Secure Cooperative Visualization and Verification over Internet for Decision Support in Product Design

Ridruejo, F.J.\textsuperscript{1}, Borro, D.\textsuperscript{2}, Griego, I.\textsuperscript{1}, Aguinaga, I.\textsuperscript{1}, and Matey, L.\textsuperscript{2}

\textsuperscript{(1)}: CEIT (University of Navarra) 
Manuel de Lardizábal 15, 20018 San Sebastián, Spain 
Phone: ++34943212800 
Fax: ++34943213076 
E-mail: {fjridruejo, igriego}@ceit.es

\textsuperscript{(1)}: CEIT and Tecnun (University of Navarra) 
Manuel de Lardizábal 13, 20018 San Sebastián, Spain 
Phone: ++34943212800 
Fax: ++34943213076 
E-mail: {dborro, iaguinaga, lmatey}@tecnun.es

Abstract: The use of cooperative tools in order to communicate different people involved in design processes is growing nowadays. We present a generic software tool to support the cooperative visualization and verification of virtual Mock-Ups of mechanical systems.

The application is able to visualize very complex and large models which other CAD applications have problems even on loading. It provides analysis and visualization techniques that favour the discussion of the users of the models.

One of the most valuable resources of many engineering companies are their CAD models. Thus communication security of the data transmitted is of vital importance in any industrial use. Real-Time communications are required even for low bandwidth connections over the Internet. The communication architecture of the application also enables the transparent connection, disconnection of new users to a collaborative session, and role interchange between different users.

Keywords: Virtual Reality, collaborative design, cooperative visualization.

1- Introduction

Nowadays the design process of complex mechanical systems is not done by a single engineering team located in a single place, but by various engineering teams that can be located in different places across the world.

These engineering teams must be able to discuss over different aspects of the design. Traditionally this would lead to time and resource consuming travels and meetings.

The existence of low cost and relatively high-bandwidth communication tool, such as the Internet or teleconference, allows the development of new communication tools. These tools permit an engineering team located in different offices to share information, with a low cost and effectively.

We present a software tool that allows the collaborative visualization of a virtual Mock-Up. This tool is specially design for a company group whose works usually falls under the following scenario. The product design involves people placed at three different sites. We will refer to them as the customer, the caster, and the modeller. The customer orders a certain number of identical parts from the caster and gives him the basic drawings that define the part. The caster passes these drawings to the modeller, who machines the moulds to cast the parts.

All the three parties need the process to be as fast as possible. Obviously, they need the model to be able to generate valid parts as soon as possible. If the cast generates wrong parts, the design cycle must be repeated. This involves a software related process, using CAD tools, and then several production processes. These production processes have a lot of manual work associated with them and are expensive and time consuming. Thus, the location of error must be done during the CAD model definition, to avoid this manual processes.

As conclusion the creation of the model requires a close cooperation between engineers from the caster’s factory and designers from the modeller’s one. Final customers are seldom involved in these discussions, but they could be. Up to now that collaboration requires daily, or even more frequent meetings and trips from the modeller’s to the caster’s because it is not enough to send the design and chat over the telephone: they need to discuss the model viewing the same aspects of the design.

The tool we present allows all the people that meet in the collaborative session to share a common view of the CAD model being analyzed. It also provides several analysis tools that help engineers in the location of problem process.
2 - Previous Work

There are two main types of cooperative visualization tools: software applications and tools based on web services, i.e. it is not necessary to set up any software but only browsing through Internet and sharing a project among different users.

If we look at the first type of tools, there are three possible options in order to accomplish a collaborative meeting. We have considered the following types of cooperative applications:

- CAD cooperative and design integration tools: the partners use the same CAD application that includes support for cooperative work. The file description is kept and thus no information is lost but this solution forces all users to use exactly the same CAD software, and therefore licenses for all the partners must be acquired. One example is the plug-in for SolidWorks called ConceptWorks.

- CAD cooperative tools: in this case no partner needs to have the same CAD application, since an external cooperative visualization tool is used. These tools have the capability of importing files using the formats defined by CAD developers. These tools usually allow the file description to be kept but error can happen during CAD data processing. Since these tools require its developers to buy licenses for the CAD file formats, they are expensive like OneSpace.Net (CoCreate) or Division2000i (PTC).

- General-purpose cooperation tools: this case is similar to the latter but the software does not read native CAD files but files standard 3D file formats such as VRML, STL, etc. This solution is cheaper and since most CAD systems are able to export data to standard formats this is not a problem. The main disadvantage of this type of software is that the original description is not kept. The transformation from the native data used in the CAD software, usually based on NURBS, parametrical surfaces, etc., to the data representation in standard file formats (triangle meshes) reduces the accuracy of the model. Some examples with different prices are Alibre Design (Alibre), AEC/VIZ (Tornado Technologies) and Magics Communicator (Materialise).

We have also analyzed applications from other areas that allow users to visualize a common environment. All this data has been compiled in a report [1].

A possible solution to visualization sharing is the use of tools like Microsoft Netmeeting. This type of tools allows users connected to a session to share their desktop. The software captures images of the users desktop, compresses them and sends them to the client users. Tests performed with application prove that, although this tool is very useful for sharing many applications, it cannot be used in cooperative visualization environments, at least with the present allowable bandwidth.

Besides these commercial applications, there are some applications that address different types of CSCW (Computer Supported Cooperative Work) in the CAD area like that reported by Qiu and Tay [2].

When we look into other areas, there are plenty of applications that allow users to participate and interact in a common virtual world through the Internet. Even more, many games allow users to interact and share an environment through the Internet. All the users of an interactive online game share a common Virtual World, that is it is loaded from their computers. Only their movements and actions are sent across the network as messages. These messages are usually sent to a server computer that controls the session status and resends the messages to all the people engaged.

So, as other authors actually made in the past [3, 4], we decided to develop a custom-made cooperative visualization tool suitable for a restricted set of users, following their needs as close as possible.

3 – Software Description

The software, we present, support three types of users connected to a collaborative session:

- A controller user can control the camera position of the model, its appearance, etc.

- An advanced viewer only sees the changes performed by the controller but he or she can request control of the session.

- A viewer can only connect to the collaborative session. This user type cannot request control of the session.

The application has three main communication tools: a chat tool, a file transfer tool and a 3D-window.

The chat tool helps in achieving two main goals. It provides a simple and straightforward mechanism of communication among users and it provides a session simple logging system. The chat tool can be used instead of telephonic communication or as a supplement to voice communication when a collaborative session is held between people of different languages, a usual problem in Europe. The decision of providing a chat tool instead of a more sophisticated audio based system was based on bandwidth concerns.

A file transference tool is required since not all the users connected to a session are guaranteed to have all the required models on their computers. This tool allows a user that lacks a model to request it to the controller node.

The 3D cooperative visualization view is used to concurrently verify the design of the model and to signal a particular feature to other participants: sometimes to explain a doubt, other times to ask about a detail.
The view of the application main window (Figure 1) is shared by all the participants in a collaborative session. For bandwidth limits, the program has an inbuilt option to control the data transfer flow: in continuous mode, state changes are sent as they occur; in send by order mode, changes are stored and sent only when the user of the controller node requests the application to send them.

Viewing the object in motion is an important cue to help our mind in the recognition of 3-D models; for this reason, continuous mode is the default working mode of the application. However, for low bandwidth Internet connections, it is worthwhile to lose this cue to avoid annoying display lag effects.

The controller can select any of the 14 predefined views, change the viewing conditions freely, or select one of the user views. The user can store or remove views from a list which can be reused in several sessions. Switch buttons define which axe of the model is mapped vertically on the screen. Cutting planes that remove part of the model are used to see the inside and get a better understanding of the structure of the model. Verification may be done visually, but users usually need to make some measurements to check angles, thickness, radiuses, distances, etc., while the discussion with other users is carried out in the CSCW session using the chat tool.

3- Modules
As is shown in Figure 2, the application comprises various modules: network, database, visualization, interface and dispatching module.

Network module is in charge of all network management.
Virtual Concept 2005 Secure Cooperative Visualization and Verification over Internet for Decision Support in Product Design

Paper Number 53

To achieve an interactive visualization the hierarchical structure of the VRML 2.0 files is used to implement two culling techniques:

- Frustum culling, discards all the geometry objects that are not contained in the viewing frustum
- Detail culling, discards all objects whose projection in the final image is expected to occupy a small number of pixels, being thus hardly visible.

The visualization module has the typical rendering options such as Point, Wireframe and Solid. Users can also change object visibility and transparency. A useful tool in a visualization tool is the possibility of defining cut planes (see Figure 3). VIVECO allows the definition of a generic plane that cuts the scene.

To achieve an interactive visualization the hierarchical structure of the VRML 2.0 files is used to implement two culling techniques:

- Frustum culling, discards all the geometry objects that are not contained in the viewing frustum
- Detail culling, discards all objects whose projection in the final image is expected to occupy a small number of pixels, being thus hardly visible.

The visualization module has the typical rendering options such as Point, Wireframe and Solid. Users can also change object visibility and transparency. A useful tool in a visualization tool is the possibility of defining cut planes (see Figure 3). VIVECO allows the definition of a generic plane that cuts the scene.

The Visualization module is in charge of model rendering on screen. With the data passed by database module, this module loads a model and displays it, allowing different rendering modes and techniques, like point, wireframe and solid or different semi transparency levels. Finally, there is the interface module, which task is to offer to user a way to interact with the program. With the visualization module, it can select different items, measure distances or define cutting planes. Also it allows doing other tasks, like to send chat messages or manage data base's models.

3.1 – Visualization

The visualization module has been designed to achieve a number of goals:

- The visualization must be of great quality and close to that of a CAD system
- This module must be able of rendering virtual mock-ups of systems containing thousands of different parts and millions of polygons
- The visualization must be interactive

To achieve this goal several strategies have been implemented in this module. As input file format, VRML 2.0 has been selected. This format is fairly standard and most CAD packages are able to export its models to it. Most CAD systems use a parametric surface representation of the geometry while VRML 2.0 uses a triangular mesh representation. This adds a certain error to the representation due to the triangulation, but since VIVECO is a visualization tool this error can be assumed as most visualization techniques and tools, such as graphic hardware are mainly oriented towards triangle meshes.

We have used a Scene-Graph structure to store the geometrical data of the program. Even if maintaining this type structure seems to be overkill for an application where parts will not be moved this division is useful for culling tasks, and thus for reducing the workload of the computer’s graphic subsystem.

This structure represents the model as a tree of nodes of different types:

- A Scene node, represents the root of the model
- Transform nodes that position the geometry
- Geometry nodes

This structure is flexible enough to allow the extension and addition of new features.
the scene allowing users to display specific areas of the virtual mock-up. The generation of an AVI file requires rendering the scene into a Bitmap in memory. The standard way of doing this task in the Microsoft Windows Operating System requires the definition of DIB and setting-up OpenGL to render into it. This will make the system to rely on a software rendering scheme that outputs a low quality image and is slow. To improve this, VIVECO renders first the image into a Hardware Accelerated buffer and then writes the rendered image into the Bitmap.

3.2 – Communications

Our cooperative visualization application is oriented to session. A session formed by several users who share the same vision of the scene, one of them having the control. The one who is in control is allowed to change the camera, select objects and execute the different actions program provides. Client users are only allowed to visualize scene, send text and, if they have enough privileges, request session's control.

The communication is based in the dynamic client-server architecture and a message system. The dynamic client-server is described in the next subsection.

Dynamic client-server architecture is exploited by the message system, whose task is to make possible the communication between all application's instances in a session. Actually, message system is an integral part of the application, because these messages are used also by the program internally. That is, when the user executes a command, a message is sent from a program's module to another module or to a remote machine. An example would be when server loads a model: in server's instance, a message is sent from interface module to data base module in order to load specified model. Also, from the same interface module, the same message is sent to network module to send this message to all other clients. When this message reaches a client, from that client's network module it is forwarded to the database module to perform the model loading.

The message system is the very kernel of the application. The application makes a heavy use of multithreading almost every module being run on a separate thread. The task to manage every message stream is assigned to a thread called Dispatcher. When a message is created in any thread of the system, it is added to Dispatcher's message queue and then Dispatcher unblocks to manage it. Depending of how the message has been created, it will be dispatched to its destiny thread's message queue, to network manager for being sent to clients or both. When Dispatcher empties its message queue it blocks waiting for a new messages. This behaviour applies also to all other instances' thread, that is, all threads are blocked waiting for activation events, allowing very good resource exploitation.

The second part of the message system are messages themselves: these have a powerful creation system based on the Pluggable Factories pattern1, and an automatic serialization system in charge of message conversion to raw byte format, to allow its sending across network. With these capabilities, it is very easy to add new messages to system.

3.2.1 –Dynamic client/server architecture

In multi-user systems, the two most used network architectures are the client/server architecture and the distributed one [5].

The advantage of the client/server architecture is its simplicity and its easiness of management. This is so, because the control of the application is centralized: the server accepts messages from clients and replicates them to the rest of the nodes in the group. The main problem with this approach is that the server node is a bottleneck when the application scales to a high number of member nodes [6]. This problem can be solved with the hybrid client/server architecture [7]. However, these solutions require the development and support of two different applications (the server and the client).

In a distributed architecture every node has the same importance. Therefore, only one application exists. As there is no server role, it is necessary to hold connections among all the hosts [8]. However, in our application, most of these connections are useless during the session because each host will only use the connections with the host that owns the control of the system. Therefore, the application looses the management simplicity.

Our application uses a different access control policy: there is always one controller or server node that spreads the changes in the scene. This node is the only one that sends visualization messages to the other members of the group. It uses TCP connections to provide reliability in the communications. But this node is not fixed on a single computer, as any client with the appropriate privileges can request control of the session at any time. So the application’s architecture is similar to the distributed one, but it achieves the advantages of the client/server architecture has.

We have used a completely distributed architecture (Figure 3), where the server changes dynamically depending on which node the control of the application has. This solution has several advantages. As in distributed architectures, ours does not need two different programs: the client and the server and, like the client/server one it does not maintain unused connections. As in client/server architectures, there are only connections between the controller node and all the other participants. If the control is transferred to another node, the nodes change their connection to the new controller node. For this reason we call this architecture a “dynamic client/server” one. The node that controls the connections with other nodes can change dynamically in the course of a session.

![Figure 3: The dynamic client/server architecture](image)

---

When a new participant joins the collaborative session, the connection is created using any of the addresses already engaged in it. If this node is not the controller node, the application redirects the new client transparently to the current controller. In this way there is no need at all for any user to know what the controller computer is.

To implement the transport level, we have chosen a policy of short messages. The messages are sent only when there is a state change in the scene (position, orientation, color, etc.). Thanks to this, the application keeps a constant and low data flow in all the communication links and usually achieves a nearly "real time" flow.

The access control to the cooperative session is managed by the token mechanism that establishes that only the controller can interact with the model. Changing the control over the system can be accomplished for two reasons: the controller does not want to own the control and gives it to a client host; and the controller wants to become the controller and requests it. This provides a higher flexibility to the system: the capability of transferring the token dynamically during a cooperative session.

### 3.3 – Security

One of the most important features of the application is security. As we have said during this paper, 3D design models are among the most valuable resources of the engineering companies. Those models being verified and reviewed by several designing teams, or by designers and customers, can not go into the wrong hands. Only viewing some of these 3D models, the right person can find out some of the key industrial secrets of the engineering company.

In this way, the application has been designed to have security measures at several levels as we show next.

#### 3.3.1 User modes of execution

There are three pre-defined execution modes for the application. They are visualization client, client and server ordered by increasing execution permissions.

The visualization client user mode is for anyone like a customer that needs to take part in a visualization session but can not be allowed to take the session control, modify session parameters or modify the configuration database. The purpose of this kind of user mode execution is that the client only can visualize the session, like viewing a video, the only actions allowed are the import of encrypted and compressed 3D models into the models database to be able to visualize the actions done by the session controller over the 3D model, and the possibility of writing comments in the chat bar.

The client user mode is a less restrictive user mode than the visualization client showed before. A user in this mode can do the same tasks as a visualization client, plus participate into a session connecting to a server and viewing the actions done on the 3D model. He also can save the 3D model in his hard disc, modify the configuration database and import 3D models into his model database. He also can do comments using the chat bar and even request the control of the session. If the request is granted by current server node, this type of user would become a server user, if he had as well a user configured in his configuration database as being server. If he had only a client user and no server user in the configuration database he can not ask for the control of the session and thus, he can’t do any action over the 3D model.

The server user mode is the less restrictive user mode. A user in this mode can create a cooperative session, accept or reject incoming connections to the sessions, import 3D models to the model database, change the model viewed during the session, can do changes and manipulate the model being viewed, can write comments in the chat bar, change the configuration of the session and add or delete users in the configuration database. Also a server user can export the model out of the model database.

#### 3.3.2 Connecting to a session

When the application is running in client or visualization client user mode, the software requires a server node data to begin a cooperative session. When a connection request is received in the server, it checks if the IP address from where the request is sent belongs to an allowed IP address listed in the allowed client IP list. If there are no restrictions on the incoming connection then a dialog appears on the server machine screen asking for explicit permission for the new client to join the session. These two methods reduce the possibility of security breaches.

#### 3.3.3 Configuration database

The configuration database is never stored in clear in the hard disc. This database is always encrypted and compressed when is stored in the hard disk.

During runtime, this database is loaded into memory and then uncompressed and decrypted. All queries and transactions to this database are made to memory and never to hard disc. When it shuts down, this memory resident database is again encrypted and compressed and then stored into hard disc, thus sensible data never appearing in a clear form in the disk.

It is of vital importance to keep this database secure, because it contains all data concerning users, their passwords and imported 3D models as well as their respective keys to decrypt and uncompress.

Loading this database always into memory does not become itself a problem, because its size will never be of importance due to the nature of the data kept in it: 3D models are not kept in this database, but only passwords and relative paths.
to find the encrypted and compressed files which store them.

3.3.4 3D models

3D models to be visualised have to be imported into the program before establishing the cooperative session. This import process encrypts and compresses a 3D model and stores it in the hard disk making it unreadable. As said in Configuration database these imported models are always stored encrypted and compressed in the hard disk. Only in runtime the model selected for the cooperative session is decrypted and uncompressed. This process is fully done in memory, splicing the model into little pieces, called chunks. Every chunk is then processed and stored in memory to rebuild the 3D model in a clear form. It is only in memory during runtime when the model is fully accessible. But as we shall see, there are other security features that have been done to protect the application from illegal accesses to memory during runtime.

3.3.5 Executable files

As we said before, the most important information of a user is stored in the hard disk encrypted and compressed, like 3D models and the confirmation database. It’s in runtime when both are in clear in memory.

To prevent illegal accesses to memory in order to extract 3D models or the configuration database, executable files and libraries have been protected using a proprietary software that protects executable files from hackers, encrypting the executable file and libraries, applying anti-debugging techniques, anti-memory-profilers, and preventing execution if runtime debuggers are found in memory. All of this protects the application from attacks based on spying the memory to find the encrypted and compressed files which store them.

3.3.6 Distributing the software

The software used to prevent illegal accesses to memory also serves to create customised licenses of use to each customer who wants to use the application. This software makes possible to create evaluation licenses with time of expiration, and to do distribution keys needed to be able to install the software. This covers two aspects, evaluation versions and commercial versions.

Commercials are also planned to carry the application with them, so that the customer being visited could see former engineering designs made by the company. Designers can create a pack for the customer with the program itself, the configuration permissions they want to give to the customer, and the encrypted models they want to show him, and can give the CD to the customer without any worry of their designs being stolen, if the CD is done correctly with appropriate permissions for not exporting the 3D models and no permission to change the configuration database, letting the customer seeing the designs but not letting him to export them out of the application.

3.3.7 Communications

A cooperative session means that several users see actions done by one user over a 3D model in real time and at the same time. This is achieved using a complex architecture of communications as showed before. Connection between clients and server is encrypted using strong encryption. To achieve secure communications between the two extremes we have used point to point SSH tunnelling between clients and the server.

3.3.8 3D model exchange

When the server chooses to open a 3D model that clients of the cooperative session do not have imported into their 3D model database, then that model is sent to all clients, but making sure that the sending and reception processes are secure.

The 3D model is sent encrypted and compressed to minimize its reception time over the network. As we have said in the former point, communication is secure between clients and server, using SSH tunnelling. Moreover, the decrypting password for that 3D model is sent encrypted over the network, and once 3D model y its password is in the client, the model is then imported into the client’s 3D model and configuration database.

4 - PERFORMANCE

From the point of view of usability two ideas must be kept in mind: the computer graphic power within the hardware and the bandwidth available in the network.

The application has been tested in the Internet with a 56Kbps modem at different hours and different internet providers. Tests have been carried out even at our busiest day hours and we have also used free internet providers. The prototype has also been tested in LANs and through firewalls and proxies. We have tested up to 5 simultaneous connections in a LAN and 3 in the Internet.

We have compared the speed in the generation of frames in a server node in the following conditions: standalone (used as a pure viewer), connected to another node, and connected to 5 nodes. The results show that the load due to the image generation is so great that the task of sending data nearly does not reduce the overall performance at all.

5- Conclusions

We present a program for the cooperative visualization of CAD models and its interest for industrial and engineering companies for a generic, light and easy solution. If the tool is compared to existing commercial systems, it has advantages as the low price or that it can be run into a low cost hardware.

We have developed an application that is able to connect several users and provide them visualization and verification capabilities like measurements and cross-sections, in order to reach an agreement during the product design cycle.

---

Encryption and compression techniques have been embedded into the software to protect sensitive data during the communication process over the net.

The networking architecture which we have implemented allows new users to connect and disconnect within the session in a very simple way, because they do not need to know which the server node is.

The software also implements several techniques to save bandwidth like configurable network message dispatching policy that allow users with low-bandwidth connections have a good visualization of the actions.

Communications use a dynamic server scheme that allows users with the appropriate privileges request the session control.

5- Acknowledgments

The author would like to thank all the staff from the Mechanics department of CEIT.

6- References


