



INTEROPERABILITY ANALYSIS OF IFC-BASED DATA EXCHANGE BETWEEN HETEROGENEOUS BIM SOFTWARE

Huahui LAI, Xueyuan DENG* 

Department of Civil Engineering, School of Naval Architecture, Ocean and Civil Engineering, Shanghai Jiao Tong University, Shanghai, China

Received 03 June 2018; accepted 24 August 2018

Abstract. Traditionally, the one-to-one interaction between heterogeneous software has become the most commonly used method for multi-disciplinary collaboration in building projects, resulting in numerous data interfaces, different data formats, and inefficient collaboration. As the prevalence of Building Information Modeling (BIM) increases in building projects, it is expected that the exchange of Industry Foundation Classes (IFC)-based data can smoothly take place between heterogeneous BIM software. However, interoperability issues frequently occur during bidirectional data exchanges using IFC. Hence, a data interoperability experiment, including architectural, structural and MEP models from a practical project, was conducted to analyze these issues in the process of data import and re-export between heterogeneous software. According to the results, the fundamental causes of interoperability issues can be concluded as follows: (a) software tools cannot well interpret several objects belonging to other disciplines due to the difference in domain knowledge; (b) software tools have diverse methods to represent the same geometry, properties and relations, leading to inconsistent model data. Furthermore, this paper presents a suggested method for improving the existing bidirectional data sharing and exchange: BIM software tools export models using IFC format, and these IFC models are imported into a common IFC-based BIM platform for data interoperability.

Keywords: Building Information Modeling (BIM), Industry Foundation Classes (IFC), data interoperability, interoperability issues, multiple disciplines, BIM software.

Introduction

In the architecture, engineering, construction and facility management (AEC/FM) industry, each discipline has its own domain knowledge. Users from various disciplines select diverse software tools for their own business tasks and then collaborate with other users for interoperability. Interoperability in practical projects involves many aspects. Chen *et al.* (2008) classified interoperability concerns into four areas: data, services, processes, and business. The data interoperability refers to the data sharing and exchange between heterogeneous BIM software, and it provides a base to facilitate interoperability of other concerns. However, the issues of data interoperability are still common in the AEC/FM industry.

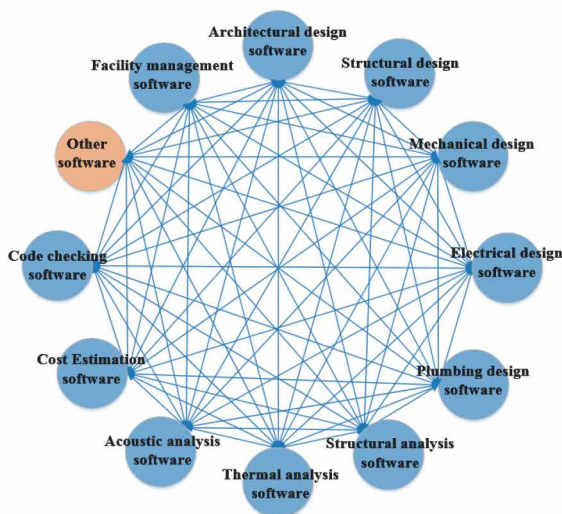
Traditional data interoperability between diverse software tools is limited to a point-to-point direct conversion based on proprietary data formats, resulting in various “Information Islands” throughout the building lifecycle, as shown in Figure 1(a). As building projects become more

complicated, the traditional method hardly meets the increasing needs of data sharing and exchange. For this purpose, BIM technology was introduced based on the concept of creating, storing and managing a great deal of information throughout the building lifecycle in an integrated way (Eastman *et al.* 2011). BIM software tools from different disciplines are widely applied to building projects (Liao, Teo 2017). Due to multiple disciplines, data sharing and exchange between diverse software tools become an inevitable need, and a public and rich data format is necessary for data interoperability (Tolman 1999; Ramaji, Memari 2018). Consequently, IFC schema was developed to support data sharing and exchange. According to a non-exhaustive survey, over 200 software tools have import or export capabilities of IFC data models (BuildingSMART 2013). After two decades of development, IFC has become a de facto standard for data interoperability between heterogeneous software tools (Olawumi *et al.* 2017). As

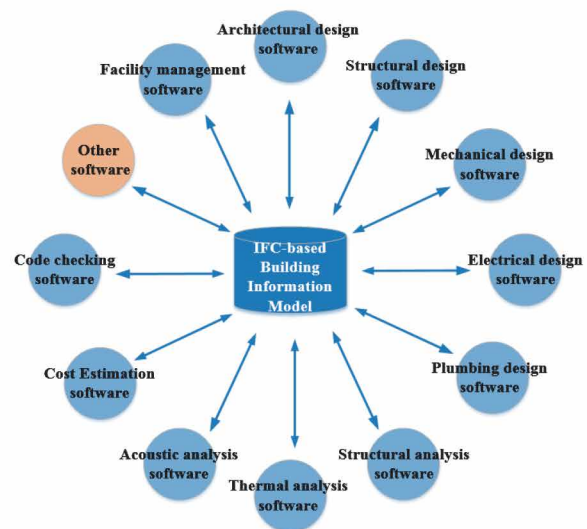
*Corresponding author. E-mail: dengxy@sjtu.edu.cn

depicted in Figure 1(b), IFC schema acts as a medium for bidirectional data sharing and exchange between heterogeneous software. Sometimes software tools require data models from other disciplines to fulfil their own business tasks, and it is inevitable that software tools import the required models created by other software. Subsequently, these imported models may be re-exported as new IFC models for other business tasks. For example, a structural engineer imports the architectural model for structural design and analysis, and then exports a new model to the architect for collaborative design. However, when the IFC data format is used for BIM interoperability in practical projects, interoperability issues (Kam *et al.* 2002; Oh *et al.* 2015; Taher 2016), such as data loss and misrepresentation, commonly arise. It results in a misunderstanding that IFC schema cannot be a data standard for BIM interoperability.

Data interoperability is a process of data import and export between heterogeneous software tools. During this process, a software tool exports a created model as IFC format for data sharing and exchange, and the other one imports this IFC model for business tasks. This process includes two mappings (Ma *et al.* 2006; Lee 2011): (1) one mapping from the internal schemas of software tools to IFC models for export, and (2) the other from IFC models to internal schemas for import. Interoperability issues are likely to arise if either of mappings is wrong. For efficient interoperability, most existing studies suggest the improvement of IFC schema and the development of IFC data interfaces of software tools. These efforts are useful for data interoperability, but do not address interoperability issues fundamentally. In this paper, a data interoperability experiment between software tools was carried out, and then the causes of interoperability issues were analyzed at the IFC data level. Finally, a suggested method for data interoperability was discussed.



(a) Traditional method



(b) BIM-based method

Figure 1. Data interoperability between various software tools

1. Literature review

1.1. Official IFC certification

BuildingSMART adopts IFC 2x3 Coordination View V2.0 as a benchmark template to validate the IFC import and export capabilities of software tools. To date, 30 software tools have been certified for IFC import, 23 tools for IFC export, and only 14 tools passed both import and export certification (BuildingSMART 2018). Numerous software tools still need to pass the official certification for the effective support of IFC models. Furthermore, the IFC export certification can be divided into three aspects: architecture, structure, and mechanical, electrical & plumbing (MEP) (BuildingSMART 2010).

The official IFC certification process includes two steps (Lipman *et al.* 2011). Step 1 includes a range of object-level models, such as beams, columns, slabs, and walls. Due to the building complexity, it is impossible for software tools to test all objects, so the most common objects are selected to test in the certification process. In Step 2, two or three project models, composed of most objects in Step 1, are used for a further certification. The certification process is largely limited to “standard objects” (Kiviniemi 2008). If complex objects are exchanged in IFC format, software tools may not guarantee lossless data exchange, even if these software tools are IFC certified. In other words, the certification process is “more of a test of the ability to exchange information via IFCs rather than the quality of the exchange” (Lipman *et al.* 2011).

1.2. Related studies on data interoperability

Data sharing and exchange is one of the key factors of collaboration throughout the building lifecycle. Various efforts from both academia and industry have been devoted to addressing current problems in data interoperability between heterogeneous software tools.

Table 1. A non-exhaustive list of studies related to data interoperability

	Test tool	Test model	Test criteria	Analysis method	Test result
Pazar and Turk (2008)	Architectural Desktop 2005, AllPlan Architecture 2005 and ArchiCAD 9	Different walls; four architectural models	Wall: file size, total entities, GUID, direction, shape; Architectural model: total entities, entity type	Visual inspection and manual comparison	Wall: differences in physical file size, GUID, number of objects, shape representation, etc. Architectural model: missing GUID, geometric misrepresentation, missing or new material, etc.
Jeong <i>et al.</i> (2009)	ArchiCAD 10.0, Bentley Architecture 8, Digital Project v1 R3, Revit Building 9.1, Tekla Structures 13 and StructureWorks Precast	Some structural elements, including precast concrete, steel and cast-in-place reinforced concrete members	Geometry, location, shape, entity type, total entities, property set	Visual inspection using textual and graphic viewers	Loss of objects, inconsistent object types, geometric misrepresentation, loss of parametric rules and properties, etc.
Choi and Kim (2011)	Revit Architecture, ArchiCAD, Digital Project and Solibri Model Checker	Architectural objects: wall, window, door, slab, and space	Geometry, property set	No mention	Differences in name and type value, missing property set, etc.
Golabchi and Kamat (2013)	Bentley Microstation V8i and Revit 2012	Some architectural elements, such as walls, beams, columns, and doors	Geometry, property	Visual inspection	Loss of properties, missing some elements, etc.
Nizam and Zhang (2015)	Revit and Tekla	A room with four walls, two slabs, four columns, two windows, and a door; A building with slabs, walls, stairs, railings, furniture, openings, and doors	File size, IFC instances, object types, attribute values, schema version	Visual inspection and automatic analysis using file analyzer	Inconsistent object types, differing numbers of instances, missing or new values, loss of numerical precision, reference number differences, etc.
Sibenik (2016)	Allplan 2016, ArchiCAD 19, Revit 2016, SCIA Engineer 15.3 and RFEM 5.05	Linear elements: columns; Planar elements: slabs, roofs, and walls; Connectivity; Foundations	Geometry, element property	Visual inspection	Different geometry interpretation, inconsistent object types, missing property, etc.
Muller <i>et al.</i> (2017)	Revit, TQS and IFC Model Viewer	A structural model with beams, slabs, columns, stairs, and ramps	Material/type, object placement, GUID, geometry	Visual inspection and evaluation using scores	Geometric misrepresentation, loss of material and load, new GUIDs, etc.

Table 1 summarizes several studies on data interoperability based on IFC schema. According to different types of test criteria, visual inspection and some specific analysis tools (e.g., IFC file analyzer) were used when analyzing interoperability issues. Furthermore, these test results demonstrate that data loss and misrepresentation is common during data sharing and exchange using IFC format, such as missing objects, geometric misrepresentation, and inconsistent object types. It is noted that these studies mainly focus on architectural and/or structural models. To track the differences between IFC files during data sharing and exchange, several tools for quantitative analysis were developed, such as EXPRESS Evaluation System (EVASYS)

(Ma *et al.* 2006), Compare P21 (Lee *et al.* 2011), and IFC-diff (Shi *et al.* 2018). These efforts provided a reference to analyze the causes of interoperability issues.

Some researchers (Sanguinetti *et al.* 2012; Oh *et al.* 2015) point out that the cause of such issues is the incomplete mapping between software native models and IFC models. Hence, the quality improvement of IFC interfaces in software tools is recommended. Additionally, other researchers attempted to study how to improve data interoperability in specific domains. Karan and Irizarry (2015) extended the interoperability across the geospatial and BIM domains by using semantic web technology and ontologies of construction operations. Ramaji and

Memari (2016) developed an interpreted information exchange method to interpret the structural analytical model from the architectural model. Kim and Yu (2016) proposed a segmentation process to divide curved walls in IFC models into segmented straight walls for building energy analysis. Hu *et al.* (2017) utilized several solutions, such as the logic chain and a transformation algorithm from BIM to Geographic Information System (GIS), to promote interoperability between MEP-based documents and intelligent MEP models.

Although plenty of research work has been conducted to improve data interoperability between heterogeneous software tools, interoperability issues do exist in the AEC/FM industry – at least it is not as accurate as users expected. To summarize, some problems that need to be solved are listed as follows:

1. The AEC/FM industry involves interdisciplinary data exchanges between heterogeneous software tools. Most test models in previous studies were limited to common architectural and/or structural objects, and there were seldom test models from other disciplines, for example, MEP models.
2. Most studies mainly presented the errors and misrepresentation in the interoperability tests, but few of them analyzed in detail the causes of poor interoperability from the views of software and IFC schema.
3. Some studies concluded with several recommendations for improved work, such as the improvement of IFC schema and software data interface. It requires significant effort and broad support from domain experts and software developers. These recommendations are beneficial to improve data interoperability to some extent, but current interoperability issues are yet to be solved fundamentally.
4. Although some methodologies (such as the ontology) have been developed to improve data sharing and exchange in specific domains, interoperability issues between heterogeneous software tools are yet to

be solved, which still exist in other domains. Hence, an in-depth analysis of data interoperability in the AEC/FM industry is of theoretical and applied value.

To solve these problems, an experiment of importing and exporting three types of models (including architectural, structural and MEP models) was conducted to look for interoperability issues. The causes of these issues would be further analyzed at the IFC data level. In addition, a suggested IFC-based method was proposed for improving the current process of data sharing and exchange.

2. Method of data interoperability experiment

2.1. Data interoperability experiment

In project collaboration, building information is required to share and exchange between various disciplines, such as architecture, structure, and MEP. To date, a large number of software tools have been developed for diverse domains in the AEC/FM industry. According to a survey (Kiviniemi *et al.* 2008), ArchiCAD, Tekla Structures, and MagiCAD are popular software tools in the architectural, structural, and MEP domains, respectively. Further, ArchiCAD has passed the IFC certification process in the architectural domain, Tekla Structures is IFC-certified for the structural domain, and MagiCAD for MEP (BuildingSMART 2018). Therefore, these three BIM software tools were selected to conduct the interoperability experiment between multiple disciplines. This experiment based on limited software tools cannot capture all interoperability issues, but major interoperability issues can be displayed because these tools are prevalent in practice. Moreover, an IFC Platform, which was developed by the authors, was also tested in this experiment in comparison with other software tools. The proposed IFC Platform is a prototype that aims to interpret and manage IFC data models. The process of this interoperability experiment is illustrated in Figure 2. As depicted, these four test tools were named Software A to

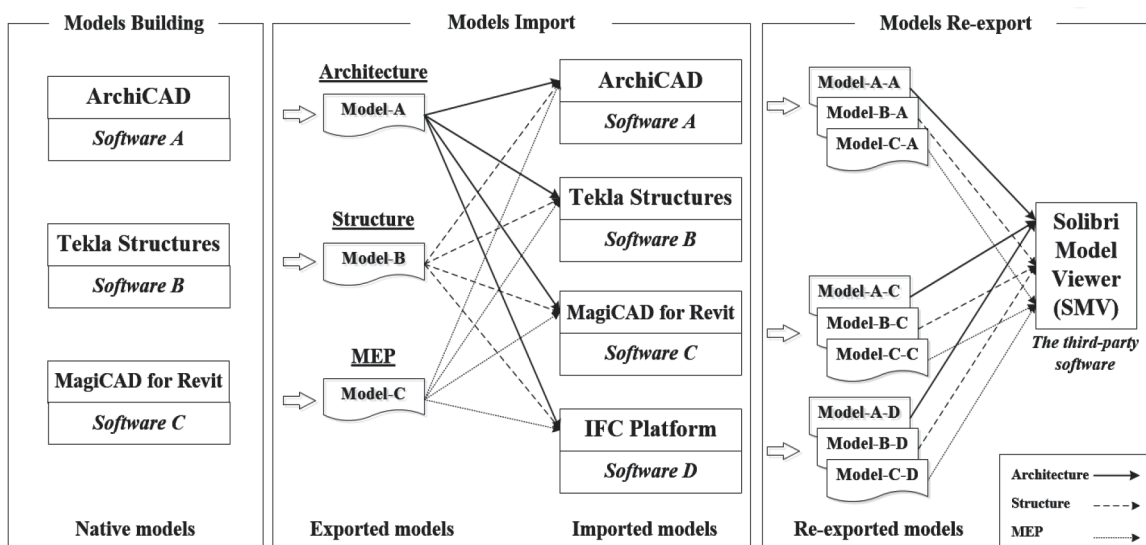


Figure 2. Data interoperability experiment for IFC data exchange between BIM software

Software D, respectively, and three types of data models (that is, architectural, structural and MEP models) were used to analyze data sharing and exchange between aforementioned software tools:



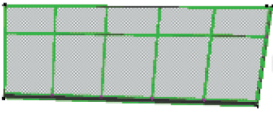
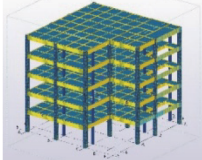
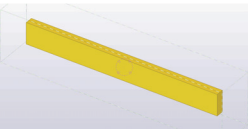
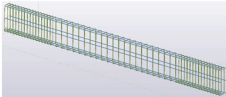
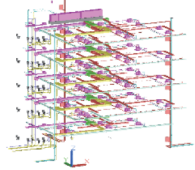

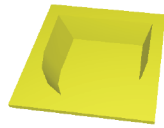
1. Models Building: Domain-specific software tools (Software A, B, and C) were used to build models of three disciplines to ensure accurate representation, and then exported IFC models within default software’s settings for IFC export. These IFC models were named Model-A, Model-B, and Model-C, respectively.
2. Models Import: These exported IFC models were imported into four software tools for the import test, including Software A, B, C, and D. Different kinds of objects from three models were selected to verify the interpretation of IFC models in these software tools.
3. Models Re-export: When importing original IFC files, these software tools converted IFC models to internal models, and then re-exported IFC files without modifications. These re-exported IFC files were imported into a third-party software tool (Solibri Model Viewer (SMV) in this example) for the export test. These models were renamed by adding the software code in the last part of the original model name. For example, Model-A re-exported from Software B was renamed Model-A-B. It is noteworthy that (a) IFC files were imported into Software B as reference models that could not be re-exported, so the re-export test in Software B was not conducted here; (b) in Software C, IFC models were re-exported through the data interface of Revit.

2.2. Test models and criteria

In this experiment, a 5-story office building with an area of about 2,400 m² was selected as the test case. It is a small project, but includes required objects and related attributes. To reflect data sharing and exchange in practice, several typical objects from three disciplines were selected: (a) the wall and window for architecture, (b) the beam and reinforcing bars for structure, and (c) the pipeline and terminal for MEP (see Table 2). Contrary to the IFC certification process that tests “standard objects”, test objects used in this experiment include some common simple-shaped objects (e.g., the wall, beam, and pipeline), but also cover those with complex geometries, such as the window, reinforcing bars, and flow terminal. These selected objects are representative in corresponding domains, and other types of objects can be tested in the similar manner. As mentioned above, three models (architecture, structure, and MEP) of this building were built by ArchiCAD 20, Tekla Structures 2017 and MagiCAD 2017, respectively. The models created by these three tools are displayed in Table 2.

Differences between IFC models were firstly discussed at the overall level, such as file sizes and numbers of objects. Different kinds of attributes contained in objects can be mapped into relevant IFC object types and their geometry, properties, and relations (Eastman *et al.* 2011). Hence, these feature information of objects was chosen as the test criteria, including the type, geometry, property, and relation. In addition, the colour of objects is important for the visual representation of building projects, so the colour

Table 2. Test models for the interoperability experiment

Test model	Test object	
 <p data-bbox="354 1506 485 1532">Architecture</p>	 <p data-bbox="693 1506 823 1532">Object 1: Wall</p>	 <p data-bbox="1047 1506 1224 1532">Object 2: Window</p>
 <p data-bbox="361 1723 469 1749">Structure</p>	 <p data-bbox="685 1723 839 1749">Object 3: Beam</p>	 <p data-bbox="1008 1723 1270 1749">Object 4: Reinforcing bars</p>
 <p data-bbox="385 1947 446 1972">MEP</p>	 <p data-bbox="669 1947 854 1972">Object 5: Pipeline</p>	 <p data-bbox="1039 1947 1224 1972">Object 6: Terminal</p>

was taken as one of test criteria in this experiment. The visual inspection method was used to view the imported IFC models in four test tools. All re-exported models were imported in SMV for viewing, and the IFC File Analyzer (National Institute of Standards and Technology 2011) was used to automatically analyze the overall information of these models. Further, the causes of interoperability issues would be manually analyzed at the IFC data level.

3. Results and analysis of data interoperability experiment

The overall results of imported and re-exported models are presented in Table 3 and Table 4, respectively. In terms of test criteria, the results would be further analyzed by the following groups: overall information of models, and types, geometry, colours, properties, and relations of objects.

3.1. Overall information of models

Through statistical analysis by using the IFC File Analyzer, information of all elements in re-exported IFC models are provided in Table 5. (1) For the re-exported architectural models, Model-A-A and Model-A-D were almost the same as Model-A, but the file size of Model-A-C increased 115.9%. Additional 68,330 IFC entities and 1,776 property sets were added in Model-A-C; (2) When Model-B were re-exported from Software A and Software C, there were significant differences in file sizes, IFC entities, and property sets in Model-B-A and Model-B-C. In addition, there was no *IfcBeam* or *IfcReinforcingBar* entity in Model-B-C, which had 333 new *IfcBuildingElementProxy* entities. No significant difference was found in Model-B-D. The results of re-exported MEP models were similar to those of re-exported structural models.

Table 5 revealed that even if no modification was made to original models, the file sizes of re-exported models from Software A and Software C could increase. The file size of Model-B-A, for example, dramatically increased 19,388.4%. The cause is that domain-specific software tools have domain knowledge to represent the information in their own disciplines, but may not explicitly interpret specific information from other disciplines. For instance, domain-specific software tools create 3D geometric shapes using parametric modeling methods, but other tools may use nonparametric modeling methods due to the lack of domain knowledge. Taking the geometric representation of reinforcing bars as an example, Model-B had numerous reinforcing bars represented by *IfcSweptDiskSolid*, a parametric shape representation. Nevertheless, neither Model-B-A nor Model-B-C had *IfcSweptDiskSolid* entities, as depicted in Table 5. It indicates that neither Software A nor Software C supports this method for shape representation. Instead of using *IfcSweptDiskSolid*, *IfcFacetedBrep* with numerous *IfcFace* entities was used in Model-B-A and Model-B-C. As a result, Model-B-A and Model-B-C had 2,371,856 and 388,112 *IfcFace* entities, respectively.

Although the method with *IfcFace* could be used to represent the geometric shape of reinforcing bars, it lost the feature geometric parameters (more details on the geometric shape of reinforcing bars were discussed in Subsection 3.3), but also required much more IFC entities than the method with *IfcSweptDiskSolid*, resulting in large file sizes. On the other hand, the file sizes of all re-exported models from Software D changed a little. As mentioned above, Software D was developed based on IFC schema. IFC models can be interpreted by software D without any conversion to other data schemas, remaining their original data structures. Only the header section and instance numbers of IFC files may be updated according to default settings.

3.2. Object types

As IFC schema provides the GUID to identify objects, test objects can be queried through their GUIDs. The types of test objects in original and re-exported models are documented in Table 6. The issues related to IFC-based object types could be concluded: (1) In Model-B-C, entities for the beam (*IfcBeam*) and reinforcing bars (*IfcReinforcingBar*) from Model-B were transformed to *IfcBuildingElementProxy*; (2) A terminal (*IfcFlowTerminal*) was lost in Model-C-A; (3) The object type of a terminal in Model-C-C changed from *IfcFlowTerminal* to *IfcBuildingElementProxy*.

In the first issue, the reinforcing bar was selected as an example for analysis. The reinforcing bar in Model-B was represented as *IfcReinforcingBar* entity, and *IfcBuildingElementProxy* entity was used in Model-B-C. The *IfcBuildingElementProxy* entity is generally used to express objects undefined in IFC schema. Software C may lack the definition or not support the interpretation of reinforcing bars, resulting in the utilization of *IfcBuildingElementProxy* entity. Although *IfcBuildingElementProxy* entity is able to express objects (Lee *et al.* 2011), this entity is not for domain-specific objects, and it is difficult for software tools to recognize and use. In the second issue, most objects of Model-C-A could be displayed, except for the terminal, as shown in Table 4. The reason for such issue is that as an architectural tool, Software A lacks the required domain knowledge to define specific information in MEP domain. According to the IFC import report, Software A didn't support the geometric representation of this terminal, resulting in its missing. The cause of the third issue was similar to the first one.

3.3. Geometry of objects

Table 3 illustrates geometric representations of Model-A, Model-B, and Model-C imported in test tools. Through visual inspection, geometric representations of Model-A and Model-B in four tools were the same as the original models. Geometric representations of Model-C in Software B, C, and D were accurate, but the terminal was missing in Software A. On the other hand, geometric representations of re-exported models in SMV are present-

Table 3. Three models imported into test software tools

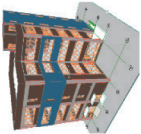

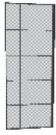
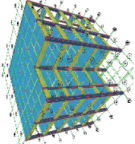

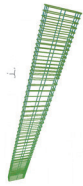
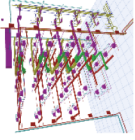

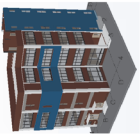


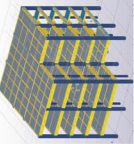


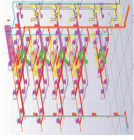


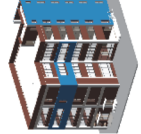


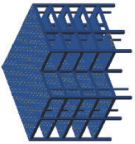


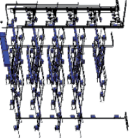


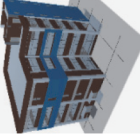


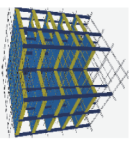


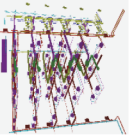


Software	Architecture			Structure			MEP		
	Model-A <i>IfcWallStandardCase</i>	Object 1 <i>IfcWindow</i>	Object 2 <i>IfcWindow</i>	Model-B	Object 3 <i>IfcBeam</i>	Object 4 <i>IfcReinforcingBar</i>	Model-C	Object 5 <i>IfcFlowFitting</i>	Object 6 <i>IfcFlowTerminal</i>
1 A									-
2 B									
3 C									
4 D									

Table 4. Three models re-exported from test software tools (in SMV)

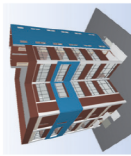


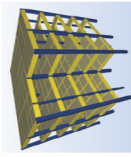

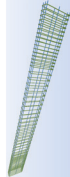
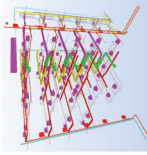

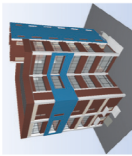


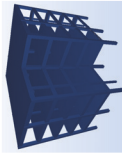


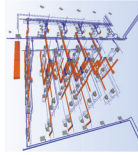

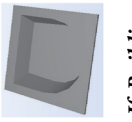
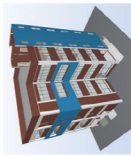


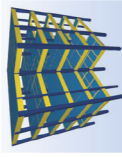

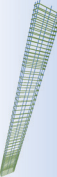
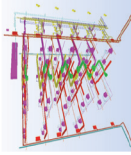


Software	Architecture			Structure			MEP		
	Model-A	Object 1	Object 2	Model-B	Object 3	Object 4	Model-C	Object 5	Object 6
1 A		 Ifc Wall Standard Case	 Ifc Window	 Model-B-A	 Ifc Beam	 Ifc Reinforcing Bar	 Model-C-A	 Ifc Flow Fitting	-
2 C		 Ifc Wall Standard Case	 Ifc Window	 Model-B-C	 Ifc Building Element Proxy	 Ifc Building Element Proxy	 Model-C-C	 Ifc Flow Fitting	 Ifc Building Element Proxy
3 D		 Ifc Wall Standard Case	 Ifc Window	 Model-B-D	 Ifc Beam	 Ifc Reinforcing Bar	 Model-C-D	 Ifc Flow Fitting	 Ifc Flow Terminal

Table 5. Re-exported models from Software A, C, and D

	Test item	Original model	Re-exported model			
			A	C	D	
			1	2	3	4
Architecture		Model-A	Model-A-A	Model-A-C	Model-A-D	
	1	File size / KB	3,789	3,742	8,181	4,650
	2	IFC entities	76,380	(-1.2%) 77,243	(+115.9%) 144,710	(+22.7%) 76,380
	3	Objects	393	393	393	393
	4	Property sets	974	974	2,750	974
	5	<i>IfcBuildingElementProxy</i>	10	10	10	10
	6	<i>IfcWallStandardCase</i>	150	150	150	150
	7	<i>IfcWindow</i>	83	83	83	83
	8	<i>IfcFace</i>	12,624	12,712	14,612	12,624
9	<i>IfcRelConnectsPathElements</i>	167	167	127	167	
Structure		Model-B	Model-B-A	Model-B-C	Model-B-D	
	10	File size / KB	2,523	491,683	75,662	2,651
	11	IFC entities	41,893	(+19,388.4%) 9,261,555	(+2,898.9%) 1,456,998	(+5.1%) 41,892
	12	Objects	843	843	429	843
	13	Property sets	374	2,578	2,929	374
	14	<i>IfcBuildingElementProxy</i>	0	0	333	0
	15	<i>IfcBeam</i>	194	194	0	194
	16	<i>IfcReinforcingBar</i>	549	549	0	549
	17	<i>IfcSweptDiskSolid</i>	1,447	0	0	1,447
18	<i>IfcFace</i>	0	2,371,856	388,112	0	
19	<i>IfcRelAggregates</i>	297	297	3	297	
MEP		Model-C	Model-C-A	Model-C-C	Model-C-D	
	20	File size / KB	12,224	20,790	91,811	13,237
	21	IFC entities	179,775	(+70.1%) 342,471	(+651.1%) 1,481,821	(+8.3%) 179,775
	22	Objects	3,860	3,825	3,860	3,860
	23	Property sets	4,739	8,529	33,553	4,739
	24	<i>IfcBuildingElementProxy</i>	370	370	685	370
	25	<i>IfcFlowFitting</i>	1,336	1,336	1,336	1,336
	26	<i>IfcFlowTerminal</i>	337	302	71	337
	27	<i>IfcFace</i>	10,775	30,237	199,206	10,775
28	<i>IfcRelAssignsToGroup</i>	11	11	0	11	

Note: In this paper, significant differences were marked as bold and underlined text.

ed in Table 4. Two types of misinterpretations could be found: (1) Misrepresentation: In Model-B-C, only a part of reinforcing bars were displayed. (2) Missing: A terminal was lost in Model-C-A.

The IFC entities used for the geometry of test objects are detailed in Table 6. Although IFC object entities generated from diverse software tools are consistent, their geometric representations may differ from the original. For example, the IFC object entity for the reinforcing bars in Model-B-A was *IfcReinforcingBar*, the same as the original one, but the geometry represented by *IfcSweptDiskSolid* was converted to the representation with *IfcFacetedBrep* in Model-B-A. Both *IfcSweptDiskSolid* and *IfcFacetedBrep* can be used to express the geometry of reinforcing bars, but the method using *IfcFacetedBrep* is more complex. Through analyzing Model-B and Model-B-A, IFC entities for geometric representation of the reinforcing bars are illustrated in Figure 3. In Model-B, one *IfcSweptDiskSolid* entity had 13 *IfcCompositeCurveSegment* entities, each of

which referred to one *IfcTrimmedCurve* entity. Nevertheless, each *IfcFacetedBrep* entity in Model-B-A included one *IfcClosedShell* with 294 *IfcFace* entities. As seen in Table 5, no *IfcFace* entity was found in Model-B, and Model-B-A generated 2,371,856 new *IfcFace* entities. This resulted in a distinct difference in the file size between Model-B and Model-B-A. In Model-B-C, the geometric representation of the reinforcing bars was also represented by *IfcFacetedBrep*, but only 388,112 *IfcFace* entities were newly generated, as shown in Table 5. This was why only a part of reinforcing bars were displayed in Model-B-C.

For the loss of the terminal in Software A, according to the import report, some polygons represented by *IfcFace* in this terminal were not perpendicular to the normal vector, and this representation method was not supported by Software A. Due to the misinterpretation during the import process, this terminal was not re-exported from Software A, resulting in its loss in Model-C-A.

Table 6. IFC object entities and their geometric representations in re-exported models

		Original model	Re-exported model		
		Model-A	Model-A-A	Model-A-C	Model-A-D
Object 1 (Wall)	Type	<i>IfcWallStandardCase</i>	<i>IfcWallStandardCase</i>	<i>IfcWallStandardCase</i>	<i>IfcWallStandardCase</i>
	Geometry	<i>IfcExtrudedAreaSolid</i>	<i>IfcExtrudedAreaSolid</i>	<i>IfcExtrudedAreaSolid</i>	<i>IfcExtrudedAreaSolid</i>
Object 2 (Window)	Type	<i>IfcWindow</i>	<i>IfcWindow</i>	<i>IfcWindow</i>	<i>IfcWindow</i>
	Geometry	<i>IfcFacetedBrep</i>	<i>IfcFacetedBrep</i>	<i>IfcExtrudedAreaSolid</i>	<i>IfcFacetedBrep</i>
		Model-B	Model-B-A	Model-B-C	Model-B-D
Object 3 (Beam)	Type	<i>IfcBeam</i>	<i>IfcBeam</i>	<i>IfcBuildingElementProxy</i>	<i>IfcBeam</i>
	Geometry	<i>IfcExtrudedAreaSolid</i>	<i>IfcExtrudedAreaSolid</i>	<i>IfcExtrudedAreaSolid</i>	<i>IfcExtrudedAreaSolid</i>
Object 4 (Reinforcing bar)	Type	<i>IfcReinforcingBar</i>	<i>IfcReinforcingBar</i>	<i>IfcBuildingElementProxy</i>	<i>IfcReinforcingBar</i>
	Geometry	<i>IfcSweptDiskSolid</i>	<i>IfcFacetedBrep</i>	<i>IfcFacetedBrep</i>	<i>IfcSweptDiskSolid</i>
		Model-C	Model-C-A	Model-C-C	Model-C-D
Object 5 (Pipeline)	Type	<i>IfcFlowFitting</i>	<i>IfcFlowFitting</i>	<i>IfcFlowFitting</i>	<i>IfcFlowFitting</i>
	Geometry	<i>IfcShellBasedSurfaceModel</i>	<i>IfcShellBasedSurfaceModel</i>	<i>IfcFaceBasedSurfaceModel</i>	<i>IfcShellBasedSurfaceModel</i>
Object 6 (Terminal)	Type	<i>IfcFlowTerminal</i>	-	<i>IfcBuildingElementProxy</i>	<i>IfcFlowTerminal</i>
	Geometry	<i>IfcShellBasedSurfaceModel</i>	-	<i>IfcFaceBasedSurfaceModel</i>	<i>IfcShellBasedSurfaceModel</i>

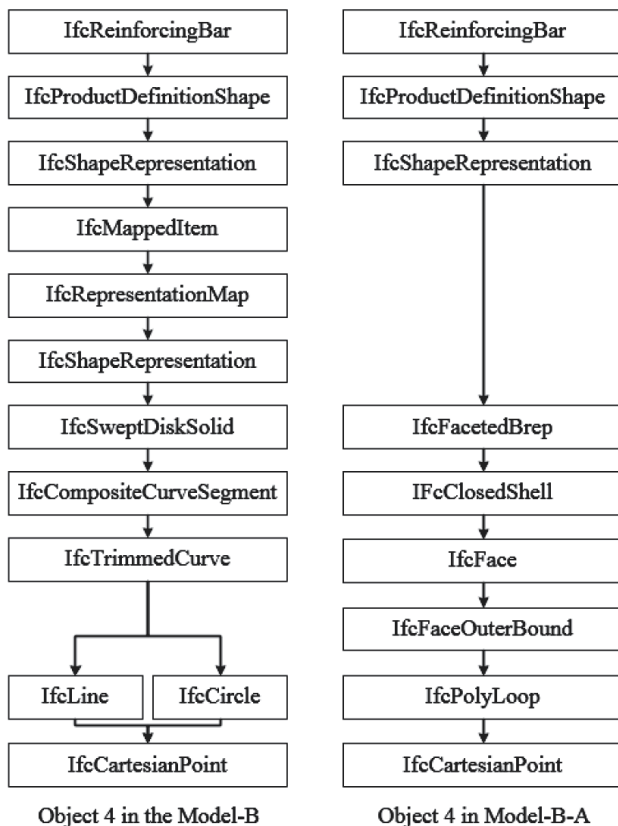


Figure 3. Geometric representation of the reinforcing bars in different IFC models

3.4. Colours of objects

When building test models, modelers defined diverse material data for different objects, and the default colours in reference to material data were retained. In Table 3, all

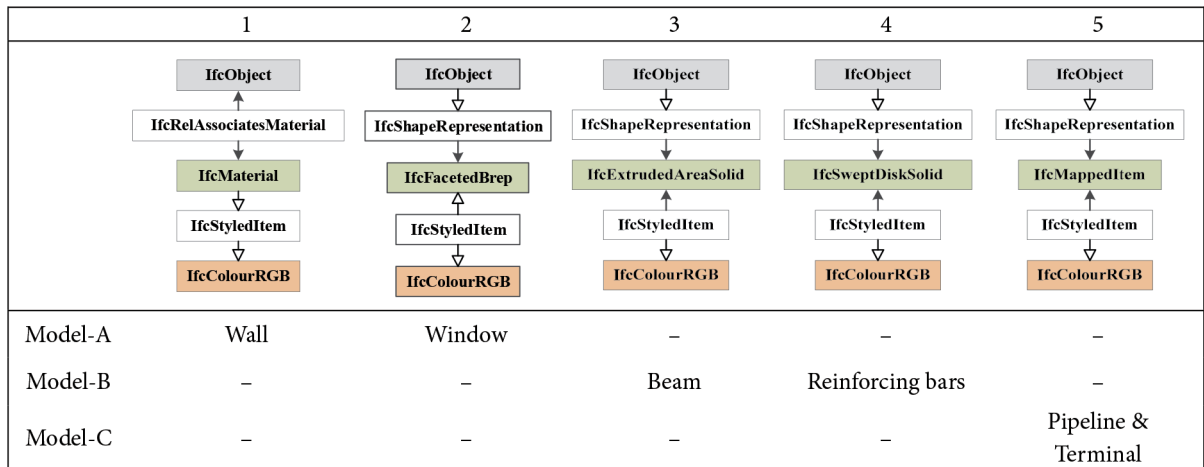
test models and their test objects imported into Software B and Software D were displayed with correct colours; except for the missing terminal, colours of other objects in test models were accurate in Software A. When Model-B was imported into Software C, the yellow-colored beam turned to blue, and the reinforcing bars became yellow from green; all objects from Model-C changed to blue in software C.

The re-exported models were imported into SMV for the visual check, as shown in Table 4. The associated results were: (1) Objects from three re-exported architectural models had the same colours as Model-A. (2) The colours of the beam and reinforcing bars in Model-B-A were accurate, but the slabs changed from blue-green to yellow. Within Model-B-C, the green-colored reinforcing bars became yellow, and blue for other objects. (3) Except for the missing terminal, other objects in Model-C-A had correct colours. Significant differences in colours were found in Model-C-C. For example, the pipeline became orange from red, and the terminal changed to gray from yellow.

IFC schema provides multiple ways to represent colours. The colour representation methods of test objects are summarized in Table 7. Software C could correctly interpret the colours of the wall and window from Model-A, but display incorrect colours of objects from Model-B and Model-C. It indicates that Software C may not support the third method to the fifth method in Table 7.

In IFC schema, colour information is generally defined by (R, G, B) values in *IfcColourRgb* entity. There are various methods to define the relations between objects and colours. Examples include *IfcMaterial* and *IfcMappedItem*, that is, the first and fifth methods in Table 7, respectively. Figure 4 illustrates the relations between objects and colours using these two methods. It demonstrates that the

Table 7. IFC-based colour representation methods



Note: “A→B” means that A refers to B directly; “A↔B” means that at least one referencing element exists between A and B, e.g., “A→C→B”:

same colour information can be represented by a variety of methods in IFC schema. As each software tool is able to support some of these methods, rather than all ones, the software tool may not interpret colour information that is represented through other methods. Multiple methods for the same information significantly add the complexity of interpretation for software tools.

3.5. Properties of objects

To analyze the interoperability issue in the property, different types of properties were manually added in test objects, as listed in Table 8.

In import test, (1) all properties of the wall and window in Software A, C, and D were consistent with those in Model-A, but the glazing area fraction of the window

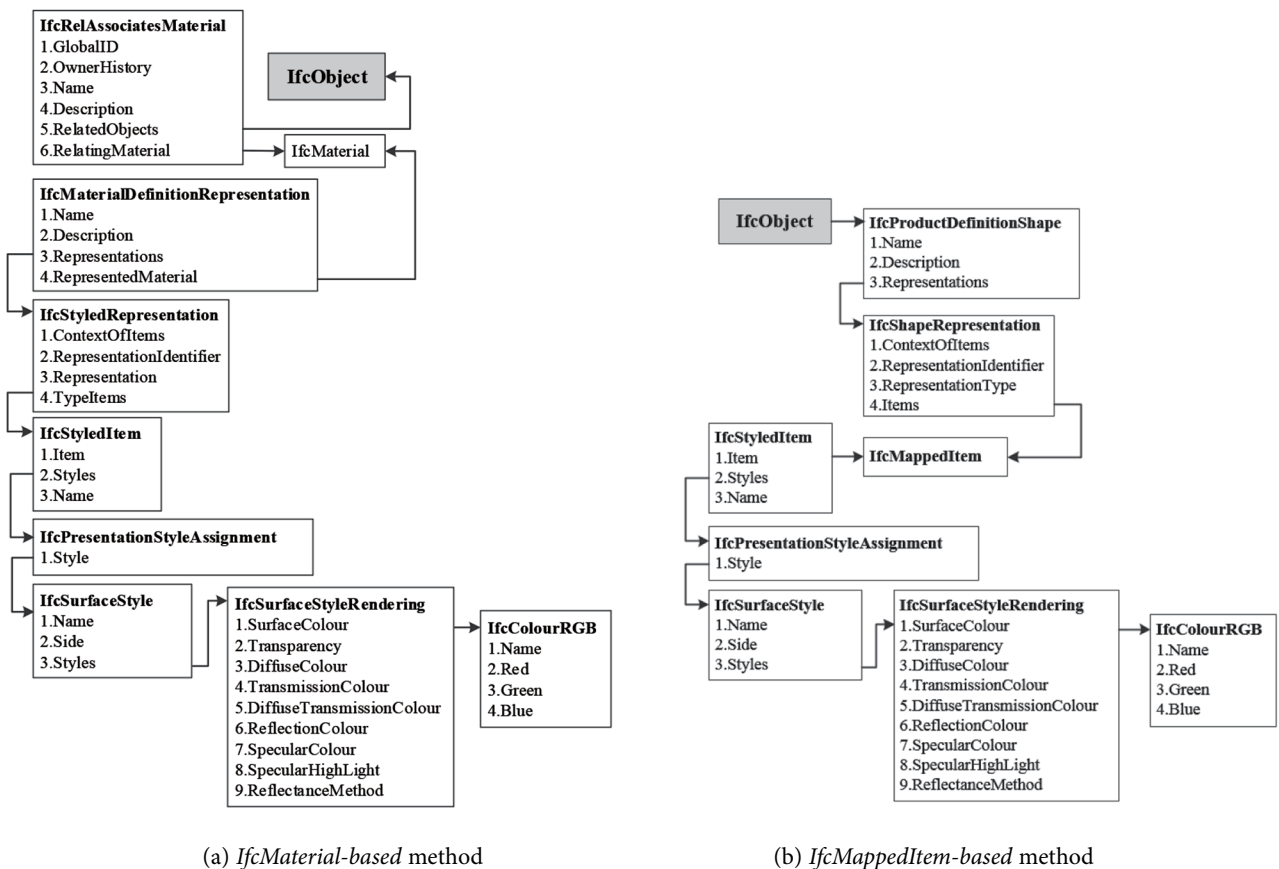


Figure 4. Examples of different methods for colour representation using IFC schema

Table 8. Properties of test objects imported into and re-exported from test software tools

Object	Property	Original value	A		B		C		D	
			Import	Re-export	Import	Re-export	Import	Re-export	Import	Re-export
			1	2	3	4	5	6	7	
Object 1 (Wall)	GUID	1qbJ49Z\$X4PRabb1rEx2pE	√	√	√	√	√	√	√	
	name	SW-015	√	√	√	√	√	√	√	
	material	brick	√	√	√	√	√	√	√	
Object 2 (Window)	GUID	2OVxuT1l5Dlwq9QU3Dz7oO	√	√	√	√	√	√	√	
	name	WD-032	√	√	√	√	×	√	√	
								WD-032.125917:WD-032.125917:22137Z		
Object 3 (Beam)	glazing area fraction	16.47	√	√	-	○	√	○	√	
								16.470000		
Object 4 (Reinforcing bar)	GUID	1PFbCy0002YZ4qE]So]Jot	√	√	√	√	√	√	√	
	name	CG1	√	√	√	√	×	√	√	
								$\frac{850 \times 300}{11182}$	$\frac{850 \times 300}{11182} \times \frac{11182:20434Z}{11182:20434Z}$	
Object 5 (Pipeline)	description	850*300	√	√	√	-	-	-	√	
	GUID	1PSXF_000drp4rC30u CZ0m	√	√	√	√	√	√	√	
Object 6 (Terminal)	name	REBAR	√	√	√	√	×	√	√	
								REBAR.23756:REBAR.23756:194903		
Object 7 (Terminal)	total weight	106.3kg	√	√	○	○	-	○	√	
								0.106t	106.300kg	
Object 8 (Terminal)	GUID	2qduGQYqL9hwiBtt9u5jg5	√	√	√	√	√	√	√	
	user code	Rectangular duct	√	√	√	√	-	√	√	
Object 9 (Terminal)	material	galvanized steel sheet	√	√	√	√	-	√	√	
	GUID	3wftq7wLnEfw0w7vepxrQw	-	-	√	√	√	√	√	
Object 10 (Terminal)	user code	wash-basin	-	-	√	√	-	√	√	
	qv_sizing flow_ls	0.6	-	-	√	×	-	×	√	
								0.00		

Note: √ : completely consistent; ○ : consistent in allowable range; × : inconsistent; - : lost.

was lost in Software B. (2) The beam and reinforcing bars interpreted by Software A and Software D had correct properties. Rather than the used unit of a kilogram (kg) in the original model, the unit of a ton (t) was used for the total weight of reinforcing bars in Software B, and its precision (106.300) in Software C was different from the original one (106.3). In addition, the beam name in Software C was wrong. (3) Software A showed correct properties of the pipeline, but lost the terminal. Both Software B and Software D could correctly interpret properties of the pipeline and terminal. Properties of the pipeline were correct in Software C, but the property value of “qv_sizing_flow_ls” of the terminal was wrong.

In re-export test, (1) properties in both Model-A-A and Model-A-D did not differ from those in Model-A. In Model-A-C, the window name was incorrect. (2) In Model-B-A and Model-B-D, no change was found in properties of the beam and reinforcing bars. In Model-B-C, their names were misrepresented, and the total weight of the reinforcing bars was lost. (3) Properties of the pipeline and terminal in Model-C-D were valid. Except for the missing terminal, properties of the pipeline in Model-C-A were correct. Several properties in Model-C-C was lost, such as the material of the pipeline and user code of the terminal.

In conclusion, three types of issues in object properties were as follows: (1) Loss of object properties. For example, the glazing area fraction of the window was lost in Software B, as well as the user code of the terminal in Software C. Through analyzing corresponding IFC entities, these properties were defined by *IfcPropertySingleValue* entities. It indicated that Software B and Software C did not accurately interpret all properties defined by *IfcPropertySingleValue*. (2) Inconsistent property values. This issue mainly occurred in the name, e.g., the beam name. The name is an attribute of *IfcRoot*, and all building elements defined in IFC schema inherit this attribute. These results implied a misrepresentation of the element name. (3) Differences in numerical precision. While the total weight of the reinforcing bars in Model-B was 106.3 kg, the corresponding values in Software B and Software C were 0.106 t and 106.300 kg, respectively. In the allowable range, these property values were identical. The precision difference was caused by the default unit and numerical precision in different software tools.

3.6. Relations between objects

Relations between objects in a building are crucial for business tasks, such as connections for structural analysis, and system relations for facility management. In this experiment, several relations between objects were designed to test how software tools interpreted relations. In Model-A, *IfcRelConnectsPathElements* was used to represent the connection relation between walls; in Model-B, the aggregation relation between concrete objects and reinforcing bars was referred by *IfcRelAggregates*; in Model-C, some objects were assigned to a system by *IfcRelAssignsToGroup*. The numbers of the aforementioned entities were provided in Table 5. (1) In architectural models, the numbers of *IfcRelConnectsPathElements* entities in Model-A-A and Model-A-D were 167, but only 127 ones could be counted in Model-A-C. (2) In structural models, Model-B-C had only 3 *IfcRelAggregates* entities, while the numbers of *IfcRelAggregates* entities in Model-B-A and Model-B-D were 297, as the same as Model-B. (3) In MEP models, both Model-C-A and Model-C-D had 11 *IfcRelAssignsToGroup* entities, and no one was found in Model-C-C. The results showed that some relations between objects could not be re-exported from test tools. The relation between the beam (Object 3) and reinforcing bars (Object 4), for example, was analyzed in the following section.

Figure 5 illustrates the relations between the beam and reinforcing bars from different models. In Model-B, *IfcRelAggregates* specified that the beam and reinforcing bars belonged to an element assembly *IfcElementAssembly*, as shown in Figure 5(a). Even though the beam and reinforcing bars could be displayed in Model-B-C, the *IfcRelAggregates* entity between them was lost. This result reveals that Software C may not support the interpretation of this relation. For further analysis, a test that there was no assembly relation between the beam and reinforcing bars was conducted, and this new model was called Model-B_{new}. In this case, both the beam and reinforcing bars were contained in *IfcBuildingStorey* by using *IfcRelContainedInSpatialStructure* entity, as shown in Figure 5(b). Although these two objects in Model-B_{new} were correctly interpreted by Software C, the assembly relation between them was lost.

Table 9. Average scores to evaluate the interoperability for test objects

	Object	Type	Geometry	Colour	Property	Relation	Total average
Architecture	Wall	1.000	1.000	1.000	1.000	0.833	0.967
	Window	1.000	0.929	1.000	0.857	–	0.946
Structure	Beam	0.667	1.000	0.714	0.857	0.667	0.781
	Reinforcing bars	0.667	0.857	0.714	0.929	0.667	0.767
MEP	Pipeline	1.000	0.929	0.714	0.929	0.667	0.848
	Terminal	0.333	0.643	0.429	0.571	0.333	0.462
Total average		0.778	0.893	0.762	0.857	0.633	0.790

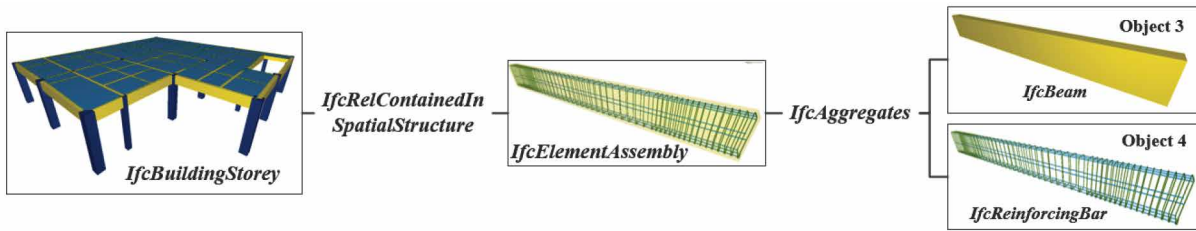
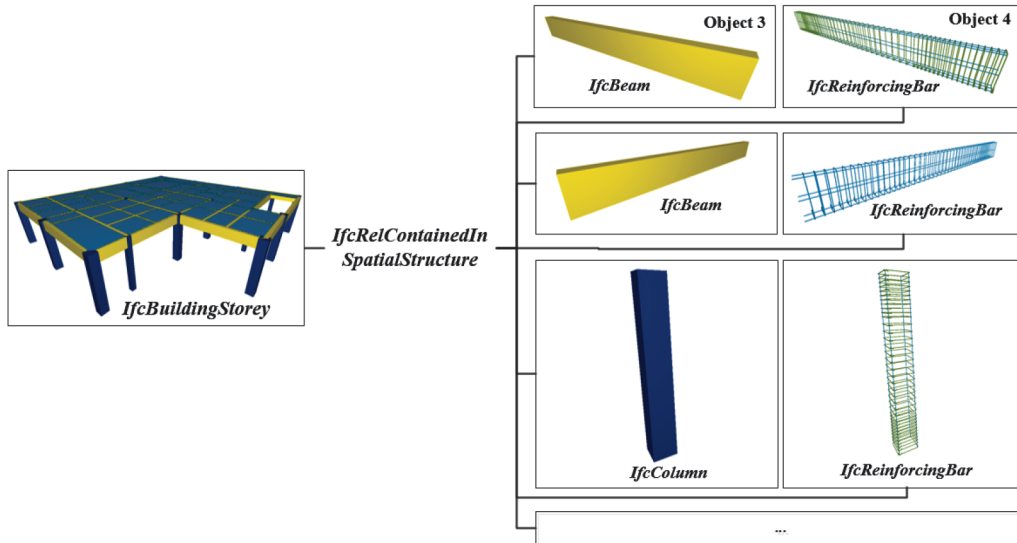
(a) Beam and reinforcing bars within *IfcElementAssembly* in Model-B(b) Beam and reinforcing bars without *IfcElementAssembly* in Model-B_{new}

Figure 5. Relations between the beam and reinforcing bars in different models

3.7. Discussion

This study adopted the scoring mechanism in Muller *et al.* (2017) to evaluate the interoperability in terms of test criteria. Similar to the Likert scale, three scores were designed to evaluate the accuracy of exchanging data: 1 for completely correct, 0.5 for partially correct, and 0 for completely incorrect. According to the results from the import and re-export tests, Table 9 presents the average scores for each test object in terms of test criteria.

According to total average scores in test criteria, the geometry and property perform the best among existing criteria, and the most serious issue is the relation information, which has a lowest average score. Based on these results, software developers can devote to improving the interpretation of corresponding object attributes, especially in the relation information. On the other hand, the score for the terminal is the lowest among test objects, because it is lost in Software A, resulting in lower scores in each criterion. Among other test objects, the score for the wall is the highest, and the reinforcing bar is the lowest. It indicates that IFC interfaces of software tools should provide better support in the interpretation of complex-shaped objects.

4. Discussion on methods for data interoperability

4.1. Current method for data interoperability

The current method for data interoperability between heterogeneous software tools is bidirectional, as shown in Figure 6(a). It is assumed that a designer uses Software A to build a data model (called Set A), and exports this model as IFC format (called Set A'). According to the requirements of business tasks, Software B imports Set A' , which will be mapped to the internal data schema of Software B. Similarly, Set B is assumed to be a set of information expressed by the internal data schema of Software B. Ideally, all elements in Set A' can be mapped to relevant elements in Set B , so that $\forall a \in A' \rightarrow a \in B$, as shown in Figure 6(b). The results from Section 3 indicate that interoperability issues do exist during data sharing and exchange between heterogeneous software tools. It means that one software tool can hardly support all information from other disciplines. As shown in Figure 6(c), while an element a_i in Set A' belongs to Set B (that is, $a_i \in (A' \cap B) \subset B$), there are other elements which don't

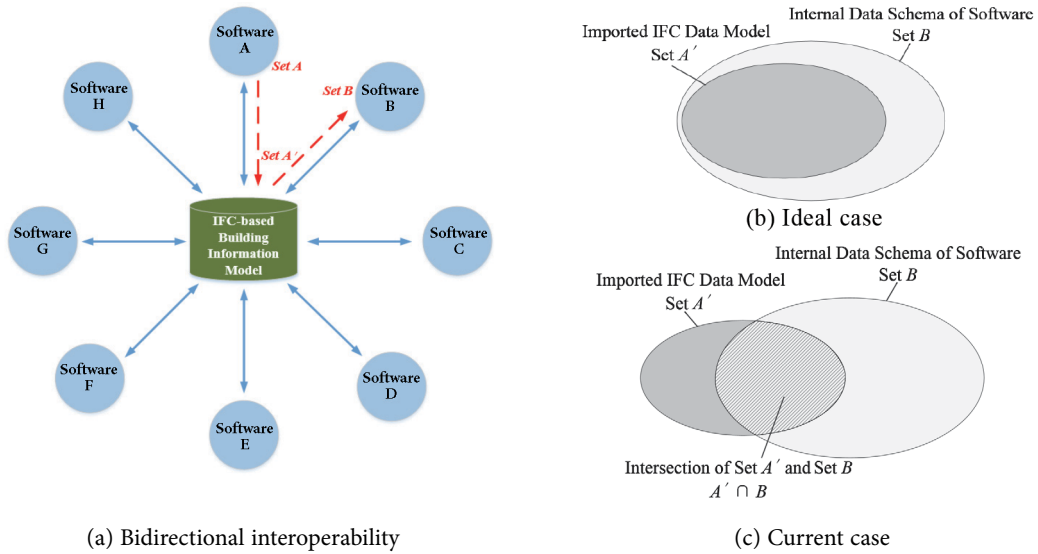


Figure 6. Data interoperability in a bidirectional way

belong to Set B, e.g., $a_j \in A' \wedge a_j \notin B$. If Software A and Software B belong to different disciplines, more elements in Set A' will not belong to Set B because of semantic differences. Mathematical notations in this paper are based on the Mathematical Logic and Set Theory (Oleary 2015).

A software tool can hardly support all information in the AEC/FM industry (Smith, Tardiff 2009). The information supported by a software tool is usually a subset of all information of building projects. When a tool imports a model which is created by itself or belongs to the same discipline, this imported model can be interpreted well. Subsequently, the re-exported model is also highly consistent with the original one. In Figure 7(a), elements a_i ($i = 1, 2, 3, \dots, n$) included in Set A' can be mapped to elements b_i ($i = 1, 2, 3, \dots, n$) in Set B through a function f , that is, $f(a_i) = b_i$ ($i = 1, 2, 3, \dots, n$). Consequently, there is an equivalence relation between the re-exported model (called

Set (A' - B')) and Set A'. If the tool imports a model which is created by a third-party tool, a part of model data is easily misrepresented and/or lost. As shown in Figure 7(b), these mappings between the internal data model and the IFC model can be classified into three cases:

Case 1: For elements a_i ($i = 1, 2, 3, \dots, n$), similar to Figure 7(a), elements a_i can be mapped to elements b_i in Set B, that is, $f(a_i) = b_i$ ($i = 1, 2, 3, \dots, n$). This part of Set (A' - B'), including elements b_i , is consistent with relevant elements in Set A'.

Case 2: For elements a_{n+j} ($j = 1, 2, 3, \dots, m$), there are no elements in Set B mapping elements a_{n+j} by one to one. These elements a_{n+j} can be mapped to one particular element (e.g., the element b_0) through a specific function. In other words, there is a many-to-one mapping relation between elements a_{n+j} and element b_0 , that is, $\exists g(a_{n+j}) = b_0$ ($j = 1, 2, 3, \dots, m$). Although the element b_0 is different

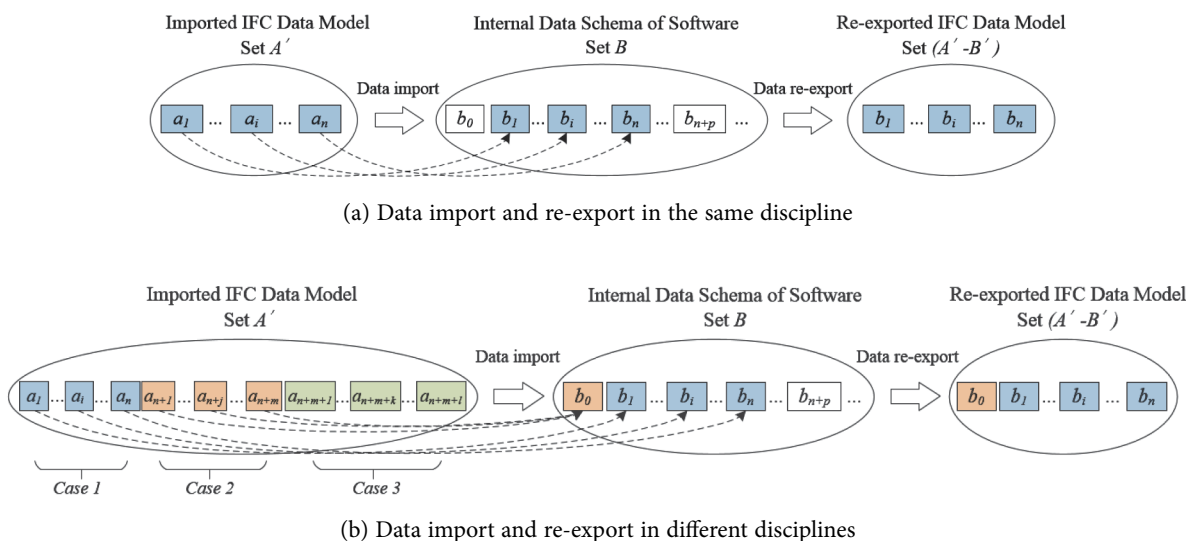


Figure 7. Discussion on mappings between internal models of software and IFC data models

from elements a_{n+j} it can be used to express information represented by a_{n+j} through a function g . In IFC schema, the *IfcBuildingElementProxy* entity is an example for such a purpose, which can be used to represent various IFC object entities. Noted that not all software tools make elements a_{n+j} map to the element b_0 due to diverse mapping mechanisms, resulting in various mappings.

Case 3: For the rest of elements a_{n+m+k} ($k = 1, 2, 3, \dots, l$) in Set A' , there is no element in Set B mapping these elements because of different disciplines, even the element b_0 . In this case, due to the lack of elements mapping to elements a_{n+m+k} , the information represented by elements a_{n+m+k} will be lost in Set $(A' - B')$. As a result, Set $(A' - B')$ has a low accuracy.

In conclusion, during data sharing and exchange between heterogeneous software tools, if the bidirectional data interoperability is used without additional improvement on exchanging data, interoperability issues will likely arise: (1) When importing model data undefined in the internal data schema of software, software tools may lose model data, or map them into other data. Due to diverse mapping mechanisms, it is more prone to various interpretations of the same model data; (2) When software tools export objects with diverse attributes, such as geometry, properties, and relations, methods to define the same attribute may vary depending on software tools, resulting in different representations. According to the existing bidirectional data sharing and exchange, it is recommended to propose a new method to achieve effective data interoperability between heterogeneous BIM software.

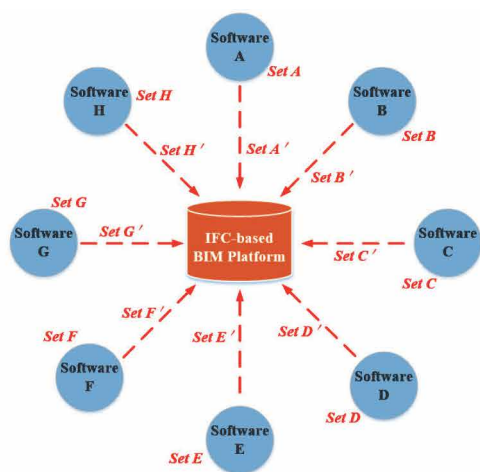
4.2. A suggested method for data interoperability

When a domain-specific software tool exports its own model as IFC format, both exported and original models are very close in information representation because of

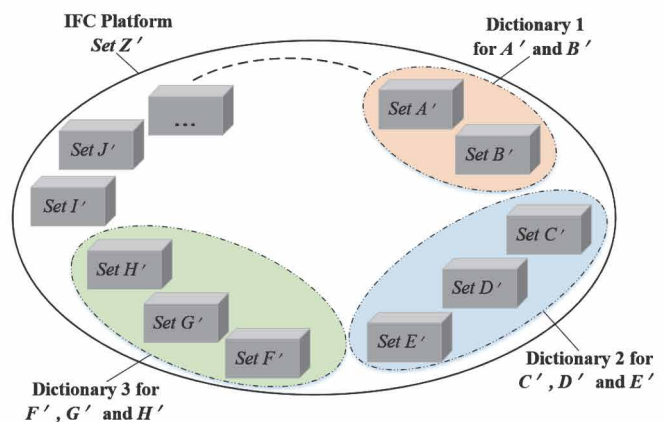
semantic compatibility. However, diverse disciplines are involved in the building project. Results in this experiment demonstrate that using one domain-specific software tool for data exchange between multiple disciplines is prone to interoperability issues. In this experiment, the prototype IFC Platform (Software D) showed a good performance of the interpretation, and relevant results demonstrated the feasibility of an IFC-based BIM platform for data interoperability in practice. Hence, a common BIM platform which fully supports IFC schema is suggested to integrate model data from multiple disciplines, as shown in Figure 8(a). Users select domain-specific software tools to build models and export IFC models. Subsequently, these models are integrated into the IFC-based BIM platform.

Due to diverse disciplines (such as architecture, structure, and MEP) in the AEC/FM industry, the common BIM platform is expected to interpret model data from various disciplines. For this purpose, data dictionaries of diverse disciplines were established in the proposed IFC Platform for mapping plenty of model data. As shown in Figure 8(b), Set Z' is assumed to be a total information set in the IFC Platform, which includes different data dictionaries. Dictionary 1, 2, and 3, for example, are used for representing architectural, structural and MEP models, respectively. It is assumed that both Set A' and Set B' belong to the information sets of architectural models. Hence, Dictionary 1 can cover elements in Set A' and Set B' . Any elements in Set A' or Set B' belong to Set Z' , that is, $\forall a \forall b [A', B' \subset Dictionary1 \wedge a \in A' \wedge b \in B' \rightarrow a, b \in Z']$. A common BIM platform with enough such dictionaries can better interpret models from various disciplines.

To date, IFC schema has defined a great deal of information for diverse disciplines, such as architecture, structure, mechanical, electrical and plumbing. Through integrating corresponding information in IFC schema, data dictionaries could be formed in the proposed



(a) Data interoperability between different software using IFC-based BIM platform



(b) Dictionaries of different disciplines for interpreting IFC data models

Figure 8. Suggested method for data interoperability using IFC-based BIM platform

platform. Furthermore, to effectively process data models, an IFC-based dataset was established in the IFC Platform. This dataset has various information units for storing different kinds of object information in a standard way, such as the name, type, geometry, property, material, colour, and relation. When importing an IFC file, the proposed platform firstly interprets model data according to the inheritance and referencing relations between IFC instances, and then converts the data to relevant information in dictionaries. A lot of building models created by different software tools (such as ArchiCAD, Tekla Structures, Revit, and Allplan) have been used to test the validity of the proposed IFC Platform, and these tests presented satisfactory results, similar to this interoperability experiment. Due to the length limitation of the paper, this paper focuses primarily on the interoperability experiment, and the methods to establish data dictionaries and associated algorithms for interpreting diverse IFC data are to be described in detail in a separate paper (under preparation). The current IFC Platform is not yet available to support all domains in the AEC/FM industry. IFC schema is being developed to cover more domains, e.g., an extension of the infrastructure domain in IFC5. As the development of IFC schema, the proposed platform is promising to better support information in the AEC/FM industry.

In addition, IFC data models exported from software tools may need to be imported into other software tools for their business tasks. For a correct interpretation, besides improving IFC data interfaces by software developers, the next research work will focus on how to import re-exported IFC models to software tools without data loss or misrepresentation.

Conclusions

As BIM software tools have been widely used in the AEC/FM industry, interoperability issues are becoming increasingly prominent. In this paper, a practical project was used to test currently bidirectional data interoperability. Compared to previous tests within one or two domains, this interoperability experiment was designed to test models from architectural, structural and MEP domains. Typical objects in each domain were selected as test objects, which were not limited to “standard objects”, but also included domain-specific objects with complex shapes. Further, designated test criteria, including the types, geometry, colours, properties, and relations, covered feature information of objects.

The results demonstrate that interoperability issues commonly arise, such as the increasing file-size, inconsistent object types, geometric misrepresentation, different colours, loss of properties and relations. Through a discussion, it is found that complex-shaped objects (e.g., the terminal and reinforcing bars) present more interoperability issues, and the biggest issue lies with the relation between objects from the view of test criteria. For such interoperability issues, the causes can be concluded as follows:

- Interoperability issues such as data loss and misrepresentation do exist, when software tools import IFC models created by other software tools. This cause lies in semantic differences. The domain-specific software tool can correctly represent information from its own domain. Information from other domains, however, may be lost or misrepresented by software due to the lack of related knowledge in the internal data schema, such as object type and geometric representation.
- When different software tools export the same information of an object, such as geometry, properties, and relations, different IFC entities may be used to define this information, leading to diverse representations. This reason for such difference is the multiple mappings between internal data schemas of software tools and IFC schema.
- When software tools re-export data models created by other software tools, the increase in physical file sizes, IFC entities, etc. can be found. The cause is that software tools prefer to use general methods to represent information undefined in internal data schemas, which will add more IFC data, e.g., nonparametric geometric representation for the complex shape.

Furthermore, the causes of interoperability issues in each characteristic have been further analyzed within the IFC data perspective. It provides users and software developers with a guide on how to improve data interoperability. Even though interoperability issues do exist during IFC-based data sharing and exchange, it cannot be concluded that the technical roadmap of IFC-based data interoperability is not feasible. According to this interoperability experiment, using the proposed IFC Platform for data interoperability has a good performance. Based on IFC schema, a set of data dictionaries were developed in the proposed platform for fully supporting information from diverse disciplines. The results demonstrate that using the IFC-based BIM platform for data interoperability is feasible in practice. Hence, a suggested method was proposed for improving data interoperability: during data sharing and exchange between heterogeneous software tools, a common BIM platform which fully supports IFC schema is suggested to integrate models from multiple disciplines.

Additionally, some limitations and other work are going to be addressed by future research and development:

- This experiment is limited to a case study with the use of limited test objects and specific software tools. To further analyze interoperability issues in the AEC/FM industry, more objects for specific data exchanges will be tested by using more software tools in the future work.
- Most software tools interpret IFC model data depending on their own mapping mechanisms, resulting in multiple mappings for the same information. The method to standardize these mappings needs to be studied. A typical example is the colour, where five representation methods are found in all test objects.

- The ontology provides necessary semantics for domain knowledge and facilitates the reusability and integration. It is useful to enable data interoperability between heterogeneous BIM software. Hence, the data dictionary based on the ontology is considered in the future work.

Acknowledgements

This work was supported by the National Key Technologies Research and Development Program of China during the 13th Five-Year Plan Period under Grant 2016YFC0702001. This support is gratefully acknowledged.

Author Contributions

Huahui LAI and Xueyuan DENG conceived the study “Interoperability Analysis of IFC-based Data Exchange between Heterogeneous BIM Software”. With the help of Xueyuan DENG, Huahui LAI designed and conducted the interoperability experiment for IFC data exchange between BIM software tools. According to experiment results, Huahui LAI and Xueyuan DENG were responsible for interoperability analysis at the IFC data level. Finally, Huahui LAI wrote the first draft of the manuscript, and Xueyuan DENG revised it for publication.

Funding

This work was supported by the National Key Technologies Research and Development Program of China during the 13th Five-Year Plan Period under Grant 2016YFC0702001.

Disclosure Statement

The authors declare that they do not have any competing financial, professional, or personal interests from other parties.

References

- BuildingSMART. 2010. *IFC2x3 coordination view version 2.0 certification workflow* [online], [cited 20 May 2018]. Available from Internet: http://www.buildingsmart-tech.org/certification/documents/bSI_IFC_Certification_2-0_Workflow_CV-V2.0_Draft1.1.pdf
- BuildingSMART. 2013. *List of software claiming IFC support* [online], [cited 20 May 2018]. Available from Internet: <http://www.buildingsmart-tech.org/implementation/implementations>
- BuildingSMART. 2018. *List of software participating in IFC certification* [online], [cited 20 May 2018]. Available from Internet: <http://www.buildingsmart-tech.org/certification/ifc-certification-2.0/ifc2x3-cv-v2.0-certification/participants>
- Chen, D.; Doumeings, G.; Vernadat, F. 2008. Architectures for enterprise integration and interoperability: Past, present and future, *Computers in Industry* 59(7): 647–659. <https://doi.org/10.1016/j.compind.2007.12.016>
- Choi, J. S.; Kim, I. H. 2011. Interoperability tests between IFC certified software for open BIM based quality assurance, in *Proceedings of the 28th International Symposium on Automation and Robotics in Construction*, 29 June – 2 July 2011, Seoul, Korea. <https://doi.org/10.22260/ISARC2011/0240>
- Eastman, C.; Teicholz, P.; Sacks, R.; Liston, K. 2011. *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*. 2nd ed. New York: Wiley.
- Golabchi, A.; Kamat, V. R. 2013. Evaluation of industry foundation classes for practical building information modeling interoperability, in *Proceedings of the 30th International Symposium on Automation and Robotics in Construction and Mining*, 11–15 August 2013, Montreal, Canada. <https://doi.org/10.22260/ISARC2013/0003>
- Hu, Z. Z.; Tian, P. L.; Li, S. W.; Zhang, J. P. 2017. BIM-based integrated delivery technologies for intelligent MEP management in the operation and maintenance phase, *Advances in Engineering Software* 115: 1–16. <https://doi.org/10.1016/j.advengsoft.2017.08.007>
- Jeong, Y. S.; Eastman, C. M.; Sacks, R.; Kaner, I. 2009. Benchmark tests for BIM data exchanges of precast concrete, *Automation in Construction* 18(4): 469–484. <https://doi.org/10.1016/j.autcon.2008.11.001>
- Kam, C.; Fischer, M.; Hanninen, R.; Lehto, S.; Laitinen, J. 2002. Implementation challenges and research needs of the IFC interoperability standard: Experiences from HUT-600 construction pilot, in *Proceedings of the International Workshop on Information Technology in Civil Engineering: Computing in Civil Engineering*, 2–3 November 2002, Washington, DC, USA, 211–220. [https://doi.org/10.1061/40652\(2003\)18](https://doi.org/10.1061/40652(2003)18)
- Karan, E. P.; Irizarry, J. 2015. Extending BIM interoperability to preconstruction operations using geospatial analyses and semantic web services, *Automation in Construction* 53: 1–12. <https://doi.org/10.1016/j.autcon.2015.02.012>
- Kim, K.; Yu, J. 2016. A process to divide curved walls in IFC-BIM into segmented straight walls for building energy analysis, *Journal of Civil Engineering and Management* 22(3): 333–345. <https://doi.org/10.3846/13923730.2014.897975>
- Kiviniemi, A. 2008. IFC certification process and data exchange problems, in *Proceedings of the 7th European Conference on Product and Process Modelling*, 10–12 September 2008, Nice, France.
- Kiviniemi, A.; Tarandi, V.; Karlshøj, J.; Bell, H.; Karud, O. J. 2008. *Review of the development and implementation of IFC compatible BIM*. Erabuild project report.
- Lee, G. 2011. What information can or cannot be exchanged?, *Journal of Computing in Civil Engineering* 25(1): 1–9. [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000062](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000062)
- Lee, G.; Won, J.; Ham, S.; Shin, Y. 2011. Metrics for quantifying the similarities and differences between IFC files, *Journal of Computing in Civil Engineering* 25(2): 172–181. [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000077](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000077)
- Liao, L.; Teo, E. A. L. 2017. Critical success factors for enhancing the building information modelling implementation in building projects in Singapore, *Journal of Civil Engineering and Management* 23(8): 1029–1044. <https://doi.org/10.3846/13923730.2017.1374300>
- Lipman, R.; Palmer, M.; Palacios, S. 2011. Assessment of conformance and interoperability testing methods used for construction industry product models, *Automation in Construction* 20(4): 418–428. <https://doi.org/10.1016/j.autcon.2010.11.011>

- Ma, H.; Ha, E.; Chung, J.; Amor, R. 2006. Testing semantic interoperability, in *Proceedings of Joint International Conference on Computing and Decision Making in Civil and Building Engineering*, 14–16 June 2006, Montreal, Canada, 1216–1225.
- Muller, M. F.; Garbers, A.; Esmanioto, F.; Huber, N.; Loures, E. R.; Canciglieri, O. 2017. Data interoperability assessment through IFC for BIM in structural design – a five-year gap analysis, *Journal of Civil Engineering and Management* 23(7): 943–954. <https://doi.org/10.3846/13923730.2017.1341850>
- National Institute of Standards and Technology. 2011. *IFC file analyser* [online], [cited 20 May 2018]. Available from Internet: <https://www.nist.gov/services-resources/software/ifc-file-analyzer>
- Nizam, R. S.; Zhang, C. 2015. Current state of information exchange between the two most popular BIM software: Revit and Tekla, in *Proceedings of the 1st International Conference on Sustainable Buildings and Structures*, 29–31 October 2015, Suzhou, China.
- Oh, M.; Lee, J.; Hong, S. W.; Jeong, Y. 2015. Integrated system for BIM-based collaborative design, *Automation in Construction* 58: 196–206. <https://doi.org/10.1016/j.autcon.2015.07.015>
- Olawumi, T. O.; Chan, D. W. M.; Wong, J. K. W. 2017. Evolution in the intellectual structure of BIM research: A bibliometric analysis, *Journal of Civil Engineering and Management* 23(8): 1060–1081. <https://doi.org/10.3846/13923730.2017.1374301>
- Oleary, M. L. 2015. *A first course in mathematical logic and set theory*. New York: Wiley.
- Pazlar, T.; Turk, Z. 2008. Interoperability in practice: Geometric data exchange using the IFC standard, *Electronic Journal of Information Technology in Construction* 13: 362–380.
- Ramaji, I. J.; Memari, A. M. 2016. Interpreted information exchange: Systematic approach for BIM to engineering analysis information transformations, *Journal of Computing in Civil Engineering* 30(6): 04016028. [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000591](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000591)
- Ramaji, I. J.; Memari, A. M. 2018. Extending the current model view definition standards to support multi-storey modular building projects, *Architectural Engineering and Design Management* 14: 158–176. <https://doi.org/10.1080/17452007.2017.1386083>
- Sanguinetti, P.; Abdelmohsen, S.; Lee, J.; Lee, J.; Sheward, H.; Eastman, C. 2012. General system architecture for BIM: An integrated approach for design and analysis, *Advanced Engineering Informatics* 26(2): 317–333. <https://doi.org/10.1016/j.aei.2011.12.001>
- Shi, X.; Liu, Y. S.; Gao, G.; Gu, M.; Li, H. J. 2018. IFCdiff: A content-based automatic comparison approach for IFC files, *Automation in Construction* 86: 53–68. <https://doi.org/10.1016/j.autcon.2017.10.013>
- Sibenik, G. 2016. Building information modelling based interdisciplinary data exchange: A case study, in *The 1st International UK BIM Academic Forum Conference*, 13–15 September 2016, Glasgow, Scotland.
- Smith, D. K.; Tardiff, M. 2009. *Building information modeling: A strategic implementation guide for architects, engineers, constructors and real estate asset managers*. New York: Wiley. <https://doi.org/10.1002/9780470432846>
- Taher, A. A. 2016. *BIM software capability and interoperability analysis: An analytical approach toward structural usage of BIM software (S-BIM)*: Ms thesis. Royal Institute of Technology, Stockholm, Sweden.
- Tolman, F. 1999. Product modeling standards for the building and construction industry: Past, present, and future, *Automation in Construction* 8(3): 227–235. [https://doi.org/10.1016/S0926-5805\(98\)00073-9](https://doi.org/10.1016/S0926-5805(98)00073-9)