Towards Cost Effective On-Line Laboratories

James Trevelyan
Associate Professor, Department of Mechanical and Materials Engineering, The University of Western Australia,
35 Stirling Highway, Crawley, Western Austrlaia 6009. E-mail: james.trevelyan@uwa.edu.au

Summary

Our aim is to develop the means to provide staff and students with remote access to laboratory equipment using the internet.

The idea that students can operate laboratory equipment remotely to allow flexibility in learning styles is not new. However, there are few instances of this reported, possibly because implemeting such schemes is often expensive and complex.

Our aim is to provide this facility in a cost-effective manner. We have made substantial progress towards our aims. We have demonstrated that LabView provides a useful software environment with a wide range of hardware connectivity, multiple platform support, with the ability to support distributed application software over the internet. Our experience in building web-operated telerobots since 1994 has provided us with useful models for designing the software to enable collaboration between users. The server and backbone of the software has been designed to provide a rugged, maintainable framework. This allows students to build software modules needed to add specific laboratory experiments to the system. By following templates and simple design rules they can produce easily maintainable software. The hardware can easily be maintained by our technicians.

Introduction

Our first project of this kind was a telerobot. (Taylor & Trevelyan 1995, Dalton 2000). However this (eventually) required a huge investment in special purpose software written in C, C++ and Java mainly by graduate students working on the project and the software is essentially non-maintainable. Much of the underlying C and C++ code is tied to the hardware we are using - the PC's, frame-grabbers, cameras and robot interface hardware. This problem (i.e. high life cycle and maintenance costs) seems to have affected other on-line web-controlled equipment projects.

In this project we would like to finish with an affordable and maintainable system. At the same time, we want students to gain practical experience by building parts of the system. This means a careful compromise that allows students design freedom while keeping to design rules that build in reliability and long-term maintainability. If we have a system that can be maintained for the full life cycle and gradually extended within the normal operating budget, then we would consider it cost-effective.

What are the maintenance resources? For hardware we rely on in-house laboratory and workshop technicians. Although they are highly skilled, we still want to minimize the proportion of their time spent on maintenance. We also have students, but with decreasing staff time resources, the level of practical skills they can develop in the short times they are with us is limited.

For this reason we have chosen to minimize special purpose hardware and software investment by building our system around LabView software and National Instruments hardware. LabView software seems to be readily understood by engineering students, at least to the point where they can make useful changes and enhancements. The hardware elements can easily be handled by our technicians.

We used Java in the telerobot project to provide a good base for internet functionality and standardization across different computer hardware and operating systems. Unfortunately, the degree of standardization is not as good as we would like: the telerobot project taught us that complex Java applications and applets particularly (running in browsers) do not provide long term performance reliability: crashes and mysterious side effects are commonly reported by users. LabView seems to provide a more robust base for cross-platform support than Java (supports all major hardware platforms and operating systems). By building the system in LabView it is less likely that software changes would be needed after replacing computer or interface equipment, or after operating system upgrades.

At first it might seem unnecessary to build special purpose software. LabView provides all the basic facilities required to implement online laboratories: internet communications, hardware connectivity, web serving technology and easy construction of user interfaces. Access to an experiment for a single user at a time is easy to implement and there are several worked examples available.

Laboratory experiments in mechanical engineering often require substantial time periods: equipment can take several minutes to reach stable running conditions. Given the need to provide facilities for large classes an effective online laboratory system must provide effective queue management and timetabling. Students are not good at managing their time: they usually leave their work to last possible moment. For this reason, timetabling is important for laboratory work. At the same time, students are not good at turning up at the right moment, particularly after normal working hours. Therefore, we have provided a combination of queueing and timetabling to regulate student access. We expect it will require considerable fine tuning.

Students need to be able to collaborate in many laboratory classes. We have designed our system to allow collaboration between groups of students working at the same time in different locations. When one student moves the controls all the other students see the controls on the own panels move at the same time and the name of the student moving the controls is displayed all the others. This arrangement also allows staff and students to use the system at the same time so that the staff can provide remote assistance. A chat window allows text communication between all simultaneous users. Another chat window allows communication between students waiting in the queue.

A robust system must allow for connection failures particularly as students use older computers and may not be able to afford high-quality internet connections. The system must be able to restore broken connections without interrupting experiments. Needless to say a student will not be able to control the equipment while his or her connection is broken but many experiments can continue and results can be recorded. The system then waits until the student re-connects. Results are sent by E-mail even if the student fails to reconnect.

The system must record actions of students to allow for the development of automatic assessment. We have learned that partial use of automatic assessment relieves staff from many routine assessment tasks to allow them to spend more time with students.

The system ensures that students have downloaded the correct versions of the laboratory control software: if they do not have the correct versions it is downloaded for the while they are waiting in the queue. If the queueing time is long students can disconnect and rejoin the queue closer to the time when they are likely to gain access. The system is capable of sending a text message to mobile telephones when students are about to get access to the experiments.

Experience with Online Laboratories

From the first months of operation, our telerobot (Taylor & Trevelyan 1995) was used from time to time, on request, as a device to help with courses on robotics. One of the earliest examples of a purpose designed laboratory experiment system on the web was described by Henry (1996a, 1996b). Salzmann *et al* (2000) and Gillet *et al* (2000) describe an internet-accessible DC servo device and how it can be used to help with student learning. Shen *et al* (1999) and del Alamo (2001) describe similar arrangements in which students can remotely test semi-conductor devices in web-enabled experiments. A further similar arrangement is being developed in Norway (Fjeldy and Jeppson 2001). Ferguson (1997, 2001) and Lemckert (2001) are working specifically on distance learning applications of this technology. Qanser (2001) offers commercial software and hardware for web-enabled control system laboratory experiments.

What has been learned from these pioneering efforts? First, while several web sites are still in operation, few show signs of recent developments. There was a surge of interest between 1994 and 1999, and the pace of development has slowed since then. The telerobot site is the only one of these to have continuously operated throughout this period. The UTC web lab (Henry (1996a) has operated since 1996 but little has changed recently. Operating and maintenance costs have been significant: special purpose hardware-related software has been a major impediment to the maintenance of several projects. Maintaining software capabilities through significant operating system, computer hardware and user software upgrades has taken significant resources and is cited as an issue of concern by most authors.

The relatively small number of working systems and the slow rate at which this technology is progressing suggest that there are some major problems still to be overcome. The problems appear in the form of high installation, operation and maintenance costs. Some of the underlying causes are:

a) Cost of maintaining the system to keep up with hardware, operating system, internet service provider and browser technology changes.

b) Complexity of technological components and of integrating a working system

c) Lack of administrative support for large classes

d) Need to deal with unreliable internet connections

e) Need for collaborating users to be able to work together

In this work we aim to provide facilities for the efficient management of large classes, queuing and timetabling of laboratory users, collaboration between simultaneous users at different locations and measures to deal with unreliable network connections. The experience of the telerobot project (Dalton 2001) has contributed a reliable system design that can meet these requirements.

Initial Investigations

We started in February 2000 with a survey of alternative technologies. To reduce hardware maintenance costs we focused on solutions that involve standard process control and automation hardware, particularly different types of fieldbus technologies. On the software side, we needed solutions that would allow us to purchase upgrades to support new hardware: we could not always rely on particular interface hardware being available.

By July we chose to base our system on National Instruments hardware and LabView software, and we have continued to be impressed by the technical capabilities of LabView, particularly multi-platform support of internet communications. Our computer science students working on the software initially were expecting to build most of the system in Java but, so far, have not written any Java at all.

By the start of December 2000 when several students joined the project for the summer vacation, we had design studies completed for a series of initial laboratory demonstrations:

- Torsional vibration experiment designed by a student some years before. A motor excites torsional vibrations in two discs connected by springs. The experiment supports advanced undergraduate coursework in vibration isolation and suppression techniques.
- Domestic electric iron instrumented with temperature sensors. This demonstrates that even the simplest feedback control devices show some interesting complications. The experiment supports introductory courses on feedback control and mechatronics.
- Sand weighing machine demonstrating automatic bulk materials handling: a machine that deposits a weighed sample of sand into a bag or hopper. This provides a challenging automation programming exercise for an introductory mechatronics course, and introduces students to standard industrial hardware. The machine has been used for 20 years, but the hardware needed a major upgrade to be ready for unattended remote operation in which the control software would not function correctly for much of the time.

By the end of February 2001 the vibration experiment was ready to use and the hardware for the electric iron and the sand weighing machine was essentially complete. On the software side, we had demonstrated all the elements required for the complete system, including a multi-user client-server software module to serve as a skeleton for future developments.

- Students can log in to a laboratory server: of the equipment is available they gain immediate access to an operation control panel and limited bandwidth video picture. The video is provided by NetMeeting or compatible video-conferencing software. If the equipment is already in use they can they wait in a queue until the equipment is ready. They can "chat" with other students in the queue, or practise using the controls using data from modules that simulate the equipment they will use later.
- If necessary, we can arrange for different shared control schemes. As in the telerobot, control can be passed from one user to another. Alternatively, we can also support joint control: all users with appropriate access rights can move the controls: the system will move the corresponding controls on all other user's control panels and report the name of the user who has just altered the settings.
- The control actions and data resulting from the experiment can be recorded for later retrieval by the student if his own computer loses its connection or crashes.
- Access for students can be based on class enrolment lists, and students can join groups of limited size to collaborate on problems.

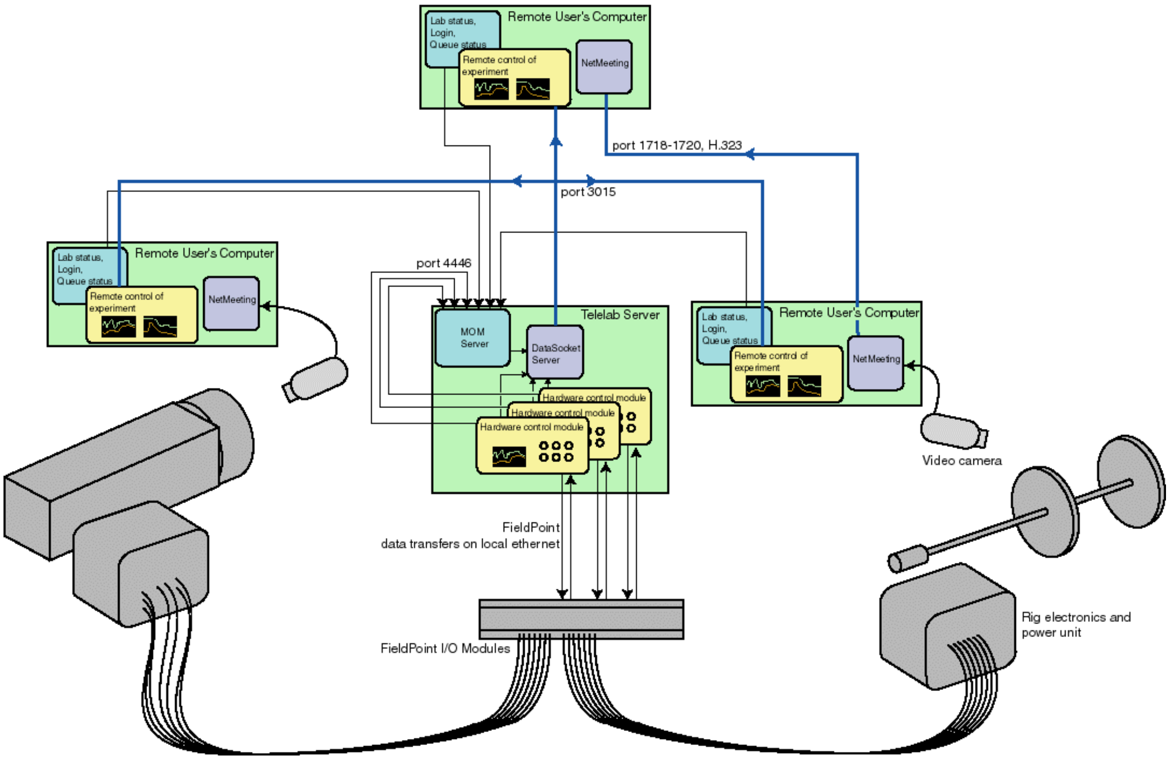


Figure 1. Telelabs software: multiple users, multiple labs. This diagram shows connections needed for a remote user to operate the equipment on the right, seeing the movements of the machine with NetMeeting. Both pieces of equipment can also be operated by students in the laboratory using the (identical) remote client that is installed in the lab PC to provide control settings and monitor the equipment. Note that the MoM server is referred to as LOL in the text.

- Students' actions can be recorded for later analysis to support assessment by supervising staff. The data is kept in secure format and is only accessible by teaching staff.

Two students developed all the software in about 4 weeks following an initial familiarization phase of about 5-6 weeks. Fortunately, there are many worked examples provided with LabView and in related publications that students could adapt for their needs.

Of course, while the elements have been demonstrated they still need to be integrated into a complete working system. The lab server software, for example, only supported one experiment.

Engineering the Server and Software Framework

The server and framework software for the system has been engineered professionally because it has to be very reliable.

There are three components of the framework: the Labs On Line (LOL) Server, the Remote User Client (RC), and the local Hardware Client (HC) that manages experiment hardware. The LOL server handles the communications between RC’s and HC’s as well as providing all the database functionality (authentication, enrolment details, etc). The RC’s and HC’s connect using a standard client module that handles the connection with the LOL server. (Figure 1, figure 2 shows alternative hardware arrangement).

The telerobot project showed that cooperative tele-operation can be conveniently implemented using a "message-oriented-middleware" (Mom) server (Dalton 2001). The server handles connections to users (or peers), authentication of users, and provides access for users to other services (such as lab hardware). Although there is a slight overhead in doing this, there is enormous simplification in design because the HC’s need only communicate with the Mom server. The Mom server in the telelabs project is designated as the Labs On Line or LOL server.

The LOL server contains the following important elements:

1) A finite state monitor that keeps track of user states (figure 3 – last page).

2) A user database handler that supports authentication, and also allows users to update their details on-line.

3) A central data-logging process that retains experimental results and control settings for users. Users can retrieve data for a given time period after using the system.

4) A logging process that records what students do to provide information for the lecturers and provide feedback on student performance.

5) A logging process that records what the Mom server does to help diagnose faults and performance problems.

6) A time-tabling process that allows students to reserve times to operate the equipment

When student logs in and has received authentication, a “hallway” application is launched. From the hallway the user can chat, ask for help and examine rooms to see what the queues for experiments are like. The queue information will contain the approximate time until an experiment becomes free. Once in a queue the student can practice with a simulation of an experiment while waiting.

Assessment and Learning Effectiveness

Our earlier work on internet learning systems has shown that the greatest benefits come from automating at least part of the unit assessment so staff can spend more time with students while they are doing the exercises. This might sound contradictory: why build a system that allows students to work at home if the aim is to allow them to work with staff?

We see the time that the students operate the lab equipment remotely as only part of the learning experience.

Students will be introduced to the exercise either in a lecture or a small group class lasting about 30 minutes with a lab demonstrator. Some of the equipment is being engineered for mobility: it can be wheeled into a class and operated in front of the students. Other equipment will be fixed: students can be introduced in small groups of 10 - 12 students each. We know from our previous experience with the telerobot that most students will benefit from first-hand contact as early as possible in their work.

A typical exercise will involve theoretical modelling and simulation studies coupled with experiments on the lab hardware. Staff interaction is likely to be more beneficial during the modelling and simulation work. We expect this to be interspersed with experiments to gather data over a period of 2 -3 weeks, rather that the more traditional laboratory format where a group of students comes to the lab for a fixed three hour session in which they have to understand the equipment, conduct the measurements and record the data.

The final assessment could be based on reports, interaction with the staff during the modelling phase and even an oral examination in front of the equipment in small groups.

Underlying all of this is the notion of transfer effectiveness (TE) - a term derived from pilot training using aircraft simulators. TE is a measure of the extra time needed in a simulated aircraft to provide the same training effect as a given number of hours in a real aircraft. In our case, we have three learning modes: simulation and modelling, remote operation and hands-on operation. We hope to work towards effective combinations of all three to provide more effective learning than traditional laboratory classes. It will be some time before we can gather data under controlled conditions to assess whether this has been achieved.

We will explore different models of assessment and different learning scenarios to gradually improve learning effectiveness. Given the significant reductions in staff resources for university teaching in recent years (more than 25% between 1995 and 2000) this is a matter of considerable urgency.

Collaboration

A student from the Australian National University helped with the first prototype of client-server software, and is continuing to develop this in Canberra where there is a vibration laboratory similar in principle to the torsion vibration machine at UWA. Several universities have expressed interest in working with us. However, we plan to have a complete multi-laboratory system running before broadening the project to other universities.

Concluding Remarks

This project is experimental and the system described here will only be implemented fully in 2002. While we are confident that it is technically feasible, it will be some time before we can collect data so we can know whether our considerable investment has been rewarded. It has taken us 6 years of experience with our on-line learning systems for analytical courses to begin to assess overall cost-effectiveness. We expect our on-line laboratory courses will take at least this time before we can be confident that our costs are comparable with alternative ways to present laboratory courses.

Acknowledgements

This is a research and development project supported by the Faculty of Engineering and Mathematical Sciences Faculty Initiatives Fund. The author would like to thank his colleagues and his students for their support, particularly Jan Baranski who engineered much of the hardware system, Alex Le Dain of ICON Technologies who engineered the core of the software system in LabView, Sabbia Tilli, Harjono, Flavio Bruni, Peter Dalla Riva, Colin Leung, and Jamie Lee, and Barney Dalton for his work on telerobot software that led to this development. Thanks are also due to reviewers who made valuable comments on the draft.

References

Dalton, B. (2001) Techniques for Web Telerobotics, PhD Thesis, The University of Western Australia (to be published in 2001).

Dalton, B. and Taylor, K. (2000) Distributed robotics over the internet. IEEE Robotics and Automation Magazine, Vol 7:2, pp 22-27. See http://telerobot.mech.uwa.edu.au/.

del Alamo, J.,Brooks, L., McClean, C.,Hardison, J., Mishuris, G.,Chang, V., and Hui, L. (2001) A web-enabled remote laboratory for microelectronic device characterization, Personal Communication (draft paper). Also see http://weblab.mit.edu/.

Ertugrul, N. (2000) Towards virtual laboratories: a survey of LabView-based teaching/learning tools and future trends. Int. Journal of Engineering Education, 16:3.

Ferguson, C. (1997) Remote access to computer-controlled manufacturing facilities for engineering students, World Manufacturing Congress, Auckland, New Zealand. On-going project at: http://www.et.deakin.edu.au/research/Eng_Education/internet.htm.

Fjeldy, T., and Jeppson, K. (2001) Internet technology in laboratory modules for distance learning. See http://www.unik.no/~torfj/remotelab.htm.

Gillet, D., Salzmann, C., and Gorrochategui, E. (2000) Remote manipulation with LabView for educational purposes. in Travis (2000).

Henry, J. (1996a) Controls laboratory teaching via the World Wide Web. ASEE Annual Conference, Session 3513, available from National Instruments Academic Resources CD-ROM 2001 Edition. See also http:// http://chem.engr.utc.edu/.

Henry, J. (1996b) Running laboratory experiments via the World Wide Web. ASEE Annual Conference, Session 3513, available from National Instruments Academic Resources CD-ROM 2001 Edition.

Lemckert, C., Fergusen, C. (2001) Remotely operated laboratory experiments in engineering for distance education students. On-going project at: http://www.et.deakin.edu.au/research/Eng_Education/flow.htm.

Quanser (2001). Company web site at http://www.quanser.com/.

Salzmann, Ch., Gillet, D., and Huguenin, P. (2000) Introduction to real-time control using LabView with an application to distance learning. Int. Journal of Engineering Education, 16:3. See also http://iawww.epfl.ch/Staff/Christophe.Salzmann/MS_HTML/ChMS.html.

Shah, F. and Travis, J. (2000) Using Java applets to control DC servo dynamometer system remotely. In Travis (2000).

Shen, H., Xu, Z., Dalager, B., Kristiansen, V., Strøm, Ø., Shur, M., Fjeldly, T., Lü J.-Q, Ytterdal, T. (1999) Conducting laboratory experiments over the internet, IEEE Trans. on Education, August, 1999. See http://nina.ecse.rpi.edu/shur/remote/.

Taylor, K. and Trevelyan, J. P. (1995) Australia’s telerobot on the web. 26th Symposium on Industrial Robotics, Singapore, October 1995, pp 39-44. See http://telerobot.mech.uwa.edu.au/.

Travis, J. (2000) Internet Applications in LabView, Prentice Hall, New Jersey USA, 600 pp.

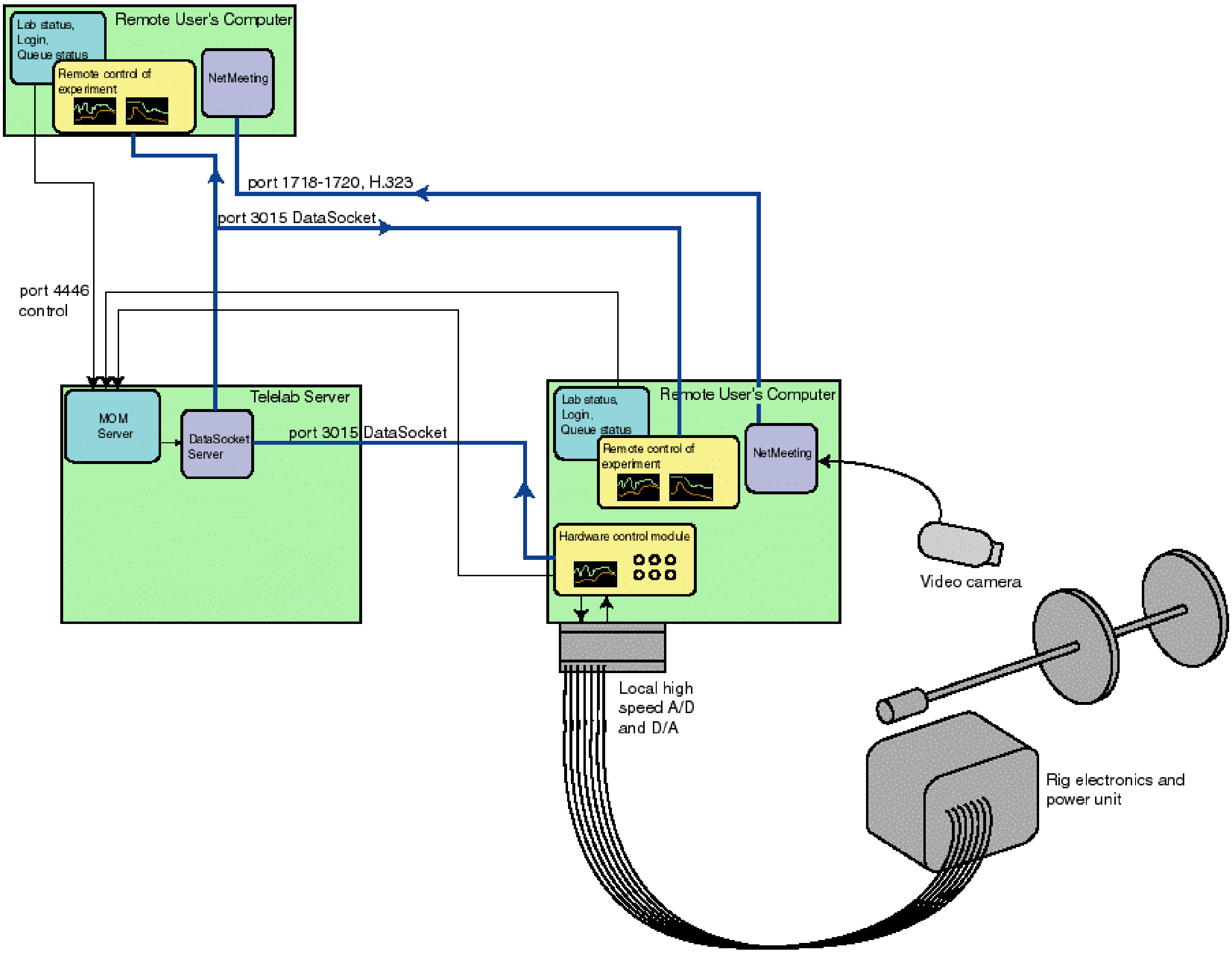
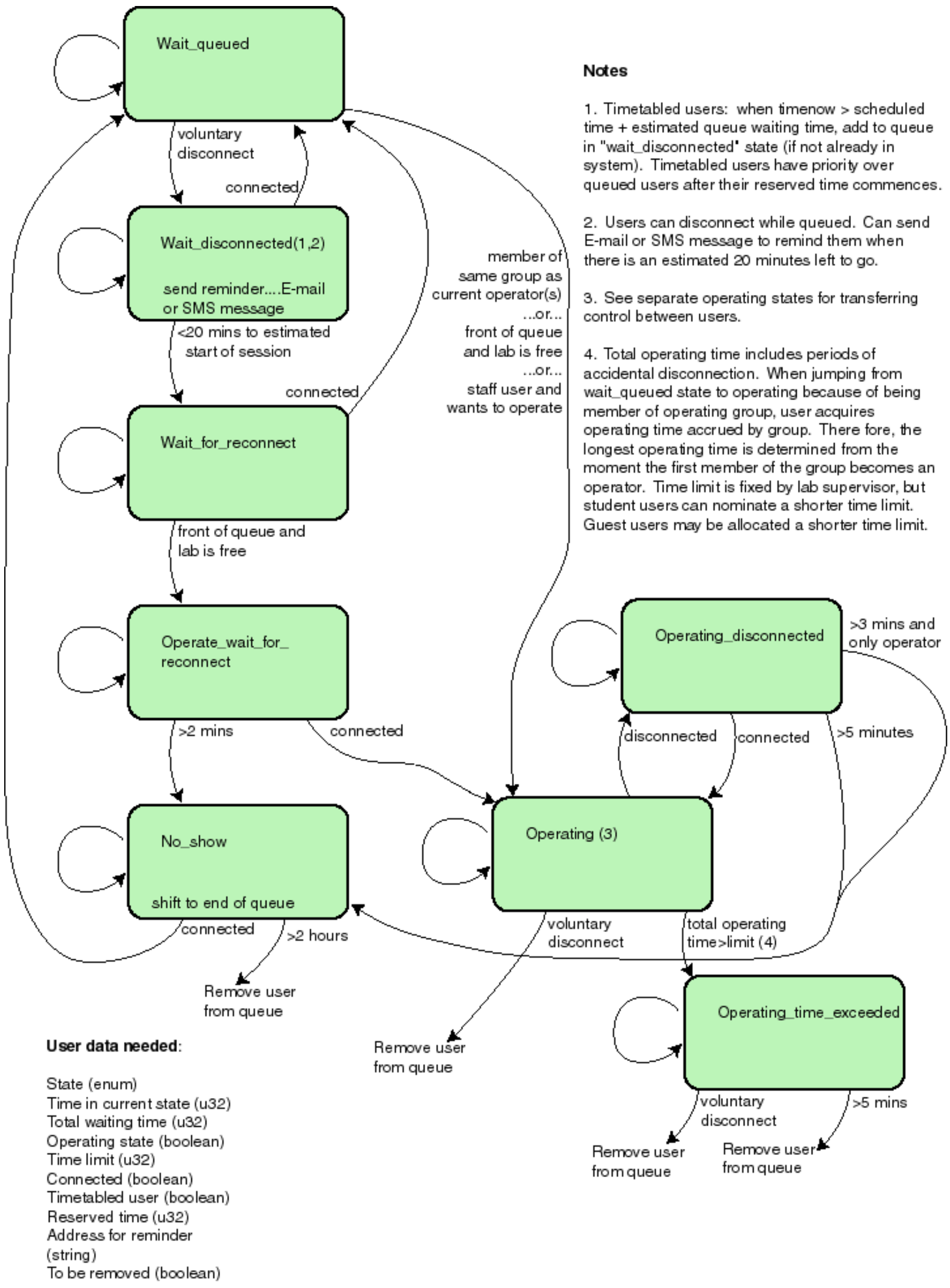


Figure 2. An alternative hardware arrangement that can also be used in our system. In this configuration, the equipment requires high speed local control by a dedicated computer that also contains the remote client. The central server only relays control information, and related data socket transfers. It is also possible that a second data socket server could be located on the control computer to handle higher traffic volumes entirely related to the hardware for this particular rig. The system provides for considerable freedom in configuration in order to reduce potential network traffic congestion.



Notes

1. Timetabled users: when timenow > scheduled time + estimated queue waiting time, add to queue in "wait_disconnected" state (if not already in system). Timetabled users have priority over queued users after their reserved time commences.

2. Users can disconnect while queued. Can send E-mail or SMS message to remind them when there is an estimated 20 minutes left to go.

3. See separate operating states for transferring control between users.

4. Total operating time includes periods of accidental disconnection. When jumping from wait_queued state to operating because of being member of operating group, user acquires operating time accrued by group. There fore, the longest operating time is determined from the moment the first member of the group becomes an operator. Time limit is fixed by lab supervisor, but student users can nominate a shorter time limit. Guest users may be allocated a shorter time limit.

User data needed:

State (enum)
Time in current state (u32)
Total waiting time (u32)
Operating state (boolean)
Time limit (u32)
Connected (boolean)
Timetabled user (boolean)
Reserved time (u32)
Address for reminder (string)
To be removed (boolean)

Figure 3. This state diagram shows how users are first queued, and then later come to operate hardware. Users can disconnect (either deliberately or accidentally) and reconnect later without losing their place in the queue. Again, users can be (accidentally) disconnected while they are operating the equipment. If they manage to reconnect within 3 minutes (or within 5 minutes if there are other students connected in the same lab group), they will be able to continue their work without having to wait for other users in the queue. This design accommodates experiments that take time to complete. Very short experiments will use the same system, and accidental disconnection is unlikely to occur.