Adapting a Remote Laboratory Architecture to Support Collaboration and Supervision

David Lowe[1], Chris Berry[1], Steve Murray[1], and Euan Lindsay[2]
[1] University of Technology, Sydney, PO Box 123, Broadway 2007 NSW, Australia
[2] Curtin University of Technology, Perth, Australia

***Abstract*— Interest in, and use of, remote laboratories has been rapidly growing. These laboratories provide remote access, via the internet, to real laboratory equipment. Under appropriate circumstances they can support or even replace traditional (proximal) laboratories, provide improved access at reduced cost, and encourage inter-institutional sharing of expensive resources. Most attention to date has been on the development of the core infrastructure that manages access and interaction, and to a lesser extent consideration of pedagogic issues such as which learning outcomes are best suited to this modality. There has however been a recent recognition of the importance of also considering how collaboration and supervision can also be supported. In this paper we discuss a novel approach to the integration of support for multi-user distributed access to a single laboratory instance. The approach retains the benefits of the lightweight client inherent in the underlying architecture.**

***Index Terms*—Remote, Laboratory, Architecture, Collaboration, Supervision**

I. Introduction

Laboratory work is recognized as a key element of many educational disciplines, particularly engineering and the applied sciences [1]. The ubiquity of network access, and the increasingly complex delivery and study patterns of students, has led to a steady increase in the development of remote laboratories over the last decade [2]. When utilized in supporting appropriate educational objectives these laboratories have a number of significant benefits, including significantly enhanced flexibility and convenience for students, improved reliability and reduced maintenance costs, and the opportunity for sharing of laboratories across multiple institutions [3].

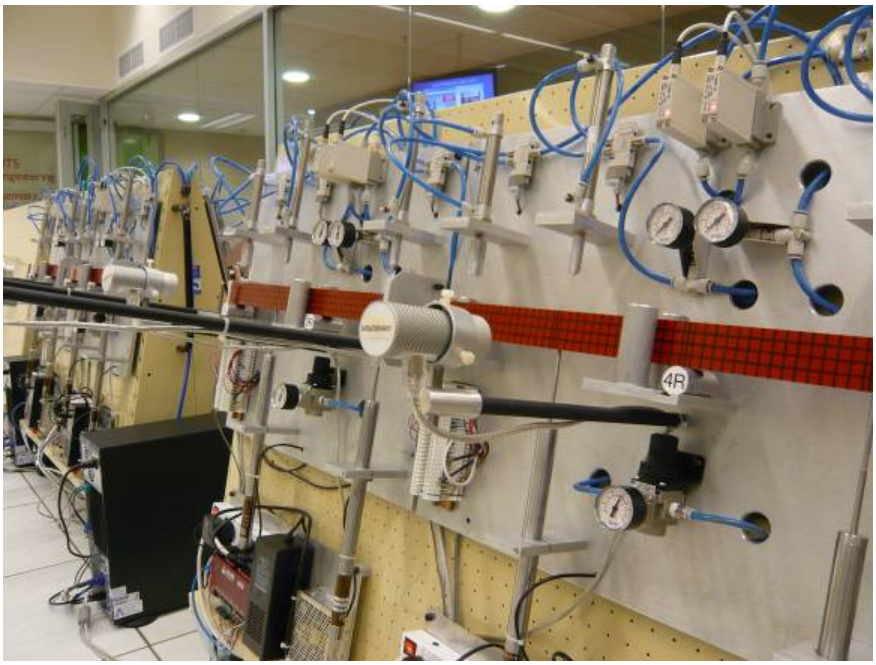
The initial focus of most remote laboratory development was on technical architectures –including experimenting with real-time audio and video streaming, and dealing successfully with the arbitration of multiple simultaneous connections to shared online laboratory apparatus and equipment. To a significant extent, most of these issues have been successfully overcome, with continuous, reliable and high quality services being maintained for much of the past decade. This progress has resulted in a recent shift in the focus of development effort away from technical refinement towards consideration of the pedagogical aspects of remote laboratory access. This has included a more reflective consideration of the laboratory learning context in general (both conventional laboratories where students are proximate to the equipment they're using as well as remote laboratories) and the place of experiment simulation [4]. For example, consideration has been given to the learning objectives of laboratories, and subsequently to which of these objectives might be suitable to the nature of remote interaction. Often the ABET work on laboratory learning outcomes has been used as a starting point for this consideration [1]. Consideration has also begun to be given to the context in which the laboratory learning is occurring – particularly the role of the interaction with other students and laboratory staff [5]. It is this aspect which we consider in this paper – especially in terms of how an existing successful architecture can be adapted to accommodate this interaction.

In the next section we begin by considering issues associated with supporting student-student and student-teacher interactions in remote laboratories. We then describe the development and architecture of the UTS remote laboratories, including the rationale for the architecture, and the subsequent impact this has had on the ability to support shared laboratories. Finally we describe the approach we have taken to modifying the architecture to support collaboration and the consequences of this modification.

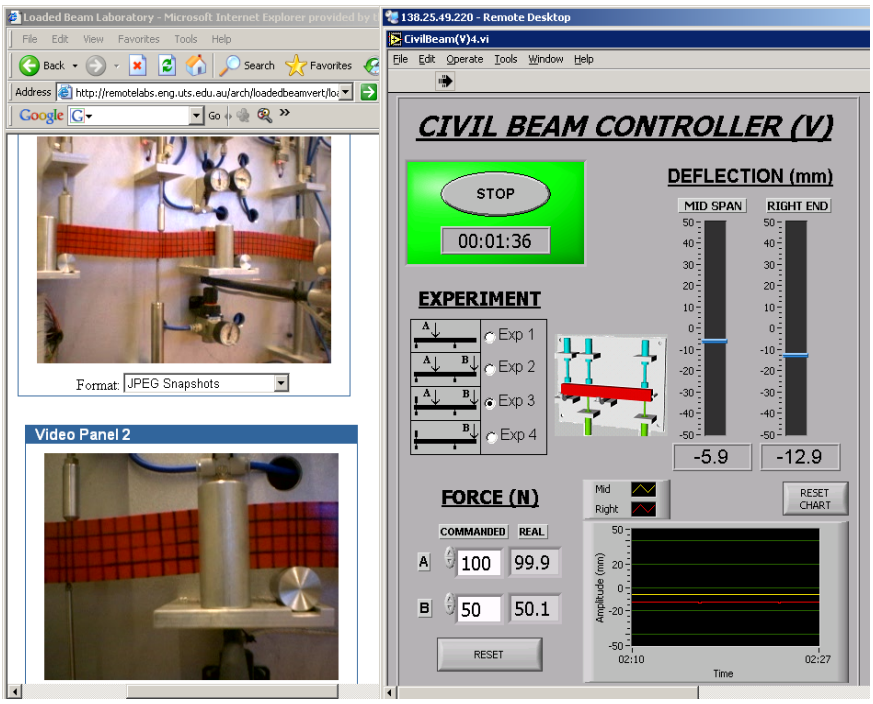
II. Supporting collaboration in Remote Laboratories

It has long been accepted that peer collaboration can play a major role in affecting student learning outcomes. This is particularly pertinent in laboratories, where the majority of conventional (i.e. face-to-face) laboratory exercises are group based – though admittedly this may often have been for logistical rather than pedagogic reasons. Despite the apparent prevalence of collaboration (both student-student and student-teacher) in conventional laboratories, the majority of current remote laboratories provide limited support for collaboration, and largely remain one-to-one connections between student and equipment. One form of support which is often provided (including in the UTS facilities) is a simple discussion board used separately from the experiment, though this does not support effective "live" collaboration in the actual experimentation.

In terms of "live" (i.e. during experiment) collaboration, where this does occur it is typically through remote co-location of the students rather than through technological support. If we are to provide support for student-student collaboration where the students are also remote from each other then several issues emerge. The first is the creation of a shared experience which can form



(a) Physical infrastructure



(b) Typical student interface

Figure 1. The UTS Remote Laboratory Facility

the basis for a common learning context. Issues arise such as:

- How do we provide each student with access to a common view of the experiment?
- Who has control of the experiment and how can this be managed?
- How aware of other students (both within their own group and in other parallel groups) can, and should, each student be?

In most cases the design of the remote laboratories has incorporated an application that supports rich interaction with the physical equipment. Whether this application runs remotely on the students' computers (as is most often the case) or on laboratory servers (as is the case with the UTS remote laboratories discussed later) the application has usually not been designed for shared access.

In terms of student communication (text chat, audio and video connections, and shared workspaces), there has been significant development of technology in these areas, and there are now numerous toolkits which facilitate integration of these functionalities into both web-based and stand-alone applications. However, a key issue which

should be considered in the design of solutions is the role not only of intentional communication (i.e. where two or more students consciously initiate communication – a focus of most existing development) but also the role of incidental and serendipitous communications. Much of the learning context for students in conventional proximal labs involves incidental interactions with students in their own laboratory groups, as well as other groups in the same laboratory. Being able to "eavesdrop" on related conversations, notice the issues confronting other students, and overhear the questions they are asking the instructor, can all play a role in assisting the learning process. It is therefore important to consider how we might support exposing this broader context to students. Partly, this is a design issue – being able to construct interfaces which expose peripheral activities – but it is also a technological issue in terms of how this rich set of information can be structured and presented to users without it being distracting. Certainly virtual reality worlds such as Second Life (http://www.secondlife.com/) can be used to provide a rich context and their feasibility is improving as the understanding of linking real-time data into these environments develops.

Similar issues to those outlined above appear in the relationship between students and instructors. To a large extent the utilization of technology will be the same as for student-student interactions, with the difference largely being in the design of access control. Typically we would want to support both student-initiated interactions ("*Please, I need some assistance with…*") and instructor-initiated interactions ("*You seem to be having trouble – can I suggest that …*"). This latter form of interaction implies a need to provide rich information to the instructor so that they can identify when students might be might be having difficulties. Some of this might be supported by allowing warning flags to be established (e.g. has the amount of time taken to perform a certain experimental stage exceeded some threshold; has some control parameter been set outside some acceptable range), but it might also be effective to provide alerts based on overall level of, or imbalances in, student-student communication, semantic analysis of any text chats, or other forms of rich data mining.

III. THE UTS REMOTE LABORATORIES

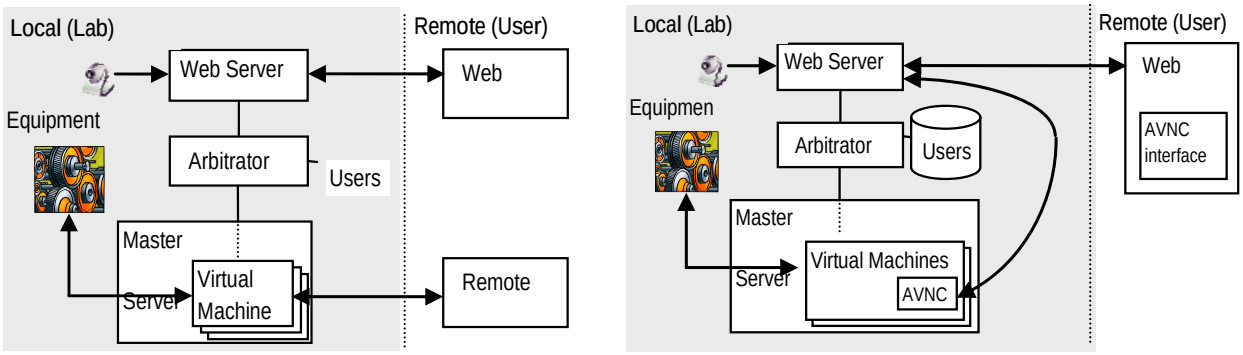
The authors have been working with remote laboratories for almost a decade, and have a very successful facility that supports significant numbers of students from multiple institutions and multiple disciplines. Following a similar pattern to that described above, the early work of the authors focused on technical architectures. Within the remote laboratories at the University of Technology, Sydney (UTS) there are currently five collections of significantly different experiment apparatus and equipment, with a number of others currently under development [6,7]:

- Microcontroller design (Embedded Operating System Experiment) – Computer Systems Engineering.
- Beam Deflection (Loaded Beam Experiment) – Civil and Construction Engineering.
- Dynamics and Control pneumatics (PLC Experiment) – Mechanical and Mechatronic Engineering.
- Fluid Mechanics (Coupled Water-Tanks Experiment) – Mechanical Engineering.
- Programmable Hardware design (FPGA Experiment) –Computer Systems Engineering.

Whilst sharing a common architecture, the specific interfaces and access mechanisms for each experiment vary. One involves the use of Linux hosted software development tools which are character-based and accessed through terminal sessions, yet provides a web-based output user interface. Others require windows based development tools to be available to the user in order to create control programs for industrial PLCs (Programmable Logic Controllers), and still others have been constructed to present a LabVIEW derived application to the user to manage the testing of control algorithms for coupled tank apparatus models. An example of the current facility is shown in Figure 1. In this example, civil engineering students perform an experiment in placing loads on a beam and evaluating the deflection of the beam.

In developing the underlying architecture for the UTS remote laboratory, a number of key design goals played a central role. The most salient of these were that we wanted to manage concurrent access to multiple experiments each with multiple rigs, and that we wanted to ensure the greatest flexibility of access. Concurrent access to many heterogeneous experiments, each with multiple rigs, was managed through the creation of an *Arbitrator* software system, which authenticates requests for equipment and then allocate apparatus to students from the pool of unused devices, queuing the allocation requests when necessary. When a session of use is completed by a student, the Arbitrator reclaims the apparatus, re-initializes it so that it has a state that is healthy for the next usage session and returns the device to the free pool.

The goal of flexible access was, in many respects, more complex to deal with. Many engineering laboratory exercises require specialized software tools to be available to the user (for example, a proprietary tool for constructing PLC programs in ladder-logic, or a LabView application for controlling the coupled tanks experiment). Ordinarily, this would require that a licensed version of the tool be installed by the student on the remote client computer that they are using to carry out a remote laboratory experiment. This creates a requirement at the client (or student) end which may not always be able to be met. For example, a student may not have the relevant permissions on the local computer to be able to install the client software. Even where they do have permissions, the requirement to download the software on each new computer which a student is using adds a layer of complexity that interferes with the natural laboratory interactions. The solution that we adopted offered an elegant side-stepping of these problems. We utilized virtualization software (VMware) to set up multiple virtual PCs running on laboratory servers connected to the physical experiments [8]. All necessary software (both operating system level and user tools) was installed into the virtual machines. When a student logs in and is allocated experimental apparatus they are provided with the IP address of the virtual machine that is connected to their experimental hardware. Students can then connect to this virtual machine and use the software running on it to access and control the experiment. In effect, the control applications run on the virtual machines (and so don't



(a) Original architecture (b) Revised architecture

Figure 2. The UTS Remote Laboratory Architecture

need to be installed on the students' local computers) but the virtual machine interface is displayed on the student's computer. This architecture is shown in Figure 2a.

This architecture allows students to utilize rich monitoring and control applications (see, for example, the application on the right-hand-side of Figure 1b) without the need to install these applications on the computer being used to access to the laboratory. This approach has proven to be highly successful and circumvents one of the major limitations of many other remote laboratories. It does however have one major limitation – the virtual machines are limited to a single connection. This means that whilst we could readily modify the Arbitrator to allow multiple students to connect to the same equipment (and hence see the equipment video feeds, since these operate through the Web interface), only one is able to connect to the virtual machine that hosts the control applications, and hence only student can control and monitor the equipment. This severely limits the ability to support effective student-student or student-tutor interactions. In the following section we describe an approach to addressing this issue.

IV. Modifications to the UTS Remote Laboratories to Support Collaboration

As discussed above, one of our key design criteria was the maximisation of student flexibility. A key component of this was, in turn, not requiring the student to install or download the control applications. This was achieved through running these applications on the server but providing a remote interface to them. Given that it is not readily feasible to support multiple remote connections to a single virtual machine, this raises the question of how we can allow multiple users to interact with a single virtual machine. The approach we proposed was based on the use of an alternative remote desktop display. We considered various alternatives, but the optimal design was to use a Virtual Network Computing (VNC) toolkit. VNC is effectively a remote desktop sharing utility that allows the display from the virtual machine containing the control application to be remotely displayed in multiple locations (i.e. all collaborating participants in a session) – thereby facilitating exactly the functionality we desired. The chosen implementation toolkit allowed the display and associated interaction to be embedded directly into a web interface, and hence the main remote laboratory screen used by students.

As shown in Figure 2b, the resultant architecture had the remote desktop interface embedded directly into the Web interface. This not only meant that we could now support collaboration, but it led to a simplification of the interface over the original design since the student no longer need to start a separate remote desktop application. There were three existing pieces of software identified that allow a remote desktop to be displayed in the browser using a combination of a server that runs on the virtual machine, and JavaScript which displays the view of the desktop to the remote user and allows for interaction. After assessing the software available, it was decided to use a product called AVNC [9], a .NET based remote desktop tool. Since this software is open source, it can be modified to support extra features such as control sharing as needed.

Having common access to the experiment's control application is only part of the design. To enrich the application further we have designed in support for student and tutor interaction. The current implementation has support only for online text-based chat, but the architecture will allow straightforward integration of video and audio interaction, as well as other coordination support tools. A survey of students who regularly use the UTS remote laboratories showed that over 40% of students identified Instant Messaging or chat as the preferred method of communication with their tutor or supervisor whilst using the lab. The approach of having multiple chat "rooms" was taken, with rooms provided for students, tutors, as well as a room for each device. We evaluated a number of open source web based chat applications and selected phpFreeChat due to its feature range and ease of use. Modifications were made to this to achieve consistency between the chat functionality and the updated user interface, with the ability to select and display rooms based on the current user activity. By providing chat facilities between students and tutors, tutors are able to conduct work in the remote laboratories much like they would in a physical laboratory. At the start of the session, students gather in the "lobby" chat room, and discuss the experiment briefly. Tutors can then introduce the experiment to the students and explain any recommended approaches. Students can then request a device and join an experiment individually or in groups. Once students have joined an experiment, tutors are able to cycle through the experiments and provide assistance, much like walking around a physical laboratory. A future extension to this is to allow tutors to assess student performance during experiments using integration with existing online assessment tools (such as online quizzes). This removes the requirement of separate, often paper based, assessment systems.

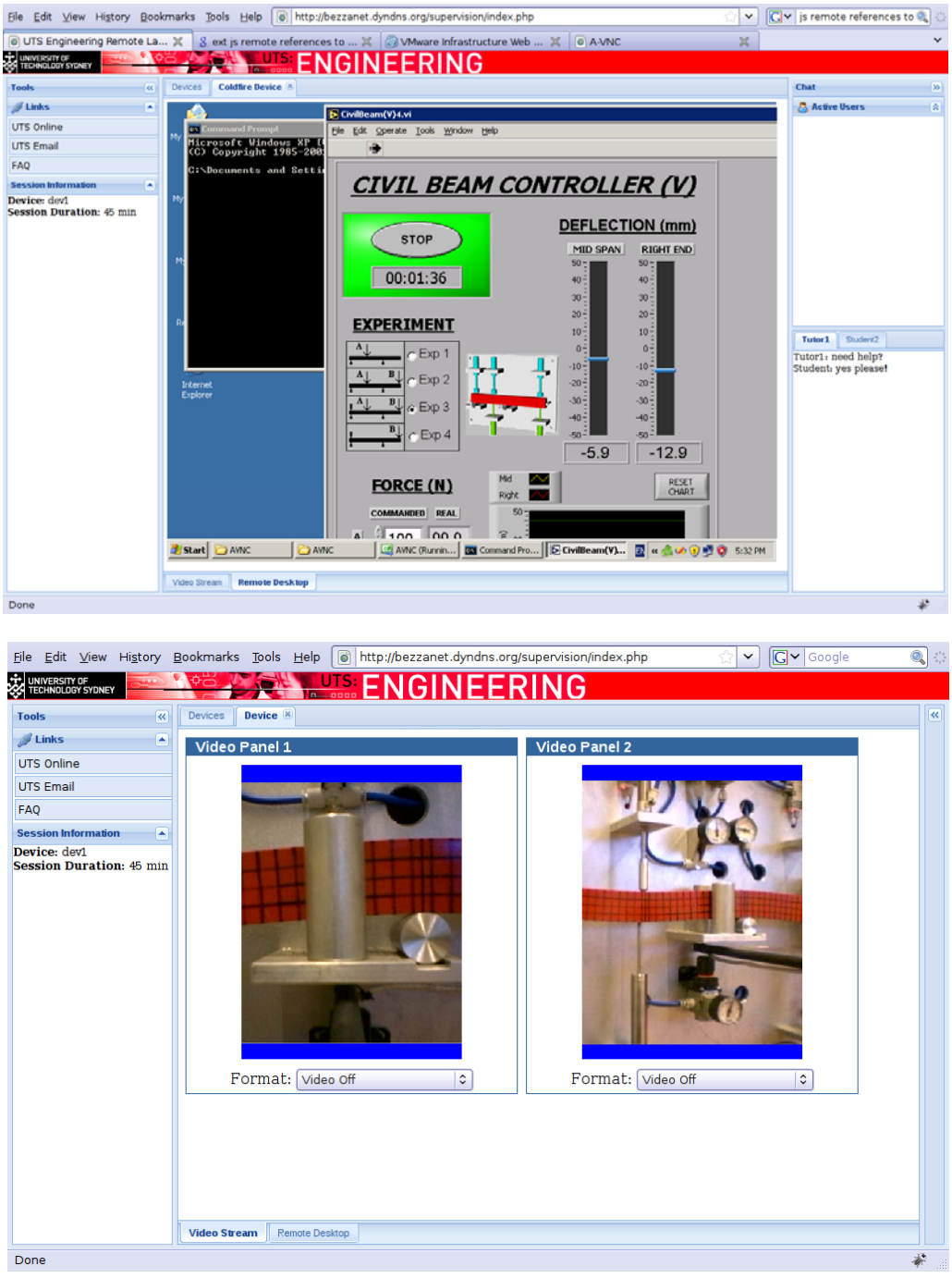


Figure 3. Example Screendumps from the Revised UTS Remote Laboratories.

To properly support supervision, the interface needed to be redesigned to allow viewing of multiple devices concurrently, much like a tutor being able to walk around the laboratories and observe each student or group of students. Students can also request assistance, which displays a message on the tutors overview page. The tutor can then choose to launch the experiment the students are using and provide assistance through the chat or by taking control of the experiment. Tutors are able to join experiments without a call for assistance, giving them the opportunity to make suggestions and observe the students interactions with the experiments, which can lead to a greater understanding of a student's approach to a given experiment, and improvements in teaching methods. As tutors, unlike students or student groups, are not associated with a single experiment, the ability to view a device without having it allocated by the arbitration system was needed. The underlying experiment access scheme was modified to support access by tutors, giving them the ability to view any experiment, including those that are not in use. As shown in Figure 3, the revised Remote Laboratories user interface allows access to extended functionality for both students and tutors. The centrally displayed remote desktop facility enables successful student to tutor collaboration. Tutors are able to interact with the desktop that is attached to the experiment, and view current setup information. They can then choose to directly modify the experiment, and have the student observe the modifications, or they can suggest a solution via the chat facilities, shown on the right hand side of the screen. If a suggestion is made, tutors are able to observe the students reaction to this suggestion and provide further assistance if needed.

The resultant system performed well, particularly in terms of the supported functionality – allowing a laboratory exercise to be carried out collaboratively by multiple students, with synchronous access by a tutor. The major limitation of the approach was discovered to be the responsiveness of the Web-based interface when interacting with a control application hosted on the server-side virtual machine. This was not problematic for

experiments that involved low-levels of interaction, but may become an issue for more rapidly changing experiments. Ongoing work is investigating optimizations that will improve this situation.

V. Conclusions

In this paper we have discussed the implementation of collaboration support in the UTS Remote Laboratories. This collaboration was achieved through a modification to the existing architecture which enabled students and tutors to share a common laboratory application interface, whilst retaining the key benefit in the current architecture of running the control applications on laboratory servers rather than remotely on the users computers. The implemented system has demonstrated the feasibility of this approach. Ongoing work will investigate student reactions to the redesigned system, and the extent to which rich pedagogically-significant interactions are supported. We are also investigating alternatives that provide a more immersive collaboration, such as the integration of real instrumentation into environments such as Second Life.

VI. References

[1] Feisel, L. D., & Rosa, A. J. (2005). The Role of the Laboratory in Undergraduate Engineering Education. *Journal of Engineering Education, 94*(1), 121-130.

[2] Corter, J. E., Nickerson, J. V., Esche, S. K., Chassapis, C., Im, S., & Ma, J. (2007). Constructing Reality: A Study of Remote, Hands-on and Simulated Laboratories. *ACM Transactions on Computer-Human Interaction, 14*(2).

[3] Corter, J. E., Nickerson, J. V., Esche, S. K., & Chassapis, C. (2005). *Remote Versus Hands-On Labs: A Comparative Study.* 34th ASEE/ IEEE Frontiers in Education Conference, Savannah, GA.

[4] Lindsay, E. D., & Good, M. C. (2005). Effects of laboratory access modes upon learning outcomes. *Education, IEEE Transactions on, 48*(4), 619-631.

[5] Ashby, J. E. (2008). *The Effectiveness of Collaborative Technologies In Remote Lab Delivery Systems*. Paper presented at the The 38th Annual Frontiers in Education Conference.

[6] Lindsay, E., Liu, D., Murray, S., & Lowe, D. (2007). *Remote laboratories in Engineering Education: Trends in Students' Perceptions*. AaeE 2007: 18th Annual Conference of the Australasian Association for Engineering Education. from http://www.cs.mu.oz.au/aaee2007/proceedings.shtml

[7] Murray, S., Lowe, D., Lindsay, E., Lasky, V., & Liu, D. (2008). *Experiences with a Hybrid Architecture for Remote Laboratories*. FiE 2008: The 38th Annual Frontiers in Education Conference.

[8] Lasky, V. L., & Murray, S. J. (2007, 14-16 March). *Implementing viable remote laboratories using server virtualisation.* Paper presented at the Web-based Education, Chamonix, France.

[9] Johanson, M. (n.d.). Alkit VNC. Retrieved 27th Oct, 2008, from http://w2.alkit.se/avnc/

VII. Acknowledgements

Support for this research has been provided from the Australian Learning and Teaching Council Ltd (ALTC), an initiative of the Australian Government Department of Education, Employment and Workplace Relations. The views expressed in this report/publication/activity do not necessarily reflect those of the ALTC.

Authors

David Lowe is the Director of the Centre for Real-Time Information Networks at the University of Technology, Sydney, PO Box 123, Broadway, NSW 2007 Australia. (Email: david.lowe@uts.edu.au)

Chris Berry is with the Faculty of Engineering and Information Technology at the University of Technology, Sydney, PO Box 123, Broadway, NSW 2007 Australia. (Email: bezzer@gmail.com)..

Steve Murray is with the Centre for Real-Time Information Networks at the University of Technology, Sydney, PO Box 123, Broadway, NSW 2007 Australia. (Email: stevem@eng.uts.edu.au).

Euan Lindsay is with the Department of Mechanical Engineering, Curtin University of Technology, Perth, Australia (Email: E.Lindsay@curtin.edu.au).