CSCV - Computer Supported Collaborative Visualization

Jason Wood, Helen Wright and Ken Brodlie
School of Computer Studies,
University of Leeds,
LEEDS LS2 9JT, UK.

Abstract : Collaboration is important in visualization: major scientific discoveries are typically the work of large research teams, not the solitary scientist. These teams need visualization tools that allow the different members, perhaps at different sites, to investigate collectively the results of a simulation or experiment. Yet current visualization systems are modelled on the solo worker: any collaboration must be done by clustering around a single workstation.

This paper describes work carried out at Leeds under the COVISA project. We show how the Haber and McNabb reference model of filter-map-render pipeline can be extended to a powerful model for collaborative visualization, where scientists may each construct their own pipeline, but share data and parameters. This acts then as an implementation model for a collaborative modular visualization environment. Indeed we have developed an extension to IRIS Explorer in this way, allowing two scientists to work together on a visualization.

1 Introduction

Visualization is a collaborative activity. Modern scientific research is rarely carried out by an individual. It needs the co-operative efforts of a group of scientists, with a range of complementary skills, to unravel the ‘grand challenges’. These scientists - maybe at different sites, different institutions, even different countries - need to collectively analyse data from simulations and experiments. Visualization is a key component of this analysis process; each member of the research team needs to look at the results from their special angle and with their special expertise, and share their interpretation with the rest of the team. Indeed special expertise may need to be brought into the team at the visualization stage - the visualization expert to recommend the best technique, the artist to suggest the most effective style of presentation. Moreover, once the research is understood, the results need to be communicated to other workers, and indeed to students. This can be done in a passive way by presenting the results as a finished image or video, but much more effectively as a live, participative exercise - education is very much a collaborative experience.

Yet current visualization systems see visualization more as a solo activity. Certainly scientists are well served by a range of powerful systems, such as IRIS Explorer [1], AVS [2] and IBM Data Explorer [3]. These are Modular Visualization Environments (MVE) where, by means of a visual programming paradigm, the user selects a set of modules to fit together in a pipeline, transforming raw data to geometry and then to images. While these modules can be distributed across a network of processors, there is a single interaction point - a single user interface. Thus the research team mentioned above needs to cluster around a single workstation; the Renaissance team of scientists, artist and visualization expert need to physically come together; there is no hope of active collaboration in the dissemination of the results. We need to do better.

A helpful starting point is the time-place model often used by the Computer Supported Co-operative Working (CSCW) community, to help position different activities [4] (see Figure 1). Existing systems clearly fit into the same time, same place quadrant of the model. Yet today’s requirements extend well into the same time, different place quadrant - even to the different time, different place scenario.

This paper will look principally at extending current practice in visualization to allow different place, with some thoughts also on different time.

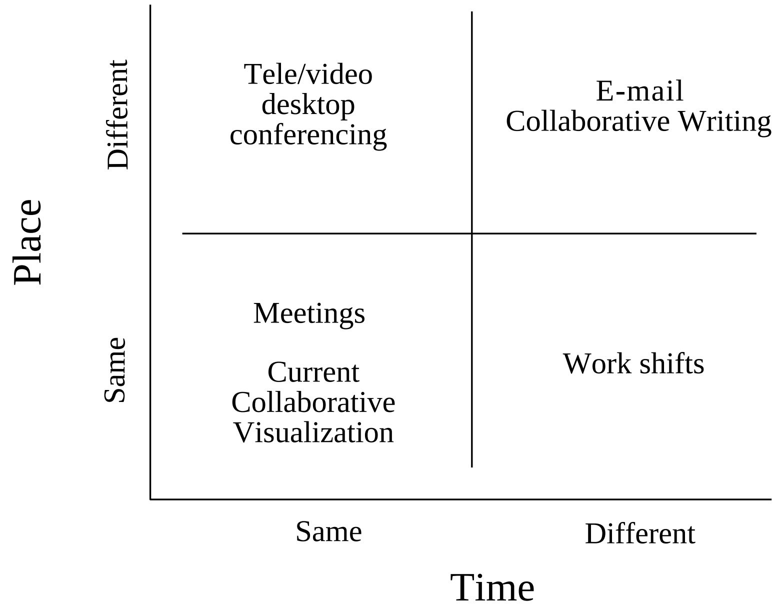

Figure 1 : CSCW time-place model

We begin by looking at other work in this new field, which we might call CSCV - Computer Supported Co-operative Visualization. It is helpful to place the different activities in the context of the elegant reference model of Haber and McNabb [5], which has acted as the foundation of the current set of systems. This leads us to the suggestion of a new model for CSCV. It is an extension of the Haber and McNabb model, powerful enough to encompass the requirements outlined at the start of this paper, and flexible enough to include as special cases the various earlier attempts at CSCV. This is described in section 2 of the paper.

In section 3, we go on to show how our CSCV reference model can act as an implementation model for the extension of current MVEs to collaborative working. A specific illustration is given, where IRIS Explorer is extended from a solo to a collaborative environment. Finally in section 4 we look ahead to future work - tackling the different time issue; extending beyond visualization to wider problem-solving environments; and the potential of the World Wide Web for supporting collaborative visualization.

2 Reference Model for CSCV

Our reference model will draw on the Haber and McNabb model for visualization in a dataflow environment. They describe the visualization in terms of its component processes, which they categorise as filter, map and render stages (Figure 2). Filtering involves taking some data from an input process and refining it, for example, to interpolate from an unstructured to a regular grid. Mapping the filtered data converts it to a geometrical representation, such as a surface plot or isosurface, and finally the rendering stage generates a visible image from this geometrical information. Although it was developed to describe the style of dataflow visualization analysis as used by the family of MVEs such as IRIS Explorer, AVS and IBM Data Explorer, the model can equally well describe turnkey systems. In these cases the individual process elements are hidden from the investigator, by contrast with an MVE where they appear explicitly as modules connected together in a pipeline.

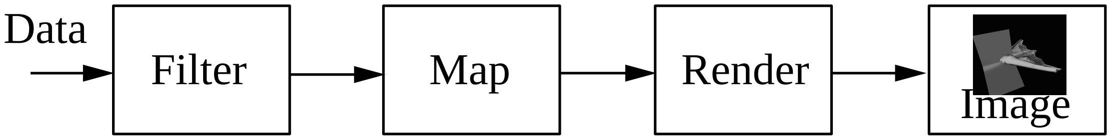

Figure 2 : Haber & McNabb visualization pipeline

The purpose of a reference model for CSCV will be to distinguish and describe different modes of working within a common framework. The following sections review current research in this area in the light of the existing Haber and McNabb model, subsequently proposing an extension to this which leads on naturally to the design of a toolkit for collaborative working within MVEs.

2.1 Sharing a complete application

We begin by examining examples of work which can generally be described as sharing a complete visualization application, whether turnkey or MVE. One way of doing this is via a product such as IBM Lakes [6] or Intel ProShare [7], where an application can be shared without alteration from its original, unshared form. The equivalent in the UNIX environment is to use a shared X mechanism [8]. Using these tools the entire user interface for the application is duplicated on a number of PCs or workstations, and a token passing system is used to determine whether a machine at any one time is acting as the master or as a slave. The master copy of the user interface is transmitted to the slaves each time a change is made so that all see the same output, thus the network load in this approach is substantial.

In terms of the dataflow pipeline model we view this as a single pipeline with many complete sets of control parameters, though the token passing mechanism ensures that only one set is active at a time (Figure 3).

A number of workers have achieved equivalent collaborative interaction by extending their existing software products or writing new ones. For example, one of the extensions which Pagendarm and Walter have made to their HIGHEND MVE system [9] allows two visualizers to work on the same data and produce identical images synchronously. Although the implementation is by means of separate pipelines operating on different machines, the synchronisation ensures that, conceptually at least, the model is of a single pipeline with multiple controls. Furthermore, network traffic is reduced in their implementation because only status, layout and remote cursor information is passed between machines. A similar capability is offered by COVISE (Collaborative Visual Simulation Environment, [10]), a purpose-built, shared visualization system whose interface is based on the visual programming paradigm. Although all partners see identical screen representations at all times, the distribution of modules across different computers in the network is managed by the controller. The complete application can thus be optimised for different combinations of network and compute-power.

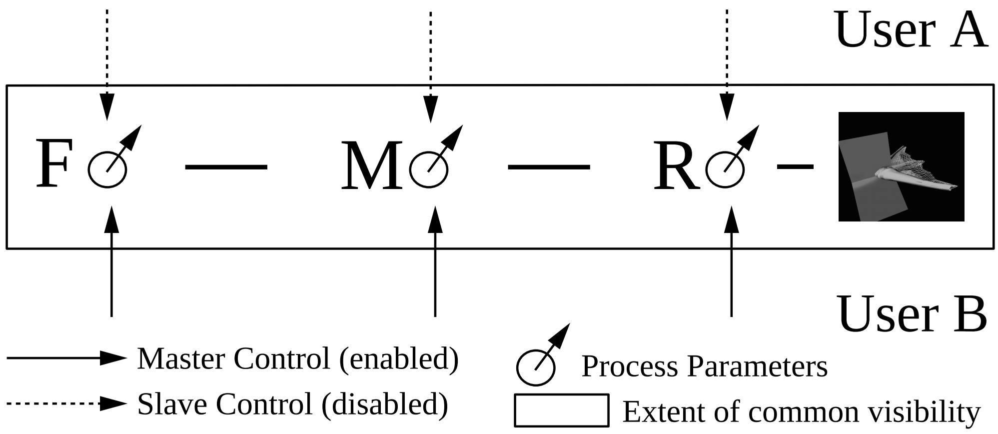

Figure 3 : Sharing a complete application - user B acting as master

2.2 Selective sharing of an application

A more complex model begins to emerge where the application is selectively shared, either in terms of the control being exerted over the processes, or in terms of the data which is flowing. One approach by Grave and co-workers within the PAGEIN project has been to extend two existing MVEs, IRIS Explorer and AVS [11]. Their approach has been to develop an interface panel containing those control parameters from within the network or map editor which are required to be shared. This panel is transmitted to all the partners who make changes in turn using a token passing mechanism. These changes can be seen by all collaborators; however, remaining parameters in the map are private to the partner possessing the base application. Further sharing arises from the use of remote rendering facilities available in IRIS Explorer and AVS, whereby the base application can push the geometrical representation generated by the mapping stage to another workstation for local rendering. Viewing parameters can then be determined locally or synchronised between partners.

In terms of the pipeline model, an example situation is represented by Figure 4. Here, each partner has control of their own rendering process, whilst control of the mapping process is switchable between partners. The remaining processes, however, such as those in the filter stage, are only ever under the control of the owner of the base application, regardless of whether they are acting as master or slave at any particular time. Furthermore, the decision as to which control parameters to share is made at the implementation stage of the shared panel and is not changed during use. Gerald-Yamasaki [12] in his collaborative fluid flow visualizer also offers the possibility to distribute geometry data, and provides for shared and private contexts for rendering control.

We can also see from Figure 4 how a shared application can be implemented either as a single pipeline with multiple controls or as a replicated pipeline with duplicated control. This is evident if we look at the bifurcated render stage and suppose that R has equivalent functionality to RR. Since both are operating on the same data, it follows that if control information is duplicated to each process, then the outcomes must be the same for user A and user B.

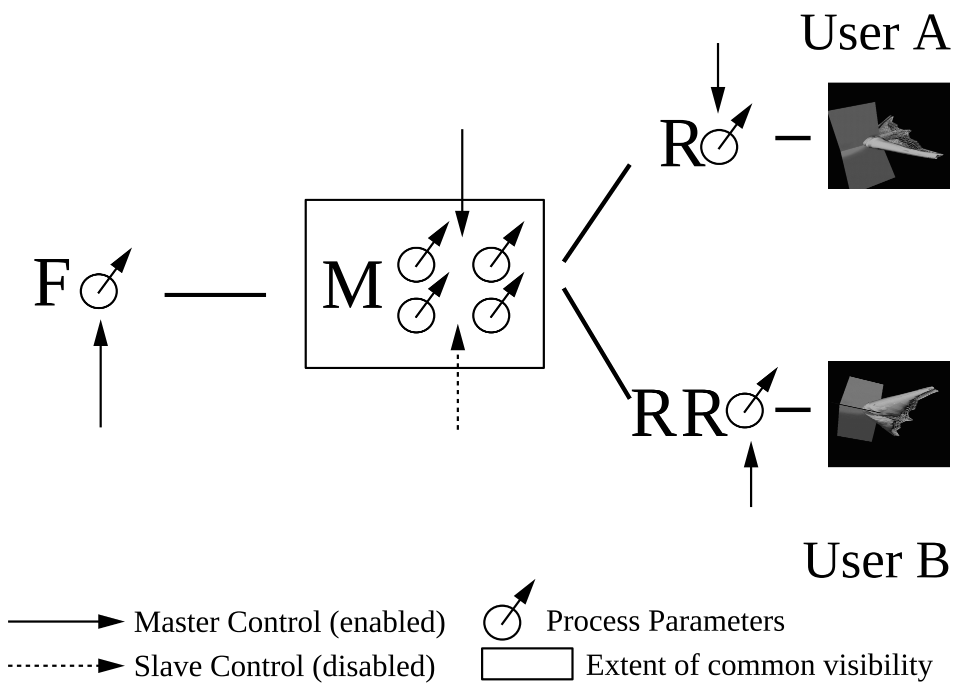

Figure 4 : Selectively sharing an application - user A acting as master.

Yet another variant of the bifurcated pipeline can be seen in the CSpray system developed by Pang et al [13]. This application is based on Spray [14], so-called because the visualization metaphor is to use spray-cans of smart particles (sparts) to highlight interesting features in the data. Sparts leave abstract visualization objects (AVOs) in their path and the use of different spray cans is analogous to making different mappings of the data, which are then converted to images. A can is private to the creator of an AVO until he or she decides to make it public, whereupon other partners can see the AVO and request to use the can. This is the situation shown in Figure 5, where user B has assumed control of the mapping process initiated by user A.

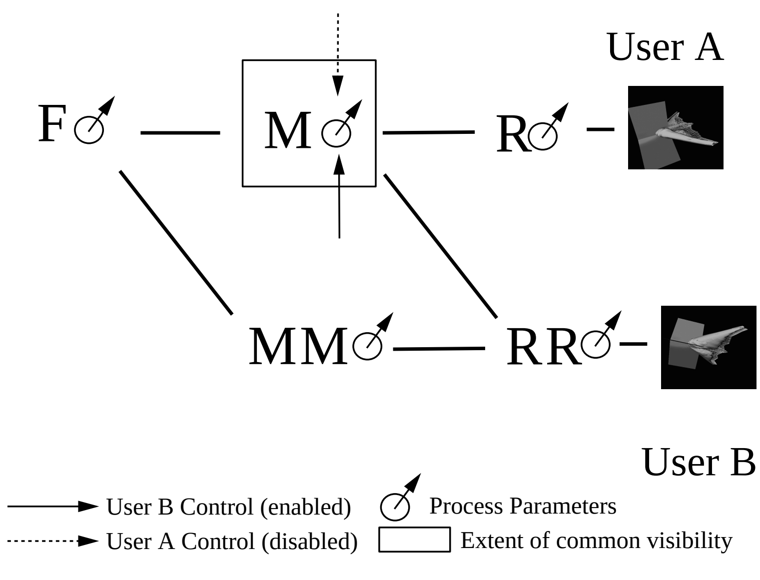

Figure 5 : CSpray - User B controlling spray can initiated by user A

2.3 Extended Haber and McNabb model

These different modes of CSCV can be drawn together into a single picture if we extend the traditional Haber and McNabb model to have intermediate import and export points for control information and data (Figure 6).

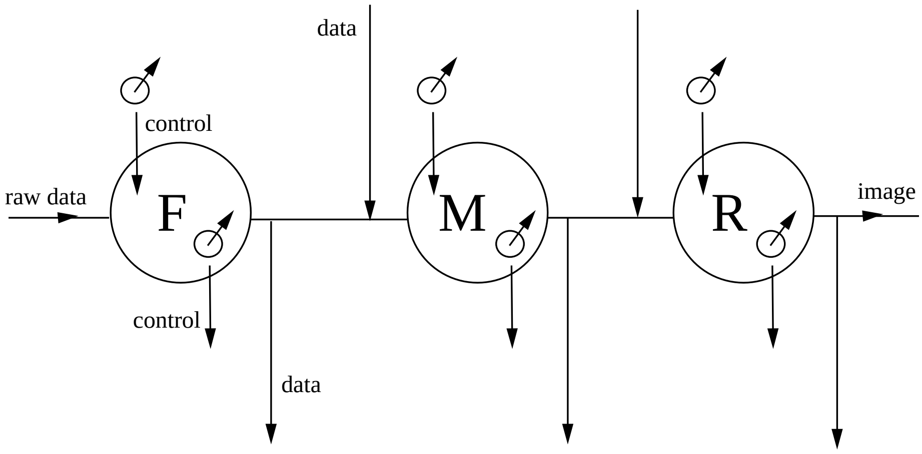

Figure 6 : Extended Haber and McNabb model with intermediate in/outlet of control and data.

By focusing now at the data and control level, rather than at the user level, we arrive at a completely general paradigm for collaborative visualization. Each stage of a complete or partial pipeline can accept data and control information to enable it to operate collaboratively and export data in order to share the outcome. Control information from one pipeline can be exported to another in order to synchronise collaboration, as in the tight coupling mode described by Pagendarm and Walter. Each partner is modelled as having their own pipeline or partial pipeline, where in some cases this might consist of just a render stage with data imported from some other partner's mapping stage. Figure 7 shows user B's pipeline accepting geometry information from user A and viewing it independently. Figure 8 shows the same data being shared but now the two partners also synchronise their views by exchanging control information from their render processes.

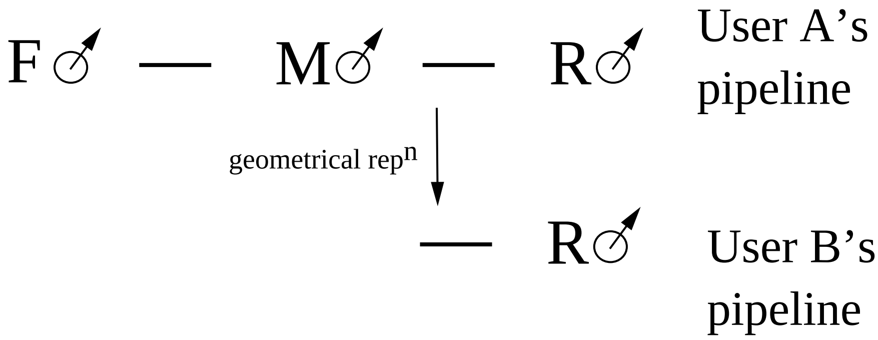

Figure 7 : Users A and B view the same data in different ways.

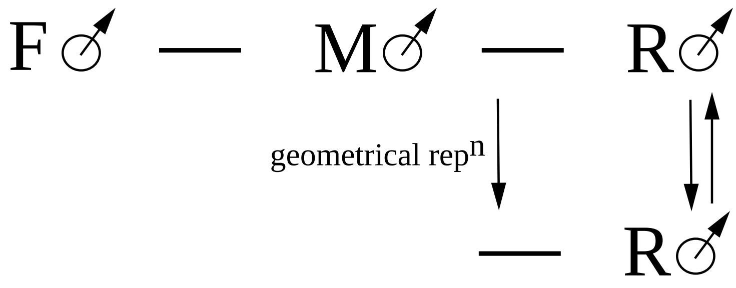

Figure 8 : Users A and B synchronise their views.

This model also allows easy resolution of the issue of public and private data. For example, if two users are collaborating on some mapping process they can proceed in two ways. The first is to export the filtered data from A's pipeline into B's mapping stage and then to exchange control information (Figure 9). However, if the filtered data is to remain private to A, the alternative is to exchange control information at the mapping stage but then to export the geometrical representation (Figure 10). In this case, B's mapping stage consists just of a 'ghost' process which can generate appropriate control parameters for A's map process.

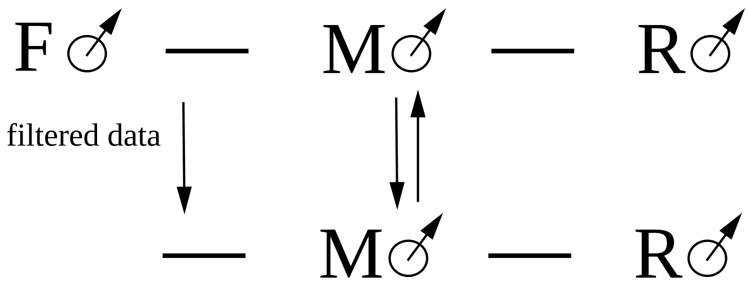

Figure 9 : Users A and B collaborate on mapping stage (public data).

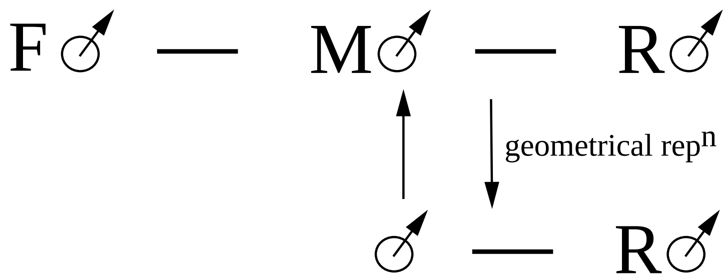

Figure 10 : Users A and B collaborate on mapping stage but the filtered data remains private to A.

A further consequence of this model is that it suggests a paradigm for collaboration which is especially appropriate for extending existing MVEs, elaborated in the next section.

3 COVISA

We have implemented the extended reference model as part of the Co-Operative working in VIsualization and Scientific Analysis (COVISA) project at the University of Leeds. COVISA is a study aimed at understanding the requirements for group working in visualization and to this end we have built several demonstrators to test out different modes of working, basing our work on the visualization system IRIS Explorer.

3.1 Toolkit approach

The reference model presented in section 2 takes the approach of sharing both data and process control at all stages of the pipeline; in an MVE this equates to 'tapping off' the data at the point where it moves from one module to another and sharing it between collaborators. Sharing control of the pipeline can be implemented similarly except that rather than share the data flowing between modules, we share the parameters which control them.

Our aim is to provide 'added value' to existing MVEs, to convert them from solo to collaborative tools. The open architecture of an MVE allows us to create new modules and add them to the system where they are used in the same way as standard modules. These new modules, when wired up in the same manner as the standard modules, can pass either data or parameters out of or into a pipeline. These extra modules then form a collaborative toolkit for the visualizer to build shared pipelines in any of the forms representable by the model. This approach is feasible in all of the MVEs though the implementation details will be different.

3.2 Architecture for shared MVEs

MVEs build data flow applications by means of a visual editor (VE). Each of the stages F, M and R of the visualization pipeline are realised by means of one or more modules within the VE, the flow of data being represented by connections, drawn as lines, between these modules. Thus the architecture for shared MVEs could be represented as in Figure 11 A. This general architecture supports different instances of shared working. For example in Figure 11 B, only the VE on the left has a filter stage, the output of which is shared by both VEs for mapping. The results of this mapping are re-combined in both pipelines, with some control being shared at the rendering level.

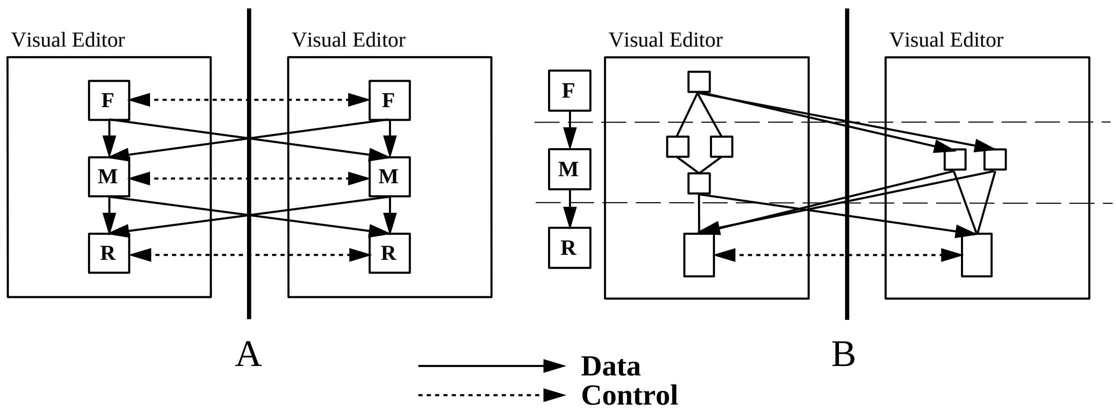

Figure 11 : Extended visualization pipeline

The modules which comprise the collaborative toolkit must be able to form connections not only to modules in their own VE, but also to 'companion' modules in the second VE. Ideally this inter-VE connection must be done seamlesly: to facilitate this, we have developed the concept of 'collaboratively aware' (CA) modules. When launched, these modules cause their companion module to be launched automatically into the second VE and a connection to be formed between them. This connection allows bi-directional flow of data or parameters.

3.3 Implementation in IRIS Explorer

We have demonstrated the implementation of the above architecture in the MVE IRIS Explorer. The collaborative toolkit comprises a suite of modules that are collaboratively aware, one module for each of IRIS Explorer's internal datatypes, and a collaboratively aware server (CAS). When a CA module is launched into the VE, or Map Editor as the VE is known in IRIS Explorer, it connects to the server on its own machine to register its existence and pass information about its type and the socket address on which it can be contacted. This information is passed to the server running on the collaborating machine which effects the launch of the 'companion' module by generating a series of SKm (scheme) commands, SKm being IRIS Explorer's scripting language. Only simple items such as floats and strings can be passed into a module by means of SKm, so the address of the first module is retrieved from the server by making a temporary socket connection to it. Once this address is collected, the second module makes a direct, permanent socket connection to the first. This process is demonstrated in Figure 12, where user B has made the initial launch.

The server can be started at the initiation of a visualization session or, if part way through a session, it may be launched into the Map Editor as a module. The only information that is required is the machine name of the collaborator.

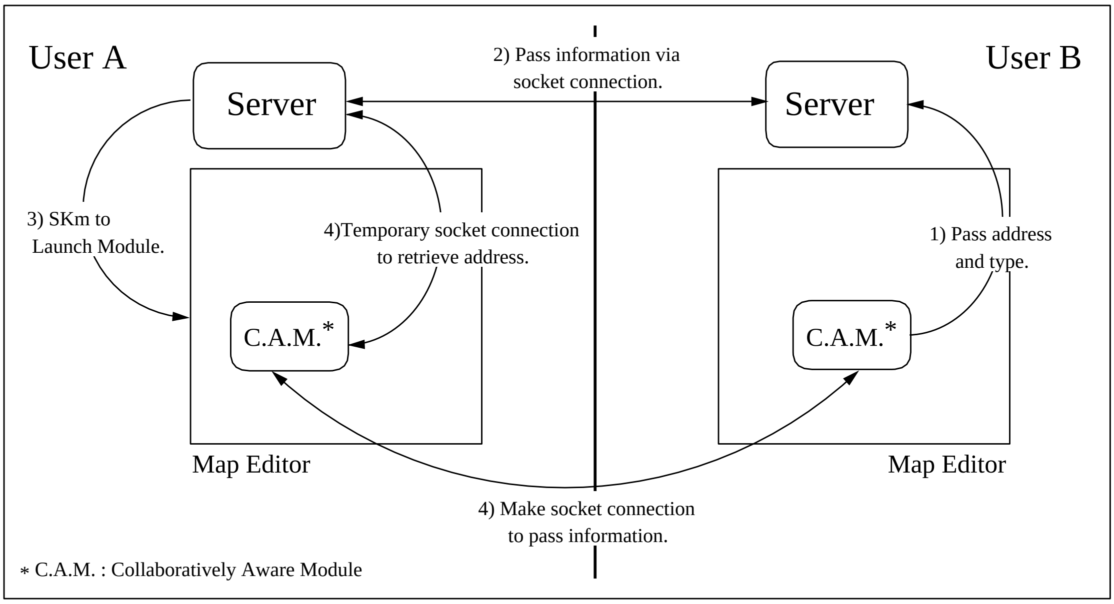

Figure 12 : Launching a collaboratively aware module in IRIS Explorer

4 Conclusions and further work

The tools described work well between two users, but in general we need to consider how to collaborate in larger teams. The extended reference model generalises to more than two collaborators, hence we should be able to extend the architecture and implementation also. We envisage doing this by means of a central server through which all information passes, in addition to the local collaboratively aware servers (Figure 13). Although a central server presents a potential bottleneck when passing large quantities of data, its advantage lies in an ability to scale with increasing numbers of collaborators. A central server also allows partners to join and leave a collaborative session part-way through. Clearly, as the number of collaborators increases, we need to make more formal provision for conference management in place of the present free-for-all mechanism.

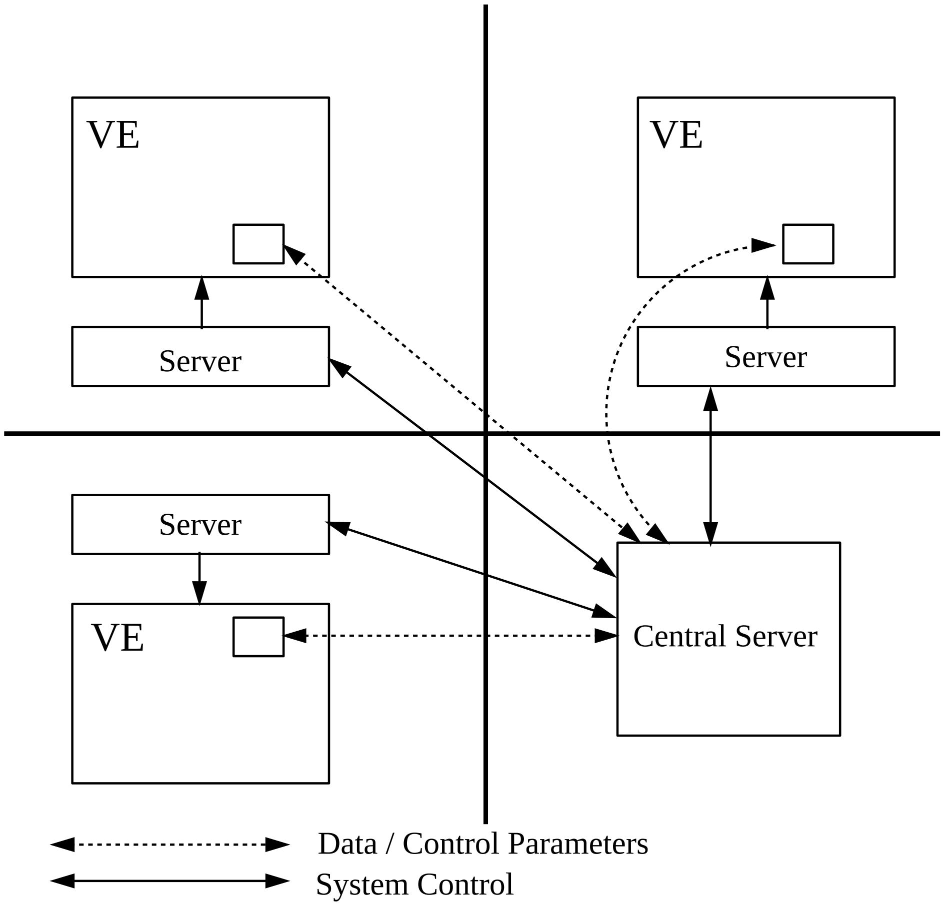

Figure 13 : Collaborative visualization architecture for more than 2 users

The toolkit has been tested during development in a real-life situation involving a fuel engineer and a computation expert. Their experiences demonstrated the soundness of the basic concept but indicated improvements which could be made in the usage of the tools. Most notable was a need for both partners to see the pipeline topology in respect of those processes which were under dual control. This is in contrast to the current situation where the owner of the shared portion of pipeline is the only partner to see the process interconnections in full.

Our proposed solution to this difficulty is to provide a mechanism for shared construction of a pipeline, whilst retaining the ability to do private work at other times. The most fruitful avenue we have identified so far will be to use a journal record to distribute the actions of one partner to another's visualization session. A journal record should contain details of process launches, interprocess connections and process control parameters; indeed, all of the user interactions with the VE which are needed to capture the construction of a portion of pipeline. To share this with another partner, the MVE must also be capable of receiving this information and carrying out the specified actions. Figure 14 shows a development of the architecture in Figure 13 which uses the journal in this way. We are about to commence a feasibility study of the MVEs referred to in section 1 to determine whether the existing facilities they offer make this approach viable.

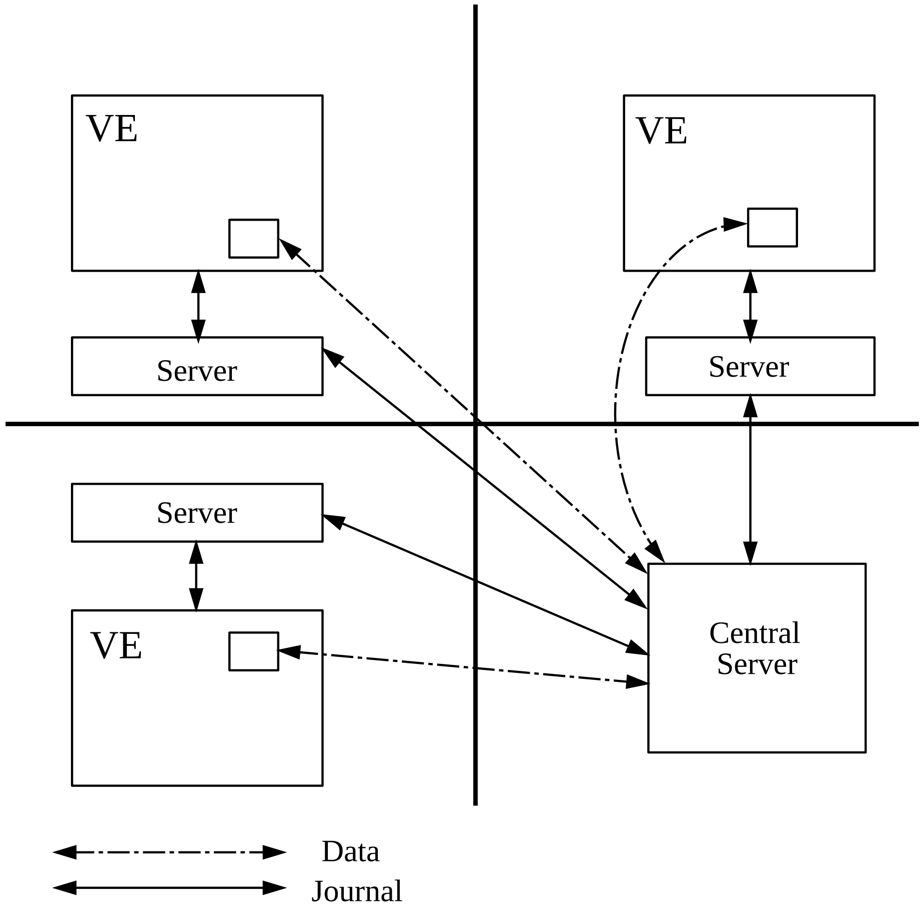

Figure 14 : Collaborative visualization utilising shared journal

We are also working on the transfer into COVISA of ideas from the recent GRASPARC project [15], which aimed to provide an integrated environment to support computation and visualization. A key feature of this support is a management process which stores computed results and simulation parameters at intermediate stages of the calculation, providing an audit of the progress of an experiment. Although originally developed for single person working, we expect the GRASPARC ideas to move across quite naturally into COVISA since both projects are focused at the data and process parameter level. In addition to supporting work in the same time, different place scenario of collaborative working, the audit trail provides a convenient mechanism to support the different time, different place mode too. For example, one worker could develop a simulation to a certain point and store the computational record in the database, which could then be picked up and continued by another worker at some later time. Furthermore, utilising a journal record as described above, the collaborating partner could also review the visualization steps which influenced the progress of the investigation. Combining this audit facility with the potential of the World Wide Web as a shared information repository opens up the possibility of truly global collaborative visualization and scientific analysis.

Acknowledgements

COVISA is a study funded by the Engineering and Physical Sciences Research Council of Great Britain. The project involves thirty staff-months effort expended over two years and is entering its second year.

Our thanks go to all who have contributed through discussion and technical assistance, notably Alison Tomlin and Justin Ware who used and commented upon the tools as they were developed. Special thanks, too, to Jeremy Walton of NAG Ltd for invaluable help with aspects of IRIS Explorer.

References

[1] IRIS Explorer 2.0, Technical Report, Silicon Graphics Computer Systems, Mountain View 1992

[2] Application Visualization System, Technical Overview, Advanced Visual Systems, Oct. 1992, Waltham MA

[3] B. Lucas, G.D. Abram, N.S. Collins, D.A. Epstein, D.L. Gresh, K.P. McAucliffe, "An Architecture for a Scientific Visualization System", proceedings *Visualization 92,* pp 107-114, IEEE Computer Society Press 1992.

[4] L M Applegate, Technology Support for Cooperative Work: A Framework for Studying Introduction and Assimilation in Organizations, J Organizational Computing, Vol 1, 11-39, 1991.

[5] Robert B. Haber and David A. McNabb, "Visualization Idioms: A Conceptual Model for Scientific Visualization Systems", *Visualization in Scientific Computing*, IEEE, pp 74-93, 1990.

[6] IBM Lakes: An Architecture for Collaborative Networking, R. Morgan Publishing, 1994, (ISBN 0-902979-13-2).

[7] ProShare Personal Conferencing, Intel Corporation, *http://www.intel.com/comm-net/proshare/index.html*, 1995

[8] Wladimir Minenko, "The Application Sharing Technology", *The X Advisor,* Volume1(1) June 1995, Discovery Publishing Group, *http://landru.unx.com/DD/advisor/docs/jun95/jun95.minenko.shtml*

[9] H.G. Pagendarm and B. Walter, "A Prototype of a Cooperative Visualization Workplace for the Aerodynnamicist", *Computer Graphics Forum*, Volume 12(3), EG 93 Conference Issue, pp 485-496, 1993.

[10] COVISE : Collaborative Visualization and Simulation Environment. RUS : Regionales Rechenzentrum Universität Stuttgart, *http://www.uni-stuttgart.de/RUSuser/vis/Covise/covise.html*, 1994.

[11] CSCW in AVS and IRIS Explorer. ONERA : Office National d'Études et de Recherches Aérospatiales. *http://visu-www.onera.fr/ONERA/onera.html*, 1995.

[12] M.J. Gerald-Yamasaki, "Cooperative Visualization of Computational Fluid Dynamics", *Computer Graphics Forum*, Volume 12(3), EG 93 Conference Issue, pp 497-508, 1993.

[13] Alex Pang, Craig M. Wittenbrink, Tom Goodman, “CSpray: A Collaborative Scientific Visualization Application, proceedings *Multimedia Computing and Networking*, San Jose. California, Feb 5 -10 1995

[14] Alex Pang, Kyle Smith, “Spray rendering: Visualization using smart particles.”, proceedings *Visualization ’93*, pp 283 - 290, IEEE Computer Society, 1993.

[15] Ken Brodlie, Lesley Brankin, Greg Banecki, Alan Gray, Andrew Poon, Helen Wright. GRASPARC: A Problem Solving Environment Integrating Computation and Visualization. In G.M. Nielson and D Bergeron, editors, *Proceedings of IEEE Visualization 1993 Conference*, pp 102 - 109, Los Alamitos, CA, 1993. IEEE Computer Society Press.