ENHANCING STATISTICS TEACHING WITH A VIRTUAL LAB
A Case Study of Seamless Local and Remote Computing

Araceli Garín
Facultad de CC.EE. y Empresariales, University of the Basque Country, Spain
mariaaraceli.garin@ehu.es

Fernando Tusell
Facultad de CC.EE. y Empresariales, University of the Basque Country, Spain
fernando.tusell@ehu.es

Aitziber Unzueta
Facultad de CC.EE. y Empresariales, University of the Basque Country, Spain
aitziber.unzueta@ehu.es

Keywords: e-learning; remote computing; virtual lab; client-server

Abstract: Some Statistics and Data Analysis courses demand a fair amount of computing. While there are excellent free source tools which can be given away to students, their seamless integration requires a fair amount of work and is challenging to the less computer-savvy students. In the past we have addressed the problem by compiling and integrating the necessary tools in CD-ROM's and providing local computing facilities, but this has proved impractical on a number of counts. Trial and error has led us to finding a setup with which we have finally solved most problems.

1 INTRODUCTION

Teaching Statistics and related subjects such as Econometrics or Operations Research to students of Economics presents various challenges; not least, applied courses on these subjects require a fair amount of practice, all of which entails the use of a computer.

In the old, pre-1990 days, we would provide access to a computer lab, typically made of a mainframe or minicomputer with terminals connected in its close vicinity. Any computer work would be done there or, rarely, from a distant location through a dial up line.

The advent and rapid dissemination of personal computers changed all this. For over a decade, the "PC room", a bunch of PC's possibly connected in a local area network, was the dominant paradigm in ours an many other schools.

In Section 2 we list the advantages and shortcomings of both the old computer lab model and the PC room-based instruction, as we see them. We also explain our moves in trying to cope with perceived shortcomings in our infrastructure. In Section 3 we describe our experience on how a dual-faceted computer Lab, fitted with PC's yet having a work-station as its cornerstone, provides the best of both models, increases the productivity of students and teachers, and can be run with a (relatively) modest input of skilled labour.

2 THE PAST

2.1 The old-style computer lab

The setup described in Section 1, which was prevalent at universities all over the world until the final eighties of the last century, was not without advantages. The requirements of skilled labour were moderate: there was a single machine, or only a few, to be run. Dumb terminals, even X-terminals with graphical capabilities, were an "install-and-forget" type of task, and run until physically worn out.

There was no question of viruses spreading, no questions of illegal copying, and much reduced concerns about security: there was little an inexperienced user could do to cause damage. Since dumb terminals were of little use to most users, hardware theft was also a rarity.

Finally, from the teaching standpoint which is what interests us here mostly, the physical proximity of students having the same problems afforded many opportunities for cooperative learning and mutual help.

The shortcomings, however, are also evident: essentially all work had to be done in one place, and users depended on computer staff for the simplest tasks, such as bringing in new data whose size meant that it could not be typed at a terminal.

2.2 The PC room

The PC rooms changed all of this. Suddenly, the users were masters of themselves. They found at the school the same hardware, the same operating system and software tools that they were used to work with at home. There was no retraining, no learning curve, no dependence on any one to move data, which could travel in floppies or, later, pen drives.

Alas, this very flexibility was not without inconvenients. PC's could be infected and their hard drives erased. In order to have shared services such as storage or printing, local area networks (LAN's) had to be set up. Linking different machines into the same LAN, managing authentication in a centralized manner, ensuring consistency and integrity of the software installed, protecting users from themselves and each other became tasks that dwarfed the effort previously required for the administration of a single machine.

Since, in addition, hardware and software have sort useful lives because of technical obsolescence, managing computer rooms is a task extremely demanding of resources. When all is taken into consideration, the advantage of standard, off-the shelf hardware is negated by the complexity of the installation and maintenance.

2.3 The "help yourself" approach

The increase in availability and quality of free software opened new perspectives: software could be given away to students in a CD or DVD and, at least for work requiring only moderate resources, they were able to work at home or wherever they could bring their laptops. It seemed for a while that providing computer resources to students was no longer a problem.

It soon became evident that this was not the case. For one thing, preparing, customizing and integrating all the necessary tools is a lot of work —and a work that needs to be redone frequently so that the software stays reasonably current. On the other hand, installation in a variety of hardware, with different operating systems or different versions of the same operating system, requires much testing, is very error-prone and in our case meant that a sizeable proportion of the students failed to have working installations.

Another drawback is that commonly used and voluminous information (like digital cartography or time series data collections) is difficult to share and keep current by way of handing over CD-ROM's.

From the point of view of learning, this approach also meant individual work, with greatly diminished opportunities for interaction among students —a factor whose importance cannot be overemphasized, as in our experience they learn much more easily applied skills by interacting among themselves than in isolation.

3 THE VIRTUAL LAB

3.1 Hardware and software setup

Around 1999 it became obvious that none of the approaches we had tried had been fully, or even moderately, successful. In an attempt to regain the advantages of the old-style computer lab and keep the advantages of modern PC's, we decided to mix both; the success has been above our expectations, and we think this success is the outcome of a delicate interaction of several factors, which we did not quite foresee.

We secured the premises and asked for the funding of a new lab, named Laboratory of Quantitative Economics (LQE) after its intended users, graduate students of said specialty. The design goals were:

1. It should provide a place for interaction among students. Thus, each of them would have his or her own desk and personal computer and there would be some facilities to be shared, like a printer, and an area for socializing.
2. It would support work in Statistics and Econometrics, and target areas that we felt in need of a boost, like Spatial Statistics and Data Mining.
3. It would have to be run on a very low overhead, without requiring dedicated staff.
4. It would be based on free software, so students willing and able to do so could replicate whatever they found useful in their private machines.

The layout of the Laboratory is completely standard and can be seen in Figure 1. A machine is acting as a server, providing among others authentication, file storage and printing services. A number of PC's (we usually have between one and two dozen machines running) are networked in the same Ethernet bus, currently 100Mbit Ethernet. Everything is connected to the Internet.

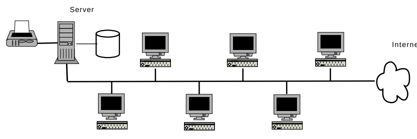

Figure 1: Sketch of arrangement at the Laboratory for Quantitative Economics.

The server is a 64bit machine running Linux; we use Debian[1], and are satisfied with it, but a number of other distributions offer also 64 and 32 bit versions of Linux. What is essential, as we will discuss below, is that the same software is available in the server and the client machines.

Students have each his own desk and PC. All PC's are fully autonomous machines running 32 bit Debian Linux. However, user files are stored in the server; the PC's mount the relevant directory so it appears to be local; this is transparent to the user.

Software installed in both the server and PC's include editors and word processing tools (we encourage the use of Emacs, (Stallman, 1997), and LaTeX, (Lamport, 1994), but users are free to use the Open Office suit or other tools), statistical and econometric software (R, (R Development Core Team, 2008), and Gretel[2], among others), GIS tools (Qgis[3], GRASS[4]), database tools (PostgreSQL[5], PostGIS[6]), graphics programs like Gimp[7], and an assortment of ofimatics and productivity tools.

The important point is that *the same* software (with very few exceptions, which are not available or are only available in a different version for 32 and 64 bit machines) is installed in the server and PC's. This brings about two benefits:

- Users can at any point log to the server to start jobs too large for their PC's. The server is a multicore machine each of whose cores runs typically 3 or 4 times faster than a single PC. Since all user files reside in the server, no file movements are involved.
- More importantly, each user is able to log remotely and see exactly the same environment he or she would see in the Lab. All the software and files are there: there is no need to carry anything in pen drives, to upload or download files.

For the second benefit to be fully realized, it is important that they have suitable means at home. Nowadays, personal machines are ubiquitous, and almost every student has one; most have also high speed Internet access.

While the above setup provides a feasible working environment, we did not reap the full benefits until we added a last and very important piece of software. There are a number of X-terminal software emulators allowing a remote machine to serve as an X-terminal. However, the protocol is fairly verbose, and the amount of information to be transferred fairly voluminous: every keystroke or click requires processing on both the local and remote machine. In our experience, this leads to latency times which are unacceptable for interactive work. Response is much too sluggish when graphical applications are used.

On the other hand, installing these emulators at home (usually on machines running various versions of Windows) was a hurdle for the less computer-savvy users, a problem compounded with the fact that additional layers of software are necessary for the underlying Secure Shell (ssh) protocol.

This problem has been much alleviated by the use of software which compress and streamline the X protocol. Several such pieces of software are in existence using the so-called NX technology[8]. After some experimentation, we have settled for the freely available (although not free source) implementation of NoMachine[9].

With this emulator installed, interactive work becomes almost as fast in a remote machine as it is locally. The only problem we have found is that X-terminal emulation is impractical or even impossible with some notebooks supporting only low screen resolution modes.

3.2 Administration.

It may appear that the setup described shares the problems associated with PC rooms. This is not the case. It is true that each PC has to be installed, but the process can be automatized to a large degree[10], especially if you have control over the network and can have the machines boot from a remote image. Even if you can't, the process of inserting a CD-ROM on each PC and answering a few questions is not overwhelming for small to moderate size labs, as it has to be done only once.

Once a minimal installation is working on each machine, everything else, from day-do-day maintenance to installation of new operating system versions or new software, can be done automatically, even for heterogeneous hardware. A tool that we have found invaluable for this purpose is Cfengine[11]. Usually the

[1]See (Krafft, 2005) or `http://www.debian.org`.

[2]See `http://gretl.sourceforge.net`

[3]See `http://www.qgis.org/`.

[4]See `http://grass.fbk.eu` or (Neteler and Mitasova, 2007).

[5]Described in `http://www.postgresql.org` and also in (Momjian, 2001).

[6]See `http://postgis.refractions.net/`.

[7]See `http://www.gimp.org`.

[8]For a description and overview of alternatives, see `http://en.wikipedia.org/wiki/NX_technology`.

[9]Available at `http://www.nomachine.com.`.

[10]See for instance FAI, Fully Automatic Installation, `http://fai-project.org`.

[11]Both free and commercial versions exists; we have

lab runs for months with no or only minimal need of manual intervention.

3.3 Benefits

The LQE has led to a tremendous increase in productivity for students, and eliminated the amount of time previously spent in installing software and transporting files, with the following advantages:

1. Instant availability, local and remote, of data which is not simple or practical to duplicate: microdata, digital cartography, etc.
2. Increased flexibility in the flow of information. Students, from wherever they happen to be, can spool files directly to their instructors' desk, print them in the lab's dedicated printer or send them in PDF format. This last option has been much used, and affords entirely paper-free interaction. Commenting and/or grading of papers is accomplished with PDF editors[12].
3. Good interaction among students, while physically at the LQE.

It is true that interaction and cooperative learning are not at the same level that the old-style computer lab forced; we estimate that about 80% of the time students work remotely, thus reducing their chances of mutual assistance. We try to enhance student-to-instructor and student-to-student interaction by exploiting forum facilities in e-learning platforms —like Moodle[13], one of the two adopted at our institution.

3.4 Validation

At the time of writing, the LQE has been in operation for over three years, and is therefore a well established experience well beyond a proof of concept. Similar experiences target energy efficiency, (NComputing, Inc., 2010), enhancing active and cooperative learning, (Moor, 2006), or web-based arrangements (see (Jalobeanu, 2006) and references therein). Closer to our design goals is (Nixon and Dwolatzky, 2002), although they mainly address (back in 2002) presential labs.

4 CONCLUSIONS

Our experience shows that a dual-faceted lab, offering both local and remote resources, is a feasible alternative, and relatively cheap to build with off-the-shelf hardware and (mostly) free software. Its success in our case has been critically dependent on a) Being able to offer a user interface that looks almost exactly the same, whether locally or remotely; b) The general availability to students of wide band connections, and c) A simple, "click-and-install" piece of software that handles remote connections efficiently. The downside is reduced student interaction, but use of fora (and, in the future, chats), may go a long way to alleviate that.

used the free version, `http://www.cfengine.org.`.

[12]We use to this effect the free tool Jarnal, `http://jarnal.wikispaces.com`; alternatives exists.

[13]In `http://moodle.org`.

ACKNOWLEDGEMENTS

Funding from grants IT-347-10 (Basque Government) and ECO2008-05622/ECON (Ministry of Science and Innovation, Spain) is gratefully acknowledged.

REFERENCES

Jalobeanu, M. (2006). Internet - a virtual laboratory for distributed computing. In Auer, M. E. and Ursutiu, D., editors, *2006 International Remote Engineering Virtual Instrumentation Symposium*.

Krafft, M. F. (2005). *The Debian System (Concepts And Techniques With Cdrom)*. No Starch Press.

Lamport, L. (1994). *A Document Preparation System. LaTeX User's Guide and Reference Manual*. Addison-Wesley.

NComputing, Inc. (2010). Santa Ana College creates green lab with virtual desktops.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Momjian, B. (2001). *PostgreSQL : Introduction and Concepts*. Addison-Wesley.

Moor, S. S. (2006). Case study: Renovating a computer teaching laboratory for active and cooperative learning. In *Proceedings of the 2006 Illinois-Indiana and North Central Joint Section Conference*. American Society for Engineering Education.

Neteler, M. and Mitasova, H. (2007). *Open Source GIS: A GRASS GIS Approach)*. Springer Verlag.

Nixon, K. and Dwolatzky, B. (2002). Computer laboratory infrastructure in engineering education-a case study at wits. In *Africon Conference in Africa, 2002. IEEE AFRICON. 6th*, volume 1, pages 431 – 436.

Stallman, R. M. (1997). *GNU Emacs Manual*. Free Software Foundation.