DVIN

A DUAL-VIEW INFORMATION NAVIGATOR

Chien-Lin Chen and Brian R. Johnson
Design Machine Group, Department of Architecture, University of Washington, Seattle

Differences in the preferred modes of representation of architects and their clients create challenges to their collaboration in the design process. Traditional two-dimensional drawings such as plans, sections and elevations form the backbone of architectural representation, anchoring text labels to record relevant non-graphical information. Nominally geometric "slices" through the proposed building volume, these drawings employ abstractions and conventions unique to professional practice. In contrast, non-architects think about building configuration largely through experiential or photographic perspective.

This challenge increases over the life of the project. Simple drawings, such as those used in schematic design, are easily understood by all parties. However, as the building design develops the architects encode more and more design detail through the drawing conventions of construction documents, inadvertently making this detail less and less accessible to non-architects.

We present DVIN, a prototype system that uses coordinated plan and perspective views for navigation of building information models, linking the information to an individual's spatial navigation skills rather than their document navigation skills. This web-based application was developed using Java and VRML. The prototype makes it easier for naive users to locate and query building information, whether they are a client, a facility manager, or possibly an emergency responder.

BACKGROUND

Architects and clients generally collaborate, in the development of a design. The primary vehicles for this collaboration have historically been face-to-face meetings supported by physical models and paper documents produced by the architect for this purpose, but based on the evolving construction document database. Certain difficulties have always existed with this circumstance, which we will describe below, but as the profession of architecture moves to adopt a data-based representation of a design—a Building Information Model—these difficulties may be exacerbated. However, we believe that ongoing changes in both the power and pervasiveness of computing technology throughout the broader culture present opportunities to address these difficulties while leveraging the BIM model and enhancing communication between the two main players, as well as others.

We believe that the primary vehicle for future collaborations will be data rather than physical models and drawings, and that face-to-face meetings will be supplemented, if not replaced, with other mechanisms, such as virtual post-it notes (Jung and Do 2000), video conferences, and so on. Many challenges must be overcome before these techniques become routine. The current project addresses the challenge of accessing building information over the web. Two different impediments exist to this: a technical one involving file formats, locations, etc., and a cultural one involving context, meaning, and information.

On the technical side, while most clients have adequate computing power and network connectivity available to be "full-fledged" partners in the BIM process, they soon discover that the models are large, the software is expensive, and the software is complex, requiring significant training to use. Further, even with the resources and will to acquire the necessary software, the orientation of these systems to the needs of professionals runs directly into the cultural problems of design data.

The Dual View Information Navigator (DVIN) project has two core objectives. The first, addressing the cultural challenge, is to show how a dual-view system using coupled plan and perspective views can help non-professional users locate specific building information. The second, addressing the technical challenge, is to create a platform, using readily available components and standards, for clients to use to access building information over the web, without the limitations imposed by proprietary software packages.

CHALLENGES FACING CLIENTS

Recorded marks (whether pen and/or pencil marks on paper or via digital media) are central to communication and thinking in the design process. Graphic methods have been used to visualize and exchange design information since at least the time of the Ancient Greeks. Graphic representation not only plays an important role in architecture but also in other design domains such as visual design, and mechanical engineering. While 3-D visualization is mainly used during conceptual design, it is also prominent throughout the building design and construction process.

Digital media and tools provide great potential for architects or professional users to store, present, simulate and think about the building information. However, it is difficult to present 3-D worlds via flat media such as screen or paper. Nonetheless, most of our construction documents, sketches, 2-D CAD files, rendered images, and digital photographs, are all representing 3-D objects in flat two-dimensional media (Issa et al. 2003). Unlike architects, non-professional users have not had the opportunity (or need) to learn the use of these tools (software), nor have they had practice "reading" complex building information from the combination of plans, sections, elevations, legends, and text labels.

Furthermore the interactions between human and digital information are not always intuitive, especially for non-professional users, because we usually perceive information through the combination of different senses such as sight, sound, touch, feel, and smell. As a result, it is difficult for non-professional users to use these complex computer-aided systems to access building information. After the development of the NCSA Mosaic Web browser in 1993, the pervasive point-and-click interface of web browsers and hyper-media has found wide acceptance. The "browser paradigm" may present a means by which to access digital information visually (Geisler 1998), and this may also help non-professional users to access building information via digital visualization.

Finally, while traditional drawings may provide a comprehensive way of representing geometric information for architects, such representations are often difficult to comprehend for non professional users. There are several variables that influence human spatial representation (Mark 1993):

1) CULTURAL AND LANGUAGE DIFFERENCES

The linguist B. L. Whorf wrote, "We cut nature up, organize it into concepts, and ascribe significance as we do, largely because we are parties to an agreement to organize it in this way—an agreement that holds throughout our speech community and is codified in the patterns of our language" (Whorf 1956). That is, we interpret our environment based on culture and language—factors that need to be considered when people perceive their environment.

2) DISCIPLINARY AND EXPERIENTIAL DIFFERENCES

Professionals are often trained to see and categorize the world in particular ways. This training influences how people describe the world in their professional practice. An architect may focus on aesthetic and perhaps spiritual value, and a craftsman may focus on different materials. Different focuses will influence the data of perception and internalization.

3) INDIVIDUAL DIFFERENCES

There are also individual differences in cognition. Some of these may be correlated with handedness or gender. Some may be associated with physiological difference, but nevertheless individual variability in spatial tasks is high (Gould 1989).

RELATED WORK

For architects, building information consists of several different kinds of data. For example, design documents usually include a great quantity of 3-D geometry data in 2-D drawings. They also include lists of materials used in the construction. It is hard to represent the variety of data in a single style or format. As a result, architects have traditionally used a mix of models, rendered images, drawings, and sketches to present their concepts. In the digital world, digital models, drawings, and rendered images are usually presented in a style similar to traditional presentation; however, the reliance on screen "snapshots" presented via media which are limited by the resolution of the program, projector, or monitors has negative influence for client comprehension of the project (Atkinson 2005).

On the other hand, the availability of Internet connections allows people to communicate quickly and easily over distance. On the Web, simple "point and click" interfaces can seamlessly interlink relevant data in an integrated environment (Rohrer and Swing 1997). Architects sometimes present their design projects on the Web using static renderings, plans, sections, text, and "walk through" movies. While accessible, these present clients with problems similar to the "in office" presentation—they cannot explore the buildings freely and they need to compare different data in different pages. Furthermore, information visualization often means representing abstract data in physical space and it improves working efficiency if the abstract data are mapped into a physical space appropriately. While visualization tools such as VRML or QTVR may be suitable for presenting 3-D environments, it is hard to present text or lists inside the scene, and people might lose the sense of direction in these scenes.

MULTI-VIEW DISPLAY

A multiple view system uses two or more distinct views simultaneously to support the investigation of a single conceptual entity. This approach is commonly used in technical applications such as computer-aided architectural design (CAAD) systems, as well as web-based route-finding and in-car navigation software. Usually, one view provides an overview for context and the other provides a zoomed-in-view for details (Baldonado, Woodruff and Kuchinsky 2000). These systems usually provide users with information that helps them make a variety of design decisions, ranging from determining layout to constructing sophisticated coordination mechanisms. The impacts of multi-view systems have been found to be:

- Reduction in time required to learn the system;
- Reduced load on the user's working memory;
- Increased exertion when comparing views; and
- Increased cognitive load when switching contexts.

While some impacts might be negative; the net result of deploying a multi-view system may be to improve usability significantly.

There are also some impacts on computer system requirements when employing multiple views:

- Computational requirements for rendering the additional display elements, and
- Display space requirements for the additional view.

PREVIOUS PROJECTS

DVIN continues a recent focus of our research group towards making building information more accessible to individuals and groups conducting reviews, *CAPRI* (Lee and Johnson 2006), and extends two earlier projects related to web-based feedback: *Immersive Redliner* (Jung and Do 2000) and *Spacepen* (Jung et al. 2001). CAPRI focuses on multi-participant access to data using a

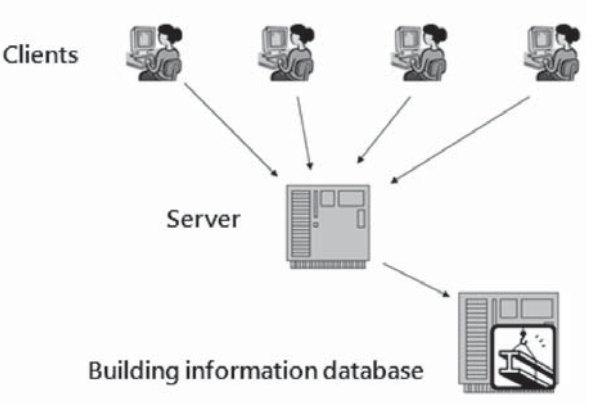

FIGURE 1 DVIN system overview

tangible interface, while *Immersive Redliner* provides a Web-based environment in which users explore design alternatives, "walk through" and inspect a 3-D model, change location, color, and texture of objects or place text notes on surfaces. *Spacepen* is also a Web-based project that utilized a pen-based interface to sketch in a 3-D environment, creating an easy-to-use pen-based interface in a shared collaborative environment.

THE ROLE OF DVIN IN THE DESIGN PROCESS

In the early phase of a design process, architects draw rough concept sketches to represent their ideas. Although these sketches usually are abstract and highly conceptual, co-workers and clients can still "read" these after suitable verbal introduction and due to the fact that these sketches do not contain much explicit detail. As the design develops, designers transform the rough concept into a more developed project by repeatedly modifying, enhancing, and refining; the result is that more and more complex building information is included in the design documents. As the design develops, this information becomes more and more detailed and esoteric. As a consequence, clients and consultants are sometimes challenged when they are asked to review the project through these design documents and provide feedback. People without professional background find it difficult to follow cross references, read overlapping information, etc. Nonetheless, the client's feedback is essential for architects, as they can produce a better design solution only when they get useful feedback from clients or consultants. The appropriate information may be available in the design database (or BIM), but it is difficult for non-professional users to access building information through professional software.

DVIN is designed to help non-professional users access complex building information. If clients and consultants are able to access building information fluidly, they could provide useful input for architects to refine their designs. The DVIN strategy is to use linked plan and perspective views to help non-professional users extract the designed building information from abstract design documents without any document navigation skills or proprietary software.

DVIN SYSTEM OVERVIEW

DVIN executes on the client's computer through a common Web browser, but the "back end" of the system resides on the Web as a Web server. The Internet provides an easily accessible platform to exchange and store information regardless of distance, so the client and the database can be in different cities or countries. At this time, extraction of BIM data is done manually—geometry must be exported as VRML models and linked to related building information through regular hyper-links. These three components are connected through high speed Internet (Figure 1). Clients can access information via the web and interact with it (e.g. delve into detail, leave comments etc.).

DVIN INTERFACE

The DVIN interface consists of three major components: plan view, perspective view, and information window. The plan and perspective views of DVIN are generated from the VRML model automatically. They provide simple locational information (plan view) on the left side and a real-time rendered image (perspective view) on the right side.

The perspective view is generated in real-time. It does not utilize pre-rendered images or movies. This means users can roam freely through the model, looking at any room or examining any detail of the design. Navigation in the perspective view is done using arrow keys; selection uses the mouse. While moving about in perspective, a small marker, located in the plan view, indicates the precise view point and orientation of the current perspective view. This enables the user to easily understand where they are and what they are looking at. The plan view may also be zoomed and panned for greater context or detail.

The perspective view is the main access point for building information. Objects in this view may include associated hyperlinks to other web pages or databases. Users can click an element (e.g. window, door, lamp, etc.) in the perspective view and further information will appear in the information window automatically. This manipulation of DVIN is similar to "surfing" on the web. As a consequence, users can operate DVIN without any training.

The information window, a java browser (Java Example), is located at the bottom of the browser window. The current system makes no fixed assumption about what would be displayed here. The window simply provides a means for the display of building information that cannot be expressed easily using shaded perspective images or plan drawings, and it will respond when users click any specific object or icon in the perspective view of the DVIN. Most of the time, the system might link to a manufacturer's database or webpage. From there, users could query the manufacturer directly. Information windows can be manipulated as any web browser. Users can click a link inside of the window, opening an additional information window. Dual information windows give users great convenience, and help them read and compare related information between two information windows.

Using DVIN users can browse their building project

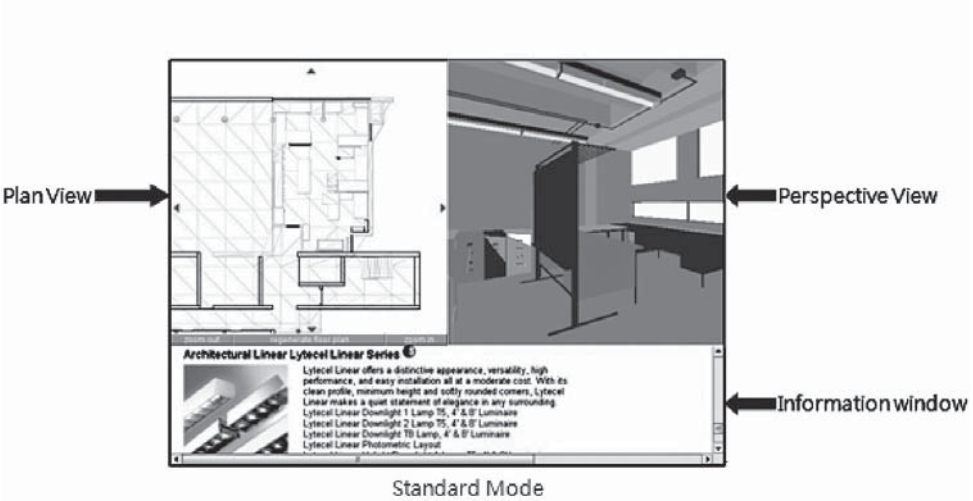

FIGURE 2 DVIN Interface

via the dual-view interface. Through their direct participation in navigating they will reinforce their comprehension of the design. At the same time, they may query the building database by clicking on objects in the perspective view such as we do in Web browsing. The related information will then be shown in the information window. In a word, DVIN helps non-professional users access building information without specific training because users can manipulate the system based on daily experience, while providing a "what's that?" through the database.

SYSTEM REQUIREMENTS OF DVIN

This section will briefly discuss the system requirement of DVIN from Client, Server, and database.

1) CLIENT

The goal was to keep the client-side very simple, so that users could launch DVIN without proprietary software. While DVIN requires that several components be installed on the user's computer, all of them are free, including JDK (Java Development Kit) 1.5, JRE (Java Runtime Environment), Java3D, and vrml97 loader package. After downloading and installing these components, users can launch DVIN through their web browser as part of a design review.

2) SERVER

The DVIN application and the target VRML data files can be hosted on any web server. Certain restrictions currently exist for the VRML files. DVIN's VRML loader can load VRML2.0 models with preset surface colors, image-mapped textures, shininess, and transparency, but they must not contain viewpoints (which cause a crash).

3) BUILDING INFORMATION DATABASE

DVIN implements a simple web browser for information display in addition to the VRML model-display viewports. It can handle standard HTML with CSS (Cascading Style Sheets). As a result, the information window can present or query further information from the manufacturer directly. This functionality may be enhanced or supported on the architect's Web site through links from BIM object instance data to online product data.

THE DVIN IMPLEMENTATION

The implementation of DVIN can be divided into 2 parts:

1) IMPORTING AND INTERPRETING 3-D VRML MODELS

The VRML Loader Package used in DVIN is a part of the Java 3D API (Sun Microsystems Java 3D Engineering). It allows a program to import a VRML model into a Java 3D universe. However, it is not sufficient for DVIN if we just import and display the VRML model. Each object in the perspective view needs to be clickable and selectable, and this behavior does not occur in the Java 3D API automatically.

Figure 3 shows the structure of loading the VRML model, recognizing shapes, and making all surfaces clickable. In fact, shapes in the VRML models must be counted and analyzed before showing in the perspective view. All geometry (boxes, cylinders, planes, triangles, and spheres) needs to be associated with a PickTool class in order to be clickable. Then the system only discerns the IndexedFaceSet of VRML geometry that most 3-D modellers use to describe the entire VRML scene. Finally, DVIN allocates each surface in the model intersection capability by means of the PickTool class thereby allowing the system to detect which surface is selected (clicked on) by a user.

2) WALKING INSIDE THE MODEL

Figure 4 shows how the walk behavior works inside DVIN. A TransformGroup object is created and receives the result of two transformation matrices; it substitutes for the original ViewingPlatform in the main class (The ViewingPlatform class is used to set up the view branch graph of a Java 3D scene graph in a SimpleUniverse object).

The VRML browser of DVIN does not support collision detection and gravity by default (meaning users might "fly" when they try to go down or "pass through" stairs they would like to go up). DVIN addresses this problem by detecting the height of objects in front of users. The users' viewpoint then changes depending on the height of object. If the object is short, like a stair step, the object is treated as a step and the system will change the height of viewpoint in the vertical direction. However, if it is significantly higher or lower, the system treats it as an obstacle and keeps the viewpoint at the current height. If the viewers elevation has changed, a new plan view is cut through the model.

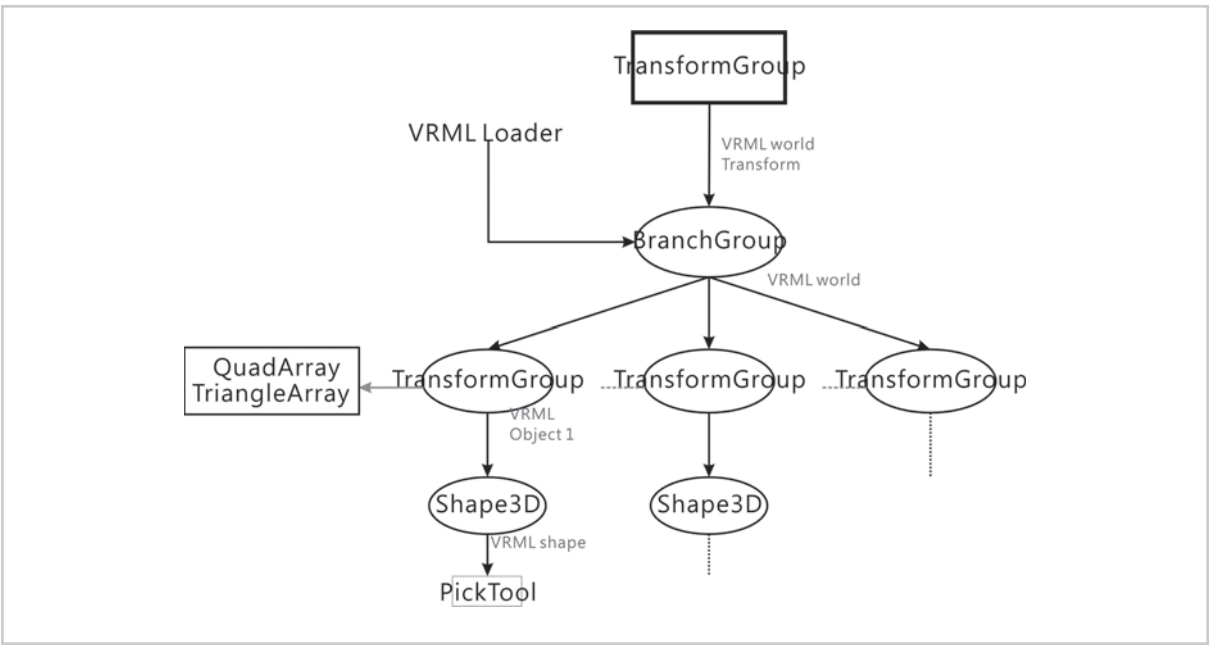

FIGURE 3 Hierarchy of the VRML model

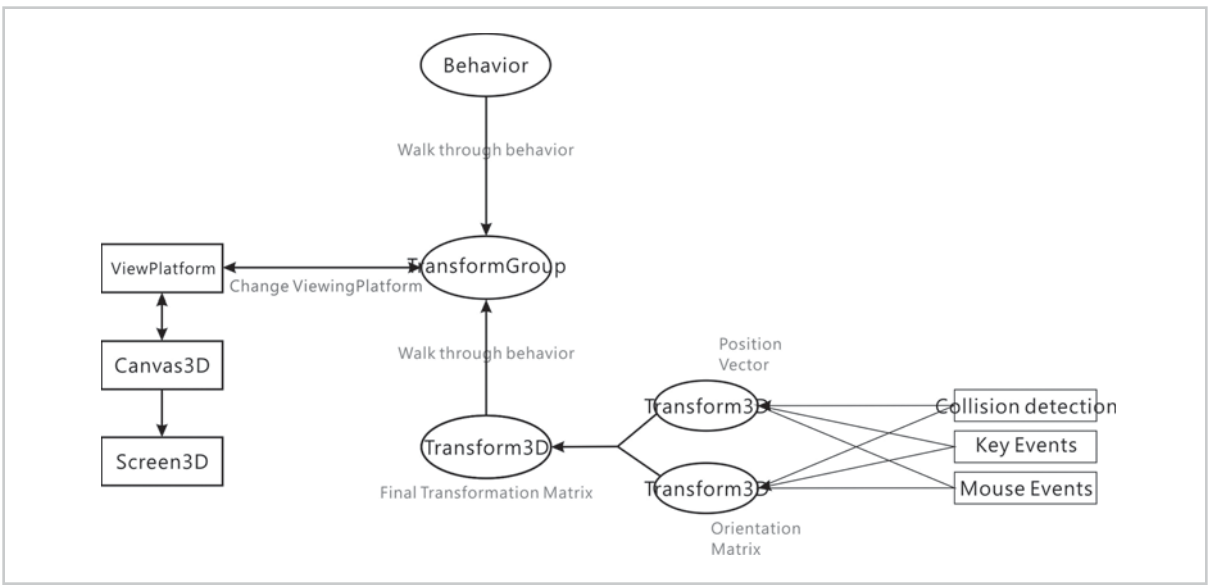

FIGURE 4 Relation of Java events to viewpoint changes