tuXlab Cookbook

Sean Wheller <sean@inwords.co.za>
Graham Goode

by Sean Wheller and Graham Goode

Published 2007-01-17 11:06:10

Contents

List of Figures

List of Tables

Preface

1. Audience

The subject of a tuXlab brings together computers and education. As such the audience for which the tuXlab cookbook is intended includes a diverse spectrum of people with either computing or educational backgrounds, or both. Hopefully, everyone reading this book shares a common interest in applying computers in educational environments in order that they may improve learning techniques and provide a place where computing and information technologies may be taught.

2. Scope

A tuXlab is a challenge - experience has shown that it takes varying degrees of input from people with different skills in order to make a tuXlab successful. Typically, each of the people participating in a project do what they are good at. Each plays a role, assuming responsibility for one or more functional perspectives of the project.

The scope of the tuXlab cookbook extends to cover all such perspectives so that the collective experience of many may be distilled and captured in order that such knowledge may be preserved in a manner that it can be referenced and transferred into the future.

3. Organisation

The tuXlab cookbook is a set of books. Each covers a specific perspective of a tuXlab that requires consideration and attention in the planning and management thereof. While the books are presented and bound as a single volume of work they are intended to be used as independent pieces. For this reason the tuXlab cookbook is bound in a large lever-arch style file that enables books to be conveniently stored at the tuXlab and, individual books, temporarily removed so that they can be read or duplicated independently.

Nobody is expected to know or be an expert in everything, it is however useful for everyone involved to have some degree of overview about what others are doing or what their role involves. Given that people in a project will bring different skills and have different interests, it is recommended that people read this book and then those books related to their role or responsibility the project.

Organisation of the books is as follows:

Introduction	The document that you are reading.
Software Module	The TuXlab Software Module covers the specific software that is used within a TuXlab. The first section reviews the topic of "Open Source

	Software" that is covered within the TuXlab Cookbook Introduction Module. The second section presents the information regarding the software components used; including the LTSP classroom server, the Wizzy server, specific applications (OpenOffice.org, Mozilla, educational software, wikipedia, and edutainment software), Project Gutenburg, Connexions, and other resources. The final section deals with the process of installing the software, including general information about installing a Linux distribution.
Hardware Module	This module deals with the hardware requirements, setup and installation, network creation, and server configuration for the TuXlab lab. The first section deals with the concepts of "thin-client" computing and use of the Linux Terminal Server Project (LTSP). The second section deals with the requirements of the thin client and server computers, including information regarding the refurbishment of older computers to use as thin-clients. The third section presents a discussion of networking, from the question of "Why network" to descriptions of the equipment needed and how to use them. The fourth section deals with the various configuration processes that need to be followed when setting up a TuXlab. The fifth and final section then deals with an evaluation of the hardware usage of the TuXlab, mentioning benefits and drawbacks. Once you have read through this module you will have been introduced to the physical machines that form the core of a TuXlab.
Infrastructure Module	This module deals with the physical infrastructure that makes up the premises of a TuXlab. The first section deals with security issues regarding windows, ceilings, and doors. Along with the discussion on lab security are recommendations for alarm systems and the type of alarm response company that should be used. The second section deals with the infrastructure requirements within the TuXlab room. This includes details about workstation desktops, the server cabinet, electrical requirements, and the network infrastructure.
Training Module	The TuXlab Training Module deals with the process

	of training TuXlab volunteers. The first section presents an overview of the volunteer training course structure. The second section presents information about the training incentives scheme offered by the Shuttleworth Foundation to TuXlab's volunteers. The final section then presents the contents and levels contained within the three training modules.
Management Module	The management module deals with the human processes of setting up a TuXlab. The first section deals with the procedures that need to be followed when applying for a TuXlab, and an overview of the installation process. The second section deals with the aspect of "Change Management", presenting information regarding executive buy-in, the computer committee, the idea of "clustering", and using a TuXlab to bring services to the greater community. The third section deals with the process of management after a TuXlab has been installed and is in use. This information includes discussion and examples regarding reporting, training, budgeting, and involvement in the incentive scheme.
Support Module	The Support Module deals with supporting and troubleshooting a TuXlab computer lab. The first section deals with the TuXlab support desk, including information on essential and extra software tools and physical tools used to find and solve problems. The second section deals with specific troubleshooting and problem solving topics.
Volunteer Module	The Volunteer Module deals with the concept and process of using volunteers within the tuXlab program. The first section presents an overview of the concept of using volunteers, including a discussion on a broad strategy and approach to finding and involving volunteers. The second section deals with the process of including and training volunteers, and utilising their skills when setting up tuXlabs.
Sustainability Module	This module deals with the concept of Sustainability. The first section covers an overview and definition of sustainable development and suggests areas where sustainability should be applied to the TuXlab project. The second section

deals with real life examples of TuXlab sustainability in practise.

4. Acknowledgements

A special word of thanks is extended to the organisations and people whose generous contributions and support have helped make the tuXlab Cookbook into the valuable resource it is today.

The information contained in this book comes from many different people. We have used the content of the first tuXlab Cookbook written by Jean Jordaan as a base for many of the modules and have added more recent information and new content. The illustrations were created by Leonora van Staden. Information was also provided by Jason Norwood-Young and Hilton Theunissen.

5. Conventions

The following admonitions will be found throughout the book:

A note presents interesting, sometimes technical, pieces of information related to the surrounding discussion.

A tip offers advice or an easier way of doing something.

A caution advises you of potential problems and helps you to steer clear thereof.

A warning advises you of a hazard condition that may be created in a given scenario.

Cross reference conventions for print will be shown as follows:

- Internal references will look like this: Section 1.1, “Overview” [1–1]. The numeral contained by square braces is the [page number].
- External reference, say to a Web Site, will look like this [http://www.inwords.co.za].

PDF, HTML or XHTML versions of this document will use hyper-links to denote cross-references.

Type conventions will be shown as follows:

- File names or paths to directories will be shown in **`monospace`** type.
- Commands that you type at a prompt will be shown in **bold** type.
- Options that you click, select or choose in the user interface will be shown in **`monospace`** type.
- When variables, parameters, SGML tags, etc. are contained within a paragraph of text, they will be shown in **monospaced** type. Otherwise they will use the normal type.

Menu selections, mouse actions, and keyboard short-cuts:

- A sequence of menu selections will be shown as follows: File → Open
- Mouse actions shall assume a right-handed mouse configuration. The terms "click" and "double-click" refers to using the left-hand mouse button. The term "right-click" refers to using the right-hand button.
- Keyboard short-cut combinations will be shown as follows: Ctrl+N. Where the convention for "Control", "Shift", and "Alternate" keys will be Ctrl, Shift, Alt, respectively, and shall mean to hold down the first key while pressing the second key.

Code conventions:

- Code and mark-up samples will be formatted in a grey block.
- Sometimes lines of code or mark-up examples will be longer than the page width. To avoid their running off the page, the slash character "\" is used to denote a soft line break. This means that the line of code is in reality meant to be on one line, but for print formatting it has been broken into two lines.

1

Introduction

This module presents an introduction to the TuXlab Cookbook in it's entirety - this module plus the other eight. The first section presents an overview of the TuXlab project. This includes discussions regarding the motivation and background behind the project, as well as the objectives of the TuXlab project. The second section presents an overview of the TuXlab computer laboratory. The general physical make up of a lab and the software components that are used are discussed. The third and final section presents an introductory discussion about software; including information regarding operating systems, distributions, and the Open Source Software movement.

1.1. Overview

For those reading this book who have never heard of a tuXlab or who have heard of about the tuXlab project but are unsure of what it entails, we have provided this overview. Just like a satellite image of a town provides general detail about the layout and location of buildings and streets but cannot show you the street signs or the people walking along the roads, so this overview provides only a general understanding of the project as a whole. You will need to read each of the sections of the tuXlab cookbook AND work with a tuXlab project before you are fully aware of all the details.

1.1.1. Motivation

There are many motivating factors for the initiation and perpetuation of the tuXlab project. There are economical, social, and personal reason's involved. One of the basic motivating factor's regarding the tuXlab project is the lower costs involved in creating and maintaining a tuXlab. The software that is used is Free and Open Sourced Software, able to be freely distributed and modified. The hardware used in a tuXlab can also be donated second had computers, further reducing the costs of the tuXlab and encouraging the cooperation and involvement of local businesses and partnerships.

The tuXlab project involves the whole school community. The teachers have to know how to set up and install the system, and are empowered through these skills to teach what they have learned to others. The students who use the tuXlab have their own local email addresses and profiles, establishing their ownership of the tuXlab project. Both the teachers and the students can look at the programming code that creates the applications that they use and learn how to produce and improve it.

All of the above reasons allow us to say that the tuXlab project is motivated by community need. Primarily the need for improved education, but also the need for community

upliftment and community ownership. These community based needs are not only present in South Africa, but are present in many other countries in Africa and around the world.

1.1.2. Background

The tuXlab project has its roots in the vision of the Shuttleworth Foundation and in the Foundation's commitment to improve the standard of education in South Africa. This is a huge task with many facets, and just putting computers in schools is not nearly enough. The one critical factor in education is the people involved: the teachers and the learners they have to educate. For computer labs to make a difference, the people who use them must understand them and be able to use them for anything that they can imagine. They must be able to take ownership of the computers and the software that they run, and to create new learning material to share with each other and with society as a whole.

1.1.2.1. Open Source Learning Centres Background

In 2002 the Shuttleworth Foundation started to actively promote the use open source software as a computer lab solution for schools. The Foundation funded several projects to establish open source based computer labs in schools, as well as an internally initiating a pilot to prove the effectiveness of open source software as a school computing solution. The Foundation also initiated a project to facilitate the involvement of volunteers in refurbishing computers and establishing open source learning centres.

Based on the success of the pilot and volunteer project, the Foundation extended the pilot to establish a further 80 open source learning centres in the Western Cape. This goal was reached in November 2004.

1.1.2.2. Original Shuttleworth tuXlab Program Outline

Under the guidance of the Shuttleworth Foundation, both primary and secondary schools were involved. In order to maximise the impact of the program, schools were (and still are) selected in clusters. Through clustering, schools are able to plan the use of the centres together, and they can share resources and community support. We believe that clustering is the best method of ensuring that schools remain self-sufficient in terms of support in the future. This is supported by the *Computers in Schools* survey of 2000, which states (my emphasis):

> Experiences from other countries, whatever their stage of development, show that factors which accompany the successful implementation of ICTs in schools are *networks of connectivity* and *structured and continuous programmes* to train teachers to use the new technology for educational purposes.
>
> — Computers in Schools, 2000 [http://www.school.za/research/uwc-epu/screen/Executive%20Summary.pdf]

Within a cluster, schools are selected based on criteria originally set by the Foundation. At a very minimum, these criteria include the availability of a secure computing environment, guaranteed commitment of the school (including governing body) to the project, as well as proof of comprehensive plans to introduce the tuXlab into school activities. Additional criteria could be set for each cluster, depending on the needs of that community.

In 2005, hundreds of tuXlabs were installed throughout South Africa in partnership with local organisations and user groups. The project was also launched in Namibia, and reports of roll-outs overseas (such as a large-scale project in Azerbaijan!) have also been communicated.

The tuXlab programme is now a fully fledged organisation. From it's beginnings in the Shuttleworth Foundation it is becoming a self organising, self sustaining entity that continues to bring quality information technology and infrastructure to the education system.

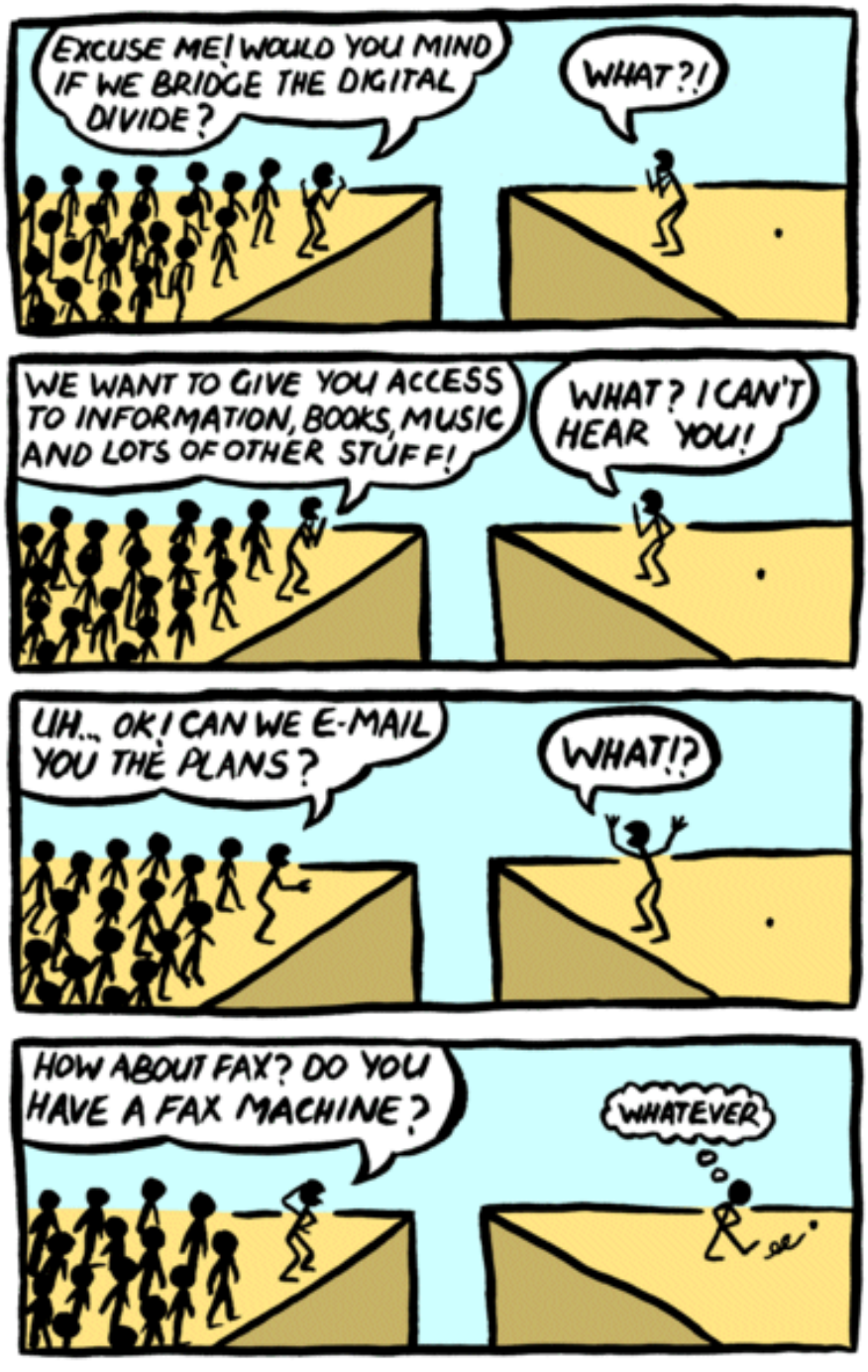
1.1.3. Objectives

While an understanding of computers and information technology is critical in our modern global society, they are not a sufficient goal in themselves. There are many other fields of knowledge that beckon to be explored, and to be discussed and debated with other people. However, publishing and distributing books on paper is expensive, and unless you are in a big city with the means to travel around easily, it is awkward and costly to take part in the global conversation. Information technology offers us a chance to leapfrog these problems by providing access to vast resources of texts, curricula, art and music via the internet, and allows us to stay in touch by electronic means, even in remote areas.

Without some degree of mastery of technology it is easy to miss the debates of real importance on the internet. To understand legislation concerning intellectual property, access to information and privacy issues, it is crucial to understand what the technology offers and how it differs from the possibilities of the past. In a sense, this is the real meaning of the digital divide: if you don't understand how computers and networks can be used, you can starve even in the midst of plenty of conversation and information.

To accomplish these objectives of social innovation and empowerment, and to further the uptake of technology in South Africa, the Shuttleworth Foundation launched the tuXlab project. The objective of this project is to establish a national network of computer laboratories based exclusively on open source software. The tuXlab project strongly believes that open source software should be the preferred choice of software for schooling in South Africa. Open source software provides users with freedoms not obtainable from proprietary software. This includes the freedom to obtain, use, modify and distribute the software.

Great, so now that we know what we have, the question is: "What can we do with it?" We can see communities using the tuXlab project to do three things:


EXCUSE ME! WOULD YOU MIND IF WE BRIDGE THE DIGITAL DIVIDE?
WHAT?!
WE WANT TO GIVE YOU ACCESS TO INFORMATION, BOOKS, MUSIC AND LOTS OF OTHER STUFF!
WHAT? I CAN'T HEAR YOU!
UH... OK! CAN WE E-MAIL YOU THE PLANS?
WHAT!?
HOW ABOUT FAX? DO YOU HAVE A FAX MACHINE?
WHATEVER

- Begin to **bridging the Digital Divide**.

  A tuXlab can provide access to information, books, music, news, and myriads of other resources. It can also provide new ways to communicate with your peers, near and far, and open channels of communication to organisations that may be hard to reach because they are far away or widely distributed.

  In order to manage this, we must build something sustainable, unbreakable and flexible. Instead of leaping ahead, for example by accepting an expensive lab that we cannot maintain, with proprietary software that we cannot study, we must build steadily from the ground up, so that we have a foundation that will last. To do this, we have to keep some things in mind:

  - The components of the tuXlab must be as cheap as possible. We must make use of what is available, and make the parts generic and interchangeable. We must also use standards, for example TCP/IP - for networking.
  - The tuXlab shouldn't require a permanent internet connection in order to access cultural goods, and enable participation in the culture.

    Especially in South Africa, a high-bandwidth internet connection is incredibly expensive. Wireless service providers are just starting to appear in the big urban centres, but even they cost hundreds of Rands per month, and that's for home users.
- In this way, we **foster self-reliance and create local expertise, while building an international community**.

  In one of his essays, Richard Stallman writes about the importance of free software in developing local IT expertise:

  > Free software permits students to learn how software works. When students reach their teens, some of them want to learn everything there is to know about their computer system and its software. That is the age when people who will be good programmers should learn it. To learn to write software well, students need to read a lot of code and write a lot of code. They need to read and understand real programs that people really use. They will be intensely curious to read the source code of the programs that they use every day.
  >
  > —Richard Stallman, http://www.fsf.org/philosophy/schools.html [http://www.fsf.org/philosophy/schools.html]

- Lastly, we can also **save money**.

  While it's essential to spend money on education, we have to make sure that the money goes as far as possible. By using only free software in tuXlabs, you save money in a couple of ways:

  1. There are no software license fees to be paid. The Linux operating system is world wide, stable, and FREE.

In addition, for every Windows software product we have included a Linux alternative that looks for all intents the same. For example, a word processor (that can read Microsoft word document files, incidentally), a spreadsheet, a publisher, an HTML editor for creating web pages, a typing tutorial, etc. [1]

2. By using free software and open file formats that you can read without needing expensive software, we save money for the community.

   Whether they intend to or not, teachers make their learners' families buy proprietary software if they use it at school. By using free software that learners can take home, the school helps the community to save money.

1.2. Lab Overview

1.2.1. tuXlab Components

Maybe you've just taken acceptance of a brand new tuXlab, or perhaps you're just dreaming about when it will all be done. Whatever the case may be, let's take a walk through the completed lab to see how everything fits together.

1.2.1.1. Lab Room

Every tuXlab consists of a room where people may come to use the facilities of the lab.

The room is secured with a gate and burglar bars on the windows, and the really expensive components of the tuXlab are locked away even further, in another room or in a cage.

The room provides a comfortable space to work in, with desks at the right height for the learners at the school, and with enough plugs and cables for all the computers. All the users of the tuXlab may sit down at any of the workstations and log in, to find their working environment just as they left it, even if they u logged in on a different computer.

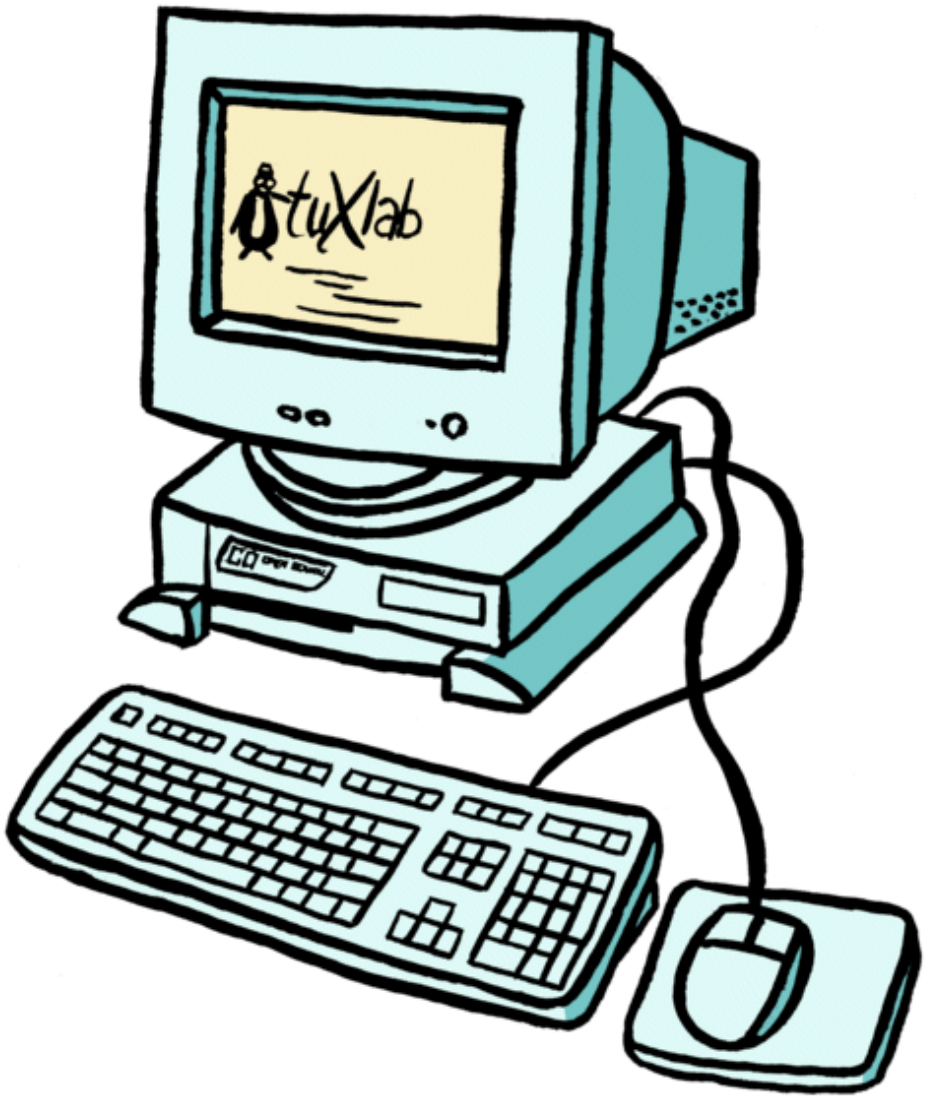
1.2.1.2. Computers

A tuXlab usually contains between 20 and 25 workstation computers, although you may add more workstations if you already have some computers, or if you can get donations. These workstations are connected to a server computer by way of a high-speed network.

The workstations are where the rubber hits the road, so to speak. They are standing out there on the desks, and everyone who uses the tuXlab is constantly using their keyboards and adjusting their screens, and so they do undergo some wear and tear. Because they are intended to be accessible to use, they are not that easy to secure. If something is locked away, it's hard to learn how to use it. To deal with this state of affairs, tuXlabs are

[1] This paragraph is from http://wizzy.org.za/article/articleview/4/1/3.

designed so as to make the workstations as cheap and as easy as possible to replace. They should be completely interchangeable, and they store nothing: no documents, and no information about any user.



A classroom workstation

The server actually does all the work in a tuXlab. If the server goes away, the workstations will just stand around scratching their heads, like bees doing nothing when something happens to the queen bee. Whatever you see when working at a tuXlab workstation has been sent to the workstation by the server over the network. All the documents you save, and all the information regarding users, everything is stored on the server.

A classroom server for a TuXlab,

1.2.1.3. Software

The software falls roughly into two categories: the operating system software, and applications. All the computers in the lab run the Linux operating system (the workstations fetch their copy from the server upon startup). This enables the server to display the applications it is running for all the workstations on the screens of the workstations themselves, using the **X Windows** system, the graphical windowing environment used by Linux.

The applications installed in a tuXlab are focused on an educational environment, and include software that is essential for general computer literacy such as word processors, spreadsheet and web browsers, as well as educational software that allow learners to practise skills (e.g. typing, arithmetic) and knowledge (e.g. spelling, geography). Besides these, a tuXlab contains a great variety of programming languages, tools, texts and examples that can be used to teach programming and to study how existing programs work, all the way from first principles to systems architecture. Nothing is proprietary: you may examine the source code of every component in the system.

1.2.1.4. Network

The final critical component of a tuXlab is the computer network that connects all the machines in the room. If you're looking at a finished tuXlab, the network cabling shouldn't be terribly obvious. However, every workstation has a Ethernet cable plugged into it, and you should be able to see the switch cabinet where all the cables go. However, if you were to look inside the trunking running along the walls or under the desks, you would see that there is an Ethernet cable for each and every workstation, connecting it to the network switch. The tuXlab server is also connected to the switch by a fly-lead (don't worry if these words don't mean much to you yet... as you read through the infrastructure and hardware sections you will gain a better understanding of what we are talking about).

In this configuration, it is as if every workstation in the tuXlab is connected directly to the server. The switch itself is transparent to the network. It sees to it that network traffic from the server is sent as directly as possible to the workstation for which it is intended. From the perspective of the computers, it looks as though they're all connected directly to each other.

The role of the server in the network is key to the functioning of a tuXlab. The server is the centre of tuXlab environment. It may be easier to think of the server as a powerful computer that is able to split itself up into many smaller, virtual computers. As a workstation computer is switched on it first communicates with the server computer. The server creates an individual space within itself that it identifies as that workstation. The workstation provides the screen, the mouse, and the keyboard, but all the applications, memory usage, and disk space are part of the server. All that the workstation has to do is take whatever has been typed or clicked and communicate this through the network to the server, and then display whatever response the server makes using that input.

If anything goes amiss in a tuXlab, there's a pretty good chance that it may be a problem with the network, since (from the workstation's point of view) a broken network is just as bad as a broken server. It can't do anything in either case.

1.2.2. Operating Software

There are many different distributions Section 1.3.2, “What is a distribution?” [1–19] of Linux that can run the tuXlab system. The following four distributions are specifically made for educational purposes and are the most likely to be installed with a tuXlab environment.

1.2.2.1. Edubuntu

Edubuntu is a Linux distribution designed for use in schools and classrooms. It is a branch of the Ubuntu Linux project, also begun by the Shuttleworth Foundation. Included with Edubuntu is the Linux Terminal Server Project, a large number of educational applications, including GCompris (educational software for children aged 2 - 10), the KDE Edutainment Project (Language, Math, Science) and Schooltool Calander.

1.2.2.2. Openlab/Openlab4

OpenLab GNU/Linux is the oldest African developed GNU/Linux distribution. Started in 2001, the product has built up a well deserved reputation for ease-of-use, innovation and user-oriented design.

The latest stable version (4.0 - Perdita), was released in September of 2005. It comes with the new installable LIVEcd, greatly improved OLAD (OpenLab ADministration tool) and many other innovations ranging from back-end systems such as the hyperdrive suite which completely de-complex-ifies the handling of removable media to user-level enhancements such as the highly integrated desktop theme featuring the award winning Nuvola icon set.

OpenLab GNU/Linux was initially developed as a niche solution for the education sector, but over its four-year life-cycle, OpenLab has grown into a complete desktop operating system. OpenLab is well suited for the home desktop, small business and many other environments, where it usually leaves other contenders far behind. Developed using principles from extreme programming, with constant involvement of a wide-range of participants in the process with frequent re-evalutation of designs, OpenLab is an innovative, vibrant and above all fun platform.

1.2.2.3. K-12

K12LTSP [http://k12ltsp.org/contents.html] is based on the RedHat Fedora distribution of Linux and the work of the LTSP. It's easy to install and configure. It's distributed under the GNU General Public License. That means it's free and it's based on Open Source software. It was started in the USA and now has *Creative Commons* Courses and Text books. The K-12 project involves the Linux distribution, a community forum, and a development

community around the world. Software bundled with K12LTSP includes web browsers, email, calender, contact managers, OpenOffice, Gimp, AbiWord, file sharing with Windows and Macintosh networks, and auto configuration of many PCI based sound cards and network adopters.

This is quite a mouthful, so I'll unpack the terms one by one.

- The *LTSP* is the Linux Terminal Server Project. This project assembles all the software components that are necessary for a computer to act as a fat server for a network of thin clients, and provides the configuration necessary for the server to function as such.
- RedHat Fedora Linux is the free distribution packaged by RedHat, Inc. The K12LTSP team has added an option to the RedHat installation menu, so that installing a classroom server is as simple as choosing the first installation option and answering some questions.

The K12LTSP distribution tries to make it as easy as possible for you. It is a regular Fedora distro with an option to install LTSP right there in the setup screen. When installing, the LTSP option is the first item on the menu, added above the default Workstation, Server and Custom options. This means that you don't have to mess around with the configuration files until you've had a chance to see what it is they do, and by then you'll probably only need to tweak them a little.

The LTSP server defaults to an IP gateway and firewall when two Ethernet cards are present. This will only be the case in tuXlabs that are permanently online, which will usually not be the case. Normally, the Wizzy server will be the gateway, as all tuXlabs with internet access use a Wizzy server.

1.2.2.4. Skolelinux

Skolelinux is a free software project started in Norway. It's name is derived from the Norwegian 'Skole' which means School. Skolelinux offers four different installation options on their installation CD that will easily install a pre-configured educational network including main server, work stations and thin client server.

1.3. Software Concepts

This section serves as an introduction to many of the concepts regarding computer software. We will examine what the definition of an Operating System (OS) is, the concept of Linux, the definition of what a Linux distribution is, and finally we will examine the world of open source software.

1.3.1. What is an Operating System?

In the earliest days of computers, a whole machine was built to do only one thing, for example numerical integration. There was no clean distinction between hardware and software, as aspects of the program might be reflected by physical switches and jumpers

set on the machine itself.

As computers became more general, the same computer could be programmed to do many different things. All these programs, however, would still have to deal with the hardware aspects of the machine, writing to the printer, reading from the magnetic drum memory, and so on. Since these jobs needed to be done over and over and over again, the bits of code that dealt with them could be shared among all the programs that run on that computer. This shared code, the code that handles the basic tasks any program needs in order to run, was the beginning of operating systems.

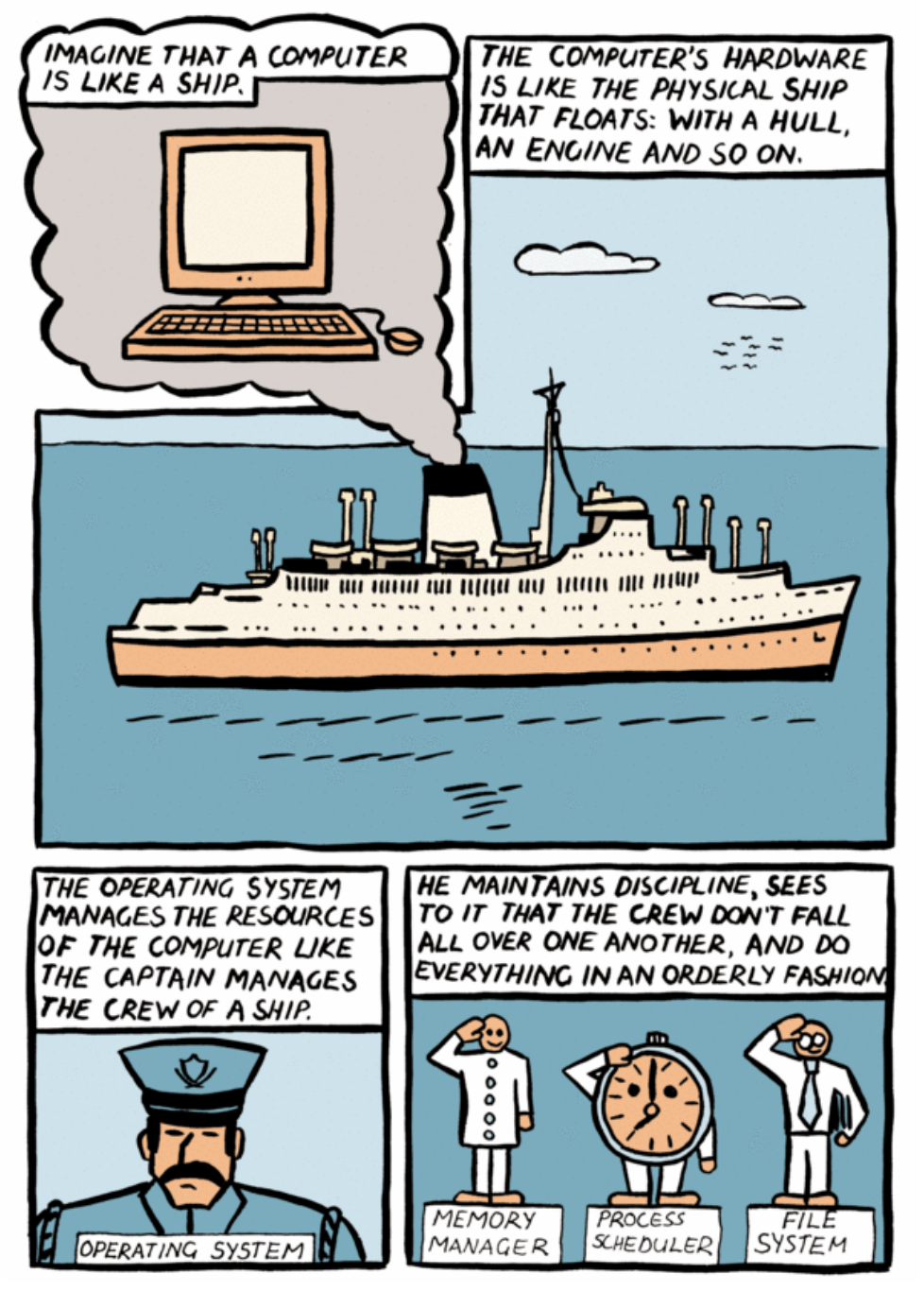
Today's operating systems are complex, sophisticated systems, that can schedule many different programs to run at once, and that provide such a comprehensive range of services that many programs can be compiled to run on many different operating systems, regardless of the variations in the underlying hardware.

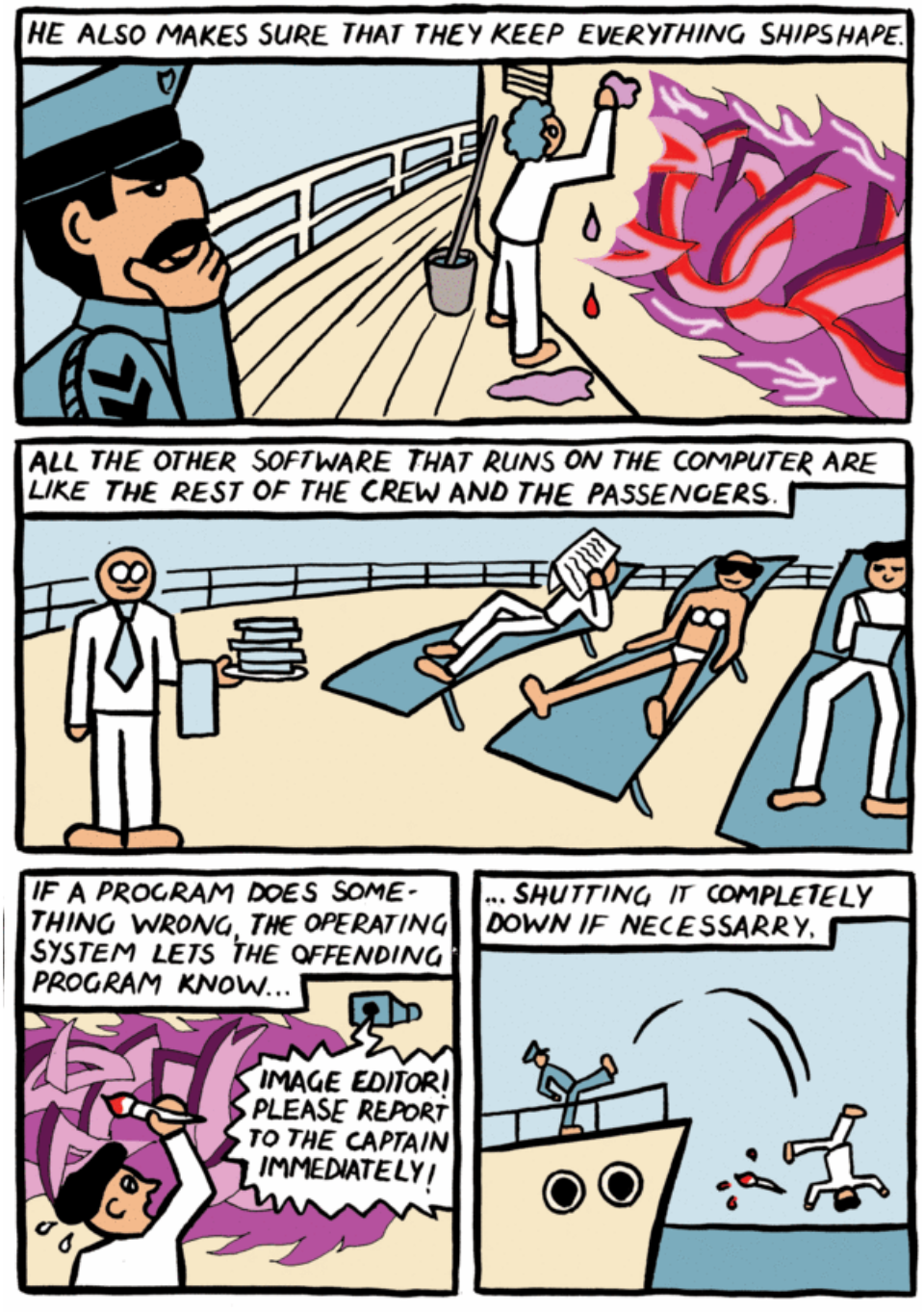
An operating system (OS for short) is the most basic layer of the software, and if your computer is switched on, the operating system kernel is the one program that will always be running. It provides a framework for all the subsystems that make up the computer.

Imagine, for a moment, that a computer is like a ship. The physical hardware is the steel or wooden hull that floats. The operating system is like the officers of a ship, and its subsystems are like the captain, the first mate, the engineer, the cook, and so on. They see to it that the engine is running and that the ship is on an even keel.

All the other software applications that run on the computer are like the rest of the crew and the passengers. The operating system manages the resources of the computer like the captain manages the crew of the ship. He maintains discipline, sees to it that the crew don't fall all over one another, do things in an orderly fashion, and keep everything shipshape.

If a program does something wrong, such as writing to memory that the operating system is using for something else (in the ship example, this is like making a hole in the hull!), the operating system lets the offending program know, shutting it down completely if necessary (it gets thrown in the brig!).


IMAGINE THAT A COMPUTER IS LIKE A SHIP.
THE COMPUTER'S HARDWARE IS LIKE THE PHYSICAL SHIP THAT FLOATS: WITH A HULL, AN ENGINE AND SO ON.
THE OPERATING SYSTEM MANAGES THE RESOURCES OF THE COMPUTER LIKE THE CAPTAIN MANAGES THE CREW OF A SHIP.
OPERATING SYSTEM
HE MAINTAINS DISCIPLINE, SEES TO IT THAT THE CREW DON'T FALL ALL OVER ONE ANOTHER, AND DO EVERYTHING IN AN ORDERLY FASHION
MEMORY MANAGER
PROCESS SCHEDULER
FILE SYSTEM


HE ALSO MAKES SURE THAT THEY KEEP EVERYTHING SHIPSHAPE.
ALL THE OTHER SOFTWARE THAT RUNS ON THE COMPUTER ARE LIKE THE REST OF THE CREW AND THE PASSENGERS.
IF A PROGRAM DOES SOMETHING WRONG, THE OPERATING SYSTEM LETS THE OFFENDING PROGRAM KNOW...
IMAGE EDITOR! PLEASE REPORT TO THE CAPTAIN IMMEDIATELY!
... SHUTTING IT COMPLETELY DOWN IF NECESSARRY.

1.3.1.1. What is *Linux*?

Linux is an *operating system* consisting of a central*kernel* as well as all the hundreds of libraries, tools and utilities that make it usable as a computing environment. The kernel was developed by Linus Torvalds, using the tools and utilities of the GNU project [http://www.gnu.org]. Together, the whole system is called GNU/Linux. Some other common operating systems are the Unix family (including members like Linux, BSD, AIX, Solaris, HP-UX, and others); DOS; Microsoft Windows; Amiga; and Mac OS. An upcoming variety of operating systems, such as Symbian and PalmOS, run on cellphones. There are special-purpose operating systems wherever you look, far more than you would have suspected.

Linux is Free Software. So, not only is it OK to make copies of Linux and give them to your friends, it's also fine to fix things while you're at it --- as long as you also freely provide your modified source code to everyone else. When you're doing this, you're not just providing a freebie to the people who get your fixes: you're also exercising your right to influence Linux, and to change the way that you want it to work. In return for this right which has been granted to you, you must allow those who come after you the same freedom in making use of your work. The issue is *freedom*, not price [http://www.gnu.org/philosophy/free-sw.html].

Charging for Free Software

How can people build businesses on software if everything must be given away? Actually, you're welcome to ask money, but then people are paying for your expertise and services, e.g. assurances of support that you may give them. You may also provide added value, such as attractive packaging and shipping. Only the source code itself must be available at nominal cost.

No one company or individual "owns" Linux. It was developed, and is still being improved, by thousands of programmers all over the world. Some are supported by businesses that make money from their work, and some are volunteers who like to help people. Some are scientists who need computers to get their work done, and who find it convenient to use Linux because they can easily adapt it to do exactly what they require to get their research done.

1.3.1.2. Which operating system is best?

Our consumer culture puts huge stress on having "the best". Usually this means wearing some fashionable brand of jeans, or driving a fancy car. Obviously this is a very superficial measurement, and we have to look deeper when trying to compare operating systems.

It's important to remember that computer science is relatively young. I don't think that any operating system has been in continuous use for longer than a single human lifespan. At

the dawn of the computer age, there was a great explosion of diversity, like in the Cambrian period in prehistory, when different designs and architectures proliferated. This was succeed by a period of consolidation, with the result that today (even though the details "under the hood" may be very different) a great deal of consensus has emerged about the types of services an operating system should provide. [2]

[2]This consensus is reflected in *standards*, such as the Portable Operating System Interface [http://www.pasc.org/#POSIX] (POSIX).


THE VERY BEST OPERATING SYSTEM EVER!
MY OPERATING SYSTEM IS SO SOPHISTICATED THAT IT COMPILES GROCERY LISTS AUTOMATICALLY!
THAT'S NOTHING!
MINE HELPS ME TO COLOUR-COORDINATE MY OUTFITS EVERY DAY.
NO, THAT'S NOTHING!
MINE CAN MAKE TEA AND ALSO CLEAN THE POOL.
WELL, MINE TAKES OUT THE TRASH EVERY TUESDAY MORNING AT SEVEN O'CLOCK PROMPTLY!!
REALLY?

At the moment, the operating systems you're likely to encounter on desktops are to some extent interchangeable. Linux, Macintosh OS X and Windows all implement POSIX to some degree. Software that was written to make use portable use of the operating system's services may be compiled to run on all of them.

These operating systems can all do the same jobs, and you may develop skills that apply to other operating systems on any of them. Therefor there is no quick, objective answer to the question as which one is "best". You have to forget that one, and ask a better question: "Which operating system is best *for me*?"

Well, of course that depends on who you are and what you need it for. If you're interested in tuXlabs then you probably don't have lots and lots of money, and you're probably interested in learning: learning about computers, yes, but also education in general. When answering this question for yourself, here are two things to keep in mind:

- **How does it impact the world I live in?** The software you choose can influence many aspects of your life. For example, you need to ask: "Who controls the technology?" If you teach yourself to use a product that belongs to some company, you'd better hope that they don't go out of business.

  Governments are also starting to use computers for tasks such as counting votes. During the previous two elections in America, there was great controversy about the results turned in by Diebold's vote counting machines. Diebold controls the software, and expect voters to take the trustworthiness of their software on faith. For matters such as this, democracies in the digital age must use open source software.
- **Current technical merit is only one measure.** Even if something is imperfect now, if you can get at it, you can make it better and learn in the process.

  For example, the open source program called the **GIMP** (the GNU Image Manipulation Program) goes head-to-head with Adobe **Photoshop**. At the moment, Photoshop still comes out pretty far ahead, but it has been under continuous development for a decade or more. Does this mean that aspiring designers should turn their back on the GIMP? If they do, they lose the things that the GIMP already gives them which Photoshop doesn't, such as the ability to write extensions in a variety of high-level programming languages, and to build on the contributions of the entire community of users.

1.3.2. What is a distribution?

In the following text, you'll read about a number of different *distributions*, e.g. Debian, K12LTSP and RedHat Fedora. This is a term that emerged from the open source way of gathering and organising software for sharing: a coherent, well-maintained, up-to-date collection of software (especially on the scale of an operating system) is called a *distribution*.

As you saw above, Linux is the source code of an operating system being written by

hundreds of volunteers who pursue their own goals and interests. It uses tools from other projects, such as the GNU project, and is used on a very wide range of machines, for widely varying goals. From this, you may imagine that it is a hugely complex system, and you'd be right. Assembling just the subset of code that is relevant to you, and compiling it to run optimally on the hardware that you happen to have, is a major undertaking requiring deep skill and experience. Keeping your system up to date with changes in disparate parts of the whole is a major undertaking in itself.

In order to have a manageable system, well-understood and reasonably easy to customise and keep up to date, groups of users began to band together to collect just the right combinations of software, and to coordinate this job. Some people built their business around the process of gathering, labelling, testing, documenting and marketing free software, for example RedHat and SuSE. Some communities assemble distributions that conform to their ideals; Debian, for example, operates in accordance with a Social Contract [http://www.debian.org/social_contract] which maintains the same principles of good husbandry and mutual cooperation that prompted Richard Stallman to found the Free Software Foundation.

tuXlabs and Wizzy actually make use of no less than *four* related distributions.

- The classroom server runs a modified K12LTSP distribution, and
- the Wizzy server is based on Whitebox Linux.
- K12LTSP, in turn, is based on RedHat Fedora;
- and Whitebox Linux is based on RedHat Enterprise.

In part, the reasons for the variation are historical: the Wizzy solution was developed independently of the tuXlab project.

1.3.3. Where it all began

1.3.3.1. Academic computer science background

The idea of free software, with source code that could be shared by everyone, started in academia. In America, it was Richard Stallman at MIT, the Massachusetts Institute of Technology, who started the ball rolling, but the person who accidentally turned it into an avalanche was Linus Torvalds, a Finnish university student.

Richard Stallman

At MIT, Stallman had been part of a community of programmers that came to an end when their work was commercialised and access to the source code was restricted. To Stallman, this felt like having the air he needed to breathe cut off. Because Stallman believed strongly that programmers should be able to help one another by sharing their source code, he set out to write a complete free operating system. He began systematically, though, first setting out to write all the supporting tools that an operating system requires to work. This is a mammoth task, and in fact they're still busy refining better and better tools.

Linus Torvalds, in Helsinki, wasn't burdened with any such a sense of responsibility or thoroughness. He just wanted to make the most of the PC he had at home, which had an Intel 80386 CPU. The 80386 contained a memory management unit, which was big news at the time. The operating systems he had at his disposal didn't take advantage of this and he dearly wanted to use it, and so he started to write his *own* operating system. To do this, he used many of the tools created by the GNU project. After months and months of steady work, he released a very early version of the kernel that became known as Linux to the internet. To his surprise, other people started sending him fixes and improvements for his kernel, and eventually he found himself managing and coordinating a global community of programmers. All of them were using and improving Linux, and building on each others' work.

Linus Torvalds

In a sense, the free software community that Stallman had known at MIT had risen anew, on a global scale. Their rallying point was a system made up of the Linux kernel, and the GNU development environment.

1.3.3.2. Open Source Software

The first one to think of a name for the kind of software that could be shared without

restrictions on how you could use it was Richard Stallman. He called it Free Software, because he wanted to emphasise the freedoms that he valued highly enough to dedicate his life to writing a free operating system.

The Free Software Foundation [http://www.fsf.org] supports the freedoms of speech, press, and association on the internet, the right to use encryption software for private communication, and the right to write software unimpeded by private monopolies. Stallman formulated a license, the GNU Public License [http://www.gnu.org/copyleft/gpl.html], which uses the mechanism of copyright to protect these freedoms, and to add the responsibility of passing them on to other users of the software.

When Linux started to be noticed by business, and when it began to be marketed as a serious IT platform, this emphasis on freedom made some people uncomfortable. The argument was that people don't run their businesses in order to advance someone's freedom of speech --- they run their business to make money!

Instead of emphasising the *freedom* aspects so dear to Stallman's heart, more emphasis was placed on the fact that everyone had access to the source code of Linux. The programmer and writer Eric Raymond wrote some influential papers [http://www.catb.org/~esr/writings/cathedral-bazaar/] in which he argued that the Linux style of community development produced *better software* than the proprietary alternative. The choice for open source software could be made on a purely pragmatic basis. The critical factor, in his view, was the availability of the source code, and therefor he coined the term *Open Source* to describe this kind of software.

The contrast may be summed up in the sentence: "Open source is a development methodology; free software is a social movement." FSF [http://www.fsf.org/philosophy/free-software-for-freedom.html]

Neither one of these approaches encompass the whole truth, they simply emphasise different aspects of a rich sphere of human endeavour. The pragmatic approach taken by tuXlabs in the selection of software for inclusion leans more toward the open source side of the debate, but the Shuttleworth Foundation's goal of "social innovation" is in line with the FSF philosophy.

Various other streams exists. The BSD license of the FreeBSD [http://www.freebsd.org] project, for example, does not specify the responsibilities of the GPL, and allows code under the BSD license to be incorporated into proprietary software. For many years, for example, the Windows TCP/IP stack was based on BSD code right up to the period of **Windows 2000**--- perhaps it still is?

1.3.3.3. Open Source Culture

As the open source culture matured, the principles of mutual education, self-sufficiency and sharing were applied to many things besides software. One of the first projects to

bring the wider world of culture into the open source community was Project Gutenberg [http://www.gutenberg.org/], a project to make available as many as possible public domain and freely redistributable texts at no cost. Due to recent changes in American legislation (which enables copyright holders to keep works out of the public domain forever), this essentially means works created before 1923. At the moment, there are more than 13,000 books available for download [http://www.gutenberg.org/catalog/].

Another open source project is the Wikipedia [http://www.wikipedia.org/], an online encyclopedia read and edited entirely by volunteers. The English edition, started in 2001, already has almost half a million articles. If your tuXlab includes a Wizzy internet server, the entire Wikipedia can be made available on the tuXlab network.

Larry Lessig [http://www.lessig.org/], a law professor at Stanford, noticed the need to apply the open source principles of collaboration and sharing in other domains besides software, and set about crafting a flexible set of licenses that could be used to bring music, books, movies and educational material into the open source world. Since his project has as a goal the re-establishment of a *commons*, a area for the use of the community as a whole, to replace the endangered public domain, these licenses are called the Creative Commons [http://creativecommons.org/] licenses.

1.3.3.3.1. Why do people do this?

There are many reasons, but I'll mention only one. Far more expensive than the recording of a song, or the writing of a book, is the task of promoting, printing and distributing it. Unless this task can be handed over to everyone who reads or listens, only large media companies are able to afford this cost. A creative commons license allows authors to publish work that the media companies are not interested in.

1.3.3.4. Open Source in education

Free Software can be a valuable resource in education, and can also promote the values of the GNU project, namely freedom and cooperation, in schools.

There are general reasons why all computer users should insist on free software. [3] It gives users the freedom to control their own computers --- with proprietary software, the computer does what the software owner wants it to do, not what you want it to do. Free software also gives users the freedom to cooperate with each other, to lead an upright life. These reasons apply to schools as they do to everyone.

But there are special reasons that apply to schools.

- First, free software saves money. Even in the richest countries, schools are short of money. Free software gives schools, like other users, the freedom to copy and redistribute the software, so the school system can make copies for all the computers

[3] Source for the following paragraphs: http://www.fsf.org/philosophy/schools.html [http://www.fsf.org/philosophy/schools.html]

they have. This is essential to help close the digital divide.

- Secondly, schools should help learners to build a strong society after they leave school. They should promote the use of free software just as they promote recycling and protecting your environment. If schools teach learners about free software, then they will use free software after they leave school. This will help communities to be more self-reliant, and will make them less dependent on big corporations who repatriate their profits to other countries.
- Thirdly, free software permits learners to find out how software really works. They can go and look at the source code to find out how the operations they use were implemented, and experiment by changing it.

Proprietary software rejects their thirst for knowledge: it says, "The knowledge you want is a secret --- learning is forbidden!" Free software encourages everyone to learn. The free software community rejects the "priesthood of technology", which keeps the general public in ignorance of how technology works; we encourage students of any age and situation to read the source code and learn as much as they want to know. Schools that use free software will enable gifted programming students to advance.

1.3.3.5. Resources

Here are pointers to a few of the organisations and projects that work to further the use of free software in education:

- The Debian Jr. Project [http://www.nl.debian.org/devel/debian-jr/] is a custom Debian distribution. It aims to make Debian an OS that children will want to use, by studying the needs expressed by the children themselves. Their initial focus is on children up to age 8. Once this goal has been accomplished, their next target age range is 7 to 12.
- Tux4Kids [http://www.tux4kids.com/] provides some great software packages for Debian Jr.
- DebianEdu [http://wiki.debian.net/?DebianEdu] is about improving Debian to make it the best distribution for educational use.
- Because they believe that free and equal access to information technology is important in modern society, the Organisation for Free Software in Education and Teaching [http://www.ofset.org/] is actively promoting and developing free software for schools.
- SchoolForge [http://www.schoolforge.net/] is an umbrella organisation or a communication channel for different groupings with the mutual goal to advance open resources at school.
- The Open Source in Education Project [http://sourceforge.net/projects/osie] (OSiE) supports and advocates the use of GNU/Linux systems in the UK.

  This is just one example of a local project. Others exist in Germany [http://fsub.schule.de/], Italy [http://www.linuxdidattica.org/], Latvia [http://www.laka.lv/], Argentina [http://www.gleducar.org.ar/], and many other

countries across the globe.

2

Infrastructure Model

This module deals with the physical infrastructure that makes up the premises of a TuXlab. The first section deals with security issues regarding windows, ceilings, and doors. Along with the discussion on lab security are recommendations for alarm systems and the type of alarm response company that should be used. The second section deals with the infrastructure requirements within the TuXlab room. This includes details about workstation desktops, the server cabinet, electrical requirements, and the network infrastructure.

2.1. Introduction

Whenever people think of a computer lab, it is usually the computers that they think of first. They do not think of the room itself, or the construction of the chairs, the desks, the electrical cabling, and the security measures that need to be in place for the computer lab to work properly. A tuXlab cannot just go anywhere. It requires a specific place with specific things in place in order to function.

The *where* and *in what* questions need to be asked when planning a tuXlab. The infrastructure surrounding the tuXlab is just as important as the computer hardware and software used. In this part of the tuXlab Cookbook we provide an overview of the environment that the tuXlab needs to operate in; the type of room, the security features, the cabling requirements, and the little details of what and where that make a computer lab function.

2.2. Security Infrastructure

The situation in South Africa may differ from other parts of the world, but these basic guidelines have arisen out of our needs and conditions. If you are building a tuXlab outside of South Africa you may need to use other security measures or methods of construction that allow your tuXlab to be usable and safe.

2.2.1. Window Security

The windows of the tuXlab need to be secured so that no one can break into the lab and steal the computers, printers, or other lab items. These are the typical requirements for securing the windows:

- Frame: 25mm square metal tubing, bolted to the wall.
- Vertical Bars: 16mm round – 120mm c/c.

- Centre Horizontal Bar: 30mm x 5mm flat bar.

Window Security

While the above may be used as a guide, the diameter and the finish of the bars can be further determined by the relevant school authorities and the Project Manager as is deemed necessary according to the known risk in the area.

2.2.2. Stone Guards

Galvanised metal mesh must be fitted on all outside windows.

Shows Fitted Stone Guards

The sides of the frame must be closed. Fasteners: Use tamper proof – Coach screws with turn off heads.

2.2.3. Non Concrete Ceiling Security

Wherever possible the room identified for the lab should have a concrete floor above it. This means that in schools with two levels it is preferable to select a room on the lower level. Where this is not possible for various reasons and the ceiling is of a hardboard nature, wire mesh or razor wire needs to be laid on top of the ceiling.

Ceiling Security

As with the other security measures, whatever is implemented must be determined by the relevant school authorities and the Project Manager as is deemed necessary according to the known risk in the area.

2.2.4. Door Security

Remember that steel frame gates are only as strong as the thickness of metal used. The door is the key security feature that prevents easy access to the tuXlab. In some cases an internal security steel cage with steel gate or double steel gates are advised, depending on finances and risks. Methods of securing and fitment of locks should be strategic to make breaking in as difficult as possible to deter theft.

Typical Specifications:

- Frame: 50mm x 50mm x 5mm Angle Iron bolted to the wall on 3 sides.
- Bolts are welded so that they cannot be undone, or they must be tamper proof.
- The fitting of two barrel type locks, and covered with metal plates is advised.

Internal Cage

Double Steel Gates

While the above may be used as a guide, the diameter and the finish of the bars can be further determined by the school authorities as is deemed necessary according to the known risk in the area.

2.2.5. Alarm System

A digital security alarm system with at least two sensors in the room is required. If the lab has a ceiling, with access from above (single storey building) or access from adjacent rooms (double story building), a sensor in the ceiling is advised. The alarm must be 24 hourly monitored with armed response.

Number and location of sensors should further be determined with input from security companies.

2.3. Internal Lab Infrastructure

2.3.1. Specifications for Desktops

The design and length of desktops in most cases are determined by the number of computers, learners and size of the room. It is advisable to install desktops for future expansion of computer network, depending on finances available.

Building the Desktops

New desktops should be postform, minimum 28 mm thick or similar stock, with a melamine/formica/varnished finish. They must be 900mm wide. Depending on the chair heights and the age of the learners, the top of the desktops should be 700 to 720mm from the floor.

2.3.2. Wall Brackets and Centre Isle Framework

Desktops should be mounted on 40mm x 40mm x 5mm angle iron brackets (on legs only if brackets not a option). Each bracket must be a minimum of 750mm x 650mm long. The brackets must be no less than 1000mm apart and each must be fixed with at least two heavy duty rawl bolts, one of which must be as high up as possible.

Where appropriate, provision must be made for ducting for electrical wiring and top for network cabling to run underneath the work surfaces, with holes in the desktops for the cabling to the computers and peripherals. The design of the framework will differ according to the makeup of the walls to which the brackets can (or cannot) be attached. The same design could apply to a centre isle when fitted. Where possible the length of the desk will allow for each computer to be spaced at 1.2m.

Modifications to bracket: 30mm hole to be drilled in angle iron so that the Electrician can install a conduit pipe along the wall. Hole should be 90mm from the top of the bracket. Fasteners: Brackets must have 3 x 10mm holes for fastening to the wall. Use coach screws 10mm x 75mm, with Fischer Plug. Brackets to be finished with black matt corrosion resistant paint. The length of the desktops is calculated according to two learners per PC giving a space of 1200mm per PC. Please see attached diagram of computer room layout, as this calculation has been made.

2.3.3. Specification for Server Cabinet

A well ventilated lockup cabinet will be built in conjunction with the desktops in labs that does not have an additional room for the server. Interior Requirements: 900mm of the cabinet must be empty, as it will house a Server, the Internet gateway with a modem. The cabinet must have no backing to allow for ventilation of the equipment and wiring. The cabinet will be set away from the wall to allow for ventilation.

A wall mounted steel mesh cabinet can also be installed. Dimensions for steel mesh cage:L x D x H - 1200mm x 750mm x 720mm. Height: 720mm Includes a 32mm x

900mm Postform Top. Doors: 2 doors 450mm wide is required on the 900mm section. Suitable locks must be fitted so that the doors cannot be easily opened.

2.3.4. Electrical Requirements

These electrical requirements are very important for the continued operation of the tuXlab. If the wiring is installed incorrectly, or is not installed in compliance with the local building codes, then the tuXlab is in danger of becoming unsafe and hazardous the all the users.

2.3.4.1. Specification of Electrical Wall Points

All electrical work must be done by a qualified technician as well as an electrical certificate of compliance should be issued to the school on completion of the work.

There must be enough (240volt) 15Amp 3-point plugs to accommodate each computer on a separate plug as well as enough for peripherals. A good rule of thumb is to have a double plug for each computer point. For safety reasons electrical wiring must be in conduit piping below the work surfaces, but Surfix-type cable and wall mounted sockets are also acceptable and should be installed below desktops.

To avoid power spikes and dips, the computer lab **must** be on its own electrical circuit and must be broken into segments each adequate to accommodate the computers and peripherals on that segment. Air conditioners in the computer lab **must not** share the circuit with the computers.

Electrical wires to an island worktop in the middle of the room must preferably be under the floor or must have suitable ducting that will ensure the safety of pupils who have to walk over these cables.

2.3.4.2. Sub-Distribution Electrical Board

The sub distribution board could be fitted with earth leakage protection, if required. A maximum of five wall plug sockets shall be connected to one circuit. Each circuit shall be protected by a 20 amp circuit breaker.

2.3.5. Network Infrastructure

As seen in the introduction to the tuXlab Cookbook, the network is the backbone of the tuXlab. There are some specific infrastructure requirements that come directly from the installation of the network.

2.3.5.1. Switch Cabinet

Where required a 4U Cabinet will be installed to house a minimum of 2 x 24 port switches unit. The positioning should decided based on lab layout and lengths of cables. If installed in the lab, the switch cabinet should be installed 1metre above desktops or below.

2.3.5.2. Network Cable Trunking

Network cabling must run in 40mm x 40mm square trunking above the desktops. Ensure10mm holes are made in trunking for networking cables ends, allow 500mm to the individual computers. Cabling to an island desktop must preferably be under the floor or must have suitable ducting that will ensure the safety of pupils who have to walk over these cables.

3

Hardware Module

This module deals with the hardware requirements, setup and installation, network creation, and server configuration for the TuXlab lab. The first section deals with the concepts of "thin-client" computing and use of the Linux Terminal Server Project (LTSP). The second section deals with the requirements of the thin client and server computers, including information regarding the refurbishment of older computers to use as thin-clients. The third section presents a discussion of networking, from the question of "Why network" to descriptions of the equipment needed and how to use them. The fourth section deals with the various configuration processes that need to be followed when setting up a TuXlab. The fifth and final section then deals with an evaluation of the hardware usage of the TuXlab, mentioning benefits and drawbacks.

Once you have read through this module you will have been introduced to the physical machines that form the core of a TuXlab.

3.1. Hardware Background

It is not hard to imagine looking into a computer lab and seeing the computers. In a tuXlab there are two subsections of hardware, thin-client workstations and the tuXlab server. Before we discuss the setup and configuration of the server and the workstations let's take some time to discuss the theory behind each of these hardware subsections.

3.1.1. Thin-client computing

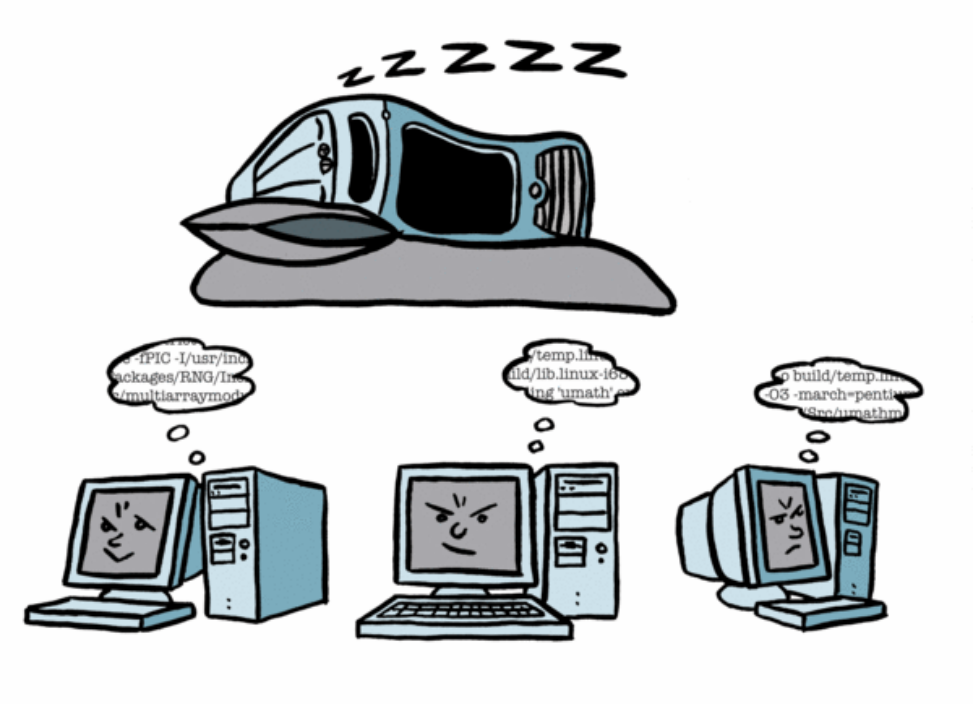
A tuXlab computer laboratory consists of a classroom full of *thin client* workstations communicating with a classroom server. The specific implementation of thin client computing used in tuXlabs is discussed in Section 3.1.2.1, “Linux Terminal Server Project” [3–3].

3.1.1.1. What is Thin Client Computing?

Thin client and *fat client* (also called "thick" or "rich" client) are mostly marketing terms for different configurations of computers. A thin client asks a central server to do most of its processing, and keeps as little hardware and software as possible on the workstation side. Ideally, the user of a thin client should have only a screen, keyboard, mouse and enough computing power to handle display and network communications --- you don't even need (or want) a hard drive. The less you have, the less there is to go wrong.

A fat client does as much processing as possible by itself, and only passes data required

for communication and storage on to the server. A standalone PC is the typical fat client with which everyone is familiar.



Fat client

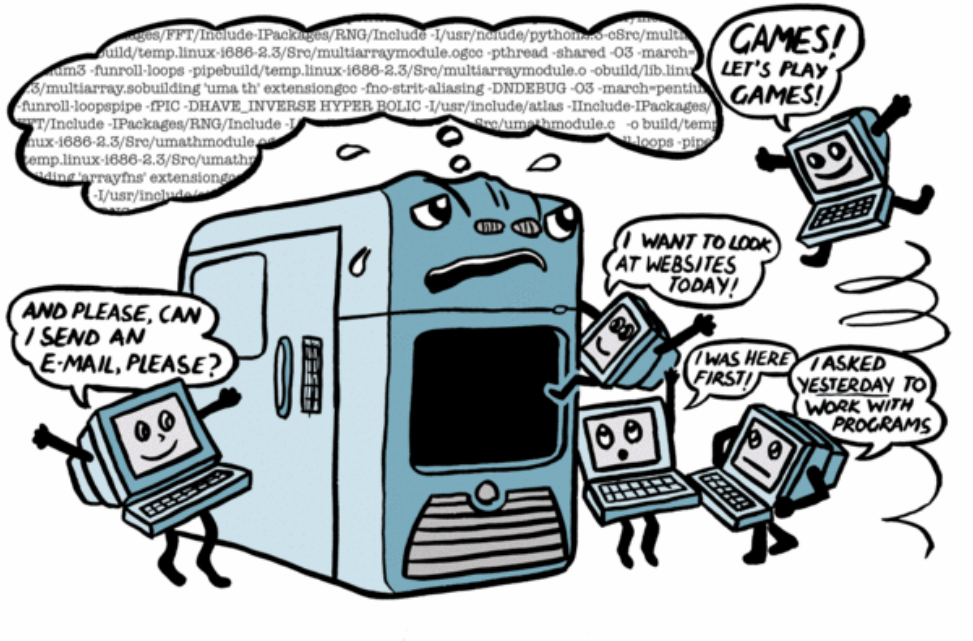
A thin client may be a software program executing on a generic PC, or it may be a hardware device, called a terminal, designed to provide only user input and display functions. Because old PCs (whether retired, written off, obsolete or just out of fashion) are easier to find than specialised thin client hardware, tuXlabs uses them as thin clients, with the appropriate software. Because they don't do much work themselves, the hardware requirements for these "old" PCs are very basic. Since every client in a thin client network asks a central server to do its work, all the individual workstations look the same: they all share the same server, and they all behave exactly like the server would if you were using it directly.

In addition, although everyone who uses the lab can have their own computing environment stored on the server, with their own files, desktop, and so on, the individual workstations can't get viruses or be misconfigured by curious learners - there simply isn't anything to configure! The thin client doesn't have enough brains to get confused.

This means that the lab computers are trivial to keep up and maintenance is restricted to the server in the back room.

Thin clients are cheaper and require less administration than fat clients. On the other hand, they tend to require far greater network bandwidth, as display data will probably need to be passed to the thin clients. They can't do a single thing on their own - for each and every action, they need to talk to the server. This means that a server for a room full of thin clients must be much more capable than a server used by fat clients.

One of the advantages that this configuration entails is that all the software resides on the server, and so you only have to upgrade it once. In a fat client configuration, every workstation has its own copies of the software, and so any upgrade needs to be rolled out to every workstation.



3.1.2. The Server

We have already spoken about the use of a server as we discussed the thin-client verses fat-client scenario in the previous section. In the tuXlab, the server does all the computing work and so needs to be a powerful computer, capable of serving all the needs of the thin clients. The Linux configuration that provides the interface that allows all this to happen is the *Linux Terminal Server Project*.

3.1.2.1. Linux Terminal Server Project

The Linux Terminal Server Project (LTSP) is a configuration of Linux that allows you to connect lots of low-powered thin client terminals to a Linux server. The LTSP provides a simple way to utilise low cost workstations as either graphical or character-based terminals

on a GNU/Linux server.

Once installed within your Linux distribution, LTSP lets you boot diskless workstations from an application server.

3.2. The Computers of a tuXlab

When speaking or reading about computer hardware there are specific words and abreviations used that are not part of common everyday usage. If you are unfamiliar with these terms please don't stress. They are relatively easy to understand, and once you know what they are you will wonder why you didn't already know them. As you are already learning all about computers as you setup and install a tuXlab, you will quickly learn these terms, and probably many others too.

3.2.1. Hardware Requirements

Now that we understand what the computer hardware will be doing, it's time to discuss the various requirements of the server and thin clients.

3.2.1.1. Server Hardware Requirements

In a thin client set up, the server needs to be powerful enough to run all the applications in the lab.

As a general rule of thumb, the server should have (at a bare minimum) 100MB RAM per thin client, and an additional 500MB for web and other services that might be running on the server. Disk space depends on the needs of the school, although the amount of disk space isn't too important. Serial ATA disks offer better performance than IDE disks. It's recommended that you add at use at least two 80GB 7200rpm SATA disks, although 200GB or more is recommended. We recommend at least a CD writer drive for backup purposes, but a DVD writer is recommended.

Minimum server specification for a 20 seat lab:

- Intel Xeon Server Board with Intel 3GHz CPU (or similar hardware)
- 2.5 GB RAM
- 2x SATA 80GB Hard disks
- CD Writer

Recommended server specification for a 20 seat lab:

- Intel Xeon Server Board with Dual Xeon CPU
- 4GB RAM

- 2x 200GB SATA or SCSI Hard Disks
- DVD Writer/RAM Drive

Memory - The server should have 2GB RAM or more (512MB for the base system, and 100MB for each additional client). As long as you're using it all up, more RAM means more speed (it doesn't help to have RAM that you don't use). Too little RAM will bring your server to a crawl as it starts swapping memory to the hard drive. If you run out of memory, performance will be unacceptable.

Hard drive - SCSI and Serial ATA disks are faster than IDE. We've seen LTSP servers slow to a crawl when more than 10 clients are running from IDE drives. SCSI drives are better equipped to handle the multiple read/write requests. SATA drives offer greater speed than IDE and are becoming more and more available.

Network - Your server will have at least one Ethernet card to create a private network (`192.168.0.x`). This card connects to a switch for terminals. If there is a school network to which you need to connect, or if the school has a internet connection via the server, it will have a second Ethernet card, which will get an IP address on the second network.

3.2.1.2. Thin client hardware requirements

You can use computers that are specifically designed to be thin clients as workstations, or you could use standard computer hardware as thin clients.

Minimum thin client requirements:

- CPU: 266mHz IA32 CPU
- RAM: 32 MB
- Bootable network card (PXE or Etherboot) or Etherboot floppy
- 2MB Display card with supported chipset
- No hard disk required

Recommended thin client specifications:

- CPU: 400mHz IA32 CPU
- RAM: 64 MB
- Bootable network card (PXE or Etherboot)
- 4MB Display card
- No hard disk required

Memory - Client workstations should have at least 32MB of RAM. Clients aren't that

dependent on swap space for extra memory capacity, since memory usage on them is reasonably constant because they don't execute applications: they only display them.

Hard drive - Client workstations have no need to write data to a local hard drive and so should not have any installed on the thin client.

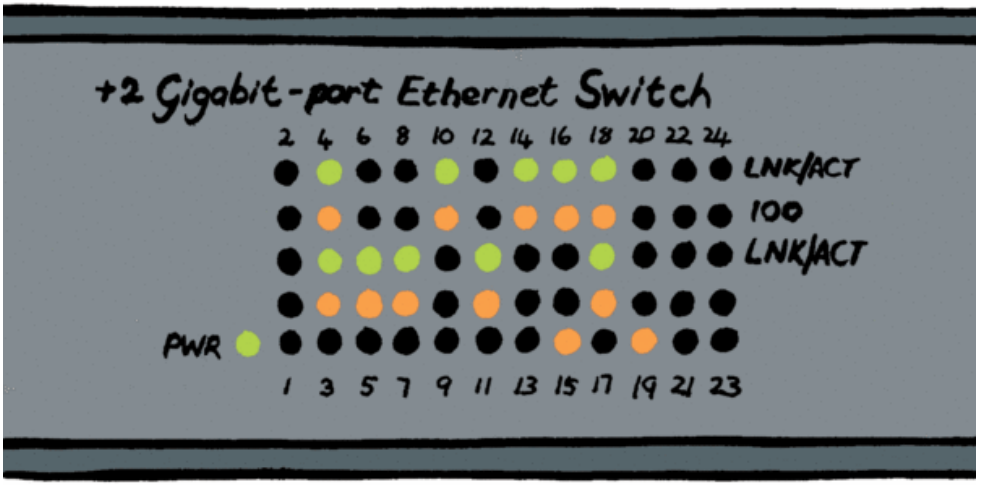
Network - Each client workstation should have one network card with a boot ROM to enable booting from the network.

3.2.1.3. Network Switch

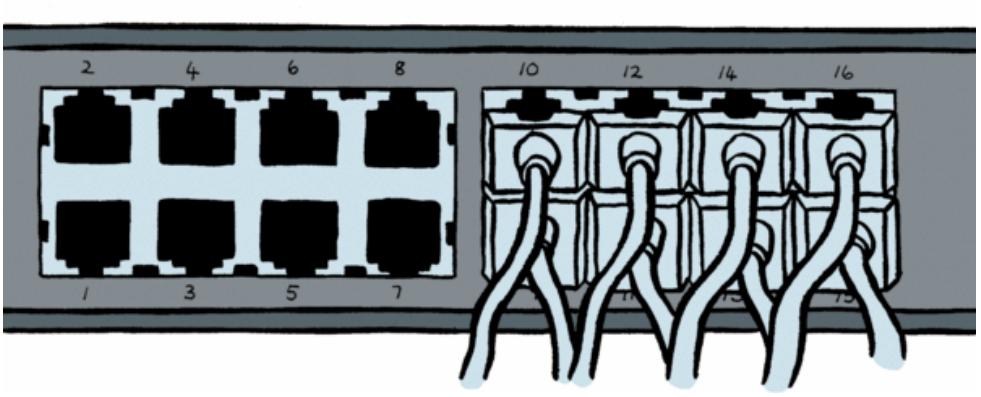
The hub of the network needs to be a switching hub capable of 100mbps for workstations, and 1000mpbs (1 gbps) for the server or uplink connections. 1000mbps would be ideal for every connection, this would however require equivalent network cards in all workstations and may present a big cost issue. 100mbps is sufficient though, and newer desktop sharing technologies are becoming more efficient with regards to network traffic.

An Ethernet hub is not acceptable (see Section 3.3.2.1, “Switches / Hubs” [3–16]), it's too slow for network boot and NFS. Having a fast Ethernet switch will make your life better and more colourful.

The number of ports on the switch must be enough for your clients and server. If you have more than 24 NICs, 2 or more 12-port/16-port fast Ethernet switches are recommended. The client ports should be 100 megabit, and there should be a gigabit port for the server. Since all clients get their display from the server, a slow link to the server would be a bottleneck for the whole lab.



The front panel of the switch shows a light for every network cable that's been plugged in. The light shows whether the link is up, whether there is traffic and what its speed is.



The cables from the workstations all terminate at the switch. If you don't label them, they get very hard to tell apart!

Please refer to the Section 3.3, “Networking” [3–14] chapter for more requirements regarding the network.

3.2.2. Refurbishment of thin clients / workstation computers

If you have computers that have been donated to your school for use in your tuXlab, you will need to refurbish them before installing them onto the network.

Refurbishment of a workstation entails:

- Cleaning the equipment.
- Removing any unnecessary hardware (such as serial port cards, hard disks, etc).
- Ensuring that the workstations can boot from an LTSP network.
- Serialising all hardware.
- Ensuring the privacy of hardware donors.

The first thing to do is to set up a static free environment in which to work on the computers. This can be as simple as setting up a static free surface on a desk or wearing an antistatic wrist band connected to the computer you are working on. If neither of these two options are available to you, then work on a clean metal desktop and regularly discharge any static electricity build up by touching the surface of the desk with your fingers before touching any computer component.

Now you need to unscrew the case of the computer and clean out any dust build up by spraying compressed air over the motherboard and components. Remember to aim the air spray in such a manner that the dust is blown out of the case, away from your face. Once the accumulation of dust has been cleaned, you need to asses what components are in the computer. Identify the graphics adapter, the amount of memory, whether the computer has

a network adapter, a sound card, a modem, or any other peripheral card within it.

Remove all of the hardware that a thin client will not be using (i.e. harddrives, modems, etc.). If you are removing a harddrive, either return it to the donor or format the harddrive before removing it so that the data originally contained on the disk is erased. Now either write serial numbers on the computer components or make stickers with serial numbers and stick them on the motherboard, cards, and computer case. Make a record of all the serial numbers in a book or in the school's inventory management system.

The final step is to put the computer case back on and attach the cables. Switch it on to check that the remaining hardware is fully functioning (as there is no harddrive you will not be able to boot until you have configured the workstation to be LTSP bootable, what we'll discuss next...).

3.2.2.1. Making a workstation LTSP bootable

There are several methods that one can use to make a computer boot to an LTSP network. This includes booting from an existing storage medium, such as CD-ROM, floppy or hard disk using an Etherboot image. This however, is not the recommended method, since CD-ROM discs and floppy disks can be removed from the computer, and old hard disks tend to have a high failure rate. Most new computers are sold with on-board network cards, which typically have PXE (Pre-Execution Environment) installed, which will work fine with LTSP. Although older versions of PXE may work fine, it is recommended that you purchase a card with PXE version 2.0 or later.

If you are using an older computer, which is not PXE capable, you might want to consider buying a network card with PXE enabled (contact your local hardware vendor), or, you could purchase a network card with an Etherboot ROM chip installed, which you can still purchase from netday.org.za [http://www.netday.org.za].

If you're on a budget, and can't afford to buy PXE or Etherboot cards, you could create a boot image for existing media using software and boot images from rom-o-matic.net [http://rom-o-matic.net]. Instructions are available on the rom-o-matic website.

3.2.2.2. Things To Look Out For

Monitors	The display size should be at least 15", and the monitor must be capable of SVGA video modes. It should also be compatible with the video card of the client.
Uniform equipment	If you use the same equipment throughout your tuXlab, it becomes easier to buy in bulk and to swap out components. It also makes it easier for tuXlabs to assist each other with skills and equipment. Heterogeneous boxes, in comparison, are harder to keep running, and more likely to be "throw-away" - not worth

trying to resurrect. As long as they don't cost you anything, this is worth it, but you have to guard against them becoming a drain on your time and resources.

3.2.2.3. Sources for Second Hand Equipment

Locally, FreeCom supplies tested refurbished computers. Because of the high volume of hardware required by the large number of tuXlabs installed, the Shuttleworth Foundation has procured the client workstations from international distributors such as Computer Aid. Other workstations have been donated by private industry.

3.2.3. Connecting to the Internet

Practically any Internet connection is compatible with a tuXlab. This includes:

- Dial-up with modem and phone line
- GPRS with a GSM phone
- ISDN with a terminal adapter and digital line
- ADSL with an ADSL modem and ADSL line
- Any existing TCP/IP gateway

A gateway/firewall machine is typically placed between the Internet and the tuXlab server. This increases network security and takes some load off the main server.

If you are using a connection which is only temporary, such as Dial-up or GPRS, then it's suggested that you use a system such as the Wizzy server. Wizzy allows you to set the firewall machine to dial-up at certain times of the day (typically in off-peak time for costs savings), and will then download all appropriate content, to be used for the next day. You may also connect directly any time of the day, and data will still be stored for off-line use.

For more information on the Wizzy Digital Courier, see: www.wizzy.org.za [http://www.wizzy.org.za].

If you are using a permanent Internet connection, such as ADSL, cable, or satellite, then the easiest method is using a Linux distribution called smoothwall. Smoothwall gives you a web interface which allows you to adjust all aspects of your firewall machine inside your web browser.

For more information on the Smoothwall distribution, see: www.smoothwall.org [http://www.smoothwall.org].

Alternatively, the Ubuntu Server distribution works well, although it will require more skill to set up. For more information, see http://www.ubuntuserver.com (scripts for this

will be available on the tuXlab add-on CD).

Connecting your tuXlab server to the gateway/firewall box, or even directly to the Internet, will automatically allow Internet access to all the thin clients connected to it.

If you have any non-LTSP machines on your network, you need to either connect the gateway machine directly to the switch, or set up routing on the tuXlab machine. Software is available on the tuXlab DVD that will assist with this.

3.2.4. How the lab works

In this section we take you through a step by step list of what is happening during the startup process of a workstation in a tuXlab.

1. **"Power On Self Test" (POST).** When you turn on the workstation, it will go through its Power On Self Test (POST.
2. **Find the boot ROM.** During the self test, the BIOS will search for expansion ROMs. The network card contains an Etherboot boot-ROM, which is an expansion ROM. The BIOS will detect the ROM on the network card (it doesn't know about the network card, it only notices the ROM).
3. **Boot.** Once the POST is complete, execution will jump into the Etherboot code.
4. **Find the network card.** The Etherboot code will scan for a network card. Once it detects the card, the card will be initialised.
5. **DHCP request.** The Etherboot code will then broadcast a DHCP request to the local network. The request will include the MAC address of the network card.
6. **DHCP request received.** The **dhcpd** daemon on the server will see the broadcast request from the workstation, and look in its configuration file for an entry that matches the MAC address of that workstation.
7. **DHCP request reply.** The **dhcpd** daemon will construct a reply packet, containing several pieces of information. This packet will be sent back to the workstation. The reply information includes:
   - an IP address for the workstation,
   - the netmask setting for the local network,
   - the pathname of the kernel to download (this is a filesystem path on the server),
   - the pathname of the root filesystem to mount as the root of the client filesystem,
   - optional parameters to be passed to the kernel, via the kernel command line.
8. **Boot ROM configures TCP/IP interface.** The Etherboot code will receive the reply from the server, and it will configure the TCP/IP interface in the network card with the parameters that were supplied. Once this is done, the client computer has an IP address on the network.
9. **Download the kernel using TFTP.** Using TFTP (Trivial File Transfer Protocol), the

Etherboot code will contact the server and begin downloading the kernel.

10. **Kernel downloaded.** Once the kernel has been completely downloaded to the workstation, the Etherboot code will place the kernel into the correct location in memory.

11. **Control passes to kernel.** Control is then passed to the kernel. The kernel will initialise the entire system and all of the peripherals that it recognises.

12. **Mount temporary boot filesystem as RAM disk.** This is where the fun really begins. Tacked onto the end of the kernel is a filesystem image. This is loaded into memory as a RAM disk, and temporarily mounted as the root filesystem. A kernel command line argument of `root=/dev/ram0` tells the kernel to mount the image as the root directory.

13. **Kernel boot sequence calls `linuxrc` shell script (before the normal boot sequence starts).** Normally, when the kernel is finished booting, it will launch a program called **init**. But, in this case, we've instructed the kernel to load a shell script instead. We do this by passing `init=/linuxrc` on the kernel command line.

14. **Identify kernel module for network card.** The **linuxrc** script begins by scanning the PCI bus, looking for a network card. For each PCI device it finds, it does a lookup in the **`/etc/niclist`** file, to see if it finds a match. Once a match is found, the name of the NIC driver module is returned, and that kernel module is loaded. For ISA cards, the driver module MUST be specified on the kernel command line, along with any IRQ or address parameters that are required.

15. **Load kernel module for network card.** Once the network card has been identified, the **linuxrc** script will load the kernel module that supports that card.

16. **linuxrc makes DHCP query. dhclient** will then be run, to make *another* query from the DHCP server. We need to do this separate user-space query. We cannot depend on the query that comes from Etherboot, because it gets swallowed up when the kernel uses it. The kernel will also ignore any NFS server that might have been specified in the root-path. This is important if you want the NFS server to be different from the TFTP server.

17. **Configure `eth0`.** When **dhclient** gets a reply from the server, it will run **`/etc/dhclient-script`**, which will take the information retrieved, and configure the `eth0` interface.

18. **Mount the root filesystem from the server via NFS.** Up to this point, the root filesystem has been a RAM disk. Now, the **linuxrc** script will mount a new root filesystem via NFS. The directory that is exported from the server is typically **`/opt/ltsp/i386`**. The new filesystem can't just be mounted as **`/`** immediately. It must first be grafted into the local filesystem by mounting it, typically on the path **`/mnt`**. Then, the client can do a **pivot_root**. **pivot_root** will swap the current root filesystem for a new filesystem. When it completes, the NFS filesystem will be mounted on **`/`**, and the old root filesystem will be mounted on **`/oldroot`**.

19. **Hand off to the normal init program (non-LTSP boot sequence continues).** Once the mounting and pivoting of the new root filesystem is complete, we are done with the

linuxrc shell script and we need to invoke the real **init** program.

Note: from this point, all file paths (except those starting with **`/oldroot`**, of course) refer to files that are served from the server via NFS.

20. **init processes `/etc/inittab`. init** will read the **`/etc/inittab`** file and begin setting up the workstation environment.
21. **LTSP starts in runlevel 2. init** works in terms of *runlevels*. A runlevel has a number, and specifies a set of services that should be available while the system is running in that runlevel. The LTSP workstation starts in runlevel 2. That is set by the `initdefault` line in the **`inittab`** file.
22. **Run rc.local.** One of the first items in the **`inittab`** file is the **rc.local** command that will be run while the workstation is in the `sysinit` state.
    a. **rc.local creates a RAM disk for volatile data during bootup.** The **rc.local** script will create a 1MB RAM disk to contain all of the things that need to be written to or modified in any way.
    b. **RAM disk mounted as `/tmp`.** The RAM disk will be mounted as the **`/tmp`** directory. This directory exists in order to hold files that need to be written during the boot process. We don't want to write to these files on the hard disk, because then we'll change them for all other clients as well, and the changes pertain only to our client while it's booting.

       Any files that need to be written will actually exist in the **`/tmp`** directory. On the hard disk of the server, there are only symbolic links pointing to these files.
    c. **The `/proc` filesystem is mounted.** This is a virtual filesystem that exposes information about all the currently running processes as a hierarchy of textfiles that may be read exactly as any other file on disk.
    d. **Network swap enabled.** If the workstation is configured to swap over NFS, the **`/var/opt/ltsp/swapfiles`** directory will be mounted as **`/tmp/swapfiles`**. Then, if there isn't a swap file for this workstation yet, it will be created automatically. The size of the swap file is configured in the **`lts.conf`** file.

       The swap file will then be enabled, using the **swapon** command.
    e. **Configure loopback interface.** The loopback network interface is configured. This is the networking interface that has `127.0.0.1` as its IP address.
    f. **Mount `/home`.** If LOCAL_APPS is enabled (see below), then the **`/home`** directory will be mounted, so that running applications can access the users' home directories.
    g. **Create `/tmp`.** Several directories are created in the **`/tmp`** filesystem for holding some of the transient files that are needed while the system is running. Directories

such as:

i. /tmp/compiled

ii. /tmp/var

iii. /tmp/var/run

iv. /tmp/var/log

v. /tmp/var/lock

vi. /tmp/var/lock/subsys

will all be created.

h. **Configure X Windows.** The X Windows system will now be configured. In the **`lts.conf`** file, there is a parameter called XSERVER. If this parameter is missing, or set to "auto", then an automatic detection will be attempted. If the workstation has a PCI video card, then we will get the PCI Vendor and Device id, and do a lookup in the **`/etc/vidlist`** file.

   If the card is supported by XFree86 31.X, the **pci_scan** routine will return the name of the driver module. If it is only supported by XFree86 32.3.6, **pci_scan** will return the name of the X server to use. The **rc.local** script can tell the difference because the older 33.3.6 server names start with `XF86_`.

i. **Generate XF86Config.** If XFree86 34.x is used, then the **/etc/rc.setupx** script will be called to build an XF86Config file for X4. If XFree86 35.3.6 is used, then **/etc/rc.setupx3** will be called to build the XF86Config file, based on entries in the **`/etc/lts.conf`** file.

j. **rc.local resumes, `start_ws` created.** When the **rc.setupx** script is finished, it will return to **rc.local**. Then the **/tmp/start_ws** script will be created. This script is responsible for starting the X server.

k. **Configure syslogd.** The **`/tmp/syslog.conf`** file will be created. This file tells the **syslogd** daemon where to send logging information (this may be any host on the network). Any program, including the kernel, that wants to record information for the purposes of monitoring, auditing, debugging or later reference can make use of **syslogd**, which sees to it that this information is written to a file, and that the logged information is eventually cleaned up.

   The syslog host is specified in the **`lts.conf`** file. There is a symbolic link from **`/etc/syslog.conf`** to the **`/tmp/syslog.conf`** file.

l. **Start syslogd.** The **syslogd** daemon is started, using the config file created in the previous step.

23. **init resumes, changes to *default* runlevel.** Control is then passed back to **init**, which will look at the `initdefault` entry to determine which runlevel to enter. As of

`lts_core-2.37`, the value of `initdefault` is 2.

24. **What the different runlevels do.** A runlevel of 2 will cause init to run the **set_runlevel** script which will read the **`lts.conf`** file and determine what runlevel the workstation will run in.

The standard runlevels for LTSP are 3, 4 and 5.

3 This will start a shell. This is very useful for debugging the workstation.

4 This will run one or more Telnet sessions in character mode. This is great if you are just replacing old serial terminals.

5 GUI mode. This will bring up X windows, and send an XDMCP query to the server, which will bring up a login dialogue box to let you log into the server. You will need a display manager listening on the server, such as XDM, GDM or KDM.

3.3. Networking

3.3.1. Why network?

On its own, a computer can be a fascinating tool. However, when you connect many computers together using a network, worlds of possibility open up. In a network, better use is made of all the connected resources, because they can be *shared*. For example, if there is one printer, everyone can use it. It is also possible to concentrate resources where they will have the greatest benefit --- all the additional memory added to the server becomes available for running the programs of *all* the clients.

The advantages of networking only *start* with economies in hardware expenditure. Another aspect (one that is really far more exciting) is the opening up of communication channels among lab users, and, if you can reach the internet, with the world at large. Only some of the lab users will be interested in computers for their own sake. Many more users will be writing essays, asking questions, drawing pictures or practising skills using the educational programs offered in a tuXlab. With a network, they can easily share documents, discuss them, and have a record of discussions for the learners that come after them.

3.3.1.1. Printing

A tuXlab will usually have only one or two printers for the lab as a whole. Since everyone will use these, it's worthwhile to get the best printers you can afford: as long as they're on a network, everyone will benefit.

Depending on the make of printer, it may be connected to the network switch with a network cable, or it may be connected directly to a print server (which may be the classroom server) with a parallel cable.

Printing in a tuXlab will be managed using CUPS, the Common Unix Printing System. It provides a web interface (accessible at `http://printserver:631/`) where you may check the status of printers and print jobs, print test pages, and so on. (`printserver` is the hostname of the printer or the server to which the printer is connected.)

3.3.1.2. Email

Email has been called the "killer application" of the internet. It's the most ubiquitous and accessible way to communicate with people across the world.

Not all tuXlabs have email. Generally, you'll only have email if a Wizzy server is installed along with the classroom application server. The Wizzy server functions as a post office and a stand-in, or a proxy, for the world wide web.

If your tuXlab is equipped with a Wizzy server, you'll be able to send mail to each other and to other schools or mailing lists all over the world.

3.3.1.3. File sharing

Without a network, transferring files from one computer to another is a difficult and inconvenient process. You have to copy the file onto some storage medium (such as a floppy disk, flash disk or a CD) and carry it over to the other computer yourself. Floppy disks tend to break or become silently corrupted. CDs can only be written once. Even rewritable CDs are slow to use, and expensive. USB Flash disks or memory sticks are currently the best medium for taking files from one computer to another without a network, but they still have to be physically taken from one USB port to another.

It's much better to shift the job to network cables. Once laid, a cable will keep on working (unless a mouse or rat decides that the cable looks like dinner). It doesn't cost anything to transfer data over it, and it's very fast.

In the thin-client configuration of a tuXlab, the reality is even better. None of the client workstations store any data, so the need for them to have internal hard disk drives has been eliminated. The only computer in a tuXlab that must contain at least one disk drive is the classroom server. Every user of the tuXlab - in other words, every person with a username and password to login at a workstation - has some storage space on the classroom server's hard drive allocated to them, where they may store their data. They all reach their data via the network.

This means that making a copy of a file for another user comes down to making a copy elsewhere on the same disk drive. Similarly, for files that many people need to share without necessarily needing their own copies to modify, a shared folder with one file provides everyone with access to the file. In a non-networked situation, every single workstation would need their own copy of such files. This is the case for the whole operating system and application software, for example. Each workstation would need a copy of the operating system and a copy of the word processing application, email client,

internet browser, etc. With a networked environment like the tuXlab you don't need all these copies.

3.3.2. Equipment

In this section, we have a look at the different kinds of equipment that we need to set up a network.

There are many different kinds of computer networks, with different strengths and weaknesses. Some might be designed for the maximum data transfer speed, some to minimise costs, and others to make it as easy as possible to connect computers to one another. In the case of tuXlabs, we need a really fast network, because everything displayed by the client workstations needs to be sent from the server over the network. We also need a standard network that allows any kind of computer or peripheral to be added to the network easily.

In order to meet these criteria, tuXlab uses an *Ethernet* network with *category 5* network cabling (CAT-5, for short).

In an Ethernet, data packets are broadcast onto the network for all connected devices to receive. The devices themselves then examine the data packet to determine whether it was meant for them. If so, they process it; otherwise, they drop it on the floor and it vanishes.

The name "Ethernet" comes from the ancient Greek concept of "ether". According to them, this was the fluid that filled the spaces between stars. Of course there isn't any such thing, but they made it up because surely there couldn't be *nothing* between stars, could there? In an Ethernet, as far as the communicating computers are concerned, there aren't any cables either. Of course there really are cables, but you don't have to send a data packet down a specific cable to a specific computer. You just entrust it to the "ether", and all computers get the packet.

3.3.2.1. Switches / Hubs

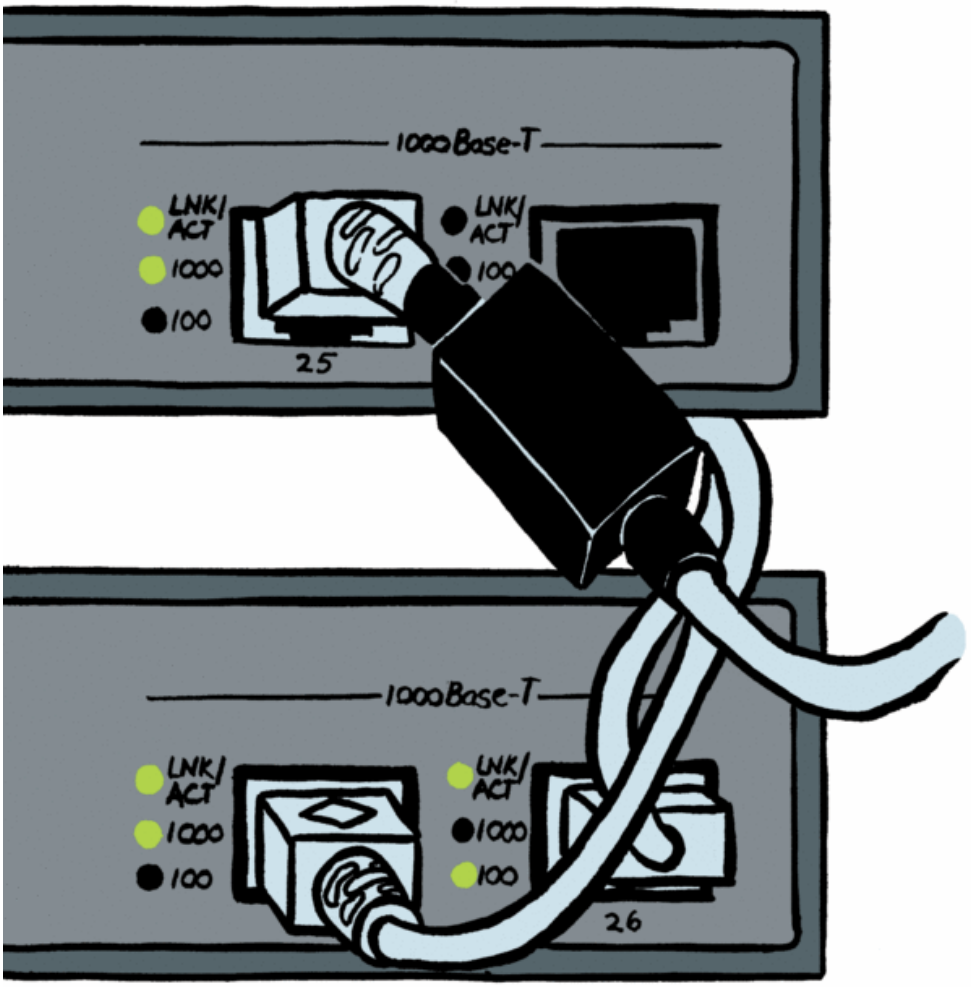
There are different ways of wiring an Ethernet local area network. One way is to simply lay coaxial cable from one computer to the next, until all the computers are connected, forming a *ring*. This is relatively simple, but the resulting network is slow, both because of the electrical properties of the coaxial cable, and because all the data has to share a single cable.

Since a tuXlab needs more speed, a *star* topology is used instead. In this configuration, a single CAT-5 cable connects each workstation to a central node. This central node acts as an interchange. In a simple network where speed isn't critical, this node can be a *hub*. This is a device with ports where you can plug in many network cables; usually 8, 16 or 24. A hub is very chatty: it simply repeats all the data coming in on one port on all the other ports. This way, the data is sure to reach the computer it's meant for. Unfortunately, it also reaches all the other computers, taking up precious network bandwidth (which is why we

don't recommend using one in a tuXlab).

Instead of a hub, you can also use a *switch*. It looks just like a hub, but it's cleverer about routing the traffic that moves across it. In short, it remembers where each computer is. This means that when it receives a data packet meant for a particular computer, it sends it only to the port where that computer is connected.

Switches can be linked together to form one bigger switch. For example, if you have a lab with 25 workstations, you can link together two 16-port switches using a fly-lead.



Every switch has a couple of special high-speed ports. These are used to link the switch to the server, or to link switches to each other.

3.3.2.2. Cabling

Category 5 cable, commonly known as CAT-5, is an unshielded twisted pair type cable designed for high signal integrity. The actual standard defines specific electrical properties of the wire, but it is most commonly known as being rated for its Ethernet capability of 100 MBit/s. Its specific standard designation is EIA/TIA-568. CAT-5 cable typically has three twists per inch of each twisted pair of 24 gauge copper wires within the cable. Another important characteristic is that the wires are insulated with a plastic (FEP) that has low dispersion; that is, the dielectric constant of the plastic does not depend greatly on frequency. Special attention also has to be paid to minimising impedance mismatches at connection points. In practise, this means that, when you attach connectors to the cable ends, you shouldn't untwist more of the cable than absolutely necessary.

3.3.2.3. Cabling Topology

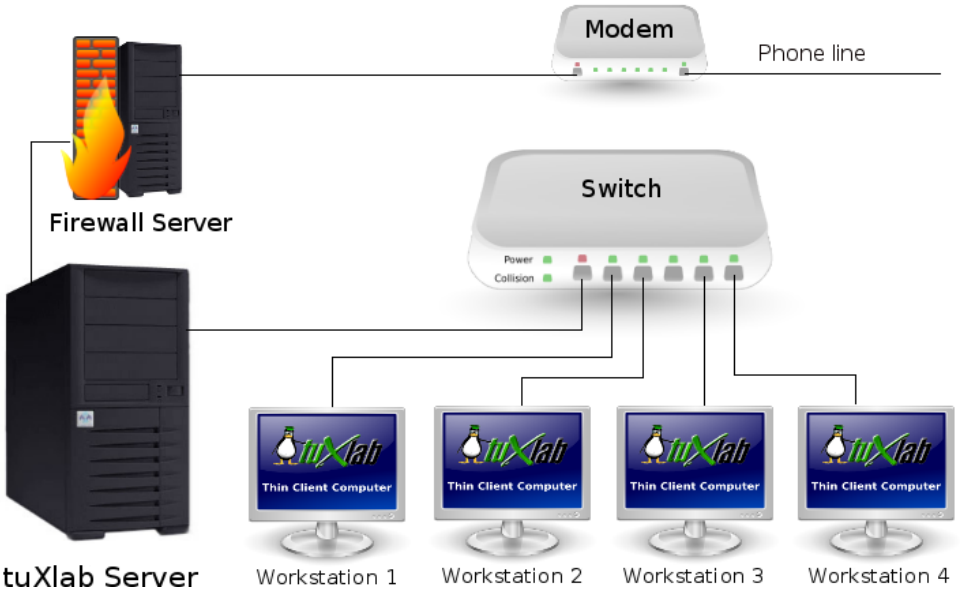
Your tuXlab uses a star topology network. This means that every thin client connects directly to the switch, which gives each client a fast connection to the tuXlab server. In other topologies, such as the ring topology mentioned earlier, messages are passed on from computer to computer. Star topologies are the fastest, and very effective, and are best suited for thin client networks.

In the diagram below, you will notice that all the thin clients, as well as the server, connects to the switch using cable, which we call “cat5” cable. The server connects to a faster port, called the “Gigabit” port (1000mbps). It has ten times the data throughput of the other ports, allowing computers to have fast access to the server.

If your lab has an Internet or e-mail setup, your server will also be connected to a “Gateway computer”. This computer will dial-up at night to collect your email and off-line content. The gateway computer will have a modem attached, that converts the sounds from your phone line to digital signals that your computer can understand, and vice-versa.

When troubleshooting, more than half the work is identifying the problem. Once you have a good overview of how things fit together, it is easier to locate the cause of the problem.

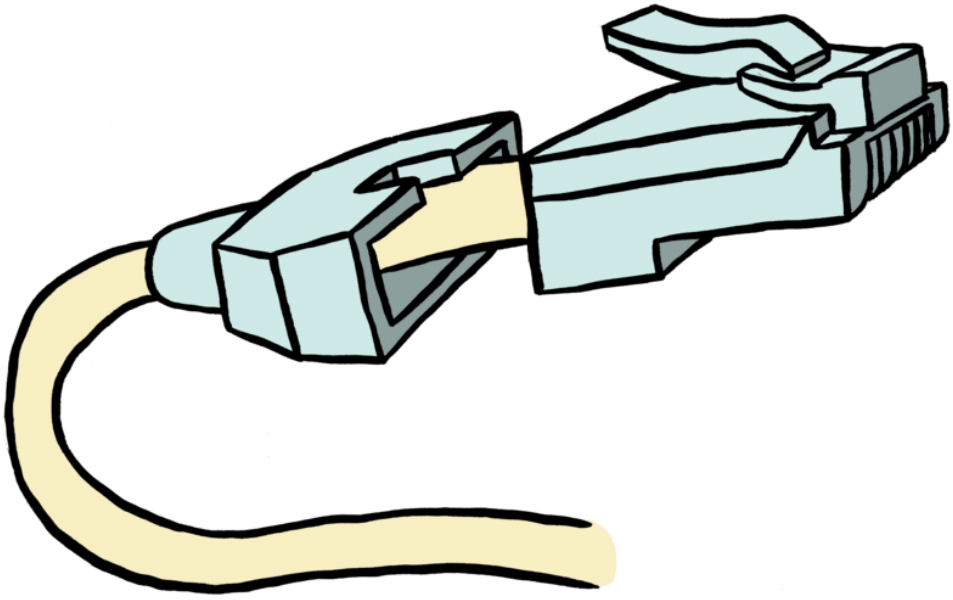
Thin clients rely on a network connection to work, so if all the computers fail to start up, you might want to check if the networking switch is turned on, or that the server is properly connected to a switch. If only one computer fails to start, it might be a problem with that particular cable.



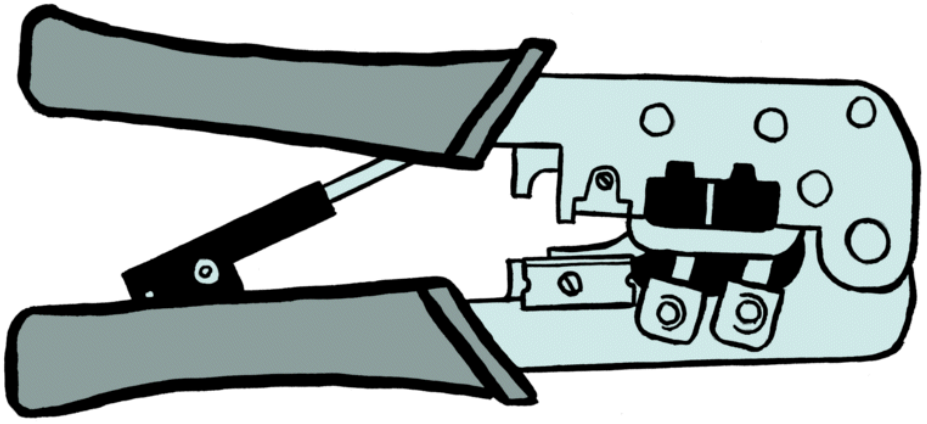
3.3.2.4. Building the network

When laying CAT-5 cable you need a crimping tool, RJ-45 jacks, and boots for the jacks (if you don't know what a *boot* for the jack is take a look at a network cable and find the ends - the boot is the covering that shields the connector).

The crimping tool is a clever piece of work. It combines the functions of a cable-cutter, wire-stripper, and a special grip specifically designed to fix the RJ-45 jack to the cable. I'll explain them as I go through the steps of preparing a cable...



An RJ-45



A crimping tool

3.3.2.4.1. Cutting the cables

The first thing you need to do is cut the cable into the appropriate lengths, using the crimping tool's cable-cutter. To do this, measure the distance from the box where the switch will be installed to the furthest computer in each row. (Usually, in a tuXlab, there will be four rows of workstations). It's easiest to use the cable itself for this, and to mark the length with a piece of masking tape.

To keep things organised, write something on the masking tape to identify the computer which the cable is meant for. Label the rows using a letter (so that you have rows A, B, C and D), and label each computer in a row with a number (so that you have A1 to A8, and so on). Once you have the longest cable in each row, you can figure out all the other lengths by shortening each subsequent cable with the distance between two workstations (normally, this will be 1200mm).

While you are cutting the cable into the right lengths, take care to keep the cables for each row together. Bind all the cables for a row together in a bundle, using masking tape. At the one end of the bundle (the switch end) all the cable ends will be together. On the other end, the ends will vary from the shortest to the longest.

Besides the cables from the switch to the workstations, you also need to cut a couple of fly leads. These are used to connect the server(s) to the switch, and also to link together multiple switches.

3.3.2.4.2. Laying the cables

Once all the cables have been cut and gathered together in bundles, you can take them into the lab. Put them on the ground underneath the desks, and ensure that the cables at the switch end can comfortably reach the switch.

If your network shares the same trunking with the electrical wiring of the lab, you MUST switch off the lab's power at the electrical subdivision board for the lab.

Now you need to put the cables inside the trunking. To do this, get as much help as you can muster, as it's hard work and no fun to do alone. Take the cover off the trunking. Note carefully where each workstation will be standing, and drill a small hole in the trunking below each workstation, for the CAT-5 cable to reach the workstation. While the cables and the covers are lying on the floor, thread each cable through the correct hole in the trunking (the cable for computer A1 goes through the hole for A1, and so on).

Once this is done, carefully put the cables inside the trunking and put the covers back on. Pass the cable ends up above the desks. You should have about 1m free cable for each workstation.

3.3.2.4.3. Crimping the cables

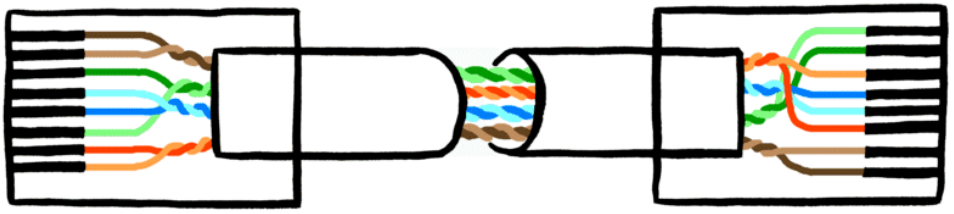
Stepping back, your lab looks the same as before, with the addition of cable ends emerging above the desks, and a whole bundle of cables terminating at the switch cabinet. Now you need to attach RJ-45 jacks to the cable ends, so that they can be plugged into the switch at the one end, and into each workstation's network card on the other end.

To do this, complete the following steps for each cable.

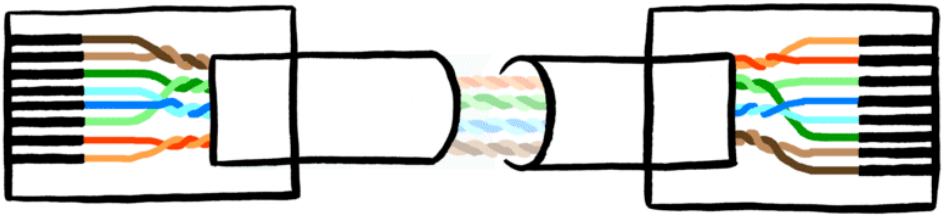
1. Insert the 'boot' over the cable. This will cover the exposed wires where the RJ-45

jack is attached to the cable wires.

2. Cut through the sheath around the cable to expose the pairs of coloured wires, without damaging them.
3. Untwist about 2cm of each pair of wires (no more, as this impairs the effectiveness of the cable for data transmission).
4. Arrange the wires in the correct colour sequence. (Straight-through cabling for cables between the switch and workstations, or Cross-over cabling for fly leads that connect switches, or that connect the switch to the server).
5. Insert the wires into the RJ-45 connector. Push them up so that all the wires terminate right at the tip of the connector.
6. Check the colour sequence of the wires again.
7. Crimp the wires to the connector using the tool. You'll notice that the connector has copper strips along the top. These connect to matching strips in the plug of the workstation's network card or the switch. When you crimp the connector, it bites into the wires through their plastic covering, connecting its copper strips to the copper wire. This is why it's critical to push the wires right up to the tip of the connector, so that the connector's teeth find the wire.
8. Test the cable using a continuity tester, if you have one. If you don't, you'll just have to figure out whether it works by trial and error later.



Cross-over CAT-5 cable



Straight-through CAT-5 cable

3.3.3. LANs and WANs

You have now constructed a *local area network*, or LAN. It's what gives your tuXlab life, but it stops at the classroom walls. To be able to send and receive email or access the internet, it is necessary to connect to further networks. This happens over a *wide area network*, or WAN. A WAN is a computer network covering a wide geographical area. The grandest example of a WAN is the internet.

WANs are used to connect local area networks together, so that users and computers in one location can communicate with users and computers in other locations. Many WANs are built for one particular organisation and are private; others, built by internet service providers, provide connections from an organisation's LAN to the internet. This is the case with a tuXlab that is connected to the internet. *Private* WANs are most often built using leased lines. At each end of the leased line, a router connects to the LAN on one side and a hub or a switch within the WAN on the other.

While a LAN is a network of computers and devices, a WAN is most often a network of networks. A router, or a computer configured to function as a router, on each network, connects to routers on other networks.

Behind every router there may be many computers (or networks) that are not directly connected to the internet. It is then the function of the internet gateway computer to route packets from outside networks to the correct computer on the inside. All the computers on a LAN share a single connection to the internet. In the case of tuXlabs, the Wizzy server, if you have one, acts as a gateway computer.

However, because a permanent WAN connection is very expensive in South Africa, especially in rural areas where telecommunication infrastructure may be lacking, a tuXlab will typically only connect to the internet only intermittently. When it is not connected, the Wizzy server acts as a proxy for the internet, serving cached requests, and queueing email to send later when the connection is established again.

3.3.4. TCP/IP

The network protocol of the tuXlab LAN is the same as that used for communication on the internet, namely TCP/IP. This is the *Transmission Control Protocol (TCP)*, encapsulated within the *Internet Protocol (IP)*. The Internet Protocol takes care of routing data packets from a source IP address to a destination IP address. An IP address consists of four numbers that look like this: `192.168.10.200`. IP packets can contain TCP packets. Whereas an IP packet only knows where it should go, TCP packets contain information about their position in a sequence of packets.

TCP is wonderful: it makes it possible to treat a flaky network as though it were perfectly reliable. When you send anything across a TCP/IP network (e.g. an email message, an image, or a document) it is broken down into many TCP packets. These are numbered and sent, one by one, to the destination computer. At the destination, the sequence number is used to put the packets in the correct order (as they may have become mixed up in transit).

If there are gaps in the sequence, only those packets are requested again. If some packets are received more than once, the extra packets are simply dropped. Once all the packets in the sequence have been received, the entire file has been transferred successfully.

Unsurprisingly, an IP address cannot be just any four numbers. Actually, there is a lot of underlying structure. In the first place, the numbers are a sequence of four *bytes*. Computers generally handle data one byte at a time, so it's convenient to specify things as a sequence of bytes. A byte consists of eight binary digits. The binary number system has only two digits, namely 0 and 1: just as 99999999 is the largest number that you can express with eight decimal digits, 11111111 is the largest number that you can express with eight binary digits. If you convert that number to decimal, you get 256. For this reason, a sequence like `300.5.502.743` does not make any sense as an IP address.

In the second place, some address ranges are reserved. For example, all the networks that start with `192.168.***.***` as their first two digits are private, not routed on the internet. The whole public IP address space is divided among ISPs. Each ISP gets a range of numbers that they may portion out between its customers. This range of numbers is described in terms of a *netmask*, a number which looks similar to an IP address, but is used to match all the IP addresses that belong to a particular network. Private networks can also be segmented into subnets using netmasks. Any computer on a network can send IP packets to any other computer on the network, but to send an IP packet to a computer on a *different* network, there must be a gateway computer which is configured to help the packets cross from one network to another.

Gateway computers also implement network management policies, e.g. by way of firewall software, that specify what traffic is allowed into and out of a network.

3.4. Configuration

3.4.1. Wizzy configuration

A Wizzy server plays two roles. In the first place, it serves as an internet proxy, serving cached pages to a lab without a permanent internet connection. In the second place, it connects to the internet and retrieves requested pages to store them locally, sends queued email, and fetches received mail.

These two roles don't have to be fulfilled by the same server: they may be split across two servers, for example if the school has no internet access. In this case, there will be a Wizzy in the tuXlab and another one at a remote location with internet connectivity.

The hardware requirements for the Wizzy server are far less than those of the classroom application server, since it has to do far less work. All it needs to do most of the time is to serve up saved pages from its hard disk. For this, a server with a 200+MHz CPU, 256MB RAM, and a 40GB RAID 1 disk array is adequate.

3.4.1.1. Software used by the Wizzy server

- The Wizzy server uses DHCP (Section 3.4.2, “Dynamic Host Configuration Protocol” [3–26]) and TFTP for booting, the same as the classroom server.
- It uses the Lightweight Directory Access Protocol (LDAP) for authentication. When users access their email on the Wizzy server they will need to supply a username and password. This information is kept by an LDAP server, which maintains a directory where user information may be looked up. It is analogous to a telephone directory.

  Ideally, there should be only one directory for a tuXlab, which contains the information on all the users and resources (such as printers) in the lab. Currently, however, the directory maintained by the Wizzy server is separate from the user database on the classroom server.
- The Unix-to-Unix Copy Protocol (UUCP) is used for *all* email sending and receiving, as well as for fetching web pages. As long as a service can be configured to use UUCP for communication, it is possible to provide the service without a permanent internet connection. For this reason, Wizzy sticks to UUCP.
- The Wizzy server uses BIND (the Berkeley Internet Name Daemon) for domain name services (DNS), on the network. DNS is how IP addresses, such as `192.168.0.254`, are resolved to readable domain names, such as `server.myschool.tuxlab.org.za`.
- For email, Wizzy uses the Courier IMAP server for inbound mail and webmail.

  IMAP (the Internet Mail Access Protocol) provides a way for all kinds of mail clients (such as Thunderbird, or Mozilla's email component) to access a mail store. In Wizzy's case, the Courier package handles mail storage and IMAP access. Wizzy also provides a webmail client to access the mail store using a web browser.
- To send mail, Wizzy uses the **exim** program.
- As discussed in Section 3.4.1, “Wizzy configuration” [3–24], one of the most important jobs the Wizzy does is to provide a browsable offline copy of all the web pages that interest tuXlab users. It does this using the wwwoffle [http://www.gedanken.demon.co.uk/wwwoffle/] program to provide a local web cache. It augments **wwwoffle** (the World Wide Web Offline Explorer) with some custom programs, since Wizzy's disconnected mode of operation goes beyond the options offered by **wwwoffle** on its own.
- The Wizzy server provides its administration functions as well as webmail as web pages, and uses the apache [http://www.apache.org/] webserver to serve these pages. They may be accessed by going to http://wizzy/ [http://wizzy/] on the tuXlab LAN.
- For a local FTP server, Wizzy uses vsftpd. This is only used if there are files that need to be downloaded from the Wizzy server via FTP.
- In its role as a connection to the internet, the Wizzy server can connect to any ISP with which the school has an account, in order to retrieve web pages. It does not, however, use any email facilities that the ISP may provide: all mail goes through Andy Rabagliati's server in Cape Town using UUCP.

- For the sake of data integrity, Wizzy uses a RAID disk system for storage. RAID is a *Redundant Array of Inexpensive Disks*. It is a way to federate multiple hard disk drives in such a way that the failure of any one disk does not result in data loss.

3.4.1.2. Wizzy as a classroom server

The Wizzy project predates tuXlabs. It can also be configured to provide a similar range of application serving functions as the tuXlab classroom server provides. In this configuration, it uses the following packages:

- Like the tuXlab server, Wizzy uses the LTSP packages for its thin-client NFS-mounted root directory.
- As a classroom server, it also uses NFS (the Network FileSystem) for home directories that are NFS-mounted by the thin clients.

3.4.2. Dynamic Host Configuration Protocol

Every computer on a local area network (LAN) needs to have a unique IP address so that it may send and receive data. The Dynamic Host Configuration Protocol (DHCP) is a networking protocol that allocates IP addresses dynamically to computers on a LAN. Without it, an administrator needs to give each client computer a static IP address manually. This may seem simple enough to begin with, but given time, it slowly turns into a nightmare: computers are added, removed or moved about, and the number assignments eventually become arbitrary and troublesome to keep track of. On a network with manually assigned addresses, it's also awkward to connect transient devices such as laptop computers that are also used on many other networks. You have to talk to the system administrator to find out the network configuration, and then check the network to find a free address. With DHCP, it's easy: just plug in an Ethernet cable for the new device, and it will immediately request an IP address from the classroom server, which will assign an unused number to it.

In a tuXlab, the classroom server is configured as a DHCP server. A system administrator assigns a range of IP addresses to the server. Each client computer on the LAN has its TCP/IP software configured to request an IP address automatically from the DHCP server when that client computer starts up. The request-and-grant process uses a lease concept with a controllable time period. This eases the network installation procedure on the client computer side considerably.

In addition to the IP address, a DHCP server can set other configuration information, such as the address of the DNS server, the DNS domain of the client, and the gateway IP address, so that the client computer can be fully functional.

Before I continue, let me explain the concepts I've just introduced. First, the DNS domain: all Linux computers are given a hostname upon installation of the OS, which is used in system messages and configuration. When the computer joins a network, its hostname and

the domain of the network together combine to form the Fully Qualified Domain Name (FQDN) of the computer. In the case of a tuXlab, the FQDN of each workstation will be something like `client1.myschool.tuxlab.org.za`. The public DNS servers that resolve the domain names of tuXlabs on the internet are maintained by SchoolNet http://www.schoolnet.org.za/schoolsurveys/suveys_index.htm [http://www.schoolnet.org.za/schoolsurveys/suveys_index.htm]

Secondly, the gateway IP address. In Section 3.3.3, “LANs and WANs” [3–23] we explained that the internet is a *network of networks*. For data packets from a computer on one network to reach a server on another network, there needs to be a gateway that is connected to both networks at once. Usually, the gateway computer will have a network card for every network to which it is connected.

By default, the LTSP server uses its first network card (`eth0`, numbered from 0 like most things in the computer world) for the classroom LAN. It runs DHCP on this card, and automatically gives out IP numbers upon request. It then accepts BootP (Boot Protocol) and PXE (Pre-boot eXecution Environment) boot requests, and passes on the Linux kernel to the client using TFTP for the transfer. Once the client has received the kernel, it boots into Linux. The default **`dhcpd.conf`** file will support over 200 clients. The LTSP server will not answer DHCP requests over `eth1` (with the default settings.)

3.4.2.1. Files

The configuration settings for the DHCP server are contained in the **`/etc`** directory (standard Linux location for configuration data) in the file **`/etc/dhcpd.conf`**. For more information about **`dhcpd.conf`** please visit Linux Reviews help section [http://linuxreviews.org/man/dhcpd.conf/] or read the **`dhcpd.conf`** man help [type man dhcpd.conf at the command prompt of your Linux server].

3.4.3. Network configuration

The first network card, `eth0`, is the interface on the thin-client side of your LTSP server. This network card connects to your terminal hub. The `192.168.0.x` address range is designated as a "private" IP range for internal networks. It is not routed on the internet. IP traffic from your clients are routed to the internet through `eth1` (Note that if there is a Wizzy server, it will be the one with the two network cards).

The classroom server has the last available address in this range, namely `192.168.0.254` (`192.168.0.255` is the *broadcast* address: packets sent to this address reach all the computers on the network). The first client will be assigned an IP number of `192.168.0.253`. [1]

3.4.3.1. Wizzy network configuration

[1] For more information, see http://www.k12ltsp.org/install.html [http://www.k12ltsp.org/install.html]

When the tuXlab has a Wizzy server, there are a couple of other aspects to network configuration.

- Protocols

  Wizzy Digital Courier relies on standard networking protocols for the interactive portions, linked by UUCP for the intermittent sections.

- Mail configuration

  During installation, you must choose a hostname for the Wizzy server. This hostname will identify the server within the mail domain, which is `wizzy.org.za` (because Wizzy provides the email infrastructure), so that your complete mail domain becomes `myschool.wizzy.org.za` (where *myschool* is the hostname you chose). This means that all the tuXlab users will get email addresses like `user@myschool.wizzy.org.za`.

  You will also need to contact Andy Rabagliati at XXX, to tell him your UUCP password, as he must set up mail routing for you.

  For tuXlabs, Wizzy servers have a special configuration available --- accessible by typing the following at the Syslinux boot prompt:

```
boot: linux ks=cdrom:/tsf-ks.cfg
```

3.4.4. Network Filesystem

tuXlab uses NFS, the Network Filesystem, to make the home directories of lab users appear to be local to the client workstations, even though they really reside on the classroom server. The NFS configuration is specified in the file **`/etc/exports`** on the classroom server.

3.4.5. LTSP configuration

The LTSP configuration is specified in the file **`lts.conf`** on the classroom server.

3.4.6. tftpboot

Upon power-up, the BIOS of each client workstation contacts the classroom server, and retrieves the Linux kernel from it via the TFTP protocol. XXX: more detail.

3.4.7. Users and groups

All the users of the tuXlab will have accounts on the classroom server. (Additionally, if they have email they will have accounts stored in the Wizzy server's LDAP directory.)

3.4.7.1. Permissions

Access to directories, files and executable programs under Linux is managed in terms of users, groups and permissions. Every user belongs to a group, and every file belongs to a user and a group. The basic permissions are *read*, *write* and *execute*. For every file and directory, these permissions can be set for the user who owns the file, the group, and for all others (i.e. everyone but the owner or the group). For example, here are the permissions of a user's home directory:

```
jean@klippie jean $ ls -ld /home/jean
drwxr-xr-x 112 jean users 6664 Des 26 17:31 /home/jean
```

The permissions are shown by the string `drwxr-xr-x`. The first character, `d`, indicates that this is a directory. You should read the following 9 characters in groups of three, that show the permissions for the owner `jean` (`rwx`), the group `users` (`r-x`), and all others (`r-x`). In this case, the owner has *read* (`r`), *write* (`w`) and *execute* (`x`) permissions, while the group and others only have *read* and *execute* permissions. In the case of a directory, *execute* permissions means that you are allowed to access the contents of the directory. This home directory may therefore be read by everyone, but only the user may change it.

Here are the permissions on the file that contains the system user database:

```
jean@klippie jean $ ls -l /etc/passwd
-rw-r--r-- 1 root root 2118 Des 1 05:32 /etc/passwd
```

These indicate firstly that this is a regular file, not a directory (the leading `-`), and that the owner `root` has *read* and *write* permissions (`rw-`), and everyone else have only *read* permissions (`r--`). In effect, this means that only the `root` user may add, modify or delete users.

You may further note that the group to which this file belongs is also `root`. This group only has one member (the `root` user), and is used for files that are under control of only this unique user.

3.4.7.1.1. The superuser

Every Linux machine normally has a user called `root`, who has all permissions. When a system administrator needs to do maintenance, they log in as `root` only to make the necessary changes, and then switch to their regular user again.

3.4.7.2. Retiring users

When a class graduates from your school, or pupils leave for other reasons, you will need

to retire their user privileges and delete their login profile and whatever storage facility (disk space, etc) was assigned to them.

3.4.8. Printing

The tuXlab uses the CUPS (Common Unix Printing System) service. This is the default printing system of most Linux distributions. It should be automatically installed when you install your Linux distribution on the tuXlab server.

For a list of compatible printers and for help regarding CUPS please visit the LinuxPrinting [http://www.linuxprinting.org/cups-doc.html] website.

3.4.9. Developing a backup procedure

The importance of backing up a system can never be stressed enough. You never know when the power may cut out or the hard drive may crash. Even though you can restore the operating system from the distribution CD-ROM, there are other files that you need to consider. What about the configuration changes that you made? There are also files created by users, what about those?

Follow these steps to create a backup plan:

1. Make a list of the files and directories that you need backups of. You'll always want to backup system configuration files in the **`/etc`** directory, other configuration files may be found in **`/usr/lib`**. In addition, you may want to backup user files in the **`/home`** directory as well as the superuser (`root`) files in **`/root`**.
2. Find a few tools to use when backing up and archiving files and directories. Several tools are available that will archive a group of files, and there are tools that will compress files and archives.
3. Decide how often the system and individual files need to be backed up. How often do your files change? If files change frequently, the your backup frequency should match the change frequency. So, you may need to perform a backup every day. If you only make one or two configuration changes on occasion, you can easily backup the configuration files only when the changes is made.
4. Select a storage medium that will store the backup file. If you have a few files to backup, you could just store them on a floppy disk. If you have more files, or larger files, you can consider using a zip drive or a CD-RW drive.
5. Store the files in a safe place. The safest place to store the backup media is at a location different from where the computer is located. To be really safe, this location needs to be protected from fire and other hazards. You may also want to keep a copy of the backup files close by so that you can quickly restore lost files.

Tip: Always make a copy of configuration files before you make any configuration changes. That way, should your new settings not work, you can restore the old

configuration files.

3.4.10. Downloading the internet

On the homepage of the Wizzy server, accessible by browsing to the server using a web browser (http://wizzy/), there is a form to request web sites for caching. Educators may preconfigure the server with a range of interesting websites, and learners may request new sites per email or in the classroom situation. When the Wizzy server connects to the internet (directly, or indirectly via a counterpart in a remote location) it fetches the requested pages, and refreshes those pages which it should keep up to date.

3.4.10.1. Accessibility

All email and websites browsed by the tuXlab users during the day are served directly from the server in the classroom. This means instantaneous and constant connectivity to a local miniature internet over a top quality hardwired network. There is no waiting. [2]

3.4.10.2. The internet as school library

Nothing will ever approach the importance of printed books for fostering a love to read. Only by curling up with a book somewhere safe and comfortable can your mind rise on the wings of the pages and take you to worlds beyond your own. Books are expensive and scarce, though. There are never enough copies. Furthermore, some books work very well in electronic form, especially reference works that you don't read from cover to cover. For these reasons, the internet can be a wonderful addition to the school library.

3.4.10.2.1. Wikipedia

If the Wikipedia has been installed, the Wizzy server will also serve a local mirror of it.

The Wikipedia [http://www.wikipedia.org] is a one-of-a-kind collaborative effort to create an online encyclopedia. It is managed and operated by the non-profit Wikimedia Foundation. In addition to standard encyclopedic knowledge, the Wikipedia project has developed offshoots that include the kind of information associated with almanacs and gazetteers, as well as coverage of current events. With a little effort, it can be used offline in schools.

The Wikipedia is being written collaboratively as an open source project, using specialised *wiki* software called `mediawiki`. This means that anyone, including you, can make changes and add information. In fact, the openness of Wikipedia to local knowledge is one of its most important characteristics. For the first time, the world is being described not by an editorial committee but by a global community of volunteers, who are able to augment academic knowledge with local experience.

A *wiki* is a simple form of web-publishing that was thought up by a software design

[2] See http://wizzy.org.za/article/articlestatic/19/1/2/

consultant called Ward Cunningham. His original wiki is still running at http://c2.com/cgi/wiki [http://c2.com/cgi/wiki]

In a wiki, every page is editable as plain text which is converted to HTML when served, following a number of simple conventions that convey the structure of the text. Specifically the convention for linking between pages is what gives wikis their identity. To create a link to another page, you simply write its title in a special format (e.g. `AnotherPage`), and if that page exists the title is turned into a hyperlink to it. If it *doesn't* exist, the title is turned into a link leading to a blank page where you may create `AnotherPage` instead. The `mediawiki` software also keeps a record of all versions of a page, so that the entire editorial history of a page may be examined.

For schools that don't have a permanent internet connection, this entire resource, because of its Open Content license, can be installed locally. Even for schools that are permanently online there are advantages to a local install, in preference to a `wwwoffle` mirror:

- It's more effective, since it doesn't fetch all the HTML furniture (everything that repeats from page to page, such as the page header and footer) repeatedly.
- It's complete. A `wwwoffle` mirror would contain only the articles that were requested, piecemeal.
- It's searchable via SQL.

A local Wikipedia mirror does introduce some complexity though. In addition to `apache`, it requires PHP, Mediawiki and `mysql`. [3]

3.5. Evaluating tuXlab Hardware Usage

3.5.1. Benefits

3.5.1.1. Easy maintenance

If a user reconfigures a workstation in a fat client computer lab, all the other users of that workstation will have to cope with these changes. This means that if someone inadvertently sets the workstation to use black type on a black background, for example, then no-one will be able to see what's going on.

In contrast, in a thin client lab, every user has their own files, their own email, and their own desktop environment that they can change to their liking without influencing anyone else, all stored on the classroom server. A configuration mistake like the above will inconvenience only themselves. [4]

[3] See http://wizzy.org.za/article/articlestatic/23/1/2/
[4] http://wizzy.org.za/article/articlestatic/5/1/2/

3.5.1.2. Cheap hardware

Most of the equipment in a computer lab are workstations for learners to use. There may be one or two printers, network switches and a server, but they are far outnumbered by between 20 and 40 client workstations.

In a tuXlab, these can all be really old, used, previous-generation computers. This is because the demands on these machines is so slight that almost anything will do. All those stacks of old computers everywhere that no one knows what to do with are suddenly useful, and saving schools vast amounts of cash that they would normally have to outlay on relatively new equipment so they can run contemporary software.

The thin client paradigm also means that requirements for uniformity among terminals is relaxed. As long as they conform to a couple of basic requirements (network boot, SVGA graphics card, enough RAM) it doesn't matter if they have idiosyncratic hardware. Only the server configuration needs to be maintained. Heterogeneous hardware doesn't make life difficult for the admin.

The thin-client network in the lab ensures that each terminal, no matter what its own computing characteristics, behaves with all the speed and capability of the server, so each user has an experience of top quality, smooth, fast computing. Unfortunately, this does mean that if some of your client workstations are powerful, modern machines, they may not be fully utilised in a default tuXlab configuration, as they will still be letting the server do all the work.

3.5.1.3. Less theft

In a tuXlab, the most accessible hardware is also the easiest to replace and the hardest to use outside the lab. A tuXlab client workstation on its own, without the classroom server, is more or less useless. It's too bulky for a doorstop, and it can't run modern software. It doesn't even have a hard drive. [5]

3.5.1.4. Mobile desktops

As the terminals only serve to display a session from the classroom server, it doesn't matter which one you use. If one breaks while you've using it, you can move to the next one, log on, and pick up where you left off. If someone is using the workstation where you were working, go to another one and log in there to regain access to your desktop.

3.5.1.5. Data easy to back up

All the data in a tuXlab resides on the disk array of the classroom server. Instead of having to backup 20 or 40 individual hard drives, it's possible to backup only one, and still get a complete backup of everyone's data.

[5] http://wizzy.org.za/article/articlestatic/5/1/2/

3.5.2. Thin Client configuration

While there is little to do for the installation of the thin client workstations, there are a couple of things you can pay attention to.

Network cards

All the client computers need to get a network card with space for a boot PROM, so that they can start looking for the server on the classroom network as soon as they are switched on. The server also needs a network card, and it needs to be a fast one (gigabit Ethernet), as the link between the server and the switch is ten times quicker than the link between the switch and the client workstations.

Boot PROMs

Depending on the network cards you managed to get, the correct Etherboot image may need to be written to the network cards. The Diskless Remote Boot in Linux [http://drbl.sourceforge.net/] project has made available an Etherboot NIC Detection Disk [http://www.k12os.org/modules.php?op=modload&name=News&file=article&sid=46&mode=thread&order=0&thold=0] which can help you to determine what you need to write to the card.

Dual-booting

The BIOS in the client workstations normally needs to be configured to boot from the network. To do this, watch the workstation screen after turning the power on --- for a couple of seconds, the BIOS will display a notice that you may press a key (such as **DEL** or **F8**, depending on the BIOS) to enter setup mode. This will enable you to specify where the computer should look for boot records during startup; for example on a CD, a floppy disk, a hard disk, or the network card. In the event that you have a relatively capable workstation with a hard drive, you may want to boot the workstation as a standalone computer from time to time. To do this, you may configure the BIOS to look for a boot record on a floppy disk before trying the network card.

One kind of scenario where this may be useful, is where you have an existing computer lab with software installed at each workstation. Boot from hard disk to access the standalone workstations, and boot from the network to have a tuXlab! It is even possible for the standalone workstations to access the tuXlab network, and to use the classroom server as a file server, and the Wizzy server as an internet proxy. Since a tuXlab is implemented using standard protocols, this will work no matter what operating system is installed on the workstation computers.

Other resources

This cookbook can do no more than scratch the surface. Some of the other resources regarding thin-client computing available on the web include:

- Diskless Remote Boot in Linux (DRBL) for Redhat 8.0, 9,

Fedora Core 1, 2, Mandrake 9.2, 10
[http://drbl.sourceforge.net/redhat/]

- Root over NFS clients & server HOWTO [http://www.tldp.org/HOWTO/Diskless-root-NFS-HOWTO.html], if your workstations have disks, and you don't want to delegate processing to the classroom server.
- Network Boot and Exotic Root HOWTO [http://www.tldp.org/HOWTO/Network-boot-HOWTO/index.html] This document explains how to quickly setup a Linux server to provide what diskless Linux clients require to get up and running, using an IP network. It is based on the Diskless-HOWTO, the Diskless-root-NFS-HOWTO, the Linux kernel documentation, the Etherboot project's documentation, the Linux Terminal Server Project's homepage, and the author (Brieuc Jeunhomme)'s personal experience.

3.5.3. Drawbacks

Every solution will have some drawbacks, and a tuXlab is no exception. I'll just mention a few in passing.

Unreal Tournament will be lethargic or lag in response rate	Graphics-intensive applications such as games will not perform well, as all the display information will have to be pushed over the network by the server. This is hundreds of times slower than driving a local graphics card. Playing action games, however, is not a goal of the tuXlab project.
All the clients run the same OS	Since it's really only one instance of Linux serving all the desktops and applications, all the clients in the lab will necessarily offer a Linux environment. It is possible for the server to run software such as Wine (which enables many Windows programs to run under Linux) or VMWare (which allows the server to run instances of other operating systems), but in these cases the underlying system will still be Linux, and the server will still be doing all the work.
Single point of failure	While it's very convenient that the thin-client workstations are interchangeable and that you can access your desktop from anywhere, it does mean that a catastrophic failure of the server will put **all** client workstations out of commission.

4

Management Model

The management module deals with the human processes of setting up a TuXlab. The first section deals with the procedures that need to be followed when applying for a TuXlab, and an overview of the installation process. The second section deals with the aspect of "Change Management", presenting information regarding executive buy-in, the computer committee, the idea of "clustering", and using a TuXlab to bring services to the greater community. The third section deals with the process of management after a TuXlab has been installed and is in use. This information includes discussion and examples regarding reporting, training, budgeting, and involvement in the incentive scheme.

4.1. Setup Process

The management of any project begins with the process of setting up that particular project. With this in mind, we start the tuXlab. Cookbook Management Module with the procedures outlining how to begin the application process. The rest of this section will describe the process of how a school goes about applying for a tuXlab.

4.1.1. Obtaining Information About the Programme

There are many ways to obtain information about the tuXlab project. The following list presents the main was of doing so:

1. **Word-of-Mouth**

   Traditionally, word-of-mouth has been the chief mechanism behind tuXlabs' popularity. Schools in similar districts are in contact with each other, principals create forums, parents have communal relationships outside of their schools and the learners themselves have informal friendship circles. A school with a computer centre, especially if it is still quite new, is quick to advertise their ownership.

2. **The Press**

   The Shuttleworth Foundation makes periodic press releases advertising their projects and programmes to the community at large. tuXlabs often advertise large install events or sponsored schools in the programme.

3. **The tuXlab Website**

   The web portal www.tuxlab.org.za [http://www.tuxlab.org.za] is the primary source of information for the programme and is the home of all current information that is tuXlab related.

4. **Help-Desk**

   The tuXlab help-desk offers telephonic and e-mail support and can answer, or redirect, all queries for the tuXlab programme. Additionally, the project administrator can be contacted telephonically or via e-mail.

5. **Mailing List**

   The creation of a Linux User Group (LUG) centred around the activities in schools can host various mailing lists pertaining to tuXlabs, e.g. announce and chat. Interested parties can register on these lists and receive information from volunteers and tuXlab team members directly.

4.1.2. Applying for a tuXlab

When applying for your tuXlab you will need to follow these steps:

1. **Read Documentation**

   As a starting point, you need to read the Introduction section of the tuXlab Cookbook. If possible please read through the rest of the Cookbook and the documents contained in the Appendix.

2. **Return the Questionnaire**

   The “School Awareness Questionnaire” (see Appendix E) needs to be filled out and returned to the tuXlab project coordinator. This is used for the tuXlab team to perform an initial assessment on the applicant and to gain various details on the school, e.g. contacts, size & demographics.

3. **Complete the Business Plan**

   The Business Plan needs to be completed and returned to the tuXlab project coordinator. There is a “Business Plan Guidelines” in Appendix A to use as a starting point for you to understand what is needed in the business plan.

4. **Attend an Installation**

   It is ordinarily required that the applying school sends a representative to a tuXlab install at another school. The obvious exception being if tuXlabs enters into a new geographic area. The representative must sign the attendance register which will be sent back to the tuXlab office by the installing school.

5. **School Visit**

   The tuXlab team will physically visit the school at this stage. The reason for this is to:

   - initiate personal contact

- visually assess the school
- check that the requirements have been met

6. **Attend Another Installation**

   The tuXlab team will notify the applicant that they are being processed further and that should attend another installation. If the school has already attended other installations then this step is waived, as is often the case.

7. **Confirm**

   The tuXlab team will then send a letter of confirmation to the school and will set up a date for the tuXlab installation.

8. **School Infrastructure**

   On confirmation, the school must prepare their proposed venue before the installation date. The *Infrastructure*, *Hardware*, and *Software* modules of the tuXlab are available to assist you in this preparation.

9. **Second School Visit**

   The tuXlab team will visit the school again before installation to ensure that the room is ready and meets the requirements.

4.1.3. Selection Process

The selection process describes the criteria that the tuXlab team considers when choosing schools for installation. It can be considered subjective in some areas, however, there is often no substitute for experience.

Core information about the school is obtained from the questionnaire, the business plan and any interview conducted during a school visit. If the schools is part of a programme run by another organisation then they will be consulted as well.

Schools are assessed not only on enthusiasm and commitment to the programme but also on their willingness and ability to contribute to its objectives.

4.1.3.1. Questionnaire

The information that the questionnaire should yield includes the following:

- Does the school already have a computer lab?
  - If so, then details on its size, usage and how it was obtained are important
  - How it is managed and incorporated is also important

    - What existing software is available and used
- The same above applies to computers in the school that are not part of a computer lab, e.g. the secretary's workstation
- What is the perceived level of computer literacy in the school
- What is the Open Source level of awareness and literacy

4.1.3.2. Business Plan

The business plan should give an indication of:

- How the school intends to use the lab and who it will be for
- How ready the school is for the physical infrastructure
    1. In terms of location and budget
    2. In terms of how the staff could be managed into a committee with a facilitator
- If the school is aware of the risks and opportunities associated with having a computer lab
- How the school intends to take ownership of the lab
- The schools' management level and history of success
- The schools' plan and vision for sustainability
- The schools' operational plan which should include a budget outlay for maintenance

In general though, the schools often don't have any frame of reference regarding computer centres to be able to supply completely relevant information. The questionnaire and business plan should highlight this.

4.1.3.3. School Visit

The school visit can be a very subjective process. A well run school is often characterised by clean grounds, disciplined children and a visible level of respect. It is also possible to get a sense of the principals' motivations behind applying for a tuXlab at this stage by analysing the type of queries they make. Occasionally it is very self evident that the prime motivation behind the application is to obtain the infrastructure only. The visit will then largely centre around the type, cost and power of the donated computer lab with very little attention drawn to any of the softer issues.

Visiting the school also affords the chance to verify the information submitted in the questionnaire and the business plan. It has happened that a school has stored their existing equipment in a store-room before applying for a tuXlab.

4.1.3.4. Selection Criteria

Choosing an appropriate schools is difficult at best if there are a lot of applicants. Generally though, schools are chosen to build clusters geographically.

4.1.3.4.1. Geographic Clustering

Each cluster contains a minimum of three schools.

The cluster model behind tuXlabs is a very powerful way of fusing communities together and encouraging support and collaboration. Strong clusters are thus desirable and it is important to ensure that schools are clustered geographically close to each other to reduce the transport burden on the facilitators for attending meetings and training sessions.

Often, an installation cycle is centred around building up clusters that show promise and enthusiasm but have large distances to cover for events.

4.1.3.4.2. Commitment

The application process requires that the applying school attend at least two installations at other school. This is done in attempt to asses their “buy-in” to the programme. Additionally, a school that attends installations, workshops, training and cluster meetings, even when they do not have a computer centre, will definitely be considered with more sincerity than those that have not shown a similar level of enthusiasm and commitment to the programme.

4.1.3.4.3. Budget and Financial Backing

The selected school is expected to provide the physical infrastructure required for housing the tuXlab. The Letter of Intent and Criteria dictates what the school is responsible for supplying and includes:

- All electrical work
- Network trunking
- Desks, chairs and storage space
- Adequate security
- Insurance

This implies an initial capital outlay from the school. The school must also be able to support its' computer lab. Hardware has a limited warranty period and the Shuttleworth Foundation only guarantees the refurbished computer equipment for a period of 3 months after installation. The school therefore needs to be prepared for maintenance expenses and, possibly in the future, upgrade expenses.

4.1.3.4.4. Resourced vs. Non-Resourced Schools

Schools in South Africa are often described as being resourced or non-resourced. Alternatively, non-resourced schools are also referred to as disadvantaged and resourced schools as ex-Model C schools.

Whatever the terminology, the Shuttleworth Foundation tuXlab programme has focused on schools that do not have the resources or the capacity to facilitate the creation of their own computer lab. Although this is the focus, there have been exceptions where an advantaged school has been provided with a tuXlab because its' business plan reflected its' plan to extend the tuXlab's influence and accessibility to the greater community. Somerset College is such an example of a resourced school whose plans where to help fund a tuXlab for a neighbouring school, Macassar Primary, as well as servicing their community.

Such variety is believed to be important as different perspectives and competence levels within a cluster can prove to be constructive.

4.1.3.4.5. Primary vs. Secondary Institutions

The demographics of the tuXlab programme have so far favoured primary schools for the following reasons:

- Primary schools are easier to provide for in terms of content and curriculum software
- Consequently the demands on the power of the computing environment are less
- The Khanya project in the Western Cape has focused more on providing computer centres in secondary schools. Primary schools did not have a similar project focused on their needs.

The tuXlab programme does include secondary schools though for the following reasons:

1. To provide a platform for developing a model that encompasses all aspects of basic education.
2. To provide a platform for piloting other Shuttleworth Foundation projects that require a computer facility

4.1.3.4.6. Exclusion

In light of the above, a school will only be denied it's application under the following conditions:

- It is guilty of misrepresentation in its application questionnaire and business plan
- The facilitators do not attend other installations
- The schools does not supply the required infrastructure
- The school does not supply the required paperwork

Generally though, the applicant will not be denied, as such, but rather not accepted. The difference being that the applicant is not formally disqualified but rather given time to qualify completely in the future.

4.1.4. Installation Process

This installation process describes the events that happen on the day.

In the past, the installation process has evolved from performing as many activities as possible with all the roll-players to performing only relevant activities. An example would be that the server is no longer installed on-site at the school on installation day. The reason for this change is that the server installation is largely automatic and quite uninteresting for the participants.

The current tuXlab install process involves the following activities:

- The workstations are refurbished before the time and delivered on to the school on the Thursday prior to a Saturday install.
- The installation date would have been announced to the volunteers on the mailing lists.
- On the install Saturday, the volunteers are rounded up and assigned rolls and responsibilities for the day.
- On arrival at the school, the participants are given a crash course in networking. The network cables are cut, crimped and laid in conduits by the school staff, learners and volunteers.
- If necessary, the network cards will be installed into the workstations on the day.
- The switch and server are installed and connected to the workstations.
- Monitors are adjusted and networks tested.

If time and resources allow, the following activities on the day can add value to the whole process and event:

- Some kind of software installation process can be shown to the participants.
- Allow the participants to get inside a computer, either by opening some up or by letting them install the network cards.
- A basic tutorial session on system administration and what the capabilities of the computer lab are.

Basically, the more interactive and hands-on the event is, the more the participants gain from the experience.

The school is expected to have at least a 50% staff attendance at the installation. This is important as it helps to build ownership of the tuXlab from within.

Finally, the attendance register is collected and the Memorandum of Understanding (MoU)(see Appendix F) is signed with the Principal. The MoU outlines the purpose of the donation and the expectations from all of the parties involved. It essentially covers the criteria laid down in the Letter of Intent and the various support documentation. The attendance register is used for the obvious purposes of tracking installation attendance for the application process.

4.2. Change Management

For a school with no previous exposure to ICTs, there are a few changes within the school management that might need to be addressed. There are some preconditions that need to be established with the school as part of the application process.

4.2.1. Executive Buy-In

The principal and the school governing body (SGB) needs to be aware of the issues obtaining a tuXlab. An initial capital expenditure will be required from the school to build infrastructure. This will normally have to be authorised by the SGB. This executive also needs to be aware that funds need to be allocated for insurance of the lab and possible future upgrades or maintenance.

Support from the principal and SGB is very important to the ongoing success of the school's tuXlab. Teachers will be more motivated knowing that a level of accountability is in place. Certain tuXlab activities, like open-days, involve the outside community and it is essential that the school executive are involved in taking ownership of such events.

The principal is also ultimately responsible for assigning rolls and responsibilities within their staff around the management of the tuXlab. tuXlab facilitators and the computer committee need to be held accountable by the principal.

4.2.2. Computer Committee

Each school is required to form a computer committee consisting of at least three members of staff. This committee will manage the day-to-day running of the laboratory as well as provide feedback reports to the Foundation. The committee is also expected to formalise new strategies for integrating the laboratory usage into the educational curriculum, create internal support and maintenance resources as well as employ or appoint a facilitator to provide training during school hours. The computer committee is also expected to promote open source software awareness to the entire school community and allow access to the centre after normal school hours.

The computer committee is ultimately the group that is responsible for the running of the tuXlab and its inclusion into the schools operations. They are required to submit a timetable to the tuXlab coordinator outlining how the tuXlab is to be included in class times.

4.2.3. Clustering

The schools are built into clusters during the selection process.

The support process hinges on the schools collaborating within their clusters to try and solve technical problems as well as share best practises around the use and integration of the tuXlabs.

To ensure that there is communication between clustered schools, the representatives of the schools in each cluster are required to form a committee which includes at least two representatives from each school. These cluster committees meet at least once a quarter to share ideas, and experiences and discuss approaches to managing and utilising the tuXlab.

In many cases the first level of support is best provided by a peer group member who has faced similar problems. Not only does this build a co-operation between schools but it also fosters the sharing of skills between administrators and learners. It also serves to remove (or break) the perceived barrier to technology skills: by helping one another, learners and teachers are not merely passive recipients of technology services but are playing an involved role in the process.

4.2.4. Servicing the Community

The tuXlab, although it belongs to the school, is intended to service the local community around the school as well. The school is encouraged to activate the parents, learners and community at large to engage in the use and future of the tuXlab through participation, support and training.

Some specific activities include the following:

1. **Open Days**

   The schools are encouraged (and incentivised) to hold community Open Days at least once a month. The objective is to open the tuXlab up to the greater community and help to increase exposure to its capabilities and opportunities. Activities could include holding short courses (e.g. basic word-processing etc.)

2. **Adult Literacy**

   Each school is urged to implement a programme that will allow community members to receive computer training courses after normal school hours. This program can be implemented to create funds for the salary of a facilitator or to cover the cost of maintenance of the lab or further expansion the lab.

3. **Open ICDL Training and Testing Centres**

   The Open ICDL is the official open source version of the internationally renowned International Computers Driving Licence. It is a certification that covers computer use centred around OpenOffice.org and connectivity.

To date, the tuXlab programme has helped to established Windermere Primary as an official testing and training centre for the OpenICDL. Schools are able to generate funding through the offering of ICDL training courses to the community.

4. **Internet Café**

   Those schools that have an internet connection to the tuXlab are presented with an opportunity to open a community internet café. The lab could be used after school hours and the school could raise funds towards paying for the connection fee and for maintaining the tuXlab.

5. **Computer Clubs**

   Computer clubs within the schools offer a platform for learners, educators and the community to share expertise, experiences and activities around the use of computers. Such clubs should be the hub of the tuXlab where members take ownership of the lab and its future.

4.3. Post Install Management

4.3.1. Reporting

To ensure the successful development and growth of the tuXlab model, accurate and relevant feedback from the schools is essential. The schools are encouraged (and incentivised) to produce reports on its activities including:

- Quarterly reports on the general use of the tuXlab within the school including:
  - how it has been incorporated into the school's timetable
  - what training is provided
  - what activities have been scheduled and held
  - a guideline for the report is available (see Appendix B)
- Curriculum aligned lesson plans, if available, should be submitted to the tuXlab coordinator
- Evaluation forms on training provided to the school should be returned to the tuXlab coordinator
- Activity reports on Open Days, after school training programmes and any event that utilises the tuXlab in a constructive way
- Attendance registers for meetings and workshops.

In essence, these reports help the tuXlab coordinator to track the usage of the tuXlabs in the schools.

The MoU actually makes such reporting a requirement. The guidelines are there for schools that are not sure how to construct the reports. The schools are welcome to submit them in whatever format they desire as long as the criteria are met for required information.

4.3.2. Training

As part of the installation process, the tuXlab team schedules initial whole school training shortly after the hardware is set up. This forms a total of 6 hours on-site training that covers some of the basic skills around using a tuXlab workstation, from logging in to writing simple documents and spreadsheets.

Additional training needs to be managed by the training providers (usually partners to the tuXlab programme, e.g. Computers 4 Kids), the school principal and the tuXlab facilitator.

4.3.3. Incentive Scheme

The incentive scheme is designed to encourage and reward the use of the correct (and required) procedures behind running a tuXlab. Points are assigned to a school for correct usage of the support process, regular reporting and hosting events and meetings. These points accumulate and can eventually be cashed in for printers, extra workstation or a host of other items. The scheme is outlined in the “tuXlab Incentive Scheme” document found both in the Appendix E and online at the TuXlab Webportal [http://www.tuxlab.org.za/files/incentives.pdf].

The incentive scheme servers two main purposes:

- on the one hand it encourages schools to engage in the tuXlab activities, and
- on the other hand it provides a mechanism and a framework for tracking of events and reports from the schools.

The various rewards available are all chosen for relevance to a computer centre, e.g. printers, extra workstations and additional training.

4.3.4. Budget

As with every education and business project there has to be a **budget**. This makes sense when you think of what installing and running a tuXlab entails.

4.3.4.1. Capital Expenditure

The tuXlab model works on the thin-client model using refurbished computers and open source software. As such it is an extremely cost effective solution.

Traditionally, the school has been responsible for providing the physical infrastructure for

the venue. This includes, as mentioned above, the security, desktops and electrical work.

The actual computer hardware is then paid for by a donor. The server and switch are procured new so that a decent hardware warranty is in place. The workstations are refurbished machines and are easy to replace should they fail.

Currently, at the middle of 2006, the cost breakdown for a tuXlab is as follows:

- Xeon Server: R14200
- 24 Port Switch: R950
- Power Cables x 20: R500
- Network Cards/Boot PROMs x 20: R1700
- Network cabling, boots and jacks: R500
- tuXlab Starter Kit: R200
- **Total: R18050**

The initial training that the school receives can be broken into 6 hours at R50/hour yielding R300 per school. Additional costs contributing to the total cost of a tuXlab would include staff costs of the tuXlab team, the running of a support desk and other external activities.

4.3.4.2. Sustainability

The SGB needs to be aware that there are running costs with regards to maintaining a working computer lab. Some of the items include:

1. making provision for a computer facilitator
   - can be a staff member
   - or external to the school
2. hardware maintenance
   - the school needs to ensure that maintenance options are available for the

     server and switch past their distributor warranty

     workstations past their 3 month tuXlab warranty
   - maintenance options including:
     - employing a service provider to maintain the equipment
     - entering a lease agreement with a computer financial services organisation (e.g. RentWorks)
     - sourcing maintenance or refurbished workstations through a local refurbishment

house (e.g. FreeCom)

Further budgetary considerations include additional training for the staff, sending facilitators on workshops or courses and purchasing peripheral equipment to help in class delivery (e.g. data projectors).

Hidden costs could include electricity, transport to training sessions, paper and printer toner. Internet connectivity is another area that a school should consider in its budget.

Please read the Sustainability Module that is also part of the tuXlab Cookbook.

4.3.5. Programme Support

You do not need to be alone when considering or utilising a tuXlab. Support for you and for your tuXlab programme is available from various sources. We discuss two of these sources here:

4.3.5.1. Department of Education

The educational sector is a difficult ground to operate in. The level of ICT competence in under-privileged communities is quite low and educators are challenged enough with current tasks to engage in furthering their computer skills.

The support of the Department of Education (DoE) is thus essential. A well placed letter can literally make a world of difference. For example, if attendance for a workshop at a school is critical to the success of a certain programme, then a letter of support from a DoE head can ensure that full attendance is attained. DoE buy-in to the programme lends it credibility.

Similarly, software, content and applications that carry a stamp of approval from the curriculum development department are more likely to be engaged by the teachers than those that aren't. If a software programme targets specific learning areas and outcomes that the DoE has identified, then the educators' task becomes a lot clearer.

4.3.5.2. Partnerships

Partnerships can include content providers, trainers, donors, sponsors or even various levels of consulting. The tuXlab definition of a partners is one in which a 3rd party supports the tuXlab programme to the extent that products or services are granted either for free or for a reduced cost to the beneficiaries.

- An example would be Edupac whose school management application is to released under an open source licence and made freely available. Additionally, Edupac agreed to install, train and support their product in as many tuXlab schools as existed at no cost. The sustainable future would see a basic reduced support cost model for the schools.

- Another example is Computers 4 Kids. Their DoE approved software was ported to the linux operating system and provided free of charge to all tuXlabs. Although not technically an open source product, the partnership is a good example of a collaborative relationship with a proprietary software provider that makes their product available at no cost for the programme. Additionally they provide training and support to the schools at no cost.

Other examples include:

- telecommunications service providers reducing their rates for public schools
- content developers releasing their material under an open licence
- large corporation donating equipment and providing media exposure
- the media themselves keeping track of the programme

All these examples are partnerships that aid the beneficiaries of tuXlabs directly.

Some guidelines should be kept in mind when dealing with social enterprises though:

- There are organisations looking only for mileage out of the ttuXlab programme and not fronting anything tangible. They generally are aiming for a “tuXlab Partner” logo by providing a product or service that does not honour the open spirit of the community or the programme. Generally such organisations are looking for the tuXlab team to coordinate a targeted marketing strategy for them. Such organisations are generally dealt with by explaining that they are welcome to approach schools directly.
- Other partner organisations might not have the capacity to provide a social enterprise offering. The expectation is often there for the tuXlab team to produce the service for them in, for example, installing their software or supporting their product. This is not always feasible and the caveat should be there when drafting a partnership agreement.
- In drafting a partnership agreement, it should be kept clear and concise what roles, responsibilities and activities lie with which parties. In light of the above, partner organisations can have a tendency to expect the tuXlab team to manage and operationalise their activities for them because of the cost sacrifice they might be making.
- Accurate reporting from partners should be maintained. It is important to receive feedback from all aspects of the programme, especially around areas of support. A partners support desk should integrate into the tuXlab support desk, at the very least on a communicative level.
- With respect to reporting, where applicable the tuXlabs support desk should be preserved. In other words, a partner organisation should preferably integrate their help desk with tuXlabs in such a manner that both operations know where to dispatch support tickets and can track problems accordingly.

5

Software Module

The TuXlab Software Module covers the specific software that is used within a TuXlab. The first section reviews the topic of "Open Source Software" that is covered within the TuXlab Cookbook Introduction Module. The second section presents the information regarding the software components used; including the LTSP classroom server, the Wizzy server, specific applications (OpenOffice.org, Mozilla, educational software, wikipedia, and edutainment software), Project Gutenburg, Connexions, and other resources. The final section deals with the process of installing the software, including general information about installing a Linux distribution.

5.1. Software Introduction

As mentioned in the Introductory book to the tuXlab Cookbook, the tuXlab project uses Open Source software wherever possible. The exceptions are only where no suitable Open Source alternative could be found. Currently, the only proprietary software content used in tuXlabs is that which is added on in specific projects, allowing for localised content or specific school management software. All the software that is part of the tuXlab falls under Open Source licenses, and the software itself will always be 100% free.

We have also already seen that Open Source software is used in the tuXlab project because it overcomes some of the barriers caused by proprietary (also often referred to as “closed source”) software. These barriers included:

- Limitations on the distribution of the software
- Prohibitively expensive licensing fees
- Limitations on customising and localising the software

The tuXlab project would simply not be possible without the use of Open Source software. We believe that Open Source software will continue to mature at an ever increasing rate, and that more and more people will find using this liberating form of technology more pleasant than the traditional, proprietary licensed counterparts.

5.2. Software components

A tuXlab includes both open source software as well as office tools and web browsers (see Section 5.2.3, “Applications” [5–5]). At its core, however, it consists of the server software that runs the classroom server and the Wizzy internet proxy server.

5.2.1. LTSP classroom server

A tuXlab classroom server requires a Linux distribution with the option to install the LTSP packages.

5.2.2. Wizzy server

A Wizzy server can be used to provide a managed internet access solution for a tuXlab, also known as a *gateway* server. Currently, a Wizzy server is an extra to a tuXlab install. The Shuttleworth Foundation may award a Wizzy server computer to a school with a tuXlab if the school demonstrates that it is making good use of the lab. Wizzy is a project of Andy Rabagliati, a techno-philanthropist who has been wiring rural schools for cheap internet connectivity for many years now.

The software is based on RedHat Linux, currently version 8.0, whereas K12LTSP is based on RedHat Fedora. Wizzy uses an adapted version of Whitebox Linux [http://www.whiteboxlinux.org/], itself a straight recompile from source RPMs of RedHat Enterprise Linux [http://www.redhat.com/software/rhel/]. Many extra packages have been added, all under the GPL license.

It performs the duties of dialup manager, firewall, mailserver , web cache, and authentication server. The Wizzy server does normal dialup via a telephone line, but it can also do ADSL and wireless connections. It would be possible to do all these things on the classroom LTSP server as well, but a custom mail arrangement would still be necessary: while the tuXlab is not connected to the internet, mail needs to be delivered to a remote server. Another reason to separate the internet gateway from the classroom server is that it is not a good idea for a classroom server to have a public-facing IP address. All machines exposed to the internet are vulnerable to hackers and new virus codes, and the classroom server contains all the user data that could be lost if it were compromised.

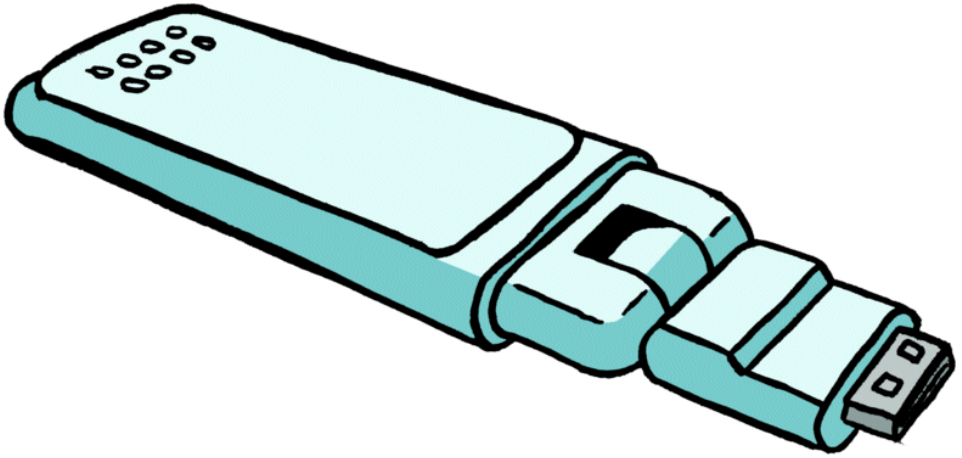
The way in which the Wizzy server sends and receives email is customised to work with an intermittent internet connection. It utilises a protocol that was more commonly used in the days before permanent internet connections became common, namely UUCP (the Unix-to-Unix Copy Protocol). UUCP is suited to queueing data to be copied to remote hosts for delivery when a network connection is established.

Wizzy has the goal of providing all its services via UUCP in order to accommodate high latency connections where there may be long disconnected periods. This type of connection includes situations where a connection is established by dialling up at night when phone rates are low, as well as situations without any dialup access at all.

In cases without dialup access, the connection can be made by carrying the data across by hand! [1] The transfer medium may be, for example, a USB memory stick, a mobile wireless-enabled laptop, or a USB hard drive. The transfer medium will contain newly fetched data and the queued requests for new web pages, as well as mail waiting to be

[1] See http://wizzy.org.za/article/articleview/18/1/3

sent. This is not a stated goal of the tuXlabs project, so if your school wishes to pursue this route, you may have to contact Wizzy to work out the details.



A wireless antenna that plugs into a USB plug on a laptop or a PC.

If you do have access to a telephone line for the tuXlab, the internet is dialled up only at night when the rates are low (Telkom South Africa provides a particular phone account that allows a single 12 hour phone call for R7). In this period, approximately 250MB of email and web material can be downloaded onto the Wizzy server in the classroom. In the morning, after the phone has been hung up, the learners can access the downloaded material as though they had a direct web connection. Their experience of email and web during the school day is almost identical to a very fast online web experience, with the exception that newly requested pages will only be available by the next morning.[2]

Normally, if you want to connect to the internet, you need a telephone line and a modem, a satellite uplink, or some other means of tapping into 'the Net'. Unfortunately, in South Africa, the necessary telecommunications infrastructure is not always there to tap into. Around a third of South Africans don't have a phone line, and most schools do not have internet access (e.g roughly 88% of schools in the Northern Province lack an internet connection). [3]

5.2.2.1. Benefits of a Wizzy server

Every student can have email and web access without the cost of being online. This is true even for schools that don't have a telephone.

While the delay in retrieving web pages is a necessary consequence of the Wizzy solution, it has a desirable side effect in that it helps educators to manage web access. Since anybody with an internet connection can publish on the Web, there is an absolute ocean of

[2] See http://wizzy.org.za/article/articleview/4/1/3
[3] See http://wizzy.org.za/article/articleview/18/1/3

complete trash that threatens to engulf the unwary browser. Groping through this mess just wastes school resources. Automated solutions to this issue have been proposed, often by companies that market filtering products. Apart from the fact that these products suffer from both false positives and negatives, blocking material unnecessarily and letting undesirable material through, it shifts responsibility from the teacher to a software product.

The Wizzy solution believes that it's better to let the teacher decide what is appropriate. This also allows the teacher to stay in touch with the material accessed by the learners he or she is teaching. Teachers can explicitly choose the web material that learners can access and definitively reject web material that is not appropriate. They can also set the Wizzy server to monitor websites, retrieving new versions as they become available.

Because the school is never in direct connection with the internet, but rather has its web content delivered upon request, students cannot "surf" the internet. However, since all the web sites that have been requested by the IT administrator of the school are delivered and stored on the school's server, the students can "surf" this local web cache as though it were a miniature internet.

These specially chosen web sites can easily number in the hundreds. They can be updated daily; and should a new site be desired on a regular basis, the school would have to wait only one day for the delivery to begin. The "surfing" of this miniature internet within the tuXlab costs nothing, and is fast enough to give you whiplash, since all communication takes place within the tuXlab, between the server and the client computers. [4]

tuXlabs aims to give educators the tools to help learners explore computers and the Wizzy server allows them to put reasonable limits on learners' access, guiding them towards independent use of the computer as they mature. tuXlabs does not make any judgement regarding what content is regarded as appropriate, but is there to help provide the tools and documentation to help educators with these decisions. The focus is more on the "guidance" aspect, rather than restriction, as the former is a positive activity while the latter is negative.

The Wizzy solution integrates seamlessly with existing labs. Normally, no additional equipment is required to have this kind of internet access.

5.2.2.2. Batch mail delivery solutions

First, a recap of the basics. [5] How does mail from far away get to your account, say: `somebody@myschool.wizzy.org.za`?

- The remote mail client probably punts the mail to a smarter host - the SMTP (Simple Mail Transfer Protocol) server of their ISP (Internet Service Provider). This will have been configured by the sender when first setting up their mail client.

[4] See http://wizzy.org.za/article/articlestatic/19/1/2/
[5] See http://wizzy.com/wizzy/mail.html

- Their ISP mail server will ask a local nameserver (i.e. DNS server) for the MX (Mail Exchange) records of `myschool.wizzy.org.za`. These are the IP addresses of machines listed as handling mail for this domain. The ISP will try to deliver directly to the lowest-numbered MX record in the list, via SMTP. If it cannot connect, it will try other MX hosts, usually in ascending order.

  The DNS service for the tuXlab and Wizzy networks is provided by SchoolNet [http://www.schoolnet.org.za/]

- With this step, or via intermediate MX hosts, the mail will eventually land on the target host - the lowest-numbered MX, often your local ISP. Most of the time, your ISP will deliver all the mail for that domain into a single mailbox, and leave you to figure out the rest.

- The reason why the ISP can't easily do more is due to the fact that SMTP is not authenticated to specific users, and therefore relies on name services (fixed IP addresses) to route messages to their destination. This makes it impractical for intermittently connected machines, that dial up nightly and are dynamically assigned an IP address by the ISP, to function as mail exchangers.

  One way to preserve the SMTP protocol across dynamic IP addresses is to batch SMTP commands in a file, and pass the file over a separate transport. [6] Batched SMTP is similar to normal SMTP, but, because it is being written to a file, all responses are ignored. (Normally, an SMTP connection is a conversation of commands and responses between the sending and the receiving servers, so that the sender may probe for the capabilities of the receiver, confirm successful delivery, etc.) Fortunately, this is not critical: as long as all the responses are known to be `OK`, everything works.

  The Wizzy solution is to write messages to a file, use UUCP over TCP/IP as an authenticated transport from the SMTP server of the ISP (in this situation, Andy's Cape Town Wizzy server will be the SMTP), and unpack the batch at the final destination (the tuXlab). The batched SMTP format is described in RFC 2442 --- The Batch SMTP Media Type [http://www.faqs.org/rfcs/rfc2442.html]

  Wizzy uses **exim** as the SMTP agent at both ends, and Taylor (GNU) UUCP as the middle transport.

5.2.3. Applications

5.2.3.1. OpenOffice.org

OpenOffice is a full office suite, intended to measure up to and surpass Microsoft Office. This is what its original author, Marco BÃ¶rries, intended when he started StarDivision to create the software that would eventually become OpenOffice in 1984, when he was just sixteen. He called it StarOffice. By the time Sun Microsystems bought Marco's company in 1999, over 25 million copies of Star Office had been sold to customers who needed platform independence and an alternative to Microsoft.

[6] See http://wizzy.com/wizzy/batch.html

In July 2000, Sun released most of the Star Office source code (about 7.5 million lines of C++) to the stewardship of the open source community, under the Free Software Foundation [http://www.fsf.org]'s LGPL license. The community project has as its goal: "To create, as a community, the leading international office suite that will run on all major platforms and provide access to all functionality and data through open-component based APIs and an XML-based file format." The open APIs and file formats are turning OpenOffice into a platform in its own right, supporting projects such as OpenGroupware [http://opengroupware.org/], which aims to provide an open alternative to Microsoft's Exchange and SharePoint products.

OpenOffice includes a word processor, spreadsheet, interfaces to many databases, a presentation builder, and a diagramming tool. It is now able to decode most variants of Microsoft Office document formats, but although it's StarBasic macro language is syntactically identical to Visual Basic, OpenOffice cannot execute Visual Basic scripts. Although these are preserved upon conversion, the scripts need to be adjusted to use the OpenOffice API before they can be used.

OpenOffice can be used to teach all the basic computer skills required to enter the job market, and as the design paradigms of the software is very close to Microsoft Office, the skills learnt are readily transferable.

5.2.3.2. Mozilla

In 1998, Netscape Communications released the source code of its Navigator web browser software as open source [http://www.catb.org/%7Eesr/writings/cathedral-bazaar/cathedral-bazaar/ar01s13.html], and the Mozilla project was born. Netscape released their software as open source in order to compete with Microsoft, who were bundling Internet Explorer with every copy of Windows sold. Netscape hoped that the cooperation of thousands of developers around the world would create a better product than Internet Explorer. It did, but it took years, and Netscape fell by the wayside. The code lived on, though.

In 2004, six years after the start of the project, the non-profit Mozilla Foundation that was created to coordinate the project finally released version 1.0 of the Firefox browser. Earlier versions of the software had been in use for years, but at last it was deemed ready for a high-profile release to the general public.

The full Mozilla suite includes far more than the web browser. It includes Thunderbird, a mail client with address books and calendaring support, and Composer, a web page editor. For developers, it includes Venkman, a Javascript debugger, and the DOM Inspector, a wonderful tool for interactively exploring the enormous, intricate internal structure of complicated web pages.

The code base has been painstakingly restructured to get as much use out of it as possible. So, for instance, all the Mozilla applications use the same rendering engine, called Gecko, to layout and display HTML pages on screen. This is critical, since it is extremely hard

work to implement the standards that govern web page structure and display correctly. Mozilla has the best support for the HTML and CSS standards of any browser out there. In the past, Microsoft has used Internet Explorer's idiosyncratic and incomplete implementation of web standards to coerce people to craft their web pages to look good in Internet Explorer at the expense of other browsers. For the web to become a dependable platform, however, developers have to be able to build on solid public standards that don't leave them at the mercy of any company's prosperity.

As with OpenOffice, the Mozilla project is becoming a platform for extensions [https://addons.update.mozilla.org/extensions/] from the community.

5.2.3.2.1. Security

As the internet has become more popular, viruses, spam, spyware and trojans have grown to be an enormous problem. These are all caused by programs that are received and executed on computers without the knowledge or consent of their users. How can this happen?

In general, there are two ways for this to happen:

- You may have given implicit permission without intending to do so;
- The unwanted program may be exploiting errors or bugs in some program in order to insinuate themselves into it, so that it can execute with the permissions of the original program.

Firefox and Thunderbird try to protect you as far as possible by setting up reasonable defaults. Nothing in an email message is executed unless you explicitly run it yourself. And execution of Javascript in web pages can be switched off at any point. Firefox also allows you to grant or revoke specific permissions (e.g. opening popups, hiding the status bar, or changing images) to Javascript.

The Mozilla suite also affords you a measure of protection from the second case. While Internet Explorer has access to practically the entire running system of a Windows PC via a powerful integration technology called ActiveX, Firefox and Thunderbird are far more restricted. It also helps to be running on Linux, where the user of the web browser will generally not be able to damage the operating system, and the system administration account is never used to run user applications.

5.2.3.3. Educational Software

There are many educational products and applications available for use on computers. The tuXlab project utilises the most comprehensive Open Source applications and specific locally produced software. This section will present an overview of this software:

5.2.3.3.1. Computers 4 Kids

The Shuttleworth Foundation and Computers 4 Kids entered into a partnership in 2004, with an agreement that C4K would supply 500 tuXlab schools with materials and software at no cost to the Foundation and schools, excluding the printing costs of such materials. Training and support is available and provided in all major areas, with support and assistance from the Computers for Kids franchisees within these areas. The Computers 4 Kids curriculum has been approved in three major regions by the Department of Education. Computers 4 kids have not applied for approval in other regions as yet. Although Computers 4 Kids curriculum can run on a linux platform, and it is included in the installation at each school, it is not truly open source, this is accepted because there are no open source equivalents at the moment. Relationships such as these need to be built and maintained, they are not guaranteed within the replication model, this is something that needs to be done by replicator. Partnerships such as these add a lot of value to the tuXlab model and assist in the and usability of the lab within the day to day use of the lab.

With the advancements in information technology, many schools are struggling to meet their own ICT needs, let alone equip their students with the critical computer skills that they will need in a work environment. Computers 4 Kids, specialise in providing a unique Integrated OBE ICT Curriculum, educational software as well as all the backup, support and in-service training to ensure your ICT centre works for you! Their integrated ICT curiculum works on both Microsoft as well as Open Source platforms.

5.2.3.3.1.1. About Computers 4 Kids

Computers 4 Kids is a Southern African computer education company run by a team of fully qualified educators. It was established in 1995 in order to address the critical need for Information and Communication Technology (ICT) training at school level. Computers 4 Kids is committed to the development and use of South African activities in Information and Communications Centres and we are able to offer both the expertise and the equipment needed in the most cost effective possible manner.

Computers 4 Kids establish and administer new and existing Computer Centres at schools by supplying Information and Communication Technology educational software, a relevant and integrated OBE curriculum and the necessary backup and support.

All curriculum materials can be used in a Microsoft as well as Linux environment.

By staying at the cutting edge of technological and educational trends, Computers 4 Kids will remain the leading supplier of integrated ICT solutions to the educational institutions of Southern Africa, empowering learners and staff with the most appropriate hardware, software, service and skills.

With information technology moving faster and faster, and becoming more and more important, many schools find it difficult to meet their own ICT needs, let alone fulfill their obligation towards equipping learners with the computer skills they will need in almost any career. Learning focuses, themes and topics have been selected from what is currently being studied in Southern African schools.

Computers 4 Kids has a strong curriculum development team which consists of experienced educators in both the educational and ICT fields. All developers have had extensive experience in teaching ICT skills using an integrated approach. We have used this experience in the field of ICT education to develop a dynamic and relevant integrated ICT curriculum which is user friendly and easy to implement.

The curriculum was also designed for use by class educators who teach ICT to their own classes. The Computers 4 Kids Integrated ICT Curriculum will ease the task of educators by providing relevant integrated ICT lessons which develop cognitive as well as ICT skills with the emphasis on using the computer to learn.

5.2.3.3.2. tuXlab Curriculum

The tuXlab curriculum consists of 160 Integrated ICT Learning Units for children aged 4 - 16. There are 20 learning units in each of the Volumes from R – 8. tuXlab is used on a Linux platform and makes use of open source software, namely: OpenOffice.org (Writer, Calc, Impress and Draw), Tux Paint and FreeMind. Some of the templates utilise Flash for which you will need a Flash player or the Flash plug-in for your browser.

The tuXlab curriculum consists of many resources for both learner and educator.

Learner resources include:

- Educational Games
- Keyboard and Mouse Skills (younger learners)
- Touch Typing Skills (older learners)
- Templates (partially completed documents which the learners modify and complete)
- Pix Linx (a library of relevant pictures where applicable)
- Resource Linx (any additional supporting information used for the Integrated Activity)
- Tech Talk Dictionary (a quick and easy reference to computer terminology)
- Website Links (saved web pages and online links to relevant websites)
- Self Assessment form (for learners to evaluate their own progress)
- Peer Assessment form (for learners to evaluate their peer's progress)

Educator resources include

- Assessments and Feedback forms
- Classroom Posters
- Planning and Organising resources
- Reference Documents

- Useful Software

5.2.3.4. Wikipedia

- At present, Wikipedia is included in the installation phase of a tuXlab, it is downloaded and placed into the cache on the tuXlab server. It has been necessary to do this in schools who do not have Internet connectivity.
- Wikipedia adds educational value to the tuXlab model, and especially valuable within a lab that does not have connectivity as learners and educators have access to a resources other wise not available.
- Downloading and storing on the server means the school can track and monitor what information is available to learners, and remove any inappropriate materials.
- We do a server upgrade once a year, at the same time we do updates to wikipedia if the school has Internet connectivity, Wikipedia can still be cached, and upgrades done on more regular basis

Wikipedia (pronounced as either [#wi#ki#pidi.#] or [#w#ki-], also [-#]) is a multilingual Web-based free-content encyclopedia. It is written collaboratively by volunteers with wiki software, which allows articles to be added or changed by nearly anyone. The project began on January 15, 2001 as a complement to the expert-written Nupedia, and is now operated by the non-profit Wikimedia Foundation. The English-language version of Wikipedia currently has more than 810,000 articles. Wikipedia has steadily risen in popularity, and has spawned several sister projects, such as Wiktionary, Wikibooks, and Wikinews.

Articles in Wikipedia are regularly cited by the mass media and academia, who praise it for its free distribution, editing, and diverse range of coverage. Editors are encouraged to uphold a policy of "neutral point of view" under which notable perspectives are summarized without an attempt to determine an objective truth. But Wikipedia's status as a reference work has been controversial. Its open nature allows vandalism, inaccuracy, and opinion. It has also been criticised for systemic bias, preference of consensus to credentials, and a perceived lack of accountability and authority when compared with traditional encyclopedias.

There are about 200 language editions of Wikipedia (about 100 of which are active). Ten editions have more than 50,000 articles each: English, German, French, Japanese, Polish, Italian, Swedish, Dutch, Portuguese, and Spanish. Its German-language edition has been distributed on compact discs, and many of its other editions are mirrored or have been forked by websites.

5.2.3.4.1. Free Content

The GNU Free Documentation License (GFDL), the license under which Wikipedia's articles are made available, is one of many "copyleft" copyright licenses that permit the

redistribution, creation of derivative works, and commercial use of content provided its authors are attributed and this content remains available under the GFDL. When an author contributes original material to the project, the copyright over it is retained with them, but they agree to make the work available under the GFDL. Material on Wikipedia may thus be distributed to, or incorporated from, resources which also use this license.

5.2.3.5. Edutainment

For the purposes of this document, we have only included what is on Edubuntu (distro installed in tuXlabs), there are more available and we will be adding these on in the future.

5.2.3.5.1. Kalzium

Kalzium

Version 1.4-pre3

© 2001-2005 by the Kalzium Team

Kalzium is an application which will show you some information about the periodic system of the elements. Therefore you could use it as an information database.

Kalzium Web Site [http://edu.kde.org/kalzium/]

5.2.3.5.2. KBruch

KBruch

Version 3.4

© 2002-2005 Sebastian Stein

KBruch is a small program to practice calculating with fractions. There are four different exercises offered.

1. Exercise Fraction Task - in this exercise you have to solve a given fraction task. You have to enter numerator and denominator. This is the main exercise. The difficulty of this task can be influenced by the user. The user can decide if he wants to solve tasks with addition/subtraction and/or multiplication/division. Also he can set the number of fractions and the maximum size of the main denominator.
2. Exercise Comparison - in this exercise you have to compare the size of 2 given fractions.

3. Exercise Conversion - in this exercise you have to convert a given number into a fraction.
4. Exercise Factorization - in this exercise you have to factorize a given number into its prime factors. Factorization is important while finding the main denominator of 2 fractions.

KBruch provides a handbook describing the different exercises and the general usage.

KBruch Web Site [http://edu.kde.org/kbruch/]

5.2.3.5.3. KHangMan

KHangMan

Version 1.5

© 2001-2005 Anne-Marie Mahfouf

KHangman is the classical hangman game. The child should guess a word letter by letter. At each miss, the picture of a hangman appears. After 10 tries, if the word is not guessed, the game is over and the answer is displayed.

he words are nouns and available in **twenty-four languages** at the moment: Czech, Brazilian Portuguese, Bulgarian, Catalan, Danish, Dutch, English, Finnish, French, German, Hungarian, Irish (Gaelic), Italian, Norwegian (Bokmål), Norwegian (Nynorsk), Polish, Portuguese, Spanish , Slovenian, Serbian Latin , Serbian Cyrillic, Slovak, Swedish, Tajik and Turkish. The program will detect which languages are present and enable them. In KDE 3.2 you have all the languages but in KDE 3.3.x and in the next KDE 3.4, you will get only English plus the data in the language you use in kde provided it exists. You will also be able to easily download other languages via the Get New Stuff dialog (3 clicks and your data will be installed).

There are 4 levels per language: easy, medium, hard and animals which contains only animals nouns.

KHangMan Web Site [http://edu.kde.org/khangman/]

5.2.3.5.4. KLettres

KLettres

Version 1.3

© 2001-2005 Anne-Marie Mahfouf

KLettres aims to help to learn the alphabet and then to read some syllables in different languages. It is meant to help learning the very first sounds of a new language, for children or for adults.

Currently available are: Czech, Danish, Dutch, English, French, Italian, Luganda, Romanized Hindi, Spanish and Slovak, you can choose using the Languages menu.

KLettres has four levels which can be changed via a Levels drop down box or the Level menu:

- **Level 1:** the letter is displayed and the user hears it.
- **Level 2:** the letter is not displayed, the user only hears it.
- **Level 3:** the syllable is displayed and the user hears it.
- **Level 4:** the syllable is not displayed, the user only hears it.

You can also choose the mode(kid or grown-up) via the colored buttons on the toolbar and KLettres will keep your settings and restore them the next time you play.

KLettres Web Site [http://edu.kde.org/klettres/]

5.2.3.5.5. KmPlot


KmPlot

Version 1.2.0

© 2001-2005 Klaus-Dieter Möller

KmPlot is a mathematical function plotter for the KDE-Desktop. It has built in a powerful parser. You can plot different functions simultaneously and combine their function terms to build new functions. KmPlot supports functions with parameters and functions in polar coordinates. Several grid modes are possible. Plots may be printed with high precision in correct scale.

Features:

- powerful mathematical parser
- precise metric printing
- different plot types (functions, parametric, polar)

- highly configurable visual settings (plot line, axes, grid)
- export to bitmap format (BMP and PNG) and scalable vector graphics (SVG)
- save/load complete session in readable xml format
- trace mode: cross hair following plot, coordinates shown in the status bar
- support zooming
- ability to draw the 1st and 2nd derivative and the integral of a plot function
- support user defined constants and parameter values
- various tools for plot functions: find minimum/maximum point, get y-value and draw the area between the function and the y-axis

KmPlot Web Site [http://edu.kde.org/kmplot/]

5.2.3.5.6. KPercentage

KPercentage

Version 1.2

© 2001-2005 Matthias Meßmer

KPercentage is a small math application that will help pupils to improve their skills in calculating percentages.

KPercentage Web Site [http://edu.kde.org/kpercentage/]

5.2.3.5.7. KStars

Version 1.1-p1

© 2001-2005 the KStars Team

KStars is a Desktop Planetarium for KDE. It provides an accurate graphical simulation of the night sky, from any location on Earth, at any date and time. The display includes 130,000 stars, 13,000 deep-sky objects,all 8 planets, the Sun and Moon, and thousands of comets and asteroids.

KStars Web Site [http://edu.kde.org/kstars/]

5.2.3.5.8. KTouch


KTouch

Version 1.5

© 2000-2005 Haavard Froeiland and Andreas Nicolai

KTouch is a program for learning touch typing. KTouch is a way to learn to type on a keyboard quickly and correctly. Every finger has its place on the keyboard with associated keys to press. KTouch helps you learn to touch typing by providing you with something to write. KTouch can also help you to remember what fingers to use.

KTouch Web Site [http://edu.kde.org/ktouch/]

5.2.3.5.9. KTurtle


KTurtle

Version 0.1

© 2003-2005 Cies Breijs

KTurtle is a Logo programming language interpreter for KDE. The Logo programming language is very easy and thus it can be used by young children. A unique quality of Logo is that the commands or instructions can be translated (please see the translation how to [http://edu.kde.org/kturtle/translation.php] if you want to help in your own language), so the 'programmer' can program in his or her native language. This makes Logo ideal for teaching kids the basics of programming, mathematics and geometry. One of the reasons many children like Logo is because of the turtle, a programmable icon which can be moved around the screen with simple commands and can be programmed to draw objects.

KTurtle Web Site [http://edu.kde.org/kturtle/]

5.2.3.5.10. GCompris

GCompris

Version

© 2003-2005

GCompris is an educational software which proposes miscellaneous activities to kids from 2 to 10.

Some activities are game oriented, but always educational. You will find activities in the following topics:

- computer discovery: keyboard, mouse, different mouse gesture, ...
- algebra: table memory, enumeration, double entry table, mirror image, ...
- science: the canal lock, the water cycle, the submarine, ...
- geography: place the country on the map

- games: chess, memory, ...
- reading: reading practice
- other: learn to tell time, puzzle of famous paintings, vector drawing, ...

All in all, gcompris proposes more than 50 activities and it continues to evolve. gcompris is a free software, you have the possibility to adapt it to your needs or to improve it, and, why not, to share your work with the kids of all the world.

GCompris Web Site [http://gcompris.net/]

5.2.4. Project Gutenberg

Project Gutenberg [http://www.gutenberg.org/] was started in 1971 by Michael Hart, with the goal to provide books electronically at no cost.

Their collection numbers more than 13,000 e-books, produced by hundreds of volunteers. Most of the Project Gutenberg e-books (but not all) are older literary works that are in the public domain in the United States. [7] All may be freely downloaded and read, and redistributed for non-commercial use (for complete details, see the license page [http://www.gutenberg.org/license]).

[7] Incidentally, this emphasises the importance of a public domain. Current legislation in the United States effectively allows work to be kept out of the public domain indefinitely. This is bad news for future generations.

Project Gutenberg Web Site [http://www.gutenberg.org/wiki/Main_Page]

5.2.5. Connexions

The Connexions [http://cnx.rice.edu/] project of Rice University [http://www.rice.edu] is an endeavour to create open source course material. On their homepage, they state: "Knowledge should be free, open, and shared. Connexions is a rapidly growing collection of free scholarly materials and a powerful set of free software tools to help authors publish and collaborate, instructors rapidly build and share custom courses, and learners to explore the links among concepts, courses, and disciplines.

They provide small "knowledge chunks", called modules, that connect into courses. Thanks to an open license, anyone can take these materials, adapt them to meet their needs, and contribute them back to the Commons.

Connexions Web Site [http://cnx.org/]

5.2.6. Resources

There are numerous sources of educational software on the web. Some starting places are:

- SchoolForge [http://www.schoolforge.net/]. SchoolForge's mission is to unify independent organisations that advocate, use, and develop open resources for primary and secondary education.
- Edu-SLUG [http://eduslug.sourceforge.net/index.php], the workspace of the Schools Linux User Group, for the creation of educational software for South African schools.
- You may find an informal collection of links loosely related to the tuXlab project at del.icio.us [http://del.icio.us/tag/tuxlab], where you may register and post your own extensions to the list.

5.3. Software Installation

5.3.1. Planning

Before you start a tuXlab software installation, you need to think about and consider how you will be using the lab.

Questions you can ask yourself:

- How many users will be using this system?
- How much disk will the users need?
- Will this server also function as a web, file, or mail server?
- Will this server act as an Internet gateway?

These factors will determine how you partition your disk, how authentication will work, how many network cards you install on the server, etc. If it's your first time installing a server, and you're unsure of advanced setup details, choose the default settings, which will give you good, generic settings.

Document every part of your setup, including the specifications of the server, how you set up your disk space layout, and what the intended use for the server is. You should make it as easy as possible for another person to work on the server and know what the server is doing.

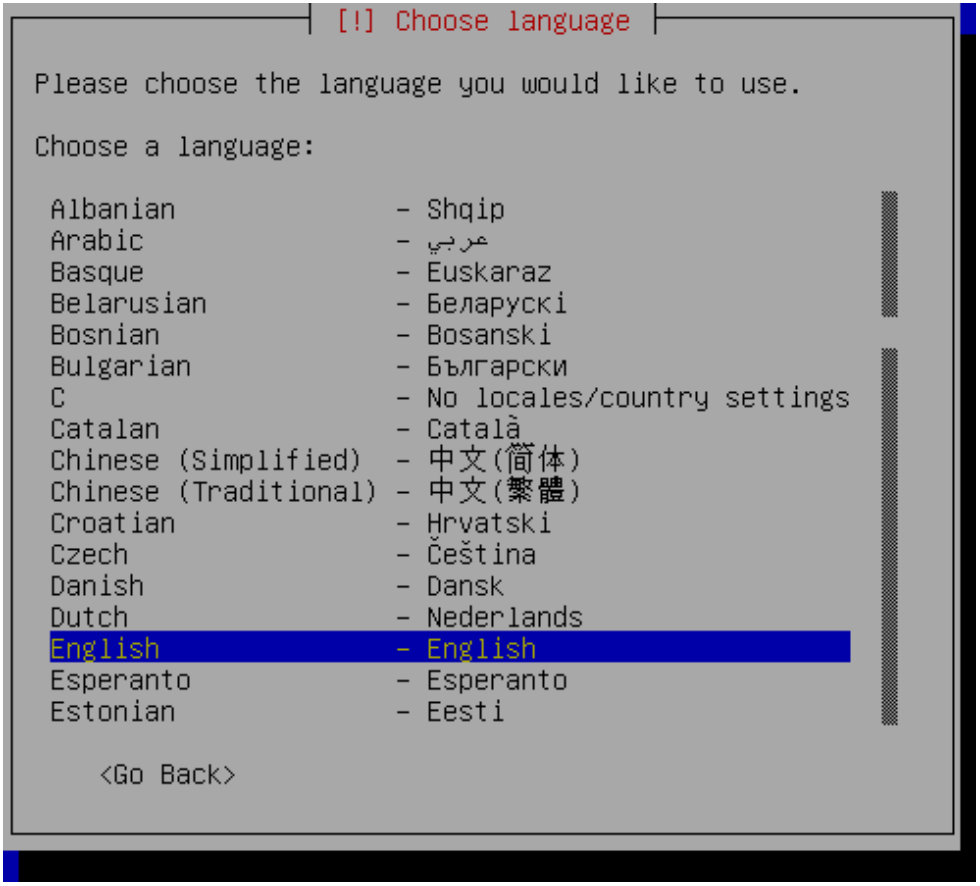
5.3.2. Installing Linux

After reviewing the initial server documentation, you can start the Linux installation of your server. Depending on which education aware distribution of Linux you have chosen

(we profiled four in the Introduction Module) you may need to refer to their particular installation procedure documentation. What we present here is a basic installation sequence that can be used as an example or comparison with the installation of the Linux distribution of your choice.

To start the installation of you Linux distribution insert the disc into your CD-drive and restart the computer. You will be presented with a welcome screen. If you do require more help on starting up there is usually an on screen command that will bring you more options (e.g. pressing F2). We recommend that you simply press the ENTER key to begin the installation process.

Next, choose your Language. (Remember that this is a sample installation and your particular distribution of Linux may look very different)

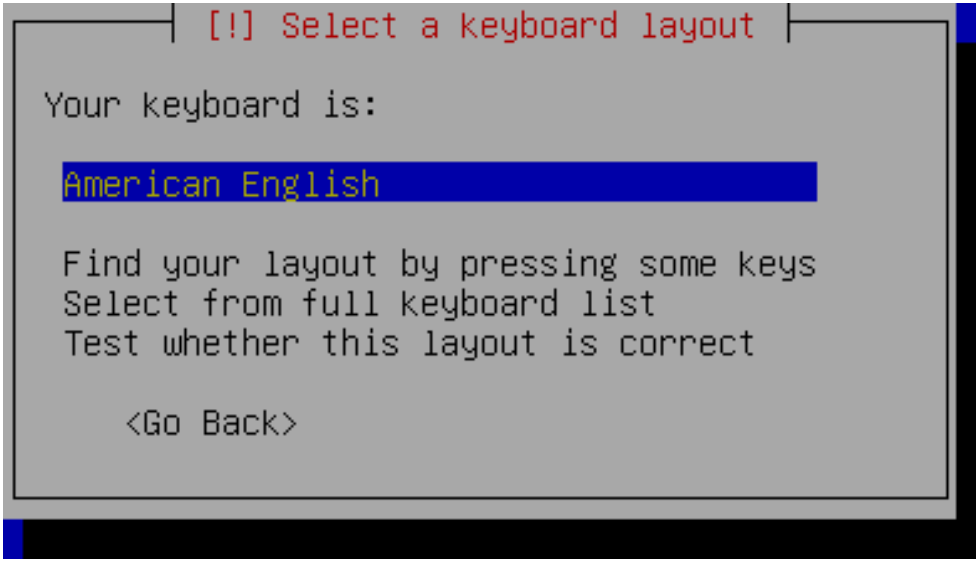


Language Chooser

Select your location:

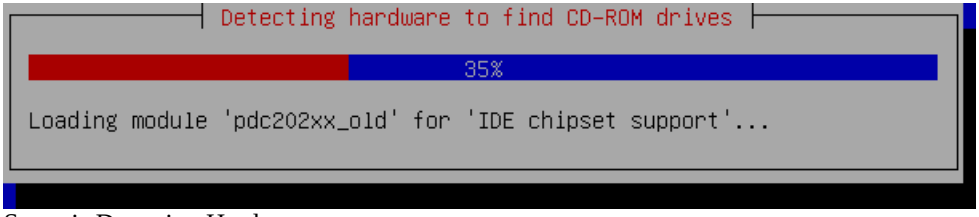
Country Chooser

Select your keyboard layout. Note that your keyboard layout might be different that the language you speak. For instance, you might be using a US English Keyboard layout, while you speak UK or International English.



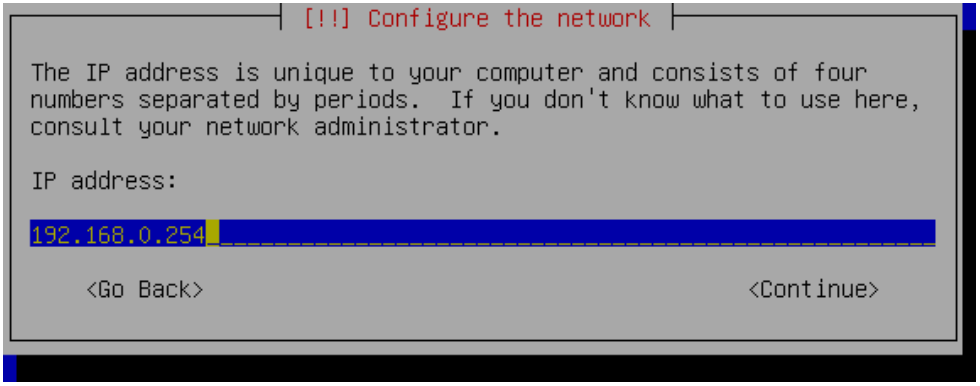
Keyboard Layout Chooser

The Linux distribution installer will usually also detect the installation media and network hardware.



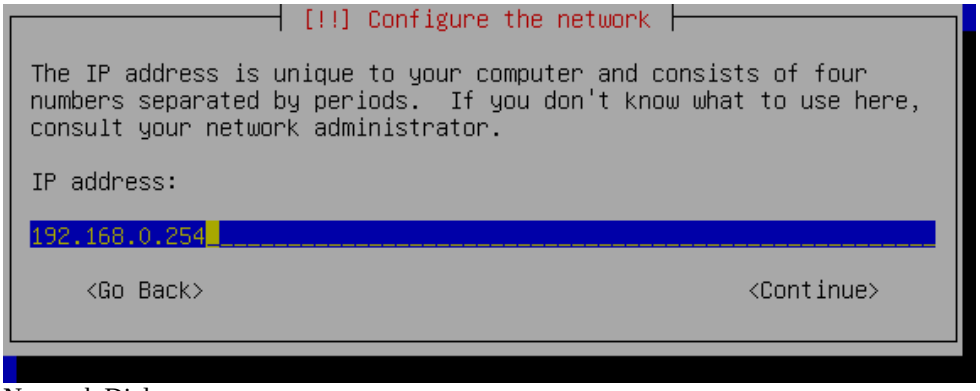
Setup is Detecting Hardware

Now you need to enter a network address for your server. This is called an I.P. (Internet Protocol) Address. If you're unsure what to enter here, put in “192.168.0.254”, such as the example below.



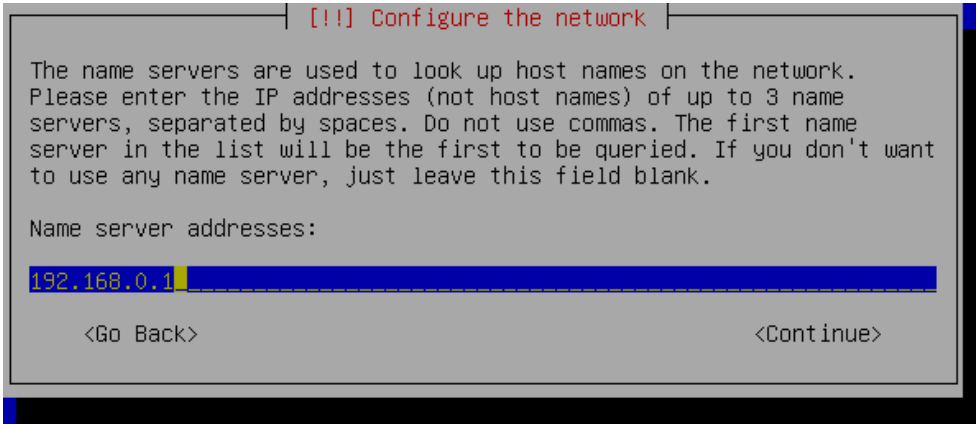
IP Address Dialogue

The next step is to configure the netmask. In most cases, this will be “255.255.255.0”.



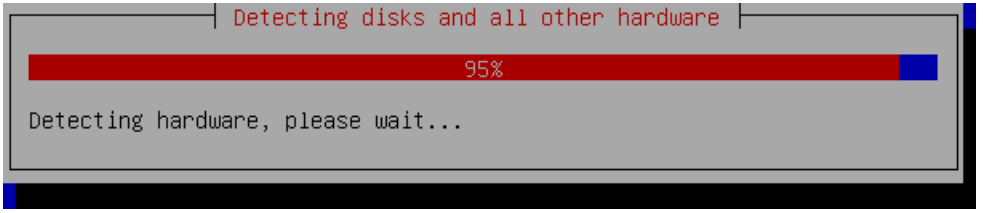
Netmask Dialogue

Enter the nameserver address. If you're unsure about the name server address, you may use the same value you have used for the I.P. address.



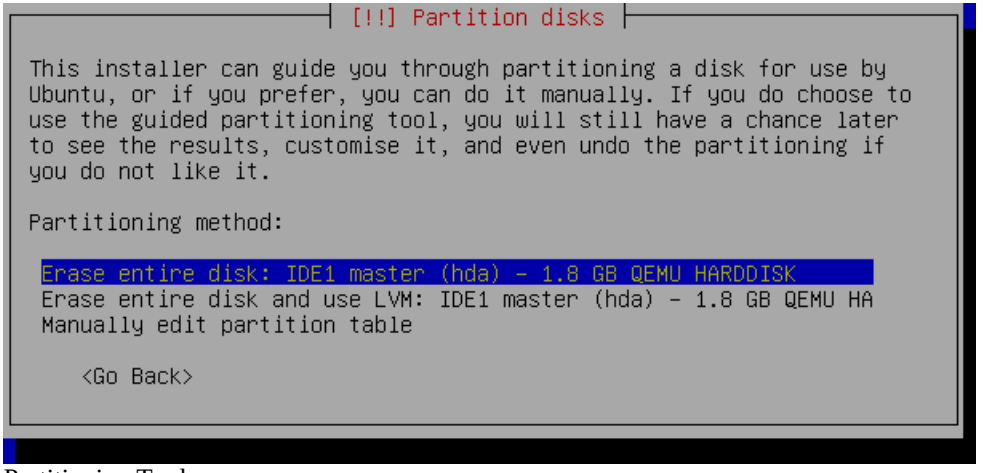
Name Server Dialogue

The Installer will detect your disks and other storage devices.



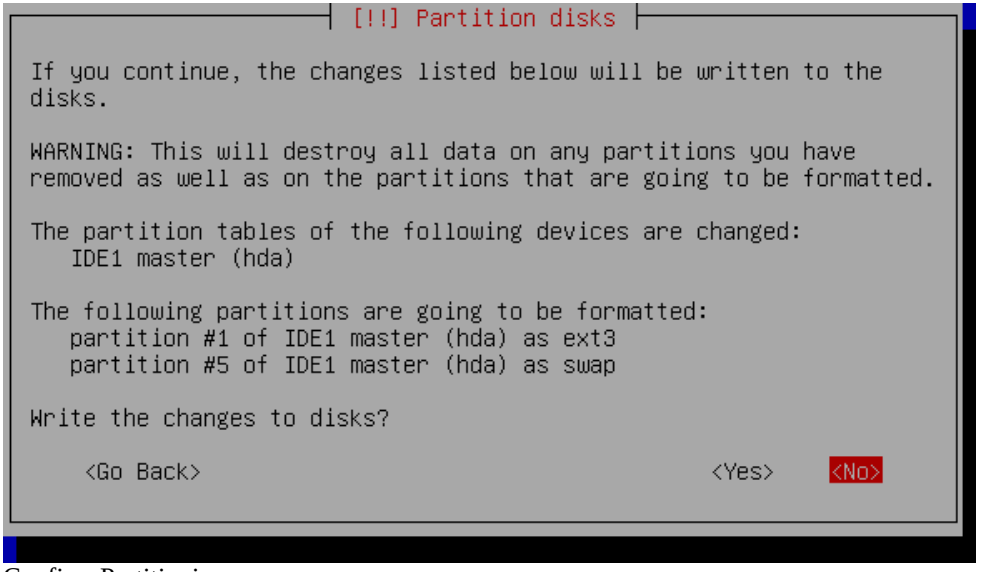
Detecting Disk Controllers

You will now be presented with a partitioning dialog. This allows you to split up your hard disk for the various types of data that will be stored on your system. It's recommended that you use the entire disk for the installation. If you have another operating system that you wish to keep, choose another option, such as resize current partition or a manual partition layout.



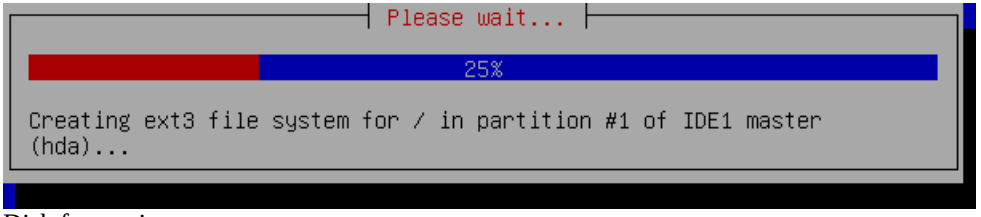
Partitioning Tool

The partitioner will give you a suggestion on how to partition the disk. Choose <Yes> to commit to the changes to the disk.



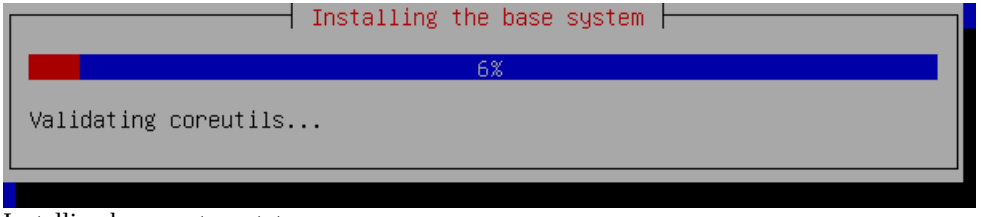
Confirm Partitioning

The disk will then be formatted.



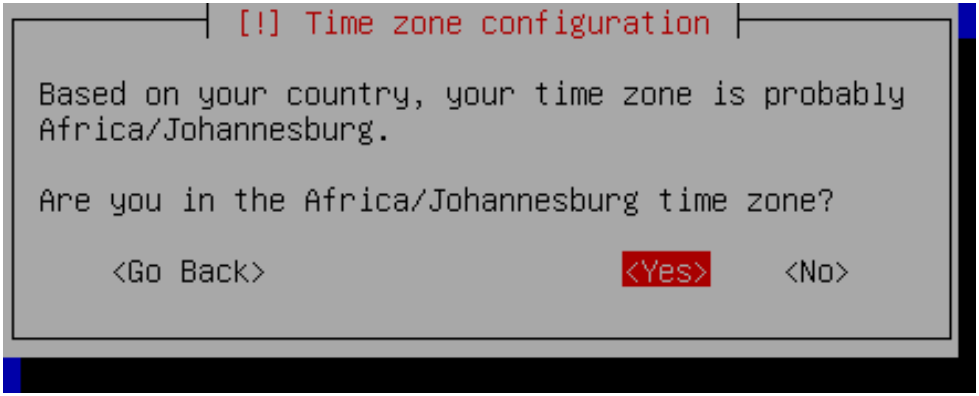
Disk formatting status

The installer will now install the Linux base system, the bare essentials that your system needs to start up and complete the installation.



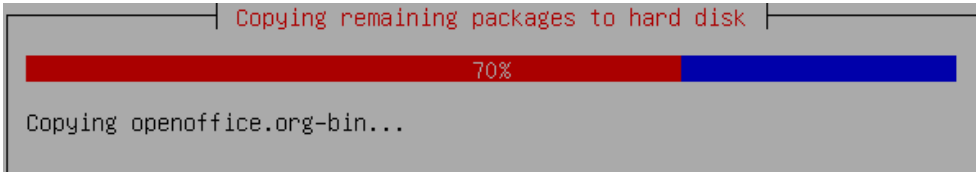
Installing base system status

The installer will attempt to guess the time zone, based on your country selection. If it's correct, choose <yes>. If you are located in another time zone, choose <No>



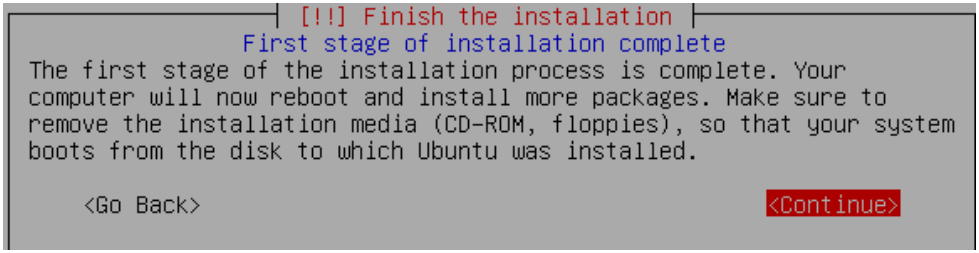
Confirm time zone

The Install procedure will now copy the remaining program files to the hard drive.



Copying remaining packages status

When the “Finish the installation” dialogue appears, choose continue to reboot.



“Finish the installation” dialogue

Remove the media you have used to boot the Linux installer. When the system restarts, the setup will be completed, and you will be able to log in as the administrator user you have created during setup.

6

Support Model

The Support Module deals with supporting and troubleshooting a TuXlab computer lab. The first section deals with the TuXlab support desk, including information on essential and extra software tools and physical tools used to find and solve problems. The second section deals with specific troubleshooting and problem solving topics.

6.1. Support Desk

6.1.1. Introduction

Support for a tuXlab lab is both formal and informal, but the starting place is the tuXlab support desk.

The support desk serves a number of roles in the tuXlab model. It is not just a technical support desk dealing with technical issues but is also an important contact point for the schools.

However its primary goal is to resolve technical issues experienced in a tuXlab as soon as possible. In our experience, issues range from simple questions about how to use applications to quite technically challenging issues that may often need to be resolved on site.

6.1.2. Essential Software Tools

There are some essential software applications that form part of the support structure that must be installed on your tuXlab server. Let's take a detailed look at what you need:

6.1.2.1. Request Tracker

We use Request Tracker to log and track issues reported to the support desk. It is an Open Source application developed by Best Practical bestpractical.com [http://bestpractical.com] and is released under version 2 of the GNU GPL license.

It allows us to accurately track support requests.

Issues are logged in Request Tracker as tickets, because tickets are not deleted but resolved it is easy to look up all the issues experienced by a certain institution.

6.1.2.1.1. Features:

- Web based, no need to install it on each support agent's computer. Our data is accessible wherever there is a Internet connection.
- Tickets can be created by end users via an email.
- Because RT keeps each ticket's history it is easy to retain knowledge and analyse trends.

6.1.2.1.2. Capabilities

- Tickets can be created by clients by sending emails to a specific address.
- Comprehensive search tool which can be used to pull up information for (as an example) a specific client between specific dates or of a specific category for a specific period. All of these search queries can be bookmarked for future use.
- Different categories (queues) for tickets can be created. Currently we have the following “queues”: Western Cape, Limpopo, Eastern Cape, Escalated. This allows for quick and easy identification and prioritisation of issues.
- Knowledge management - using RTFM module of Request Tracker

RT can be downloaded from: bestpractical.com/rt [http://bestpractical.com/rt]

6.1.2.1.3. Requirements

Your server will need the following prerequisites installed on it in order to use the Request Tracker:

- Perl 5.8.3
- Mysql 4.0.13 (with InnoDB) or Postgres 7.2 or Oracle 9iR2
- Apache 1.3.x or 2.x with mod_perl or with FastCGI [http://www.fastcgi.com]

6.1.2.1.4. Installation

A default installation of Request Tracker is relatively easy to do. In a Debian based system (like Ubuntu & Edubuntu) you only need to type:

```
$ sudo apt-get install request-tracker
```

apt will download and install the necessary packages for you.

Generic Installation instructions can be found at the bestpractical website [http://wiki.bestpractical.com/index.cgi?InstallationGuides] or in the README file in the installation package you have downloaded.

6.1.2.1.5. Concepts

Support requests are logged in RT as Tickets. The person who logs the ticket in RT is called the Requestor, tickets can be created by clients (schools) or by the support desk staff. Typically RT is set up so that clients can view their own tickets but not modify them in any way.

We use the Queues in RT to classify tickets. It is possible to set up many different queues. In our installation, we have queues for the different locations (Western Cape, Eastern Cape and Limpopo), we have a queue for Escalated issues, and one for software bugs that needs to be solved and implemented.

Tickets are also classified by their Status. In a normal RT installation the following statuses are available:

- New – ticket created but not worked on yet.
- Open – the ticket is being worked on
- Stalled – due to circumstances beyond your control the ticket can not be worked on now.
- Resolved – the issue has been solved
- Rejected – no work needs to be done on the ticket but it should stay logged in RT
- Deleted – ticket should not be in the queue, typically used to remove emailed tickets that are actually spam.

Watchers are people who are interested in a ticket to some extent. There are a number of different “watcher” roles in RT. These roles are shown under the “people” section of a ticket's display and also when you create a ticket.

- **Owner** – The help desk agent who is responsible for the resolution of the issue, this is not the requester of the ticket, Each ticket can only have one owner.
- **Requester** – the person who logged the ticket in RT, can be a support agent or a client (teacher). This person can see (gets emailed) replies to the ticket, but not comments added to the ticket.
- **CC** – someone who should get emailed copies of replies to the ticket. This person does not necessarily have permissions to view or modify the ticket directly. Does not receive comments added to tickets. Can be more than one person.
- **AdminCC** – someone who gets copies of replies and comments added to the ticket. Typically this is someone who is also able to modify the ticket. In our environment this is the support desk manager.

Please note that any of these roles can be assigned any right, but generally it is a good idea to limit what the client (typically the requester) and the CC user can do to the ticket.

6.1.2.1.6. Common tasks

- create a new ticket
  - Choose a queue from the drop down list and click on “New ticket in”
  - From the Homepage, choose “quick ticket creation” choose he appropriate queue and click on “Go”
- Display ticket contents
  - edit the URL from something like this: http://your.rt.url/rt/Ticket/Display.html?id=509 (which displays ticket number 509 to http://your.rt.url/rt/Ticket/Display.html?id=540 to display ticket number 540
  - click on the ticket's subject in a search list
  - enter the ticket number in the textbox next to “search” and click on “search”.
  - changing the status of a ticket
  - if the ticket is still “new” you can open or resolve the ticket by viewing it and clicking on “open” or “resolve” on the top of the ticket.
  - the status can be changed in “The Basics” section of the ticket as well. Click on “basics” in the side menu or “The Basics” in the ticket display itself to open this section of the ticket.
- resolving a ticket
  - view the ticket and click on resolve, you will be able to add a reply to the ticket then click on Update Ticket to change the status to resolved
  - view a ticket, click on Basics and change the status to Resolved, click on Save.
- take ownership of a ticket
  - view a ticket and click on Take or Steal, if the ticket does not have a owner Take would be available, if it does, Steal would be.
  - view the ticket, click on People then change the ownership and click on Save

6.1.2.1.7. Examples of Request Tracker

screenshot showing home page of RT

Figure 6.1. Screenshot showing home page of RT

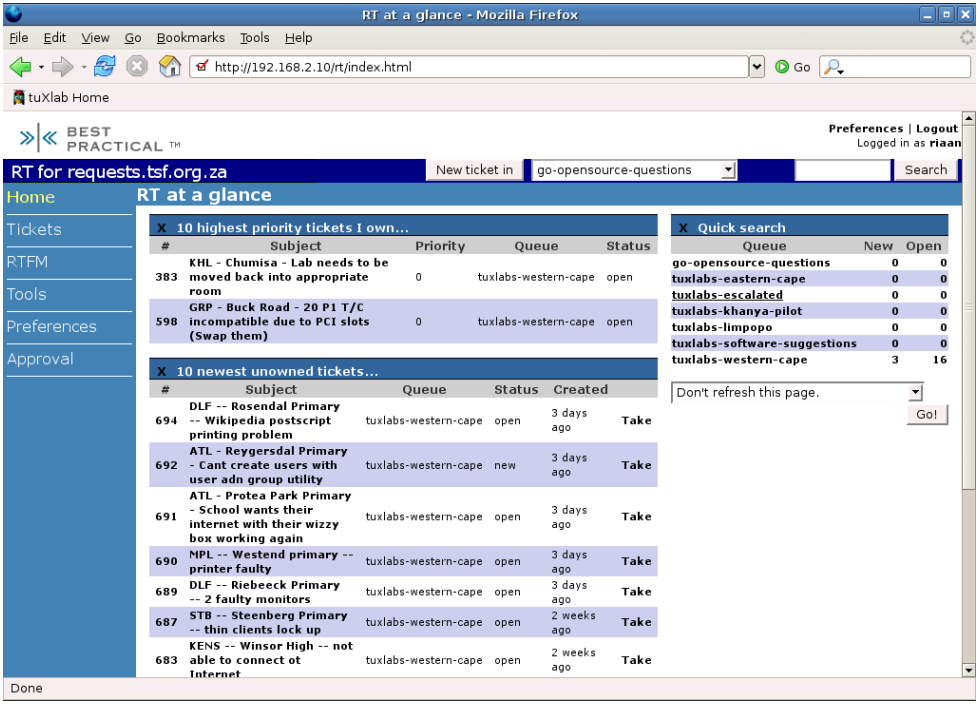


Figure 6.2. Example of a Ticket Basics

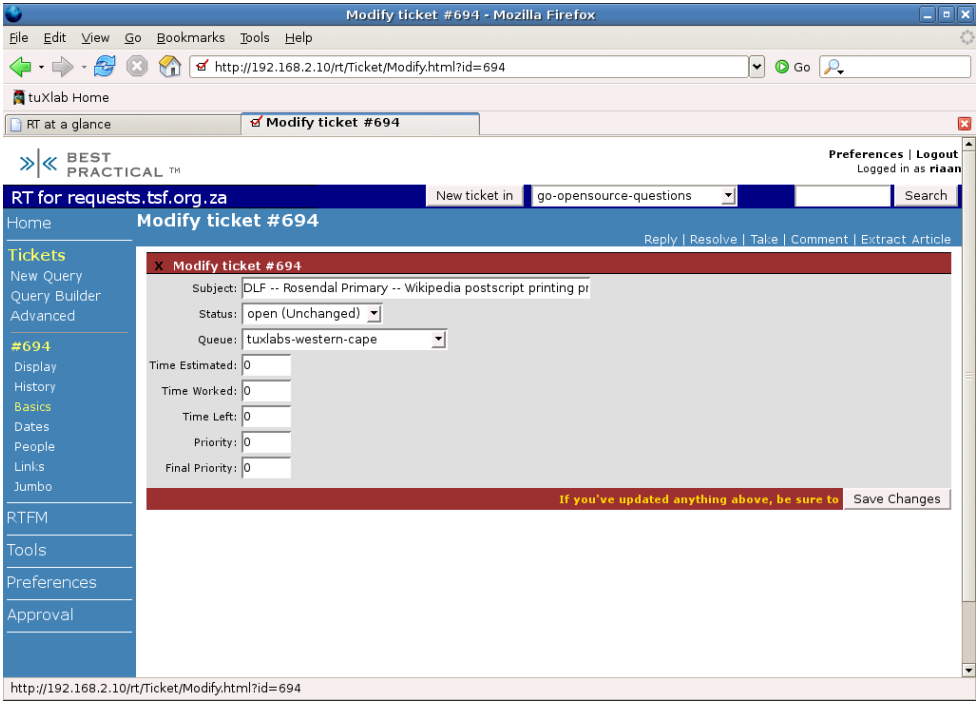


Figure 6.3. Example of a Ticket - People

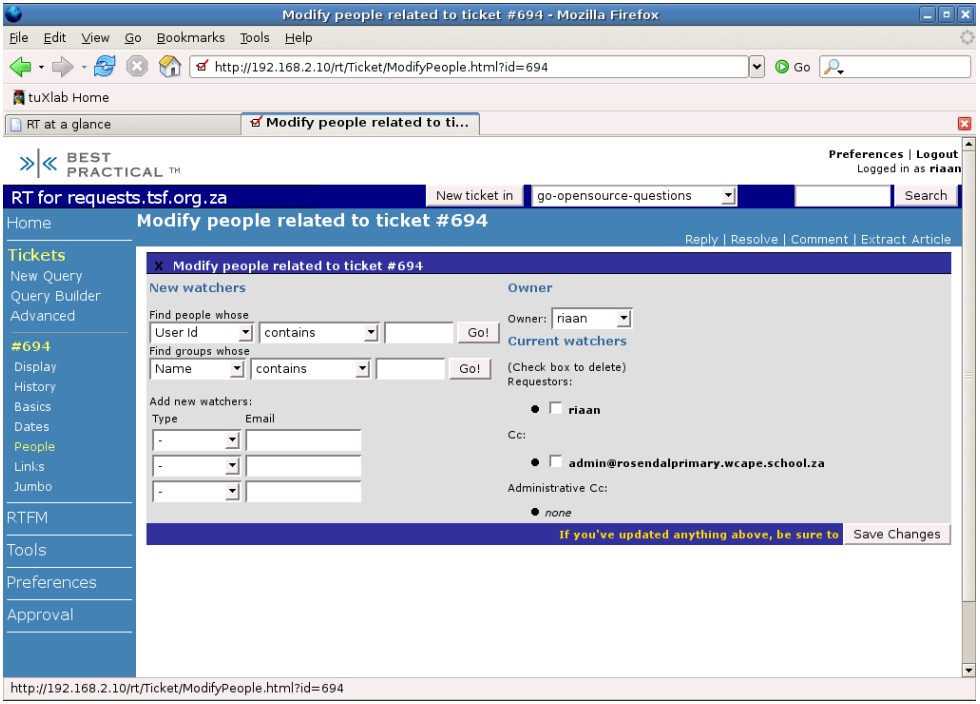
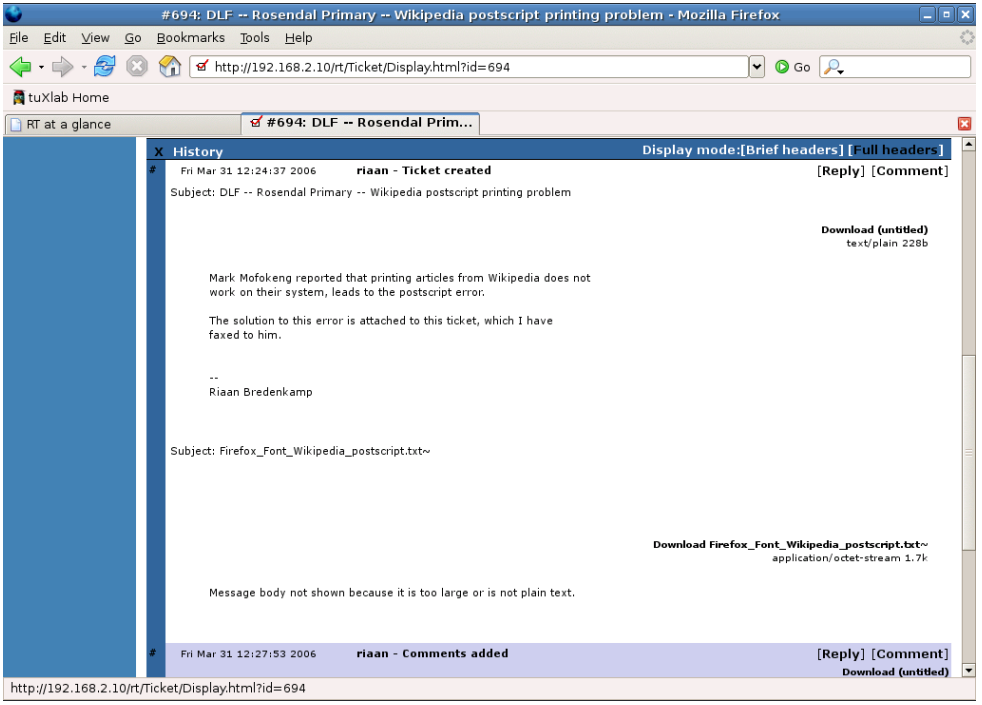
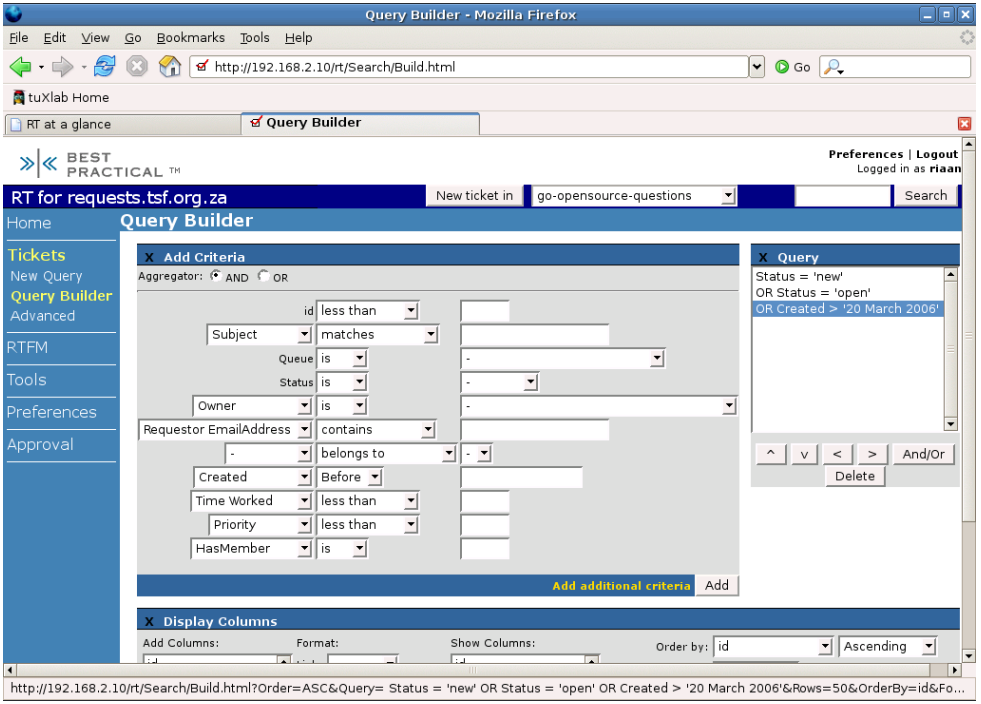


Figure 6.4. Example of a Ticket - Relationship Detail



The following screenshot shows a search result demonstrating different search parameters when building a query

Figure 6.5. Example of a Search result



6.1.2.2. LTSP Installed

The on-site support technician should have a laptop with the Linux Terminal Server Project installed on it. This will enable them to test thin clients, switches and even the server on-site. How this is done is dealt with later in this module.

6.1.2.3. Live GNU Linux distributions

Many Linux distributions exist which can be booted off a CD. Some that we use include:

- Ubuntu Live [http://www.ubuntulinux.org/] - used to boot a system which cannot boot and recover files. also used to check hardware compatibility in new servers.
- Knoppix [http://www.knoppix.org/] - used to boot a system which cannot boot and recover files. also used to check hardware compatibility in new servers.
- Stress Linux [http://www.stresslinux.org/] - used to test a server under load and to monitor the health of the system under load. Useful if you suspect that the server might have faulty RAM modules, faulty hard disk drives or even a faulty CPU.

6.1.2.4. Wiki

Internal communication between the team members is extremely important to the success of the help desk. because contact time with the schools are is so limited it is important to log all information that gets gathered during conversations with schools.

Request tracker is a good tool to log issues, but it lacks the ability to give one a quick, clear overview of the history and current status of a tuXlab.

We use MoinMoin Wiki [http://moinmoin.wikiwikiweb.de/], which is licensed under the GNU GPL.

A wiki is a type of website that allows users to easily add and edit content and is especially suited for collaborative writing. [from http://en.wikipedia.org/wiki/Wiki]

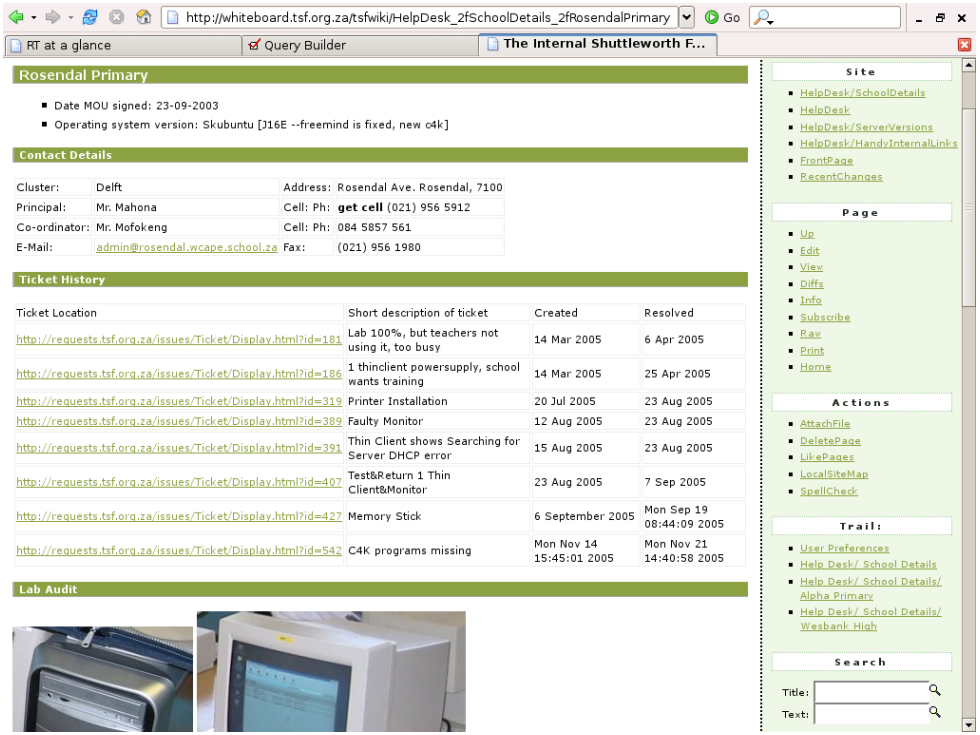
Because it is web based and easily editable by anyone it is ideal for team collaboration and logging of information quickly. Of course it has a version tracking mechanism, which means that you can easily revert to previous version of a page, in case a mistake is made.

6.1.2.4.1. Structure

We have a wiki page for each school. On it we log the following information:

- Date the tuXlab was installed
- Version of operating system running on the server
- Contact details for the school, including:
  - principle's name and phone number
  - fax number
  - email address
  - coordinator name and phone number
- a list of resolved issues the school has experienced, with a link to the appropriate ticket in Request Tracker
- System specifications of the server
- type and specifications of the thin clients
- lab layout diagram
- training received by the school.

Figure 6.6. Example of the School page Wiki



6.1.3. Essential Hardware Tools

Most of the issues logged by the help desk are hardware related. This means that the on site technician needs to have a tool kit at his disposal to open the machines and identify the problem.

Most often the issue involves:

- re-crimping of network points
- faulty power supplies on the thin clients
- faulty floppy disk drives
- faulty network cards

In order to deal with problems like this the on site technician needs to have the following tools:

- screwdriver set - to open thin clients or server
- crimping tool – to re-crimp network points

- Cat5 cable – to replace faulty network cable
- External hard drive – with operating system installation data, OS updates and school backups)
- box cutter – to cut holes in cable trunking
- side cutter – to cut network or power cables
- cable ties – to make things neat

6.1.3.1. Testing a thin client

The following is necessary to test a faulty thin client:

- boot-able network card – often it is not clear whether or not it is the network card that is faulty or the network cable. One way to test for this is to plug another network point into the problem thin client and to see if it boots – if the machine still does not boot it is quite likely that the network card is faulty. The final test, of course, is to actually use another network card and boot the thin client with it.
- PCI display card – sometimes the display device becomes faulty, whether it is a on-board or PCI device. Use a PCI display card to test if the problem is with the display device or some other component of the thin client.
- RAM modules – replacement Ram modules to test if the problem lies with the RAM in the thin client, often a thin client will not load the BIOS if it has a faulty RAM module.
- laptop with Linux and the LTSP installed on it – when none of the thin clients is able to boot and you need to determine if the problem lies with the server or the network switch you can use a laptop (or a desktop machine with LTSP installed on it) to determine which is faulty – the switch or the server.

6.1.3.2. Testing the network

There are many ways to test whether or not the network has been crimped properly, simply booting thin clients off the server could serve as proof enough that the networking has been done correctly but you can also use a network testing device. This allows you to quickly test all the network cables in the lab. It is a device that connects to both ends of the network cable and then checks that a current can pass through each of the strands in the network cable. It is also a good way to find which point on the switch end corresponds with a network point in the class, if you have not marked each cable while they were installed.

6.1.3.3. Testing the server

If all of the thin clients are not able to boot off the server you need to test whether or not it is the network switch or the server which is faulty. A laptop which has a PXE enabled network card can be connected to the server with a cross over cable to see if it can boot of

the server. If it can the the network switch is most likely the problem, else it is quite likely that there is something wrong with the switch – see the software module for information on how to troubleshoot this issue.

6.1.3.4. Testing the network switch

A laptop can be used to test the switch as well, use it instead of the server if your thin clients are unable to boot. If they still cannot boot then it is most likely that the switch needs to be repaired.

The laptop needs to have a network card that is able to boot off a network (PXE bootable)

6.1.4. Extra Tools

There are a few applications and tools that can make your life a little easier. These are not necessary for supporting the tuXlab, but are useful nevertheless:

6.1.4.1. Asterisk

Astrerisk is a software PBX (private branch exchange) that runs on Linux, BSD and MacOSX. It is released under the GNU GPL version 2 (or later) license.

it supports:

- Voice over IP protocols
- Voice mail services with Directory
- Call Conferencing
- Interactive Voice Response
- Call Queuing
- three-way calling
- caller ID services
- ADSI
- IAX
- SIP
- H.323 (as both client and gateway)
- MGCP (call manager only)
- SCCP/Skinny

Asterisk can be downloaded from:/www.asterisk.org [http://www.asterisk.org/download].

The help desk plan to use its call recording, Interactive Voice Response and caller ID services. Currently we are not using Asterisk to its fullest potential, but some of the possibilities we have identified include:

- use caller ID to identify the caller and send this information to Request Tracker's query tool. Request Tracker can then pull up all the tickets associated with that number and display them to the agent as he or she answers the call. This can be very helpful for the agent.
- se the Interactive Voice Response to setup a system where people are routed to the correct support agent depending on their query.
- Record calls for quality assurance. Calls can be reviewed by the help desk manager and the support agent to make sure that the correct information is passed on to the clients.

6.1.4.2. Multimeter

Used to test whether or not the power supply is still functional.

6.1.4.3. Cordless screwdriver

Used to open and close computer cases, can be a big time saver.

6.1.5. People

Currently the Shuttleworth Foundation tuXlabs comprises of 178 schools in 3 geographically distinct areas. The support desk mainly services the schools in the greater Cape Town area, which is a total of 120 schools. The support desk has 2 telephonic support technicians, one on site support technician and one manager.

Schools can make contact with the support desk in three ways:

- telephonically – to a “sharecall” number (the schools pay as if it is a local call). The vast majority of our contacts come in through this interface.
- via email – either through a mailing list or directly to the support desk.
- via a portal on our website.

The telephonic support agents are the first line of contact the schools have from the foundation. Typically they will log a issue with the help desk telephonically. A process (see later in this module) is followed where the telephonic agent logs and attempts to resolve the issue over the phone. If the issue can not be resolved within a set number of days – or the problem is identified as one that can not be resolved telephonically – the on site support agent takes ownership of the issue.

The on-site support technician is briefed by the support desk manager and the telephonic

support agent who had ownership of the issue. This should include an explanation of the issue, what the support agent has asked the tuXlab coordinator to do to resolve the issue and any historical issues in the specific lab which might have an impact on the on-site support agent's understanding of the issue and the proposed solution. A range of possible solutions are discussed and a strategy is formulated.

The successful resolution of the issue the first time the on-site support agent visits the school depends on how well she is prepared. Because we are dealing with schools we have a limited time in which to access the tuXlab and the tuXlab coordinator, so every effort needs to be made to correctly identify the problem before the on-site support agent visits the school. This is often not possible but there is nothing as frustrating as finding that though the solution is easy you are unable to resolve it because you are not adequately prepared.

6.1.5.1. telephonic agents

6.1.5.1.1. Roles

- second tier technical support for tuXlabs – both software and hardware
- preparation of servers and thin clients before it is installed in a tuXlab

6.1.5.1.2. Responsibilities

- resolving technical issues in tuXlabs
- researching solutions for issues intuXlabs
- creation of documentation explaining how to resolve common issues and how to use software applications shipped with tuXlab.
- logging and updating issues in Request Tracker
- updating school information – this includes contact details, a summary of past issues experienced in a lab, data about the equipment installed in a lab (type of thin client, network switch and server used)
- contacting schools who have not contacted the support desk recently to “touch base” - check status of the lab, inform them of news with regards to the tuXlab project and to check that contact details are still correct
- educating the teachers tuXlab coordinator on the cause and solution to issues encountered in the tuXlab.
- feeding back information to the rest of the support desk team on issues experienced, how to resolve them or if that was not possible, what they have tried to resolve the issue

6.1.5.1.3. Resources

The telephonic agents have the following resources available to them:

- Thin clients running off a server using the same software as is used in classrooms. This is useful for testing and troubleshooting issues.
- Access to the Internet – research new issues, stay up to date with changes in technology
- Fax machine – to fax instructions to schools. This is helpful if you need the teacher to enter a number of commands on the server.
- Access to a ticket tracking system, in our case that is Request Tracker.
- Access to a internal wiki – used to document school contacts, frequently asked questions.

6.1.5.1.4. qualifications

- matric
- A+, N+
- Linux +

6.1.5.2. on-site technician

6.1.5.2.1. Roles

The on-site support agent's roles is to do technical support for schools in the tuXlab project. Typically this includes refurbishing of “swap-out” thin clients (clients that the support desk use to replace faulty ones in schools) installation of Linux on the servers, troubleshooting and fixing of software issues on tuXlab server and the installation of any server updates on-site.

6.1.5.2.2.

- resolving issues that can not be resolved telephonically
- stock management of donated computer equipment
- feeding back information to the rest of the support team on hardware and software issues. The on-site support agent is often the most informed member of the team with regard to hardware issues and her knowledge of which thin clients are reliable can be invaluable for future rollouts.

6.1.5.2.3. Resources

The on-site support agent has the following resources available:

- ticket tracking system, in our case this is Request Tracker

- stock of refurbished machine to salvage parts from. Typically these include power supplies, floppy disk drives, CPU heat sink fans etc.
- a delivery vehicle – used to visit schools and transport equipment. The on-site support technician typically drives between 80 and 100 kilometres per day.
  - capital cost: R80 000
  - fuel expenditure: R2 400 per month
  - service costs: R 6 000 per year
- Thin clients running off a server using the same software as is used in classrooms [as a test bed that the support person can use]. This is useful for testing and troubleshooting issues.
- Access to the Internet – research new issues, stay up to date with changes in technology
- Fax machine – to fax instructions to schools. This is helpful if you need the teacher to enter a number of commands on the server.
- Access to a ticket tracking system, in our case that is Request Tracker.
- Access to a internal wiki – used to document school contacts, frequently asked questions.

6.1.5.2.4. qualifications

- A+
- N+
- Matric
- Linux +

6.1.5.2.5. On the Job

Once an issue has been handed over to the on-site support technician she prints a copy of the ticket from Request Tracker. This gets taken and showed to the tuXlab coordinator when the school gets visited, the aim of this is to encourage the teachers to use the help desk and ensure them that their problems are being tracked and dealt with.

Because schools are generally only accessible from 8 AM till 3PM the on-site support agent needs to ready the equipment that will be needed on the afternoon before visiting the school.

Once an issue is resolved by the technician she completes a job sheet for the issue in a duplicate book, the following details is included:

- ticket number
- date

- school name
- brief description of the issue
- how the issue was resolved

This is then signed by the teacher or headmaster and the technician. A copy is given to the teacher which is then, together with the print out of the ticket in Request Tracker added to the tuXlab log book. This helps them identify the issue and resolve it for themselves in the future.

6.1.6. Process

8 Step Process - This is the current eight step process schools are asked to follow when they encounter a problem in their tuXlab. Schools are required to follow the tuXlab support process, whenever a problem occurs.

1. **Identify the problem**

   This is often a difficult task for schools with new labs. The first step is to write down the nature of the problem in the ttuXlab log book. If a teacher or volunteer can not identify the source of the problem, they should make a note of the symptoms that they are experiencing.

2. **Make a note of the problem (or its symptoms) in the log book**

   Consult the tuXlab log book to establish if this problem has been experienced before, and if an appropriate solution was found. If this is the case, the school can also consult with the help desk with the relevant ticket number to gain further insight to the problem.

3. **XOLA – tuXlab Online Assistant**

   Included in the default tuXlab installation is a web application that will assist tuXlab coordinators identify and resolve issues. It is an interactive system that allows the teacher to identify the symptoms that she is experiencing in the tuXlab and then by a process of elimination, get to the appropriate solution.

   Included in this will be:

   - links to the appropriate HowTo documentation
   - “screencasts” [captured desktop sessions] showing how to do tasks, like adding users, using applications etc.
   - diagnostic information, showing
     - DHCP server status
     - NFS server status

- Software versions
- Uptime of the server

4. **Consult with the facilitator and computer committee**

   Find out if the computer committee has a solution to the problem. The members of the computer committee often attend training that would assist them in finding common solutions to problems in a lab. The computer committee is responsible for the upkeep of the lab, and they need to be informed of any problem that takes place within the lab. The computer committee can make use of the tuXlab troubleshooting guide, available in the tuXlab starter pack.

5. **Consult with neighbouring schools in cluster**

   Contact other tuXlabs in the area, and find out if they have a solution to the problem. When cluster meetings happen, schools can arrange that it can happen in a different school each month, so that a school will have a chance each month to ask questions to other schools about their lab.

6. **Contact the local user groups**

   Contact local volunteer groups, such as SLUG, PE-LUG and LimpLUG [de contextualise by referring to GLUE - Groups of Linux Users Everywhere -] Volunteers who are close-by will be able to assist the school faster than anyone else. Ask a volunteer if he/she is willing to adopt the school. This way you'll have someone close by who can keep an eye on the school who will be just a call away.

7. **Phone the help desk on 0860 67 4357**

   The tuXlab help desk will attempt to solve any problem that the school is experiencing telephonically. If you have any problem with your tuXlab that you are having trouble solving, then please contact the tuXlab help desk at 0860 OS HELP (0860 67 4357). The help desk will attempt to solve the problem telephonically. If the problem can not be solved telephonically, the help desk will provide the school with a ticket number, and put the school in a support queue. Oldest tickets are attended to first, and if there are no other solution, a technician is sent out to the school to assist. If it's a hardware problem, the help desk will log the problem. A tuXlab technician will visit the school, if appropriate, to replace hardware that's under guarantee.

8. **Contact the Foundation**

   If no solution has been found, contact the Technical Co-ordinator at the Shuttleworth Foundation on (021) 970 1200. The Foundation will then evaluate the problem, and find out why it couldn't have been fixed through the previous steps, and find ways to prevent the problem from occurring again in the future.

6.1.6.1. Evaluation

The support system and process has evolved to meet the needs of the schools that currently have tuXlabs. As more and more schools join the program and have access to the system it may need to adapt and change. The current support structure, however, has specific purposes in mind within it's structure. As we go though the pros, cons, and strategy evaluations of the tuXlab support system you will be able to evaluate them for yourself.

6.1.6.1.1. Pros

The first four steps of the seven step process is intended to foster skills transfer and self study in he coordinators of the tuXlab. The log book is intended to become a store of knowledge for future tuXlab coordinators, if the current coordinator leaves his experience and knowledge can be accessed by the next coordinator.

Another benefit is that it aims to improve communication between the tuXlabs in the same cluster and so instil a culture of cooperation and knowledge sharing.

t increases the sustainability of the tuXlab, if for some reason the foundation (... or driving force behind the project) can no longer offer support to the tuXlab steps 1 to 5 should be sufficient to enable the coordinator to get support for her issue long after the foundation has stopped offering official support for the tuXlab.

6.1.6.1.2. Cons

The support process is very reliant on the community – both the school community (schools in the same cluster) and the general open source community.

We have found that where the schools are communicating and helping each other, before becoming part of the tuXlab project, they are quite successful in helping each other with issues. For instance in the Paarl and Atlantis clusters (two areas which are not in Cape Town) the schools have cooperated with each other before becoming part of the tuXlab project, this cooperation has continued in the tuXlab project. Often one coordinator will help another school with issues in their lab.

However, in areas where the schools have not communicated with each other in the past it is quite difficult for them to start helping each other with their tuXlabbs.

Identifying the problem correctly and asking for assistance from the general Open Source community (typically using a mailing list) has also proven to have its difficulties. Teachers seldom have the technical expertise to correctly identify the cause of the issue and report it to a mailing list with enough information that will enable the people on the mailing list to respond with solutions. Connectivity is also a problem in South Africa, this means that tuXlab coordinators either cannot post an support request to the mailing list or is not able to post responses to questions or suggestions posed by members of the mailing list. This can lead to frustration and apathy for both parties.

6.1.6.1.3. Strategy

Issues are logged into Request Tracker immediately after receiving a call or email from the schools. This makes it easy for us to assign priorities to tickets according to the date the issue was logged.

We use a “first in, first out” system to prioritise issues. However the following circumstances does influence the priority that we assign to a ticket:

- Errors on the server or switch that leads to the whole lab not being functional is given the highest priority. Support is still done telephonically, unless it becomes clear that an on-site visit is needed to resolve the issue. Examples include:
  - corrupt disk partitions after hard shut down of the server
  - boot loader errors
  - critical software updates – typically updates are considered critical if the venue is used for teacher training, eg. Computers4Kids cluster training, ICDL training, etc.
  - faulty network switches, or
  - faulty network cable between the switch and the server
- Special project schools, for example department of education partner schools.
- Schools that have been identified as part of the “tuXlab reality therapy” program. These are schools who needs some extra care to bring them into the program again.

6.1.7. Statistics

The following analysis was done to investigate which issues are most commonly encountered in ttuXlabs. This will allow you to plan for the most common points of failure that you are likely to encounter in a tuXlab

6.1.7.1. Raw Data

Table 6.1. Least Frequent Issues

least frequently encountered issues	number of issues	Percentage [specific to this category]
root password	9	18.37
server software upgrade	8	16.33
switch hardware	7	14.29
power cables	6	12.24
mount	5	10.2

least frequently encountered issues	number of issues	Percentage [specific to thi category]
Wizzy Internet install	4	8.16
server motherboard	3	6.12
same user account	2	4.08
graphic chip set	2	4.08
software configuration	1	2.04
thin client memory	1	2.04
software bug	1	2.04

Table 6.2. Most Frequent Issues

most frequently encountered issues:	number of issues	Percentage [specific to thi category]
monitor failure	41	16.08
power supply failure	31	12.16
network point failure	27	10.59
server configuration	22	8.63
other	20	7.84
locked network switch	17	6.67
NIC	17	6.67
install software	16	6.27
mouse	15	5.88
keyboard	14	5.49
printer configuration	12	4.71
thin client BIOS	12	4.71
server hard disk drive	11	4.31

- Each of these issues refer to a instance where a support request was created, thus 1 “monitor” issue can actually represent multiple monitor failures. The goal of this analysis is to determine the most common technical failures experienced in tuXlabs, not to analyse ,for instance, how many monitors failed and was replaced.
- The analysis was done on 250 tickets which spans over a period of seven

months [January 2005 to August 2005]

6.1.7.2. Explanation of issues

monitor failure – faulty monitors can show various symptoms, including:

- fuzzy display
- no display at all
- burned-in images
- unable to display at least 800*600 [resolution]

power supply failure – the power supply unit of thin clients also fails often. Specialised knowledge is needed to repair these units. Typically we replace the power supply unit, or if we do not have a compatible power supply, the whole thin client.

network point failures – network points may need to be re-crimped, typical symptoms of a faulty network point includes:

- thin client does not boot at all
- thin client locks up
- thin client boots slowly

server configuration – the following falls into this category:

- display driver and resolution for thin clients - /opt/ltsp/i386/etc/lts.conf
- create user accounts
- IP address - /etc/network/interfaces
- other miscellaneous configurations

other – issues that occur very infrequently and that does not form part of the other specified categories. Examples include:

- how to use software
- faulty CDROM in server
- faulty power switches on thin clients

locked network switch – the switch does not allow connections between the thin clients and the server. It needs to be reset to allow thin clients to connect to the server

successfully.

Network Interface Card – most often this issue relates to us forgetting to replace network cards in thin clients. Occasionally the network card does fail, but not very often.

install software– during this period we added Computers 4 Kids to our distribution, there were cases where we had to manually install or update software.

mouse – refurbished mice do fail.

keyboards – refurbished keyboards do fail.

printer configuration – configuring a server to allow it and the thin clients to print to a server.

thin client BIOS – When the CMOS battery fails the BIOS settings on the thin client is lost, often resulting in the thin clients not booting. Settings that are changed in the BIOS:

- boot configurations – boot off network
- power saving – disable power save mode, monitor can still power down after a period of inactivity, but not the thin client.

server hard disk drive – the hard disk drives can fail due to:

- improper shutdown methods
- problems with the power supply
- factory defects

6.1.7.3. Hardware warranty period

One of the key concepts of the tuXlab project is that the school needs to take ownership of their tuXlab. To this end the foundation offers a limited time guarantee on the refurbished hardware supplied to schools with the understanding that the school will replace or repair faulty equipment themselves once the guarantee has lapsed.

In the tuXlab project the foundation guarantees refurbished equipment for 3 months after the installation date.

Technical support is not discontinued after the hardware guarantee lapses, the support desk will still assist the school to repair or replace the faulty hardware. This includes:

- identifying the fault with the hardware
- suggesting possible fixes for the issue, for instance if the floppy device is faulty and is of a type that can be replaced the support desk would suggest to the school that they purchase a floppy disk drive.

- assist the school to purchase the correct equipment, advise the school on prices and suitability (if needed)
- configure and install the replacement for the faulty device in the tuXlab (if needed)

Care should be taken to explain the hardware guarantee period correctly to schools to avoid confusion or confrontation in the future. It should be made clear that technical support is still offered – often schools are more than ready to pay for the equipment but because they lack technical knowledge feel they can not identify the issue and which equipment to replace or feel that they will not be able to replace the hardware once they have the replacement equipment.

New equipment, like the server and network switch, should be covered by the dealers warranty. Be sure to keep invoices and log which server and switch was installed in which school. A few of our network switches have had to be sent in for repairs and some of the hard disk drives in the servers have had to be replaced.

6.1.7.4. Tracking

Computer and monitors are identified with a barcoded label that is attached once it has been refurbished and is ready to be installed in a tuXlab. The purpose of tracking the hardware is to streamline the process of identifying out of guarantee hardware and to track which hardware is most reliable.

The barcode should also have a human readable alpha numerical code on it, so that tuXlab coordinators could read this number back to the support staff over the telephone.

An example of a machines human readable barcode is:

C01034

For this we can tell that it is a computer (C) [as opposed to a monitor] and that it is unit number 1034. This can then be looked up against a spreadsheet which will tell us exactly when the machine was installed in a school and its specifications.

This information is logged on a refurbishment sheet while the machine is being refurbished, an example follows. (Include a filled in form)

We do not, yet, have a asset tracking system. We are considering to use an add-on to Request Tracker called Asset Tracker (released under GNU GPL) a demo site can be found here [http://rt.chaka.net/AssetTracker/index.html?user=guest&pass=guest]

Tracking hardware will add some overhead onto the support process, but being able to correctly identify hardware hat is still under guarantee and also which hardware is most reliable is significant benefits. This will allow future equipment purchases to be done on accurate information with regard to stability and durability, something which is currently left up to the experience of the tuXlab technical coordinator

6.2. Problem solver

6.2.1. Introduction

Welcome to the troubleshooting part of the cookbook. It is aimed at the tuXlabb administrator who needs to find a quick solution to a common problem. [1]

Administrators are encouraged to involve their entire computer committee when troubleshooting issues. This will facilitate a skill transfer process that will improve the sustainability factor of your tuXlab.

6.2.1.1. Sections

6.2.1.1.1. Basic overview

In this section, this guide provides a basic overview of how the tuXlab works on both the hardware and the software side, which should give you at least a vague idea of where the problem might lie.

6.2.1.1.2. Troubleshooting problems

In Section 6.2.3, “Troubleshooting common tuXlab problems” [6–28], problems are listed by symptom, which allows you to quickly find the section relevant to you. Most of explanations in this section refer to common tasks performed by tuXlab administrators. Instead of explaining these tasks in every part, it refers to the Section 6.2.5, “Troubleshooting reference” [6–36], where you can also find a walk-through for the **`lts.conf`** file.

6.2.1.1.3. Known issues

In this section, you can find more information on known issues that need to be worked around. This will typically be issues that need to be solved by the developers of some software package installed in a tuXlab, or limitations of the hardware in use.

6.2.1.1.4. What to do when you get stuck

Section 6.2.6, “Further reading” [6–40] contains an explanation of the ttuXlab support process, so that the tuXlab administrator knows when to contact the appropriate person/organisation.

If anything in this document is unclear, remember that you may phone the tuXlab help desk on 0860 67 4357 for further information and explanations. They will also be able to log requests if you need any forms (such as the hardware order form or a tuXlabb audit form).

[1] This section incorporates the *tuXlab Troubleshooting Guide* originally written by Jonathan Carter.

6.2.2. Basic overview

6.2.2.1. tuXlab Network Topology

Your tuXlab uses a star topology. This means that every thin client connects directly to the switch, giving it a fast connection to the tuXlab server. In other topologies, such as ring topologies, messages are passed on from computer to computer. Star topologies are the fastest, and are best suited for thin client computers.

In the diagram below, you will notice that all the thin clients, as well as the server, connect to the switch using CAT-5 cable. The server connects to a special high-speed port on the switch, called the "Gigabit" port. It has ten times the data throughput of the other ports, allowing computers to have fast access to the server. This is necessary because all the clients are constantly talking to the server, and not to each other.

If your lab has an internet or e-mail setup, your server will also be connected to a *gateway* computer. This computer will dial up at night to collect your email and off-line content. The gateway computer will have a modem attached, that converts the sounds from your phone line to digital signals that your computer can understand, and vice-versa.

With troubleshooting, more than half the work is actually identifying the problem. Once you have a good overview of how things fit together, it is easier to locate the cause of the problem.

Thin clients rely on a network connection to work, so if *all* the computers fail to start up, you might want to check if the networking switch is turned on, or that the server is properly connected to a switch. If only one computer fails to start, it might be a problem with the cable connecting that computer to the switch. To make sure of this, plug this cable into a computer that starts successfully. If it stops working while using this cable and resumes working normally when the original cable is replaced, you know that the problem lies with the cable.

To fix the cable, you can try crimping the cable ends again. Examine both the switch end and the workstation end of the cable, making sure that all the copper wires in the CAT-5 cable are pushed up right to the end of the cable. If you notice something wrong, cut the cable a couple of centimetres beyond the RJ-45 jack, and connect a new RJ-45 jack with the crimping tool

6.2.2.2. Thin client startup process

- As soon as you switch on a thin client, it does a Power On Self Test (POST). This is the part where you normally see numbers counting, and a prompt to press **F1**, **F2**, or **DEL** to enter setup.
- After the POST is completed, control is given over to your network card's boot PROM (Programmable Read-Only Memory). This is a small piece of memory on the network card that allows a computer to boot from the network. The two methods of booting off

the network are called **Etherboot** and **PXE** (Pre-boot eXecution Environment). If you see one of these words, then your network card is configured.

- If your network card detects a connection to a switch, it will attempt to search for the server using DHCP. DHCP is the thin client's way of screaming "I am here! Please find me! Give me an address so that I can boot up!" The server will then give the thin client an address, called an IP (Internet Protocol) address. If a thin client is stuck at this point, it often means that the switch is on, but the server doesn't respond, or is not plugged in, or DHCP is not running on the server.
- DHCP tells the thin client that it should use FTP (File Transfer Protocol) to download the Linux kernel (a file called **`vmlinuz.ltsp`**) from the server. If a thin client gets stuck here, it's often a problem with the FTP server or a firewall blocking the FTP port. Perhaps the FTP server is not being started automatically?
- The next step is to gain access to a filesystem on the server over NFS (Network File System). This is the filesystem that all thin clients use to boot Linux after the kernel load, and can be found on the server at **`/opt/ltsp/i386`**.
- XDMCP is the X Display Manager Copier. It sends the graphical user interface (GUI) from the server to your thin client. If XDPCP doesn't start, it's often because GDM (Gnome Display Manager) isn't starting up. This is normally where the problem lies if you get a grey screen with an X cursor.

6.2.3. Troubleshooting common tuXlab problems

6.2.3.1. Thin client stops at "Searching for DHCP "

6.2.3.1.1. Symptom

Thin client stops at "Searching for DHCP...".

Possible causes:

1. DHCP server not running
2. Server not connected to switch (either the cable is not plugged in, or it's faulty)
3. tuXlab server not powered up

6.2.3.1.2. Solutions

1. If the DHCP server isn't running, you can restart it by opening a terminal, logging in as root, and then typing in the following:

```
# /etc/init.d/dhcpd restart
Stopping DHCPD [FAILED]
Starting DHCPD [SUCCESS]
```

In the transcript above, you'll notice that the attempt to stop the DHCP server failed. This means that it wasn't running when you attempted to boot.

2. Check if the green light is burning on the network card on the server. You can also check for a connection on the switch on the appropriate port. In some cases, it works better if the server is connected to a specific port (some switches have *two* gigabit slots). If only one thin client has this symptom, check that the connection light is burning on the network card. Sometimes all that's needed is to press the "Esc" button on the thin client's keyboard.
3. Check that the server is powered up. If the previous steps failed to solve your problem, attach a display, keyboard and mouse to your server to see any possible error messages.

6.2.3.2. Thin client stops at "Loading vmlinuz.ltsp..."

6.2.3.2.1. Symptom

Thin client says "Loading vmlinuz.ltsp..." then stops.

Possible causes:

1. TFTP Server is not running
2. Firewall is blocking TFTP port

6.2.3.2.2. Solutions

1. Trivial File Transport Protocol (TFTP) is used to download the Linux kernel to the thin client. It is started by a service called **xinetd**. Try to restart the server as root:

```
# /etc/init.d/xinetd restart
Stopping xinet [SUCCESS]
Starting xinetd [SUCCESS]
```

2. Disable the firewall. Otherwise, you need to reconfigure your firewall to open the TFTP port.

   Note: If your classroom server connects directly to the internet (without using a gateway machine), the firewall needs to be enabled. This will only be the case in a non-standard tuXlab that you may have modified yourself. Normally, the classroom server will not connect directly to the internet, but will go through a Wizzy server.

6.2.3.3. Thin clients have grey screen showing only an X cursor

6.2.3.3.1. Symptom

Thin clients display grey screen with an X for a mouse pointer instead of login screen.

Possible causes:

1. XDMCP is disabled
2. GDM is not running
3. Thin clients are connecting to wrong server

6.2.3.3.2. Solutions

1. Check if XDMCP is enabled. To do this, go to the server (you may have to connect a display, keyboard and mouse), and click on the main menu button on the bottom of the screen. Go to the *System Settings* menu, and choose "Login" to modify your login settings. You will have to enter the root password. Then, click on the "XDMCP" tab, and ensure that the box that says "enabled" is selected.
2. Check if GDM is running. You can gain access to the server by using a secure shell from one of the thin clients (refer to Section 6.2.5.1.3, “Using a secure shell from one of the thin clients” [6–37]). To check if GDM is running (once logged in), type:

```
# ps -e | grep " .dm"
```

   If GDM is running, you will see an output similar to this:

```
1973 ? 00:00:00 gdm
3226 ? 00:00:00 gdm
```

   If GDM is not running, or if you'd like to restart GDM, you can type:

```
# gdm-safe-restart
```

3. The server to be used for LTS (Linux Terminal Services) is specified in the **`lts.conf`** file using the "server" directive. This value is normally `192.168.0.254`, but may be different depending on your specific setup:

```
# from lts.conf:
server = 192.168.0.254
```

If you have more than one tuXlab server, or if you have more than one computer network in your school, then this address might have to be slightly different.

6.2.3.4. Thin client displays message "Link cable error"

6.2.3.4.1. Symptom

Thin client does not connect, and complains about cable error.

Possible causes:

1. Networking switch is switched off (in which case *no* clients will be able to connect), or faulty
2. Cable is not properly crimped
3. Cable is broken at some point

6.2.3.4.2. Solutions

1. Check that the switch is powered up. If none of the lights are on, or they don't flash, then the switch might be faulty. In this case you need to consult with neighbouring schools and, if necessary, the tuXlab help desk. If the switch is less than a year old, then it is still under guarantee, and it needs to go back to its supplier. The Shuttleworth Foundation will arrange a loan switch, if possible.
2. If the cable isn't properly crimped, you need to cut the network point off, and re-crimp it. If you do not have a crimping tool of your own, consult with a neighbouring school.
3. If the cable is damaged at any point, it needs to be replaced.

6.2.3.5. Thin client screen goes black or fuzzy at startup

6.2.3.5.1. Symptom

Thin client starts up normally, and displays text on screen, but as soon as the graphical login manager starts, the screen goes blank or fuzzy.

6.2.3.5.2. Possible causes

1. Display resolution is set too high
2. Thin client is using incorrect display driver

6.2.3.5.3. Solution

1. Some screens have trouble displaying resolutions of 1024x768, and you might need to set it down to 800x600. We do not recommend using 640x480, and if it can only handle that resolution, we consider it faulty.
2. To change to the correct display driver, you need to know which type of display card the thin client has. The best way to determine this is to open up the box and take a peek. The name of the card will be written on a black chip on the display card. If it's an on-board card, it will often be the chip closest to where you plug in the monitor. [[XXX: A couple of example images and names would be cool.]]

6.2.3.5.3.1. How to change the resolution and display driver

Thin client settings are all stored in a file called **`lts.conf`**. The full path to this file on the server is:

```
/opt/ltsp/i386/etc/lts.conf
```

You can edit this file with a plain-text editor of your choice (such as **vim**, **gedit**, **kate**, etc.). At the very bottom of this file, you can configure individual thin clients such as the following example:

```
[A1:00:08:53:F1:01]
    XSERVER = cirrus
    X_MODE_0 = 800x600
```

The line in brackets is the thin client's unique network address. This address is called a MAC address, and is determined by the network card. You can find out what the MAC address of a workstation is by pressing **CTRL**-**ALT**-**F2** on the thin client to get a text console terminal session, and then typing in:

```
# ifconfig
```

This will give you information such as the following:

```
eth0      Link encap:Ethernet  HWaddr 00:10:A4:7B:A7:CC
          inet addr:192.168.0.50  Bcast:192.168.0.255  Mask:255.255.255.
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:11 Base address:0x4c00
```

In the above, the MAC address is called HWaddr, and is `00:10:A4:7B:A7:CC`.

After saving the **`lts.conf`** file on the server, you can press **CTRL**-**ALT**-**F1** (to go back to the graphical console), and then **CTRL**-**ALT**-**BACKSPACE** (to restart the X server) to test your setting.

The XSERVER directive specifies what driver this machine should be using. In this case, it's a generic Cirrus Logic display card. If all fails, you might want to try a `vesa` driver, which is a very generic display driver that should work with most display cards.

The X_MODE_0 directive specifies which resolution this thin client should use. If the resolution is too high, text will become too small. Your ideal resolution should be 1024x768, followed by 800x600.

6.2.3.6. Thin client freezes / reboots regularly

6.2.3.6.1. Symptom

Thin clients kicks user out or freezes when under load, especially when using software such as Mozilla or OpenOffice.org.

6.2.3.6.2. Possible causes

1. Thin client runs out of memory
2. Thin client may contain faulty RAM modules
3. Thin client may have cooling problems

6.2.3.6.3. Solution

1. Enable swap over NFS. In most tuXlabs, this should already be enabled. To check, press **CTRL**-**ALT**-**F2** (from any thin client after logging in), and then type:

   ```
   gedit /opt/ltsp/i386/etc/lts.conf
   ```

   Scroll down, and check that the file contains the following lines under the `[defaults]` section:

   ```
   ENABLE_NFS_SWAP = Y
   SWAPFILE_SIZE = 32m
   ```

Swap file size should be a minimum of 8MB, but at least 16MB is recommended. Since we have lots of disk space on the server, we can make it 32MB. When a thin client runs short on memory, it will use swap space over the network system to avoid running into problems.

2. Check for faulty RAM modules. The best way to do this is to remove the memory modules from one thin client, and swap it with another. If the other thin client gives problems, then you know it is due to faulty RAM.
3. Check the cooling of thin client. If the CPU overheats or there are critical fans that are faulty, then it needs to be replaced.

6.2.3.7. Mouse doesn't move

6.2.3.7.1. Symptom

Mouse is stuck, and doesn't move at all.

6.2.3.7.2. Possible causes

1. Incorrect mouse type specified in **`lts.conf`** file
2. Mouse plugged into incorrect port

6.2.3.7.3. Solution

1. If you replace a PS/2 type mouse with a serial mouse, then you need to specify this change in your **`lts.conf`** file (editing **`lts.conf`** is explained in Section 6.2.3.5, “Thin client screen goes black or fuzzy at startup” [6–31].

   For serial mice (D-shape connector):

   ```
   [00:08:A1:48:F1:08]
       X_MOUSE_PROTOCOL = "microsoft"
       X_MOUSE_DEVICE = "/dev/ttyS0"
   ```

   In some cases, `ttyS0` might have to be `ttyS1`. If you have used serial mice in Windows before, then you will know `ttyS0` as `com1` and `ttyS1` as `com2`.

   For PS/2 mice (small, round connector):

   ```
   X_MOUSE_PROTOCOL = "PS/2"
   X_MOUSE_DEVICE = "/dev/psaux"
   ```

If it's a PS/2 scroll mouse, you can change "PS/2" to "IMPS/2", although this might not work on all configurations.

2. Check that the mouse and keyboard are plugged into the correct port. On most computers, the keyboard should be plugged into the bottom socket (purple), while the mouse is plugged into the top socket (green).

6.2.3.8. Keyboard doesn't work

6.2.3.8.1. Symptom

Keyboard is dead.

6.2.3.8.2. Possible causes

1. Keyboard is damaged
2. Keyboard is plugged into incorrect socket

6.2.3.8.3. Solutions

1. Check that the pins on the keyboard are not bent. If the keyboard is damaged, it should be replaced.
2. Please refer to step 2 in Section 6.2.3.7, “Mouse doesn't move” [6–34].

6.2.4. Known issues

6.2.4.1. Thin client capabilities

Thin clients, being network computers, have limits on what they're capable of. It allows easy administration, and maintenance, but it has the following limitations:

- 3D graphics are slow/unusable
- Full motion video playback is not possible on all the thin clients at the same time
- OpenGL drivers are not available for older display cards
- Software that uses lots of moving graphics do get slow

6.2.4.2. Known software issues

Some of the software we install is very new, and may still be under development. In this section, we list the software that we know have issues.

6.2.4.2.1. GCompris

GCompris is a relatively new piece of software, and in some parts of the program, it might kick you out of the program. GCompris is a great piece of software, especially with the lower grade students, we've included it for the parts that do work. We encourage teachers to inform our help desk of specific GCompris issues. We can then compile a list of issues experienced, and forward it on to the GCompris developers.

6.2.4.2.2. Tuxracer, Chromium

These are games that require a 3D graphics or OpenGL graphics. They are not supported on thin clients and we recommend that you remove these two games from your system.

To remove Tuxracer and chromium from your system, click on the main menu button on the bottom of your screen, point to "System Settings" and choose "Add/Remove Software". There you will have the option to remove this software under the Games section.

6.2.5. Troubleshooting reference

6.2.5.1. How do I get access to a terminal?

Simply put, a terminal is a place where you can type instructions for your computer. Terminal access is often the easiest way to adjust a wide variety of settings in your tuXlab.

There are several ways to gain access to a terminal. In the following, we'll show a few of the ways.

6.2.5.1.1. Opening a terminal screen from your GNOME desktop

If you're already logged in, you can right-click on your GNOME desktop and click on "Open Terminal" or "New Terminal" (depending on the version of GNOME you are using). This will open a new terminal window on your desktop.

6.2.5.1.2. Using a console terminal

Sometimes, it's not possible to log in using a normal account from a thin client machine. If this is the case, you might want to use a console terminal on the server. In order to do this, you need to have a display and keyboard attached to the server. To access a console terminal on the server, press **CTRL-ALT-F1**. You can then log on as root from there to change settings. To return back to the graphical user interface, press **CTRL-ALT-F7**. If you don't have a display and keyboard attached to your server, and your thin clients can still start up, you might want to access the server with a secure shell as described in the next section.

6.2.5.1.3. Using a secure shell from one of the thin clients

Thin clients are generally just connections to the server, but each thin client also runs its own little Linux system (once it has booted up) that you can use to access the server. To gain access to the Linux system, press **CTRL-ALT-F2** on any thin client. To connect to the server from a thin client, type:

```
# ssh server
The authenticity of host 'server (192.168.0.254)' can't be established.
RSA key fingerprint is 9e:8e:0e:9a:88:b9:a0:1e:0c:80:15:41:54:47:dd:fa.
Are you sure you want to continue connecting (yes/no)?
```

You will then see a message warning you about a RSA key, this is normal. Type "yes" and press enter. You must then enter the server's root password and press enter again. When you're done, you can press **CTRL-d** to exit the secure session, and **ALT-F1** to return to the thin client's graphical user interface.

6.2.5.2. How to reset the root password

There are two types of users: those who have forgotten their password, and those that will forget their password. This is often the case for system administrators as well.

To reset your root password, you'll need:

- Physical access to your server. Note well that anyone with physical access to your server can reset the root password, so take care to keep it locked up.
- A keyboard and display attached to the server.

Resetting your server in 4 easy steps:

1. Restart the server
2. When the **GRUB** [2] screen is displayed, press the **e** button on your keyboard, then the **Down arrow** button, then **e** again. Then press the **end** button, type in a comma, a space and the letter **s** (so that it's ", s" at the end of the line) and press the **b** button twice. This will boot you into single user mode.
3. Once the server has started into single user mode, type in:

```
# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

[2] GRUB is the first software program that runs when a computer starts. It is responsible for loading and transferring control to the operating system kernel. You can read more about GRUB in the FSF software directory. [http://www.gnu.org/software/grub/]

After typing **passwd**, enter your password, and then re-enter it. Your password (or password length) will not be displayed on the screen for security purposes, in case someone is watching over your shoulder.

6.2.5.3. `lts.conf` walk-through

The **`lts.conf`** file stores information about your thin client setup. Generally, thin client settings are detected automatically. However, there are times when you want to adjust certain settings manually. Remember to make comments when you change settings (it will make it easy for other people to troubleshoot the system). Comments are any lines in your **`lts.conf`** file that start with a #.

6.2.5.3.1. The `[default]` section

This is where all the default settings are stored that applies to **all** your thin clients:

```
[default]
    SERVER = 192.168.0.254
    XSERVER = vesa
    #used to be s3, changed to vesa on 10 December 2004 by Jonathan
```

In the section above, the LAN IP address of the Linux terminal server is set to `192.168.0.254`, and the default display driver for thin clients is set to `vesa`. The last line is a comment, which explains that the setting has changed from `s3` to `vesa`. Whenever possible, try to put reasons for the changes in your comments as well, as well as identifying yourself and the date of the change.

```
X_MOUSE_PROTOCOL = microsoft
X_MOUSE_DEVICE = /dev/ttyS0
X_kb_Symboles = us(pc101)
X_kb_Layout = us
```

The `X_MOUSE` and `X_kb` section sets the default keyboard and mouse settings for all the thin clients. Refer to Section 6.2.3.8, “Keyboard doesn't work” [6–35] for more information on changing pointing device settings.

```
USE_XFS = N
LOCAL_APPS = N
SCREEN_01 = startx
SCREEN_02 = shell
USE_NFS_SWAP = Y
SWAPFILE_SIZE = 24m
```

USE_XFS	Specifies whether we want to use an X font server or not. We generally say N for no here.
LOCAL_APPS	Specifies whether we want to use thin clients to run their own programs. This can be handy if you have powerful thin clients (Pentium 2 or better), because it takes load off the server, and can cause programs to run faster. Even though this can cause huge performance increases, it's lots of work to set up, and is still quite experimental. Future versions of LTSP will use local applications more as it matures and as hardware improves.
SCREEN_01 and SCREEN_02	These settings explain what we want to run on our thin clients. In this case, we will run our graphical user interface on the first screen of our thin clients, while we run a text mode shell on the second screen. To switch between screens on the thin clients, press CTRL+ALT+F1 for screen 1, and CTRL+ALT+F2 for screen 2.
USE_NFS_SWAP and SWAPFILE_SIZE	These are used to set up swap files for the thin clients on the server. When a thin client runs out of memory, it will use disk space on the server for extra swap memory. Enabling swap over NFS greatly improves thin client performance as well as stability. If SWAPFILE_SIZE is set too high, then there will be a waste in hard disk space and some loss in performance. It is recommended that you use a swap file of 8-32MB. 24MB generally works well.

```
RCFILE_01 = usb
RCFILE_02 = floppyd
```

RC (Run Command) files are scripts that are started up when the thin clients boot. **usb** and **floppyd** allows thin clients to connect to local devices using *mtools* (a set of programs that allow you to use floppies formatted as VFAT, the Windows disk format, without needing to mount and unmount them). If your thin clients have floppy drives, it's strongly recommended that you enable the **floppyd** script.

```
SOUND = Y
SOUND_DAEMON = esd
# SOUND_DAEMON = nasd
```

```
VOLUME = 75
SMODULE_01 = sb io=0x220 irq=5 dma=1
```

If you want to enable sound on the thin clients, then `SOUND` needs to be set to `Y` and it needs to be uncommented. `SOUND_DAEMON` selects the sound server to be used. Normally, only one of the two will work. The best way to set it up is to try the one, and if it doesn't work, try the other one. `SMODULE_01` selects the driver module you want to use with the thin client's sound card. Generally, `sb` will work with most sound cards. [[XXX: How are the io and irq parameters determined?]]

```
#60 hz resolutions:
X_MODE_0 = 1024x768 65 1024 1048 1184 1344 768 771 777 806 -hsync -vsy
```

You will normally find a whole bunch of mode lines like the one above in your **`lts.conf`** file. `X_MODE_0` specifies the resolution your thin clients should use. 1024x768 at 60hz is usually the best choice.

To change the resolution of one specific thin client, you can refer to it by its MAC address:

```
[00:08:A1:F1:EE:01]
    X_MODE_0 = 800x600
    LOCAL_APPS = N
    USE_NFS_SWAP = N
```

The numbers between brackets is the thin client's MAC address. This is the unique address of each thin client determined by the thin client's network card. `X_MODE_0` specifies the screen resolution. `LOCAL_APPS` is disabled for this thin client, as well as swap over NFS. Any of the settings in the `[default]` section can be applied to a specific thin client as well.

6.2.6. Further reading

6.2.6.1. tuXlab support process

The tuXlab support process is fairly simple. The support process poster in your tuXlab explains the steps:

1. **Inform your computer committee that you have a problem.** It's important that everyone in your computer committee knows what's going on in your lab. If a representative from SLUG or the foundation phones your school, they will ask to speak to a tuXlab computer lab committee member.
2. **Open up your logbook and note the problem in your logbook.** This allows you to

keep a record of problems that have been experienced in your lab, along with the solutions. If you or a school in your cluster experience the same problem again, you'll have a reference to fall back on.

3. **Allow staff and students and your facilitator to try and identify what the problem is.** For your tuXlab to be truly sustainable, you need to have as much local skills as possible. To promote your own skill levels, you have to try and solve as many problems as you can yourself.
4. **Contact the schools in your cluster to find out if they know a solution to your problem.** If a neighbouring tuXlab family school have experienced a similar problem, they will be able to assist you with your problem. If you get stuck, mention your problem at your next cluster meeting, or phone the schools directly and speak to their tuXlab committee.
5. **If no solution has been found, contact the tuXlab help desk at 0860 OS HELP (67 4357).** The help desk also keeps track of problems in tuXlabs, as well as other general requests. You may also phone the help desk at any time for help on any problem or help that you may need in your tuXlab. They will provide you with telephonic support.
6. Contact SLUG, the Schools Linux Users Group, a volunteer group who are very much dedicated to the tuXlab project. They can generally be reached via email. For more information, see http://www.slug.org.za.

   These are students and volunteers who assist schools with setting up their tuXlabs. They also assist schools after the labs have been installed. In return, they are encouraged via incentive by the Foundation with training and certification exams.
7. If no solution was found using these steps, you can contact Jonathan Carter at the Shuttleworth Foundation on (021) 970 1230.

   This is your last resort helpline. If no one else can help you, you have to phone Jonathan with a detailed explanation of your problem.

If you experience any hardware problems, it's generally a good idea to skip ahead to step 5 and phone the help desk, since other tuXlab schools might not always be able to help you there. Most tuXlab cluster leaders will have spare parts with them, contact your cluster leader to find out if they have any spares.

6.2.6.2. Where to source hardware from

If you'd like to buy additional hardware for your tuXlab, or if you need to replace faulty equipment of which the guarantee has expired, you can source hardware from the following suppliers:

6.2.6.2.1. Additional server hardware

Rectron: (021) 555 7111

For better pricing, you may order server hardware from Rectron via the Shuttleworth Foundation.

6.2.6.2.2. Networking hardware

Scoop distribution: (021) 555 4740 (Neal Andrews)

Scoop distribution supplies networking equipment such as switches, cabling, crimping tools, RJ-45 connectors and boots.

6.2.6.2.3. Thin clients

Additional thin clients may be obtained through the Shuttleworth Foundation. Please contact them to get the hardware order form, Shuttleworth Foundation Web Site [http://www.shuttleworthfoundation.org/].

6.2.6.3. Additional resources

These are just pointers to sites on the World Wide Web where you can read more about some of the topics mentioned, and where you can search for further information.

6.2.6.3.1. Google for Linux

Google has a section dedicated to Linux related questions: http://www.google.com/linux

6.2.6.3.2. LTSP Related

- The K12LTSP project home page [http://www.k12ltsp.org].
- The LTSP project homepage [http://www.ltsp.org]. This is the distribution from which K12LTSP is derived.
- The LTSP Wiki [http://wiki.ltsp.org]. A *wiki* is kind of site that's easy for visitors to edit. The name apparently means "quick" in Hawaiian, and it's intended to convey the informality of the typical wiki. It's often used to collect user-contributed documentation.

6.2.6.3.3. Helpful Linux sites

The Linux questions forum [http://www.linuxquestions.org], where you may submit questions to be answered by other members of the community. Remember to RTFM first!

The Linux Documentation Project (TLDP) [http://www.tldp.org]. This is a magnificent collection of documents that range from short HOWTOs to full-length books on all aspects of Linux system administration.

7

Sustainability Module

This module deals with the concept of Sustainability. The first section covers an overview and definition of sustainable development and suggests areas where sustainability should be applied to the TuXlab project. The second section deals with real life examples of TuXlab sustainability in practise.

7.1. Sustainability

7.1.1. What Do We Mean by SUSTAINABILITY

In January 2005, the United Nations declared a Decade of Education for Sustainable Development . This coincides with the goals of the tuXlab project and the goals of this cookbook. A good definition of development sustainability is“the continuation of benefits after major assistance from the donor has been completed” (Australian Agency for International Development 2000). Ensuring that development projects are sustainable can reduce the likelihood of their collapse after the initial pilot projects. It also reduces the financial cost of development projects and the subsequent social problems, such as dependence of the stakeholders on external donors and their resources. All development assistance, apart from temporary emergency and humanitarian relief efforts, should be designed and implemented with the aim of achieving sustainable benefits. There are ten key factors that influence development sustainability:

- Participation and ownership. Get the stakeholders (men, women and children) to genuinely participate in design and implementation. Build on their initiatives and demands. Get them to monitor the project and periodically evaluate it for results.
- Capacity building and training. Training stakeholders to take over should begin from the start of any project and continue throughout. The right approach should both motivate and transfer skills to people.
- Government policies. Development projects should be aligned with local government policies.
- Financial. In some countries and sectors, financial sustainability is difficult in the medium term. Training in local fundraising is a possibility, as is identifying links with the private sector, charging for use, and encouraging policy reforms.
- Management and organisation. Activities that integrate with or add to local structures may have better prospects for sustainability than those which establish new or parallel structures.
- Social, gender and culture. The introduction of new ideas, technologies and skills

requires an understanding of local decision-making systems, gender divisions and cultural preferences.

- Technology. All outside equipment must be selected with careful consideration given to the local finance available for maintenance and replacement. Cultural acceptability and the local capacity to maintain equipment and buy spare parts are vital.
- Environment. Poor rural communities that depend on natural resources should be involved in identifying and managing environmental risks. Urban communities should identify and manage waste disposal and pollution risks.
- External political and economic factors. In a weak economy, projects should not be too complicated, ambitious or expensive.
- Realistic duration. A short project may be inadequate for solving entrenched problems in a sustainable way, particularly when behavioural and institutional changes are intended. A long project, may on the other hand, promote dependence.

7.1.2. Specific Applications

The key areas within the tuXlab that need to be thought about and addressed in regard to sustainability include:

- Budgeting scenarios
  - School maintains lab, or
  - 3rd Party maintains lab
- Maintenance and support
  - Hardware
  - Infrastructure
  - Software
- Skills development
  - Training
  - Capacity building
- Management
  - Champions
  - Principals
  - Planning
- Stakeholders
  - Department of Education
  - Community

- School staff

7.1.2.1. Management

For further information on "Partnerships" please read the Management Module .

7.1.2.2. Training

For further information on "Volunteer Training" please read the Training Module .

7.1.2.3. Support

For further information on "Schools Linux Users Group" please read the Support Module .

7.2. Articles, Reports and Case studies

7.2.1. Gauteng

This information was first published in Tectonic, May 15 2006 by Jason Norwood-Young.

St Martin's High School in Orlando West, Soweto, has a 30-PC tuXlab computer lab. Running Edubuntu – an education-specific port of Ubuntu – the lab sports brand new equipment from Acer. Unlike some other computer laboratories, that face a loosing battle against support needs and unskilled teachers, St Martin's laboratory will be run by Sowetan entrepreneur Clifford Nkosi.

"What the pilot will try to address within schools is the constant problem of sustainability and promoting use of labs, maintenance, and general issues around computer problems. Schools themselves don't have the time or resources to take advantage of the fact that there's a community around them that also requires these services, so they can't use the computers and internet access to generate funds. By using an entrepreneurial partner, one can address issue of sustainability and funds, that can be used as funds for school, to maintain the lab, and pay someone's salary," says Hilton Theunissen, open source project lead, The Shuttleworth Foundation.

To stimulate interest amongst students, Nkosi has started a "geek club" amongst learners. The geek club will be exposed to open source in the workplace, with trips to open source companies, and will also be able to get computer certifications like the OpenICDL and LPI certifications.

As a marketing drive to the general community, Nkosi has offered 20 unemployed members of the community an introductory course to computers for free. Lab training and services will also be available to the general community from 2:00pm to 5:00pm for a fee.

Nkosi says that his main concern at this point is cash flow, and he is currently looking for

funding, particularly to pay an assistant for both training and support. The ex-Acer employee says that he's always had a nack for entrepreneurship, having started his first small business selling clothing in high school.

"His main aim is to make money in a socially conscience way," says Theunissen. "He's pushing the boundaries on this one."

If the project is successful, Theunissen hopes that the model can be adopted both nationally and internationally. Acer is particularly interested in the results, he says, and could consider this as a model for other sustainable development projects.

8

Training Module

The TuXlab Training Module deals with the process of training TuXlab volunteers. The first section presents an overview of the volunteer training course structure. The second section presents information about the training incentives scheme offered by the Shuttleworth Foundation to TuXlab's volunteers. The final section then presents the contents and levels contained within the three training modules.

8.1. Volunteer Training

8.1.1. Course structure

The training of tuXlab volunteers consists of levels or modules that take a beginning novice and, through learning and experience, create a tuXlab expert. Each module, from 'Boot camp' to 'Reservists', has a specific training course attached. These course give a comprehensive overview of the features and functionality of Linux, preparing the volunteer for certification of these skills. In-depth detail is provided for key concepts. Many Linux concepts and utilities are identical regardless of the specific distribution of Linux that is being used. The nature of Linux and Open source software is such that changes to source code and changes to functionality of any given component are happening continually. However, the underlying concepts of Linux capabilities and functionality remain consistent throughout distribution, kernel and software changes.

8.1.1.1. Introduction to Linux – Fundamentals

This course provides a 3 day introduction and in-depth explanation of the concepts and principles that are necessary to install a Linux system and navigate the command-line and GUI environments. It is specifically aimed at Linux novices and people making the transition to Linux. Focus is also given to the history of Linux and open source software.

8.1.1.2. Linux System Administration

This 4 day course will provide participants with a foundation in the concepts and principles that are necessary to administer a Linux system. The scope of an administrator's tasks may be very broad. Structure and function of the Linux kernel, key administrative topics of managing software packages, processes, disk space, backups and users as well as scheduling tasks and system logs are covered in this course.

8.1.1.3. Advanced Workshops

This will be the dynamic course in the offering. If a volunteer finishes both the beginners and intermediate courses, they are free to attend as many of these courses as they like, provided they meet the requirements set out in the training incentive scheme (see Section 8.2, “The Incentives Process” [8–2]. This course will take the form of a 1 day workshop, covering advanced Linux services and Open Source topics. This workshop will be run monthly by volunteer trainers, once enough volunteers have completed the first two courses.

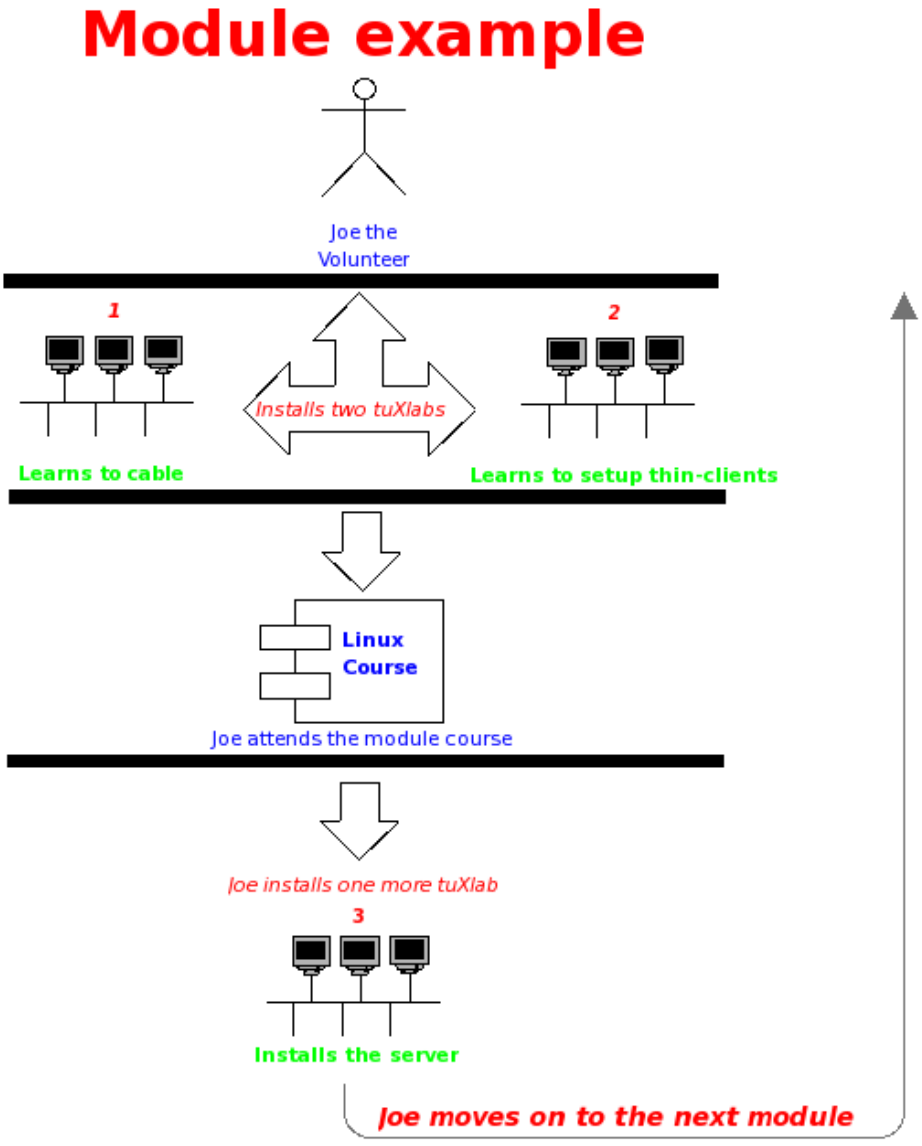
8.1.1.4. Facilitator workshops

A hands-on workshop covering all the technical principles and configuration aspects regarding the running and maintenance of a tuXlab lab. Example lesson plans for introducing learners to the tuXlab will also be provided and expanded on. This course will be run on a per cluster basis and will be exclusive to the computer committee from each cluster school, as well as the lab facilitator or volunteers offering free support to the cluster. Eventually, this training will be handled by the veteran volunteers.

8.2. The Incentives Process

The main incentive offered by the Foundation, to it's volunteers, is GNU/Linux training. In order to be trained on GNU/Linux and it's varying levels of complexity, the volunteer must pass through the following modules, satisfying all the associated requirements. Please refer to the following diagram for a better understanding of the process.

Figure 8.1. Training Modules Process



1. Joe would start out by helping install atuXlab. He would learn how to network a computer room and know the basics behind cabling.
2. At the next tuXlab installation, Joe will learn how to prepare the workstation computer and discover how the network card, in the computer, accesses the server.

3. After those two practical sessions, Joe will register for the module's course. If he's a beginner, the first course will be Linux Fundamentals. Joe will have the choice of doing this 3 day course either full-time or part-time.
4. Once Joe has finished the module's course, he will be expected to help install one more tuXlab in order to move on to the next module and receive more advanced training.

8.3. Training Modules

8.3.1. Module 1 - "Boot camp"

Candidates will have to actively participate in 2 (two) tuXlab installations, prior to the "Introduction to Linux - Fundamentals" course, and 1 (one) installation thereafter. These activities will be run in succession.

8.3.1.1. Level 1 - "Scrubs"

The volunteer will be assigned to do the cabling in a tuXlab. Instructions will be provided by the lab representative (see assessment) or module 2 volunteer, for the layout of the cabling, as well as the physical detail of the cable, connectors and crimping tool. Worksheets will be provided with all the theory required for advancement. At any given stage the participants will be asked to demonstrate the process to new volunteers or to the lab representative (see assessment).

Only once the volunteer has fully participated in this activity, and can demonstrate his/her new skill, will they be able to move to the next level in the next tuXlab installation.

Level 1 - Skills objectives:

- Know the make-up of cat5 cabling.
- Know the differences between popular network topologies. (Star, bus, etc.)
- Know how to crimp network points.
- Learn how to diagnose and troubleshoot connectivity issues.

8.3.1.2. Level 2 - "Newts"

At this level, the volunteer will be responsible for the set-up and operation of the thin-client terminals. Instructions will be provided on a worksheet, with additional verbal instruction from the lab representative (see assessment). Participants will learn about the requirements for thin-client computing as well as the theory of operation. Once again, volunteers will be asked to demonstrate their activity to new volunteers or to the lab representative (see assessment).

When the lab has been completed and the accepted level of skill has been displayed, he/she be able to advance to level 3 in the next tuXlab.

Level 2 – Skills objectives:

- Know how to insert a network card into a computer.
- Understand how the boot-prom works and how it is enabled.
- Troubleshoot and diagnose network card and boot-prom problems.
- Learn how to use the computer's BIOS options.

8.3.1.3. Level 3 - “Newbies”

Level 3 will be the “Introduction to Linux – Fundamentals” course – the first incentive. A maximum of 30 people can be accommodated at a time. Volunteers who were left behind in previous module1 activities and have finished the required pre-training requirements may also enrol for this training course. If demand requires, the course will be run when needed to avoid any bottlenecks in the process. Refer to the appropriate section in the “Course structure” for the overview.

Level3 – Skills objectives:

- History of Linux and open source software.
- Linux system installation.
- Navigating the command line.
- Using the “vi” text editor.
- File system and directory structures.

8.3.1.4. Level 4 - “Gnats”

After receiving the training in level 3, volunteers will be required to set-up a tuXlab server. They will be exposed to the “real-world” challenge of deciding on hard disk partitioning schemes, integrating the tuXlab with existing network structures and services as well as troubleshooting core tuXlab configuration. Instruction will be provided by the lab representative (see assessment) and participants will receive documentation on the technical aspects of the tuXlab theory of operation.

Level 4 - Skills objectives:

- Installation and configuration of a Linux server.

- Configuration of thin-client settings.
- Network and DHCP configuration.
- User and group administration as well as any environmental configuration (network integration etc.).

8.3.2. Module 2 - “In-service training”

The module 2 volunteers will be responsible for co-ordinating the practical activities (tuXlab activities) of the next batch of module 1 volunteers. Their “Basic Training” skills will be tested by teaching the new module 1 volunteers how to complete each practical level of the 1st module. This group will, for example, teach the new volunteers how to cable a lab and crimp network points, etcetera.

8.3.2.1. Level 1 - “Squire”

The volunteer will have up to four module1 “Scrubs” assigned to him/her. General instructions will be provided by the lab representative. It is up to the module 2 volunteers to make sure that his/her “Scrubs” can complete all the tasks and meet the module 1 – level 1 skills objectives. Once the lab representative is happy that the “Squire's” have passed on their knowledge to the “Scrubs”, and the “Scrubs” can demonstrate their new skills to the lab representative, will the “Squires” be allowed to advance.

Level 1 – Skills Objectives:

- Communication skills.
- Basic team leadership.
- Knowledge retention and recall.
- Skills evaluation.

8.3.2.2. Level 2 - “Delagator”

Again, the volunteer will have one to four module1 “Newts” assigned to him/her. General instructions will be provided by the lab representative. It is up to the module 2 volunteers to make sure that his/her “Newts” can complete all the tasks and meet the module 1 – level 2 skills objectives. Only once the lab representative is happy that the “Delagators” have passed on their knowledge to the “Newts”, and the “Newts” can demonstrate their new skills to the lab representative, will the “Delagators” be allowed to advance.

Level 2 – Skills objectives:

- Team leadership or one on one interaction.

- Technical communication.

8.3.2.3. Level 3 - “Initiates”

Level 3 will be the “Linux System Administration” course. A maximum of 30 people can be accommodated at a time. Volunteers who were left behind in previous module2 activities and have finished the required pre-training requirements may also enrol for this training course. If demand requires, the course will be run when needed to avoid any bottlenecks in the process. Refer to the appropriate section in the “Course structure” for the overview.

Level 3 – Skills objectives:

- Understanding the Linux Kernel.
- System backups.
- File system Hierarchy Standard.
- Package management and
- System log files.

8.3.2.4. Level 4 - “Top Dogs”

Only two volunteers will be selected to fulfill the role of training module1 “Gnats” to set up a tuXlab server. This is a specialized activity that requires a good understanding of how a tuXlab works. Unfortunately, an excessive number of volunteers participating in this activity will not benefit anyone. The two “Top Dogs” will be selected from the level 3 system administration course and selection will be based on individual performance during the the training. Instruction will be provided by the lab representative and participants will receive documentation on the technical aspects of the tuXlab theory of operation.

The rest of the module 2 volunteers can proceed on to the next module once they have received the tuXlab theory of operation document.

8.3.3. Module 3 - “Reservists”

The process of setting up a tuXlab doesn't change much. The cabling, thin-clients and server still need to be set-up and configured. In this stage the volunteer may be expected to supervise any of the Module1 and 2 activities - even then only a finite number of new experiences can be gained at each tuXlab. Module3 volunteers will be required to participate in the set-up of at least 1(one) tuXlab in order to proceed to the next training class, “Advanced Workshops”. That way, once everything is known about setting up a tuXlab, volunteers will not get bored with the activities but instead see them as a means to “pay” for GNU/Linux training. The Shuttleworth tuXlab program will therefore be successful if volunteers have something tangible to attain – certifiable skills (see

“Training”). Also, as the process of setting up tuXlabs becomes more easy for the majority of volunteers, the Foundation can step back and merely facilitate the process.

During this phase, additional incentives may be provided at the Foundation's discretion. Examples include, but are not limited to: t-shirts, Linux copies and food at events.

8.3.4. Public recognition

Setting up an tuXlab is great. Receiving public acknowledgement is even better! Once a volunteer has completed all the GNU/Linux courses and reached module 3, he/she will have the facility to manage an on-line “portfolio”.

The Foundation will showcase it's successful volunteers by giving them a web page containing details of their activity within the Shuttleworth tuXlab project. The page will display a picture of the volunteer and have all their relevant statistics, how many labs they participated in, peer statements, what training they received etc. This could be used as an employment tool for the unemployed volunteer or as a bragging piece for the employed section.

9

Volunteer Module

The Volunteer Module deals with the concept and process of using volunteers within the tuXlab program. The first section presents an overview of the concept of using volunteers, including a discussion on a broad strategy and approach to finding and involving volunteers. The second section deals with the process of including and training volunteers, and utilising their skills when setting up tuXlabs.

9.1. Volunteer Support Strategy

9.1.1. Overview

Our strategy is to leverage and support the intense volunteer spirit that surrounds open source software (OSS) at present. We aim to stimulate and nurture the creation of a well managed and well coordinated team of OSS volunteers by providing minimalist but highly specific support, infrastructure, resources and encouragement. Our funding will be minimal but targeted at energising the group, helping it to coordinate itself, meet regularly, develop it’s skills, identify interventions, attract new members and spread to other areas.

We will create a training program consisting of a series of modules like cabling, network installation, system administration, computer literacy and business use of computers, as the primary means of attracting volunteers.

This document brings the skills development aspect of our mandate into focus and defines the role that we will play in increasing open source skills and volunteerism. It also focuses on how we plan to apply the open source software development model, that relies on volunteers, to our tuXlab community project.

9.1.2. Concept

The Shuttleworth Foundation (the Foundation) is committed to the improvement of education, with a specific focus on mathematics, science and technology, as well as the promotion of open source software in South Africa. As a propagator of `social innovation', the main objective of the Shuttleworth Foundation is to invest in projects that provide innovative solutions to educational challenges in order to unlock the creative and intellectual potential of the South African youth, allowing them to live the dream that 'anything is possible'.

To further the uptake of technology in South Africa, the Shuttleworth Foundation supports the use of open source software. The term `open source' refers to computer software that

has been built by a global community of volunteers who make the source code freely available, as opposed to the more widely known style of proprietary development by a single company. This software is usually available on-line at no cost to the user. The Shuttleworth Foundation strongly believes that open source software should be the preferred choice of software for schooling in South Africa.

Open source software provides users with freedoms not obtainable from proprietary software, this includes the freedom to obtain, use, modify and distribute the software. The Shuttleworth Foundation believes that the use of open source software will provide effective and economical access to information and communication technology. The Foundation thus aims to initiate, facilitate, support and fund initiatives that will:

- Increase open source awareness
- Increase open source adoption and usage
- Increase open source skills
- Increase access to open source software

One of the major challenges that the Foundation faces is creating awareness of the functionality, flexibility and accessibility of open source. Up to now it would appear that an awareness of open source has been limited to sophisticated technology users. The Foundation aims to bring open source to learners from all levels of society – not only to make them aware of its existence, but also how this revolutionary, boundless technology can be used in software development and training.

9.1.3. Broad Strategy

The Shuttleworth Foundation has four core values which it applies to all the projects it funds and supports:

- Sustainability
- Innovation
- Empowerment
- Access to information

With this in mind, the Shuttleworth tuXlab program was initiated which aims to establish eighty (80) open source based computer labs in schools in the Western Cape. The primary objective of the tuXlab project is to make the school community aware of open source software and at the same time create an innovative replicable model for a cost-effective computing platform. Effort is also directed at making this project sustainable and the primary means of achieving this is through skills transfer and development with teachers, learners and volunteers. This process empowers staff members and thus builds capacity within the schools so that maintenance or expansion is controlled by them.

Given all of the above, the volunteer support strategy was developed and aims to provide a support structure to volunteer resources. The volunteer program will run concurrently with the tuXlab project, in a complementary fashion. The main aim is to empower the volunteer group by developing open source skills amongst them, so that they may run this project one day and play an active role in the local open source community.

The volunteer support strategy aims to solve the following problems:

9.1.3.1. Awareness

Many computer users are unaware of all the choices they have available to them when it comes to choosing software for their needs. By exposing people to open source software we expose them to the choice and generate awareness.

9.1.3.2. Usage

Users are generally advocates. If we break down the barrier of entry, we will increase the usage of the software and and also the amount of advocates for it. The barrier of entry is the presumption many people have about open source, often citing the argument that it is only suited for the more technically able users. That is not the case.

9.1.3.3. Skills Development

Skills development also goes a long way to break down barriers of entry. If volunteers get the opportunity to try the software at no risk to their own computers and data, they would be far more at ease and could concentrate on learning how to use the software. The tuXlab project also plays a major role in skill development, volunteers will learn many skills when they participate in the implementation of a tuXlab, such as network cabling and Linux installations.

9.1.3.4. Solving Other problems

The volunteer support strategy also solves some problems with existing models:

- **Cost of training**

  The cost of IT training is a prohibitive factor for many people wanting to delve into the world of computers. We provide a training incentive for volunteer participation. If a person volunteers at tuXlab installations then she/he will be eligible to attend regular, free GNU/Linux courses offered by us.

- Beginner unfriendly user groups – elitist culture

  Beginners in the open source world are often intimidated by the groups of people that are there in support of them. Many new users feel that their questions are dealt with in haste or disdain, often rendering them, in their own minds, better-off not asking for anymore help.

With this volunteer support strategy, new users who volunteer at tuXlab installations will build up their confidence and ask questions in a less anonymous and less faceless environment.

- Volunteer spirit – open source models for social development

  Volunteerism is one of the key factors that secured the success of the open source movement. Open source communities are generally formed by open source developers, administrators and users that freely provide their expertise and time to open source projects which interest them. The success of any great open source project can be directly attributed to the formation of a supportive community around a particular project. While the contribution of each individual member may be small, they have an extraordinary impact when combined, with an entire community and more benefiting from that collaborative effort.

  Contributing to an open source project does not just take the form of submitting source code, but rather any activity which benefits the community or open source in general.

9.1.4. Objectives

Strategic objectives

- Foster the spirit of volunteerism in the Western Cape by incentivising involvement in the tuXlab program.
- Develop a sustainable volunteer support structure for tuXlab schools.
- Promote and raise awareness about open source software.
- Grow the number of open source users in the Western Cape.
- Empower individuals with open source software training, in order to make them more “marketable” for future employment (and to support tuXlabs).
- Establish a Linux User Group (LUG) to use as volunteer hub, as well as an informal support forum for schools.

Strategic Goals

- Train 100 volunteers on the basics of GNU/Linux.
- Ensure a 50% pass-rate for external GNU/Linux examinations.
- At least 1 tuXlab open day every weekend.

9.1.5. Approach

We want to educate people on GNU/Linux, pass on open source skills and provide a loose-nit support structure to tuXlab schools. By utilising volunteers to help facilitate the process of installing tuXlabs, we will provide an unequalled method of continuing education for skilled and unskilled volunteers alike. The potential to reach communities and build one at the same time is the underlying idea behind both the tuXlab and volunteer support strategies.

Establishing tuXlabs is similar in many respects to an open source software project – rather than develop software, the Foundation is installing labs, but still using the concept of open source volunteerism to facilitate this process. The Foundation fulfils the role of 'Gatekeeper', acting as a facilitator and guardian, but using volunteer resources to complete the task.

9.1.5.1. How do we do this?

We do it by providing incentives. Volunteers provide labour and support to tuXlabs and in return we give them free GNU/Linux training and support.

There are two types of volunteer. Those that do it for no reward and are there to pass on their skills. And those who are trying to acquire new skills. The former, prefer to call themselves community builders, and we consider them to be partners in our strategy and goals.

9.1.5.2. How do we incentivise volunteers?

Any relationship is a two-way process. Taking, without giving back, will severely limit the longevity of any relationship. With this in mind, we will create a training program consisting of a series of modules like cabling, network installation, system administration, computer literacy and business use of computers, as the primary incentive for volunteers.

9.1.5.2.1. Training

We will create a training program consisting of a series of modules like cabling, network installation, GNU/Linux system administration, computer literacy and business use of computers, as the primary means of attracting volunteers.

Some of the informal training courses will happen at tuXlabs installed by the volunteer group. We can use such labs one Saturday every month. When we have setup 80 tuXlabs we should have 15-20 training facilities active every weekend. Short-courses will be presented at training facilities according to a schedule published in advance. Trainers will be graduates of the program or IT practitioners who have proven their ability to teach the subject matter. This initiative is called tuXlab Open Days.

- requirements for participation

  Volunteers are required to attend at least three tuXlab implementations to receive

training, two before the training course and one after. Once one level of training and implementation is completed, the volunteers move on to the next series of tuXlab installations and with it, the next level of training.

- training sessions

  GNU/Linux training sessions will be run both full-time and part-time. The duration of a full-time course lasts between 3 and 5 days. Part-time courses have two streams, half day training and Saturday training.

  General training on open source desktop tools, will happen on Saturdays at tuXlab Open Day venues. A tuXlab Open Day refers to an existing tuXlab school that opens it's doors for 1 Saturday every month, for the purpose of exposing the school community, local business and volunteers to specific open source desktop/productivity tools.

- levels of training

  There are various levels to the GNU/Linux training, which include the practical components. Candidates will have to actively participate in 2 (two) tuXlab installations, prior to the module course, and 1 (one) installation thereafter. Once the module is complete then the volunteer moves on to the next level of training, and with it a more advanced GNU/Linux training component. At this stage there are 3 (three) modules, Fundamentals, System Administration and Network Administration. We will add two new advanced modules as the need arises, Shell Scripting and Linux Internals.

- certification – LPI

  An agreement with the Linux Professional Institute (LPI) put us in a position to deliver popular, Internationally recognised GNU/Linux certifications at a fraction of the normal price. The LPI mission:

  "The Linux Professional Institute (LPI) serves the community of Linux and open source software users vendors and developers, in the interest of increasing and supporting professional use of such software throughout world. LPI will seek to improve the skills and resources of Linux and open source professionals, by providing services and setting standards which are relevant, of high quality and widely accessible." -- http://www.lpi.org

  We will train our volunteers to meet the level required in order for them to pass the LPI level 1 examinations, and then give them the opportunity to write the examinations. These examinations will be open to all, not just tuXlab volunteers.

9.1.5.2.2. Peer-respect

- training and participation levels

  Those that advance through the training and it's varying levels of complexity, will gain

the respect of their peers for their new found skills.

- presentation of training

  Those that have completed the training, could receive trainer training. So that they may deliver training courses to their peers.

- presentations at tuXlab schools

  Those who are progressing through the program can be nominated to do the presentations at school presenting the volunteer program.

9.1.5.2.3. Financial

The main incentives are and will always be educational and skills based. These will be continually updated to be current and of consequence. However, some incentives may take a financial form, and will only be awarded to volunteers who have demonstrated continued enthusiasm and commitment:

- reimbursement for successful LPI exams

  At the end of the qualification, we will provide access to the LPI certification - as an incentive, we will offer to refund those who pass, limited to the tuXlab volunteers.

- gift vouchers

  Gift vouchers can be purchased and given to those who do very well and deserve additional rewards and incentives.

9.1.5.2.4. Digital identity establishment

Granting access to special on-line services and clubs to volunteers who complete all the training and still continue to volunteer. Examples include:

- on-line volunteer portfolio/profile

  A facility for the volunteer to maintain an online profile, indicating his/her current status in the programme as well as interests, experience, qualifications, etc.

- personal web space

  Those who complete the program get to host their own websites (under an acceptable use policy).

- tuXlab e-mail club

  <name>@tuxlabs.org email addresses for those who finish the training program and continue to volunteer.

9.1.5.2.5. Digital freedom

This is about providing access to dedicated open source and Internet facilities. We will scout for venues, equip them and then provide access to volunteers who have made headway in the training program. Access will be controlled by an electronic tag system which will only be issued to the respective volunteers.

9.1.5.3. How do we ensure sustainability of the program?

We plan to run this like an open source development project. So how do open source development projects become sustainable?

9.1.5.3.1. Gatekeepers

A person or group of people that oversee the project and make decisions on it's daily operations, it's future and it's goals. These people oversee all the processes in the project. So in the volunteer support strategy, for the training, they would arrange training venues, lecturers and the registration of participants.

9.1.5.3.2. Sponsorship

Companies or individuals that derive benefit from the open source project, are known to sponsor development of certain pieces that would meet their needs and in some cases even give blanket-sponsorship. This volunteer support strategy hasn't required much financial input from the gatekeepers or third parties.

9.1.5.3.3. Retained enthusiasm

Enthusiasm comes from seeing what your efforts produce. Setting up a tuXlab every weekend or reaching a certain level in the training is bound to keep volunteers enthusiastic.

9.1.5.3.4. volunteer rollover

New volunteers bring with them new ideas and high levels of energy and enthusiasm, which will rejuvenate the program.

9.1.5.4. What support do we provide?

The support we provide is to the volunteers. In turn the volunteers provide support for the schools. Our support takes the following form:

- Social fora
  - mailing lists

We host numerous mailing lists, covering all the broad topics to do with tuXlabs, open source and volunteers. These lists are open to everyone.

- social meetings

  Occaisionally we host a get together for all the volunteers, at a tuXlab school, and provide lunch.

- transport

Transport to tuXlab installations, for most of our volunteers, is a problem. We are looking into ways of addressing this.

9.1.6. Stakeholders

1. The Shuttleworth Foundation

   The Shuttleworth Foundation initiated this strategy, and will continue to support it in whatever capacity. At present that means managing and facilitating the whole program. Eventually, however, we ultimately see this strategy in the hands of the volunteers and open source community, who will be responsible for it's evolution.

2. Schools & other institutions

   The tuXlab installations at schools are providing more than just a computer lab. We place emphasis on the fact that the tuXlab is a tool to promote open source awareness in the local community. All the schools participating in the tuXlab project, buy-in to this idea and are providing community access to their labs once a month under the Open Day program. This is integral to both the tuXlab and volunteer strategies.

3. Volunteers

   The volunteers and community builders are essentially the owners of this initiative, if they make use of the opportunities presented, then they will derive the most benefit from the relationship with the Shuttleworth Foundation.

   Graduates of the volunteer program will be the ones who will eventually own, modify and execute this strategy. We will support them, should they wish to manage or "own" any piece of the volunteer process.

4. Partners

   Partners will be sought to help execute this strategy and take it to other areas. This is the only way to get this program beyond the borders of the Western Cape. Current partners include SmartSource corporate training, the MTNSciencentre and the LEAP maths and science school.

9.1.7. Final Word on Strategy

By not generating and feeding skilled volunteers through the tuXlab program, the project would start to rely heavily on the our continued hands-on involvement. This would mean that the program would be limited to Cape Town schools and potentially not go beyond the borders of the Western Cape. We can not make a success of the tuXlab program, or reach the goal of 80 schools in 2004/5, without the involvement of volunteers.

The tuXlab program will be the springboard for this volunteer support strategy. A program like this has the potential to outlive the Foundation's strategies and focus, therefore it is crucial that we build capacity amonst the volunteers, pass on the torch and have it managed so that it can give the volunteer group longevity and purpose beyond the Shuttleworth tuXlab project.

9.2. Volunteer Process Control

9.2.1. Capacity

As a minimum, 8 volunteers working in two teams of 4, can install 2 tuXlabs in one day. If there are more than 10 participants at one tuXlab, generally you will find that most stand around without anything to do. In an effort to address this, we appoint a project leader for the installation and 2 generals under him/her. They are tasked with ensuring that everyone participates.

9.2.1.1. Volunteers

It takes 5 (five) volunteers one day, from 09:00 to 16:00, to install a tuXlab from the ground up - cabling, thin-client preparation, server installation and troubleshooting. Although there are usually more than 5 volunteers at a tuXlab installation, 5 is all it will take to install one lab.

However, the installation model is continuously evolving, teams of 4 volunteers go out to a school on a Thursday, to do the cabling. This makes the Saturday event less work orientated, and streamlines the installations. Schools that have had cabling done on a Thursday, generally have their tuXlab installed before lunch on Saturday. Focus is now directed at educating the staff about open source software and giving them ideas on how to integrate their classroom activities into the tuXlab.

9.2.1.1.1. Volunteer life span

Volunteers will not be expected to move on from the project, unless they contravene the code of conduct (Appendix 2 – Code of Conduct) or unless the project is shut down. Most of our current volunteers consider the tuXlab project to be a hobby, and thus really enjoy participating in the weekly activities. A distinction must also be made between regular volunteers and community builders. We are fortunate to have both in our group. Community builders are there for the eventual goal of bettering education through open

source software, and don't expect incentives. We consider these people to be partners in our strategies and goals.

9.2.1.2. Trainers

9.2.1.2.1. Train the trainer

During the training, certain volunteers who show potential, will be earmarked as future trainers and will be given all the attention needed to get them to the point of training a class.

9.2.1.2.2. Trainer life span

Once they are in abundance, trainers will assign themselves to a course on the schedule. However, during the start-up period of the training program, volunteer trainers will be scarce and will have to be managed efficiently. The first quarter of 2004 will be the ramp-up period for the training strategy, so fewer courses will be run. If demand requires then the tempo will be upped for the second quarter. Trainers, like volunteers, will not be required to leave the program after a set period. They will be able to participate for as long as they like. Assessment criteria will be developed once the training program is off the ground. Trainers will be assessed on their performance.

9.2.2. Support structure

9.2.2.1. Informal call-centre

The LCB is equipped to handle basic support calls. It will also be used as a dispatch office for “Volunteers on patrol”. If the problem can’t be handled over the phone then the person on duty at the LCB will refer the call to a nearby “Reservist” (See: 15.1.3 Module 3 - “Reservists”) who will then , hopefully, visit the school and fix the problem.

9.2.2.2. Volunteers on patrol

Once a volunteer has gone through all the training on offer and still participates, will be in a position to support a tuXlab from the ground up. If the volunteer still wishes to participate in the program, then they will be given a list of tuXlab schools in their area that could do with some support. He/she will then be expected to offer expertise to keep that school’s tuXlab running smoothly. These “Volunteers on patrol” would be a source of help for the school’s computer facilitator. If the schools in the area do not have facilitators, then these individuals could be regarded as independent, mobile facilitators.

9.2.2.3. Facilitators

Facilitators, if the schools can afford them, will be the first line of support for the tuXlab. A facilitator makes sure that the lab remains operational and that any problems with

hardware or software is resolved as soon as possible. Currently a facilitator is placed at each tuXlab school. The volunteer group provides a source for future facilitators.

An ideal situation would be for one facilitator to control a cluster of tuXlabs schools. Existing clusters are going to adopt this approach after the first year of operation.

9.2.3. Partners & Co-sponsorship

Within the workings of the tuXlab there are partnerships regarding training course venues and school installations. In this section we introduce you to a few of them.

9.2.3.1. Course Venues

- Smart-Source

  Smart-Source corporate training has made one of it’s 15 seat labs available to the Foundation for the purpose of training volunteers. If capacity demands, more facilities will be sought.

- MTN Sciencentre

  The MTN Sciencentre has made it’s auditorium facility available for 1 day workshops. They have also made their “Mecer Zone” computer lab available for volunteer training. The only problem with this venue is the fact that their machines are low-powered and not ideal for running Linux.

- Shuttleworth Advanced Labs

  We will be installing new equipment in two labs in the 2004/5 cycle, these labs will be shared with the school and will be used as training centres for volunteer training.

9.2.3.2. SLUG

The School Linux Users Group (SLUG) was established as an additional vehicle for volunteer participation. It provides a forum for volunteers to share knowledge and plan activities while collectively forging a unique identity for themselves as they go. The Shuttleworth Foundation is steering the group in it's early stages, until members take the initiative and run with it.

The Foundation is also providing web hosting services for the organisation. Mailing lists have been set-up for members to communicate, ask questions and announce events. The next step for SLUG is to formalise itself by electing a committee, forming a constitution and deciding it's future.

Slug provides a natural mechanism for sourcing training participants as well as managing the practical component of the training. SLUG has been the primary vehicle for attracting volunteers with the training component as an added benefit of participating in SLUG. The

training program requires practical experience in the tuXlab programme, of which SLUG is a major partner.

9.2.3.3. Courseware

Course notes will be available for download from the Foundation site once the training materials project has finished. Companies wishing to get involved in this project could be asked to make copies of these notes for volunteer training.

9.2.4. Legal Considerations

Due to current employment regulations in South Africa, the two legal documents (Appendix C and D) serve to protect both the volunteer and the Shuttleworth Foundation, from any confusion regarding the legal status of volunteer versus employee.

The Linux Computer Bank (LCB) is a facility that provides a central support hub for a tuXlab cluster. tuXlab volunteers would assist the Foundation in the refurbishment of old computers at the LCB facility. As the tuXlab project moves on and SLUG continues to be the driving force behind the supply of volunteers, so the distinction between the LCB volunteers and the SLUG volunteers blurs – in fact, one cannot draw a comparison, they are one in the same.

The documents attached in appendix C and D, were drafted for the LCB volunteers who have since become members of SLUG. The LCB facility is still used as the central support hub that it was intended to be - except it is now referred to as SLUG Head quarters. SLUG, being an independent organisation, will draft their own versions of the samples below.

Bibliography

Books

... . Copyright © ... .

Web Resources

.... URL Here. “...”. ... . Copyright © ... . URL.

A

Sample Business Plan

This appendix outlines an example of a business submitted by Maitland High School in January 2005.

A.1. Executive Summary

Description

Description of the school.

- Type of school: public, secondary etc.
- Learner demographics: income group
- Area: Western Cape
- Size: No of learners
- School structure

Mission

The schools mission statement and vision

Services

The learning areas the school caters for:

- Additional educational areas they cater for: computer sciences, etc.
- Any vocational skills training offered
- Plans for entrepreneurial or computer skills training
- The identified stakeholders for supplied services

Financial Forecast

- Commitment from school governing body to the programme.
- The preparations that the schools is making for the tuXlab programme
  - staffing requirements
  - infrastructure requirements

- any partnerships that might help

Financial Requirements

- Projected expenditure required by the school
- Any projected budgets for connectivity
- Any potential co-funders that have been identified

Business Structure

- Nature of the school as an entity: section 21, etc.
- Current subsidies
- Supervision of the programme and accountability

A.2. Market Research

Economic and Social factors

Factors influencing the functioning of the tuXlab as an entity

- Awareness of social responsibility
- Industry growth in the IT sector
- Focus on Outcomes Based Education

Target

The target market for the tuXlab

- learners
- educators
- ex-pupils
- parents
- immediate community

A.3. Business Strategy

Volunteer Incentives

- The value of a volunteer programme

- structure of a volunteer programme
- identified needs for a volunteer programme
- communication plan

A.4. Operations

Hours of Operation

- Installation date (date of readiness)
- activities for weekends
- activities for after-school hours
- in-school hours and integration into the timetable
- workshops and other activities

Staffing

Stakeholders included in implementing the operations and activities

Funding Options

- Costs for sustaining the tuXlab
- Costs for improving the tuXlab
- Connectivity costs
- Possible income generation scenarios

Long-Term Goals

- Break-even forecast
- Projected financial growth

Steps for Achieving Goals

- Establish volunteer programme
- recruitment of staff
- delivery and provision of education
- creation of library/tools for operation
- any offerings for income generation

B

Sample Quarterly Report

This appendix outlines an example of a quarterly report supplied by Alicedale Primary School in December 2005.

B.1. tuXlab Report

Meetings

- Number of meetings attended: cluster meetings, monthly meetings
- Any contingencies experienced in this quarter that prevented their attendance
- Any meetings hosted
- Details or executive summary of the meetings

Open Days

- Dates of open days held at the school
- Support of the event
- Activities

Computer Literacy Classes

- Received training events for educators
- Feedback on the training
- Identified needs for further training
- Literacy classes aimed at learners by educators
- Feedback on service providers or content providers
- Training events organised by 3rd parties: Computers 4 Kids, ICDL, etc.
- Adult literacy classes

Internet Cafe

- Operating hours
- details of events

- support of the initiative
- use of connectivity in-school
- technical issues
- additional training around internet usage

Help-Desk

- Report back on usage of the tuXlabs support system
- Technical issues
- Identified needs

Incentives

- Overview of incentive points
- Plans for awards

B.2. Additional Material

- Minutes of all meetings and all related communications are included in the report.
- Agendas and programme outlines for open days
- Attendance registers for events
- All relevant information around the marketing and use of the internet cafe
- Programmes and events around the use of the internet
- Usage reports for the internet cafe

C

Social Contract

Shuttleworth TuXlab Social Contract

Adopted by Shuttleworth TuXlab volunteers at the Linux Computer Bank (LCB).
Nooitgedacht Primary School
Bishop Lavis
Cape Town
South Africa

C.1. Introduction

The Linux Computer Bank is a social investment project with a dual focus. The primary focus is the refurbishment, configuration and provision of Shuttleworth tuXlabs (tuXlab) to schools in the Western Cape. The secondary focus of the LCB is skills transfer and development via the volunteer program.

C.2. Recordal

It is specifically recorded that the volunteers of the Shuttleworth Linux Computer Bank (LCB) – trainers, students, unemployed individuals and others who choose to serve the LCB are volunteers and not employees(hereafter referred to as “volunteers”) and they endorse and pledge to implement the following social contract amongst themselves:

C.3. To be involved in any classes or workshops given at the LCB:

In order to fully participate in and benefit from the LCB project, the volunteers must attend the relevant training sessions freely hosted by Jason Hudson and other volunteer trainers.

C.4. To volunteer our time to LCB Open Source Learning Centre activities:

Volunteers agree and undertake to pledge their collective skills and time, circumstances permitting, to the LCB project to ensure that the experience and skills afforded to them by the LCB project can be transferred to other individuals

C.5. To agree to the LCB code of conduct:

All volunteers must act and behave in accordance with the LCB code of conduct. Failure

to adhere to this code of conduct can result in exclusion from the LCB project.

In signing this document, the volunteer agrees and undertakes to abide by the provisions contained herein.

Signed:_____ Date:_____

Name: _____

D

Code of Conduct

D.1. Shuttleworth TuXlab Code of Conduct

D.1.1. Introduction

It is necessary to maintain a reasonable code of conduct in the workplace. The Shuttleworth Linux Computer Bank (LCB) project takes working condition issues seriously and that is why we have worked pro actively to develop and implement a Workplace Code of Conduct that is lawful, safe and ethical and which sets forth in detail our firm commitment to professional and ethical business conduct. The Code describes the behaviour that the LCB expects from it's volunteers.

D.1.2. Code of Conduct

Harassment or abuse. No volunteer shall be subject to any physical, sexual, psychological or verbal harassment or abuse. Every volunteer shall be treated and is expected to treat others with respect and dignity.

Non-discrimination. The LCB workplace must be free of all forms of illegal discrimination and accepting of a diverse volunteer base. The LCB complies with the laws governing child labour. No person shall be subject to any discrimination in volunteering, on the basis of gender, race, religion, age, disability, sexual orientation, nationality, political opinion, or social or ethnic origin.

Health and Safety. Each volunteer has active responsibility for maintaining a safe and healthy workplace for all volunteers. Violence will not be tolerated. Volunteers are prohibited from bringing any firearm and weapon to LCB gatherings or functions, including regular LCB activities. Any form of substance abuse by a volunteer whilst participating in an LCB activity will be deemed grounds for project exclusion.

Proper Use of Property. The property of the LCB and associated companies must not be misused or used for the personal benefit of an volunteer. For example, it is improper to make or install unlicensed copies of proprietary computer software.

Media Relations. Enquiries from the media, such as newspaper reporters, should be responded to only by the LCB member designated to handle such matters.

D.1.3. Discipline

Discipline. Volunteers engaging in improper workplace conduct will be subject to exclusion from the LCB project. The following acts have been identified as examples of unreasonable conduct which justify exclusion The list is not intended to be exhaustive and any other act not mentioned hereunder may warrant disciplinary action if the circumstances justify such action.

- Assault
- The carrying of dangerous weapons at LCB gatherings or functions, including regular LCB activities.
- Violence
- Alcohol abuse
- Drug abuse or the abuse of illegal drugs
- Fraud
- Grossly offensive behaviour
- Retaliation
- Dishonesty
- Theft

D.1.4. Summary

In summary, the LCB expects all volunteers to conduct their business dealings in accordance with the letter and spirit of all laws and LCB policies and to refrain from any form of illegal or unethical conduct or conduct that may harm good relations within the LCB.

E

School Awareness Questionnaire

E.1. Introduction

The Shuttleworth Foundation is committed to the improvement of education, with a specific focus on mathematics, science, entrepreneurship and technology, as well as the promotion of open source software in South Africa. As a propagator of `social innovation', the main objective of the Shuttleworth Foundation is to invest in projects that provide innovative solutions to educational challenges in order to unlock the creative and intellectual potential of the South African youth, allowing them to live the dream that `anything is possible'.

To further encourage the uptake of technology in South Africa, the Shuttleworth Foundation supports the use of open source software. The Shuttleworth Foundation strongly believes that open source software should be the preferred choice of software for schooling in South Africa. Open source software provides users with freedoms not obtainable from proprietary software, this includes the freedom to obtain, use, modify and distribute the software. The Shuttleworth Foundation believes that the use of open source software will provide effective and economical access to information and communication technology.

In 2002 the Foundation started to actively promote the use open source software as a computer lab solution for schools. The Foundation funded several projects to establish open source based computer labs in schools, as well as an internally initiating a pilot to prove the effectiveness of open source software as a school computing solution. The Foundation also initiated a project to facilitate the involvement of volunteers in refurbishing computers and establishing Shuttleworth tuXlabs.

Based on the success of the pilot and volunteer project, the Foundation extended the pilot and established a program with the ambitious ideal of establishing a further 80 Shuttleworth tuXlabs in the Western Cape over the next two years.

The Foundation is requesting that schools fill in this questionnaire in order for them to ascertain the requirements of schools for computer technology implementations. The questionnaire covers computer use and management, technology and open source knowledge and recognition of the foundation within schools. This data will allow us to identify areas where our focus and resources may be required to assist with ICT in Education. We appreciate your support.

If you would like assistance with the questionnaire or require any further information about this research, please contact

Hilton Theunissen

Open Source Project Manager

E-mail: hilton@shuttleworthfoundation.org

Url: www.shuttleworthfoundation.org

Tel: 021 970 1212

Fax: 021 970 1213

Please complete and return questionnaire below to :

Casey-Lea Olson

Open Source Project Administrator

Tel: 021 970 1232

Fax: 021 970 1233

E-mail: casey@shuttleworthfoundation.org

E.2. Questionnaire

Table E.1. Participant info:

School name	
Address	
Tel. no	
Fax. no	
E-mail	
Website	
Principal	
Secretary	
Computer Co-ordinator	
No. of staff	
No. of Learners	
No. of Governing Body Members	

Table E.2. Computers in your school:

How many computers are used in the school?	
Where are the computers used:	
Admin Block: No. of computers:	
Staff room : No. of computers:	
Computer Lab: No of computers:	
Are the computers networked?	
Does the school have a computer committee?	
How many members does the committee consist of?	
Who are the members?	
Do the learners pay a fee for participation?	
Is the focus computer literacy or curriculum delivery?	
Does the school have a teacher who facilitates the classes and does he/she have any formal computer training.	
Who is responsible for daily management of the lab?	
Who was responsible for cost of :	
1. security of the lab	
2. furniture in the lab	
3. electrical plugs installation	
4. computers in the lab	
When did the school acquire the computers in the lab?	
How was the equipment obtained?	
What speed are the majority computers? e.g. Pentium 1,2,3 or higher	
Did the school contract a service provider for installation and setup?	
If so, who	

Did any staff members participate in setting up the lab?	
Who is responsible for software support?	
Who is responsible for hardware maintenance?	
What operating and office application software is being used?	
Did the school pay for this software?	
If No, how did you get the software?	
Are you using educational software? If so, what?	
Did the staff get literacy and technical training?	
Was there any form of certification?	
Is the school staff utilising this lab?	
Does the school have Internet access in the lab?	
Who pays for the connectivity?	
What is the monthly cost?	

Table E.3. Open source:

Has the school heard of Open source software before?	
What does the school understand about the term?	
Where did the school learn about Open source?	
Has any staff member attended a tuXlab installation?	
What Open Source software does the school know or use ?	
How is the Open source software being used?	
Has the school heard about the Shuttleworth tuXlabs program?	
Would the school like to be included in our	

tuXlab program?	
Where did you hear about them?	
Has the school heard of our volunteer program?	
If so, where?	

Table E.4. The Shuttleworth Foundation

Has the school heard about the Shuttleworth Foundation?	
Where did the school hear about the foundation?	
What does the Foundation do?	
Does the school know of any of our projects?	
Has school visited our website before?	
Has the school seen any of our newsletters?	
Has the school submitted a proposal previously to the Foundation?	

Other comments:

Thank you for taking the time to fill out this form as accurately as possible. We greatly appreciate your assistance.

Confidential Disclosure:

We hereby confirm that the school details and information completed will not be made available to anybody outside the Foundation unless approved by the school by means of letter. The Foundation will provide the school with copy of results and the original will be

stored at the Foundation's offices.

F

Memorandum of Understanding (MoU)

F.1. PARTIES

This agreement is made between:

1. The Shuttleworth Foundation of 12 Plein Street, Durbanville, 7551 ("Donor"); and
2. School Name (the"Donee").

F.2. PURPOSE OF DONATION

1. The Donee hereby undertakes to utilise the donation for the exclusive purpose agreed to by the Donor. The specific purpose and project for which the donation has been requested and is made, is:
   a. The establishment of an open source based computing facility on the Donee's premises. This computer lab will be used for the purpose of providing computer instruction and computer-based learning to educators, learners and volunteers participating in Shuttleworth tuXlab Program.
   b. The establishment of and involvement in a support cluster among neighbouring schools participating in the Shuttleworth tuXlab Program.
   c. The growth of open source skills through the encouragement and use of volunteers in the installation and operation of the lab.
2. In order to best achieve the purpose stated in 1 [F–1], the following conditions hold:
   a. It is the responsibility of the Donee to provide a suitably prepared and secure environment for the operation of the tuXlab facility. A full list of requirements has been provided to the Donee by the Donor. See attachment 1.
   b. The Linux operating system and associated open source software will always be used for all facility workstations as well as the facility server. The use of open source software will always have preference over using proprietary software; the installation of proprietary software shall only be allowed should no suitable open source equivalent exist.
   c. The Donee will be obliged to identify and research open source educational software for computer-based curriculum delivery.
   d. The Donee shall establish a computer committee consisting of no less than three school employees. This committee will be responsible for coordinating the daily

operation and ensuring optimum usage of the facility.

e. The Donee's staff will be responsible for providing computer training and lab support.

f. The Donee shall ensure that a representative number of its educators are involved in the setting-up of the facility, performing tasks such as the installation of network cabling and configuration of the workstations. The majority of educators will also be required to participate in all training sessions offered by the Donor or computer facilitator.

g. Together with neighbouring schools participating in the Shuttleworth tuXlab program, the school will be responsible for the establishment of a local cluster committee, consisting minimally of the computer lab committee members from each individual school.

h. The school's lab committee is required to actively participate in the cluster committee. Each member will be required to relay information regarding their lab, such as experiences, problems, suggestions, as well as provide support to the other cluster participants.

i. The school's lab committee is required to assist with installation of two labs at future selected schools, within a two month period from school's lab installation.

j. The Donee shall prepare a plan outlining the intended use of the facility and present this to the Donor. The Donee shall ensure that this plan indicates how the lab will be incorporated into classroom activities so that all learners and teachers can benefit from the lab.

k. If a school has existing hardware, it may be replaced by the Donor with refurbished equipment if the existing equipment is not suitable. The existing equipment will be used to form part of an incentive scheme to participants from the school community. The Donor and Donee will set the criteria for selection of incentive recipient.

l. The Donee will make the facility available at least once a month for the purpose of open day workshops. Dates must be mutually agreeable by both parties. The Donee will ensure that a staff member will be present during these sessions.

m. Any course material or content developed by the Donee in the tuXlab will be made freely available to the Donor for use in the tuXlab programme and in other programmes initiated and sponsored by the Donor.

3. In the event that:

a. it is apparent to any duly authorised representative of the Donor that any donation which has already been made available to the Donee (or any representative of the Donee) are in fact being used for any purpose not reasonably contemplated in 1 [F–1], above; or

b. the Donor has cause to reasonably suspect that the donation provided by the Donor is being or is about to be used for a purpose other than that contemplated in 1 [F–1],

hen the Donor shall, in its sole discretion be entitled to cancel this agreement and to the extent that the donation has been used for an unauthorised purpose, to claim the return of any amount donated to the Donee in terms thereof together with any equipment or other asset acquired with such funds and the Donee shall not be entitled to hold the Donor liable for any damages flowing from such cancellation or claim for repayment of the donation.

F.3. PROVISION OF DONATION

1. The Donor shall internally allocate funding to the amount of R 00,000.00(AMOUNT IN WORDS SEE BELOW TABLE Rand) which will be used for the purpose of obtaining hardware, services and funding miscellaneous costs in order to establish the computer lab.
2. Should any funding be made available to the Donee, it is specifically recorded that the funding made available by the Donor shall be used for the purpose set out in 1 [F–1] and in the event that the Donee shall require that any part of the funding shall be used for a purpose not contemplated in 1 [F–1], the Donee shall, prior to incurring any such non-authorised expenditure, obtain the prior written approval of the Donor.
3. The donation shall, subject to 2 [F–1] be made available as follows:
   a. Funds will be released on an ad-hoc basis, at the Donor's sole discretion, based on the progress of the project. It is specifically recorded that funds allocated, subject to 1 [F–3], to the procurement of equipment for the project are part of the Donor’s 2005/2006 budget.
   b. The purchasing of all equipment, contracting of services and miscellaneous requirements will be coordinated by the Donor. All payments for project expenses will be made directly by the Donor.
   c. he following equipment shall be purchased by the Donor on behalf of the Donee (indicated costs are approximates):

Table F.1. Costs

1	Xeon Server unit	R	00.00
1	24 port switch	R	00.00
20	Power cables	R	00.00
20	Network cards and boot proms	R	00.00
20	Network cabling, RJ45's and boots	R	00.00
1	tuXlab kit	R	00.00

	Total	R	00.00

d. The Donor does not provide any warranties on equipment; the suppliers of the equipment shall enforce warranties in the instances where warranties exist. Additionally, the Donor shall not be held responsible for the reparation or replacement of any equipment that is damaged or stolen while that equipment is in the possession of the Donee.

e. An additional discretionary amount will be made available to the Donee to contribute towards expenses such as provision of resource manuals, lab launch, volunteer travel expenses, etc.

4. The Donor shall have the right, in its sole discretion, not to make any further donation available should the progress of the project not be satisfactory or alternatively the Donor may delay any proposed donation until it is satisfied entirely within its own discretion that the progress of the project is satisfactory or the Donee has given such other assurances to the Donor as it may find acceptable to ensure the progress of the project.

5. In the event that the Donor:

   a. is or becomes aware that the donation is being utilised for an unauthorised purpose; or

   b. if there is a reasonable suspicion thereof; or

   c. if the Donee fails to submit any report required pursuant to 6.2 and 6.5;

   d. if the Donee fails to keep the necessary records as required by clause 6;

   e. If the tuXlab is not used for a period of 6 months or more;

   the Donor shall be entitled to refuse to make available any future proposed donation unless the Donor receives satisfactory assurances from the Donee that such donation will not be utilised for an unauthorised purpose.

6. If the Donee utilises or there is a reasonable suspicion by the Donor that the donation will be utilised for an unauthorised purpose (as contemplated in 3.5 above):

   a. the Donor shall be under no obligation whatsoever to make any further donation available to the Donee; and

   b. the Donor shall be entitled to reclaim payment of any donation paid to the Donee, but as yet unspent; and

   c. take possession of all assets (whether tangible or intangible) acquired with the proceeds of the donation.

F.4. NAMING RIGHTS

The Donor reserves the right to select an official name for the facility for which the

donation is being provided., the purpose of which is outlined in 1 [F–1]. Should the Donor not elect to choose an official name, the Donee may select an official name that the Donor must then approve.

F.5. FORCE MAJEURE

The Donor shall not be liable for the performance of any of its obligations in terms of this agreement where performance of same is rendered impossible or impractical by virtue of the occurrence of any event beyond its reasonable control which renders such performance impossible or impractical including, but not limited to, theft of equipment or funds or any increase in costs not contemplated at the time of entering into the agreement, the occurrence of which event shall entitle the Donor to immediately cancel the agreement and the Donee shall not be entitled to hold the Donor liable for any damages flowing from such cancellation.

F.6. REPORTING

1. The Donee shall, if required by the Donor, present the Donor with a proposal document setting out the nature of the project, event or series of events for which donation is sought, the amount of donation sought and such other information as the Donor may reasonably require so as to enable it to obtain a clear understanding of the event or activity for which the donation is required.
2. The Donor requires that the Donee submit a comprehensive written report (or a further written report as the case may be) setting out details of the progress of the events or activities for which the donation has been obtained. This report should be submitted quarterly for the first three years of operation of the facility.
3. Such interim report as contemplated in 6.2 above, shall contain such information as the Donor shall reasonably require so as to enable it to make an assessment of the progress and continued viability of the project or activity for which the donation is sought, as described in 1 [F–1] above.
4. The Donee shall ensure that a representative will attend scheduled monthly Shuttleworth tuXlab workshops.
5. The Donee shall ensure that from the date that the donation is made available to the Donee to the date of completion of the agreed project or activities (or if terminated earlier pursuant to this agreement, the termination date) that accurate records are kept of all expenses incurred by the Donee together with all supporting vouchers and invoices.
6. Any donation which remains un-utilised by the Donee at the conclusion of the project shall be repaid to the Donor immediately on completion of the project.

F.7. PROMOTION AND ADVERTISING

1. Both parties shall be entitled, without restriction, to promote and advertise its

association with each other.

2. Both parties shall use their best endeavours to promote the tuXlab by providing editorial copy relating to and facilitating media coverage of the donation and activities relating thereto.
3. Both parties shall be entitled, without restriction and where deemed appropriate, to make and publish performance and evaluation reports on the proejct and on each other. Both parties shall be entitled to use any media, including (but not limited to) Internet websites, radio, television and print media such that any member of the public may be aware of and have access to such performance reports and evaluations.

F.8. DOMICILIUM AND NOTICES

1. The Parties choose their domicilium citandi et executandi ("Domicilium") for the purposes of the giving of any notice, the payment of any sum, the serving of any process and for any other purpose arising from this Agreement at their respective addresses stipulated in clause 1 above.
2. Each of the Parties shall be entitled, from time to time, by written notice to the other, to vary its Domicilium to any other address within the Republic of South Africa which is not a post office box or poste restante.
3. Any notice required or permitted to be given in terms of this Agreement shall be valid and effective only if in writing.
4. Any notice given and any payment made by one party to another (the "Addressee") which:-
   a. is delivered by hand during the normal business hours of the Addressee at the Addressee's Domicilium for the time being shall be presumed, until the contrary is proved, to have been received by the Addressee at the time of delivery; or
   b. is given by telex or by facsimile shall be deemed (in the absence of proof to the contrary) to have been received within 24 (twenty four) hours of transmission where it is transmitted during normal business hours of the receiving instrument and within 48 (forty eight) hours of transmission where it is transmitted outside those business hours.

Should the terms and conditions be accepted by you, kindly indicate your acceptance thereof by signing this agreement in the space provided below and initialling all annexures hereto and return a copy thereof to our offices at your earliest convenience.

FOR THE SHUTTLEWORTH FOUNDATION	FOR DONEE

DATE	DATE
AS WITNESSES	AS WITNESSES
1.	1.
2.	2.