28 Collaborative Design System with Network Technologies in Design Projects

Claude Comair, Atsuko Kaga and Tsuyoshi Sasada

Osaka University, Faculty of Engineering,
Environmental Engineering (Sasada Lab.), Osaka 565

This paper depicts the work of the team of researchers at the Sasada Laboratory in the area of collaborative design and the integration of global area network such as the Internet in order to extend the architectural studio into cyber-space.

The Sasada Laboratory is located at the University of Osaka, Faculty of Engineering, Department of Environmental Engineering, Japan. The portfolio of the Laboratory is extensive and impressive. The projects which were produced by the men and women of the Laboratory range from the production of databases and computer simulation of several segments of different cities throughout the world to specific studies of architectural monuments. The work performed on the databases was varied and included simulation of past, present, and future events.

These databases were often huge and very complex to build. They presented challenges that sometimes seemed impossible to overcome. Often, specialised software, and in some cases hardware, had to be designed on the "fly" for the task. In this paper, we describe the advances of our research and how our work led us to the development of hardware and software. Most importantly, it depicts the methodology of work which our lab undertook. This research led to the birth of what we call the "Open Development Environment" (ODE) and later to the networked version of ODE (NODE).

The main purpose of NODE is to allow various people, usually separated by great distances, to work together on a given project and to introduce computer simulation into the working environment. Today, our laboratory is no longer limited to the physical location of our lab. Thanks to global area networks, such as the Internet, our office has been extended into the virtual space of the web. Today, we exchange ideas and collaborate on projects using the network with people that are spread over the five continents.

KEYWORDS: Computer Assisted Design, Computer Languages, Computer Generated Databases, Computer Graphics, Three-Dimensional Computer Simulation

INTRODUCTION

The aim of our research is to facilitate the collaboration among the various people involved in the design process of an urban or architectural project. This includes various designers and engineers, the client and the citizens who may be affected by such a project. For this purpose, we have conceived and assembled a system of software and know-how which we call the Open Design Environment (ODE) and the Network version of ODE (NODE).

NODE is a system that includes presentation tools, review and revision tools, and design tools for the production of all the final material related to the design project.

However, our research has proven to us that there are two major issues to consider when working with a group of people that is spread over a large geographic area. These two issues are related to decision making (initiatives) and project co-ordination (timing). If these two issues are well balanced we can achieve a good and smooth communication among the various people concerned with the project. NODE attempts to resolve these two issues simultaneously, and not independently.

Recent progress in hyper medium and network technology gave us new solutions to simultaneously address the issues related to work co-ordination and decision making. NODE benefited from new technologies such as:

- URL (Uniform Resource Locator),
- VRML (Virtual Reality Modeling Language),
- HTML (Hyper Text Mark-up Language),
- API (Application Programming Interface),
- 3-D object modelers and 3D renderers on the network.

This paper aims to present NODE in action showing how it implements a smooth collaboration among the various parties involved in the design process. As mentioned earlier NODE uses the latest technologies that are available on global networks, such as the Internet, in order to extend the architectural studio into cyber-space.

1.0 TOTAL DEFINITION OF OBJECTS

There are two major issues when dealing with communication among various people that are trying to solve a common problem. First, they have to be able to define the problem, and second, they have to be able to communicate using a certain language or protocol. Architects and engineers, for example, use various symbols when working on their drawings, the various meaning of these symbols are well defined and standardised. These symbols are used to define buildings and their contents. Once the building is erected, architects are no longer concerned with any change that does not affect the architecture of the building. For example a change in the ownership of the building is of little importance to the architect but of crucial importance to the city office. Therefore, we cannot limit the definition of an object of the real world to the sole description of its shape or even less, to the resolution of a computer screen. Total definition of objects is even more important when defining databases that would be used to simulate events of the real world. In order to totally define an object of the real world in the database, we should include in its description virtually everything we know about the object. In the case of the definition of a building for example, the object must include the dimensions, colour, material, the name and address of the management/owner, the list and names of the inhabitants, the name of the various service companies, and more. Many computer simulation environments limit the definition of an object in the database. CADD (Computer Aided Drafting and Design) systems, for example, limit the definition of the object to the its shape data textures and colour. This would be of little help if we are simulating a disaster situation such as a fire or earthquake, for example. Emergency

information necessary for quick damage estimates and evacuation planning would be impossible to retrieve from a CADD database.

2.0 INTELLIGENT DATABASES

Total definition of objects of the real world implies the inclusion inside their databases' decision making statements or flow control statements. Flow control statement are usually supported by computer languages and are loop statements and conditional jumps. Loops allow the repetition of events while conditional jumps allow decision making. By adding these capabilities to the objects of the database, we add artificial intelligence to the definition of the objects. By doing so we transfer the control from the running software or interpreter of the database to the database itself. Consider the following examples: the first example depicts a static object versus Example 2 that does the same job in a more concise way using the "for" loop which is a flow control statement. Furthermore, for the sake of simplicity, plain English is used in the definition of the two objects.

```
Example 1:

create object DumBoxes

        {

        cube (1,1,1);

        cube (2,2,2);

        ...

        cube (9,9,9);

        cube (10,10,10);

        }

sample 2:

create object Intelligent Boxes

        {

        for (i = 1; i<=10;i++) cube(i,i,i);

        }
```

The "for" loop used in Example 2 is understood as follows:

for i = 1 and as long as i is less or equals than 10 do the cube of size (i,i,i) while incrementing "i" by i (i++) after each iteration of the loop.

Example 3 builds on Example 2 by adding a decision making statement to allow the modified object, Intelligent Boxes, to show spheres instead of boxes if the viewing camera is closer than 20 feet.

Example 3:

```
create object IntelligentBoxes
        {
        if (camera < 20 feet)
                for (i = 1; i<=10;i++) cube(i,i,i);
        else
                for (i = 1; i<=10;i++) sphere(10,10,i);
        }
```

In the previous examples, flow control statements were included in the database in order to simulate decision making and automatic repetition of certain events. This gave the objects shown in the previous examples some kind of an artificial intelligence or behaviour. Therefore, data objects were able to behave differently under different conditions. This decision making capability was embedded into the database of the objects and not into the interpreter. This fact has many advantages. Only a few will be named in this brief presentation:

- This fact allows us to write data that can mimic real life situations.
- Building databases is costly and complex. It is in the database where most of the investment should be made and should remain. Usually large databases are costly and time consuming to build. Such databases are meant to survive the test of time and the changes made to hardware and software. This means that transporting the database from a technology to another must be made easy. If all the decision making data as well as static data are embedded into the database, all we need to transport the database to a newer environment is to re-write the interpreter for the new target technology. Databases of cities certainly fall under this category since they are usually extremely large and must be updated on a regular basis.
- Flow control allows us to produce very concise databases as we have seen in Examples 1 and 2 above.

3.0 COMMUNICATION AND NETWORKING

It is now evident that in order to share the objects of the databases among the various people that are concerned with the development of the final project, and decision making people that may affect the final outcome of the project, an elaborate networking environment must be set in place. This environment is made of hardware (computers, routers, modems, and cables) and supporting software.

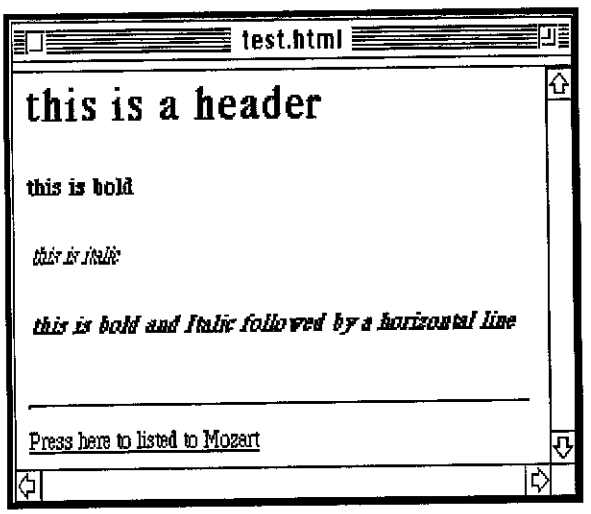
In our research we opted for one of the most affordable solutions. We used personal computers connected to the Internet, loaded with hyper browsers, several 3-D object modelers and renderers depending on the task in the design processes, and information databases. The designers and clients exchanged information and

opinions over the network and electronic storage media in order to complete the collaborative experimental design.

Communicating over the Internet is quite simple and could be compared to using a telephone. The user uses a specialised software usually called a browser on his/her computer to dial the unique number of another computer. The dialing computer in such a case is called the "client". The contacted computer, usually called a "server", has a specialised software running on it. As its name suggests, the server responds to the various requests of the client. In most cases, the server is loaded with various files that are made available to the client. Servers would be of limited help if they answered requests of a single user one at a time. Servers are usually multitasking machines. This means that they can service several users simultaneously. Therefore, several users connected to the same server could be reviewing the same file. This allows them to be working on the same file towards a final result, hence collaborating together.

According to the kind of data the clients would like to access on the server, they have to use the appropriate browser. There are several commercially available browsers for hypertext mark-up languages (HTML) for example. NetScape and Mosaic are some of the most popular. HTML documents are stored on a server running an HTTP (Hyper Text Transfer Protocol) deamon. The deamon would respond to the request sent by the browser by sending the appropriate file. The HTML file may contain only text, but as the name HTML indicates, the text could be marked to appear in a certain specific fashion. In order to demonstrate the marking of text in an HTML file, consider the simple HTML code:

```
<Title>test.html</Title>
<h1>this is a header</h1>
<P>
<B>this is bold</B>
<P>
<I>this is italic</I>
<P>
<B><I>this is bold and Italic
followed by a horizontal line</I></B>
<P>
<hr>
<A HREF = MOZART1.MOV>Press here to listed to Mozart</A>
```



In the above example normal text is surrounded by mark-up tags. The meanings of the tags used in the above example are as follows:

<Title> : Starts a title string to be displayed as the title of the window.

</Title>: Closes the previous tag.

<h1>: Starts a level 1 header.

</h1>: Ends it.

<B>: Bolds the enclosed text.

</B>: Ends bolding.

<I>: Italicises the enclosed text.

</I>: Ends Italics.

<hr>: Draws a horizontal line.

<A>: Is an anchor that could send you to another area in the same file or to a designated area in another file on the same server, or to another location of another file on another remote server. In this example, the anchor will load the local file entitled MOZART.MOV. The special character ">" which follows the name of the file starts the text that would appear highlighted or underlined indicating the existence of a hyperlink.

</A> Ends the anchor.

In this case, clicking the mouse pointer onto the anchor would cause the server to send the music file MOZART.MOV over the network, and hopefully the receiving browser can interpret this file and play it. In the same way, one could view graphics and play digitised video, as well.

Digitised video images could be assembled in a panoramic fashion to represent a 3D world. Then using a special software, such as QuicktimeVR, the viewer can actually "move" into the constructed world. It is obvious that in order to accomplish a realistic and seamless 3D world from a collection of 2D images, a fair amount of planning is necessary, but the result could be very convincing.

It is also possible to view and fetch 3D data that is located on remote computers over the Internet. Special browsers and description languages have been developed and are now readily available on the Internet. One example of a technology that is becoming quite popular over the Internet is the Virtual Reality Modeling Language or VRML. Web Space is an example of a VRML browser.

Various real-time or near real-time video teleconferencing software is also readily available on the Internet. We have experimented with these technologies and found them to be very effective in connecting together various groups of people collaborating on the same project.

The only problem with this piece-mill assembled system is that it is a general purpose communication system. It is suitable for most situations, but it fails to address certain specific needs related to the fields of architecture and urban planning, mainly in the areas of total definition of objects and imbedding intelligence into the definition of these objects. These are specifically the two topics that were presented earlier.

Architectural databases along with databases for urban projects are often complex and huge. It is often the case that such databases are spread over many computers. It is also often the case that the objects contained in such databases are themselves made out of a collection of simpler objects that could be spread over several computers. Our group has made clear advances in this area of creating and maintaining very large databases of several cities in Japan and around the world. Some of these projects are listed in the table below:

- **Osaka City (Feb. 1983)**
- **Portions of the subway system in the city of Osaka (1983).**
- **Kyoto (Aug. 1983)**
- **Shinjuku, Tokyo (Oct. 1983)**
- **Portopia, Kobe (Mar. 1984)**
- **Design work of Media City Osaka (1984)**
- **Renewal project in Sanda City (1984)**
- **Various animations for the Japanese pavilion at Tsukuba Expo 1985 (Dec. 1984)**
- **Various animations for Parigraph '85**
- **ABC complex Osaka (May 1985)**
- **Sewer System in Osaka (Feb. 1986)**
- **Kobe new transportation system (Mar. 1986)**
- **Shanghai City (Oct. 1986)**
- **New Kansai International Airport (1987, 1993)**
- **Parthenon (1989)**
- **Senri International Business Complex (Sept. 1990)**
- **Japanese Railway (JR.) animation (1991, 1993)**
- **Animation of the site of Tae Jon Expo '93 Korea (Sept. 1991)**

A new 3D computer language has been developed at our Lab. It is called Vu (pronounced Vee-You). Vu was developed in the early 80's by C. Comair while conducting his research at Osaka university in the Sasada Lab. Vu is a modern computer language. It supports most of the features found in general programming languages. Vu is a problem oriented computer language. It offers a specific support to the creation and maintenance of city planning related databases. Vu has evolved over a period of more than ten years and it is still under constant upgrade.

Once defined, Vu objects can be submitted to a variety of mathematical transformations in order to achieve various simulations. Furthermore, Vu objects are organised into Vu catalogues. Vu programs may contain several Vu catalogues, but can only contain a single Vu project. On the other hand, Vu library files contain only Vu catalogues. The Vu project actually calls the necessary Vu objects from the various Vu catalogues found in the current program or from the various Vu catalogues found in separate Vu library files. Furthermore, Vu files (libraries and programs) could be located on several Vu servers.

Vu servers are computers that are running a special Vu deamon that waits to be contacted by a client running a Vu interpreter. The Vu interpreter uses the "import" Vu statement to request from the deamon the dispatching of single objects, catalogues, or entire files. The "import" command uses the following syntax:

```
import
        {
        domain ("digipen.com");
        server ("iris");
        path ("/usr/comair/vu/furniture.lib");
        catalogue ("famousChairs");
        object ("RedAndBlue");
        }
```

If the object specification is omitted, then the entire catalogue is imported from the remote server at the domain. On the other hand, if the domain and the server are omitted, then the local machine is assumed. The description of the Vu language is far beyond the scope of this paper. Interested readers will find more information about Vu in the publication listed below. The section below lists some of the properties of the Vu interpreter.

4.0 THE VU INTERPRETER

Vu is a three-dimensional programming language. It provides the user with the following:

- Loops, case statements, and conditional jumps to alter the normal execution of a program.
- Procedures or Objects along with catalogues and projects to store and re-execute portions of the code. Furthermore, Vu Objects can be indexed and may be declared with default parameters. This allows the creation of "intelligent objects".
- Various types of variables (double, strings, vectors, matrices, cameras, large ascii entities, and large binary entities) and a rich array of operators.
- A large set of 2D and 3D primitives.
- A set of commands to control the hypothetical camera and viewport.
- A large set of mathematical functions, including vector and matrix operations.
- The possibility of importing and exporting several types of files.

- The possibility of creating 3D objects embedding more than the 3D data. Objects could contain any kind of information (price, volume, etc.).

- A 3D modeling stack, and a set of commands to manage the stack. The stack allows the creation of complex transformations that will be applied to the 3D entities; thus, creating different worlds where objects are temporally affected.

- Vu allows the easy creation and maintenance of large 3D databases. These databases can be used to represent any 3D object regardless of the size, complexity, or the nature of the object. Vu has been used to present objects as simple as cubes, vases, and tables, and as complex as planes, trucks, houses, and cities.

- Vu transfers the control from the application to the database. This means that the database controls the flow of the execution of the program and not the application that is reading the database. This allows the creation of concise but powerful databases.

- Vu allows the distribution of the database over several servers, thus, allowing objects to be reused and shared among several databases.

CONCLUSION

As a conclusion the various phases of the Schematic Design of the Mikata Jomon Park are presented. This design project uses the Internet in order to expand our Laboratory into the virtual space of the Internet.

1. Outline

By Mikata Lake near the centre of the coastal area along the Japan Sea, the Mikata Jomon Park is located. The Mikata Jomon Park is made up of the public facilities which include the museum of Jomon, the training facilities, and recreation facilities.

Before our schematic design, there was a proposal by an architectural firm. The client, which is the local government of Mikata Town, asked us to review the design. We reviewed it using 3-D models, computer generated images and movies made of a photo-montage and viewed using the QuickTimeVR technology. It is then that we discovered some of the design problems. This lead us to propose some revisions.

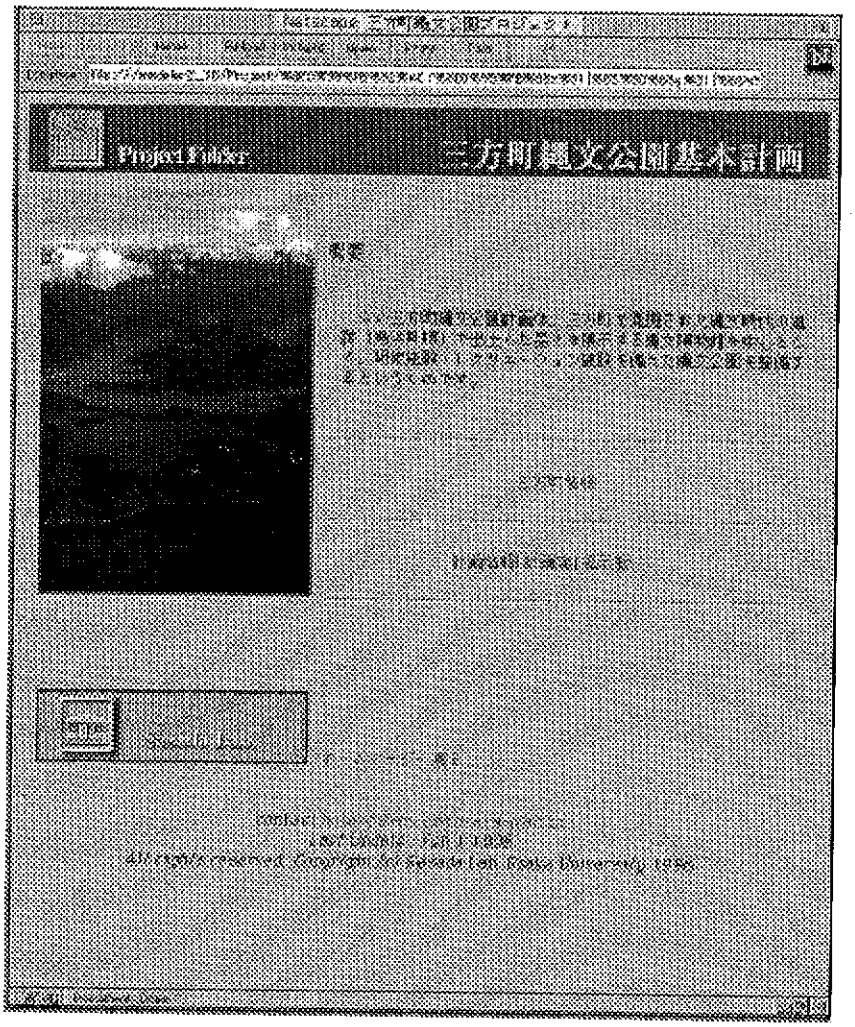


Fig. 1

We used 3-D models and computer generated images from the very early stages of the design process. In the production of the schematic design, we used very simple 3-D models and renderings. We communicated the results to the local government using the World Wide Web and the Internet. (Fig. 1)

2. System Structure (Fig. 2)

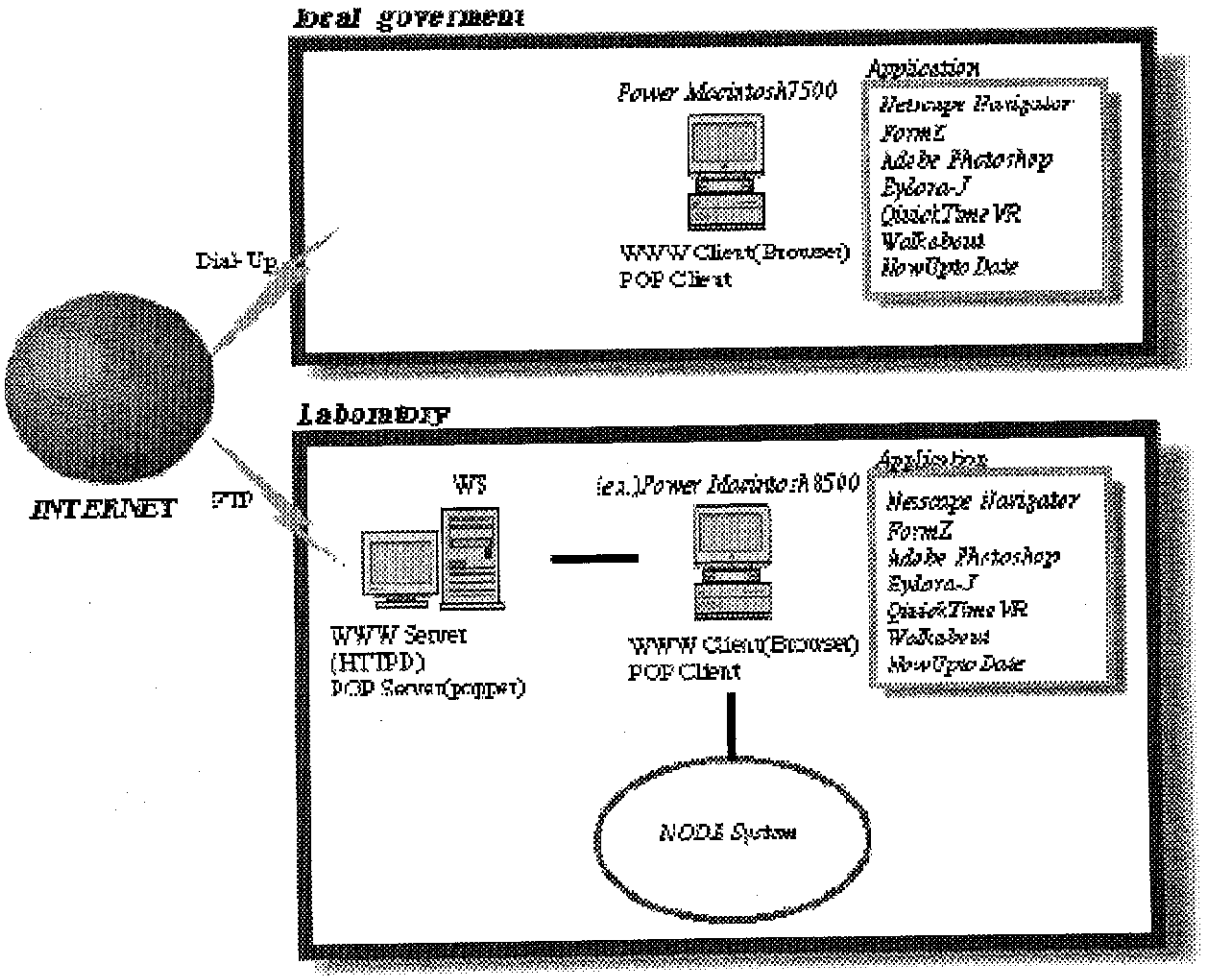


Fig. 2

Shows the schematic of the connection we used between our Laboratory and the local government in Mikata Town. The electronic connection was based on the Internet.

3. Sorted Information of the Schematic Design (Fig. 3).

This figure shows how we sorted the design information, site condition, schematic design condition, and reference materials. We separated the site's conditions from the natural and social conditions.

The sorted information of the schematic design on the World Wide Web is as follows:

- Natural conditions
- Pictures and QuickTimeVR movies in the field
- Direction of wind
- Water level of Mikata Lake
- Social conditions
- Population in Mikata Town
- 'Jomon' Festival
- Transportation
- Schematic design condition
- Previous design proposal by an architectural firm
- Contents of facilities
- Reference materials

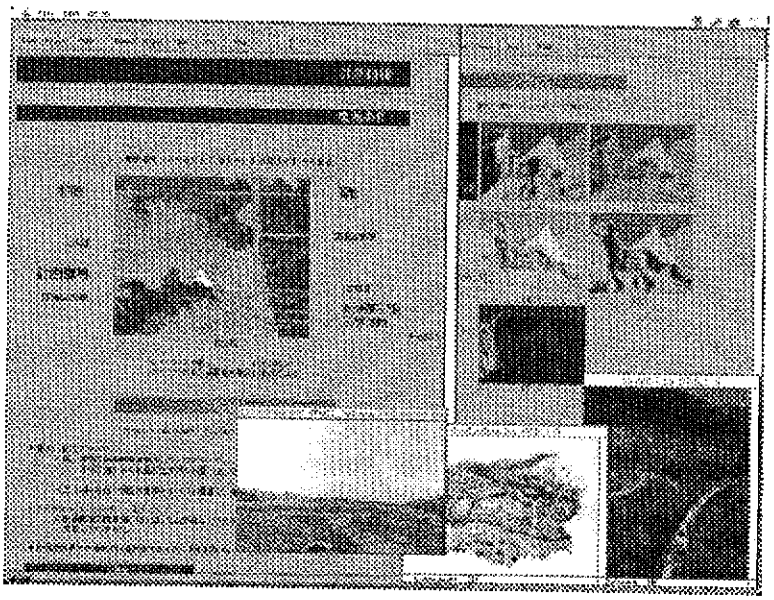


Fig. 3

4. Scheme Database(Fig. 4)

Users can review in real time the proposed scheme as a multimedia presentation. 3D-models can be checked using 'Walkabout', a 3DMF data browser. Images can be checked using 'Adobe Photoshop', a 2-D graphic application. Therefore, it is easy to check the scheme anytime, anywhere.

5. The Architectural System

5-1. Schedule System(Fig. 5)

The schedule for the project was put on the World Wide Web. Therefore, the schedule was readily available to Sasada Laboratory and local government simultaneously.

5-2. Security(Fig. 6)

We divide the information into two categories: information that was open to the public and information that was kept private. For example, the town guide was accessible to all, while the schematic design was open only to the persons and parties concerned in the design process. We ensured the security of the private information by setting up a login filter for the HTTP server.

5-3. System for Sending Comments (Fig. 7, Fig. 8)

Figures 7,8 show the system for sending and exchanging comments about the schematic design. This includes comments, images and 3-D models. We use Eudora, an electronic mail application, similar to MIME (Multipurpose Internet Mail Extensions) to send multimedia data which includes images, movies, sounds, and other file formats as well.

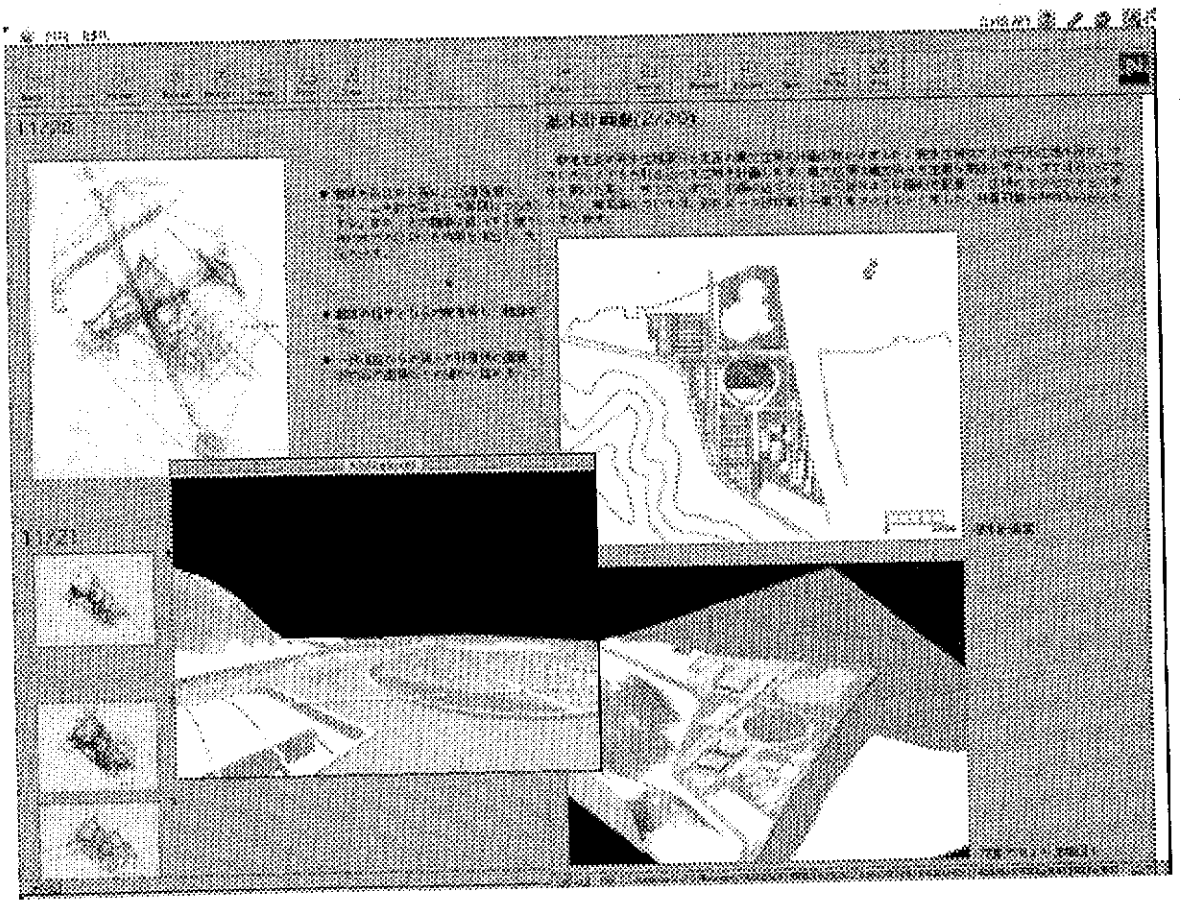


Fig. 4

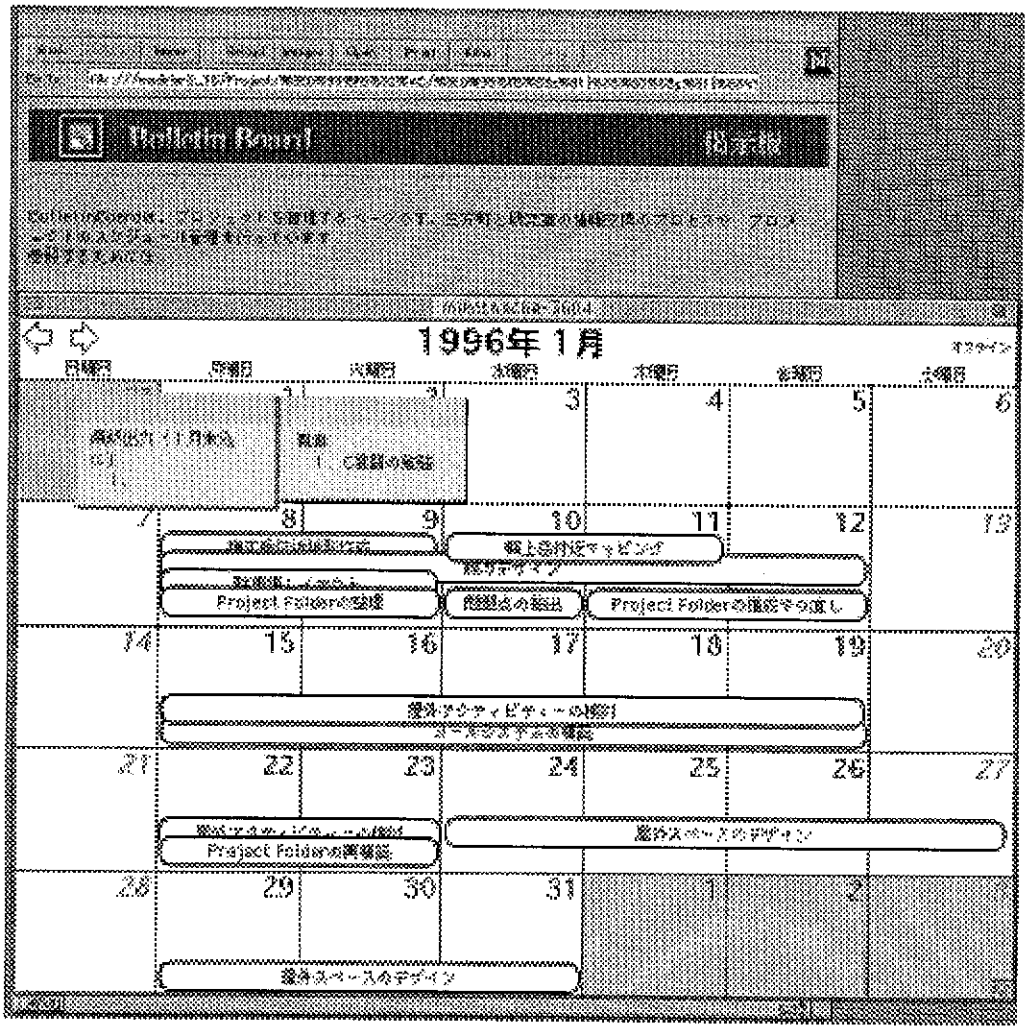

Bulletin Board
1996年1月
3
4
5
6
8
9
10
11
12
13
Project Folderの登録
14
15
16
17
18
19
20
21
22
23
24
25
26
27
Project Folderの作成
28
29
30
31
1
2

Fig. 5

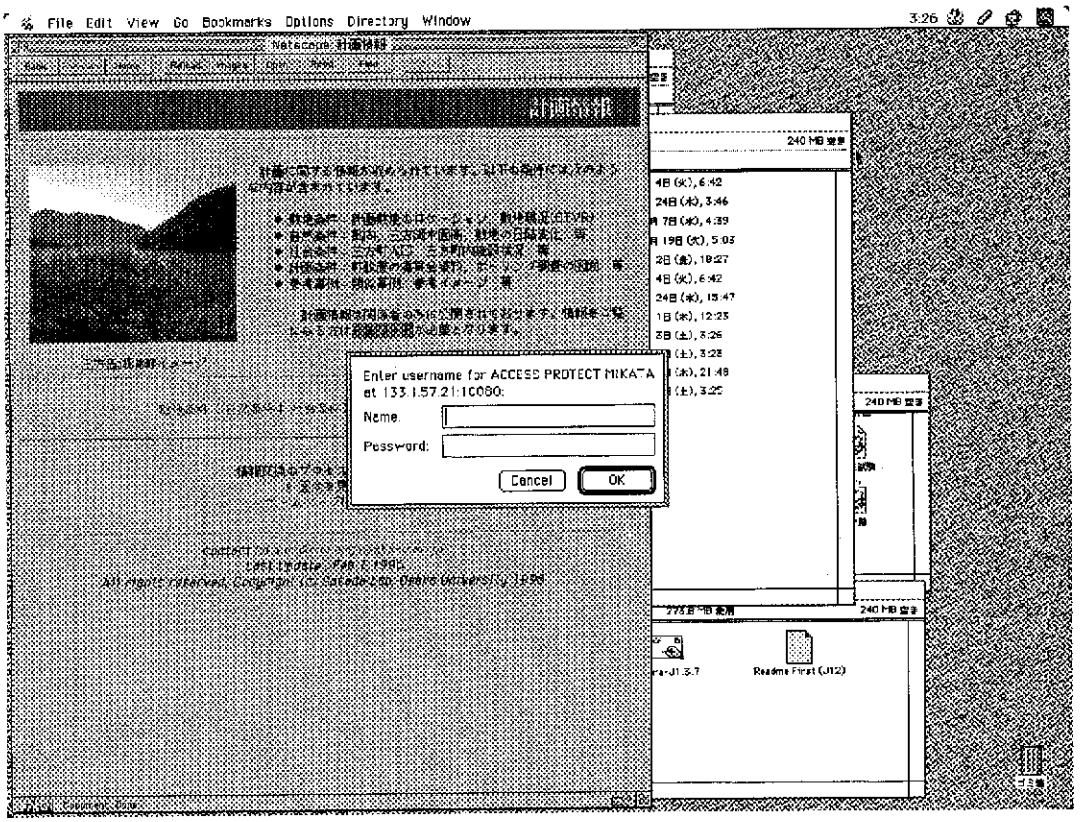

File Edit View Go Bookmarks Options Directory Window
Enter username for ACCESS PROTECT NIKATA at 133.1.57.21:10080:
Name:
Password:
Cancel
OK
Readme First (J12)

Fig. 6

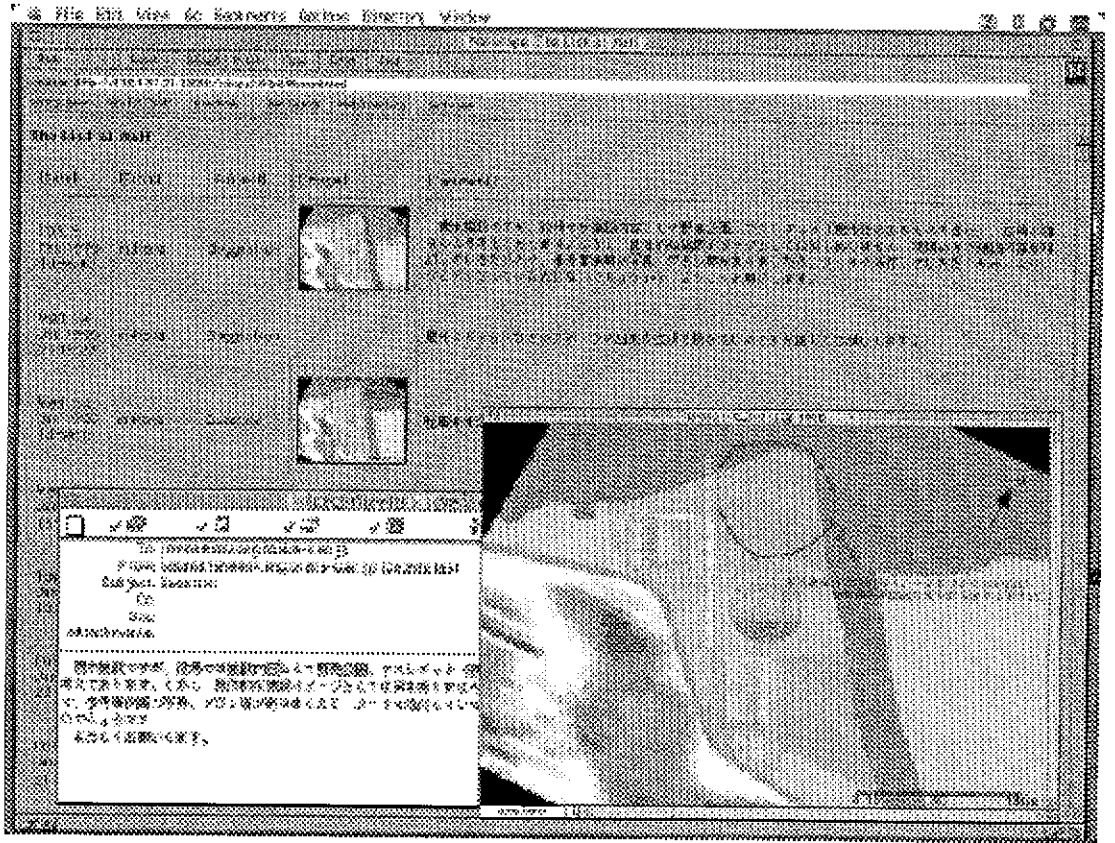


Fig. 7

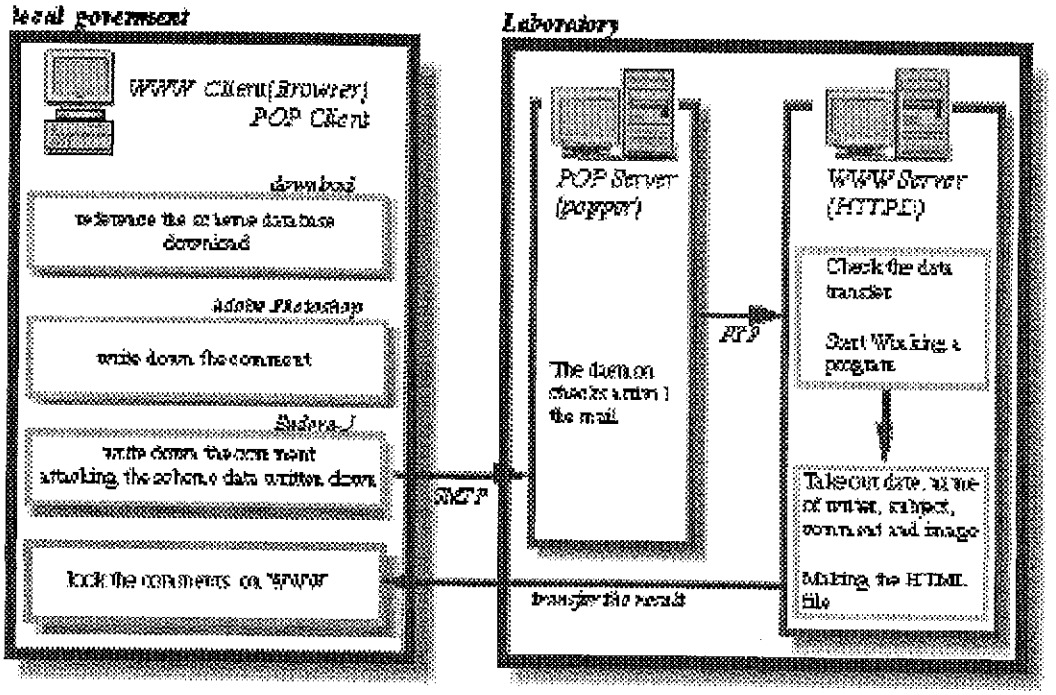

local government
WWW Client(Browser)
POP Client
download
reference the or terne database
download
Adobe Photoshop
write down the comment
Sedora
write down the comment
attaching the scheme data written down
look the comments on WWW
SMTP
Laboratory
POP Server
(popper)
The daemon
checks arrival
the mail
RFP
WWW Server
(HTTPD)
Check the data
transfer
Start Working a
program
Take out date, name
of writer, subject,
comment and image
Making the HTML
file
transfer the result

Fig. 8

BIBLIOGRAPHY

Sasada, T. 1995. "Computer Graphics as Communication Medium in the Design Process" In Proceedings of International Conference on Computer-Aided Architectural Design 1995.

Kaga, A.; S. Woo; C. Comair; T. Sasada. 1995. "Design Projects and Computer Supported Co-operative Works (1)" In Proceedings of the Eighteenth Symposium on Computer Technology of Information System and Applications 1995.

Woo, Sungho; A. Kaga; C. Comair; T. Sasada. 1995. "A Study on Computer Aided Co-operative System for Design Projects" In Proceedings of the Eighteenth Symposium on Computer Technology of Information System and Applications 1995.

Comair, C.; A. Kaga. 1995. "Open Design Environment (ODE): Global Design Studio, Experiments in 3D City Simulation" In Proceedings of International Conference on Computer-Aided Architectural Design 1995.

Comair, C.; A. Kaga. 1995. "Vu: A Database Computer Language for the Simulation of Events in A City (Part I)" In Proceedings of European Simulation Multi-conference 1995 .

Comair, C.; A. Kaga. 1995. "Vu: A Database Computer Language for the Simulation of Events in A City (Part II) " In Proceedings of Pan Pacific Conference on Information Systems 1995 .

Sasada, T. 1994. "Open Design Environment and Collaborative Design" In Proceedings of The 12th European Conference on Education in Computer Aided Architectural Design, 3-6.

ABOUT THE AUTHORS:

Claude Comair: DigiPen Corporation, Doctorate Course, Department of Environmental Engineering, Osaka University.

Atsuko Kaga: Hankyu Corporation, Doctorate Course, Department of Environmental Engineering, Osaka University.

Tsuyoshi Sasada: Professor, Dr. Eng., Department of Environmental Engineering, Osaka University.