INTERNET-BASED ARCHITECTURAL VISUALIZATON

A. SCOTT HOWE, ARCHITECT[1]
University of Michigan / Kajima Corporation

INTRODUCTION

The types of information utilized in the field of architecture are many and varied. These range from the simplest of attributes included in the description of a bolt or nail to the complex geometries and abstract concepts required to fully articulate the human-spatial interactions of a city. Architects must process enormous amounts of raw data. The traditional methods of information visualization, which include paper drawings and scale models, are gradually giving way to electronic media. Nevertheless the general trend has been weighted toward an attempt to force the new technologies onto traditional visualization and construction methods rather than to search for more fitting solutions.

This paper discusses what is believed to be sound characteristics for an architectural visualization system that fully embraces new technologies and methods. First a concept or specification for the system will be established, then a simple current research project, the network-based kit-of-parts virtual building system using VRML (which is an initial attempt at implementing the concept), will be demonstrated.

PART I: INFORMATION VISUALIZATION: ARCHITECTURAL INFORMATION

Architectural information can be divided into two major categories: 1) information about the building artifact itself, and 2) methods for bringing the artifact into being: 1) From the top down, a hierarchy of information types and elements can be used to describe a building. A building can be expressed in terms of its functions or activities. The functions or activities can be expressed in terms of their defining systems or containing spaces. Systems can be expressed in terms of their components, and spaces defined in terms of their enclosing structures. Components and Structures can be expressed in terms of their composition and materials. Materials can be expressed in terms of their defining attributes. 2) Methods for constructing the building include rules of how the elements fit together and procedures for their assembly. Often the design of the building is greatly influenced by the methods of construction, and the component size by their ability to be transported.

Having the ability to isolate each bit of information when needed for a specific task while filtering out enormous volumes of unneeded information is the purpose of information visualization.

TRADITIONAL METHODS OF VISUALIZATION

TWO DIMENSIONAL EXPRESSION: The major part of an architect's training is concerned with visualization by way of drawings. Architects are trained to be proficient in the use of a variety of tools, ranging from a simple pencil to a complex computer system as a means of expressing the nature of the building they've conceived. Drawing is so important that in many countries including the United States it is a prerequisite for local government's approval of a building project. It is for this reason that the general thrust of the development of tools for architects has been to address this need to produce drawings.

THREE DIMENSIONAL EXPRESSION: In architecture, the primary goal is not the drawings themselves but the production of a dynamic three dimensional collection of spaces, defined by structure, function, and aesthetics. Plans, sections, and elevations, which constitute the main types of views included in working drawing packages, are merely two dimensional slices of the three dimensional object. As a partial attempt to overcome this inadequacy, three dimensional representational drawings (perspectives and axonometrics) and scale models have been a traditional spatial visualization technique.

STATIC INFORMATION: In the preparation of working drawings and specifications, it is often required that certain bits of information be repeated for clarity. For example, "calling out" material designations on an elevation may require the same information to be noted in various places on the same drawing, with references to the same material included in the specifications. In traditional drawings, problems occur when changes occur during the design process and the content of the information changes. In the example of the elevation material designations, the "call outs" are static and each of them would need to be changed if the material is changed.

DYNAMIC INFORMATION: The use of computers has greatly improved our ability to solve the problems associated with static information. Drawings and other means of architectural expression which have traditionally been conceived as separate entities can often be derived from core information. "Call-outs" can be handled as keynotes which are dynamically linked to specifications such that redundancy becomes only a matter of appearance: repetition becomes instances of the same bit of information. In the dynamic use of information, changing a core piece of information will automatically change all of the instances.

Still, many problems remain. Reliance upon two dimensional media for the representation of innately three dimensional artifacts continues to be the primary function of architectural visualization. Information required for structural, environmental, daylighting, acoustical, energy, and other forms of analysis is for the most part still unlinked and static within the scope of the entire project. In addition, currently the role of the building representation (in other words drawings, computer models, etc.) is viewed as having ended its usefulness when the object it represents is built. This ignores the potential wealth of information regarding adequacy of design that could be gleaned over the entire life span of the building.

VIRTUAL BUILDINGS

Some of these problems can be solved with the Virtual Building concept. Smith (1992) explained:

A finished building is the best model of what the building will be... We don't usually have the luxury of constructing the complete building as a model or prototype. If we could do this, we could identify its deficiencies and our errors, and then tear it down and construct an improved version. What is needed is a Virtual Building that we could design and construct as many times as we please.[2]

A Virtual Building is a three-dimensional representation of an innately three-dimensional artifact. All the types of information, from functions to spaces to structure to material to attribute, are fully represented and linked. A designer can manipulate simple volumes which are abstracted versions of the building's major form elements. In the same model, walk-throughs and fly-throughs can be accommodated in real time using virtual reality, such that materials, spaces, and sunlight can be experienced. Tools for performing complete analysis by filtering out only that data which is needed for the specific task would be linked to the same model. By keeping the Virtual Building active after the construction of the real thing, facility management could be facilitated effortlessly. Using sensors, the virtual building could be linked with the real one for the purpose of monitoring performance along its entire lifetime.

In virtual buildings, drawings would be incidental to the representation instead of being the primary objective. When a designer manipulates the simple volumes, any drawings that represent two dimensional slices would be automatically updated. With the use of linked expert systems, elements that represent structural members would be linked to the volumes, and recalculation would occur along with automatic resizing of the members.

Virtual buildings can be created with various tools available on the market today. However, they will require large databases and multiple data formats. Actual implementation of virtual buildings as a product would require the development of an enhanced visualization system or browser which could potentially be linked with expert systems.

SPECIFICATION FOR A VISUALIZATION SYSTEM

Due to size, especially with large projects, it may be impractical to contain all of the project database on a single system all at the same time. A one-minute animation alone could take up as much space as 800 megabytes depending on resolution. Instead, proposed is the use of networks such as the Internet with its potentially unlimited number of memory space and information sources coupled with real-time visualization and filtering techniques. The assumption would be that in order to fully implement this system, building component manufacturers and materials suppliers would need to maintain servers on the Internet, and architects and designers would need to have access thereto.

Basic Characteristics: LEVELS OF ABSTRACTION & FILTERS: It would be necessary for the browser to have the capacity to view the project database in various ways, or at different levels of abstraction. Geometrical properties, textures, wireframe, textual descriptions, and diagramatical or symbolical views of the same database as well as the ability to apply filters would be necessary. Viewing all of the information at once would be unintelligible, but applying filters would select only that which is needed at the time. Other filters might include hither and yon sectioning (conical, orthographic, and zoned) with automatic line width adjustment for the cut edge, dimensioning filters with automatic orthographic or perspective linked dimensioning, underlay and overlay filters which can handle linked images or linked text, etc.

WINDOWS, SUBWINDOWS & BOOKMARKS: The browser would necessarily have the capacity to support multiple windows. Each window would be independent of each other in that different levels of abstraction or filters could be applied to different active windows. Within each window, the ability to insert a subwindow or lens with a different level of abstraction or filter would also be useful. Each window should be storable. Independent of the project database, a feature allowing casual visitors to the virtual building to save their own views or "bookmarks" (similar to the way one can save bookmarks on the Web) would be required.

Possible Plug-In Modules:INPUT & EXPORT: A function allowing the import and export of various data formats such as DXF, IGES, STEP, VRML, 3DMF, OBJ, PICT, TIFF, RGB, MOV, live video, and sounds as well as views, bookmarks, windows, subwindows, and filters would also be necessary.

SHEETS: The capacity to arrange and store sheets for the purpose of plotting output would prove useful. A sheet would consist of an arrangement of windows (as defined earlier) that are either bordered or borderless and can overlap with adjustable degrees of transparency. In developing a sheet editor, PAD++[3] with its infinite zooming capabilities and infinite work surface, could be considered.

WORD PROCESSOR & SPREADSHEET: Having full word processing and spreadsheet characteristics in a fully linked manner would be required. The spreadsheet would need to be three dimensional, similar to Microsoft Excel.[4]

TEXT-BASED VISUALIZATION TOOL: For use with visualizing large volumes of textual data, it would be necessary to include the capacity to view certain parts of the database and have the remaining parts compressed "around the edges" using Fish-eye Views[5] and Butterfly[6] visualization, or data hierarchies with Tree Maps and Clustering.[7]

PRODUCT MODEL VISUALIZATION TOOL: It would be necessary to have the capacity to represent the model in product model coding such as NIAM8, as proposed by the International Standards Organization,[9] or IDNA.[10]

MODELING: Modeling which supports manipulation of all of the geometric data elements would be necessary. Point manipulation, line & curve editing, surface operations, solid Boolean operations, as well as advanced modeling techniques such as mesh, fillet, sweep, rounding, parallel object, text editor would be needed. Excellent modelers include CATIA[11]

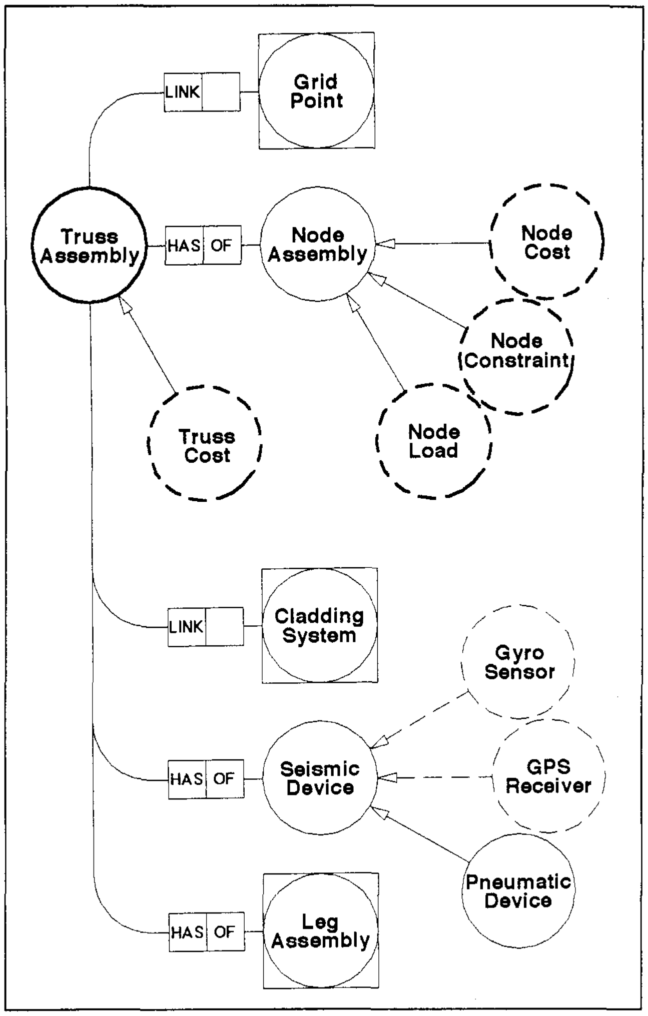

Figure 1: Product model visualization

and Form Z.[12]

RENDERING: Required would be the ability to apply texture mapping, bump mapping, fractal mapping, and decals. The full range of wire frame, hidden line, flat shading, gouraud shading, phong shading, Z buffer rendering, scan line rendering, radiosity, and ray tracing rendering is required. Powerful renderers include Form Z RenderZone and KPT Bryce.[13]

ANIMATING: The ability to create walk-through paths, animations, quick-time movies, and video input & output is necessary. Exemplary animation software might include Shade III.[14]

KINEMATICS: A feature for creating self-contained (behavior scripted) kinematic mechanisms, with the capacity for interactive operation or independent operation programming would be important. CATIA has a powerful kinematics module.

EXPERT SYSTEMS: Plug-in virtual designer, virtual structural engineer, virtual mechanical engineer, virtual electrical engineer, virtual civil engineer, etcetera could be developed. In order to develop a browser, a powerful, flexible standard for data structure is required. In the scope of this paper, all the possibilities will not enumerated but a brief discussion of three basic types of elements will be made: points, lines & curves, flat & curved faces.

Data Structure: POINTS: In typical modelers points are represented as simple three-coordinate entities. Proposed is a point entity that by default and at the simplest form of abstraction is a three-coordinate element in the usual sense, but at higher levels of complexity could have an unlimited number of attributes attached, such as pointers to objects.

LINES & CURVES: Typically lines are represented as simple entities with a starting and ending point. Proposed is a line entity, that by default and at the simplest form of abstraction, will have the conventional characteristics. At higher levels of complexity these entities could become sweep paths with an unlimited number of attributes attached, which may include cross-sectional shapes. Curves, including splines, would be pure mathematical curves but would be represented by segmented lines whose visual roughness can be adjusted. Curves would be similar to lines in that they could become sweep paths with unlimited number of attributes attached at higher levels of complexity.

FLAT & CURVED SURFACES: In conventional modeling flat surfaces are defined by a set of points and curves which represent the surface edge. Proposed is a surface that by default at the simplest form of abstraction could have conventional characteristics, but at higher levels of complexity could become plates or panels with an unlimited number of attributes attached, including thickness and multiple parallel layers. Curved surfaces, including spline-based surfaces would be stored in pure mathematical form but would be represented by segmented surfaces whose roughness can be adjusted at the time of modeling. Curved surfaces would be similar to flat surfaces in that they could become curved plates with thickness, multiple parallel layers, and an unlimited number of attributes attached.

Applications of Data Structure: Models using this data structure can exist in an extremely lean form as far as the number of bytes of raw data is concerned, and utilize attributes as pointers to fetch more data over the Internet (or dump some of it) in changes of the levels of abstraction and filters. For example: in simple form a series of points can be given the attributes of circles or round-cornered rectagonal shapes lying on plane z=0 with lines attached. Raise the level of complexity and they would become rooms connected to hallways. The simple level of abstraction was probably where the architect started, with bubble diagramming and simple circulation flows. By adding levels of complexity, the original bubble diagrams are not lost; only viewed differently.

In another example, a polygon may represent the wall of a room. In a higher level of complexity, all the apparent edges could be set back from the actual edges to make room for a column at the corner. The column would simply be a section attribute sweeping the line which is the edge of the wall polygon. If the designer decides to poke holes in the column, Boolean operations of subtraction are performed and held as attributes at that and all higher levels of complexity.

Although the browser does not yet exist in the specified form, some research has been done in the area of network-based modeling, using hypothetical manufacturers of building components located at different areas in the world. The research reflects how a browser would function in its basic form.

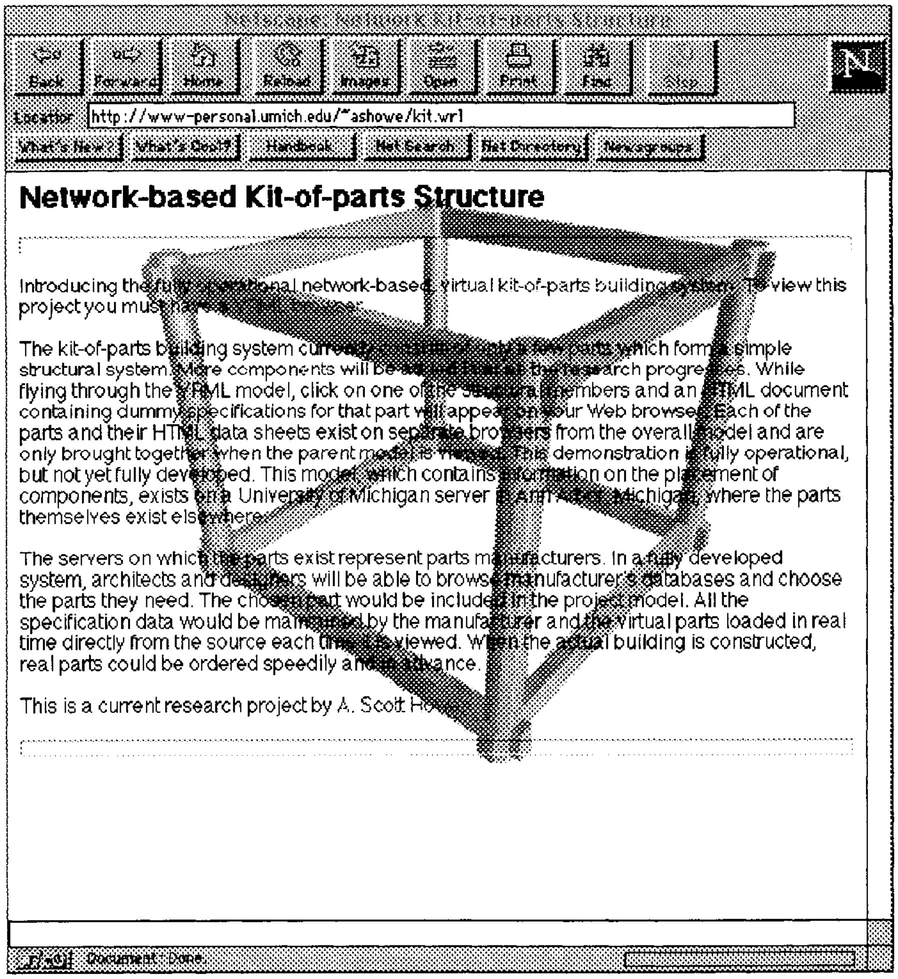

Figure 2: Introductory page for network-based building system

PART II: NETWORK-BASED BUILDING SYSTEM: KIT-OF-PARTS ASSEMBLIES

Although the concept will not be discussed within the scope of this paper, the term "kit-of-parts" needs to be defined. A kit-of-parts is a collection of discrete components that are pre-engineered and designed to be assembled in a variety of ways. Imagine the toy "Lego" and the concept should click. A kit-of-parts approach was chosen for this research for two reasons: 1) it is a clean way of demonstrating a network-based architectural visualization system without having to deal with the complexities of raw materials (though in actuality there should be no difference if parametrics are facilitated), and 2) the author is involved in research with kit-of-parts building systems in parallel with the work described in this paper.

The demonstration project consists of a simple structure which uses only three components: a column, a beam, and a joint. More variations of components will be added later as the work progresses, such as flooring, roofing, cladding, and mechanical equipment. At the time of this writing, the models of the three components exist on Kajima Corporation's server in Tokyo, Japan. The models are stored in VRML[15] format, which is a powerful network-based modeling language. The parent model is located on a server at University of Michigan and has no components in it at all, only pointers which will tell the browser where to find the parts. Kajima represents a steel manufacturer located in Japan, and Michigan represents the project's architectural firm.

By launching Netscape,[16] figure 2 shows the introductory page for the network-based building system.[17] The original of this document is in color so the hypertext links are clearly visible, but in the black and white printed copy the links might not be so apparent. In order to view the model, a VRML browser is required (Caligari Fountain[18] for the PC is recommended).

To download the model, click on the "virtual kit-of-parts building" hypertext. At this point the browser will connect to the Michigan server and parse the parent model file. With each pointer to components which appears in the parent file is an attribute giving placement and orientation. In this simple model there are four instances of the same column, eight instances of the same beam, and eight instances of the same joint. Figure 3 shows a small piece of VRML coding which contains the pointer to one instance and its placement coordinates and orientation.

The "matrix" section in the coding contains three attributes which represent axis rotation and a fourth attribute which is the absolute coordinates of location in the model. The "inline" section contains the pointer, giving an address on the Internet to go fetch the component named "column1.wrl," which is located in Japan. Each of the twenty instances of parts need no more than this to reference them in the parent model.

The information is scattered over the Internet at remote corners of the world, but using pointers the browser will be able to assemble it all together into a coherent whole. Since the information is assembled in real time, all of it is current and up-to-date.

After parsing the parent file, instances of the components are put into place and the simple structure appears, as in figure 4. Even before the model is loaded, the browser will allow

```
DEF column1_2 TransformSeparator
{       MatrixTransform
        {       matrix
                1.000 0.000 0.000 0
                0.000 1.000 0.000 0
                0.000 0.000 1.000 0
                5.000 0.000 0.000 1
        }
        DEF column1_2 WWWInline
        {       name
"http://www.kajima.co.jp/vrml/column1.wrl"
        }
}
```

Figure 3: VRML coding containing instance pointer and location

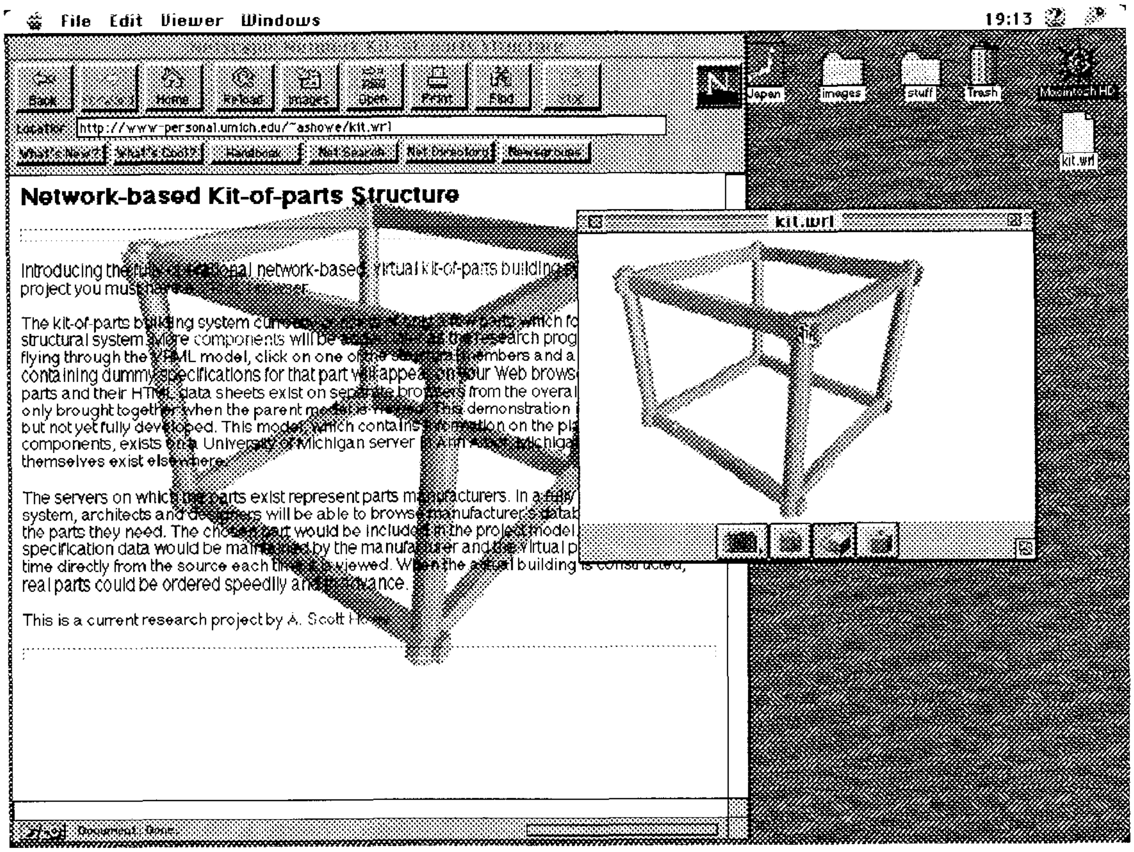

Figure 4: Browsing the VRML parent model

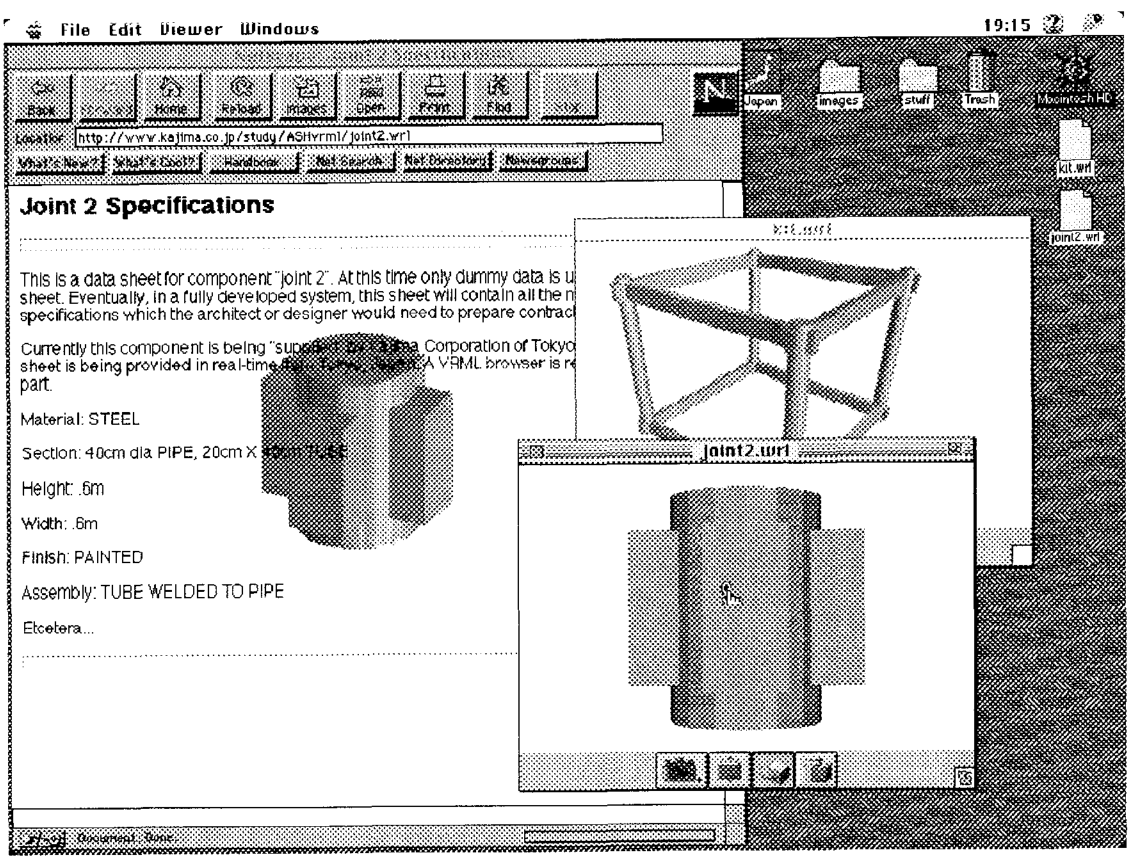

Figure 5: Searching the model for component specifications

fly-through navigation into the scene. Using the mouse in an intuitive manner, forward movement, backward movement, and turning allows the person viewing the model to wander around in real time.

Whenever the cursor moves through space and lands on one of the parts, it changes into an anchor icon. On some browsers the anchor icon is a small hand symbol. When the anchor icon appears it means that there is a source available for more information. By clicking on one of the joints, Netscape brings up a data sheet on the joint containing pertinent specifications and other information. The data sheets of each of the components of this study also exist on the Kajima server and would be maintained by the respective manufacturer or supplier.

Through the data sheet, the model of a single joint can be downloaded and viewed, as in figure 5. In the same way, data sheets for the column and beam can be viewed and the respective models downloaded for incorporation of their addresses into project databases.

The model in this demonstration is simple. The sequence is easily accomplished, but the implications are significant. With this unsophisticated but fully operational network-based kit-of-parts building system, it is hoped that the potential of real-time information sources and the immense computing resources available on the Internet will be obvious. The demonstration building system has some rudimentary characteristics necessary for the implementation of network-based virtual building technology.

VRML needs to remain lean in its content in order to facilitate speedy parsing and real time navigation for virtual reality visualization. VRML as a standard network modeling language is extremely powerful in its current version but could be greatly enhanced by allowing attributes and expanding level of abstraction capabilities. Level of abstraction is currently addressed by the LOD node, which stands for "Level of Detail." Expansion of the LOD capabilities must necessarily include levels that are not meant to be rendered but could function as schematic representations of objects. If these enhancements could be facilitated in VRML, the remainder of the data structure and browsing functions could be handled by the browser or modeler. Models could be standard VRML with attributes either pointing to enhanced structure at the end of the file or pointing to resources over the network.

CONCLUSION

Design and visualization would be enhanced with the development of virtual building technology using networks. An architectural visualization system that utilizes the resources of networks such as the Internet might effectively eliminate the potential storage problems that will plague large architectural databases necessary to contain virtual buildings. Keeping the responsibility of database management at the source, which is the materials suppliers and building component manufacturers, accurate up-to-date information can be accessed in real time, decreasing the amount of data required for a virtual building database.

Virtual buildings could be implemented using powerful network modeling languages like VRML with enhancements to the data structure. Implementation of a simple network-based kit-of-parts building system shows the practicality of network distributed models and information sources.

NOTES

1. A. Scott Howe is an architect with Kajima Corporation of Tokyo, Japan and is currently in the Rackham Doctoral Program in Architecture at the University of Michigan. Home Page URL:http://www-personal.umich.edu/~ashowe. E-mail: ash@ipc.kajima.co.jp
2. Edward F. Smith, "Virtual Buildings: Knowledge Based CAD Models for Design, Analysis, Evaluation and Construction," *Computer Solutions*, summer 1992, pp30-32.
3. Benjamin B Bederson, Larry Stead, James D. Hollan, "Pad++: Advances in Multiscale Interfaces," SIGCHI '94 short paper.
4. Microsoft Corporation, Excel 5.0
5. George W. Furnas, "Generalized Fisheye Views," *Human Factors in Computing Systems CHI '86 Conference Proceedings*, (Boston, 13-17 April 1986), pp16-23
6. J. Mackinley, "An organic user interface for searching citation links," *Human Factors in Computing Systems CHI '95 Conference Proceedings*, (ACM, 1995), pp.67-73
7. Sougata Mukherjea, James D. Foley, "Visualizing the World Wide Web with the Navigational View Builder," *Proceedings of the 3rd International World-Wide Web Conference*, (Darmstadt, Germany, 10-14 April 1995) Computer Networks and ISDN Systems; v 27 n 6, pp1075-1087.
8. James Turner, "Guide to Reading NIAM Diagrams," (The University of Michigan, Architecture and Planning Research Laboratory, Ann Arbor, Michigan, 22 May 1991).
9. James Turner, et.al., "AEC Building Systems Model," ISO TC/184/SC4/WG1, Document 3.2.2.4, (The University of Michigan, Architecture and Planning Research Laboratory, Ann Arbor, Michigan, 2 August 1990).
10. A. Scott Howe, "A Genesis System," *Special Research Report of the Sensitivity Engineering Product Development Research Group: A Sensitively Designed City*, (Japanese Ministry of International Trade and Industry, Tokyo, Japan, 1993).
11. Dassault Systemes, CATIA Concurrent Engineering System, Version 4, France.
12. Auto-des-sys Inc., Form Z, Version 2.6, Columbus, Ohio.
13. HSC Software Corp., KPT Bryce, Santa Monica, California.
14. Imagine Corporation, Shade III, Tokyo, Japan.
15. Mark Pesce, VRML Browsing & Building Cyberspace, (Indianapolis, IN: New Riders Publishing, 1995, ISBN 1-56205-498-8). VRML = Virtual Reality Modeling Language.
16. Netscape Communications Corporation, Netscape Navigator 1.1, Mountain View, California.
17. A. Scott Howe, Home Page, URL: http://www-personal.umich.edu/~ashowe/netkit.html
18. Caligari Corporation, Fountain, URL: http://www.caligari.com/ws/fount.html