Transfer of Architectural Data from the IFC Building Product Model to a Fire Simulation Software Tool

MICHAEL SPEARPOINT*

Department of Civil Engineering, University of Canterbury
Christchurch, New Zealand

ABSTRACT: A standardized, object-oriented building product model for buildings is introduced that can be used as a means of electronic exchange between various software tools. The ability to transfer architectural data between a commercially available computer-aided design (CAD) program and a widely available zone fire simulation tool illustrates the applicability of this model in fire engineering. This article describes the software developed to interpret the building product model and the test buildings used to verify the exchange process. In general the building geometry, topology, and other properties can be transferred satisfactorily but some inconsistencies exist due to the structure of the building product model, the CAD implementation of the model, and the simplifications required by the zone modeling approach.

KEY WORDS: fire simulation software, building product modeling, IFC building product model.

INTRODUCTION

THE INCREASING POWER and availability of personal computers coupled with advances in fire science means that the use of fire simulation software tools (commonly referred to as 'fire models') has become more prevalent over the past two to three decades. Prior to execution of fire simulation software tools, basic information, such as the geometry and topology of a building, the location of items within the building, and the

*E-mail: michael.spearpoint@canterbury.ac.nz

1042-3915/07/04 0271–22 $10.00/0 DOI: 10.1177/1042391507074681

SAGE Publications, Los Angeles, London, New Delhi and Singapore

properties of those items may be required. Depending on the fire simulation software tool being used, the fire performance features associated with the structure, contents, or fire protection equipment may also be specified. The input of this information can take a considerable amount of time with the possibility of mistakes and missing information leading to inappropriate output from the simulation tool.

Computer-aided design (CAD) software is widely used to document the design of a building and the capacity to interpret CAD files by simulation software tools is not a new concept. For example, Massa and Cappuccio [1] concluded that it was readily achievable to adapt a proprietary topological analysis CAD system for modeling bomb blast effects as an interface to fire simulation software although it is unclear whether any subsequent development was undertaken.

In cases where the exchange of CAD data with fire simulation software has already been attempted, efforts have been hindered by the inability of the simulation software to interpret CAD data and the inability of CAD systems to adequately describe a building in a way that is meaningful to simulation software. Frost et al. [2] developed methods to transfer DXF files (a standard CAD format) into the SMARTFIRE computational fluid dynamics fire simulation software. Similarly, the Simulex evacuation model [3] allows the direct import of CAD data through DXF files. However, these methods generally require that the original files be manually 'cleaned' of data that would otherwise be incorrectly interpreted by the intended simulation software. The limitations of formats such as DXF mean that only basic geometric primitives can be derived from the files because more detailed definitions of objects cannot be included [4].

Overcoming the current limitations of data exchange between CAD and fire simulation software requires a much richer description of buildings than traditional CAD systems can provide. Previous work by Mowrer and Williamson [5] identified object-orientation, the association of attributes with objects and the ability to extract attributes from a CAD-developed drawing database as the key features required by CAD systems to permit integration with fire simulation software. This article demonstrates how the IFC building product model provides the necessary richness and in particular the ability to exchange its architectural data with a commonly available zone fire simulation tool.

It is not the intention of a building model or the exchange process to take away the 'engineering' portion of an analysis but to try and allow the fire engineer to focus on the design issues rather than spending time setting up a simulation tool. Electronic data exchange software does not mean that fire simulation tools become 'black-boxes' to be used by those unfamiliar with their appropriateness and limitations.

BUILDING PRODUCT MODELS

General Description

A 'building product model' is a collection of 'entities' that represent building elements, their relationship, and their properties. Entities may be physical objects, such as walls, doors, windows, etc. or more conceptual entities, such as zones, processes, or contractual details. Properties associated with entities can be divided into several categories [6], for instance, thermophysical (such as density, thermal conductivity, etc.), derived properties (for example the range of operation of a device), regulatory (the fire resistance rating of an element), or descriptive (color, shape, style, etc.).

Building design and operation can be thought of as being split into a number of specific domains including architecture, structural engineering, environmental engineering, building services, and facilities management. Each domain has its own specialized needs but in many instances there will be parameters that are common across two or more domains. Typically these common parameters are the building geometry and topology, the materials and components used in the construction, and the location of the structure within the broad environment. The ability to share this information in an electronic form should lead to more effective building design, construction, and management. The potential benefits of using building product models for fire engineering have been discussed previously [7].

IFC BUILDING PRODUCT MODEL

Description

The Industry Foundation Classes building product model or 'IFC model' began development around 1996 and is an extension of a number of earlier related projects, such as COMBINE [8] which aimed to integrate energy efficiency software tools for buildings. The version of the IFC model used in this article is 2× Edition 2, with corrections published in Addendum 1 [9,10] and is referred to as 'IFC 2×2' hereafter.

The IFC building product model supports the generation and sharing of project data through the complete building lifecycle amongst a diverse range of domains. The model is organized into five layers [10] which consist of:

- Resource layer – the lowest level layer which consists of general purpose concepts or objects that are independent of application which rely on other items in the model for their existence.

- Kernel – provides generalized concepts and determines the model structure and decomposition.
- Core extensions – specialization of concepts defined in the kernel.
- Interoperability layer – describes concepts common to two or more domains.
- Domain layer – the highest level layer that provides details for specific domains.

The model does not attempt to include every aspect of a building as this would likely be too complex and take too long to develop, instead entity types are described at a relatively generic level. The limited scope of the IFC model is addressed by the use of 'property set definitions' which allow for the extension of these generic level entities. As a result, the IFC model is being implemented and enhanced in many areas of the construction industry. Examples include concrete structures [11]; heating, ventilation, and air conditioning systems [12]; timber buildings [13]; and concrete slab bridges [14].

Structure

The contents of the IFC model are identified with a specific convention which is followed in this article and is indicated hereafter in italics text. Entities are prefixed with *Ifc*; property sets prefixed with *Pset* and enumerated lists of entities or properties with the addition of *Enum*.

The IFC building product model is essentially created as a tree structure with a top level 'parent' entity giving access to child entities below. However, the object-oriented design of the IFC model includes the ability for inheritance amongst entities and allows complex relationships to exist between entities. The version 2×2 release of the IFC model contains over 600 different entities, almost 300 property sets plus another 300 supporting data types. Abstract entities are used to describe collections of related building elements with specific elements defined as sub-types of the generic abstract entity. Entities have associated attributes which are either required or optional as specified by the IFC model.

The IFC building product model is formulated using the EXPRESS data modeling language as defined by the standard for the exchange of product model data (STEP) ISO 10303: Part 11 [15]. The structure of the model can be viewed using the EXPRESS-G notation which is a graphical notation for identifying classes, attributes, and relationships developed within the STEP standard.

Figure 1 for example uses an EXPRESS-G diagram to show the *IfcSpace* entity and its relationship with the *IfcRelSpaceBoundary* entity and other

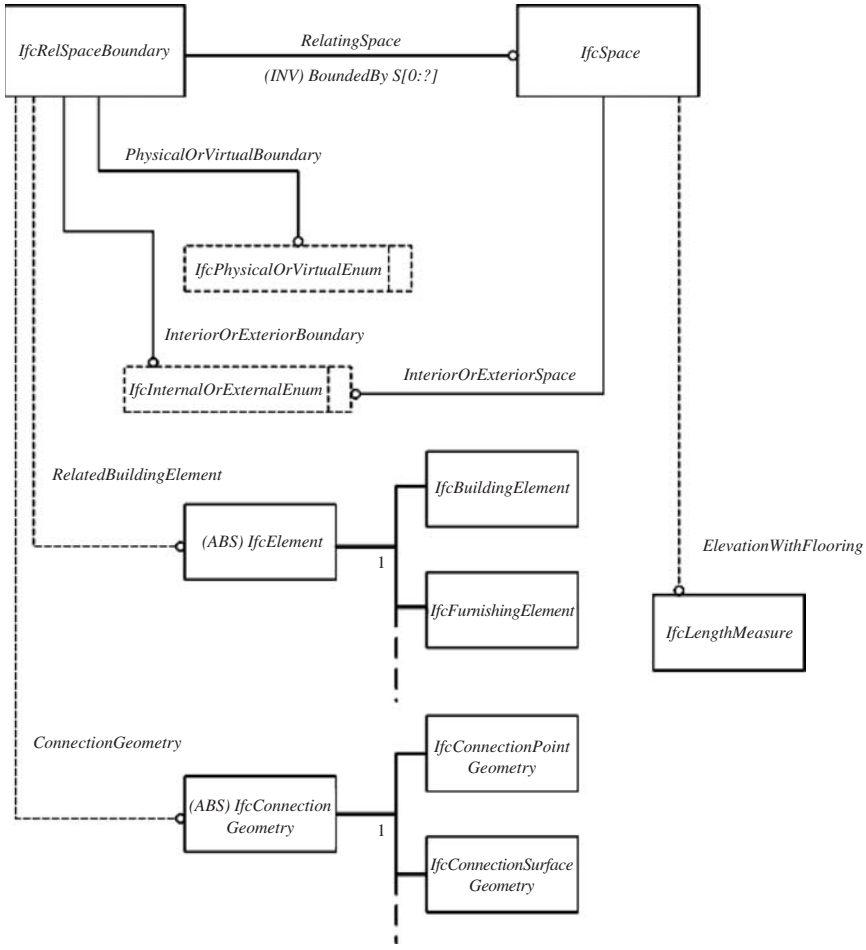

Figure 1. *EXPRESS-G representation of the IfcSpace entity and related entities, adapted from [9].*

connected entities. An *IfcRelSpaceBoundary* entity has an associated *IfcSpace* entity connected by the *RelatingSpace* property and inversely an *IfcSpace* can be *BoundedBy* a set of zero or more *IfcRelSpaceBoundary* entities (denoted by the (INV) and S[0:?] notation, respectively). An *IfcSpace* entity can optionally (as indicated by the dotted line) have an *ElevationWithFlooring* property specified as an *IfcLengthMeasure* entity. The *IfcSpace* must also have the *InteriorOrExteriorSpace* property designated which is specified by the *InteriorOrExteriorEnum* enumeration. Abstract entities are denoted with the (ABS) prefix and these entities cannot exist in themselves but only as one

of their subtypes. For clarity, only examples of the available subtypes of the *IfcElement* and *IfcConnectionGeometry* entities are shown in Figure 1. The *IfcSpace* entity also inherits properties from higher level entities, such as the *IfcObject*, *IfcProduct*, etc. entities although these inheritances are not shown in Figure 1.

File Format and Generation

IFC files are primarily exchanged using the STEP encoding specifications given in ISO 10303: Part 21 [16] although an extensible markup language (XML) version of the IFC model has also been made available [17]. The STEP encoding is an electronic data interchange standard which has the aim of completely representing a product over its whole lifetime in a neutral format. The representation includes geometric data and non-geometric data, such as properties and costs [18]. A physical STEP file uses only ASCII characters and is a human readable format although software tools are often used to parse a file.

In this work IFC 2×2 conforming STEP files were generated using the commercially available ArchiCAD software [19] from Graphisoft with an IFC 2×2 export module included, also written by Graphisoft. ArchiCAD is an integrated architectural design tool that represents buildings using a 'virtual building' model. Building elements, such as slabs, walls, doors, etc are used to construct a three-dimensional model of a building. ArchiCAD building elements are intelligent objects that have their own associated properties and behavior. The object-oriented nature of ArchiCAD means that the building model is more than the two-dimensional line representation of a building that is common with traditional CAD systems. The building model allows the user to obtain additional information about the building, such as bills of materials. ArchiCAD can be used to view a building model not only as plans, elevations, and sections but also can generate perspective and virtual reality presentations of the building. Since the building model is stored in a central database, any changes are automatically reflected in all the views.

EXCHANGE PROCESS

IfcSTEP Parser Software Development

The core exchange software developed in this research, referred to herewith as the IfcSTEP Parser, is primarily written in C++ using the Microsoft.NET environment. The current version of the IfcSTEP Parser

software was built on earlier work in which the previous Release 2 of the IFC model was accessed through its XML encoding [20]. The object-oriented nature of the C++ language enables a set of classes to be developed for selected IFC entities and additional classes can be added as the functionality of the exchange software increases. The reading of STEP files was accomplished by incorporating the IFC SECOM Server [21] into the IfcSTEP Parser program. The IFC SECOM Server tool includes routines to interrogate and extract elements from a specified STEP file and it can be interfaced to Visual Basic or C++ source code.

The complex nature of the IFC model means that it is impractical to completely develop the IfcSTEP Parser software to process every available entity in the model. Instead, the initial software implementation focuses on the interpretation of the geometry of the spaces, connections between spaces, and the basic material properties of geometrical elements, such as walls and doors.

In order to interpret a STEP file, the IfcSTEP Parser software is written to navigate the IFC model tree. Figure 2 shows how the software identifies the top level *IfcProject* entity and then moves down through the tree extracting the *IfcSite*, *IfcBuilding*, *IfcBuildingStorey*, *IfcSpace* entities and so on. C++ classes defined in the software interpret the EXPRESS relationships associated with each entity, such as those illustrated in Figure 1 for the *IfcSpace* entity. Figure 2 shows the IfcSTEP Parser navigation process through 27 of the IFC model entities. A dotted arrow is shown where additional entities are accessed deeper into the tree. The version of the IfcSTEP Parser developed for this research is currently able to parse 46 of the IFC model entities sufficient to extract the relevant architectural information required.

The C++ classes interpret the IFC model into a set of intermediate data structures from which interfaces to specific fire simulation tools can then be developed (Figure 3). Each simulation tool interface is developed separate from the core IfcSTEP Parser and separate from interfaces to other simulation tools. The advantage of this intermediate approach is that as the IFC model changes only then does the IfcSTEP Parser software need updating. Similarly, if the target fire simulation software tool is modified then only the interface module needs to be adapted.

The execution of an IfcSTEP Parser interface generates a log file which allows the user to investigate how the interface converts the intermediate internal data structures into the requirements of the target fire simulation software input file. This log file can be used to track inconsistencies in the exchange process and make subsequent changes to the desired fire simulation software scenario before running any simulations.

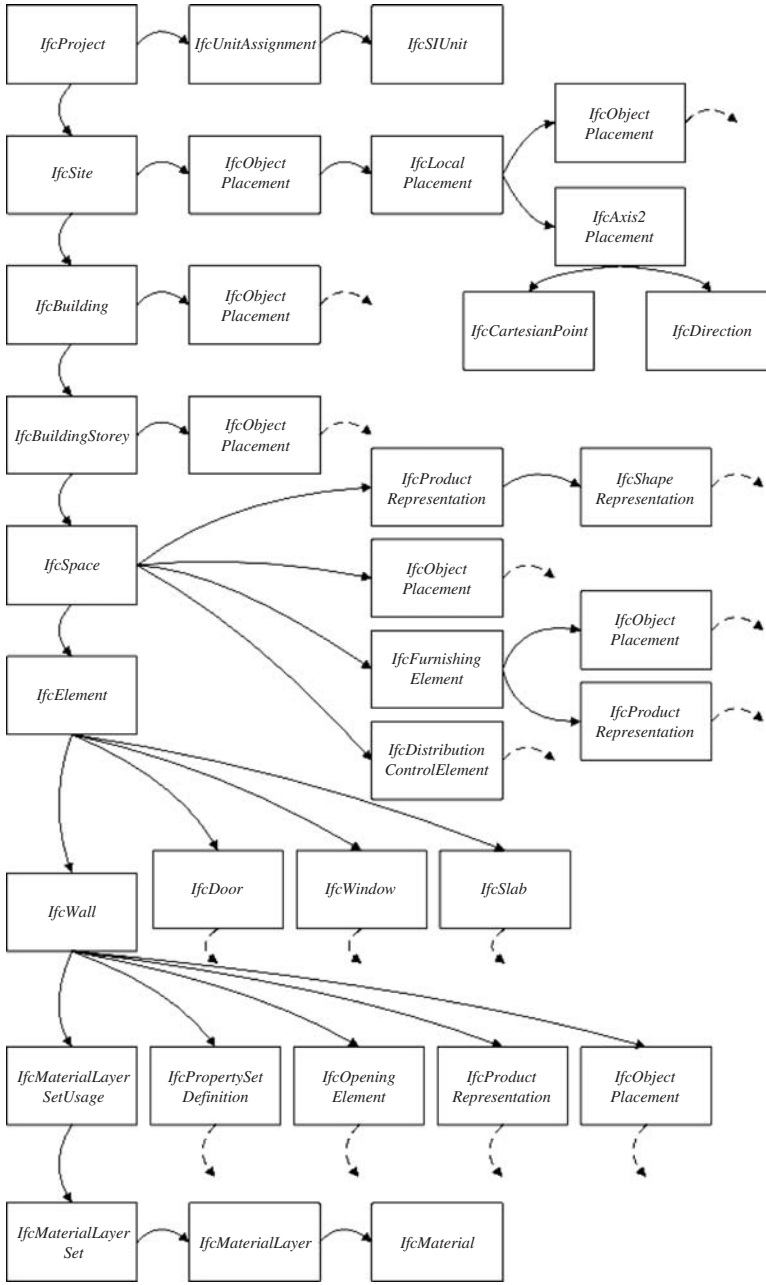

Figure 2. *IfcSTEP Parser extraction of IFC model entities (dotted arrows lead to further entities).*

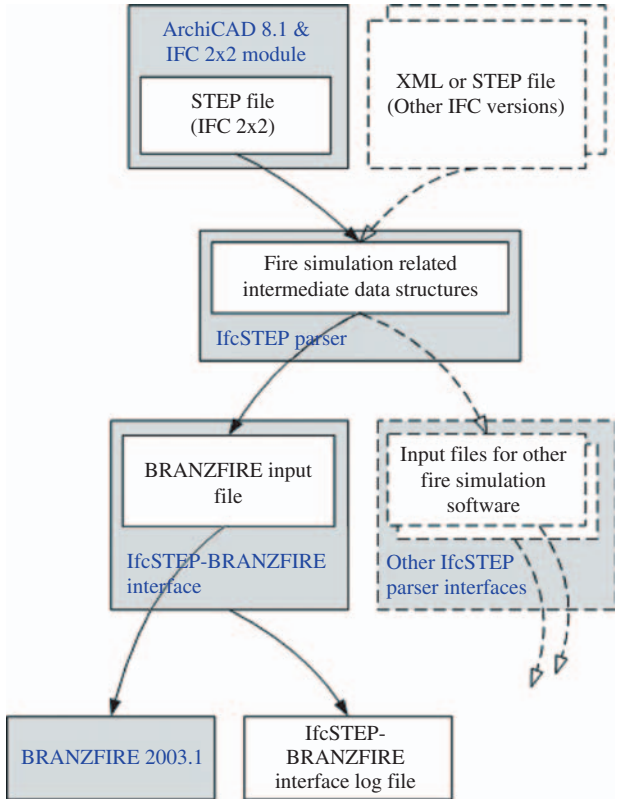

Figure 3. *IfcSTEP Parser data exchange process (dotted items not currently implemented). (The color version of this figure is available online.)*

BRANZFIRE Fire Simulation Software

In order to demonstrate the ability to exchange IFC model data using the IfcSTEP Parser, BRANZFIRE version 2003.1 was selected as the target fire simulation software. An earlier mapping exercise using IFC 2×2 showed that IFC 2×2 can lend itself to interpretation by fire simulation models, such as BRANZFIRE [22] and so an IfcSTEP-BRANZFIRE Parser interface was written which transformed the intermediate data structures into a BRANZFIRE '`.mod`' input file.

BRANZFIRE [23,24] is a widely available multi-compartment zone model. It can simulate the movement of smoke between up to 10 interconnected spaces. Fires are specified by a rate of heat release curve or using a built-in fire spread model in the case of room linings. The model also has the ability to incorporate sprinkler and smoke detector

activation, the breaking of window glass [25], and the effects of mechanical fans. The underlying conservation equations and relationships that are used by most zone fire simulation software including BRANZFIRE are described by Quintiere [26]. The fire plume transports combustion products from a cool lower zone to a hot upper zone and the physical conditions within these zones are considered vertically and horizontally uniform. Fire gases flow through vertical and horizontal openings (often referred to as vents in fire simulation software, such as BRANZFIRE) in compartment boundaries into neighboring spaces or the 'outside'.

Project Settings

Since BRANZFIRE requires geometrical dimensions in meters for building elements and millimeters for items, such as material thicknesses it is necessary to substantiate the units being used in the IFC file. The *IfcProject* entity is used to obtain the units being used through the *IfcUnitAssignment* entity (Figure 2). The *IfcProject* entity can also be used to extract the project title and add that into the BRANZFIRE input file.

Rooms

Zone fire simulation software tools often make a number of simplifications regarding the geometry of the spaces being modeled. For example, it is typical but not necessarily always the case to assume that spaces have a rectangular footprint and that ceilings are smooth and horizontal. Some zone fire simulation software tools have a limited number of allowable spaces and/or a limited number of connections, such as doors between spaces. In the zone fire simulation software these connections might be assumed to be in the center of the parent wall rather than at its actual location. Assumptions such as these need to be accounted for when exchanging the IFC model description of a real building to the idealized view required by zone fire simulation software.

In BRANZFIRE, the basic building topology is specified by a collection of interconnected rooms. The structure of the IFC model allows each individual room to be identified through the *IfcSpace* entity. The IfcSTEP-BRANZFIRE Parser maps *IfcSpace* entities to the BRANZFIRE rooms with the *IfcSpace* entity label used to identify the name of the room in the fire simulation input file. The height of a space is obtained from the height of the bounding box of the *IfcSpace* entity. BRANZFIRE requires rectangular-shaped rooms and thus spaces with complex footprint geometries will not convert well. Spaces are normally bounded by a wall,

open to a neighboring space or open to the outside and the mapping of these IFC entities are described here.

Walls

As illustrated in Figure 1, IFC model *IfcSpace* entities are associated with one or more *IfcElement* entities using the *IfcRelSpaceBoundary* entity. These bounding entities can be *IfcWall* entities or some other appropriate entity. The IfcSTEP-BRANZFIRE Parser identifies the *IfcWall* entities bounding a space and determines its position and dimensions through the *IfcObjectPlacement* and *IfcProductRepresentation* entities, respectively (Figure 2).

Walls may be constructed of one or more layers each of different materials and thicknesses. Wall layers are obtained through the *IfcMaterialLayerSetUsage* entity that is referenced by an *IfcWall* entity. Each layer thickness is obtained from the *LayerThickness* property associated with the *IfcMaterialLayer* entity and each layer's material type is taken from the *Description* property of the *IfcMaterial* entity as illustrated in Figure 2.

Single material layer and multiple material layer walls can be defined in ArchiCAD. The IfcSTEP-BRANZFIRE Parser maps the *IfcMaterialLayerSetUsage* entity associated with each *IfcWall* entity to BRANZFIRE wall materials. The outer most facing *IfcMaterialLayer* entity is assigned to the BRANZFIRE wall lining material and thickness. The next *IfcMaterialLayer* entity in the *IfcMaterialLayerSetUsage* entity list is assigned to the BRANZFIRE substrate and subsequent *IfcMaterialLayer* entities are ignored since BRANZFIRE only handles walls defined with one or two layers.

There are several general issues associated with the translation of bounding entities from the IFC model to BRANZFIRE. An *IfcSpace* generally has more than one bounding wall and these are not necessarily defined as having similar construction. BRANZFIRE assumes that a room has a single type of wall surrounding it and this can lead to inconsistencies when mapping from the IFC model. Where no wall bounding a space has a description then the equivalent BRANZFIRE room walls will not be assigned material properties. Where a set of walls bounding an *IfcSpace* has a mix of *IfcMaterialLayer* entities, the BRANZFIRE wall materials will be assigned the properties of the 'last' *IfcWall* entity in the set. The 'last' bounding *IfcWall* entity will depend on how ArchiCAD outputs its IFC file and thus cannot be necessarily predicted or assumed to be the same each time the file is saved. Furthermore, where multiple layer walls include air gaps it is possible that one of the BRANZFIRE wall lining properties

(generally the substrate) will be designated as air rather than as a solid material. The user needs to be aware of these issues and check the BRANZFIRE scenario setup, such as by viewing the IfcSTEP-BRANZFIRE Parser log file, before continuing with any simulations.

Openings

Space boundaries may contain openings, such as doorways, windows, and holes that connect to neighboring spaces or the outside. Doorways and windows may have the opening completely filled by the door leaf or the pane of glass. Alternatively, there may be an open path through the opening if the door leaf or window is partially or fully open.

The structure of the IFC model permits the association of an *IfcOpeningElement* with the *IfcElement* abstract entity through the *IfcRelVoidsElement* relationship. This structure allows for the identification of the parent entity for a particular opening (e.g., in which wall does one find a particular door) and also the identification of a group of openings for a specific entity (e.g., which doors form openings in a given wall).

As shown in Figure 2, the IfcSTEP Parser program identifies all the *IfcDoor* and *IfcWindow* entities associated with each room during the parsing of the *IfcSpace* entities. Similar to the *IfcWall* entity, the position and dimensions of these are determined with the *IfcObjectPlacement* and *IfcProductRepresentation* entities, respectively. During the parsing of *IfcWall* entities any *IfcOpeningElement* entities that do not have a parent *IfcDoor* or *IfcWindow* entity are determined. The *IfcOpeningElement* entity associated with *IfcDoor* and *IfcWindow* entities are not processed to avoid duplication of connections. As each *IfcSpace* entity is processed, each vent identified by the IfcSTEP-BRANZFIRE Parser is added to global lists of connection types. These lists are used to create the connection map between spaces as described here.

Connection Map

Once each *IfcSpace* entity has been processed the IfcSTEP-BRANZFIRE Parser builds a connection map that identifies the connecting vents between rooms. The method in which each space is processed results in connecting elements, such as internal doors, appearing twice in the list of connections: once as a connection between space A and space B; and then again as a connection between space B and space A. Thus multiple connection instances are removed from the processing list before the final connection routes are determined. Connections with only one associated space are assumed to connect with the ‘outside’. This is the case for most windows and

for doors located on external walls. If a user has only selected a portion of a larger building the connections to spaces not in the selected building portion will be treated as outside connections.

In the current version of the IfcSTEP-BRANZFIRE Parser, all connections are treated as fully open. In future developments of the parsing tool it should be possible to close off connections and then provide properties for the closing element, such as window systems since the IFC model includes properties for glazing through the *Pset_DoorWindowGlazingType* property set.

Floors and Ceilings

BRANZFIRE generally requires that rooms be given a floor and ceiling material type and thickness similar to walls. In a multistory building it is possible that a single building element may represent both a ceiling and a floor and this needs to be considered when interpreting a building model.

The IFC 2×2 model does not explicitly contain ceiling or floor entities and one simple approach is to represent the construction of floors and ceilings using *IfcSlab* entities. The IFC model does define an *IfcRoof* entity which is an assembly that groups together related entities that make up the roof, such as slabs (*IfcSlab*), rafters and purlins (using the *IfcBeam* entity), or other included roofs, such as dormers which would be also specified with an *IfcRoof* entity. Nevertheless, properties of floors and ceilings can be allocated using property sets included in the IFC model which are associated with the *IfcCovering* entity. The *IfcCovering* entity is used to cover some part of another element such as an *IfcSlab* entity.

ArchiCAD slab elements can be used to represent the construction of floors and ceilings. Similar to walls, slabs can be allocated material composites with layers of different material types and thicknesses. The IfcSTEP-BRANZFIRE Parser needs to determine what a slab represents in an IFC file so that it can correctly construct the BRANZFIRE input file. The IfcSTEP-BRANZFIRE Parser uses the position of the slab compared with the dimensions of the associated *IfcSpace* entity. Slabs located at the highest vertical dimension of a space are assumed to be a roof slab and slabs located at the lowest vertical dimension of a space are assumed to be a floor slab. However, this method will not be able to identify elements such as suspended ceilings that may be contained with a space.

An alternative approach is for the user to define slabs on particular ArchiCAD drawing layers and currently the IfcSTEP-BRANZFIRE Parser will interrogate the *PSet_Draughting* property set associated with an *IfcSlab* entity and assume that slabs on a layer containing the word ‘Ceiling’ is a ceiling and slabs on a layer containing the word ‘Floor’ is a floor. Thus the

user needs to ensure that specific drawing layers exist in their CAD environment and that slabs are placed on those specific layers. Although this approach is somewhat more flexible than the previous method it contravenes an overall philosophy of providing building representations in a form that does not require any user-specific configurations.

Complex floor or ceiling geometries will not convert well in the current version of the IfcSTEP-BRANZFIRE Parser and better algorithms for the identification of floors and ceilings need to be developed.

TEST CASE BUILDINGS

Two simple test case buildings were created in order to verify the effectiveness of the translation process. In each case a virtual building model was setup in ArchiCAD 8.1 and was saved as an IFC 2×2 file directly from ArchiCAD without any changes or 'cleaning' necessary. The IFC file was then processed by the IfcSTEP-BRANZFIRE interface and the resultant. `.mod` file was input into BRANZFIRE. The `.mod` file generated by the IfcSTEP-BRANZFIRE interface was written for BRANZFIRE 2002.7 for backwards compatibility with previous releases of BRANZFIRE.

Room/Corner Test

A representation of the ISO 9705 [27] room was used as initial test case building. This simple structure was used to check that the basic room geometry and material properties could be extracted from an IFC file. Figure 4(a) and Figure 4(b) show the ArchiCAD plan and an isometric view respectively of the ISO room with the internal dimensions and door dimensions as specified by ISO 9705. The walls of the test room were set in ArchiCAD to 100 mm thick concrete since ISO 9705 only requires that the room be constructed of non-combustible material with a minimum thickness of 20 mm and a density of 500–800 kg/m^3.

ArchiCAD slab elements were used to represent the floor and ceiling of the room. The slab elements were located at highest and lowest vertical room dimensions and also placed on the 'Floors: Slabs' and 'Roof: Ceiling Slabs' ArchiCAD default drawing layers, respectively. This allowed the identification of the room floor and ceiling by either of the two methods described earlier. The properties of the ceiling slab were also specified as 100 mm thick concrete while the floor slab was set to a composite of 2 mm plaster over 100 mm concrete so as to test the ability of the IfcSTEP Parser to identify entities consisting of layers of materials.

Walls can be created in ArchiCAD in such a way as to represent the boundaries of a space. However, these will not explicitly define the

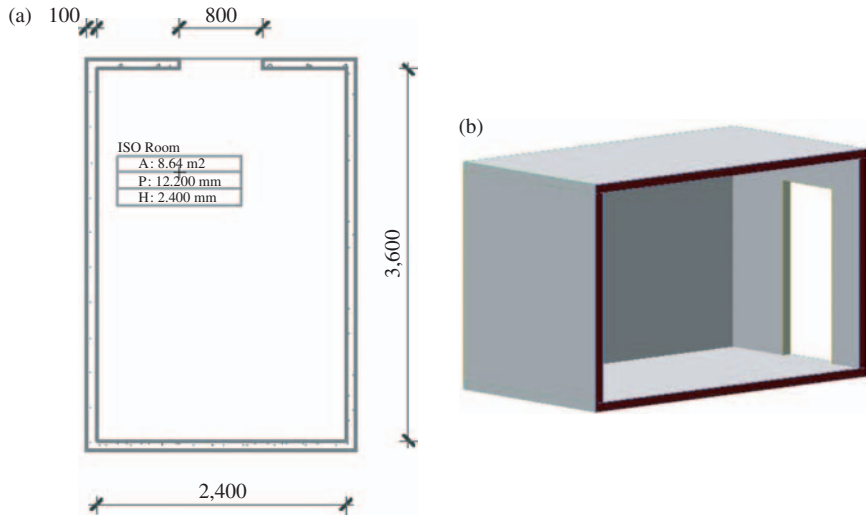

Figure 4. *ISO 9705 test room: (a) plan elevation and (b) cut-through isometric view. (The color version of this figure is available online.)*

contained space. Each space in a building representation must be assigned an *IfcSpace* entity for the IfcSTEP Parser to correctly interpret an IFC file. In ArchiCAD the 'zone tool' is used to identify separate areas in a building and these areas are exported as *IfcSpace* entities in an IFC file. The ArchiCAD zone tool also allows the user to specify the characteristics of the space and also automatically presents information, such as the net floor area, perimeter, and height on the drawing plans as illustrated in Figure 4(a).

Figure 5 shows the BRANZFIRE room design windows which are utilized by the user to define room parameters after importing the resultant `.mod` file into BRANZFIRE. Figure 5(a) shows that the correct room dimensions have been determined and Figure 5(b) shows the door (wall vent) has also been properly extracted. The wall material and thickness is shown in Figure 5(c) and the composite floor has been correctly interpreted in Figure 5(d). Note that the material names are those specified by the ArchiCAD material list and not from the BRANZFIRE database.

Cardington House

Figure 6 shows the ground floor of a residential house based on the geometry of a two-story building which has been used for various fire engineering-related studies [28]. The end elevation and isometric view are automatically generated from ArchiCAD's virtual building model. The second story of the house has been omitted because the current version of the

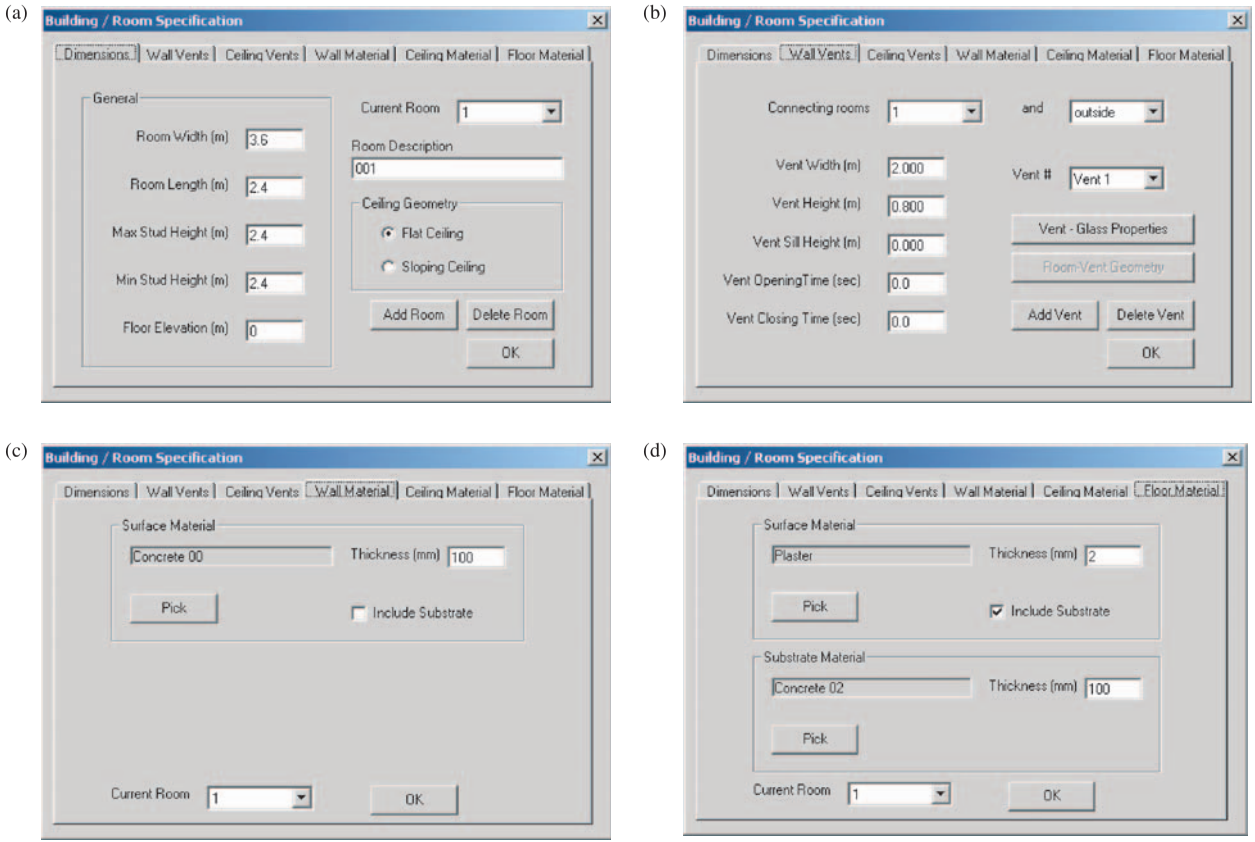

Figure 5. *BRANZFIRE room design windows for the ISO 9705 test building: (a) room dimensions; (b) wall vent; (c) wall material; and (d) floor material. (The color version of this figure is available online.)*

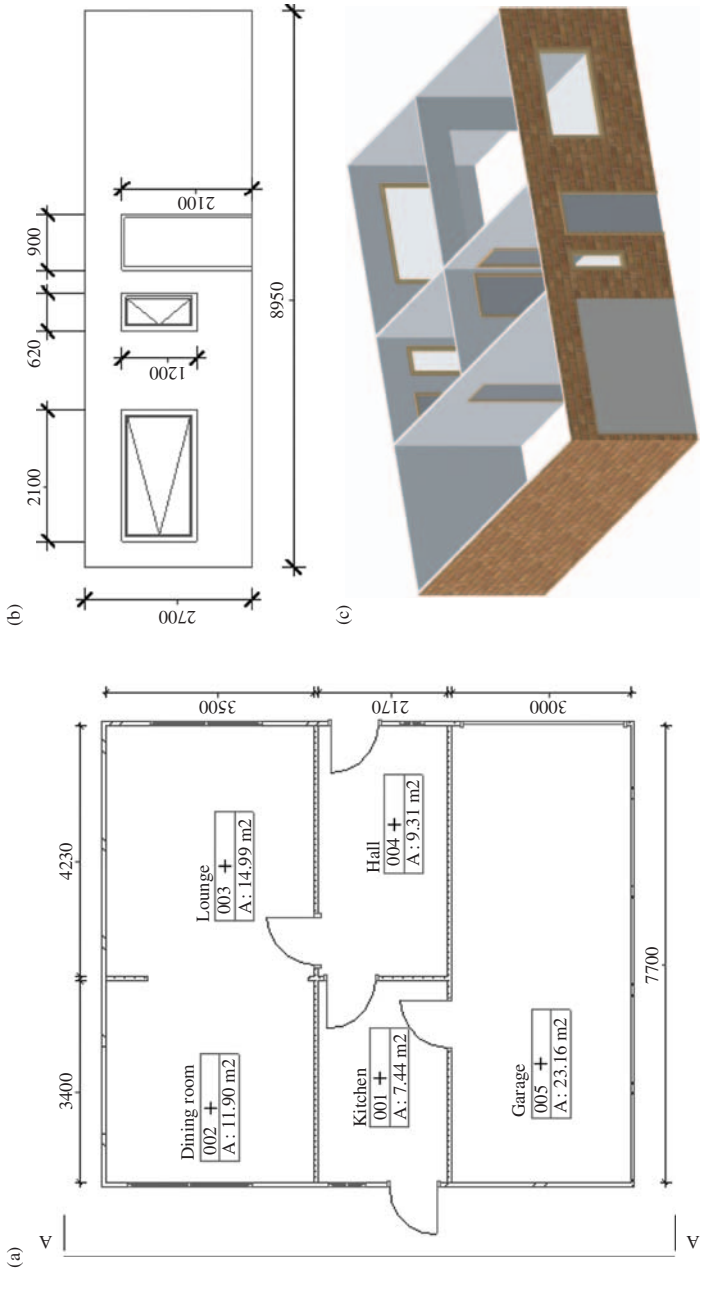

Figure 6. *Ground floor of the two-story residential building: (a) plan elevation; (b) end elevation; section A-A; and (c) isometric view. (The color version of this figure is available online.)*

IfcSTEP-BRANZFIRE Parser does not process *IfcStairs* entities. The stairs have therefore not been included in the hallway of the virtual building model.

The purpose of setting up this building is to verify the topological and geometrical translation process within the IfcSTEP-BRANZFIRE Parser. The geometry of the spaces is kept simple with rooms represented by rectangular footprints. For the purposes of the verification process, no roof or floor slabs are included in the virtual building model and material properties for building elements are minimal.

The house has several interconnected rooms with one or more connections. Connections are either internal standard-size doors or an opening in a wall plus doors and windows on the external boundaries including the large garage door.

Figure 7 shows the log file output from the IfcSTEP-BRANZFIRE interface and it can be seen that the dimensions of rooms, doors, windows, and openings have all been properly interpreted. Furthermore, the connections between rooms have been correctly identified. It should be noted that in this example it has been assumed that all vents are fully open since the purpose of the exercise is to specifically test the connection mapping functionality of the IfcSTEP-BRANZFIRE Parser. The ability to assign descriptive labels to building elements has not been utilized in ArchiCAD with the result being door, window, and opening names are listed as '$' in the log file.

Material properties for the walls have been determined where 'Double 1:8' is the style of brickwork specified for the outside walls. Note how some spaces have their walls set to 'Plaster' while others are 'Double 1:8'. This is a result of rooms having walls assigned different properties and the IfcSTEP-BRANZFIRE Parser using the 'last' wall in the IFC file to define the wall properties for the whole space.

DISCUSSION

Efficiency Gains

The objective of using a standardized model of a building to obtain data for simulation software is an increased level of efficiency and whether this has been achieved is a reasonable question. On the positive side, this work shows that it is possible to obtain geometrical and topological data from rooms with a simple shape. However, where rooms are complex in shape or in the properties associated with the building elements, the user still is likely to have to intervene before executing their simulations. This raises the issue of whether the effort to create a virtual building model in a tool, such as

```
Log file for IfcSTEP-BRANZFIRE Interface
version 0.1 (Jan 20 2005)
Started at 16:46:49 on Thursday 20 January 2005

**********************************************

IfcSTEP-BRANZFIRE Interface
---------------------------------------

(c) Michael Spearpoint
University of Canterbury
New Zealand

Interface version 0.1 (Jan 20 2005)
IfcSTEP Parser version 0.1
BRANZFIRE version 2002.7

**********************************************


Space '001' created as BRANZFIRE room 1
    width = 2.17 m  depth = 3.40 m  height = 2.70 m
    Wall lining thickness set to '70.00'
    Wall lining material set to 'Plaster'

Space '002' created as BRANZFIRE room 2
    width = 3.40 m  depth = 3.50 m  height = 2.70 m
    Wall lining thickness set to '70.00'
    Wall lining material set to 'Plaster'

Space '003' created as BRANZFIRE room 3
    width = 3.50 m  depth = 4.23 m  height = 2.70 m
    Wall lining thickness set to '70.00'
    Wall lining material set to 'Double 1:8'

Space '004' created as BRANZFIRE room 4
    width = 2.17 m  depth = 4.23 m  height = 2.70 m
    Wall lining thickness set to '70.00'
    Wall lining material set to 'Plaster'

Space '005' created as BRANZFIRE room 5
    width = 3.00 m  depth = 7.70 m  height = 2.70 m
    Wall lining thickness set to '70.00'
    Wall lining material set to 'Double 1:8'

Door '$' created as BRANZFIRE vent
    width = 0.90 m  sill = 0.00 m  soffit = 2.10 m
    connecting from room 1 (001) to room 5 (005)

Door '$' created as BRANZFIRE vent
    width = 0.90 m  sill = 0.00 m  soffit = 2.10 m
    connecting from room 1 (001) to room 4 (004)

Door '$' created as BRANZFIRE vent
    width = 0.90 m  sill = 0.00 m  soffit = 2.10 m
    connecting from room 1 (001) to outside

Door '$' created as BRANZFIRE vent
    width = 0.90 m  sill = 0.00 m  soffit = 2.10 m
    connecting from room 3 (003) to room 4 (004)

Door '$' created as BRANZFIRE vent
    width = 0.90 m  sill = 0.00 m  soffit = 2.10 m
    connecting from room 4 (004) to outside

Door '$' created as BRANZFIRE vent
    width = 2.80 m  sill = 0.00 m  soffit = 2.10 m
    connecting from room 5 (005) to outside

Window '$' created as BRANZFIRE vent
    width = 0.62 m  sill = 0.90 m  soffit = 2.10 m
    connecting from room 1 (001) to outside

Window '$' created as BRANZFIRE vent
    width = 2.10 m  sill = 0.90 m  soffit = 2.10 m
    connecting from room 2 (002) to outside

Window '$' created as BRANZFIRE vent
    width = 1.90 m  sill = 0.90 m  soffit = 2.10 m
    connecting from room 3 (003) to outside

Window '$' created as BRANZFIRE vent
    width = 0.42 m  sill = 0.90 m  soffit = 2.10 m
    connecting from room 4 (004) to outside

Opening '$' created as BRANZFIRE vent
    width = 2.70 m  sill = 0.00 m  soffit = 2.10 m
    connecting from room 2 (002) to room 3 (003)
```

Figure 7. *IfcSTEP-BRANZFIRE log file output (in two column format).*

ArchiCAD is worthwhile compared to directly setting up scenarios in the target fire simulation tool.

Fire simulations conducted by a fire engineer invariably require some form of graphical record of the building being analyzed. An ArchiCAD virtual building model provides such a graphical representation that can be used to communicate with others in a written or oral report. Furthermore, where the building model has already been created by someone else, such as the architect, it still saves the manual re-entry of basic geometrical information even if some further manipulation is necessary. The benefits

of using a common electronic building description become greater when multiple simulation tools are considered. Changes to the building model are reflected in all subsequent runs of any simulation tool rather than each change being individually incorporated into each simulation tool.

Future Work

The complexity and scope of the IFC model means that there are still considerable enhancements that can be added to the IfcSTEP Parser software. The current version can only handle $<10\%$ of the entities available in IFC 2×2. However, it should be recognized that there may be many of the IFC entities which will have no direct use in fire simulation software.

In the longer term it is envisaged that IFC building product model exchange with other commonly used fire simulation software, such as computational fluid dynamics, people movement, and structural fire analysis tools, will be created. There is also the need to continue testing the capabilities of the parsing algorithms using additional buildings. Complex buildings, such as multi-story structures, those with non-rectangular footprints, and composite material properties would all challenge the current version of IfcSTEP Parser and would almost certainly require modifications to the program. Further testing could be undertaken that consists of blind trials in which the IfcSTEP Parser representation is compared with that obtained by a fire engineer's interpretation for one or more scenarios. It is likely that the outcome would not be exactly the same but variations would need to be explainable and not result in significantly different conclusions.

Finally, the current focus of the IfcSTEP Parser development has been to demonstrate the viability of using the IFC model as an electronic description of buildings. As a result, very little effort has been undertaken to develop a user-friendly interface. The current version executes as a simple command line program, but the core source code for the IfcSTEP Parser is written in a way which enables a suitable front-end to be added reasonably easily.

CONCLUSIONS

This article demonstrates how the IFC building product model can be used to transfer electronic building descriptions between a commercially available CAD system and the BRANZFIRE fire simulation tool. The transfer process is not without its limitations which stem from a combination of the current content of the IFC model, the implementation of the IFC model in a CAD package, the ability to populate and extract

entities from an IFC file, and the mapping of those entities to a simplified form suitable for zone fire simulation representation. Further work is required to enhance the capabilities of the exchange software including expanding the scope of IFC entities that can be processed, improving the mapping of the IFC entities to the requirements of zone fire simulation software, and enlarging the range of fire simulation software that can interface to the software.

ACKNOWLEDGMENTS

The author would like to thank Colleen Wade of BRANZ for the use of the BRANZFIRE fire simulation model and the specification of the input file format. The author is supported through funding from the New Zealand Fire Service Commission.

REFERENCES

1. Massa, R.J. and Cappuccio, J.A., "Architectural CAD-Based Topological Building Models for Fire Hazard Analysis," In: Orlando, F.L., Lund, D.P. and Angell, E.A. eds, Proc. International Conference on Fire Research and Engineering (ICFRE), 10–15 Sept 1995, Society of Fire Protection Engineers, Boston, MA, 1995, pp. 327–331.
2. Frost, I., Patel, M.K., Galea, E.R., Rymacrzyk, P. and Mawhinney, R.N., "A Semi-Automated Approach to Cad Input into Field Based Fire Modelling Tools," In: Proc. Interflam 2001, Edinburgh, Scotland, 2001, pp. 1421–1426.
3. Thompson, P.A. and Marchant, E.W., "A Computer Model for the Evacuation of Large Building Populations," Fire Safety Journal, Vol. 24, No. 2, 1995, pp. 131–148.
4. Anon, "Beyond CAD," CAD User AEC Magazine, Vol. 16, No. 07, July/Aug 2003.
5. Mowrer, F.W. and Williamson, R.B., "Room Fire Modeling within a Computer-Aided Design Framework," In: Wakumatsu, T., et al., eds, International Association for Fire Safety Science, 2nd International Symposium, June 13–17, Tokyo, Japan, Hemisphere Publishing Co., New York, 1989, pp. 453–462.
6. Spearpoint, M.J., "Properties for Fire Engineering Design in New Zealand and the IFC Building Product Model," In: Amor, R. ed., Proc. 20th International Conference on Information Technology in Construction, CIB Publication 284, ISBN 0-908689-71-3, University of Auckland, New Zealand, 2003, pp. 333–340.
7. Spearpoint, M.J., "The Potential Impact of Building Product Models on Fire Protection Engineering," Fire Protection Engineering, Issue 19, 2003, pp. 42–48.
8. Augenbroe, G.L.M., "An Overview of the COMBINE Project," In: Proceedings of the First ECPPM Conference, Dresden, A. A. Balkema Publishers, Rotterdam, The Netherlands. 1994.
9. Adachi, Y., Forester, J., Hyvarinen, J., Karstila, K., Liebich, T. and Wix, J., Industry Foundation Classes IFC2× Edition 2, Modeling Support Group, International Alliance for Interoperability, 2003.
10. Liebich, T. ed., IFC 2× Edition 2 Model Implementation Guide, Version 1.6. Modeling Support Group, International Alliance for Interoperability, 2003.

11. Rönneblad, A., "Product Models for Concrete Structures," Licentiate Thesis 2003:22, ISSN: 1402–1757, Luleå University of Technology, Sweden, 2003.
12. Bazjanac, V. and Maile, T., "IFC HVAC Interface to EnergyPlus – A Case of Expanded Interoperability for Energy Simulation," SimBuild 2004, IBPSA-USA National Conference, Boulder, CO, August 4–6, 2004.
13. Osterrieder, P., Richter, S. and Fischer, M., "A Product Data Model for Design and Fabrication of Timber Buildings," In: Proc. World Conference on Timber Engineering WCTE2004, ISBN 951-758-444-X, RIL, Helsinki, Lahti, Finland, Vol. 1, June 2004, pp. 47–52.
14. Yabuki, N. and Shitani T., "An IFC-based Product Model for RC or PC Slab Bridges," In: Amor, R. ed., Proc. 20th International Conference on Information Technology in Construction, CIB Publication 284, ISBN 0-908689-71-3, University of Auckland, New Zealand, 2003.
15. ISO 10303-11:1994, "Industrial Automation Systems and Integration – Product Data Representation and Exchange – Part 11: Description Methods," The EXPRESS Language Reference Manual, 1994.
16. ISO 10303-21:2002, Industrial Automation Systems and Integration – Product Data Representation and Exchange – Part 21: Implementation Methods: Clear Text Encoding of the Exchange Structure, 2002.
17. Liebich T., XML Schema Language Binding of EXPRESS for ifcXML. MSG-01-001(Rev 4), International Alliance for Interoperability, 2001.
18. King, B.J. and Norman P.W., "A Step in the Right Direction," Professional Engineering, November 1992.
19. Anon, ArchiCAD 8 User Guide. Graphisoft, 2003.
20. Spearpoint, M.J., "Integrating the IFC Building Product Model with Zone Fire Simulation Software," In: Proc. International Conference on Building Fire Safety, QUT, Gardens Point Campus, Brisbane, Australia, 20–21 November 2003, pp. 56–66.
21. SECOM Co., Ltd., IFC SECOM Server. http://groups.yahoo.com/group/ifcsvr-users/files/IFCsvr300/.
22. Spearpoint, M., "Fire Engineering Properties in the IFC Building Product Model and Mapping to BRANZFIRE," International Journal on Engineering Performance-Based Fire Codes, Vol. 7, No. 3, 2005, pp. 134–147.
23. Wade, C.A. BRANZFIRE Technical Reference Guide, BRANZ Study Report 92 (revised). Building Research Association of New Zealand, Judgeford, Porirua City, New Zealand, 2003.
24. Wade, C.A., A User's Guide to BRANZFIRE 2003, Judgeford, Porirua City, New Zealand, Building Research Association of New Zealand, 2003.
25. Parry, R., Wade, C.A. and Spearpoint M.J., "Implementing a Glass Fracture Module in the BRANZFIRE Zone Model," Journal of Fire Protection Engineering, Vol. 13, No. 3, 2003, pp. 157–183.
26. Quintiere, J.G., Compartment Fire Modeling, Chapters 3–5, The SFPE Handbook of Fire Protection Engineering, 3rd edn, National Fire Protection Association, Quincy, MA, 2002, pp. 3-162–3-170.
27. ISO 9705:1993, Fire Tests on Building Materials and Structures – Part 33. Full-scale Room Test for Surface Products, 1993.
28. Spearpoint, M.J. and Smithies, J.N., "Practical Comparison of Smoke Detector Sensitivity Standards," In: Proc. 11th International Conference on Fire Detection, AUBE '99 Conference, Duisburg, Germany, 1999, pp. 576–587.