Knowledge graph-driven smart contract for metadata checking in blockchain-based BIM collaboration

Xingyu Tao[1], Zhaoji Wu[1], Chengliang Zheng[2], Hao Liu[1], Moumita Das[1], Jack C.P. Cheng[1]*
[1]Department of Civil and Environmental Engineering, The Hong Kong University of Science and Technology, Hong Kong, China
[2]Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China
cejcheng@ust.hk (Jack C.P. Cheng)

Abstract. The increasing digitalization, especially the implementation of building information modelling (BIM) in the construction industry, has inevitably introduced cybersecurity vulnerabilities like data manipulation and denial of access. There is a trend of integrating BIM with blockchain in a design or construction process to secure data integrity and traceability. Metadata data (e.g., ID, version, digital signatures) of BIM models can be stored in a decentralized and unchangeable manner by blockchain. However, one limitation is that the current blockchain cannot detect the correctness of input metadata, contaminating the quality of BIM data in a shared blockchain and thereby hindering collaboration efficiency. Data interaction with blockchain relies on blockchain smart contracts. Therefore, this paper proposes a knowledge graph-driven smart contract (KGSC) for metadata checking in blockchain-based BIM collaboration. Two contributions of this work are: (1) 29 rules are reasoned and generated from the ISO 19650 knowledge graph, and (2) a KGSC algorithm is developed and verified. Results indicate that BIM data processed by the KGSC has higher compliance and accuracy.

1. Introduction

Building information modelling (BIM) gains increasing importance in the construction digitalization transformation. It is a digital foundation for collaborative design in AEC (architecture, engineering, and construction) projects. Interdisciplinary team members are allowed to collaborate in a BIM environment by sharing design attributes, changes, and issues. However, these technological transformations bring new challenges, including cybersecurity. According to Salami Pargoo and Ilbeigi (2023), in North America in 2020, The construction industry was the third target of ransomware attacks after government bodies and the manufacturing sector. Most existing BIM collaborations rely on centralized databases or platforms controlled by third-party vendors or one project administrator, risking security vulnerabilities like data manipulation, denial of access, and traceability loss. The *State of Ransomware Report for Architecture, Engineering, and Construction* indicated that AEC companies were more than twice as likely to be the target of database attacks (e.g., ransomware) than other customers in the sample (Egnyte, 2021). This is because they are very schedule-driven, and delays due to lack of access to their files will significantly impact their costs and damage their reputation. However, comparatively few contractors have created strategies to fully identify and quantify their cyber risks or transfer that specific risk.

Blockchain, a decentralized transaction and data management technology, has attracted mounting interest from both academic and industrial aspects. Each block in a blockchain holds a certain amount of information, and they are connected into a chain in the order of the time they were created. This chain is kept in all the servers, and the whole blockchain is secure as long as one server works in the system. These servers are called nodes/peers in the blockchain system, providing storage space and arithmetic support for the whole blockchain system. To modify the information in the blockchain, one must obtain the consent of more than half of the nodes and modify the data in all the nodes, which are usually in the hands of different subjects, making it extremely difficult to tamper with the information in the blockchain. Compared with traditional networks, blockchain has two core features: one is the difficulty of data tampering, and the other is decentralization. Based on these two features, the information recorded in blockchain is more authentic and reliable, which can help solve the problem of people's mutual distrust. Briefly, in a blockchain network (Figure 1), the consensus mechanisms guarantee data consistency, hash encryptions keep data unchangeable, and distributed ledgers enable every project to have a data copy to avoid unilateral control (Tao et al., 2022).

The AEC industry has investigated its capability to secure BIM data. Owning to the block size limitation, blockchain is not fit for storing large-sized data (e.g., BIM models in giga-byte). Therefore, metadata data of BIM, including the model's ID, version, ownership, and fingerprint, are placed in the blockchain ledger for traceability and integrity guarantees. For example, Tao et al. (2021) proposed a distributed common data environment for secure BIM-based collaborative design. Attributes and digital signatures of BIM data are stored in a blockchain as tamper-proof records. Xue and Lu (2020) captured the semantic differences among BIM models and put these changes in the blockchain. Dounas et al. (2020) introduced a BIM-blockchain integrated workflow to guarantee transparent and consistent collaboration. The Winfield-Rock Report indicated that blockchain shows excellent potential to overcome many BIM security issues by providing change-resistant and hack-resistant design records (Winfield, 2018).

The data interaction between BIM and blockchain is empowered by a smart contract, a piece of code that executes automatically when predefined conditions are triggered. Smart contracts automate building processes by removing intermediary steps (such as manual inspection and processing) and create data guarantees by avoiding fraudulent signatures. Several blockchain smart contracts for BIM collaboration have been developed (McNamara and Sepasgozar, 2021; Ahmadisheykhsarmast and Sonmez, 2020. Nevertheless, these smart contracts can only support basic functions like uploading and querying BIM metadata (Tao et al., 2021) and transferring payment (Erri Pradeep et al., 2021). The current smart contract cannot check the correctness and compliance of input data, resulting in the low quality of records in the blockchain ledger, inevitably leading to design/construction rework and even disputes.

Therefore, this paper aims to propose a new smart contract algorithm to ensure input quality. Two objectives are:

(1) **To generate BIM metadata checking rules (MCR).** The ISO 19650 standards have recommended practices for BIM collaboration, including data quality management solutions. However, these knowledge are scattered, and some are even embedded. Therefore, knowledge graph (KG) technology is leveraged to extract information regarding BIM information exchange. Furthermore, Semantic Web Rule Language (SWRL) is adopted to mine the embedded knowledge and generate MCRs.

(2) **To develop the KGSC algorithm.** A new KGSC algorithm will be developed by integrating MCRs. The algorithm will be validated in a BIM design scenario to test its workability in checking blockchain input correctness.

The remainder of this paper is structured as follows: Section 2 introduces ISO 19650 KG establishment and rules generation. Smart contract algorithm development will be elaborated on in Section 3. Section 4 is the validation, and Section 4 is the conclusion.

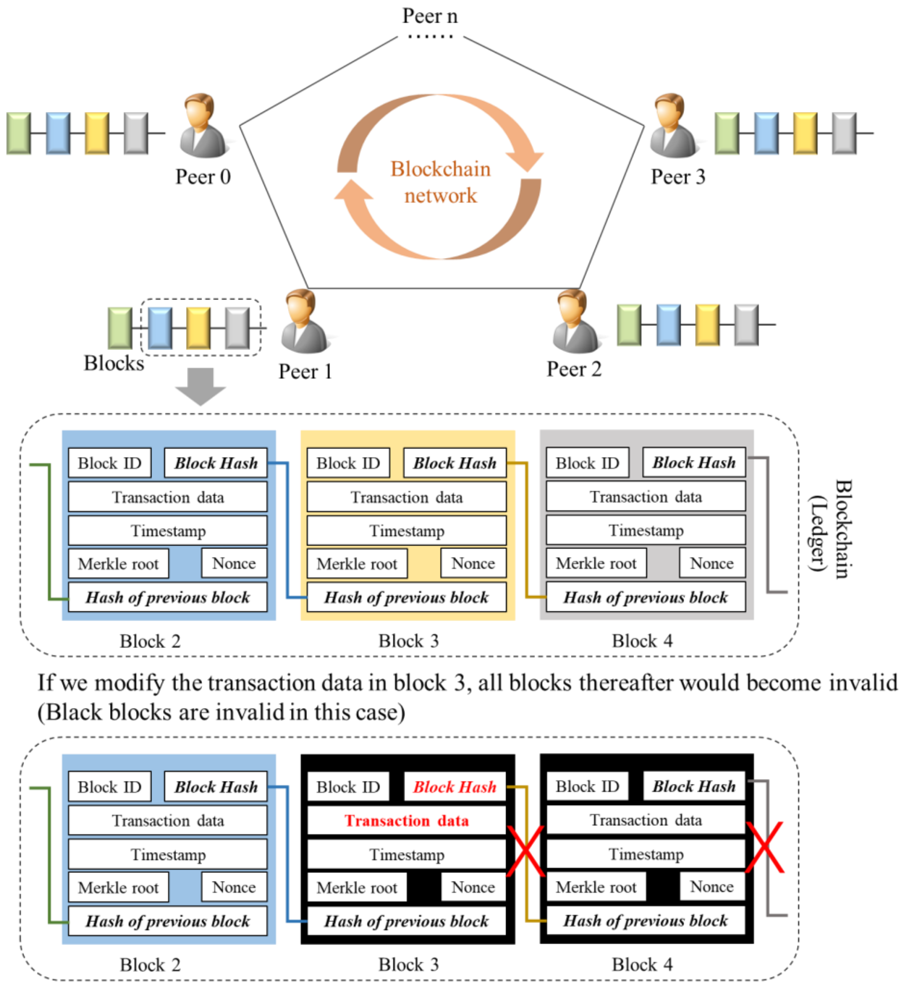


Figure 1. Example of blockchain network

2. MCR generation

2.1 ISO 19650 knowledge graph

ISO 19650 standards is a set of international information management standards that employ building information modeling (BIM) throughout the life cycle of a built asset. This set of standards is the knowledge source for BIM metadata checking for three reasons: (1) ISO 19650 introduces a common data environment (CDE) approach to standardize collaboration workflow. Metadata like data status code, version, author, checker (or reviewer), and naming conventions are all regulated. For example, according to the standards, BIM data in Work in Progress (WIP) should have version numbered using decimals, while models in the SHARED state have integer versions. (2) The ISO 19650 standards offer data exchange requirements covering the BIM lifecycle from design to delivery and operation. Therefore, a wide range of knowledge of data exchange is provided. (3) These requirements in the ISO 19650 standards are generic and can be adopted in various project management scenarios. Implementing ISO 19650 in BIM collaboration is becoming a trend.

However, rules in the standards are not directly available, for the information is unstructured and scattered, and some are even embedded. For example, ISO 19650-1 states the BIM status code (e.g., S0 for WIP and S1 for SHARED), while the reviewer assignment guidelines are introduced in ISO 19650-4. Rules such as who should check models in a specific status code are not indicated. Thus, this paper builds a knowledge graph (KG) to represent knowledge of BIM information exchange during collaboration. To our best knowledge, it is one the very first time to have a KG for ISO 19650. Establishing the KG following instructions in (Fensel et al., 2020), which include three main steps: (1) configuring data sources, (2) extracting knowledge triples.

Configuring data sources. As mentioned, the KG is built on the ISO 19650 series (Figure 2 (a)). ISO 19650-1 covers general information for BIM collaboration, including the CDE workflow, participant responsibility, and project management process. ISO 19650-2 focuses on project information modeling and delivery. Metadata requirements in BIM modeling and delivery are presented in this standard. ISO 19650-3 is specific to the asset management phase. Therefore, BIM data management and metadata requirements for operation are regulated. ISO 19650-4 was published in 2022 and documents information exchange and interaction details in a CDE. Metadata like decision points, federation strategy, and review appointments are listed here. The final standard, ISO 19650-5, introduces the data security protection approaches. Security-related data (e.g., BIM fingerprint) are recommended.

Extracting knowledge triples. A KG is a network that depicts the relationship between entities such as objects, events, situations, or concepts. This data is typically stored in a graph database and represented as a graph structure. The basic unit of a KG is a trip, which consists of a head, relation, and tail. The first step is to define what information should be obtained from the ISO standards. Thus, a blockchain transaction data model is illustrated in Figure 2 (b). Transaction serves as the information carrier between off-chain and on-chain. BIM attributes (or metadata) that should be placed in the blockchain are defined in transactions. The second step is extracting triples and building KG based on the identified attributes. 273 triples, and the ISO 19650 KG are shown in Figure 2 (c).

(a) Knowledge source: ISO 19650 standards

Model sharing transaction
ID
Name
Version
Classification
Type
Timestamp
Role / Domain
Author
Receiver
Reviewer / Checker
Status code
Status name
Decision point
Description
Sensitivity
Fingerprint

(b) Transaction data model

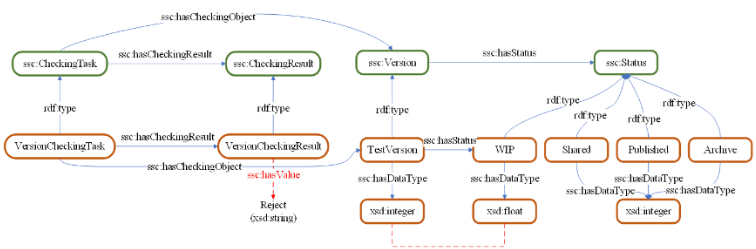
HEAD	RELATIONS	TAIL
information container	should have	persistent identifier
information container	should have	metadata
metadata	is used for	data retrival
infoamtion exchange	ensure	data completeness
infoamtion exchange	comply with	exchange schemas
infoamtion exchange	ensure	concepts continuity
infoamtion exchange	should eliminate	spatial and specification conflicts
WIP	is used for	information development
Information development	is not visible to (other)	task team
check/review/approve transition	is based on	information delivery plan
check/review/approve transition	check	agreed standards
check/review/approve transition	check	methods for producing information
check/review/approve transition	check	procedures for producing information
check/review/approve transition	is performed by	originating task team
shared	support	development of the information model
shared	support	information consultant
shared	support	information coordination
shared	support	information access
shared	generate	issues
issues	is sovled in	WIP
issues	is sovled by	authors
shared	include	client shared
client shared	can be shared with	appointing party
review/authorize transition	check	coordination
review/authorize transition	check	completeness
review/authorize transition	check	accuracy
review/authorize transition	can chenge state to	published
review/authorize transition	can chenge state to	WIP
published	can be used for	detailed design
published	can be used for	asset management
published	can be used for	construction
PIM (at the end of a project)	only contain	published
AIM	only contain	published
archive	hold a journal of	shared
archive	hold a journal of	published
archive	hold a journal of	audit trail of information container development

(c) Example of extracted triples

Figure 2. establishment of ISO 19650 knowledge graph

2.2 ISO 19650 KG reasoning

Some metadata checking rules (MCR) can directly get from the KG, like "the model ID should not be none" or "the model author should be consistent with his/her department". However, it is very likely that some knowledge or rules are hidden and cannot be captured from Figure 2. Knowledge graph reasoning aims to learn new things from old ones. New knowledge can be divided into two types in knowledge graphs: new entities and new relationships. A Semantic Web Rule Language (SWRL) is employed for reasoning. Figure 3 (a) displays an example of an SWRL algorithm developed to deduct a new rule for version checks. As a result, 20 direct rules (in black) and 9 reasoned rules (in red) are generated in (Figure 3 (b)). Given the page limitation, completed SWRL reasoning algorithms are available by contacting the authors.



SWRL reasoning for version compliance checking

ssc:Version(?version)
^ssc:Status(?status)
^ssc:CheckingTask(?checkingtask)
^ssc:CheckingResult(?checkingresult)
^ssc:hasCheckingObject(?checkingtask,?version)
^ssc:hasStatus(?version,?status)
^ssc:hasDataType(?version,?datatypea)
^ssc:hasDataType(?status,?datatypeb)
^differentFrom(?datatypea,?datatypeb)
^ssc:hasCheckingResult(?checkingtask,?checkingresult)
->ssc:hasValue (?checkingresult,"Reject")

Version	Rule - Version should not be none
	Rule - If status name is WIP, version should be float, otherwise, reject
	Rule - If status name is not WIP, version should be integer, otherwise, reject

(a) Example of SWRL reasoning

Transaction attributes	Rules / key knowledge
ID	Rule 1 - Information container should not be none
Name	Rule 2 - Name should not be none
Version	Rule 3 - Version should not be none
	Rule 4 - If status name is WIP, version should be float, otherwise, reject
	Rule 5 - If status name is not WIP, version should be integer, otherwise, reject
Classification	Rule 6 - Classification should not be none
	Rule 7 - Classification can be in ICE, IE, or ICFD (optional)
Type	Rule 8 - Type should no be none (e.g., document, image, or geometrical models)
Time stamp	Rule 9 - Time stamp should not be none
Role / Domain	Rule 10 - Each information container should be linked to a task team
Author	Rule 11 - Author should not be none
	Rule 12 - In ICE, author should be consistent with Role / Domain
Receiver	Rule 13 - Appointing party receives the information model in ICFD
	Rule 14 - If the status name is WIP, receiver should in the same domain with author
	Rule 15 - The receiver should not be none or be all member if the container is sensitive
Reviewer/checker	Rule 16 - If the status name is not match with the checker, an error should be shown
Status code	Rule 17 - Each information container should be assigned a status code (not be none)
	Rule 18 - If the status name is not match with the status code, an error should be shown
Status name	Rule 19 - Not empty
	Rule 20 - Status name outside the mentioned name, should be incorrect
Decision point	Rule 21 - Decision point should not be none
	Rule 22 - In WIP, decision point should be A
	Rule 23 - In shared, decision point should be B
	Rule 24 - Task team leader checks decision point A
	Rule 25 - Lead appointed party checks decision point B
Sensitivity	Rule 26 - Sensitivity should not be none
	Rule 27 - Sensitivity should be tagged as "yes" or "no"
	Rule 28 - Sensitive information should indicate the access range
Fingerprint	Rule 29 - Fingerprint should not be none

(b) Metadata checking rules

Figure 3. ISO 19650 KG reasoning and rules generation

3. Development of KGSC

Figure 4 is the diagram explaining the KGSC algorithm. Seven steps are involved in the algorithm. In the first step, a project member raises a transaction. The second step is metadata correctness checking. The KGSC executes the transaction to see if the input complies with the predefined rules. If inappropriate results appear, the KGSC will reject the transaction; otherwise, the transaction will be executed by endorsing peers (Step 3). The endorsement process detects whether the blockchain can accept the transaction in terms of its format, legality, and excepted results. This execution will not be recorded by blockchain. In Step 4, the verified transaction, signed by the endorsing peers, is sent back to the smart contract, which then forwards the transaction and signature to an ordering service. In Step 5, the ordering service will package transactions during a certain period into a block (Step 6). After receiving a new block, each peer will verify it again before adding it to their local ledger (Step 7). Compared with existing smart contracts that directly put transactions in a blockchain, the developed knowledge graph-driven smart contract (KGSC) integrates MCRs to perform compliance checking. It should be noted that the KGSC is developed based on the Hyperledger Fabric. This open-source and permissioned blockchain platform only allows authorized project members to enroll to avoid disclosing sensitive information.

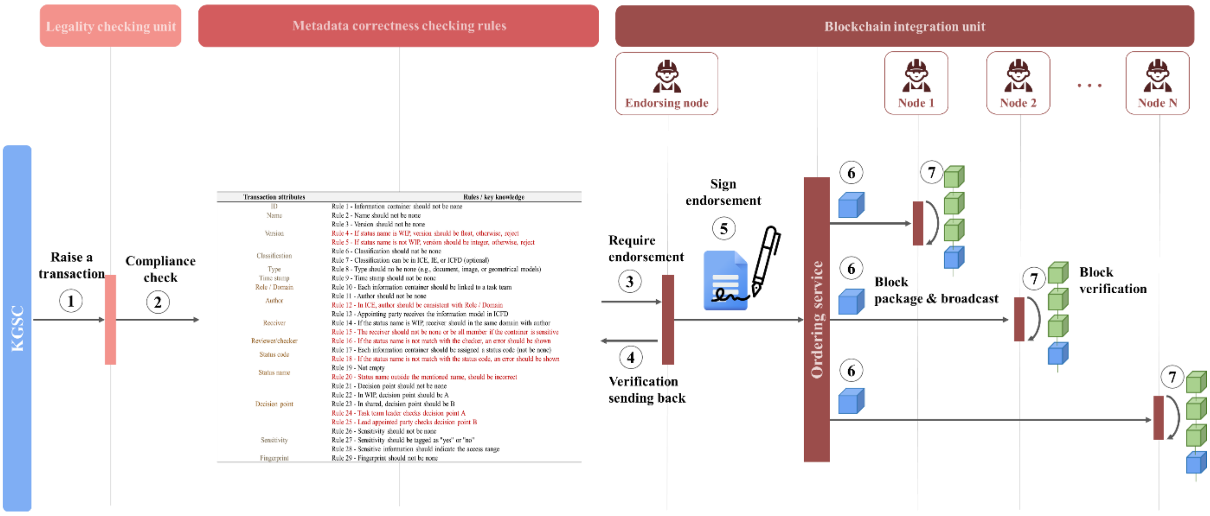
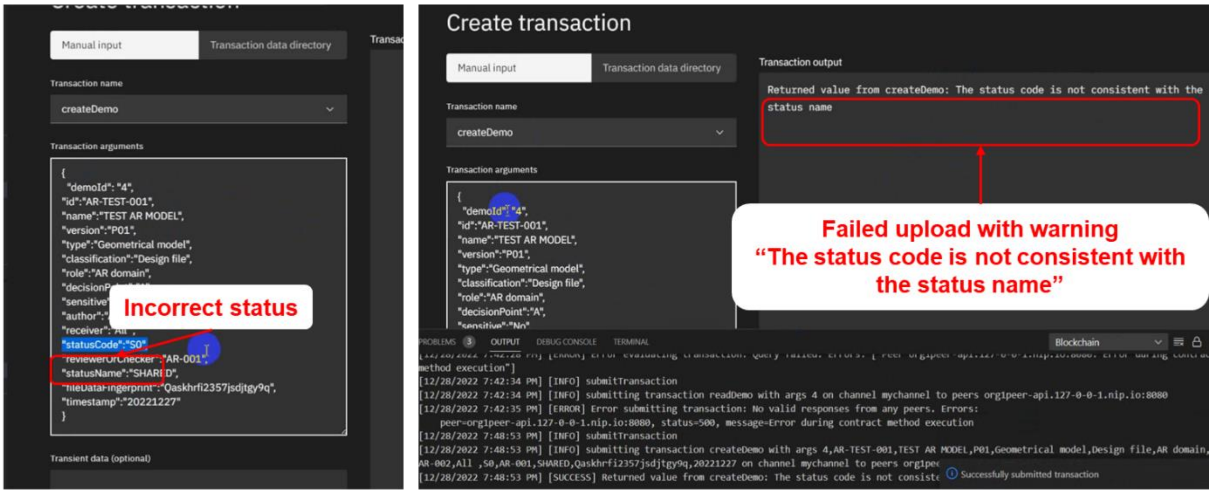


Figure 4. Diagram of KGSC algorithm

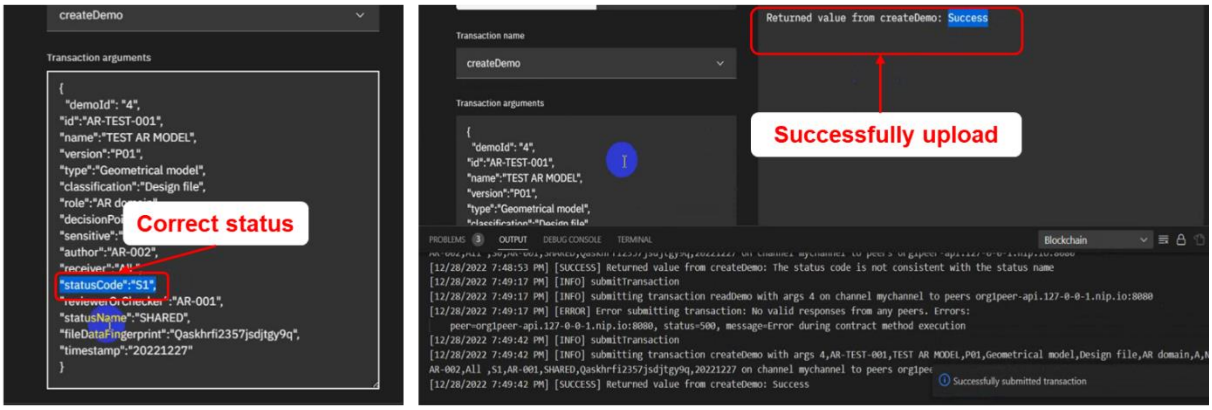
4. Validation and evaluation

This section validates the workability of the KGSC and evaluates its computing performance. The scenario-based approach is adopted in this work. The KGSC is implemented in a BIM design scenario to assess whether it can check out and reject incorrect input to the blockchain. The smart contract is coded using GO language, a statically typed, compiled high-level programming language designed at Google. GO is also recommended by the Hyperledger community for programming smart contracts.

In the BIM design scenario, an architectural designer tends to push a BIM model from WIP to SHARED state for cross-discipline coordination. The metadata of this model is shown in Figure 5. The model ID is "AR-TEST-001" in version P01. The transaction proposer specifies the right values for all attributes except the status code. Since the data are transferred to SHARED, the code should be S1, indicating the BIM model is suitable for coordinating. The transaction in Figure 5 (a) has a wrong code with S0 due to human error. When interacting with the blockchain, the KGSC detected this mistake and rejected this input, avoiding contaminating the blockchain database. A warning message is shown to the user: "The status code is inconsistent with the status name". After revising the transaction to the correct status code, the input is successfully uploaded to the blockchain, safeguarding data quality. As a result, the workability is validated.



(a) A BIM designer tends to share information of a new BIM model in the blockchain. However, one attribute of the BIM, status code, is wrong due to human error. The SSC integrating CCRs has detected this error and declined the request.



(b) The BIM designer correct the error and successfully re-uploaded the transaction

Figure 5. KGSC validation results in BIM-based design scenario

The computing performance of KGSC is assessed. According to Tao et al. (2021), smart contract latency is one representative metric in performance evaluation. Thus, a ten-round latency testing is conducted to see how fast the KGSC processes transactions and whether the latency satisfies requirements in the construction industry, which needs blockchain latency at a millisecond level. The latency is higher than that (i.e., DCDE) in Tao et al. (2021), because the KGSC has encoded MCRs, requiring additional processing time. Results in Figure 6 show that the KGSC still has acceptable computing in 378 milliseconds on average.

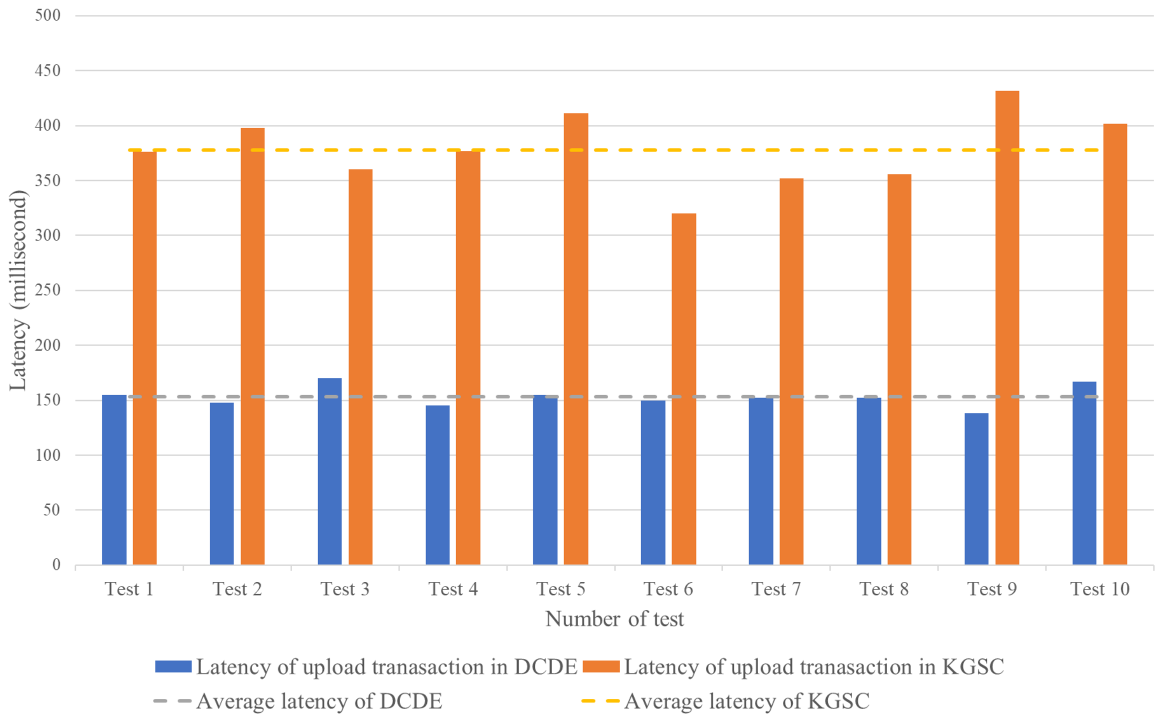


Figure 6. Latency testing and comparison

5. Conclusion

Although blockchain is gaining attention in the construction industry for trustworthy digital collaboration, limited efforts have been devoted to ensuring the correctness of the input data. In existing works regarding BIM-blockchain integration, BIM metadata like ID, fingerprint (or hash signature), and version are stored in a blockchain instead of the whole models due to data size restrictions. Lacking metadata correctness checking inevitably compromises BIM data quality in the blockchain. Therefore, this paper developed a new smart contract algorithm to enhance input quality. Two objectives have been realized. First, an KG is built on top of the ISO 19650 standards, offering information sources for metadata requirements. Furthermore, 29 rules are extracted and reasoned based on the KG using the SWRL method. Second, a KGSC algorithm is developed by integrating rules. The algorithm is validated and evaluated in a BIM scenario. Reasonable results are obtained in terms of feasibility and latency. One limitation is that the rules are restricted to ISO standards, although they provide comprehensive knowledge. The future work is to design a more dynamic rule generation approach that can adapt to different project scenarios with specific checking requirements.

References

DOUNAS, T., LOMBARDI, D. & JABI, W. 2020. Framework for decentralised architectural design BIM and blockchain integration. *International Journal of Architectural Computing*, 19**,** 157-173.

EGNYTE 2021. State of Ransomware Report for Architecture, Engineering and Construction.

ERRI PRADEEP, A. S., YIU, T. W., ZOU, Y. & AMOR, R. 2021. Blockchain-aided information exchange records for design liability control and improved security. *Automation in Construction,* 126**,** 103667.

FENSEL, D., ŞIMŞEK, U., ANGELE, K., HUAMAN, E., KäRLE, E., PANASIUK, O., TOMA, I., UMBRICH, J., WAHLER, A., FENSEL, D. J. K. G. M., TOOLS & CASES, S. U. 2020. How to build a knowledge graph. 11-68.

SALAMI PARGOO, N. & ILBEIGI, M. 2023. A Scoping Review for Cybersecurity in the Construction Industry. *Journal of Management in Engineering,* 39**,** 03122003.

TAO, X., DAS, M., LIU, Y. & CHENG, J. C. P. 2021. Distributed common data environment using blockchain and Interplanetary File System for secure BIM-based collaborative design. *Automation in Construction,* 130**,** 103851.

TAO, X., LIU, Y., WONG, P. K.-Y., CHEN, K., DAS, M. & CHENG, J. C. P. 2022. Confidentiality-minded framework for blockchain-based BIM design collaboration. *Automation in Construction,* 136**,** 104172.

WINFIELD, M. 2018. The winfield rock report: Overcoming the legal and contractual barriers of BIM. UK BIM Alliance.

XUE, F. & LU, W. 2020. A semantic differential transaction approach to minimizing information redundancy for BIM and blockchain integration. *Automation in Construction,* 118**,** 103270.