Multicast Application Sharing Tool – Facilitating the eMinerals Virtual Organisation

Gareth J. Lewis[1], S. Mehmood Hasan[1], Vassil N. Alexandrov[1], Martin T. Dove[2], and Mark Calleja[2]

[1] Advanced Computing and Emerging Technologies Centre, School of Systems Engineering, University of Reading, Whiteknights, P.O. Box 225, Reading, RG6 6AY, United Kingdom
{g.j.lewis, s.m.hasan, v.n.alexandrov}@rdg.ac.uk
http://acet.rdg.ac.uk

[2] Department of Earth Sciences, University of Cambridge, Downing Street, Cambridge, CB2 3EQ, United Kingdom
{martin, mcal00}@esc.cam.ac.uk

Abstract. The eMinerals Virtual Organisation consists of a consortium of individuals affiliated to geographically distributed academic institutions. Collaborative tools are essential in order to facilitate cooperative work within this Virtual Organisation. The Access Grid Toolkit has been widely adopted for this purpose, delivering high quality group-to-group video and audio conferencing. We briefly mention this technology and describe the development of a Multicast Application Sharing Tool designed specifically for this environment.

1 Introduction

The eMinerals project [1] is a NERC funded test-bed project, whose primary aim is to use grid computing methods to facilitate simulations of environmental processes at the molecular level with increased levels of realism. The project involves partners from seven academic institutions throughout the United Kingdom. In this setting, there is a clear need for inter-institute collaboration to collectively achieve the aims and objectives of the project.

Collaborative Computing systems aim to complement human face-to-face communication by providing various tools which enhance users' experience. The emergence of multicast triggered a rush in the development of group communication software. Collaborative tools have also become popular with the widespread availability of broadband. The Access Grid [2] has become well known for high quality group-to-group collaboration across the Internet. It has been widely adopted by the academic community and uses multicast for streaming audio and video. Unfortunately, multicast is still not available to many institutions

V.S. Sunderam et al. (Eds.): ICCS 2005, LNCS 3516, pp. 359–366, 2005.

and home users (due to the reluctance of the Internet Service Providers (ISPs) in adopting this technology).

Person-to-person communication is enriched by an ability to share, modify, or collaboratively create data and information. Our aim at The University of Reading is to provide an Application Sharing tool which allows effortless sharing of legacy applications. The Application Sharing tool is developed by the authors, specifically to be used in a group communication environment. This tool will be integrated with Access Grid 2.x to provide enhanced functionality. We are also interested in developing an inclusive collaborative system, allowing unicast participants to interact with multicast groups in a dynamically changing environment.

2 The eMinerals Virtual Organisation

The eMinerals consortium consists of collaborators from seven different academic institutions UK-wide. The project has three components: the science driving the project, the development of simulation codes, and setting up a grid infrastructure for this work. Each of these components are dealt with by the scientists, application developers and the grid experts respectively.

The eMinerals Virtual Organisation consists of individuals affiliated to different institutions. These individual bring with them their own ideas, expertise (in their respective fields), and resources which are bound by their local administrative rules. The eMinerals mini-grid provides the grid infrastructure necessary to manage these resources in order to support the Virtual Organisation (VO).

Collaborative tools are essential in facilitating cooperative work within a Virtual Organisation. These tools provide synchronous and asynchronous communication between geographically distributed participants. The Access Grid currently allows synchronous collaboration using video, audio and other services. Multicast Application Sharing Tool (MAST) complements the available Access Grid services, allowing sharing of arbitrary legacy applications.

3 The Access Grid

The Access Grid is an advanced collaborative environment, which is used for group-to-group collaboration. To achieve high quality collaboration with the Access Grid, a specialised suite can be constructed. A typical Access Grid suite consists of a machine for the audio, video and for the display. The video streams are displayed through several high quality projectors and specialist sound equipment to enhance the sound quality and to reduce the echo. Multicast is used for the transport of video and audio data to multiple hosts. Access Grid uses the concept of Virtual Venues (VV) to allow groups with similar interests to interact, an example of this is The University of Reading VV. Each of the VV's has associated multicast addresses and ports over which streams the video and the audio. The VV uses different ports and sometimes different multicast addresses

to distinguish between the audio and video streams. This provides flexibility for the user to decide whether to receive audio, video or both. Unicasting and broadcasting provide extremes in addressing - unicast uses a single IP address and port as an endpoint, and broadcasting propagates to all IP addresses on a subnet. Multicast provides an intermediate solution, allowing a set of IP addresses to be identified and ensures that datagrams are only received by interested participants.

There are two major versions of the Access Grid Toolkit. The initial release AG 1.x used the Robust Audio Tool (RAT) [4] for audio and the Video Conferencing tool (VIC) [3] for video. Recently, the 2.x version of the Access Grid Toolkit was released. The new version still uses the stand-alone tools VIC and RAT for the audio and video conferencing, but also includes enhanced features, such as, a new Venue Client and Certificate Management functionality.

4 Application Sharing

Video and audio are essential for an effective collaborative experience, allowing participants to mimic natural human interaction. Another major component involves the sharing of material between participants. This includes activities, such as collaborative viewing of visual data, document editing by various colleagues and collaborative code development between geographically distributed peers. Application sharing is the ability to share and manipulate desktop applications between multiple participants, thus facilitating the aforementioned activities.

There are two approaches to implementing Application Sharing. The first, namely colloaboration-unaware, involves sharing of applications in a transparent manner to the application developers. This method does not require application developers to adapt their applications to be shared by distributed participants. The method generally works by sharing a visual representation of the legacy application, without sharing application specific data structures. The second approach, namely collaboration-aware, relies on the applications having an awareness of the collaborative functionality.

The decision upon which of the methods to use depends on the specific scenario. The collaboration-unaware model allows any legacy application to be shared without modification of the legacy applications source code. The collaboration-aware model requires the legacy application to be adapted to include the functionality necessary for collaboration. An advantage of the collaboration-aware method is that the amount of data that has to be transferred between the participants is significantly lower and so the response time is comparatively high.

In a situation where a single application was to be shared between many participants, (and the functionality of the application was known and could be modified) the collaboration-aware model would probably be preferred. The functionality is known and so the code could be modified to include the necessary collaborative features. This model has been used in another project involving the authors, namely The Collaborative P-GRADE Portal [6], where a work-

flow editor is shared between several participants. By providing collaborative functionality and ensuring synchronisation between participants, one can allow distributed participants to work collaboratively on a work-flow.

When designing a tool to share many different, legacy applications across different platforms, the collaboration-unaware model is usually preferred. It is not possible to adapt every legacy application to be used collaboratively, and so in this situation the Application Sharing tools tend to transfer the visual data between participants. The advantage of this is that any application could theoretically be shared, and the application only needs to be running on one of the participants machines. The main disadvantage, as mentioned earlier, is the relatively slow response times due to the amount of visual data being transferred.

4.1 Related Work

There are several applications sharing tools available, which are currently used within the Access Grid. The main examples are; IGPix [7], Distributed PowerPoint (DPPT) [8], and Virtual Network Computing (VNC) [5].

DPPT is one of the most widely-used applications for sharing slide presentations across the Access Grid. It provides a mechanism by which a presenter can control a PowerPoint slide show on multiple sites from a single machine. DPPT is specifically designed for slide shows and users are unable to share any other application. Each participant must also obtain a copy of the slides prior to a session. IGPix overcomes some of the problems associated with DPPT, specifically that, slides do not have to be circulated amongst the group members. IGPix is being used extensively with AG Toolkit 2.x for sharing PowerPoint slides. However, IGPix does not have a scalable model since it uses a client/server topology, with multiple participants connecting to a central point. Another problem is that only one node is able to present via IGPix at any given time.

Virtual Network Computing (VNC) was the most popular tool to be used in Access Grid 1.x in order to share applications during meeting. VNC shares the entire desktop with the participants in the group. However, it can be easily modified to share a specific application.

VNC has become the leading solution for desktop sharing. It is designed for point to point communication, consisting of a server (Xvnc) and a light-weight viewer (vncviewer). A participant wishing to share a particular application runs the application and allows the rest of the group to make individual TCP connections to his/her machine.

The main disadvantage associated with the aforementioned application sharing solutions (which makes them unsuitable for use within the scalable Access Grid) is their use of Unicast protocols for the transfer of the visual data. Sending multiple packets of the same data over the network is not scalable and can lead to performance degradation.

4.2 Multicast Application Sharing Software (MAST)

Application Sharing tools fall into one of the two categories, those that share collaboration-aware applications and those that share collaboration-unaware ap-

plications. As previously discussed, there are advantages and disadvantages associated with both approaches, which makes them best suited for different scenarios. In the case of MAST, the main objective is to be able to share a range of applications between groups of distributed participants. To achieve this main objective, MAST is designed to share legacy applications without any modification to the shared applications source code.

There are two versions of MAST, one for Microsoft Windows and the other for Linux, which can be used in conjunction with the Access Grid Toolkit to enhance the group-to-group collaborative experience. There are several factors that had to be considered when designing MAST:

- Allows Multicasting and Unicasting: To to be in-line with VIC and RAT, MAST allows the data to sent to the participants using multicast or unicast. If multicast is enabled on the network, then the application can send multicast packets. However if the multicast is not enabled, the application can be sent unicast to a multicast bridge.
- Simple Configuration: MAST has a settings menu which allows a participant to select whether the data is sent multicast or unicast. The user can also add details about themselves to be seen by other participants.
- Using a Single Multicast group: Each Virtual Venue within the Access Grid has a unique multicast address and port which is used for video (VIC) and audio (RAT) streams. A similar idea is used for MAST so that participants in a particular VV can share applications on the multicast address associated with the VV using a unique port. Since the multicast address and port combination is well-known within the VV, participants are able to collaborate with ease. The same multicast address and port combination must be used for all the shared applications within a VV. To enable this, MAST has to be able to deal with multiple streams of graphical data being transferred in a single multicast group. MAST uniquely identifies each of the shared application streams and lists these applications with the participants name.
- Reducing Screen wastage: Due to the lack of screen space when using the desktop Access Grid tools, such as PIG [9]/PIGLET [10], we felt that it was important to reduce the screen area required by MAST. The GUI of MAST resembles that of many Instant Messengers, with a simple list of participants, that can be expanded to show all the applications currently being shared by a participant. The user can enlarge a particular application to its normal size within the sharing window. There is only one sharing window, as it was felt that having multiple sharing windows would cause wastage of valuable screen space.

Image Capture. The main goal of MAST is to get the visual data associated with a single application and send this data to other participants within multicast group. An important issue involves obtaining the graphical data. The visual data for the whole application could be acquired and sent to other participants each time an event occurs or after a set interval. Sending visual data associated with the whole application would be inefficient. If only a small proportion of the screen

is changing, it would make more sense to send only the changes. MAST achieves this by splitting the visual representation of the legacy application into sections. When an event occurs or the timer expires, each section is checked to see if there is a change. If the section has changed, it is sent to the other participants. The receiving participants identify the section that has changed and update their local view of the application.

Changes to the visual representation of an application can occur due to hardware or software events. If the user clicks a button on the application, the view of the application will change in response to the event. Similarly if an application initialises an event, such as, an animation, or a progress bar, the visual representation of the application will change. There are two possible methods for reacting to these events - the first is to check the visual representation after a default interval. This method works well for software events, by updating the screen to show changes not induced by external events, such as mouse or keyboard events. The interval between checking for updates is extremely important. The default interval value must attempt to provide good responsiveness, whilst ensuring relatively low overhead. To reduce wastage of processor time, the interval must be relatively high. This makes using the interval method unacceptable for changes due to external events. In a standard operating system users would expect the visual response to an event within several hundred milliseconds. If the interval is set to one second, then the shared application will appear "sluggish" to the user. The alternative to setting the interval is to check sections after an external event. It is sensible to assume that the visual representation of the shared application will change once it receives an external event. Using this method ensures that the shared application will remain responsive.

MAST combines the two methods, the interval can be set relatively high to avoid undue wastage of processor time, whilst still capturing changes that are not due to external events. Checking the segments after each external event (associated with the shared applications window) means that visual changes due to external events are processed quickly, improving the interactive performance.

Transport Design. The visual data must be transferred to each of the participants within a multicast group. Ideally, the data would be sent to each of the participants using multicast. As with the two main AG tools, VIC and RAT, the data can be sent via multicast, if it is enabled on the local network, or by unicast to a bridge. The transport system has been designed to allow the transport of any type of data and this has been achieved by creating template transport classes. The advantage of this is that other applications can easily be developed to use the same transport system.

To be used successfully within a group-to-group collaboration, MAST uses multicast to send the visual data. IP Multicast is an unreliable protocol meaning that packets of data could be lost or received out of order. If some packets are lost then the remote participants view of the shared application could be incomplete. MAST does not have any knowledge of packet loss and therefore it will assume that the section is updated and will not resend the section. To overcome this problem, and the problem of latecomers, MAST must resend sections periodically

even if they appear to be unchanged since the previous send. Re-sending sections could be done all at once after a set number of intervals, at that moment it could take a relatively long time to obtain visual data for the entire application not to mention the increased load on the network. During this high overhead period, external events from the user could take longer to process and so the responsiveness of the shared application would be effected. MAST attempts to balance this load by re-sending a few sections after each interval, this reduces the overhead associated with refreshing the shared application, and maintains the responsiveness to the user.

Session Management. An important aspect of the transport system is the identification of the separate participants and their shared application streams. MAST does not use a central server, which means that the participant list cannot be managed centrally. Each participant must manage their own participant and application lists - this is achieved by uniquely identifying each participant and application stream. When a new application stream is received, MAST checks if the owner of the shared application is present in its own participant list. If the participant is present, then this application is added to the list. If the participant is not present, then the participant and the application name is added to the list. Each instance of MAST must be responsible for detecting participants that leave the Virtual Venue or applications that are no longer being shared. Detecting leaving participants is relatively simple - while a stream is being received a flag is set to indicate that a participants application is active. After a set interval, the flags are cleared - if any streams have a cleared flag at the next interval the application name or the participant will be removed from the list.

5 Conclusion

In this paper, we introduced the eMinerals Virtual Organisation and described the factors that motivated us to develop the Multicast Application Sharing Tool. We described the collaborative model used along with its advantages and disadvantages. The paper gives a detailed description of some of the design and implementation issues associated with the development of MAST.

Our work in the immediate future will be to complete integration of the two versions of MAST to provide a single tool that can be used between the two platforms, and the integration with the current AG Toolkit.

References

1. Dove, M.T., Calleja, M., Wakelin, J., Trachenko, K., Ferlat, G., Murray-Rust, P., H De Leeuw, N., Du, Z., Price, G.D., Wilson, P.B., Brodholt, J.P., Alfredsson, M., Marmier, A., Ptyer, R., Blanshard, L.J., Allan, R.J., Van Dam, K.K., Todorov, I.T., Smith, W., Alexandrov, V.N., Lewis, G.J., Thandavan, A., Hasan, S.M.: Environment from the molecular level: an escience testbed project. AHM 2003 (Nottingham 2-4/9/2003)

2. The Access Grid Project website. Available on: http://www.accessgrid.org.
3. Videoconferencing Tool (VIC) website. Available on: http://www-mice.cs.ucl.ac.uk/multimedia/software/vic/
4. Robust Audio Tool (RAT) website. Available on: http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/
5. Richardson, T., Stafford-Fraser, Q., Kenneth, Wood, R., Hopper, A.: Virtual Network Computing. IEEE Internet Computing, Volume 2, Number 1 January/February 1998
6. Nemeth, C., Dozsa, G., Lovas, R., Kascuk, P.:The P-GRADE Grid Portal. Computational Science and Its Applications - ICCSA 2004 International Conference Assisi, Italy, LNCS 3044, pp. 10-19
7. The IGPix website. Available on: http://www.insors.com
8. The DPPT website. Available on: http://accessgrid.org/agdp/guide/dppt.html
9. Personal Interface to Access Grid (PIG) website. Available on: http://www.cascv.brown.edu/pig.html
10. Personal Interface To Access Grid, Linux Exclusively Thanks (PIGLET) website. Available on: http://www.ap-accessgrid.org/linux/piglet.html