Validation of IFC model based on high-level interface: Specifications for rule-checking error reporting

Donghoon Yang, donghoon.yang@coa.gatech.edu
Georgia Institute of Technology, Atlanta, United States

Charles M. Eastman, chuck.eastman@coa.gatech.edu
Georgia Institute of Technology, Atlanta, United States

Abstract
There are two potential user groups in validating exported IFC instance files. First is IFC expert and second is software user using the export. The expectations from these two user groups are different in the sense of error reporting. While the expert group may need to decode each line of IFC instance file, the software users cannot understand internal structure IFC instance.
This paper presents an error reporting specification for software users not familiar with IFC in validating their IFC exports and correcting their native BIM models. We use the notion of the concept template and the semantic exchange model, which are high-level modules encapsulating underlying IFC data structure that can represent data exchange requirements in domain specific terms familiar to software users.
We lay out the proposed error reporting specification in a prototype validation system on top of the current validation module of ifcDoc and compare it with the existing validation tools.

Keywords: Validation, Concept Template, Semantic Exchange Model, IFC

1 Introduction

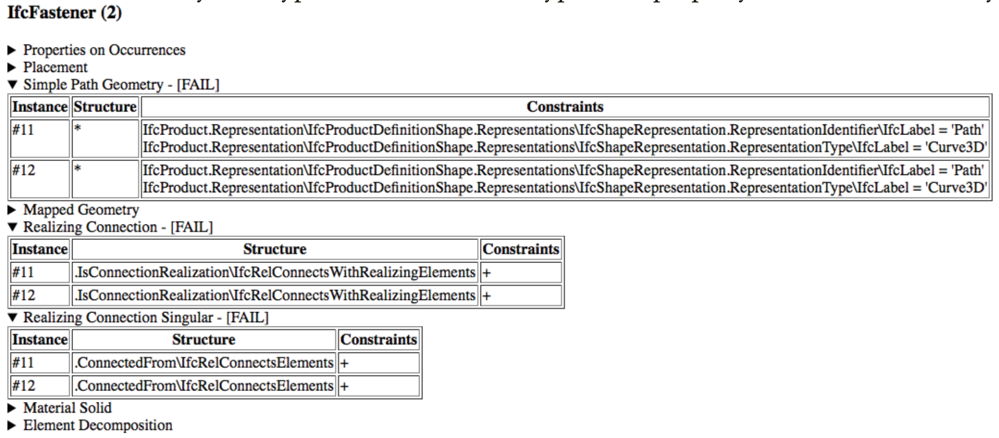
Data exchange and collaboration is essential in enhancing productivity in the building industry. The interoperability through open standards, Industry Foundation Classes (IFC), has become a standard medium (Eastman, 2011). IFC data model (IFC schema) needs to address data types used in the life cycle of a building, which have flexibility for representing multiple ways of representing the same information in different data types, as well as the multiple views of the same object depending on the domain of and LOD of the building industries.

There are two types of validation for IFC models. First is the rule checking of building designs as in (Eastman et al., 2009, Beach et al., 2013, Beach et al., 2015, Malsane et al., 2015), which validate rules of building codes or building programming such as circulation requirements. These approaches use added expert knowledge and manipulation of available information in the BIM model. The rules are often defined in a rule definition language (Lee, 2010, Lee, 2011) or built into the validation system (Choi et al., 2014). Its goal is to evaluate the quality of a building design instance. Evaluating the quality of data is found in (Zhang et al., 2014), which deals with rules defined in an MVD. Validation against a Model View Definition (MVD) checks the integrity of the populated data. Rules are described on Concept Template, and stored to mvdXML. The rule checking of building design using IFC file is information source of validation but it is not validating whether the data stored in IFC file is populated according to an MVD.

MVD is a subset of the IFC schema that specifically indicates what data type is needed for a particular use case. An MVD addresses the needs by restricting the usage of data types as well as adding required user-defined properties if needed. MVD defines a set of data types used for a specific data exchange. For example, the Coordination View MVD targets the coordination between the architectural, mechanical and structural engineering tasks during the design stage(buildingSmart International, 2013), and the AISC steel detailing view is for a coordination between structural steel detailing and steel fabrication(AISC, 2011).

Increasing numbers of MVDs are being defined and utilized and there are some checking platforms available, such as ifcDoc from buildingSmart International, and IFC BIM Validation Service(Digital Alchemy, 2013). The latter's implementation started prior to the development of mvdXML, and MVD specification for a formal description of MVD(Chipman, 2012b, mvdXML Group, 2014). mvdXML supports MVD and rule types that evaluate existence, quantities, correctness, uniqueness, and conditional dependencies. Its implementation relies on hard coded rules.

ifcDoc was developed '*to improve the consistent and computer-interpretable definition of MVD as true subsets of the IFC Specification with enhanced definition of concepts*'(Chipman, 2012a). Figure 4 shows HTML version of a validation report from ifcDoc. Validation report identifies two errors in two instances of ifcFastener. The errors are generated because ifcFastener instances didn't have proper geometry representation nor connection relationship instances. When validating an IFC file, domain experts should know IFC schema to interpret ifcDoc validation results. However, expecting domain users to understand the IFC schema and sub-schema can be unrealistic considering IFC4 schema has 768 entity data types, 130 defined data types, 410 property sets and more data types.



IfcFastener (2)

▶ Properties on Occurrences
▶ Placement
▼ Simple Path Geometry - [FAIL]

Instance	Structure	Constraints
#11	*	IfcProduct.Representation\IfcProductDefinitionShape.Representations\IfcShapeRepresentation.RepresentationIdentifier\IfcLabel = 'Path' IfcProduct.Representation\IfcProductDefinitionShape.Representations\IfcShapeRepresentation.RepresentationType\IfcLabel = 'Curve3D'
#12	*	IfcProduct.Representation\IfcProductDefinitionShape.Representations\IfcShapeRepresentation.RepresentationIdentifier\IfcLabel = 'Path' IfcProduct.Representation\IfcProductDefinitionShape.Representations\IfcShapeRepresentation.RepresentationType\IfcLabel = 'Curve3D'

▶ Mapped Geometry
▼ Realizing Connection - [FAIL]

Instance	Structure	Constraints
#11	.IsConnectionRealization\IfcRelConnectsWithRealizingElements	+
#12	.IsConnectionRealization\IfcRelConnectsWithRealizingElements	+

▼ Realizing Connection Singular - [FAIL]

Instance	Structure	Constraints
#11	.ConnectedFrom\IfcRelConnectsElements	+
#12	.ConnectedFrom\IfcRelConnectsElements	+

▶ Material Solid
▶ Element Decomposition

Figure 1 ifcDoc validation error report

This paper proposes a user interface for domain experts to validate IFC file against MVDs. We present a user interface prototype based on high-level interface, which hides underlying IFC data structure from the domain experts. To provide context for this change,we revisit the role of 'Concept' for defining MVD, its use in ISO 10303 and in STEP-NC. Aspects of these other approaches are adapted, and proposed in a new interface based on a 'Black Box'.

2 'Concept' in IFC

2.1 Concept Template

Validation of an MVD relies on the notion of 'Concept'. A 'Concept' is a collection of data types including entity types, defined types, rules, functions and property sets of IFC schema, which can represent specific information semantics. Therefore, it can be used as a module for constructing a subset of whole IFC schema (Hietanen, 2000). Defining reusable subset of IFC schema in defining MVD are found in Concept Template, Semantic Exchange Module, and Concept Template.

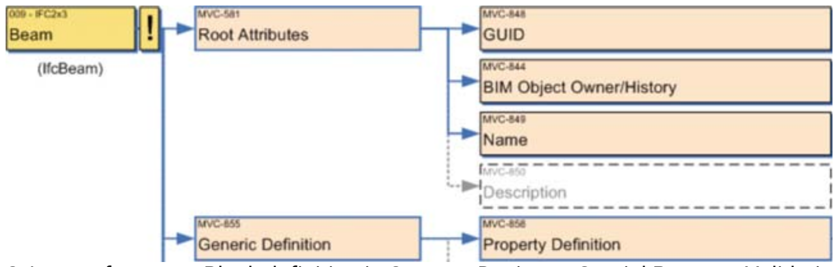


Figure 2 A part of concept Block definition in Concept Design to Spatial Program Validation MVD

Figure 2 is an implementation diagram that shows a Concept Block definition as defined in IFC Solutions Factory (BLIS Consortium and Digital Alchemy, 2012), which specifies a concept block of Beam. Each box in the diagram represents individual concept block and arrowed line represents Beam with Root Attributes, Generic Definition and other sub components that can have their own sub components respectively. A Concept Block can represent IFC entity data type, attribute or collection of both. Root Attributes have collection of attributes that belong to IfcRoot entity data type, where BIM Object Owner/History is another entity data type while GUID (Globally Unique IDentifier), Name and Description are attribute of IfcRoot entity data type. In this use of Concepts, they are defined ad hoc and opportunistically for possible re-use within an MVD.

2.2 Semantic Exchange Module

Semantic Exchange Module(SEM)(Venugopal, 2011) is semantically sound Concept Block that can be used to develop component based software engineering (McIlroy, 1968), where each SEM becomes a component and later used for implementing IFC import/export modules for BIM authoring tools. Another goal of SEM is to have larger subset of IFC data types. A SEM definition of ReinforcingBar in Figure 3 includes reinforcing bar's relationship to building element, placement, geometry representation method and other necessary information. These would have been defined as individual subset in Concept Block.

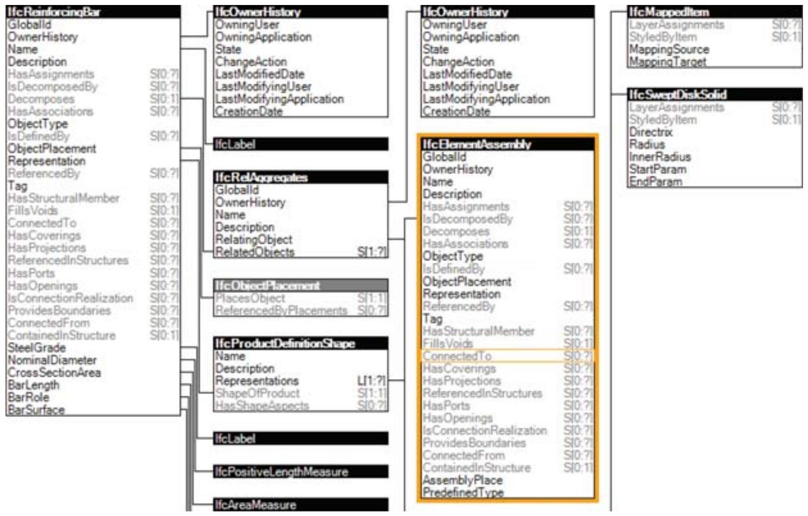
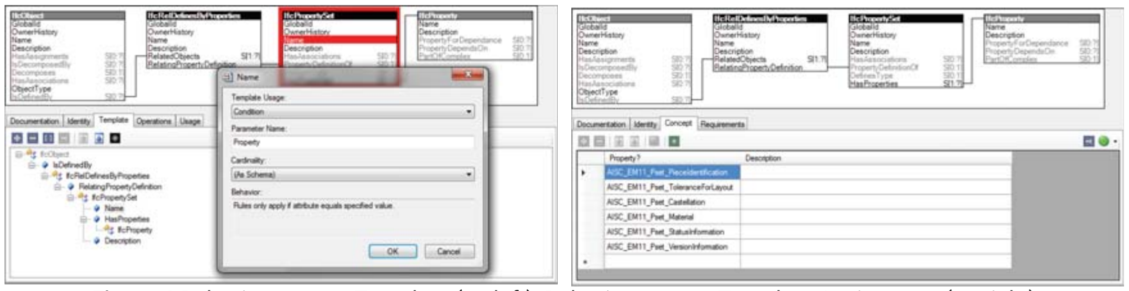


Figure 3 Part of Precast Rebar Assignment SEM Concept Block in Precast Model View

2.3 Concept Template

Concept Template defines data types that can be used with assignable entity data types in IFC schema. **Figure 4** represents a Concept Template definition named 'properties on occurrences' relating IfcObject to IfcRelDefinedByProperties and IfcPropertySet for defining properties of IfcObject. This can be assigned to any subtypes of IfcObject representing any semantically treated thing or process.



Name

Template Usage: Condition

Parameter Name: Property

Cardinality: (As Schema)

Behavior: Rules only apply if attribute equals specified value.

Property?	Description
AISC_EM11_Pset_PieceIdentification	
AISC_EM11_Pset_ToleranceForLayout	
AISC_EM11_Pset_Castellation	
AISC_EM11_Pset_Material	
AISC_EM11_Pset_StatusInformation	
AISC_EM11_Pset_VersionInformation	

Figure 4 Rules in Concept Template (on left), Rules in Concept Template Assignment (on right)

The pop-up window defines a conditional rule that can specify the value of IfcPropertySet.Name attribute. This definition of rule is later assigned and possible values of IfcPropertySet.Name are specified in assignment of Concept Template. The validation process will check if an IfcRelDefinesByProperties instance attribute points to an instance of IFC entity and ifcPropertySet instance, of which Name attribute is one of listed values. A Concept Template can also be defined with preset values of attributes, such that IfcPropertySet.Name has a list of possible values in the Concept Template definition, and assigned without further specification of possible values.

3 Application Protocol and STEP-NC

ISO 10303(ISO TC184/SC4, 1992) Application Protocols (APs) provide comprehensive requirements for concept implementations by defining the application domain (or context), which is similar to MVD in IFC. An AP comprises four main sections: Scope, Requirements (ARM), AIM and Conformance requirements. The scope is described in plain English, Application Activity Model (AAM) and Planning Model. (Owen, 1993, STEP Tools, Hardwick and Spooner, 2001)

ISO 10303 requires having a mapping table between Application Reference Model (ARM) and Application Interpreted Model (AIM) in documenting Application Protocol (AP). ARM is 'an information model that describes the information requirements and constraints of a specific application context', AIM is 'an information model that uses the integrated resources necessary to satisfy the information requirements and constraints of an application reference model, within an application protocol', and AP 'species an AIM satisfying the scope and information requirements for a specific application'(ISO TC184/SC4, 1992). In other words, ARM is a collection of data types defined in plain language, AIM is one defined in ISO 10303 data types, and AP includes scope of exchange, process model, exchange requirements and both ARM and AIM. In IFC, ARM is similar to Exchange Requirements in IDM, AIM is IFC schema or MVD. While ARM uses plain languages, some AP utilizes EXPRESS modeling language for its representation.

Table 1 Modeling Language Application (NIST, 1999, Owen, 1993, Patil et al., 2000)

STEP Architecture Component	Role of Component	Descriptive Language Used
AAM	to capture activities to be supported by the AP	IDEF0, SADT
Application Objects and assertions	to specify information requirements of the AP	English
Planning Model	to show high level representation of the AP	IDEF1X, NIAM, EXPRESS-G
ARM	to facilitate understanding of the AP information requirements	IDEF1X, NIAM, EXPRESS-G
AIM	to capture the activities and information flows to describe the interpretation of generic facts about a product in a specific application context	EXPRESS, EXPRESS-G
Mapping Tables	to trace the application information requirements	Mapping Table syntax (derived from EXPRESS)
Integrated Resources	to define the resource constructs for a particular AP	EXPRESS, EXPRESS-G

Mapping between ARM and AIM mostly is documented in tabular form in AP. STEPview, ARM / AIM Browser and Editor, P.D.I.T.'s ARM Translator Interfaces (ATI), and JSDAI are some of commercial ARM and AIM mapping software. (Wilson, 1998, Phadnis and Nazemetz, 1995, Rech et al., 2004) P.D.I.T claims that developers of STEP translators can create and read AIM-based STEP data files (ISO 10303-21 format) in a matter of days with ATI, because ATI provides ARM-based application object creation and retrieval calls that create and navigate AIM data structures. (Wilson, 1998)

Use of ARM as a direct data population method can be found in STEP-NC. STEP-NC has ARM version defined in ISO 14649 and the AIM version defined in ISO 10303 AP238. ISO 14649 provides a detailed analysis of the requirements of CNC applications, which are the specific objects and the relationships among them. AP238 maps ARM into IRs of ISO 10303. (Feeney et al., 2003, Wolf, 2003) STEP-NC shows benefit and shortcomings of using ARM in data exchange as in Table 2. ISO 14649 is suitable for exact information from the shop floor, and ISO 13030 AP 238 is suitable for complete design and incorporating other types of STEP data.

Table 2 Comparisons between an ARM and AIM model (Xu, 2006)

Comparison criteria	**ISO 14649 (ARM) model**	**ISO 10303-238 (AIM) model**
Storage needed	-10 times less than AIM	-10 times more than ARM
Programming	Easy	More complex
Human readable	Difficult	Almost impossible
Compatibilities with STEP	Partly compliant	Fully compliant
Data consistency	Original design information is abandoned	Original design information is preserved

According to the experiment of Kramer et al. (2006) on ARM and AIM STEP-NC interpreters for milling operations, the ISO 14649 interpreter has approximately 42,000 lines of automatically generated code from ST-Developer of STEP Tools and 12,000 lines of hand written source code. The AP 238 interpreter has about 60,000 lines of automatically generated code and 17,000 lines of source code hand written, which comprises of 12,000 lines largely similar or identical to 12,000 lines of ISO 14649 interpreter and 5,000 lines of data access code. The data access is to utilize STIX (STEP Index Library) of STEP Tools (STEP Tools, 2003), which is a C++ library of functions for creating and modifying applications using STEP-NC (AP 238) machining data, and is proven to be an effective method of manipulating the complex data model mapped into AIM (Liu et al., 2006). Kramer et al. (2006) concluded that building a STEP-NC interpreter is a difficult job for a system programmer, and implementing AP 238 is substantially more difficult that implementing ISO 14649.

4 Black Box

In defining information items in AISC Steel Detailing Model, a simple table has been utilized as shown in Table 3. The table lists information items as depicted by domain experts. Modeling experts organized the information items later to have consistency and increase readability. Throughout meetings among domain experts, software developers and modeling experts, the information items are finalized and the properties that have to be associated with the information items are identified to lower level so that they can clearly identify what are the actual data in the Steel Detailing Model data exchange.

Table 3 Part of a mapping table for AISC Steel Detailing MVD

Exchanged Requirement (ARM)	**Black Box**	**IFC Schema (AIM)**
Assembly	Assembly	IfcElementAssembly
Identification	Assembly Identification	
Guid	Assembly Identification	IfcElementAssembly.GlobalId
Assembly name	Assembly Identification	IfcElementAssembly.Name
AssemblyMark	Assembly Identification	IfcProperty.NominalValue
ClientMark	Assembly Identification	IfcProperty.NominalValue
PrelimMark	Assembly Identification	IfcProperty.NominalValue
ShippingMark	Assembly Identification	IfcProperty.NominalValue
BarCode	Assembly Identification	IfcProperty.NominalValue
Shape	Geometry	
Location in space	Product placement	IfcProduct.ObjectPlacement
Relations		
Elements in the assembly	Components	IfcElementAssembly.IfcRelAggregates
Spatial structure	Placed In	IfcElementAssembly.IfcRelContainedInSpatialStructure

The Black Box column in the table represents a new approach defining reusable modules. It follows the same approach as in Concept Template/Block or SEM. The naming of Black Box comes from encapsulation in object oriented programming, which modularizes a set of attributes and methods into an object definition and hides their implemental structure, and makes an object in software engineering look like a black box to software engineers. Therefore, encapsulation enables the separation of the *what* from the *how*. *What* specifies what behavior an object is capable of and *how* specifies how the data and methods of an object are implemented(Poo et al., 2008).

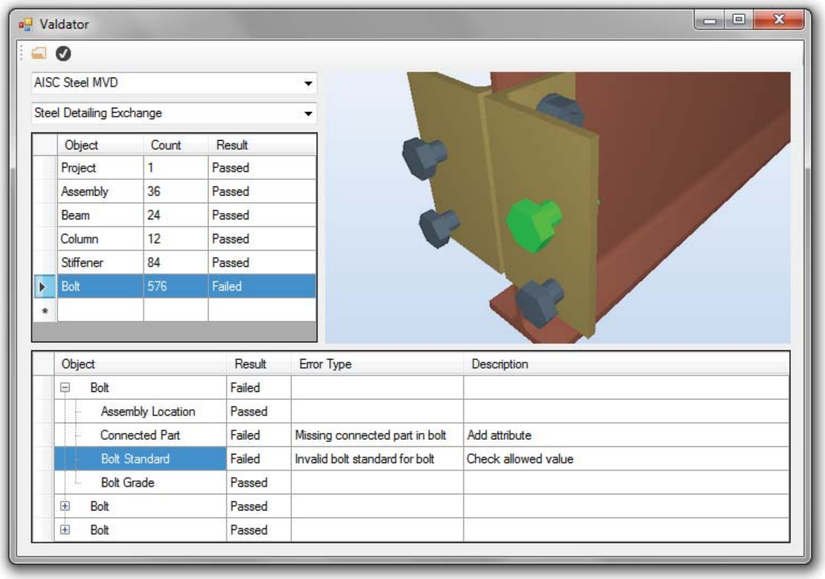
Similar to STEP-NC, the goal of Black Box is to use domain specific terminologies rather than IFC terminologies of IFC schema. There are three groups of experts in defining MVD or open data formats. First group is the domain experts who generate or consume BIM model in their fields of expertise when data exchange is needed. Second group is the software engineers who build BIM software applications to be used by the domain experts. Third group is the information modelers who specify open data formats. The current practices of IFC related data exchanges ask software developers to have thorough understanding of IFC. The currently available IFC related toolkits from STEP Tools, EPM technologies, Eurostep or others require developers to have understanding of IFC data schema along with targeted MVD. Domain experts also need to have some degree of understanding in order customize IFC export to meet with certain requirements of exchange (Autodesk, 2015, Graphisoft, 2015).

An implementation of Black Box module in programming language provides two types of methods. One is for creating IFC data and the other is for retrieving IFC data. Internally, it has attributes and mapping methods between exchange requirements (ARM) and IFC data types (AIM). Therefore, software developers can develop software modules that instantiate or retrieve data IFC instances without knowing IFC data structure (AIM), and the Black Box provides base functionality to validate IFC instances in high level representation (ARM) in domain expert vocabularies instead of IFC data types. This encapsulated mapping capability in the Black Box allows direct utilization of mvdXML definition in IFC data type description (AIM) and generate error report in domain specific terminologies (ARM).

5 Prototype Black Box

The prototype shows validation of an IFC file against an Exchange in a MVD. The validation checks IFC data and reports validation results through the prototype's graphical user interface. The validation process first requires user to open an IFC instance file. According to the MVD and exchange name specified in FILE_DESCRIPTION attribute in the HEADER of the file, validation process reads in the MVD from local storage or accesses the MVD database. Validation reports the object types and count. Detailed information of each failed object shows error types and descriptions with a highlighted 3D object with the error.

Figure 1 is a snapshot of a prototype system showing validation result. The prototype is yet in its drawing board, actual implementation needs further revision on the internal structure of the Black Box module and formal representation of mapping between AIM and ARM.



Object	Count	Result
Project	1	Passed
Assembly	36	Passed
Beam	24	Passed
Column	12	Passed
Stiffener	84	Passed
Bolt	576	Failed

Object	Result	Error Type	Description
Bolt	Failed		
Assembly Location	Passed		
Connected Part	Failed	Missing connected part in bolt	Add attribute
Bolt Standard	Failed	Invalid bolt standard for bolt	Check allowed value
Bolt Grade	Passed		
Bolt	Passed		
Bolt	Passed		

Figure 5 Prototype Interface for Validation

6 Summary

We suggest definition of a mapping table between exchange requirements (ARM) and IFC data types (AIM) by utilizing mapping capability of the Black Box. By having mapping between ARM and AIM in mvdXML, validation on IFC file against MVD can generate domain expert friendly error reports as well as software engineers' benefit of using easier development processes with the Black Box module, as illustrated in STEP-NC.

This prototype user interface is mainly designed for domain experts, but it can be also used by software engineers only if they use the Black Box module in their implementation of IFC import / export modules. We stated that this approach is currently only for validating against MVD, because mvdXML is not yet able to represent both design validation rules and computational geometry algorithms.

References

AISC (2011) *BIMsteel initiatives* [Online]. Available: http://www.aisc.org/bimsteel [Accessed May 31 2015].

Autodesk (2015) *Exporting to Industry Foundation Classes (IFC)* [Online]. Available: http://knowledge.autodesk.com/support/revit-products/learn-explore/caas/CloudHelp/cloudhelp/2015/ENU/Revit-DocumentPresent/files/GUID-6EB68CEC-6C17-4B16-A509-30537F666C1F-htm.html [Accessed May 31 2015].

Beach, T. H., Kasim, T., Li, H., Nisbet, N. & Rezgui, Y. (2013) Towards Automated Compliance Checking in the Construction Industry. *In:* Decker, H., Lhotská, L., Link, S., Basl, J. & Tjoa, A. (eds.) *Database and Expert Systems Applications.* Springer Berlin Heidelberg, pp. 366-380.

Beach, T. H., Rezgui, Y., Li, H. & Kasim, T. (2015) A rule-based semantic approach for automated regulatory compliance in the construction sector. *Expert Systems with Applications,* 42 (12), pp. 5219-5231.

BLIS Consortium & Digital Alchemy (2012) *IFC Solutions Factory* [Online]. Available: http://www.blis-project.org/IAI-MVD/ [Accessed May 30 2015].

buildingSmart International (2013) *Coordination View Version 2.0* [Online]. Available: http://www.buildingsmart-tech.org/specifications/ifc-view-definition/coordination-view-v2.0 2105].

Chipman, T. (2012a) *ifcDoc Tool Summary* [Online]. Available: http://www.buildingsmart-tech.org /specifications/specification-tools/ifcdoc-tool [Accessed May 31 2015].

Chipman, T. (2012b) mvdXML 1.0. buildingSmart International Model Supporting Group.

Choi, J., Choi, J. & Kim, I. (2014) Development of BIM-based evacuation regulation checking system for high-rise and complex buildings. *Automation in Construction,* 46, pp. 38-49.

Digital Alchemy (2013) *IFC BIM Validation Service* [Online]. Available: http://www.digitalalchemypro.com /html/services/IfcBimValidationService.html [Accessed May 31 2015].

Eastman, C., Lee, J.-m., Jeong, Y.-s. & Lee, J.-k. (2009) Automatic rule-based checking of building designs. *Automation in Construction,* 18 (8), pp. 1011-1033.

Eastman, C. M. (2011) BIM handbook a guide to building information modeling for owners, managers, designers, engineers and contractors. 2nd ed. Hoboken, NJ: Wiley.

Feeney, A. B., Kramer, T., Proctor, F., Hardwick, M. & Loffredo, D. (2003) STEP NC implementation - ARM or AIM? *ISO T24 STEP Manufacturing Meeting.* San Diego, USA.

Graphisoft (2015) *Creating and Editing IFC Data* [Online]. Available: http://helpcenter.graphisoft.com/guides/archicad-18-int-reference-guide/interoperability/file-handling-and-exchange/working-with-ifc/working-with-ifc-data/creating-and-editing-ifc-data/ 2015].

Hardwick, M. & Spooner, D. L. (2001) STEP Services for Sharing Product Models. *Journal for Computing and Information Science in Engineering,* 1 (3), pp. 266-268.

Hietanen, J. (2000) *The concept block approach to view definitions* [Online]. Available: http://www.blis-project .org/ [Accessed May 31 2015].

ISO TC184/SC4 (1992) ISO 10303 Part 1 Overview and Fundamental Principles.

Kramer, T. R., Proctor, F., Xu, X. & Michaloski, J. L. (2006) Run-time interpretation of STEP-NC: implementation and performance. *International Journal of Computer Integrated Manufacturing,* 19 (6), pp. 495-507.

Lee, J. K. (2011) Building environment rule and analysis (BERA) language. Atlanta, Ga.: Georgia Institute of Technology.

Lee, J. M. (2010) Automated checking of building requirements on circulation over a range of design phases. Atlanta, Ga.: Georgia Institute of Technology.

Liu, R., Zhang, C. & Newman, S. T. (2006) A framework and data processing for interfacing CNC with AP238. *International Journal of Computer Integrated Manufacturing,* 19 (6), pp. 516-522.

Malsane, S., Matthews, J., Lockley, S., Love, P. E. D. & Greenwood, D. (2015) Development of an object model for automated compliance checking. *Automation in Construction,* 49 (Part A), pp. 51-58.

McIlroy, M. D. (1969) Mass produced software components. Software Engineering: Report of a conference sponsored by the NATO Science Committee, 1968 Garmisch, Germany. pp. 138-151.

mvdXML Group (2014) *mvdXML Requirements and Examples* [Online]. buildingSMART International. Available: https://github.com/BuildingSMART/mvdXML/tree/master/mvdXML1.1 [Accessed May 31 2015].

NIST (1999) STEP The Grand Experience. *In:* Kemmerer, S. J. (ed.). NIST.

Owen, J. (1993) *STEP An Introduction*, Information Geometers Ltd.

Patil, L., Dutta, D., Bhatt, A. D., Jurrens, K., Lyons, K., Pratt, M. J. & Sriram, R. D. Representation of Heterogeneous Objects In ISO 10303 (STEP). ASME International Mechanical Engineering Congress and Exposition, 2000 Orlando, Florida. pp. 355-363.

Phadnis, V. V. & Nazemetz, J. W. (1995) *Impediments to STEP: A Quantitative Approach* [Online]. Oklahoma State University.

Poo, D., Ashok, S. & Kiong, D. (2008) Object-Oriented Programming and Java. London: Springer-Verlag London Limited.

Rech, R., Klein, L., Randis, R., Baltramaitis, T. & Nargelas, V. (2004) Integrating Distributed Applications on the Basis of STEP Data Models. LKSoftWare GmbH.

STEP Tools, I. (2003) *STEP Tools, Inc. Launches STEP Index Library (STIX) Library for Creating and Modifying Applications using STEP-NC Machining Data* [Online]. [Accessed May 31 2015].

STEP Tools, I. (2014) *STEP NC Frequently Asked Questions* [Online]. [Accessed May 31 2015].

Venugopal, M. (2011) *Formal Specification of Industry Foundation Class Concepts using Engineering Ontologies.* PhD, Georgia Institute of Technology.

Wilson, P. R. (1998) *EXPRESS Tools and Services* [Online]. Available: http://www.mel.nist.gov/sc4/tools/express/etools98.htm [Accessed Apr 9 2012].

Wolf, J. (2003) Requirements in NC machining and use cases for STEP NC Analysis of ISO 14649 (ARM) and AP 238 (AIM). *ISO T24 STEP Manufacturing Meeting.* San Diego, USA.

Xu, X. W. (2006) STEP-NC to Complete Product Development Chain. *In:* Ma, Z. (ed.) *Database Modeling for Industrial Data Management: Emerging Technologies and Applications.* IGI Global, pp. 148-184.

Zhang, C., Beetz, J. & Weise, M. (2014) Model view checking: automated validation for IFC building models. *In:* Mahdavi, A., Martens, B. & Scherer, R. (eds.) *eWork and eBusiness in Architecture, Engineering and Construction.* Vienna, Austria.