Implementation of IFC-based web server for collaborative building design between architects and structural engineers

Po-Han Chen\textsuperscript{a,*}, Lu Cui\textsuperscript{a}, Caiyun Wan\textsuperscript{a}, Qizhen Yang\textsuperscript{b}, Seng Kiong Ting\textsuperscript{a}, Robert L.K. Tiong\textsuperscript{a}

\textsuperscript{a}School of Civil and Environmental Engineering, Nanyang Technological University, Singapore 639798, Singapore
\textsuperscript{b}Singapore Institute of Manufacturing Technology, Singapore 638075, Singapore

Accepted 27 August 2004

Abstract

This paper presents the implementation of an Industry Foundation Class-based (IFC-based) information server for web-enabled collaborative building design between the architect and structural engineer. In this research, the Java 2 Platform, Enterprise Edition (J2EE) standard is employed to build the framework of a design information server; the web and XML technologies are used to implement the collaboration and information sharing mechanisms in the server. The Industry Foundation Classes (IFCs) are adopted as the information model of the server to facilitate the interoperability among multidisciplinary AEC software applications. The current implementation of the server system supports the automatic transformation of the design model contents and representations from the architectural domain to the structural domain, and remote visualization and interaction by Java applet and the Java3D technology. An algorithm that is able to deduce the topological relationship among different structural elements has been proposed in this paper. A case study is presented at the end of the paper to illustrate the use of the information server for the architectural and structural design collaboration.

Keywords: Industry Foundation Classes (IFCs); Information server; Collaborative building design; Web-enabled services

1. Introduction

Traditionally, architectural design and structural design are two separate steps with distinct objectives in the building design process. Architectural design is focused on defining the space arrangement of various architectural elements, while structural design on analyzing the mechanical properties of building elements and structure. The two design processes are closely related as architecture design, which defines the geometric information about building elements, provides the input information for structural design. Thus, the success of a building design process is highly dependent upon effective coordination among diverse design teams. With the prevalence of information technology (IT) applications in the AEC industry, the
efficiency and effectiveness of building design could be achieved through the interoperability and data exchange among different computer-aided design and analysis tools. Therefore, an IFC-based web server for collaborative building design between architects and structural engineers was proposed to meet this purpose.

Industry Foundation Classes (IFCs) [1,2] are a set of building product model specifications developed by the International Alliance for Interoperability (IAI) [3,4] for product data representation and exchange in the Architecture, Engineering, Construction/Facilities Management (AEC/FM) industry. The IFC standards suitably provide an information sharing and exchange platform, which can couple with the network technology to form a central database server to facilitate information exchange between different parties from various geographic locations (Fig. 1).

Without a software that can automatically extract necessary information for structural analysis from architecture design nowadays, structural engineers have to interpret and manually input the information either from architectural drawings or 2D CAD files. As beams and columns are usually represented as separate lines in 2D format, it is hard to extract the geometric information needed for structural analysis from 2D architectural drawings due to the difficulty on determining whether two adjacent lines belong to the same beam or column. The situation has changed with the presence of 3D attribute-driven object-oriented computer-aided design (OO-CAD) software [5–7]. In OO-CAD, each element is represented as a distinct object with its own attributes. For example, a beam is an object with lines (edges) as its attributes. A line, in turn, is an object with the coordinates of the two endpoints as its attributes. The easy acquisition and recognition of points and lines in OO-CAD make it possible for automatic extraction of geometric information in 3D format. An IFC-based web server was proposed in this paper to help extract needed information from architectural design for structural analysis based on the OO-CAD concept.

Up to now, the IFC Releases 2.0 and 2.x have covered six different AEC/FM domains—architecture, HVAC engineering, codes and standards, cost estimating, facilities management and simulation. The IFC Release 2.x has included the definitions of geometric resources, which follow the attribute-driven and object-oriented geometric framework and provide similar functions as OO-CAD. On the other hand, the Structural Domain Group of the German IAI Chapter proposed the ST4 extension to the IFC Release 2.x, which defines the data structure for part of the structural analysis domain. However, as the standard for AEC/FM product data representation and

---

Fig. 1. (a) Information flow between different parties in AEC without a central database. (b) Information flow between different parties in AEC with a central database.
exchange, IFC does not include the algorithm for deducing geometric information needed by structural design from architectural drawings. Thus, this functionality has to be added externally to improve the efficiency and effectiveness of building design.

2. Geometric representation of products in IFC

The definitions of geometric representations in the IFC Releases 2.0 and 2.x are quite close to the well-approved STEP geometric definition of ISO 10303-42:1994 [8]. Any object in IFC with a geometric representation has two attributes: ObjectPlacement and Representation. ObjectPlacement has the type of IfcObjectPlacement and stores the placement information of an object. It could be absolute (relative to the global coordinate), relative (relative to the ObjectPlacement of another product), or constrained (relative to the grid axes). Representation has the type of IfcProductRepresentation and stores the shape representations of an object. Its location is defined within the context of ObjectPlacement. Fig. 2 shows the EXPRESS-G (a graphic notation for EXPRESS) diagram for the geometric representation of a product, which could be either a beam or a column.

The default setting for the placement of an object is relative placement, defined by IfcLocalPlacement with a PlacementRelTo attribute to specify the referencing spatial context. For example, point A in Fig. 3 has a local placement value of (2, 2, 1) with respect to a coordinate \(x'\ y'\ z'\). The origin of the context \(x'y'z'\) has a local placement value of \((-1, 5, 3)\) with respect to another local coordinate \(x'\ y'\ z'\). In turn, the origin of context \(x'y'z'\) has a local placement value of \((1, 0, 3)\) with respect to the global coordinate \(xyz\). In web applications, the local placement of an object should be transferred to the global placement for the deduction of topological relationship.

There are two geometric representations of shape: the explicit geometric representation of shape and the attribute-driven representation of shape. Explicit
geometry is a geometric representation devoid of semantic meanings for its parts. It uses pure geometric definitions in terms of points, curves, surfaces and solid primitives. For example, a cube could be defined in terms of 8 points, 12 edge curves, 6 bound surfaces or combinations of some of the above. Attribute-driven geometry is a geometric representation driven by attributes. For examples, a cube could be defined using a placement entity and three “driving” attributes—length, width and height. Generally, IFC has chosen to use attribute-driven geometric definitions for objects. Fig. 4a shows the representation of a beam in its local coordinate. The attributes of the beam in Fig. 4a are:

- ExtrudedDirection: the direction in which the cross section is to be swept;
- Depth: the distance the cross section is to be swept;
- Cross section profile: for this case, it is a rectangle with a height of YDim and a width of XDim.

3. Deduction of topological relationship

According to Nauyen and Oloufa [9], there are five categories of topological relationships among building elements and space:

- One component is adjacent to the other (adjacency);
- Components are separate from each other (separation);
- One component lies within the other (containment);
- One component intersects the other (intersection);
- One component connects to the other (connectivity).

For structural analysis, only the last two relationships—intersection and connectivity—are used. If intersection is treated as a special case of connectivity, the deduction of topological relationship for structural analysis can be ascribed to finding the connectivity among structural elements. In the prototype implementations, a simple algorithm has been proposed to deduce the connectivity between prisms, which represent the most common 3D shape of structural elements.

In the algorithm, the box-boundary of a structural element has been used for the purpose of optimization. A box-boundary of a structural element is the minimum orthogonal box parallel to the global coordinate system, which can contain the structural element. Fig. 4b illustrates the concept of box-boundary of a structural prism.

It is obvious that if two box-boundaries are not connected, the prisms contained are also not connected. The algorithm for deducing whether two box-boundaries are connected is much easier than that for deducing the connectivity of prisms. Box-boundary connectivity is usually checked before prism connectivity, as the latter is more time-consuming.

The checking for prism connectivity is based on the occurrence of collision between the edges of a prism and the surfaces of the other. If two prisms are connected, an edge of one of the prisms must intersect with a surface of the other. Thus, the algorithm for connectivity checking is to find out whether a spatial
A line segment has a common point with a planar three-vertex facet (Fig. 5a). The examination of collision between complex polygons can be started by triangulating them into three-vertex facets.

The coefficients, $A$, $B$, $C$, and $D$, for the spatial plane that contains the facet in Fig. 5a can be calculated from the coordinates of the three vertices using Eqs. (1) and (2), where $P_{ax}$, $P_{ay}$, and $P_{az}$ refer to the $x$, $y$, and $z$ values of the coordinate of point $P_a$.

\[
(A, B, C) = (P_b - P_a)(P_c - P_a) \tag{1}
\]

\[
D = - (AP_{ax} + BP_{ay} + CP_{az}) \tag{2}
\]

Four steps are summarized below for intersection checking between a line segment and a facet, as shown in Fig. 5a:

1. Check whether the line that contains the segment and the plane that contains the facet are parallel;
2. Find the intersection of the line that contains the segment and the plane that contains the facet;
3. Check whether the intersection point is inside the line segment;
4. Check whether the intersection point is inside the facet.

$P$ in Fig. 5a illustrates the intersection point. Its position can be determined through the calculation of coefficient $\mu$ in Eq. (3), where $P_{1x}$, $P_{1y}$, $P_{1z}$ and $P_{2x}$, $P_{2y}$, $P_{2z}$ refer to the $x$, $y$ and $z$ values of the coordinates of point $P_1$ and $P_2$, respectively.

\[
\mu = \frac{AP_{1x} + BP_{1y} + CP_{1z} + D}{A(P_{1x} - P_{2x}) + B(P_{1y} - P_{2y}) + C(P_{1z} - P_{2z})} \tag{3}
\]

If the denominator of Eq. (3) is zero, then the line is parallel to the facet. If $\mu$ is between 0 and 1 (inclusive of 1), the intersection point is inside the line segment.

The final step, which checks whether the intersection point is inside the facet, is based on the calculation of area. If the intersection point is inside the facet (Fig. 5b), the total area of the three triangles $\Delta PP_aP_b$, $\Delta PP_bP_c$, and $\Delta PP_cP_a$ (Eq. (4)) must equal the area of $\Delta PP_aP_bP_c$ (Eq. (5)).

\[
\Delta PP_aP_b + \Delta PP_bP_c + \Delta PP_cP_a = \frac{1}{2} ||(P_a - P)(P_b - P)|| + ||(P_a - P)(P_c - P)|| + ||(P_c - P)(P_b - P)|| \tag{4}
\]

\[
\Delta PP_aP_bP_c = \frac{1}{2} ||(P_c - P_a)(P_b - P_a)|| \tag{5}
\]

4. Construction of structural model

A mechanical model for architecture should include idealized structural elements, their mechanical connectivity, support conditions, mechanical properties and loadings [10]. Among all the items, only idealized structural elements and connectivity information could be deduced from 3D architectural models. The others need to be input externally. There are three steps in the deductive process:

1. Finding the connectivity between structural elements;
2. Defining the idealized structural elements;
3. Finding the joints between two connected structural elements.

For a prism, its corresponding idealized structural element is the line segment connecting the centroids of the two end cross-sectional surfaces (Fig. 6). A joint can be defined as the intersection point of two idealized structural elements if they are on the same spatial plane or the nearest point of one element to the
other element if they are not on the same spatial plane. In prototype implementation, a structural element includes the location and index of its two joints.

5. Software implementation

The prototype system proposed in this paper was object-oriented and the Unified Modeling Language (UML) was adopted for analysis and design. Java, a widely used pure object-oriented language for network servers, was chosen as the software implementation language. Java is platform-agnostic and compiled Java binary codes can run on a variety of operating systems, such as Win32, UNIX, Linux and Mainframe systems. Furthermore, Java is strong on both server and client sides. On the server side, the Java 2 Platform, Enterprise Edition (J2EE) is a mature standard for the implementation of enterprise level web application servers and has been accepted by the software industry. On the client side, Java applet can run on any Java-enabled web browsers and provide two-way communication between the client and the server.

In the web server implementation, a set of Java classes was created for the corresponding entities in the IFC EXPRESS schema (Fig. 7). All the Java objects have been initiated and ready for use once the parser reads STEP IFC data files in the business logic process.

First, the architect logs in the server via a web browser, followed by identification and authorization check by the server. Once the channel is established, the architect could submit architectural design files in IFC format to the server. After receiving and saving the IFC data files, the server would trigger a corresponding application to retrieve the geometric information of structural elements (such as beams and columns) and deduce the topological relationships and joints between the elements. The obtained idealized structural elements and joints would be saved in XML format in the server for structural analysis. The server is now in an intermediate state, waiting for the structural engineer to react. After security checking, the structural engineer logs in and receives from the server a transmitted web page containing the Java applet which can show the 3D structural model on the browser. The Java applet enables the structural engineer to choose any specific structural element and add in new properties or comments. However, change of dimensions of structural elements is not allowed for the structural engineer by the Java applet, as the privilege is given only to the architect. The structural engineer could only feedback on possible geometric modifications. Once the additional information input by the structural engineer reaches the server, the application on the server would add the information to the Property Sets of the IFC data file and notify the architect of updates. The architect, upon receipt of notification, could download the IFC file from the server and review the updates in his CAD software. If any further modification is needed, another round of reviewing will start. Otherwise, the server will enter a final state and generate a web page for the architect and structural engineer to download the architectural design file in IFC format and structural analysis model in XML format. Fig. 8 illustrates the use case diagram in Unified Modeling Language (UML).

Fig. 9 shows the five possible states of the server in the use case. All possible interactions between the architect and structural engineer in the design process have a corresponding transition from one state to another.

6. Use case analysis

This section shows a typical use case, which involves both the architect and structural engineer. Fig. 6. Idealized structural elements and a joint of two structural prisms.

7. Client–server architecture

Fig. 10 illustrates the architecture of the web server system. It follows a three-tier model: the client, server...
and data storage. Currently, the client tier only supports Java applet enabled web browsers.

There are three layers inside the server: the data access layer, business logic layer and web component layer. The data access layer ensures that the server is data storage independent and IFC standard independent. The data access interface allows the user to read, write and modify IFC data in the web environment without knowing how the data are stored. If the way of data storage is changed, the only thing that needs to

Legend

Class

A has B as an attribute

B is a super class of A

Fig. 7. UML class diagram of Java classes corresponding to the entities defined by IFC EXPRESS schema.
be modified is the coding of the data access interface. As the IFC EXPRESS schema is currently undergoing some changes, to ensure the reusability of the codes, the properties of building components are encapsulated in Java Beans, which could be initiated by accessing IFC data through the data access interface.
Business Logic Beans could read the encapsulated information from the building component Java Beans without knowing IFC schemas. If the IFC EXPRESS schema changes, only the initiation method in the building component Java Beans will be affected. Business Logic Beans contain all the algorithms that are needed for the processing of geometric information. The Model Bean is a high-level interface for the upstream web components and organizes the business logics to finish the integrated tasks requested by web components. StructManager Bean can generate structure models and PropertySetMan-
ager Bean can add property sets to IFC files. GeoManager Bean, TopologicalManager Bean and XMLOutputter Bean are for special functions like generating global coordinates, deducing topological relationships between structural elements and outputting structural models in XML format. The hierarchical structure of Business Logic Beans makes the software design work easier to manage. The web component contains JavaServer Pages (JSP), which could dynamically generate web pages according to the state of the server, and Java Servlet Control, which could receive input from the client and dispatch the client’s request to the Model Bean. This design follows the popular Model-View-Controller (MVC) architecture, which is illustrated as the class diagram in Fig. 11.

The client side comprises a Java-enabled web browser which supports Java3D. The Java 3D Display Bean reads XML-format structural models from the server and displays 3D outcomes on the browser. It allows the user to input data to an element through a pop-up window which could be triggered by clicking on the element. As Java is platform independent, there are no specific requirements on the operating systems used by client machines.

8. Case study

This section shows the implementation of collaborative building design with the developed web server.

First, the architect logs in the web site from a remote computer (Fig. 12a). After identification and authorization check on the architect, the server would generate a JSP page that would allow the architect to upload an IFC-format seven-story building with a total of 741 beams and 442 columns (Fig. 13a). Upon receipt of the building file, the server generates a corresponding structural analysis model and saves it in XML format.

The structural engineer at another location could log in the server via a secured channel and view the structural analysis model on his browser. Although the structural engineer could not directly modify the structural analysis model generated by the server, he
Fig. 12. JSP pages for interaction with the web server. (a) JSP page for logging in from a remote PC. (b) Final JSP page for file downloading by architects and structural engineers.
could add new properties or give comments to any structural elements (Fig. 13b).

Java applet allows the structural engineer to zoom in, zoom out and walk through the 3D structural analysis model to have a close view. When a structural element is clicked on, a pop-up window would appear and list the dimension information of the chosen element and the deduced joint locations of its corresponding idealized element (Fig. 13b). The pop-up window enables the structural engineer to add new Property name/Property value pairs to selected elements. A text area under the Comments tab allows the structural engineer to comment on the locations or dimensions of elements. All the changes made by the structural engineer would be sent back to the server and incorporated into the IFC data file after the ‘Feed
Back” button is pressed. The updated IFC data file needs to be reviewed and accepted by the architect. If any disagreements exist, the reviewing and modifying process has to repeat until no further changes are required by both the architect and structural engineer. Up to this point, the server would generate a web page to close the information feedback process and allow the architect and structural engineer to download the architectural model in IFC format and the structural analysis model in XML format (Fig. 12b).

The XML-format structural analysis model has Document Type Definition (DTD) as follows and can be easily transformed to other data formats used by structural analysis software.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!ELEMENT StructureModel (StructLinearElement+ )>
<!ATTLIST StructureModel xmlns CDATA #REQUIRED>
<!ELEMENT StructLinearElement (Type, GUID, Profile, Depth, StartPointGlobalCoordination, EndPointGlobalCoordination, JointCollection )>
<!ELEMENT Type ( #PCDATA )>
<!ELEMENT GUID ( #PCDATA )>
<!ELEMENT Profile ( Width, Height )>
<!ELEMENT Width ( #PCDATA )>
<!ELEMENT Height ( #PCDATA )>
<!ELEMENT Depth ( #PCDATA )>
<!ELEMENT StartPointGlobalCoordination (X, Y, Z )>
<!ELEMENT EndPointGlobalCoordination (X, Y, Z )>
<!ELEMENT JointCollection ( Joint+ )>
<!ELEMENT Joint ( Index, GlobalCoordination )>
<!ELEMENT Index ( #PCDATA )>
<!ELEMENT GlobalCoordination ( X, Y, Z )>
<!ELEMENT X ( #PCDATA )>
<!ELEMENT Y ( #PCDATA )>
<!ELEMENT Z ( #PCDATA )>
```

The current IFC Release 2.x does not include the structural analysis domain. Therefore, analyzed structural models cannot merge with IFC data files. This situation would be changed in the near future with the efforts from the Structural Domain Group of the German IAI Chapter.

9. Conclusions

With the development of Industry Foundation Classes (IFCs) and network technologies, efficient collaborative design between architects and structural engineers becomes realistic. In this paper, an IFC-based web server that could automatically extract geometric information from OO-CAD (3D attribute-driven object-oriented computer-aided design) architectural drawings for structural analyses was developed using the J2EE (Java 2 platform, Enterprise Edition) standard. Together with the web server, an algorithm was proposed to deduce topological relationships between different structural elements. To illustrate the application of the developed web server, a simple case study was provided to show how an architect could collaborate with a structural engineer through the use of the IFC-based web server and how information flows at the collaborative design phase. In the case study, only beams and columns are taken into account. Although the developed web server was applied to collaborative architectural and structural design, extensions to other AEC/FM domains are possible. When IFCs and other standards for AEC/FM interoperability are more mature and better implemented, an integrated design, construction, operation and maintenance process throughout an entire building life cycle is likely to happen.
References