PAPER REFERENCE: T191-4

INTERNET-BASED COMPUTER-AIDED DESIGN: LEVERAGING PRODUCT MODEL, DISTRIBUTED OBJECT, AND WORLD WIDE WEB STANDARDS

Charles S. Han, John C. Kunz, Kincho H. Law

EXTENDED ABSTRACT

Non-Internet and Internet-based technologies have matured to a point that Internet-based CAD systems can be implemented. This paper reviews and discusses the technologies in relation to a prototype Internet-based CAD system that is being developed. In addition, we discuss further enhancements necessary to make the prototype a viable commercial working system.

Regli has outlined the technologies now available to make Internet-based, or Network, CAD a reality. Among the various issues discussed are leveraging the Internet, online documentation and services, data modeling, and security. We have experimented and integrated the available technologies to demonstrate a simple working prototype that focuses on a browser-based user interface and a backend distributed object environment.

The three technologies we present and have utilized in our prototype are

1. A standard product model based on the International Alliance for Interoperability Industry Foundation classes. We use this product model as the point of departure.

2. A distributed object environment that allows for the development and transfer of a design using the product model. We are utilizing the Common Object Request Broker Architecture (CORBA).

3. Standard browser technologies that serve as the user interface (UI). We are the ISO standard VRML 97 for the graphical portion of the UI and a Java applet window as the command/text portion of the UI. We use the External Authoring Interface (EAI) to communicate between these two modules of the UI.

Our research focuses on issues concerning the Architecture-Engineering-Construction (AEC) industry so the product model discussed here relates to buildings and building facilities. However, one can easily substitute the product model for one that is more applicable to a specific discipline.

Though our initial efforts have incorporated network computing technologies that will enable a model for the next generation of CAD systems beyond current CAD technology, a complete Internet-wide infrastructure which includes the participation of vendors and product-related services is necessary to realize the full potential of Internet-based CAD. We briefly review and discuss the necessary components for a successful implementation of our Internet-based CAD system.

PAPER REFERENCE: T191-4

INTERNET-BASED COMPUTER-AIDED DESIGN: LEVERAGING PRODUCT MODEL, DISTRIBUTED OBJECT, AND WORLD WIDE WEB STANDARDS

Charles S. Han[1], John C. Kunz[2], Kincho H. Law[1]

[1]Department of Civil and Environmental Engineering, Stanford University
Stanford, CA 94305-4020

[2]Center for Integrated Facility Engineering, Stanford University
Stanford, CA 94305-4020

ABSTRACT

Non-Internet and Internet-based technologies have matured to a point that Internet-based CAD systems can be implemented. This paper reviews and discusses the technologies in relation to a prototype Internet-based CAD system that is being developed. In addition, we discuss further enhancements necessary to make the prototype a viable commercial working system.

INTRODUCTION

With the maturation of several technologies, both non-Internet- and Internet-based, concepts that have been proposed for the development Internet-based Computer-Aided Design (CAD) can now be realized. We present a brief overview of these technologies along with a demonstration system leveraging these technologies. The three types of technologies discussed in this paper are:

4. The use of a standard product model.

5. A distributed object environment that allows for the development and transfer of a design using the product model.

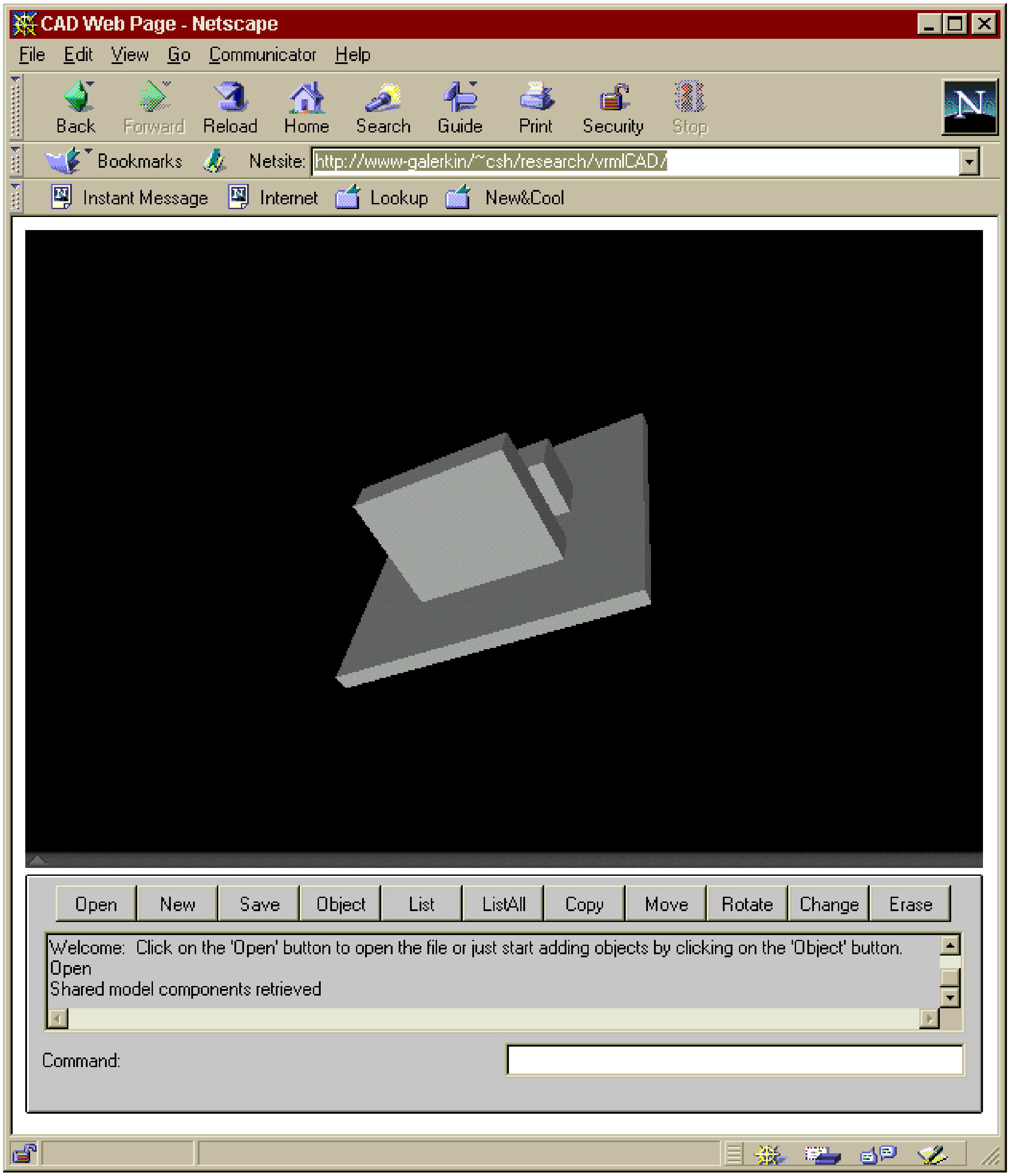
6. Standard browser technologies that serve as the user interface (UI).

This paper provides an overview on the structure of an Internet-based CAD prototype we have developed (see Figure 1). Our research focuses on issues concerning the Architecture-Engineering-Construction (AEC) industry so the product model discussed here relates to buildings and building facilities. However, one can easily substitute the product model for one that is more applicable to a specific discipline.

Though our initial efforts have incorporated network computing technologies that will enable a model for the next generation of CAD systems beyond current CAD technology, a complete Internet-wide infrastructure which includes the participation of vendors and product-related services is necessary to realize the full potential of Internet-based CAD. We briefly review and discuss the necessary components for a successful implementation of our Internet-based CAD system.

BACKGROUND

Regli has outlined the technologies now available to make Internet-based, or Network, CAD a reality [Regli97]. Among the various issues discussed are leveraging the Internet, online documentation and services, data modeling, and security. We have experimented and integrated the available technologies to demonstrate a simple working prototype that focuses on a browser-based user interface and a backend distributed object environment.



Netsite: http://www-galerkin/~csh/research/vrmlCAD/

```
Welcome:  Click on the 'Open' button to open the file or just start adding objects by clicking on the 'Object' button.
Open
Shared model components retrieved
```

Command:

Figure 1: The Web page UI for the prototype.

THE PRODUCT MODEL

The development of a standard product model is essential for efficient transfer of design information between various disciplines as well as to standardize and automate design analyses. The standard model must provide more information than what is contained in drawing primitives such as lines and arcs that are supported by current CAD systems. Of course, these primitives can be grouped to form objects that are associated with desired behaviors, but beyond certain standards such as dimension lines, these objects are not universally recognized. Currently, there is an effort by the International Alliance of Interoperability (IAI) to develop standards for a project model that enables interoperability between applications by different software vendors [IAI97]. The IAI is a consortium of CAD vendors and other AEC industry partners, and it has defined a set of objects called Industry Foundation Classes (IFCs) that conform to current object-oriented philosophy. Figure 2 shows part of the IFC class hierarchy.

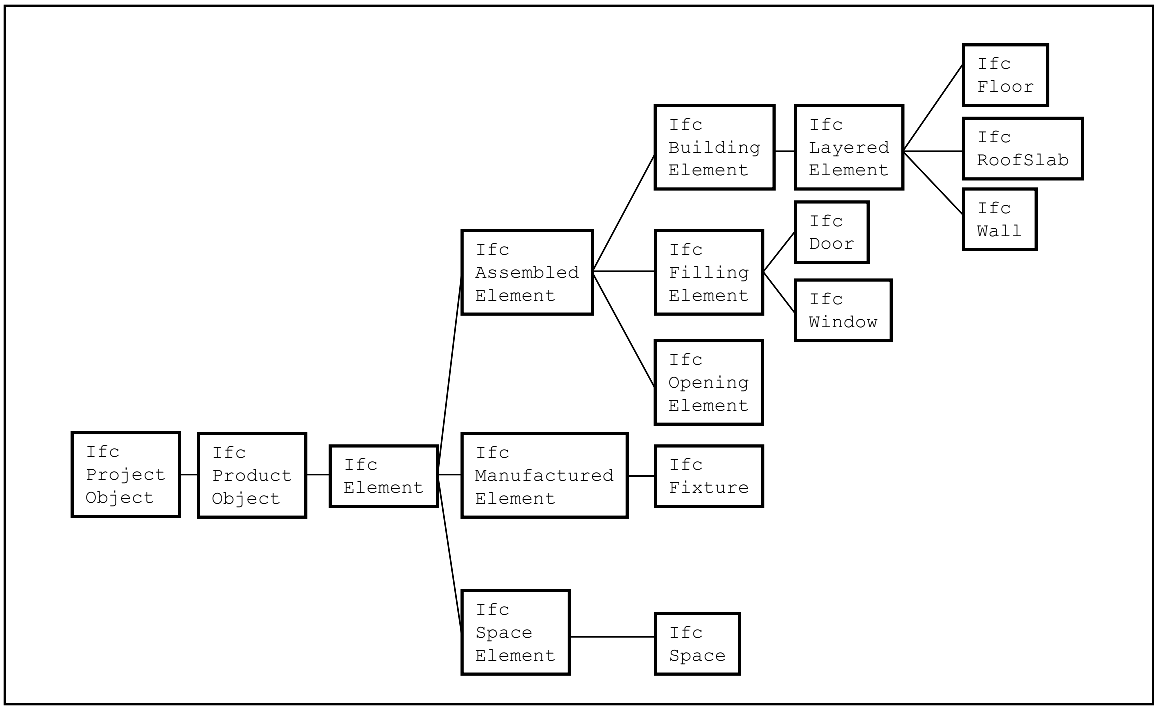


Figure 2: Part of the IFC class hierarchy

We have adopted the IFC model as a point of departure for the product model implemented in our prototype. The classes and their respective attributes and relationships that have been developed by the IAI are building-industry-specific. For example, the various professions related to the construction industry, the relationships between walls, openings, doors, and windows are critical, and these relationships have been addressed in the IFC product model. The class `IfcBuildingElement` (of which `IfcWall` is a grandchild class) has a container class `IfcRelVoids` which can hold instances of openings (`IfcOpening`). An `IfcOpening`, in turn, has a container class `IfcRelFills` which can hold instances of `IfcFillingElement` (of which `IfcDoor` and `IfcWindow` are child classes). The EXPRESS schema for the class `IfcBuildingElement` is illustrated in Figure 3.

IFC Release 1.0 currently defines two standard formats for sharing project data: standard EXPRESS file format and software interfaces [IAI97]. In the prototype, we have selected to implement our own version of the Interface Definition Language (IDL) for the software interface since we have implemented the prototype in a distributed object environment.

EXPRESS:

```
ENTITY IfcBuildingElement
 ABSTRACT SUPERTYPE OF (
  ONE OF (
   IfcLayeredElement,
   IfcCoveringElement,
   IfcBuiltIn,
   IfcProfiledElement
  )
 )
 SUBTYPE OF (
  IfcAssembledElement
 )
INVERSE
 HasOpenings : SET [0:?] OF IfcRelVoids FOR
  VoidsBuildingElement;
END_ENTITY;
```

IDL:

```
interface IfcBuildingElement : IfcAssembledElement {
 typedef sequence <IfcRelVoids> HasOpenings;
};
```

Figure 3: EXPRESS schema for the class `IfcBuildingElement` and the corresponding IDL interface.

A DISTRIBUTED OBJECT ENVIRONMENT

Leveraging network computing technologies is a logical extension to current CAD systems. Network technologies ranging from information and service retrieval and transfer of design data to the client, consultants, and contractors are examples of increased communication flow. From a practical and organizational standpoint, implementation of network technologies should be transparent to the users—the user should have the same ease of access to information whether it is local on one's own computer or on a remote server. The implementation of the World Wide Web (WWW) is an obvious example in which users have easy access to information regardless of the location of the source. In order for such transparent access to information within a CAD environment, an underlying distributed object environment must be implemented.

There are several emerging distributed computing environments. For the prototype, we have used the Common Object Request Broker Architecture (CORBA), a standard that is published by the Object Management Group (OMG). CORBA provides both the transparency and the object-oriented structure. From [Vogel97]:

> "The ORB provides *location transparency* to objects. That is, an object is represented by an *object reference* on which a client can invoke operations as if they were methods on a local object. Object references may refer to objects that are in the same problem, in a different process on the same machine, or located on a remote machine…The other transparency that assists in the integration of distributed applications is *programming language transparency*…These transparencies are achieved by the use of OMG's *Interface Definition Language* (IDL)."

Using IDL, we define the class hierarchy of the IFC objects discussed in the previous section with the appropriate attributes and relationships. The IFC definitions are then used to generate the objects in the desired programming language, in this case, Java. In addition, we define the methods in IDL that operate on the specified object, and these methods are in turn mapped to methods of the appropriate objects in the desired programming language (in our implementation, also Java). As noted, there is no requirement that both the server and the client be written in the same programming language. Java is used on the server side as a matter of convenience, but a commercial implementation in which computing speed is a factor would benefit from using a programming language that can be compiled to take advantage of the server architecture.

We use IDL to define the IFC class hierarchy and methods that the server application needs to access the IFC classes—in this case, the prototype's server application is a simple building component repository (BCR) that has methods for saving, deleting, and retrieving the objects. See Figure 3 for an example of the EXPRESS schema-to-IDL mapping of an example IFC object. The IDL-to-Java utility also generates Java code for the client application implementation of the objects and methods—in this case, the prototype's client application is the CAD system's UI.

STANDARD BROWSER TECHNOLOGIES

As noted in the previous section, the implementation of the World Wide Web (WWW) has made ubiquitous network access a reality. It is logical, then, to leverage browser standards to implement the client side of the prototype. Thus, anyone with a network connection and standard browser software (as well as the necessary permissions) can use the prototype. In this section, we discuss the standard browser technologies implemented for the UI.

JAVA

Java has been integrated into standard browser technology for the following reasons. Paraphrasing from [Cornell96]:

1. It is a true programming language making a Java applet more responsive than other Web page technologies such as exclusive use of a CGI script.

2. Processing is off-loaded to the user's system thus following a client-server model.

3. Since Java uses a byte-code model, it is platform independent as long as the particular architecture fully supports the Java virtual machine.

4. And, finally, from the Java "White Paper" from Sun Microsystems, Inc.:

   > "Another aspect of being simple is being small."

From a software development standpoint, Java is an efficient, straightforward, and, relative to C++, headache-free implementation of an object-oriented programming language. Examples of this simplicity include elimination of pointers and elimination of multiple inheritance, and replace it with the concept of interfaces. Another claim made in the "White Paper" is the simplicity of network or distributed programming in Java. While this is indeed true and would seem to be important for a distributed CAD system, since the prototype uses the CORBA distributed architecture in which remote objects are transparently accessed, the distributed programming simplicity is not a factor. The main motivation for writing the prototype client side in Java is the ubiquity of the standard browser technologies, among which is the integration of Java in the form of applets.

For security reasons, Java applets cannot directly store objects on the server (there is no persistence of objects). This is, of course, the reason that some kind of component database such as our CORBA implementation is necessary. A Java applet can communicate with a socket on the server, and this communication protocol used to handle communication between the applet and the BCR server.

VRML AND THE EXTERNAL AUTHORING INTERFACE

Using the Virtual Reality Modeling Language (VRML, formerly Virtual Reality Markup Language) for the graphical portion of the UI as opposed to developing a graphical UI within the client side of the prototype was consistent with leveraging existing standards. As opposed to Java, which is currently not a formal standard, VRML97 (almost identical to VRML 2.0) is the informal name of the International Standard file format for describing interactive three-dimensional multimedia on the Internet [ISO97].

Briefly, VRML allows the author to construct objects in a three-dimensional world in a logical hierarchical structure. This hierarchical structure is made up of *nodes* that have specialized functionality. For example, the `Shape` node is used to describe the physical properties of an object, the `Transform` node describes an object's orientation in space, and a sensor is a node that senses some kind of interaction with an object (for example, clicking on an object). Sensor interaction is one type of *event*. Other events include changing certain properties (attributes that are *exposed*, or allowed to be changed) of a node. For example, an event sent to a `Material` node can change the `ambientIntensity` attribute of an object thus allowing an object to be highlighted in a certain state.

The External Authoring Interface (EAI)[1] is a proposal for a VRML97 Informative Annex that allows a Java applet to send and receive events to and from a VRML world within the standard browser environment. Thus, we have utilized the EAI to communicate between the command window the graphics window.

INTEGRATION OF TECHNOLOGIES

Figure 4 illustrates the structure of the prototype. To simplify this discussion of the integration of the aforementioned technologies, we step through the current version of the prototype that implemented a primitive object.

Development and implementation of the prototype was carried out on a Sun Ultra1 which has both the CORBA ORB and the Web server. However, since the CORBA environment (both the Sun implementation and the prototype server) has been written in Java, any platform that supports the Java Virtual Machine and has a Web server can theoretically run the prototype.

As illustrated in Figure 4, we have tried to keep the different functional modules of the client-side of the prototype as separate as possible. One module, the command interface (the class `CommandInterface`), sends and receives messages from the CORBA module (the class `CORBAInterface`) to communicate with the BCR. It also sends and receives messages from the VRML module (the class `VRMLInterface`). Currently, the prototype is accessible by viewing the Web page using Netscape Communicator 4.04 with Cosmo Player 1.0.2, the VRML *plugin* from Silicon Graphics, in the Microsoft Windows environment. This is currently the only platform for which Java Applet-EAI-VRML support exists. In addition to leveraging the ISO VRML97 standard, using Silicon Graphics' Cosmo Player VRML viewer technology relieves us of writing viewpoint navigation methods. These methods include zooming, panning, and rotating the view in the three-dimensional environment. Note that the navigation methods of a particular VRML viewer are not part of the ISO standard [ISO97].

[1] As proposed by Chris Marrin, Silicon Graphics. See http://vrml.sgi.com/moving-worlds/spec/ExternalInterface.html.

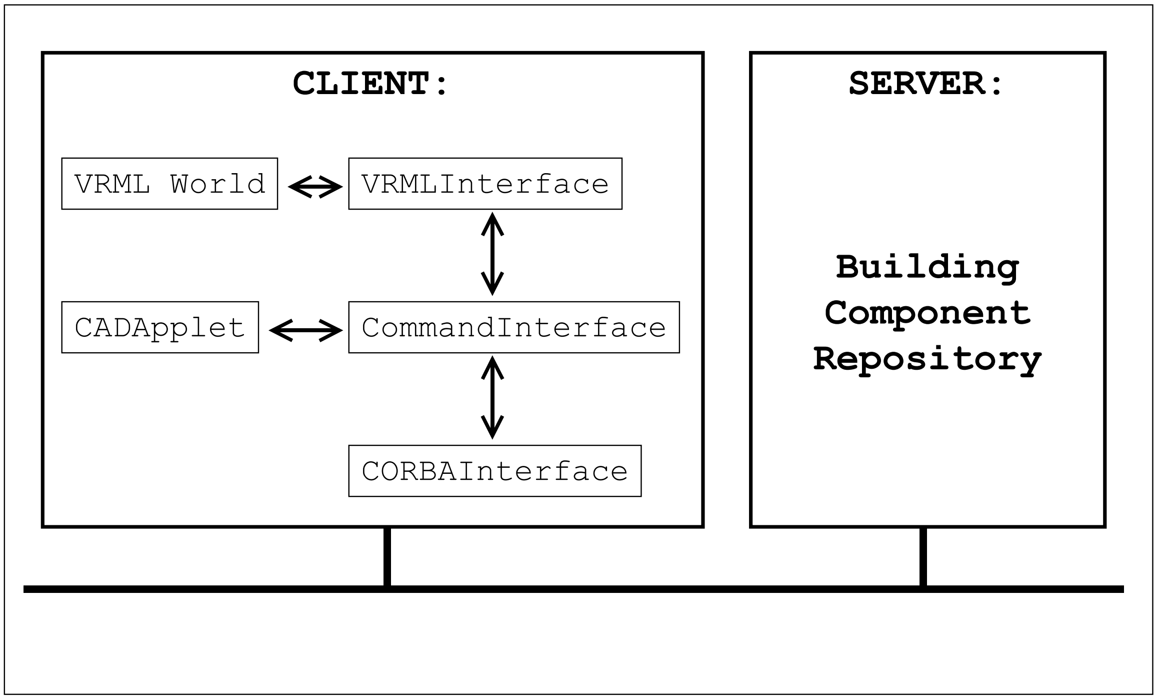


Figure 4: The basic prototype structure.

DISCUSSION

We have integrated existing technologies to demonstrate the feasibility of an Internet-based CAD system. We have focused on implementing a CAD browser UI with a backend distributed object environment that allows a user to retrieve and update a building component repository. The next step is to address issues concerning security and services. The BCR represents a portion of the shared model paradigm [Fischer96]. With a shared model, issues of security including digital signature and revision control must be implemented. The protocol for the implementation of services is an additional component of the shared model environment.

Since we have utilized the CORBA environment, we envision implementing services using the CORBA Trading Service protocol which has its basis in the ISO Open Distributed Processing (ODP) standards [Vogel97]. In this environment, a service will register its services with the trading service. Another application can query this trading service about the registered services. For example, in the prototype, suppose the user would like to design a bathroom with a certain fixture in mind. The user could insert a fixture, and if the maker of the specific fixture is registered with a predetermined trading service, the critical attributes of the fixture could be automatically retrieved by the BCR and passed along to the CAD UI. For more complex services, a querying protocol must be considered [Arnold97]. Such a protocol would adhere to standards such as Parts Library (PLIB) [ISO95].

ACKNOWLEDGMENTS

This research is partially sponsored by the Center for Integrated Facility Engineering (CIFE) at Stanford University.

REFERENCES

[Arnold97] Arnold, James Andrew, Paul Teicholz, John Kunz (1997). "An Approach for the Interoperation of Web-distributed Applications with a Design Model," *submitted for publication in Automation in Construction.*

[Cornell96] Cornell, Gary and Cay S. Horstmann (1996). *Core Java*, Prentice Hall, Upper Saddle River, NJ.

[IAI97] International Alliance for Interoperability (1997). *Industry Foundation Classes Release 1.0, Specifications Volumes 1-4, Washington DC.*

[ISO95] *ISO CD 13584-10:1995 Parts Library – Part 10*, ISO, Geneva, Switzerland.

[ISO97] *VRML97, ISO/IEC 14772-1:1997Information Technology—The Virtual Reality Modeling Language*, ISO/IEC, Geneva, Switzerland.

[Regli97] Regli, William C. (1997) "Internet-Enabled Computer-Aided Design," *IEEE Internet Computing,* IEEE 1(1), pp. 39-50.

[Vogel97] Vogel, Andreas, and Keith Duddy (1997). *Java Programming with CORBA*, John Wiley and Sons, Inc., New York, NY.