed Comput. **134**, 180–197 (2019)

[27] Ding, S., Cao, J., Li, C., Fan, K., Li, H.: A novel attribute-based access control scheme using blockchain for iot. IEEE Access **7**, 38431–38441 (2019)

[28] Lao, L., Dai, X., Xiao, B., Guo, S.: G-pbft: A location-based and scalable consensus protocol for iot-blockchain applications. 2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 664–673 (2020)

[29] Liu, H., Han, D., Li, D.: Fabric-iot: A blockchain-based access control system in iot. IEEE Access **8**, 18207–18218 (2020)

[30] Han, D., Zhu, Y., Li, D., Liang, W., Souri, A., Li, K.-C.: A blockchain-based auditable access control system for private data in service-centric iot environments. IEEE Transactions on Industrial Informatics (2021)

[31] Kang, J., Xiong, Z., Niyato, D., Ye, D., Kim, D., Zhao, J.: Toward secure blockchain-enabled internet of vehicles: Optimizing consensus management using reputation and contract theory. IEEE Transactions on Vehicular Technology **68**, 2906–2920 (2019)

[32] Jiang, T., Fang, H., Wang, H.: Blockchain-based internet of vehicles: Distributed network architecture and performance analysis. IEEE Internet of Things Journal **6**, 4640–4649 (2019)

[33] Javaid, U., Aman, M., Sikdar, B.: A scalable protocol for driving trust management in internet of vehicles with blockchain. IEEE Internet of Things Journal **7**, 11815–11829 (2020)

[34] Wang, X., Zeng, P., Patterson, N.J., Jiang, F., Doss, R.: An improved authentication scheme for internet of vehicles based on blockchain technology. IEEE Access **7**, 45061–45072 (2019)

[35] Liu, H., Han, D., Li, D.: Behavior analysis and blockchain based trust management in vanets. J. Parallel Distributed Comput. **151**, 61–69 (2021)

[36] Zhou, Z., Wang, B., Guo, Y., Zhang, Y.: Blockchain and computational intelligence inspired incentive-compatible demand response in internet of electric vehicles. IEEE Transactions on Emerging Topics in Computational Intelligence **3**, 205–216 (2019)

[37] Liu, H., Han, D., Li, D.: Blockchain based trust management in vehicular

networks. In: BlockSys (2020)

[38] Turkanovic, M.: Eductx: A blockchain-based higher education credit platform

[39] Harthy, K.A., Shuhaimi, F.A., Ismaily, K.K.J.A.: The upcoming blockchain adoption in higher-education: requirements and process. 2019 4th MEC International Conference on Big Data and Smart City (ICBDSC), 1–5 (2019)

[40] Oganda, F.P., Lutfiani, N., Aini, Q., Rahardja, U., Faturahman, A.: Blockchain education smart courses of massive online open course using business model canvas. 2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS), 1–6 (2020)

[41] Li, D., Han, D., Zheng, Z., Weng, T.-H., Li, H., Liu, H., Castiglione, A., Li, K.-C.: Moocschain: A blockchain-based secure storage and sharing scheme for moocs learning. Computer Standards & Interfaces (2021)

[42] Wang, S., Ouyang, L., Yuan, Y., Ni, X., Han, X., Wang, F.: Blockchain-enabled smart contracts: Architecture, applications, and future trends. IEEE Transactions on Systems, Man, and Cybernetics: Systems **49**, 2266–2277 (2019)

[43] Li, J., Yao, W., Zhang, Y., Qian, H., Han, J.: Flexible and fine-grained attribute-based data storage in cloud computing. IEEE Transactions on Services Computing **10**, 785–796 (2017)

[44] Chen, L., Moretto, A., Jia, F., Caniato, F., Xiong, Y.: The role of digital transformation to empower supply chain finance: current research status and future research directions (guest editorial). International Journal of Operations & Production Management (2021)

[45] Du, M., Chen, Q., Xiao, J., Yang, H., Ma, X.: Supply chain finance innovation using blockchain. IEEE Transactions on Engineering Management **67**, 1045–1058 (2020)

[46] Wang, Z., Wang, Q., Lai, Y., Liang, C.: Drivers and outcomes of supply chain finance adoption: An empirical investigation in china. International Journal of Production Economics **220**, 107453 (2020)

[47] Pan, A., Xu, L., Li, B., Ling, R.: The impact of supply chain finance on firm cash holdings: Evidence from china. Pacific-Basin Finance Journal **63**, 101402–101402 (2020)

[48] Pellegrino, R., Costantino, N., Tauro, D.: Supply chain finance: A supply chain-oriented perspective to mitigate commodity risk and pricing

volatility. Journal of Purchasing and Supply Management **25**, 118–133 (2019)

[49] Lin, Q., Xiao, Y., Zheng, J.-H.: Selecting the supply chain financing mode under price-sensitive demand: confirmed warehouse financing vs. trade credit. Journal of Industrial and Management Optimization **13**, 0–0 (2017)

[50] Lam, H.K., Zhan, Y.: The impacts of supply chain finance initiatives on firm risk: evidence from service providers listed in the us. International Journal of Operations & Production Management (2021)

[51] Ali, Z., Gong-bing, B., Mehreen, A.: Predicting supply chain effectiveness through supply chain finance: Evidence from small and medium enterprises. The International Journal of Logistics Management **30**, 488–505 (2019)

[52] Garg, C., Kashav, V.: Modeling the supply chain finance (scf) barriers of indian smes using bwm framework. Journal of Business & Industrial Marketing (2021)

[53] Nguema, J.-N.B.B., Bi, G., Akenroye, T.O., Baz, J.E.: The effects of supply chain finance on organizational performance: a moderated and mediated model. Supply Chain Management (2021)

[54] Gelsomino, L., Boer, R.D., Steeman, M., Perego, A.: An optimisation strategy for concurrent supply chain finance schemes. Journal of Purchasing and Supply Management **25**, 185–196 (2019)

[55] Ding, W., Wan, G.: Financing and coordinating the supply chain with a capital-constrained supplier under yield uncertainty. International Journal of Production Economics **230**, 107813 (2020)

[56] Moretto, A., Grassi, L., Caniato, F., Giorgino, M., Ronchi, S.: Supply chain finance: From traditional to supply chain credit rating. Journal of Purchasing and Supply Management **25**, 197–217 (2019)

[57] Su, Y., Lu, N.: Simulation of game model for supply chain finance credit risk based on multi-agent. Open Journal of Social Sciences **03**, 31–36 (2015)

[58] Su, Y., Zhong, B.: The credit risk assessment model of internet supply chain finance: Multi-criteria decision-making model with the principle of variable weight. Journal of Computational Chemistry **05**, 1–11 (2016)

[59] Lu, Q., Gu, J., Huang, J.: Supply chain finance with partial credit guarantee provided by a third-party or a supplier. Comput. Ind. Eng. **135**,

440–455 (2019)

[60] Xiao, Z., Tan, M.: Research on smes credit risk evaluation of supply chain finance based on the third-party b2b platform. DEStech Transactions on Social Science, Education and Human Science (2019)

[61] Liao, H., Wen, Z., Liu, L.: Integrating bwm and aras under hesitant linguistic environment for digital supply chain finance supplier section. Technological and Economic Development of Economy **25**, 1188–1212 (2019)

[62] Song, H., Li, M., Yu, K.: Big data analytics in digital platforms: how do financial service providers customise supply chain finance? International Journal of Operations & Production Management (2021)

[63] Jia, F., Zhang, T., Chen, L.: Sustainable supply chain finance:towards a research agenda. Journal of Cleaner Production **243**, 118680 (2020)

[64] Abdel-Basset, M., Mohamed, R., Sallam, K.M., Elhoseny, M.: A novel decision-making model for sustainable supply chain finance under uncertainty environment. Journal of Cleaner Production **269**, 122324 (2020)

[65] Tseng, M., Lim, M.K., Wu, K.-J.: Improving the benefits and costs on sustainable supply chain finance under uncertainty. International Journal of Production Economics **218**, 308–321 (2019)

[66] Imronudin, I., Hussain, J.: Sustainable supply chain finance process in delivering financing for smes: The case of indonesia. International Journal of Supply Chain Management **9**, 867–878 (2020)

[67] Tseng, M., Wu, K.-J., Hu, J., Wang, C.-H.: Decision-making model for sustainable supply chain finance under uncertainties. International Journal of Production Economics **205**, 30–36 (2018)

[68] Bui, T.-D., Tsai, F.-M., Tseng, M., Tan, R., Yu, K.D., Lim, M.K.: Sustainable supply chain management towards disruption and organizational ambidexterity: A data driven analysis. Sustainable Production and Consumption **26**, 373–410 (2020)

[69] Emtehani, F., Nahavandi, N., Rafiei, F.M.: An operations-finance integrated model with financial constraints for a manufacturer in a multi-supplier multi-product supply chain. Comput. Ind. Eng. **153**, 107102 (2021)

[70] Sang, B.: Application of genetic algorithm and bp neural network in supply chain finance under information sharing. J. Comput. Appl. Math. **384**, 113170 (2021)

[71] Yin, F., Yang, R., Yu, H., Zhou, W., Zhao, Y., Zhang, S.: Edge intelligenceenabled supply chain financial model based on businessto-business ebusiness platforms. Concurrency and Computation: Practice and Experience (2021)

[72] Abbasi, W., Wang, Z., Zhou, Y., Hassan, S.: Research on measurement of supply chain finance credit risk based on internet of things. International Journal of Distributed Sensor Networks **15** (2019)

[73] Liu, Z.: Literature review of supply chain finance based on blockchain perspective. Open Journal of Business and Management **9**, 419–429 (2021)

[74] Tribis, Y., Bouchti, A.E., Bouayad, H.: Supply chain management based on blockchain: A systematic mapping study. (2018)

[75] Liu, J., Yan, L., Wang, D.: A hybrid blockchain model for trusted data of supply chain finance. Wireless Personal Communications, 1–25 (2021)

[76] Li, J., Zhu, S., Zhang, W., Yu, L.: Blockchain-driven supply chain finance solution for small and medium enterprises. Frontiers of Engineering Management **7**, 500–511 (2020)

[77] Shi, X., Yao, S., Luo, S.: Innovative platform operations with the use of technologies in the blockchain era. International Journal of Production Research, 1–19 (2021)

[78] Wang, R., Wu, Y.: Application of blockchain technology in supply chain finance of beibu gulf region. Mathematical Problems in Engineering **2021**, 1–10 (2021)

[79] Chen, J., Cai, T., He, W., Chen, L., Zhao, G., Zou, W., Guo, L.: A blockchain-driven supply chain finance application for auto retail industry. Entropy **22** (2020)

[80] Wang, R., Peng, Y., Wen, S.: Research on the application of blockchain technology in supply chain finance. (2020)

[81] Huang, Y., Liang, Z., Wang, Y., Lu, G.: Research on the application of blockchain in supply chain finance. (2018)

[82] Jiang, C., Ru, C.: Application of blockchain technology in supply chain finance. 2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE), 1342–1345 (2020)

[83] Wang, Y.: The exploration of blockchain technology in supply chain finance. (2021)

[84] Li, Y., Wang, L.: The role of blockchain technology on supply chain finance. In: DSIT (2020)

[85] Yuyan, L., Lan, W.: The role of blockchain technology on supply chain finance. Proceedings of the 3rd International Conference on Data Science and Information Technology (2020)

[86] Yaksick, R.: Overcoming supply chain finance challenges via blockchain technology. (2019)

[87] Dong, L., Qiu, Y., Xu, F.: Blockchain-enabled deep-tier supply chain finance. Operations Management eJournal (2021)

[88] Xu, Z., Gao, D.: Research on application of supply chain finance based on block chain. (2020)

[89] Zhang, J., Wang, F., Pu, Y., Li, P., Ma, Y., Li, Z.: Analysis of the application of blockchain technology in the field of supply chain finance in the natural gas industry. (2021)

[90] Fernández-Caramés, T., Blanco-Novoa, Ó., Froiz-Míguez, I., Fraga-Lamas, P.: Towards an autonomous industry 4.0 warehouse: A uav and blockchain-based system for inventory and traceability applications in big data-driven supply chain management ? Sensors (Basel, Switzerland) **19** (2019)

[91] Kim, H.M., Laskowski, M.: Towards an ontology-driven blockchain design for supply chain provenance. ArXiv **abs/1610.02922** (2018)

[92] Huertas, J., Liu, H., Robinson, S.: Eximchain: Supply chain finance solutions on a secured public, permissioned blockchain hybrid. (2018)

[93] Terzi, S., Zacharaki, A., Nizamis, A., Votis, K., Ioannidis, D., Tzovaras, D., Stamelos, I.: Transforming the supply-chain management and industry logistics with blockchain smart contracts. Proceedings of the 23rd Pan-Hellenic Conference on Informatics (2019)

[94] Liu, H.: A novel supply chain model based on smart contract. Proceedings of the 2020 2nd International Electronics Communication Conference (2020)

[95] Su, S., Wang, K., Kim, H.: Smartsupply: Smart contract based validation for supply chain blockchain. 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), 988–993 (2018)

[96] Ma, C., Kong, X., Lan, Q., Zhou, Z.: The privacy protection mechanism of hyperledger fabric and its application in supply chain finance. Cybersecurity **2**, 1–9 (2019)

[97] Schär, F.: Decentralized finance: On blockchain- and smart contract-based financial markets. (2020)

[98] Reza-Gharehbagh, R., Asian, S., Hafezalkotob, A., Wei, C.: Reframing supply chain finance in an era of reglobalization: On the value of multi-sided crowdfunding platforms. Transportation Research Part E: Logistics and Transportation Review **149**, 102298 (2021)

[99] Fabric-sdk-go. https://github.com/hyperledger/fabric-sdk-go

[100] Fabric-sdk-node. https://github.com/hyperledger/fabric-sdk-node

[101] Fabric-sdk-java. https://github.com/hyperledger/fabric-sdk-java

[102] Fabric-sdk-py. https://github.com/hyperledger/fabric-sdk-py

[103] Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., Caro, A.D., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Smith, K.A., Sorniotti, A., Stathakopoulou, C., Vukolic, M., Cocco, S.W., Yellick, J.: Hyperledger fabric: a distributed operating system for permissioned blockchains. Proceedings of the Thirteenth EuroSys Conference (2018)