Addressing BIM versioning challenges: A blockchain-ready IFC schema extension for semantic change management

Lingming Kong[1][1*], Fan Xue[2]

Abstract

A BIM project often encounters many major and minor BIM versions throughout the project's lifecycle. The main reason is expansive and inevitable changes in the conduction of lengthy, complex, and uncertainty-rich construction project. Recently, a semantic differential transaction (SDT) approach was proposed to minimize the data redundancy in the versions for blockchains, by extracting the semantically essential design changes between open BIM versioning. However, the current IFC schema cannot store, verify, or utilize such changes. This paper presents an IFC schema extension of a blockchain-ready change history for BIM versioning. The proposed extension supports both local storage of the changes and remote blockchain verifications. Firstly, the extended IFC entities define a mapping from the SDT model to the up-to-date IFC schema. Then, the new entities are managed and structured by a Java-based compiler-compiler approach; Meanwhile, the extended BIM files are verifiable using the official IFC verification tools. Finally, a pilot study is conducted to validate the technological feasibility of the proposed IFC schema extension.

Keywords: BIM semantics, IFC schema, IFC extension, BIM versioning, Blockchain

1 Introduction

Building Information Modeling (BIM), the digitally constructed virtual model of building assets, has been embraced as a major technology to support collaborative construction activities[1]. However, traditional centralized file-based BIM exchange encounters challenges of inefficiency and lack of trust. The centralized exchange of complicated BIM versions has low efficiency in handling the inevitable changes during the design and construction phases. It not only hinders efficient cross-discipline exchange but is also vulnerable to malicious attacks such as data leakage and spiteful BIM edits. Blockchain technology is an emerging and promising technology to solve this issue. Blockchain is a robust self-sustaining system that

[1] Lingming Kong
Department of Real Estate and Construction, Faculty of Architecture, The University of Hong Kong, Hong Kong
E-mail: u3009509@connect.hku.hk

*: Corresponding author,

[2] Fan Xue
Department of Real Estate and Construction, Faculty of Architecture, The University of Hong Kong, Hong Kong
E-mail: xuef@hku.hk

guarantees data traceability and immutability by integrating peer-to-peer networks and consensus mechanisms[2]. Without a central authority, peers in a blockchain network (i.e., project stakeholders) maintain replicated data to guarantee data integrity. Although some existing studies[3][4][5] evaluate the feasibility and benefits of integrating BIM and blockchain, data redundancy is the unresolved pain point for industry-level applications.

Recently, a Semantic Differential Transaction (SDT) approach was proposed by Xue and Lu to minimize the data redundancy for BIM and blockchain integration[6]. The unchanged part of the tree-like semantic hierarchies is deleted by comparing the IFC history of the incremental BIM model. The differences between the changed parts are then recorded as transactions in the blockchain network. Compared with other BIM changes tracking methods which focus on the byte-level, the SDT approach captures the BIM changes in the semantic level. The meanings, systematic relations, and hierarchies of the changed part is captured and structured thorough the SDT approach. Thus, the bi-directional computation of BIM semantic changes is achieved. However, a BIM represented in the latest IFC schema cannot store or utilize these changes captured by extra computing processes. To improve the ability of IFC to represent semantic changes, this paper presents an extension of the IFC schema that supports both local storage and blockchain verification of BIM versioning.

Industry Foundation Classes (IFC) is a *de facto* standard that describes building assets in a digital way and aims to provide vendor-agnostic domain data sharing. The IFC extensions are widely regarded as a fundamental approach for domain-specific applications. For example, *IfcRoad*, *IfcBridge,* and *IfcRail* are added in IFC 4.3 for presenting infrastructure information[7]. Söbke et al.[8] present an IFC schema extension for describing wastewater treatment plants. Jaly-Zada et al.[9] have extended the IFC schema for BIM version management. They proposed an extension of the existing IFC schema to record evolution in BIM and IFC. However, the relationship-based description of changes in BIM leads to two challenges: (1) the file size explodes for limited version changes and (2) the change information cannot export as partial files to minimize redundant information. Therefore, a new extension for IFC based on the SDT approach is required to support efficient version management in the blockchain-ready BIM. This paper firstly gives an IFC extension method based on the SDT approach that one abstract IFC entity and five sub-entities are developed. Then, an initial validation of the extended entities is conducted. Finally, a simple case is illustrated the feasibility of the proposed IFC extension. This paper specifically focuses on the mapping from the SDT model to the extended IFC schema. The combination of the SDT model and blockchain is beyond the scope of this paper.

2. Versioning as semantic differential modeling for IFC

A BIM naturally and incrementally represents the final state of the current building asset, which is continually updated over the course of a project lifecycle. IFC, a standardized, text-based digital description framework, is used to represent the multiple versions of BIMs.

A Semantic Differential Transaction (SDT) approach has been proposed recently to capture the evolution of BIM[6]. It focuses on semantic changes and enables a bi-directional mapping of BIM onto the blockchain. The kernel of this approach is the comparison between two tree-like hierarchy objects, which are represented in IFC's non-STEP formats such as IFCXML or Afsari et al.'s IFCJSON[10]. Fig. 1 presents the pseudo-code of computing differences between two consecutive models namely ifc_0 and ifc_1, through the SDT computation algorithm. The two input IFC files are firstly processed by the "semantic interoperability" function to formulate tree-like objects without random contents (line 1 and line 2). Afterward, a quick comparison of these two objects on lines3-5 eliminates the unchanged IFC entities from σ_0 and σ_1, resulting in the intersection tree. This facilitates the final step on line 6, which involves computing the differences between the changed tree-like IFC objects (σ_{0c} and σ_{1c}). The bi-directional processes, which add up all the transactions on the base model, i.e., $\sigma_k = \sigma_0 + \sum_{i=1}^{k} \Delta_k$, restore the BIM semantic hierarchy.

```
procedure compute_SDT
input: $ifc_0$, $ifc_1$                              // Two changed IFC at time $t_0$ and $t_1$
#1  $\sigma_0 \leftarrow semantic\_interoperability(ifc_0)$;   // Semantic interoperability layer process
#2  $\sigma_1 \leftarrow semantic\_interoperability(ifc_1)$;
#3  $\sigma^* \leftarrow \sigma_0 \cap \sigma_1$;                  // Calculate the intersection (unchanged) tree
#4  $\sigma_{0c} \leftarrow \sigma_0 - \sigma^*$;                 // To remove unchanged objects
#5  $\sigma_{1c} \leftarrow \sigma_1 - \sigma^*$;
#6  $\Delta_\sigma \leftarrow tree\_diff(\sigma_{0c}, \sigma_{1c})$;     // Compute the difference between changed objects
#7  return $\Delta_\sigma$
```

Fig. 1. Pseudo code of the SDT computational algorithm adopted from [6]

According to the SDT model, BIM models of various versions exhibit differences at the semantic level, which are regarded as semantic changes. These changes can be classified into three types: (1) property changes, such as the name, material, size, etc.; (2) geometric changes, including the location and shape of instances; (3) relationship changes caused by property changes and geometry changes. Each type includes three change actions, namely add, delete, and modify. **Table 1** displays semantic changes examples for each category.

Table 1. List of categories and actions of semantic changes in IFC

Category	Change actions	Example in IFC
Property (quantity)	Add	Add an *IfcProperty* by *Hasproperties* in *IfcPropertySet*
	Delete	Delete an *Ifcproperty* by *Hasproperties* in *IfcPropertySet*
	Modify	Modify *IfcMaterialConstituent* by *IfcMaterial*
Geometric (Shape representation, location)	Add	Add an *IfcRepresentation* in Representations of an *IfcProductRepresentation* (e.g., Representation of *IfcWall*)
	Delete	Delete an *IfcRepresentation* in Representations of an *IfcProductRepresentation*
	Modify	Change an *IfcRepresentation* in representation of an *IfcProductRepresentation*
Relationship	Add	Add a relationship between *IfcPropertySet* and *IfcObject* (e.g. add an *IfcPropertySet* to *IfcWindow*)
	Delete	Delete the relationship between two IFC instances, an IFC object and *IfcProperty*, or *IfcPropertySet* and *IfcProperty*
	Modify	Change an *IfcObject* in RelatedObjects of an *IfcRelDecomposes*

3 IFC schema extension

A solution to enhance the representation of semantic changes in BIM is to map the captured changes into the state-of-the-art IFC schema. Therefore, the extension of the IFC4 schema (version 4.0.2.1) is required, which involves implementing and verifying the extension of the IFC schema.

3.1 Implementing the IFC schema extension

Mapping the SDT model with the current IFC schema should be governed by the following standards: (1) the principle of abstraction, where the extended concepts are highly generalized for specific subjective valued purposes; (2) the scalability theory, where the elements extended in this paper can be further modified in conjunction with specific information classifications, codes, dictionaries or limits; (3) the reusability principle, the expanded notions should be reusable in any data exchange requirements.

To enable assignment to an object or object type, the proposed entity *IfcChangeSet* is derived from *IfcPropertySetDefinition*. It serves as the abstract supertype for all semantic changes associated with objects. These changes include geometry changes, property changes, quantity changes, placement changes, and relationship changes, each represented as separate IFC entities:

1. *IfcGeometryChange* describes the changes in the geometry representation of an object, utilizing attributes such as Version, ChangeType, and reference geometry objects. These attributes provide information about the version of the change, the type of change, and the geometry objects that are affected.
2. *IfcPropertyChange* is used to record change information related to the properties of objects. It includes the property names and their nominal values as part of the recorded information.
3. *IfcQuantityChange* is specifically defined to represent changes in the measurement properties of physical objects. This IFC entity involves three specific attributes: (1) *quantity_Old* represents the previous quantity value in the previous BIM version, (2) *quantity_New* represents the latest quantity value in the current BIM version, and (3) *quantityRef* represents the associated changed IFC entity.
4. *IfcPlacementChange* is utilized to represent changes in the location of instances. It refers to the previous and latest placement to capture the shift in position. Additionally, the calculated transform vector is represented by the *TransVector* attribute, providing further information about the transformation applied.
5. *IfcRelationshipChange* represents the relationship changes amongst objects and properties by referring the *IfcRelationship* entities in two IFC files.

These extended entities are generated by *ifcDoc*[11], a software package that provides a graphical interface for users to modify schema files. In Fig. 2, they are represented by blocks with red wireframes which have grey or dark backgrounds, representing abstract and non-abstract definitions respectively. Fig. 3(a) to Fig. 3(e) illustrates the subtype entities of *IfcChangeSet* in the EXPRESS-G notation.

3.2 Verification of the IFC extension

To verify the extended entities, a Java-based application framework is utilized. It aims to assess whether the extended entities and their corresponding files adhere to the IFC schema, ensuring proper depiction and compliance. The verification processes include two steps. In the first step, the extended entities are generated as Java classes through the compiler-compiler approach. These classes are then merged with the existing IFC schema to create a consolidated representation. Secondly, the syntactic correctness of the generated IFC files is validated by the official tools provided by *IfcDoc*.

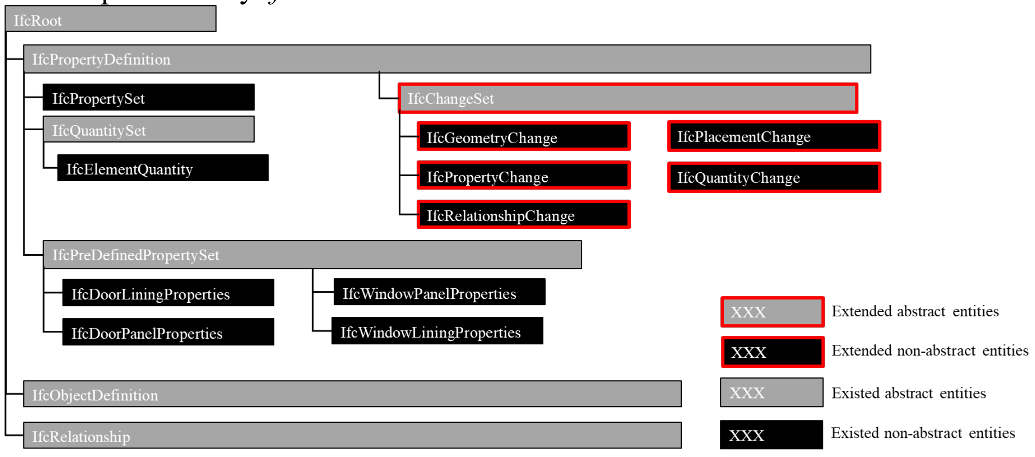


Fig. 2. Extracted IFC schema extension.

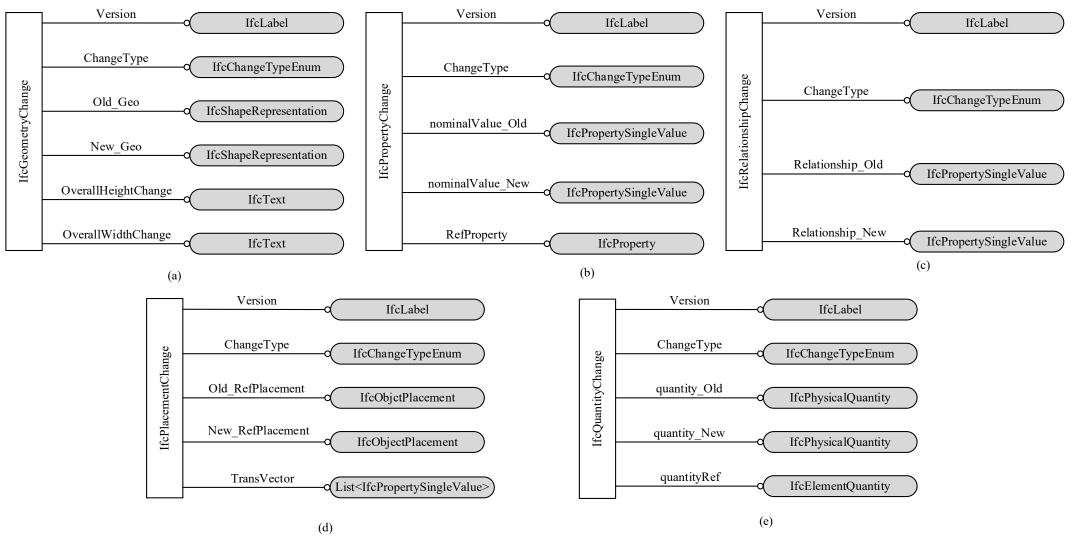


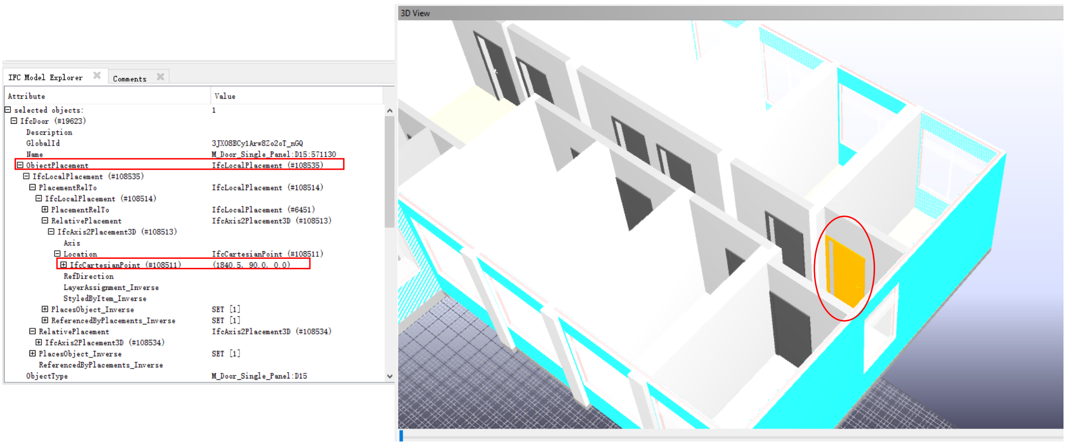
Fig. 3. EXPRESS-G notion of the proposed extended entities

Fig. 4 provides an example of the extracted *IfcPlacementChange* entity in the STEP format, which illustrates a placement movement of a door object. The local placement coordinates of a door instance, identified by the STEP-Id "#19623," have changed from the original position of (1845.0, 90.0, 0.0) to the new position of (1400.0, 90.0, 0.0). This change is represented by the IFC instance *IfcPlacementChange* (with a STEP-Id “#116821”) and linked to the *IfcDoor* (with a STEP-Id “#19623”) as properties. The syntactic correctness of the extracted IFC file in Fig. 4 (a) is validated through *IfcDoc* and the generated validating report is shown in Fig. **5**.

By implementing the proposed IFC extensions, the change model view definition (CMVD) can be defined to extract the partial change model for collaboration. As depicted in Fig. 5(b), a CMVD model is extracted from the whole IFC model, which includes two *IfcPlacementChange* entities. These entities represent the location changes for both the opening element and the door instance separately, which is captured by the SDT approach as semantic hierarchies shown in Fig. 5(a).

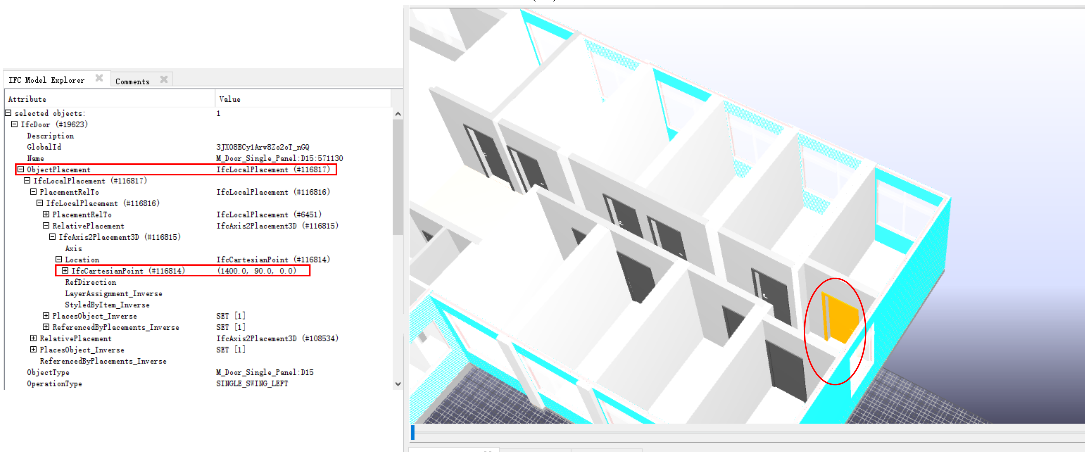
```
ISO-10303-21;
HEADER;
……
ENDSEC;

DATA;
……
#19623=
IFCDOOR('3JXO8BCy1Arw8Zo2oT_nGQ',#42,'M_Door_Single_Panel:D15:571130',$,'M_Door_Single_Panel:D15',#116817,#19616,'571130',2200.0,850.0,.DOOR.,.SING
LE_SWING_LEFT.,$);

……

#108511= IFCCARTESIANPOINT((1840.49999939798,90.0,0.0));
#108513= IFCAXIS2PLACEMENT3D(#108511,$,$);
#108514= IFCLOCALPLACEMENT(#6451,#108513);
#108534= IFCAXIS2PLACEMENT3D(#6,#20,#14);
#108535= IFCLOCALPLACEMENT(#108514,#108534);
……
#116818= IFCPROPERTYSINGLEVALUE('TransVector_X',$,IFCREAL(1750.0),$);
#116819= IFCPROPERTYSINGLEVALUE('TransVector_Y',$,IFCREAL(90.0),$);
#116820= IFCPROPERTYSINGLEVALUE('TransVector_Z',$,IFCREAL(0.0),$);
#116821=
IFCPLACEMENTCHANGE('8B6SzpsdTjK7_6P2OYtfzA',#42,'doorLocationChange',$,'Version_1.0',.MODIFIED.,#108535,#116817,(#116818,#116819,#116820));
#116822= IFCRELDEFINESBYPROPERTIES('as1oxsIrTBaZvASoQZjzzA',#42,$,$,(#19623),#116821);
#116823=
IFCPLACEMENTCHANGE('f6LdycK9TIaPnNYOdyAFsw',#42,'openingLocationChange',$,'Version_1.0',.MODIFIED.,#108514,#116816,(#116818,#116819,#116820));
#116824= IFCRELDEFINESBYPROPERTIES('O0fua2bISSCDKrP51fK2bw',#42,$,$,(#108516),#116823);
```

(a)



```
IFC Model Explorer    Comments
Attribute                                  Value
selected objects:                          1
 IfcDoor (#19623)
   Description
   GlobalId                                3JX08BCy1Arw8Zo2oT_nGQ
   Name                                    M_Door_Single_Panel:D15:571130
  ObjectPlacement                          IfcLocalPlacement (#108535)
   IfcLocalPlacement (#108535)
    PlacementRelTo                         IfcLocalPlacement (#108514)
     IfcLocalPlacement (#108514)
      PlacementRelTo                       IfcLocalPlacement (#6451)
      RelativePlacement                    IfcAxis2Placement3D (#108513)
       IfcAxis2Placement3D (#108513)
         Axis
        Location                           IfcCartesianPoint (#108511)
         IfcCartesianPoint (#108511)       (1840.5, 90.0, 0.0)
         RefDirection
         LayerAssignment_Inverse
         StyledByItem_Inverse
      PlacesObject_Inverse                 SET [1]
      ReferencedByPlacements_Inverse       SET [1]
    RelativePlacement                      IfcAxis2Placement3D (#108534)
     IfcAxis2Placement3D (#108534)
    PlacesObject_Inverse                   SET [1]
     ReferencedByPlacements_Inverse
   ObjectType                              M_Door_Single_Panel:D15
```

(b.1)

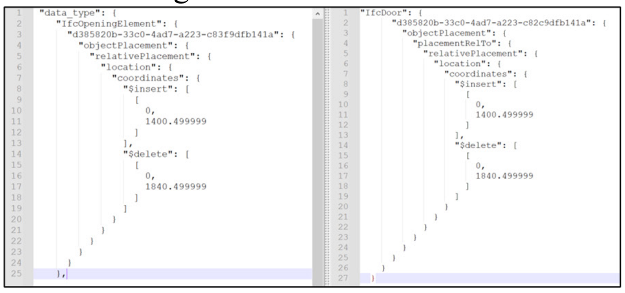


```
IFC Model Explorer    Comments
Attribute                                  Value
selected objects:                          1
 IfcDoor (#19623)
   Description
   GlobalId                                3JX08BCy1Arw8Zo2oT_nGQ
   Name                                    M_Door_Single_Panel:D15:571130
  ObjectPlacement                          IfcLocalPlacement (#116817)
   IfcLocalPlacement (#116817)
    PlacementRelTo                         IfcLocalPlacement (#116816)
     IfcLocalPlacement (#116816)
      PlacementRelTo                       IfcLocalPlacement (#6451)
      RelativePlacement                    IfcAxis2Placement3D (#116815)
       IfcAxis2Placement3D (#116815)
         Axis
        Location                           IfcCartesianPoint (#116814)
         IfcCartesianPoint (#116814)       (1400.0, 90.0, 0.0)
         RefDirection
         LayerAssignment_Inverse
         StyledByItem_Inverse
      PlacesObject_Inverse                 SET [1]
      ReferencedByPlacements_Inverse       SET [1]
    RelativePlacement                      IfcAxis2Placement3D (#108534)
    PlacesObject_Inverse                   SET [1]
     ReferencedByPlacements_Inverse
   ObjectType                              M_Door_Single_Panel:D15
   OperationType                           SINGLE_SWING_LEFT
```

(b.2)

Fig. 4. An example of the *IfcPlacementChange* entity

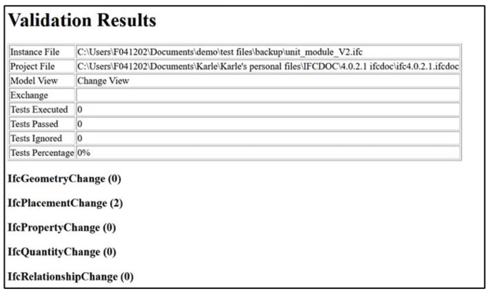
An actual construction project is tested as a pilot case to verify the feasibility of the proposed IFC schema extension. There is total 5 changes for each proposed type are conducted. Due to the limited space, the details of the other four change types and pilot cases will not be illustrated. The number of lines added in the IFC file due to changes and increased file sizes are listed in BIM change **contract**. The incremental semantic changes are stored as semi-formatted data through a BIM change contract (BCC), which automatically uploads these data to the blockchain network. Its structure is illustrated in Fig. 6. The preferred block size of the de-facto opensource blockchain framework Hyperledger-fabric ranges from 512KB to 1MB. Thus, the files size of the increased partial model is on the kilobyte level and reasonable for blockchain storage.



```
"data_type": {
   "IfcOpeningElement": {
     "d385820b-33c0-4ad7-a223-c83f9dfb141a": {
       "objectPlacement": {
         "relativePlacement": {
           "location": {
             "coordinates": {
               "$insert": [
                 [
                   0,
                   1400.499999
                 ]
               ],
               "$delete": [
                 [
                   0,
                   1840.499999
                 ]
               ]
             }
           }
         }
       }
     }
   },
```

```
"IfcDoor": {
         "d385820b-33c0-4ad7-a223-c82c9dfb141a": {
           "objectPlacement": {
             "placementRelTo": {
               "relativePlacement": {
                 "location": {
                   "coordinates": {
                     "$insert": [
                       [
                         0,
                         1400.499999
                       ]
                     ],
                     "$delete": [
                       [
                         0,
                         1840.499999
                       ]
                     ]
                   }
                 }
               }
             }
           }
         }
}
```

(a)



Validation Results

Instance File	C:\Users\F041202\Documents\demo\test files\backup\unit_module_V2.ifc
Project File	C:\Users\F041202\Documents\Karle\Karle's personal files\IFCDOC\4.0.2.1 ifcdoc\ifc4.0.2.1.ifcdoc
Model View	Change View
Exchange	
Tests Executed	0
Tests Passed	0
Tests Ignored	0
Tests Percentage	0%

IfcGeometryChange (0)

IfcPlacementChange (2)

IfcPropertyChange (0)

IfcQuantityChange (0)

IfcRelationshipChange (0)

(b)

Fig. 5. The example of (a) SDT results of the door placement change; (b) the validation results of the added IFC entities.

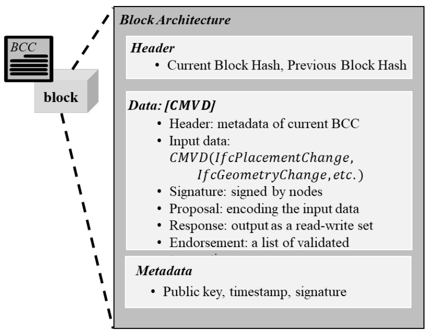

BCC
block
Block Architecture
Header
• Current Block Hash, Previous Block Hash
Data: [CMVD]
• Header: metadata of current BCC
• Input data: CMVD(IfcPlacementChange, IfcGeometryChange, etc.)
• Signature: signed by nodes
• Proposal: encoding the input data
• Response: output as a read-write set
• Endorsement: a list of validated
Metadata
• Public key, timestamp, signature

Fig. 6. Structure of a BIM change contract

Table 2. test results

Change type	Added lines	Increased STEP file size (kb)
Geometry change	226	24
Property change	5	1
Quantity change	5	1
Placement change	7	1
Relationship change	10	1

4. Conclusion

Multiple BIM versions are generated by the incremental and iterative design changes, and are expected to be captured and stored on trustworthy media, such as blockchain. The SDT approach proposed by Xue and Lu[6] minimizes the redundancy of semantic change data for blockchain-ready BIM integration. However, the current IFC schema cannot store, verify, or utilize these changes.

To overcome the addressed limitation, this paper proposes an IFC extension for the semantic changes on top of the SDT model. In this extension, the SDT results are mapped to the IFC schema by utilizing the standard *IfcChangeSet* and its five extended IFC subtype entities. This approach allows for effective incorporation of the semantic differential model's outcomes into the existing IFC schema. To gauge the feasibility of the proposed IFC schema extension, we implemented a Java IFC platform and tested the validation on the official IFC certification program. Finally, a pilot case was studied to validate the feasibility of the proposed extension. As a result, all the semantic changes in the pilot case were captured by the SDT approach and represented with the extended IFC schema, where IFC files can be structured and validated in accordance with the IFC schema syntax. However, there are still some limitations in this research which requires further exploration:

- First, the proposed IFC extension represents different categories of semantic changes separately. However, the five semantic change types interact with each other. For instance, changes in properties, quantities, locations, and geometries always along with relationship changes. And geometric changes also result in the change of quantities. The interactive relationship between those semantic changes should be considered in the future research.
- Besides, the pilot case is tested in a laboratory setting. A more complex case study should be done to explore the feasibility or limitations of the proposed extension methods in terms of the industrial project level.
- Finally, the efficiency of computing the IFC-represented semantic changes and uploading to blockchain network is not discussed, which requires further research.

Acknowledgement

The work presented in this paper was supported by the Hong Kong Research Grants Council (No. 17200221) and in part by the Hong Kong Innovation and Technology Fund (No. ITP/029/20LP).

References

[1] Lou, J., Lu, W., & Xue, F. (2021). A review of BIM data exchange method in BIM collaboration. Proceeding of the 25th International Symposium on Advancement of Construction Management and Real Estate. Singapore: Springer, Singapore.

[2] Scott, D. J., Broyd, T., & Ma, L. (2021). Exploratory literature review of blockchain in the construction industry. *Automation in Construction*, 103914. doi:https://doi.org/10.1016/j.autcon.2021.103914

[3] Celik, Y., Petri, I., & Rezgui, Y. (2023). Integrating BIM and Blockchain across construction lifecycle and supply chains. *Computers in Industry, 148*, 103886. doi:https://doi.org/10.1016/j.compind.2023.103886

[4] Saah, A., & Choi, J.-h. (2023). Blockchain technology in the AEC industry: Scientometric analysis of research activities. *Journal of Building Engineering, 72*, 106609. doi:https://doi.org/10.1016/j.jobe.2023.106609.

[5] Zhao, R., Chen, Z., & Xue, F. (2023). A blockchain 3.0 paradigm for digital twins in construction project management. *Automation in Construction, 145*, 104645. doi:https://doi.org/10.1016/j.autcon.2022.104645

[6] Xue, F., & Lu, W. (2020). A semantic differential transaction approach to minimizing information redundancy for BIM and blockchain integration. *Automation in Construction, 118*, 103270. doi:https://doi.org/10.1016/j.autcon.2020.103270

[7] buildingSMART. (2023, 5 15). *buildingSMART domains*. Retrieved from https://www.buildingsmart.org/standards/domains/

[8] Söbke, H., Peralta , P., Smarsly, K., & Armbruster, M. (2021). An IFC schema extension for BIM-based description of wastewater treatment plants. *Authomation in Construction, 129*, 103777. doi:https://doi.org/10.1016/j.autcon.2021.103777

[9] Zada, A., Tizani, W., & Oti, A. (2014). Building information modelling (BIM)—Versioning for collaborative design. *Computing in Civil and Building Engineering*, 512-519.

[10] Afsari, K., Eastman, C. M., & Castro-Lacouture, D. (2017). JavaScript Object Notation (JSON) data serialization for IFC schema in web-based BIM data exchange. Automation in Construction, 77, 24-51. doi:https://doi.org/10.1016/j.autcon.2017.01.011

[11] buildingSMART. (2023, 5 15). IfcDoc User Guide. Retrieved from Github: https://github.com/buildingsmart-private/IfcDoc/wiki/IfcDoc-User-Guide