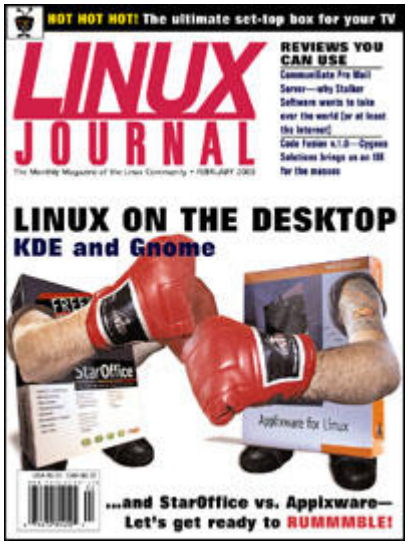
Linux Journal Issue #70/February 2000



Focus

Linux on the Desktop *by Marjorie Richardson*

Features

KDE—The Next Generation *by Kalle Dalheimer*
Ready to jazz up your KDE desktop—get KDE 2.0.

GNOME, Its State and Future *by George Lebl, Elliot Lee and Miguel de Icaza*
The GNOME team bring us up-to-date on the progress of this popular desktop environment.

Artists' Guide to the Linux Desktop, Part 1 *by Michael J. Hammel*
The first in a series by our favorite artist to take a look at the most commonly used window managers.

Office Wars: Applixware and StarOffice *by Jason Kroll*
Office suites are the mainstay application for any OS; Linux has two competing for your business.

Forum

LaTeX for Secretaries *by Jacek Artymiak*
How to survive without Microsoft Word.

Matlab—A Tool for Doing Numerics *by Tobias Vancura*
An introduction to a command-line program for matrix manipulation.

Remind: The Ultimate Personal Calendar *by David F. Skoll*

Reviews

Columns

Departments

Strictly On-Line

Focus: Linux on the Desktop

Marjorie Richardson

Issue #70, February 2000

Today we have two desktops being developed for Linux: KDE and GNOME, with KDE having a bit of a head start.

It has long been agreed that for Linux to succeed in a business environment, it needed to have a user friendly desktop that competed with MS Windows and the Macintosh, and included all the usual applications for the office. Today we have two desktops being developed for Linux: KDE and GNOME, with KDE having a bit of a head start. Both have their supporters and both are in active development by team members. Reports from those teams tell us just what each is up to and where they are headed in order to make our job of choosing one or the other easier.

Linux Journal's publisher, Phil Hughes, feels GNOME should be dropped in favor of getting KDE to the finish line. I feel differently. One of the pluses of the Open Source movement is that we have options. We're not stuck with an environment unsuited to our purposes just because it happens to have a stranglehold on the market. Both of these desktops look good and both have made a good deal of progress toward that finish line. Let's support both and offer the world a choice.

Applications are presently another area of choice—isn't it wonderful that so many are appearing each day? Two office suites have been available to the Linux user for some time now: Applixware and StarOffice. This issue, our Technical Editor and product reviewer Jason Kroll looks at these two products. Next month he will review the spreadsheet XESS. Let him help you make the decision about which product is best suited for your office.

Even with window managers, we get choices. Michael Hammel begins a new series in this issue to tell us about them from an artistic perspective.

Choices—don't you love them? I do.

—Marjorie Richardson, Editor in Chief

Contents

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

KDE—The Next Generation

Kalle Dalheimer

Issue #70, February 2000

Ready to jazz up your KDE desktop—get KDE 2.0.

The KDE team is working full steam on the next release of the K Desktop Environment which is planned for spring 2000, so it is time to look at what the new version will have in store.

When you first install one of the beta releases and use a plain old setting, you will probably not notice much difference between KDE 1.1 and KDE 2.0. However, the more you explore, the more you will find things that have changed. Also, many changes have been made under the hood.

Java Support

Let's start with some of the core components. As a programmer, you might be interested to hear that the library interface has been cleaned up. As a user, this probably won't interest you half as much as the fact that the web browser now supports JavaScript and Java. The Java support is not bound to any particular implementation of the JVM; you can use any fully compliant implementation. We do our testing with the blackdown port of the Sun JDK, but barring some bugs, you should also be able to use kaffe, for example. Also notable is that not just the HTML widget and applications can use Java—any KDE application can now embed a Java applet in its windows. JavaScript support was partly available in KDE 1.1 already, but it was so rudimentary no one really used it. KDE 2.0 features an all-new implementation that is much more complete; it enables you to view 90% of all web pages using JavaScript.

Speaking of the browser, the whole HTML widget, i.e., the part that is responsible for all HTML display, has been completely rewritten. In particular, the display of tables as well as the speed has been greatly improved and the internal structure now conforms to DOM, the Document Object Model put forth by the World Wide Web Consortium (W3C). The HTML display component is

fully HTML4-compliant, and—if the patent dispute between the W3C and Microsoft is sorted out—will most likely support CSS1.

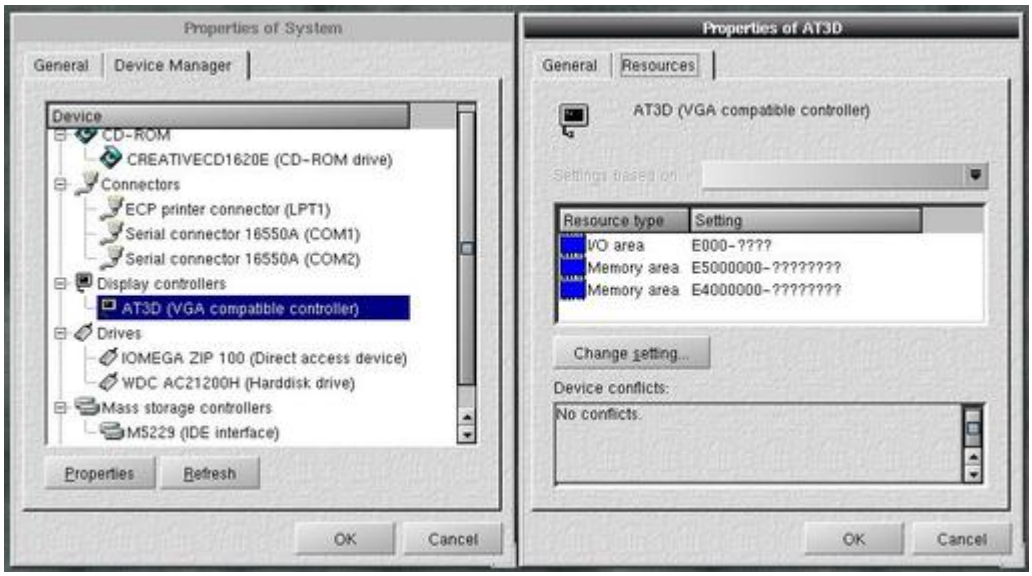


Figure 1. Device Manager

The KDE web browser/file manager has been renamed Konqueror. The name choice is obvious: history tells us the conqueror always comes after the navigator and the explorer. It has also become more flexible. The KDE PostScript viewer **kghostscript** and the KDE DVI viewer **kdvi** can now be used in Konqueror. The same goes for the information and help page viewers, and more can be easily integrated in the future.

Kwin and Kicker

Two more core components have been completely re-implemented: the window manager, now called **kwin**, and the panel, now called **kicker**. **kwin** is much more flexible than the old **kwm**, thanks to its very modular design. For example, it will be possible for applications to ask the window manager for decorations of internal windows. This is useful for applications like StarOffice that use the “window manager in a window” paradigm. Until now, these applications have had to emulate the look-and-feel of one window manager, which was awkward for users who used another manager on their desktop. For KDE 2.0, this feature is still disabled, because it wasn't tested enough. However, the code is there and will make its way into one of the following releases. The author of kwin, Matthias Ettrich, is working with other notable window manager authors on a common window manager specification (the so-called NET protocol), so that window managers on Linux will be easier to exchange in the future. As with KDE 1.1, you can run KDE with window managers other than kwin, but you may lose some functionality.

As for kicker, the new panel, it will be extremely easy to write applets that either run inside it or stand alone. This was already possible with **kpanel**, but with kicker, the burden for the developer has been eased even more. Also, applications can now dynamically add or remove submenus in the K menu at runtime. The taskbar can be included inside the panel (many people wanted that), and the whole panel can be dragged around on the screen. It is much more configurable than the old one.

Customization of the Desktop

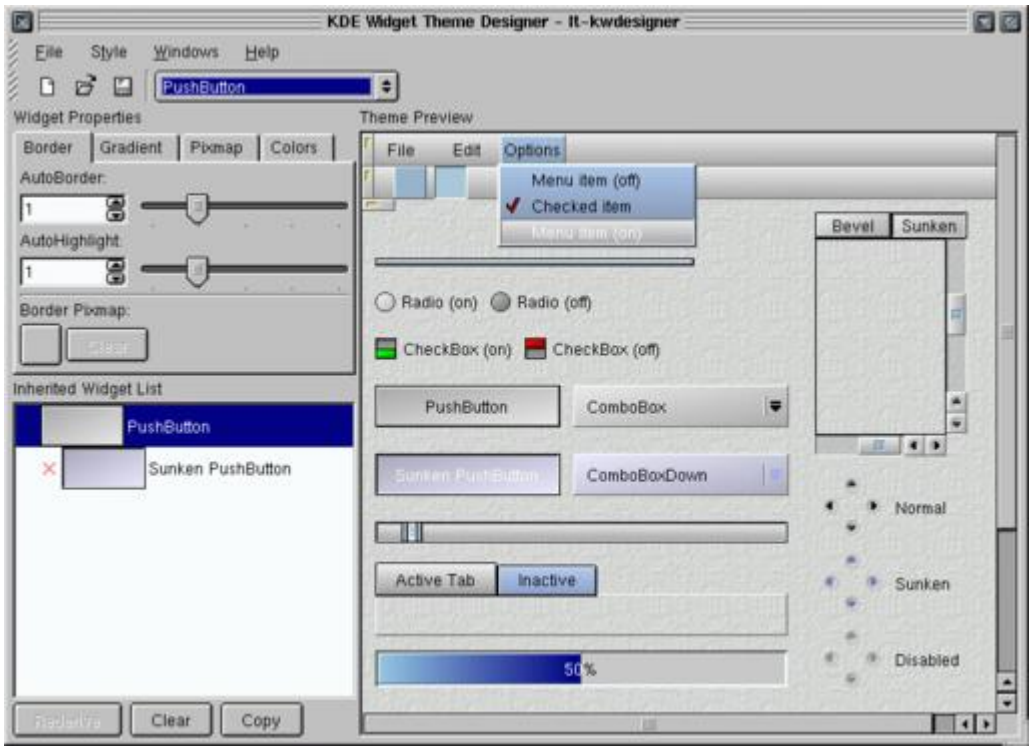


Figure 2. Theme Designer

So far, KDE has had the reputation of being stable and solid, but a bit boring when it comes to looks. This has completely changed. KDE 1.1.2 already contained some theme ability and a theme manager; KDE 2.0 can be graphically customized in any way you like—the possibilities are limitless (see Figure 2). Everything from a very basic and subdued look to a completely Gothic desktop is possible. It's up to you and your artistic capabilities, but of course, there are many ready-made themes for you to try (see http://kde.themes.org/).

Kparts and DCOP

When starting the development of KDE 2.0, the KDE team was focusing on using CORBA, the Common Object Request Broker Architecture, a middleware framework for distributed object communication. However, during development it turned out that while CORBA is a cleanly designed architecture useful for large and high-latency things such as corporate databases, it is not so

well-suited for tasks such as user interfaces where speedy responses are needed. There are several reasons for this. CORBA is distributed in nature, which is very good if you need distributed objects, but more of a burden than a gain if you don't need them—and you don't need them in an office suite. Also, CORBA is too static. In a desktop, objects and applications come and go all the time. This is very hard to track reliably with CORBA.

Thus, the KDE team had to find something else in two areas. KOffice needed a new underpinning for embedding applications into each other and general desktop communication between the control center and the applications. Something more lightweight was needed, and this led to two new inventions that are now used in KDE 2.0: KParts and DCOP.

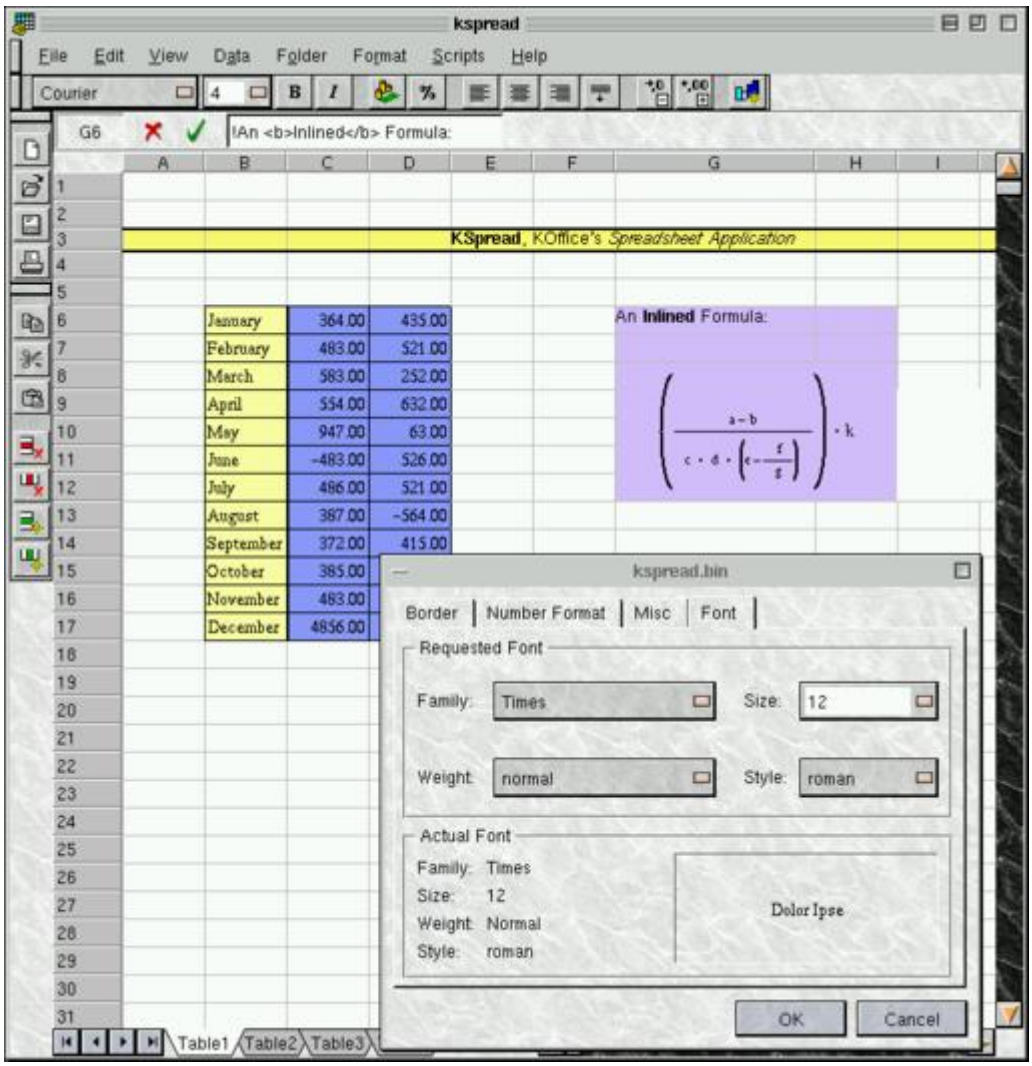


Figure 3. KSpread

KParts is the new object framework used in KOffice for embedding application windows into each other. You can, for example, embed a spreadsheet window from KSpread into the word processor window from KWord, just as you could with the previous development versions. Since all KOffice components are now simple shared libraries and the whole document is handled in the same

process, this happens much faster and smoother than before. Actually, the previous embedding framework was so slow (and buggy) it was hardly ever used. With KParts, there is no speed difference between working on the container part (the word processor in our example) and the embedded part (the spreadsheet in our example). In addition, embedded views can now be tilted, twisted, rotated and sheared, something which to our knowledge is not available on any other platform. I'll talk more about KOffice a bit later in this article.

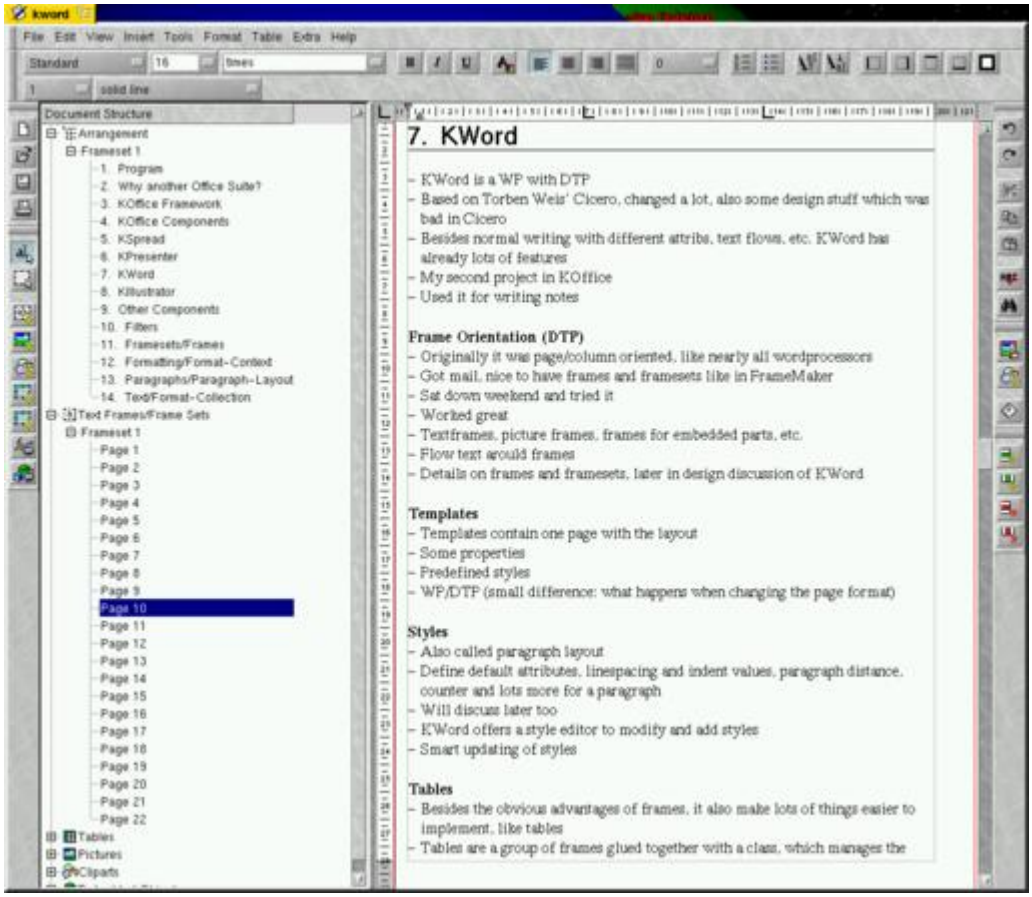


Figure 4. KWord

DCOP, which stands for Desktop Communication Protocol and is pronounced “dee-cop”, is the other new communication facility. DCOP uses a standard mechanism, ICE from X11R6. ICE was written mainly by the X developers to implement session management, but as it turned out, it can also be used for other things. Since it is standard, available in all X11R6 implementations (including XFree86), fast and lightweight, it was a natural choice. For those interested, communication between processes in ICE is done via either UNIX domain sockets, TCP/IP sockets or DECnet. DCOP is just a thin layer on top that handles the marshaling of data and makes sure all communication streams reach their destination. You can imagine DCOP as a traffic cop who stands in the middle of a road crossing and directs the cars into their lanes, so that everyone reaches their destination, no accidents occur and no passengers end up in the wrong cars. (Okay, this is where the analogy breaks down.)

DCOP is used in KDE to unify the different communication means, like X atoms for communication between the control center and the application or TCP/IP sockets. In KDE 2.0, developers need to learn only about DCOP and can forget about X atoms and other communication means. Also, since DCOP is based on the X11R6 standard ICE and is open and documented, it is not bound to KDE. Non-KDE applications that wish to take part in these communication channels can either link directly to the KDE implementation of DCOP or easily implement a compatible implementation themselves. Also, there is already an XML-RPC-to-DCOP bridge, and people are working on a general IIOP-DCOP bridge as well. XML-RPC is especially interesting, because it will allow scripting the whole desktop from just about any language you like, including /bin/sh.

This does not mean KDE has completely dropped CORBA; the KDE team just has a much more judicious view on where CORBA is useful and where it is not. There are areas where CORBA is still deemed useful; one of them is the new KDE multimedia framework **aRts**.

Multimedia

In the current version, aRts is an audio server and mixer that can handle many different incoming audio streams, manipulating, mixing and outputting them to an audio device like an ordinary sound card. At the KDE developers' conference KDE II in early October in Erlangen, Germany, aRts' main author Stefan Westerfeld gave a very impressive demo of aRts mixing two MP3 streams: one of them manipulated with additional sound effects and one synthesized in real time from the MIDI events being played on a MIDI keyboard.

Again, CORBA has to be used judiciously. For a full-blown multimedia application like Brahms, the KDE synthesizer, CORBA is the right tool. For a non-multimedia-oriented application that just wants to play a warning beep when something goes wrong or a lengthy background task is finished, CORBA is way too much, especially since this application would have to link to the full CORBA machinery just to play its beep. This is why aRts also provides a lightweight, raw TCP/IP-based mechanism for playing simple .wav files. (The applications cannot do this themselves because on Linux, only one application at a time can open a sound device.) Also, the KDE team is evaluating another alternative, MCOP, the Multimedia Communication Protocol. By the time you read this, it may have turned out that MCOP is better suited for this task than CORBA, and if it is better, then it will probably already be in use for aRts.

Configuration Files

Now let's go back to some more basic components. The KDE configuration files have always been pure text files. This had the advantage that nothing could go corrupt—you could, if necessary, edit the text files by hand or use a Perl script

to change the configuration files of a few hundred users at once. So, text-based configuration files are desirable. However, a typical KDE desktop has hundreds of these files not only for each application, but also for each MIME type, each service type and so on. This led to slow application startup, because all these files had to be opened and parsed. The solution to this is KSycoca, the KDE system configuration cache. KSycoca is a sort of daemon running in the background and watching for changes in the configuration files. KSycoca reads all configuration files and builds a binary database which can be accessed much faster than the individual, text-based files. However, those are still the authoritative source of all configuration information, so you can edit them by hand or by scripts. In case anything goes wrong with KSycoca, it can simply dump its database, because it is nothing more than a cache and contains no information unavailable elsewhere. Compare that to operating systems with binary registers which can get messed up so badly you cannot boot up your operating system!

Koffice

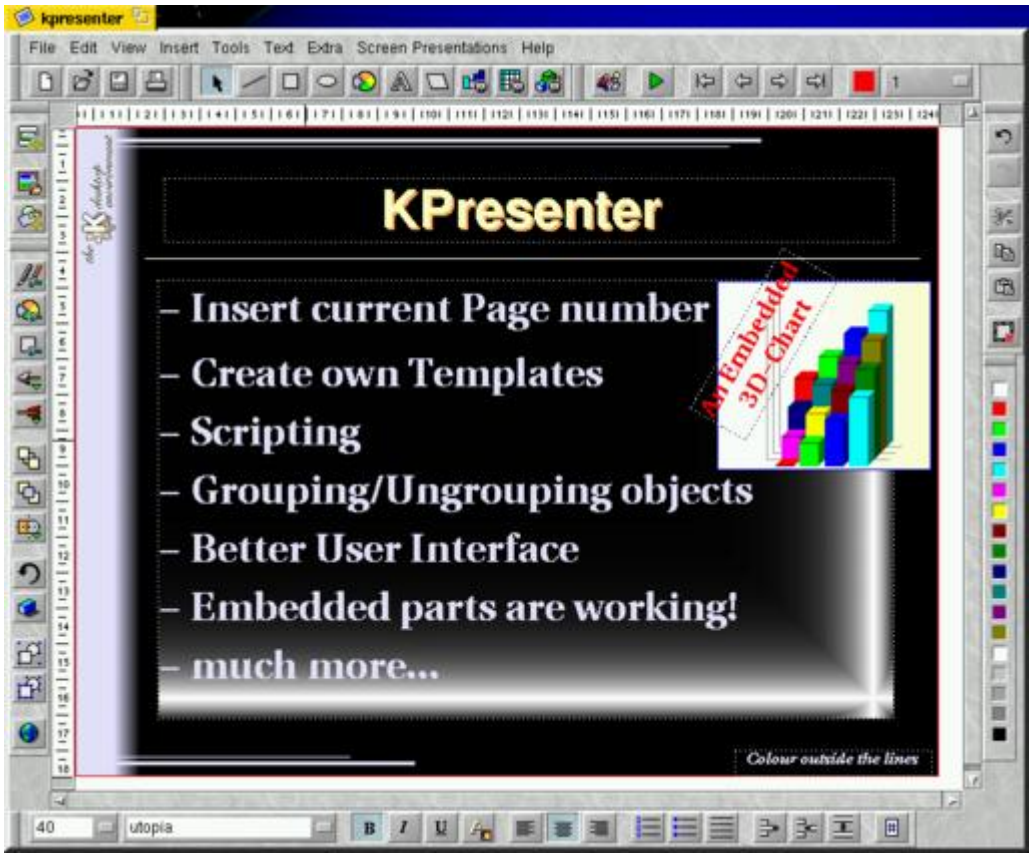


Figure 5. Kpresenter

We have already talked about some internal changes in KOffice. The development of new features is also progressing. Implementing import filters is a very hard task, but Werner Trobin has already made some progress, so there is hope that KWord (see Figure 4) will be able to import Word 97 documents in

the not too distant future. KPresenter (see Figure 5) now supports presentation templates, and KSpread (see Figure 3) has also received many more features too numerous to list here—check it out yourself. However, there are (besides the "old" applications KSpread, KWord, KPresenter, KIllustrator, KChart and KFormula) two new kids on the block. KImageShop is a multi-threaded image-manipulation program that aims at being as powerful as the GIMP, but easier to use. Also, KImageShop has a very clean C++ design that will make extensions much easier. KImageShop will not be ready for public use any time soon, but shows some very promising results. Finally, **kdatabase** is a database front end that aims at making database power more accessible for the end user, and of course, features embedding into other KOffice applications.

KDevelop

KDevelop, the KDE Integrated Development Environment (IDE), is one of the sub-projects that is perhaps making the most progress. By the time you read this, version 1.0 should be out, and it features, among other new things, support for Objective-C and a very nice class tree view. KDevelop's development is a bit decoupled from KDE, i.e., it does not follow the same release plans. However, the KDevelop people put much effort into ensuring KDevelop always works with the current KDE versions.

Documentation

Another important change comes from the documentation team. All the documentation is moving from the LinuxDoc SGML DTD to the more powerful and professional DocBook DTD that most professional publishers use. This move is also being executed by the LinuxDoc people, so that DocBook will become the pre-eminent standard for Linux documentation in the future.

All those changes, new features and new applications, impressive as they may seem, are simply nothing against the one true KDE killer application that has appeared recently: **ky**, the KDE implementation of the classic UNIX **yes** tool!

Kalle Dalheimer (kalle@dalheimer.de) is the president and CEO of Klarälvdalens Datakonsult AB, which specializes in cross-platform software development and technical documentation. He has been a member of the KDE team since its inception and currently works on KChart and (still) cleaning up the libraries.

GNOME, Its State and Future

George Lebl

Elliot Lee

Miguel de Icaza

Issue #70, February 2000

The GNOME team bring us up-to-date on the progress of this popular desktop environment.

The GNOME Project is aimed at making UNIX attractive and easy to use. To help achieve the goals of the GNU project, we want to make sure that users are presented with a full suite of applications, as well as a desktop that enables them to manage their computers effectively. The GNOME team has been focusing on creating a reusable infrastructure of development libraries and tools, along with productivity applications based on this infrastructure.

GNOME Overview

The goals of the GNOME Project can be divided into three areas:

- a full-featured desktop environment
- a set of interoperable applications with a consistent, easy-to-use interface
- a powerful application development framework

The Desktop

The desktop environment is not the set of applications, such as a web browser or a spreadsheet, with which a user interacts with the system to do useful tasks; rather, it is the utilities which provide the user with control over the working environment. As the most immediately apparent part of GNOME, the desktop environment includes the file manager, panel and help browser, as well as other utilities necessary for the day-to-day maintenance of one's computing environment.

The GNOME session begins with the GNOME Display Manager which grants access to the system. The beauty of this process is that the author of GDM rewrote the whole login sequence to be secure and extensible. The GDM code base is designed to be robust and secure.

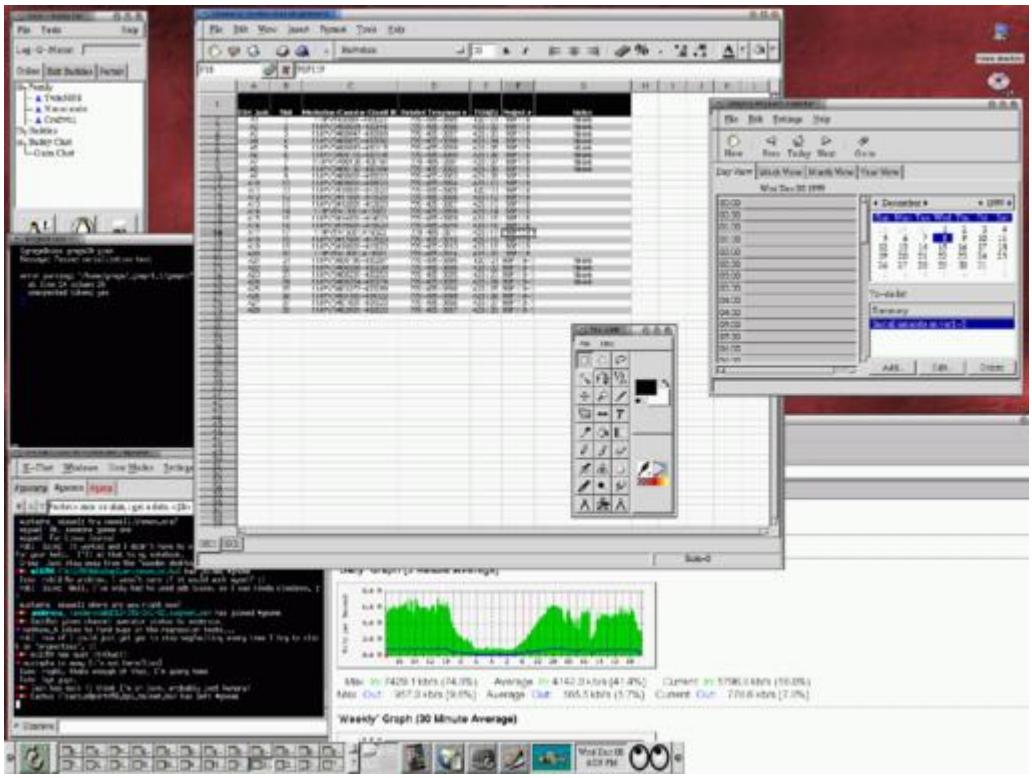


Figure 1. GNOME Screenshot

From there, the GNOME panel and the GNOME file manager provide the desktop functionality to let users launch applications and manage their information see Figure 1).

The GNOME desktop was the first desktop to include application themes. Application themes are a way to make applications look different. People can choose to make their desktop look like other popular systems, or tune it to suit their needs and personal interface desire. The next major release of GNOME will include a new, updated and better integrated theme mechanism, and also theme packages that will affect the entire desktop, not just the applications.

We also integrate the GNOME personal information management system (calendar, address book, task list) with the Palm Pilot, and more systems can be plugged into the system. To learn more about this feature, visit http://www.gnome.org/gnome-pilot/.

The GNOME Workshop

Just being able to choose a screen saver, organize icons, browse application menus and move files doesn't mean you are a productive member of society.

What users want is a set of applications to help them accomplish actual work. This is where the GNOME Workshop project comes in. Many applications not done by the core GNOME team are available, but would be much more useful if they were integrated with each other and the desktop. The GNOME Workshop project wants to make a set of highly integrated applications to do what you need, whether it is managing finances, writing letters or editing a picture. Components of GNOME Workshop that have already reached a functional state include a highly capable spreadsheet (Gnumeric), a word-processing application (AbiWord), and an image-editing application (the famous GIMP). Other component applications are coming along quickly, and news of their releases will be listed on the GNOME Workshop home page.

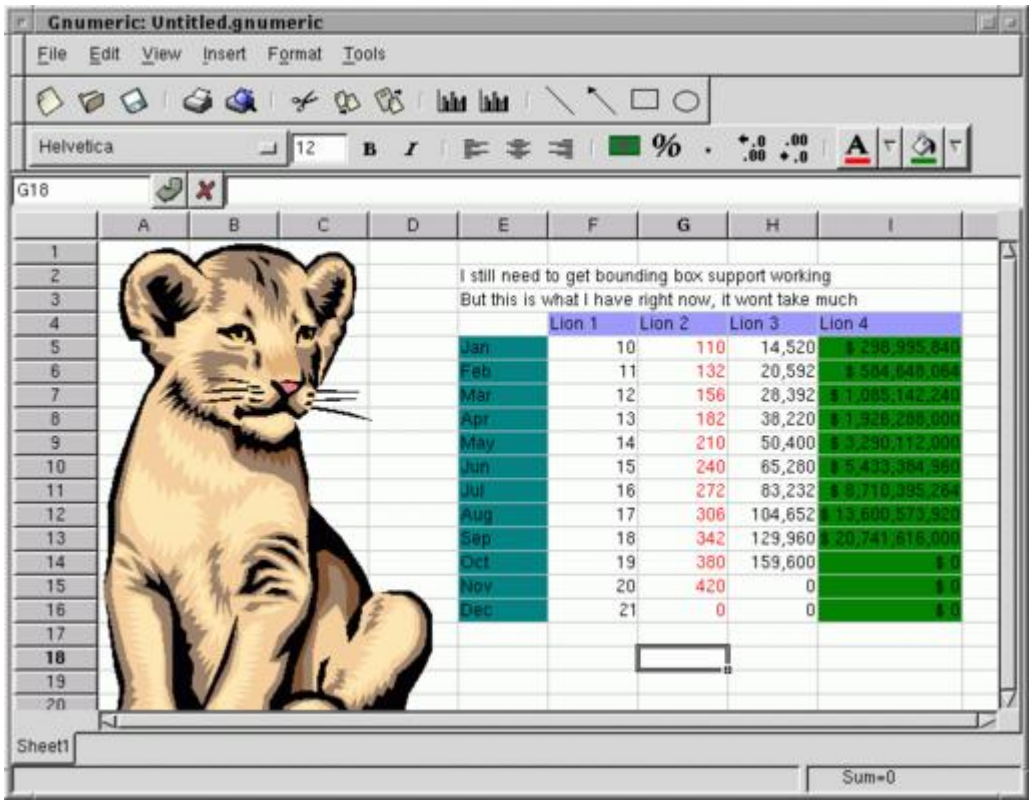


Figure 2. Gnumeric and Bonobo

Developer Goodies

Another important part of GNOME is the development environment. UNIX has not had a history of applications with a consistent and powerful graphical interface. The few graphical applications that existed all behaved and looked a little differently, usually did not have a powerful interface, and were not easy for their developers to write. GNOME addresses this last need by simplifying the development of applications, allowing the creation of easy-to-use and powerful graphical interfaces.

GNOME provides a high-level application framework which frees the programmer from having to worry about the low-level details of graphical

application interfaces, allowing him to concentrate on the actual application. Glade, a tool for user-interface design used by many GNOME applications, takes this concept a step further by allowing graphical creation of a program's user interface. The Libglade library allows user interfaces to be created at runtime from the XML interface description files saved by Glade. GNOME also recognizes that not every programming language is useful for every kind of job. We paid special attention to making the GNOME APIs easy to wrap and export to other programming languages, to let people develop their GNOME-based applications in their language of choice.

In addition to C, which the core GNOME libraries are written in, there are bindings for many languages, including C++, Objective C, Guile, Python, Perl, Ada95, Tom, Pascal, Haskell and others. Java bindings are in development; when coupled with **gcc**'s ability to compile Java code, Java may become a viable alternative for GNOME programming.

The GNOME Workshop components

The Gnumeric Spreadsheet

The objectives of the Gnumeric spreadsheet (see Figure 2) are to include all the features you can find on proprietary spreadsheets. Gnumeric has implemented most of the built-in functions available in Excel 2000, and by the time you read this, it should be complete.

Users can write their own functions in Python, Perl or Scheme, and access them through the Gnumeric engine. This enables people to adapt Gnumeric to their needs. Of course, if you like to use C, C++ or assembly, you can also hook up your functions to the spreadsheet.

We are copying many good ideas and usability features from popular spreadsheets and including them in Gnumeric.

Gnumeric also serves as a test-bed for the various GNOME technologies: CORBA, the Bonobo component and document model, and the GNOME printing architecture. Gnumeric was also one of the first applications that used libxml (the Gnumeric native file format is XML-based).

Gnumeric's rich set of import and export filters make it quite a valuable addition to your tool box, and if you are looking for an application to load your Excel spreadsheets, look no further: Gnumeric has the best Excel importing functionality available for GNU/Linux as tested by *LinuxWorld*.

Not only can you define your own functions for spreadsheets in the languages mentioned above, but you can actually script the entire application from any language that supports CORBA. Anything the user can do by using the spreadsheet user interface can be done through a CORBA communications channel, enabling you to use Gnumeric as a reusable engine for your applications.

The AbiWord Word Processor

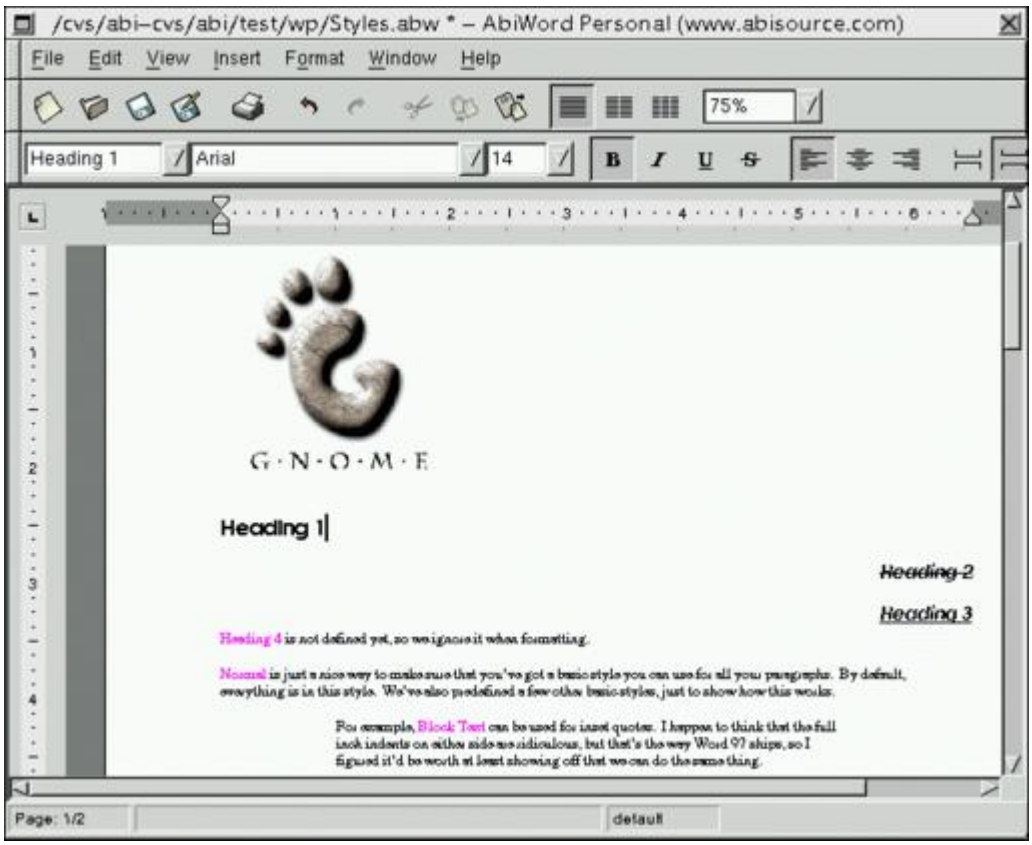


Figure 3. AbiWord with GNOME Support

AbiWord (see Figure 3) is a project led by Source-Gear. It is a cross-platform word processor that can import your Word files. The word processor has gotten attention from many communities, since they can run the same word processor across UNIX, Win32, BeOS and MacOS. Their web site is at http://www.abisource.com/; you can download it from there. Abi contains a number of interesting features found in the proprietary word-processing equivalents.

The GIMP

The GIMP is the de facto standard for image editing, photo retouching, image authoring and composition. It is hard to list all the features, as it is one of the most successful free software projects. The GIMP is among the most actively developed applications, and Yosh Manish acts as its maintainer and coordinator.

Dia

Dia is a diagram application that can be used as a replacement for the popular proprietary application “Visio”. Dia supports various kinds of high-level objects for drawing: UML, Network, databases, circuit objects, flow charts and more. Dia is easy to extend with new object collections, as the various objects are defined using an XML-based file format.

It has quickly become the tool of choice for GNOME developers for generating diagrams and communicating graphical information with other developers. The Dia team consists of seven programmers and is commanded by Alexander Larsson. The Dia community is very active.

GNOME DB

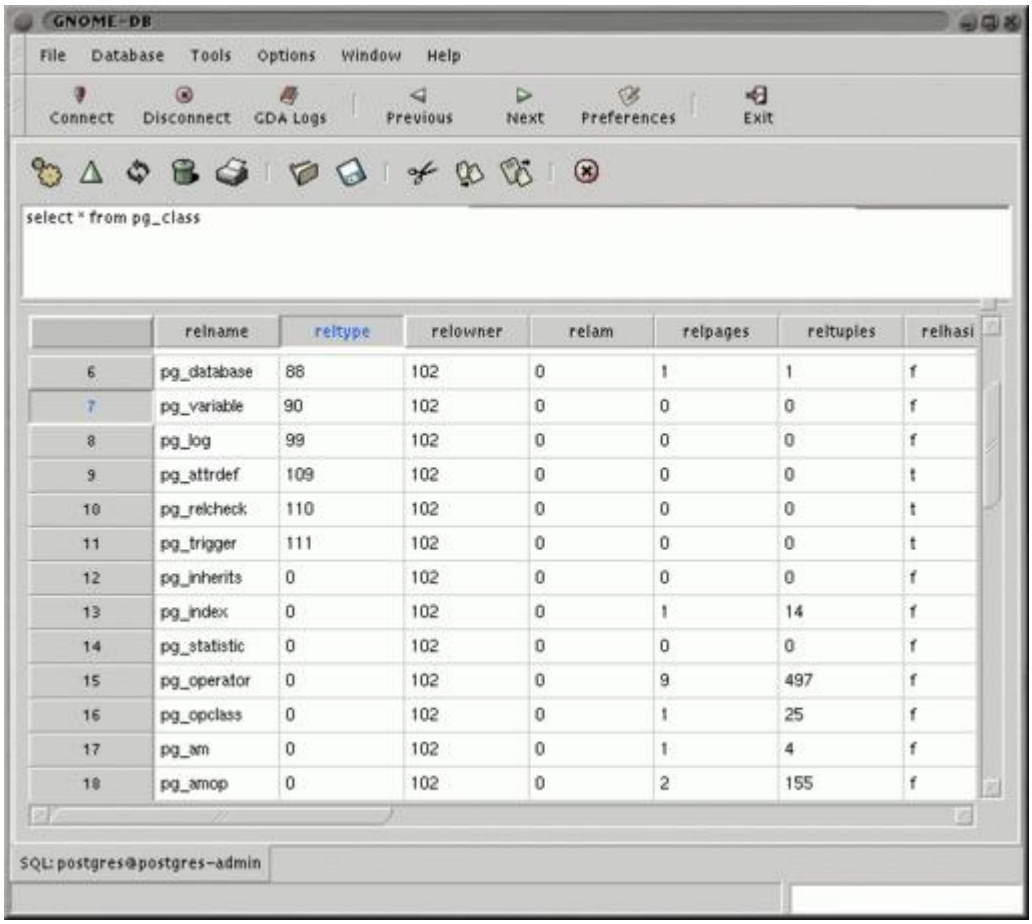


Figure 4. GNOME DB User Interface

GNOME DB (see Figure 4) is a framework for creating database applications. It provides both a common API with pluggable back ends to various database sources, and specialized widgets for handling various database tasks. The back ends, in the typical GNOME tradition, are based on CORBA. Michael Lausch and Rodrigo Moya are the main developers of this project.

GNOME-DB is composed of three separate, independent layers. The lowest level contains the database servers, which are CORBA servers that map the database-specific APIs to the GDA model. The middle layer is the GDA client library, which sits on top of the CORBA interface to allow an easier access from client applications. The middle layer also contains the GDA UI library, which provides a way of easily accessing the client library from GNOME applications. The top-level layer is composed of the applications making use of the middle-level layer, such as the Rolodex application and the SQL front end.

Gill and Eye of GNOME

Gill (GNOME Illustrations) and Eye of GNOME (image viewer and organizer) are the most recent additions to the suite of productivity applications for GNOME.

Gill is particularly interesting, since its native file format is the Web Consortium's Scalable Vector Graphics (SVG). Internally, the application has been organized around a Document Object Model (also a W3C standard). Gill is still in its infancy, but given that it is based on a strong foundation, it is already a fairly powerful display.

Eye of GNOME is also in its infancy, but has served as a test bed for various new GNOME technologies: our new imaging system, various new GNOME Canvas improvements and the new model-view-controller-based widgets. It is our image viewer of choice right now and amazingly fast, too.

Development Tools

CORBA

The lack of a standard, freely available system for component programming is a problem present in the UNIX world today. One of the issues being addressed by the GNOME project is providing such a framework. The GNOME framework is based on the CORBA object model. During the design of the GNOME component interfaces, we tried to address a range of needs:

- Automation: allow applications to be remotely controlled. People should be able to launch and control GNOME applications remotely. This is achieved by the GNOME Object Activation Directory (GOAD), the supporting libraries (the GNORBA libraries) and making use of the CORBA facilities for binding CORBA to scripting languages.
- Compound document creation: it is important to design and implement the GNOME applications in such a way that they will let the user create compound documents (those with contents that may have been produced by different tools).

- In-place document editing: the next step in compound document creation is providing ways to edit embedded documents inside a container application. This means it should be possible to make changes to an embedded spreadsheet document inside a word-processing document in a seamless fashion: it is important to make this integration simple and easy to use.
- Component reuse: filters and pipes proved to be important building blocks in UNIX, but they are very limited: the flow of control usually goes in a single direction, and the protocol used to exchange information is too simple to meet today's needs.
- Desktop integration: the GNOME desktop deals with CORBA interfaces to services. As far as an application is concerned, only the published interface a program provides is used.

In GNOME, interfaces for specific tasks play an important role. It is up to the user to choose which implementation of those interfaces he uses. For example, the Mailer interface is implemented by the GNOME Balsa mail reader, but it can also be implemented by Emacs RMAIL, Emacs GNUS or the Mozilla mail reader. If any of those provide the CORBA Mailer interface, they will work properly with the GNOME desktop.

All these needs can be met by the use of CORBA, an OMG standard (http://www.omg.org/). We are using ORBit as our CORBA implementation, with very good results. ORBit was designed to be small, fast, robust, reliable and efficient. Many times people hear the word CORBA, and they immediately think “bloat”. This is not the case with ORBit. ORBit is thin: for most applications, the working set of ORBit's ORB is around 30K, which makes it suitable for embedding in almost every application. This is exciting, because applications such as Gnumeric and a few others export their internals to the desktop as highly specialized reusable components (think “UNIX filters on steroids”).

CORBA has proved to be very useful, as we can use it to run regression tests on our applications from a Perl script using Owen Taylor's wonderful CORBA-Perl (people.redhat.com/otaylor/corba/orbit.html). This just happens to be one of our favorite CORBA bindings for a scripting language, but you can get CORBA bindings for almost every language.

By using CORBA as our foundation, we ensure interoperability with existing systems, and anyone who supports CORBA can talk to our applications. CORBA in GNOME plays the same role COM plays in the world of Microsoft Windows.

Bonobo

Bonobo is the GNOME foundation for writing and implementing reusable software components. Components are pieces of software that provide a well-defined interface and are designed to be used in conjunction with other components. In the Bonobo universe, CORBA is used as the communication layer that binds components together. More about component programming can be found in the Bonobo documentation.

In short, Bonobo provides the following features to an application developer:

- An infrastructure for creating reusable components as full-blown processes, shared libraries or remote processes.
- An infrastructure for reusing existing components. For example, you can use Gnumeric to provide data-entry facilities in your own application or as a computation engine.
- An infrastructure for creating persistent controls.
- An infrastructure for creating compound documents. This means not only reusing existing applications in a vertical application, but creating single documents composed of various parts: spreadsheets, equations, graphics and so on.

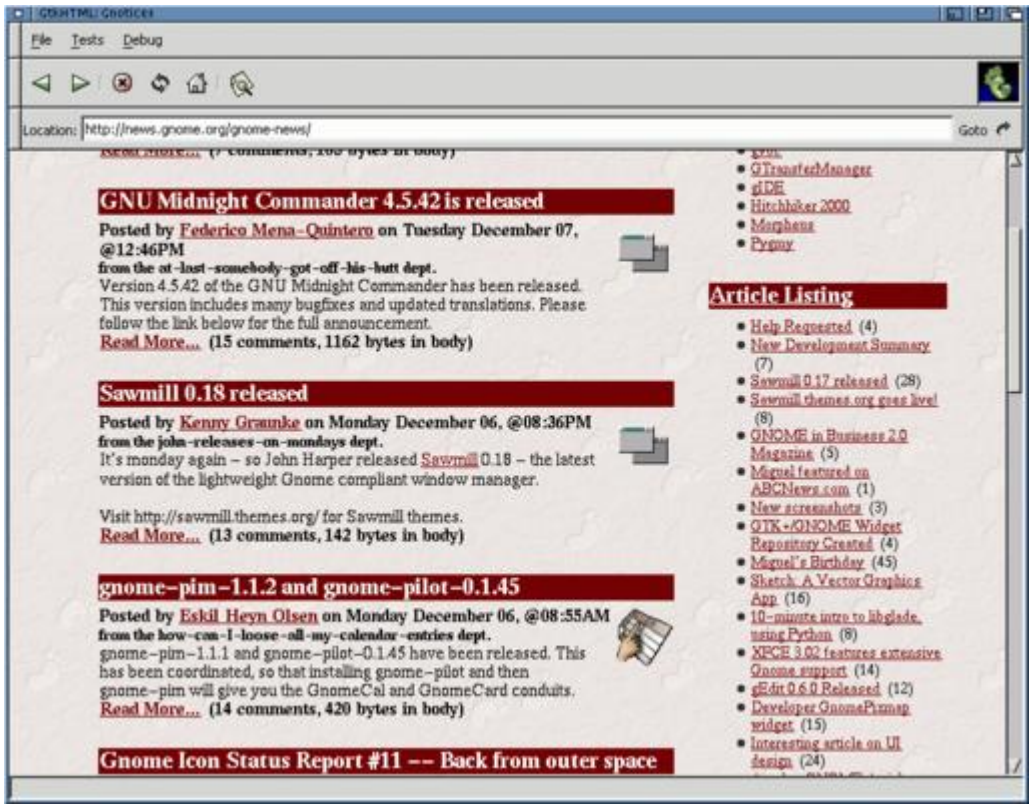


Figure 5. Updated GNOME Screenshot: New HTML Engine

Image Handling

GNOME 1.0 utilized the Imlib software library for image loading and manipulation. While Imlib fits the needs of some applications, its design does not mesh well with the typical GNOME use case. As a replacement, libart, an RGBA image-manipulation library, and GdkRGB, an API to allow high-performance display of RGB images, are integrated together in gdk-pixbuf, an image-loading library that solves the problems of Imlib and adds features such as anti-aliasing and high-quality rendering.

Canvas and Libart

GNOME 1.0 included libart and the anti-aliased canvas, but it was not used very much, and the anti-aliased canvas was marked as unstable. In the future, GNOME will use libart and the anti-aliased canvas much more. The new GNOME panel and GNOME pixmap widget already use libart and gdk-pixbuf to provide anti-aliased icons.

Glade and Libglade

Glade is a GUI designer which is currently being used in much of the new GNOME development. Normally, Glade will generate source code to build the interface you create. However, the truly revolutionary and useful way to use Glade is in combination with Libglade.

Libglade is a library that will load a saved Glade project and build the interface for you—on the fly. This means you could conceivably have different interface files for different languages, application modes or themes. It also gives the user the ultimate power of customization, since the interface of the application can be modified without any programming skill.

Various GNOME applications are now designed with Glade, and this substantially reduces their development time. It is easy to prototype, easy to customize and easy to extend. The joy has arrived.

Printing

UNIX printing is both a blessing and a curse, and while it may be flexible, it has always been hard to set up, and output quality was often low on non-PostScript printers. In addition, most applications didn't even include printing support, because no convenient, unified API was available for printing on UNIX. GNOME Print is a library which allows developers to easily add printing capabilities to their applications, and users to quickly and easily access all the output parameters through a consistent graphical interface.

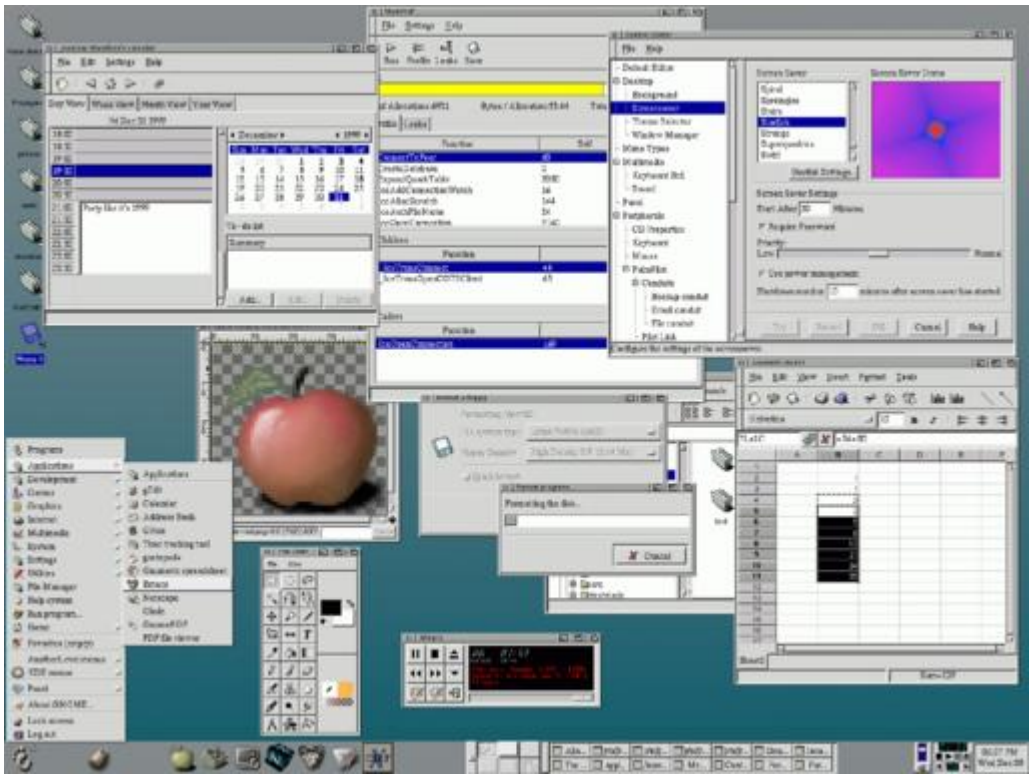


Figure 6. GNOME Desktop, Using Anti-Aliasing

The GNOME Print imaging model is based on PostScript, with two extensions: anti-aliasing and transparency. Please note that I said “an imaging model based on PostScript”, not PostScript. You have the same imaging model, but the way you print is by using an API exported to your favorite language.

Currently, GNOME Print includes a PostScript driver, an on-screen driver (for doing previews), meta-file drivers (for storing printed information, transferring it, and rendering it on a scaled context) and a generic RGB driver (on which we will build the per-printer actual printer drivers).

As you might expect, we do reuse our technologies. The rasterization engine used in the canvas is the same rasterization engine used for the on-screen preview and for rasterizing the output for the native printer drivers. If you are interested in working on the native drivers, we most definitely welcome your help.

Work in Progress

The GNOME Project is not resting on its laurels. While the 1.0 release series addresses the basic need for a UNIX GUI, portions of the existing implementation need refinement, as well as additional features which are becoming essential on a modern desktop.

At the same time, the GNOME 1.0 API will still be supported through compatibility libraries, so that the migration of existing source code to the new

libraries will be painless. This will allow developers to focus on the application development instead of having to worry about following GNOME API changes.

The New File Manager

The file manager is being completely redesigned. The new version will feature an asynchronous, network transparent virtual file system which is usable from all applications independent of the file manager. This will make it easy to have applications that are network aware, and it will make network administration a snap. This step forward from traditional virtual file system libraries, which are not asynchronous, allows better responsiveness from graphical applications.

Panel Improvements

In addition to the file manager, another highly visible part of the desktop is the panel. It is a place to put icons to launch applications, access application menus, manage open windows and run small utility applications that display the status of the machine, play CDs or display time. We recognize that people are not the same, and different people like to work differently. That is why the panel is highly customizable, and this customization has been greatly extended in the new version. The panel supports many new and different modes of operation. For one, you can select a size for your panels depending on your tastes and screen size. There are also new modes to place panels anywhere along an edge, and even anywhere on screen. In addition, all icons on the panel are now anti-aliased, and external applets have the ability to use anti-aliasing for their display. Many other smaller additions make the panel more configurable and easier to use, and your GNOME experience more pleasurable.

Configuration Data Storage

Another noticeable improvement coming to GNOME is GConf, a new configuration API and back end. This will add the features not provided by the simplistic configuration API in GNOME 1.0. It will make it easy to plug in different back ends for the actual storage, so you can change how and where the data is actually stored without touching the applications themselves.

Object Activation Framework

Since many GNOME applications utilize CORBA, a framework for locating and activating these applications is necessary. OAF, the Object Activation Framework, provides a simple method for finding and running the CORBA objects available on a computer system. Distributed operation is supported, allowing activation of objects on a network of computers being used in a GNOME desktop. The flexibility of OAF enables it to be used outside of GNOME

programs, allowing non-GNOME CORBA applications to be utilized alongside GNOME applications.

Pango: Reaching Out to the World

Supporting the various human languages is a complex task, not because of the difficulty, but because of the wide range of communication systems humans have came up with in the last few thousand years.

Since our goal in the GNOME project is empowering users and giving them a chance to run free software, we have to make sure everyone on this planet can use our tools with their language, and that our applications can be used by all people.

Gtk+

Gtk+ is the toolkit used by GNOME and the various GNOME applications. The Gtk+ team led by Owen Taylor and Tim Janik is making steady progress towards the Gtk+ 1.4 release.

The major highlights of Gtk+ 1.4 include flicker-free drawing, better internationalization support (through Pango), and integration of the BeOS and Win32 ports.

Applications written against the Gtk+ 1.4 and GNOME APIs will be portable to Windows and BeOS. (Keep in mind that GNOME/Gtk+ applications talk to a windowing system layer called Gdk, which is independent of X11. This is why it is simple to port them to other architectures.)

User Interface

The people who brought you GNOME are programmers, and most are not graphic designers. They do not have all the experience required to make the best user interface possible. It is hard to write good user interfaces, and we are trying to address this. The GNOME User Interface team is responsible for redesigning the look and feel of various GNOME applications by studying the current application failures and using what is good from other applications and systems.

The Importance of Free Software

GNOME is part of the GNU project, and it is free software (some people call this open-source software), created for the people by the people. We want to make software that grants users various freedoms.

It is not software owned by some large faceless company. You, therefore, are the person best qualified to contribute to its improvement. Although programming help is extremely welcome, you don't need to be able to program in order to help. Documentation, translations, web site maintenance, packaging and graphic design are just a few of the many areas where people are already making contributions. If you dislike an aspect of GNOME and want it improved or want a totally new feature added, the way to make this change happen is to start contributing.

You might think any contribution you could make would be unimportant, but if many people make small contributions, the result is a large increase in the progress being made. Your efforts for GNOME will continue to make the power of UNIX easily accessible to average users.

Resources

Thanks

Contributing to GNOME

George Lebl (jirka@5z.com) is an independent consultant in San Diego, California. He has been involved in GNOME since the very early days.

Elliot Lee (sopwith@redhat.com) is a programmer for Red Hat Advanced Development Labs at Red Hat Software.

Miguel de Icaza (miguel@gnu.org) is a programmer at Helix Code, Inc.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Artists' Guide to the Linux Desktop, Part 1

Michael J. Hammel

Issue #70, February 2000

The first in a series by our favorite artist to take a look at the most commonly used window managers.

A few years ago, I did a four-part series on the GIMP for *Linux Journal*. Those articles served as the basis for my book, *The Artists' Guide to the GIMP*, which was published about a year later. I like to write about using Linux from an end-user perspective. Developers are expected to be able to dig and scratch for information. End users need a little more hand-holding. Since the GIMP helped me pretend to be artistic, I tend to approach my writing from what I think an artist might want. In this way, I can show that Linux is more than just a place to dig and scratch. You can actually get real work accomplished here.

In this new series of articles, I'm going to look at the coming of age of the Linux desktop. In particular, I'll talk about window managers. The desktop has not been the strongest arena for Linux. Many users have complained that the desktop is too clunky or lacks basic features. But the problems that have been pointed out simply require time—time for Linux to mature and developers to write code. What Linux currently lacks in basic features, however, it makes up for in pizzazz. The Linux desktop is an extremely personable place to work.

Some might say it's fluff. Others refer to it as eye candy. Call it what you want, the Linux desktop has loads of personality. Window managers are a major part of this, although they should not be mistaken for the entire desktop environment. In fact, this is where we'll start. Just what is the difference between a window manager and a desktop environment?

GNOMEs and Sprockets and Bears, Oh My!

The current buzz in the Linux desktop world focuses on two major players: GNOME and KDE. In nerd parlance, these are desktop environments. GNOME comes to us from Miguel de Icaza, et al. via support from Red Hat, and is based

on GTK, the software libraries that are used for the buttons, sliders and so forth you see in GIMP. KDE is based on QT and comes from Matthias Ettrich and the KDE Development Group. Both QT and GTK are known as widget sets. Usually, you use widget sets to build applications, like GIMP or KIllustrator, but GNOME and KDE go beyond just providing applications. They provide infrastructure—underlying rules and guidelines for how applications should look and act. They provide software libraries for allowing applications to communicate with one another. This is how things like drag and drop get implemented. Without GNOME and KDE, dragging a file icon into a wastebasket or onto the printer icon wouldn't be possible.

There are other environments besides GNOME and KDE. The most widely used environment on UNIX systems is probably CDE, the Common Desktop Environment. CDE is based on a commercial package which uses Motif, and is included on just about every major UNIX workstation these days. Versions of CDE are available for Linux, but most people have stuck to either KDE or GNOME because they have lower licensing fees. Motif used to be very expensive, and although its price has dropped considerably, it's still relatively expensive to license for Linux distributors. Since the desktop environment is expected to be part of any OS distribution, Linux distributors had to either charge extra for CDE or work with KDE and GNOME. Their choice was obvious.

KDE and GNOME both offer very similar features. Each provides a panel—a bar along one edge of the screen used to launch applications, show system status and allow embedded applications—and can be used with any compliant window manager. Session managers and CORBA support are less visible than the panel, but are probably more important to users. Session managers allow you to start an application at the same place you left it the last time you used it. CORBA allows for interaction between applications. Session management and CORBA are provided as a set of rules (via programming libraries, if you care about that) applications must follow in order for them to work properly in these environments.

Desktop environments provide the entire desktop experience for the user. This ranges from pagers and drag and drop to standardized applications. Both KDE and GNOME have file managers—graphical interfaces for managing files. Window managers don't usually provide such a tool. Calendars, word processors and spreadsheets are also applications that exist for use with KDE and GNOME, but which are not considered part of those environments; these are just ordinary applications which comply with the rules KDE and GNOME provide for interoperability. Although you'll hear both GNOME and KDE talk about these tools as part of their environments, don't be fooled—you can pick your own spreadsheet application; you don't have to use the ones that come with KDE or GNOME.

One feature that is extremely useful is the pager. The Linux desktop allows you to have off-screen areas you can jump to. These are called virtual desktops, and each desktop often has multiple pages. Essentially, each of these is like an extension of your visible screen—extra screen space with which to work.

Both KDE and GNOME provide their own pagers, as do most window managers. In order for a window manager's pager to work properly in its respective environment, the window manager needs to be compliant with that environment. Fortunately, the window managers we'll be discussing are at least partially compliant in one or both environments.

There are a few other differences between GNOME and KDE. KDE already has a built-in help system, while GNOME just recently released one. GNOME is considered by some users to be a little top-heavy—it can use a lot of memory, for example. It's also less stable than KDE. GNOME, as a project, is about a year younger than KDE, but both have strong development teams working on them. In general, you'll probably be fairly happy with either environment. In fact, if you're like me and don't use drag and drop, file managers or worry about interoperability between applications, you can live without either KDE or GNOME. I don't run either—at least not yet.

Window Managers

Where KDE and GNOME provide that overall desktop experience, window managers do something a little more basic. They provide the facility which allows you to move your windows around the screen, iconify and maximize them, and kill off those applications which just refuse to go away. Window managers also determine the look of the windows—the borders and title bars, the background images and so forth. The configurable framings you specify for your windows are called themes. It's the pizzazz we talked of earlier. For artists and anyone interested in personalizing their private desktops, the window manager is the focus.

KDE and GNOME provide default window managers: kwm for KDE and Enlightenment (or, more recently, Sawmill) for GNOME. A desktop environment wouldn't be very useful without a window manager. It would, in fact, be almost useless. The relationship between desktop environments and window managers is fairly symbiotic. However, you don't have to use the default window managers provided by these environments. You can run any window manager—and there are several—with either environment. In order to make use of all the underlying features of KDE or GNOME (such as session management), you need to run a window manager which is compliant, and not all window managers are. In fact, very few are fully compliant with either KDE or GNOME, much less with both of them. For this reason, I'll be focusing on three window managers that are compliant with at least one of these environments

and which provide support for themes: Enlightenment, Window Maker and AfterStep.

Like their desktop environment counterparts, window managers usually provide extra features which border on being applications on their own. Pagers fit in this category. Whether you use the environment's pager or the window manager's pager depends mostly on choice, although use of the panel in either KDE or GNOME may force you to use their pagers as well. Window managers also provide management of desktop icons. In most cases, this is hidden from the user—you get a graphical tool for specifying where the icons get placed (left side, right side, restricted to a specific region, etc.). In some window managers, you get a visible box for the icons, called the icon box. Enlightenment, for example, uses a very obvious box by default which can be configured to be essentially invisible.

Themes

After all this technical jargon, we get to the part which artists will likely love the most—themes. Themes do nothing more than specify the look of backgrounds (on the root window or in any other window), window borders, icons, and in some cases, the applications themselves. It's fluff, but interesting fluff—a personalization of the computer on which you spend so much of your life.

Window managers provide most of the features you need for themes. KDE and GNOME have support for themes, but they do it differently. KDE's default window manager, kwm, supports themes like any other window manager. GNOME gets its support from GTK, the widget set I spoke of earlier. GTK is interesting because it allows you to give the same theme to any application based on it. In other words, GNOME applications, which are generally based on GTK, can all be changed at the same time by changing the GTK theme. This sort of change affects not just the window borders, but the buttons, menus and other visible pieces of an application. KDE, Enlightenment, Window Maker and AfterStep change only the look of window manager features. They don't change the look of application buttons, menus and so forth.

KDE allows you to change the current theme on the fly using its Control Panel. GNOME does too, using its own Control Center. Both of these are accessed from their respective environment's panels. Window Maker, AfterStep and Enlightenment allow you to change the theme on the fly as well by using tools that may or may not be included in their respective standard distributions. You can also change the themes by hand if you're so inclined. However, themes are a relatively new phenomenon, and standardized ways of defining and installing them have not been set. In most cases, you'll want to use the graphical tools for installing the themes. Creating new themes is rather complex and very specific to each environment or window manager. In this series of articles, I'll stick to

discussing installing themes. In the future, I'll look into writing about making your own themes, once the process of doing so isn't quite so convoluted.

What to Look For?

Since window managers keep less information on running applications than the desktop environments do (via their session management features), they tend to take up less memory. Of course, there are exceptions to every rule. AfterStep has a small memory footprint, but because Enlightenment is so graphics-intensive, it can have a huge footprint. On your desktop, which has a gigabyte of memory so you can work on huge print images in the GIMP, it probably won't matter which window manager you use based on memory usage. But it will make a difference on memory-limited laptops. Memory usage is something to be aware of, depending on where you're working.

A good pager is essential. I don't work with GNOME or KDE (they don't provide features I need right now), so I can choose any window manager. For a long time, I used only FVWM2, specifically because of its pager. The desktops and their pages were displayed as small boxes right on the screen, and I could drag windows from one desktop to another simply by dragging it from the pager into the main screen. The pager also showed the names of the windows (from their title bars), albeit in truncated form. The sizes of the windows in the pager were proportional to their actual sizes in their desktops. Enlightenment now has a very similar pager which I find very attractive in its look and usage. Window Maker, on the other hand, has a pager which is just a button that bounces you around the desktops, but I can't see what's in each of them until I get there. I also can't divide the pager into multiple desktops with multiple pages in each. In Window Maker, I get essentially one page per desktop. Whether you consider this good or bad is a matter of personal taste. What is important is that you consider the pager in the window manager you choose, since you will be using it quite a bit.

On-line help is useful, but only initially. Both desktop environments have on-line help built into them. Enlightenment has its own help system; Window Maker and AfterStep have HTML-based help. The usefulness of this depends on how long it takes you to become familiar with the environment. In my opinion, if you need to refer to the help system for months on end, the system is far too complex and you should consider looking at another window manager.

Now, if we go beyond these basic necessities, we get into the niceties. It would be helpful if your window manager had the ability to add dockable applications. These are sometimes called applets, epplets and various other things (everyone has their own terminology, apparently). It's an application with a very specific function—e.g., monitoring the time you've spent on-line with your ISP—that runs in a small icon or, more commonly, in a panel. You'll want your window

manager to provide features which can maximize screen space, too. Things like window shading—rolling a window up into its title bar to hide it, and unrolling it back down again. This reduces the visual clutter of your desktop, but it's certainly not required for the window manager to be useful. A pager that can be hidden would also be nice. Enlightenment provides this, but Window Maker doesn't because its pager doesn't take up as much space as Enlightenment's can. Of course, you'll want theme support. After all, pizzazz is the whole point.

Changing Window Managers

How do you use your window manager of choice? If you're using KDE or GNOME, you're probably logging in and everything is already set for you.

With GNOME, you can specify a separate window manager just by editing your $HOME/.Xclients file. You probably don't have one of these to start, but you can create it by hand. For example, to run GNOME with Window Maker, your .Xclients file would look like this:

```
gnome-session&
wmaker
```

KDE is a little different. It needs to start a couple of different programs. If you have the **startkde** script (available on Red Hat 6.*x* or similar systems, possibly on others), you can replace kwm in that script with the window manager of your choice.

Whether you start X manually (using **startx**, for example) or have it start automatically when you log in, modifying your .Xclients file should be sufficient to get your window manager going.

So that's an introduction to desktop environments and window managers. It can seem complex, but it really isn't hard to understand after you've used it for a while. As with anything new, unfamiliarity breeds contempt. You just need some time to make friends with this new world.

In the next article, I'll cover the Enlightenment window manager in depth. This will include issues on building, installing, configuring and using it. That will be followed by articles on Window Maker and AfterStep. In the end, you'll become familiar enough with these so that you can pick and choose from the many window managers available. At that point, your computer will no longer be someone else's idea of a desktop. It will truly be yours.

Michael J. Hammel (mjhammel@graphics-muse.org) is a graphic artist wanna-be, a writer and a software developer. He wanders the planet aimlessly in search of adventure, quiet beaches and an escape from the computers that dominate his life.

Archive Index Issue Table of Contents

Advanced search

Office Wars: Applixware and StarOffice

Jason Kroll

Issue #70, February 2000

Office suites are the mainstay application for any OS; Linux has two competing for your business.

We humans love glowing boxes. Monitors, TV sets, suns, moons, lanterns, candles—it all goes back thousands of years, when we sat around in tribes staring into the fire. Computers? As long as we're comfortable while staring into the glowing box, it matters little what exactly we're doing. The light is comforting, hypnotizing. Still, some crazy person got this idea that computers ought to have a use, nay, a killer app. Now we're cursed with spreadsheets, word processors, databases, development tools, office suites, mathematical packages... Well, Scott McNealy doesn't want your money (at least not right away), so we might as well make the best of it.

I must admit, the “warring office suites” cover has an air of commercialism to it, but it's not far from the truth. At the same time, it doesn't tell the whole story—Sun and Applix have their minds on bigger things. It is true that on one level, we have a price war in a particular market (cross-platform office suites for Linux/UNIX), but Sun's plans for StarOffice and Applix's ideas for Applixware are fixed on a less immediate horizon. Before we look at the offerings of these respective packages, it's worth the effort to find out where these products are heading.

Sun acquired StarOffice last August and has been giving it away free of charge. It's not open source, not guaranteed to always be free, and it is proprietary software replete with generic licensing nonsense—still, you can now download all 65MB of it without paying a fee. You can also order it on CD for $10 or $40 if you want documentation and support. The question is, where is Sun going with this? A typical cynic might expect Sun to dump StarOffice on the market, drive out the competition, and then start charging for the updates—typical rent-seeking behavior. However, Sun's plan is a bit more visionary. Remember web-based e-mail? Sun wants a web-based office suite.

StarPortal will be the name of this “portal computing” office suite, but since it's first-class vaporware, we don't know how it will be implemented (well, Java). Still, everyone wants to cash in on the falling cost of bandwidth. StarPortal will have many obstacles to overcome (stability, security, accessibility/availability, even the client/server implementation itself), but at least Sun has the size and capital to pull off this kind of project as far as it can go, and Sun owns Java.

Applix, not to be left behind, in fact to be in front with software instead of vaporware, is also expanding into the thin-client office suite market. Anyware Office, Applix's answer to the portable office problem, is an 800K Java applet which allows a user to have office suite access from within any Java environment (Netscape, IE, JavaOS, etc.). You need to have Applixware on your home machine, of course, but if you've got this much, you can set up a home office and contact it from any terminal whenever necessary. It's quite a clever model, one both Applix and Sun are pursuing. One difference in strategy is that Sun hopes to have StarOffice accessible even from PDAs (personal digital assistants with their half-functional web browsers) while Applix, at least for now, is staying solidly on the functional web-browser level.

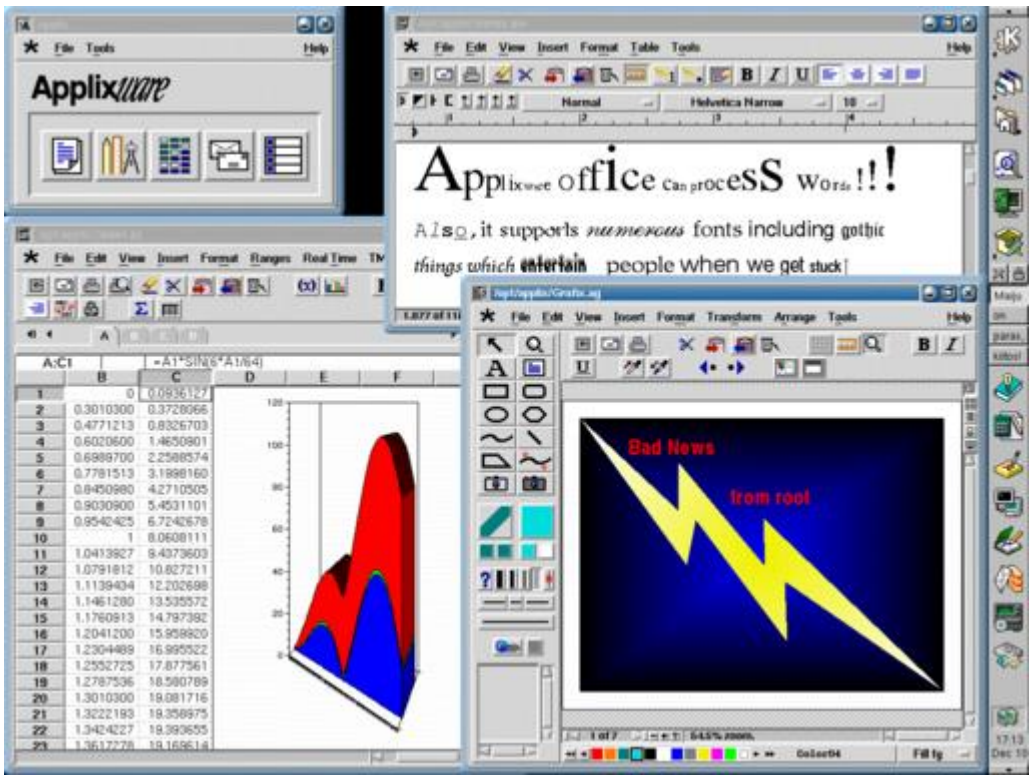


Figure 1. Applixware Screenshot

Another issue is the use of Java. Years ago, when asked for my opinion of Java, I said it was not a serious language, but only a distraction, trendy and not worth the effort. Nowadays, Java has grown in popularity, not entirely on its own merits but because of C's memory management issues and C++'s tendency to produce memory-leaking monstrosities (well, low-level languages do expect you to deal directly with memory). Java has been quite successful on the

Internet and intranets and is enthusiastically supported on account of its attempt at platform independence. However, developers must be aware that Java is a proprietary language. The specification is open: anyone can write Java programs, compilers or interpreters, but Sun owns the rights. It's probably not a good idea to become dependent on proprietary languages, and much better languages are available, but Java is nevertheless where our Linux-based office suites are headed.

Now we know where these office suites are going, so how do we choose between them? This assumes, of course, that we are willing to rely on commercial software. It's mildly difficult to get by without using commercial wares from time to time, so if you can, maybe you *are* a saint, or at least truly clever. Office suites are often the justification for getting a computer, and if you've been in school recently, you know exactly how much of your otherwise pleasant life is spent in front of a word processor. Likewise, science types may often find themselves chained to spreadsheets and need to have a fast, stable package with all the formulas and graphing functions that might come up. In any event, if you want to use a computer with only one OS on it and expect to get something done other than hacking, a word processor is a good idea.

At a Glance

Precisely what is included in an office suite varies from one vendor to another. Applixware and StarOffice provide many of the same applications, but each offers a couple of things the other does not. StarOffice has quantitatively more offerings, although one is well-advised to consider quality over quantity. And of course, the real question isn't “who's best?”, but which one makes you happiest. To complicate matters, StarOffice is free of charge at the moment while Applixware costs $100, so there is definitely a price incentive to go with StarOffice. At the same time, office suites are like office chairs; you should be sure you have the right one, because you're going to be spending a lot of time with it. Also, there is something to be said for the danger of letting larger businesses affect Microsoft-style price wars (<\#224> la Internet Explorer), but one wants to go with the best product either way, and Sun may adopt a semi-open-source policy at some point. Regardless, Table 1 is a summary of the similar offerings from Applixware and StarOffice.

Application Summary

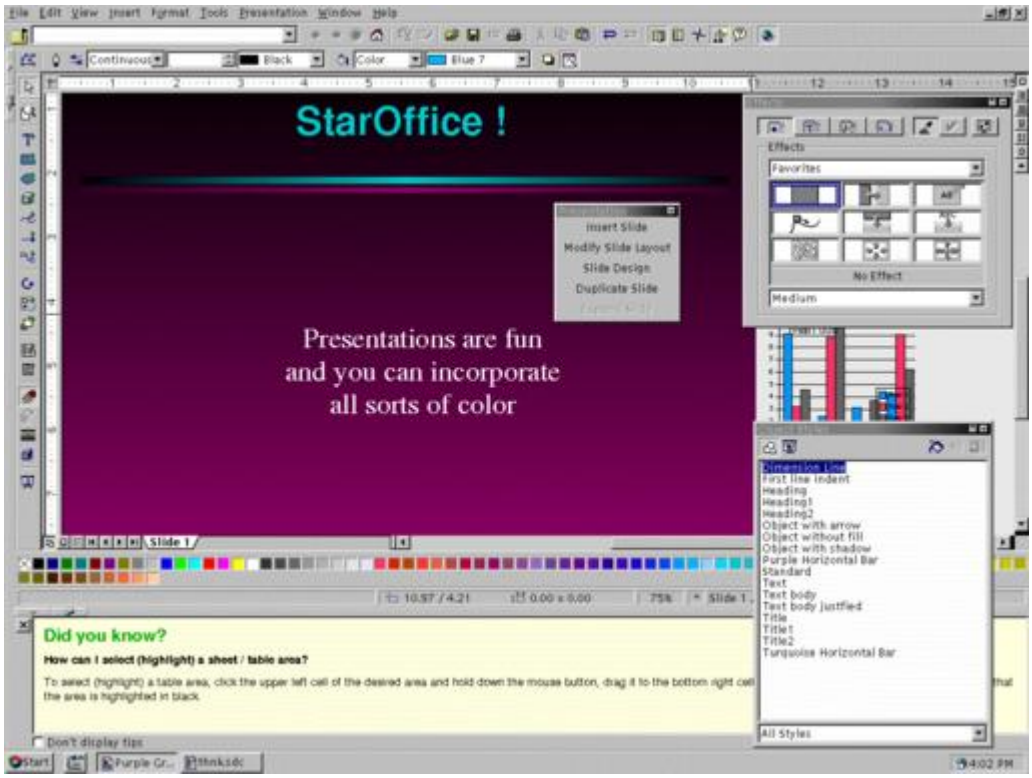


Figure 2. StarOffice Screenshot

Superficially speaking, the look-and-feel difference between StarOffice and Applixware is that StarOffice features itself as having a self-contained, uniform office environment, all in one window (see Figure 1), whereas Applixware's components are integrated but have individual windows operating within the window manager's environment (see Figure 2). The advantage of StarOffice's approach is that users have a complete office environment they can use on any platform with any window manager, without having to deal with anything unfamiliar. Likewise, the advantage of Applixware's approach is that it gives more flexibility to the user to manipulate the windows in whatever ways happen to work best. Hence, the disadvantage, if you call it that, is users will need the brainpower to cope with a window manager. Still, it must be said that window managers for Linux have become nothing short of amazing, while StarOffice's environment does not look as nice or work as well as any of our typical window managers or desktop environments. In fact, StarOffice's habit of restricting everything to one window loses the benefit of virtual desktops and can lead to several layers of clutter. However this is entirely an issue of personal taste, and while I prefer Applixware's flexibility of windows, many people will prefer StarOffice's self-contained environment, and none of this should really matter compared to the capabilities of the suites.

Word Processors—Applix Words and StarWriter

The word processor may be the most popular computer application and consequently has become a flagship product in office suites. Corel, for example, is planning to deliver an office suite for Linux based around the

strength of WordPerfect. Word processors don't actually do anything phenomenal and they're very similar these days, so as a result of an effort to improve word processors that are already just fine, they have become overrun with gratuitous functions, while it seems some basic areas are given less attention than they might deserve.

An important consideration for Linux users who want to purge MS Word from their life, but need to read and write .doc files, is how well a Linux word processor can import and export different file formats. StarOffice Writer and Applix Words both export and import MS Word files and RTF (Rich Text Format), but Applixware supports more import/export formats including WordPerfect, Frame MIF and various ASCII formats, while StarOffice has very strong support for export formats but less support for importing. “It's not us vs. them, it's us *and* them” says Applix's web site, and this consideration is taken more seriously with Applixware. Sun probably doesn't see things this way and has the capital to make this an “us vs. them” struggle. In fact, Sun has developed a program to wean people from MS Office and get them trained for StarOffice. Still, cooperation is nice, and these processors would benefit from increased support of standards like PostScript and PDF. The existing support for importing and exporting document formats is not quite perfect in either, and there seems to be some resentment on all fronts to keep up with everyone's formats. However, documents are less complicated than spreadsheets, so you can expect things to be all right here.

As for the gratuitous functions, both Applix Words and StarOffice Writer are loaded. Obviously, these days you can easily import whatever you want into your document, especially anything created with any other component of the office suite. This is exceedingly useful, but a host of other features go a bit over the top. Both suites can make all sorts of formatting contortions, format automatically, correct errors, offer thesaurus support, insert all kinds of objects and fields, sort, merge and manipulate whatever you like. And, of course, there's clip art—lots of it. Word processing has become synonymous with desktop publishing, and now it's moved into the new territory of the Web.

HTML—Applix HTML Author and StarWriter

For your desktop web-publishing needs, Applix offers HTML Author, a variation of Applix Words designed specifically to create web pages. StarOffice Writer, the word processor, is likewise equipped for producing web documents. Already you can include hyperlinks in normal text files, but with these packages, you can create and format documents for the Web. After all, it's the most logical extension of desktop publishing, since web content can reach an enormous audience while consuming practically no resources (compared to the tree-slaughtering, postal-service-dependent printing press). It seems a bit like

cheating to use these visual web programs instead of just writing the HTML yourself, but WYSIWYG can be more appealing than the trial-and-error (type this and see what happens) approach. The HTML capabilities are more of a convenient extension of desktop publishing functionality than a substitute for Netscape (or preferably just learning HTML), although you can also browse the Web with them; StarOffice Writer works much better for this. Also, there's no clever caching as in Netscape, so you're best off with Netscape/Mozilla or KFM for web browsing with graphics.

Graphics—Applix Graphics and StarDraw

If you want serious graphics, you need either the GIMP or to wait for CorelDRAW. For any small or medium-size drawing projects especially producing objects for your documents, spreadsheets or presentations, the Applixware and StarOffice drawing applications can do what you need quickly and easily. Applix Graphics looks simple on the surface, like X Paint, but has much to offer including support for many graphics formats. It is simple and easy to use, but not exactly full-featured. In contrast, the StarOffice Draw and Image programs are well-designed for graphics components in an office suite (the phrase “almost engineered” comes to mind). There is a good assortment of brushes, effects and functions, all of which are quite flexible. The brush control is particularly nice. It's a bit complicated to get around, but if you can figure it out, you can produce usable graphics without much effort.

Spreadsheet—Applix Spreadsheets and StarCalc

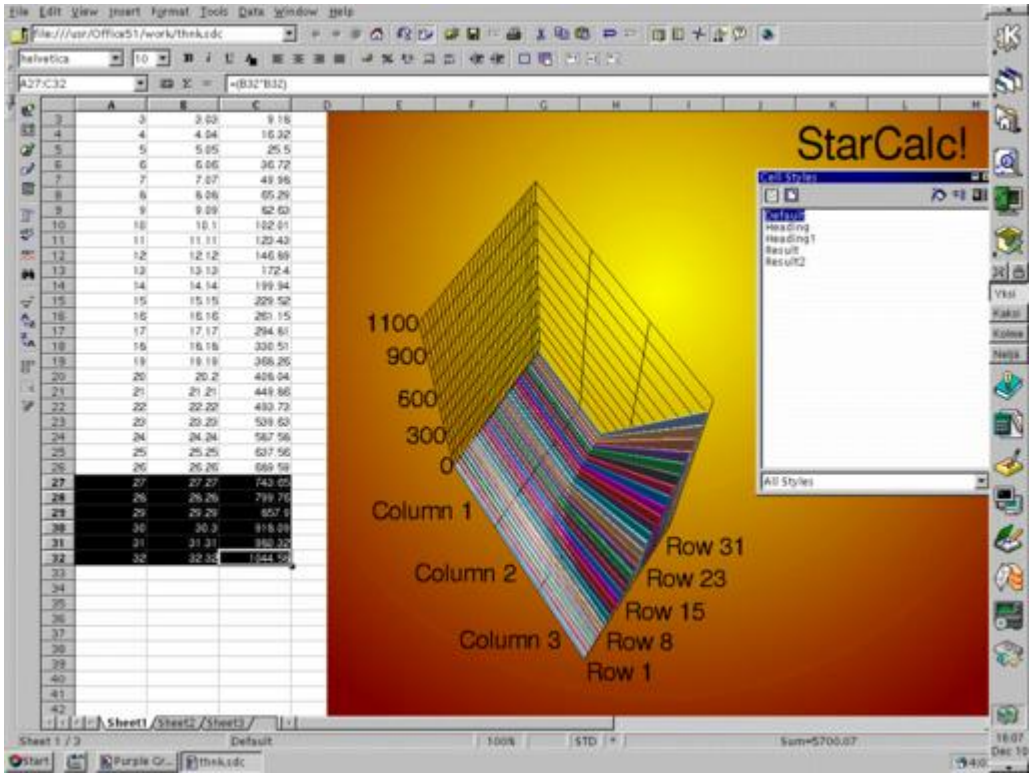


Figure 3. StarCalc Screenshot

The spreadsheet was once the killer app of the computer world, and maybe it still is. As useful as word processors are, it is the spreadsheet that best exploits the computer's number-crunching capabilities. From office workers to statisticians, scientists and students, it seems we're all on spreadsheets at one point or another. Spreadsheets can dig into databases, crunch enormous amounts of information, spit out colorful and elegant graphs, and provide answers for our mathematical and statistical curiosity; hence they're quite popular. However, spreadsheets are tricky. There are so many different functions and syntaxes to support, it becomes very difficult to give comprehensive support to numerous popular spreadsheet file formats. The Applixware and StarOffice spreadsheet programs are quite good in their own right, but still have compatibility issues. Of course, Excel doesn't exactly support Applixware and StarOffice formats, either.

Cross-platform support is a big problem with Applixware Spreadsheets and StarCalc. While supporting word processor formats is not so difficult, the complexity involved in formulas, arrays, titles, graphics, tables, relationships, etc. makes it excruciating to program a spreadsheet that can deal with so many formats. If you import complicated spreadsheets, expect to find some titles missing and some functions listed as unimportable.

Ideally, you will be running your business on Linux/UNIX machines, so you shouldn't have to put up with Windows formats on a regular basis. There is a growing awareness of Linux as a business platform, and we can all help this along by taking the leap to break away from our dependency on non-native spreadsheet formats.

As for the spreadsheets themselves, they're fast and high quality, the difference being that StarCalc (see Figure 3) has far more to offer in terms of functions while Applixware Spreadsheets is fast and easier to use. Applix says Wall Street brokers use their software, and it's pretty good in the financial department. However, Applixware is missing too many formulas for it to be useful for serious statistical work (without having to program your own formulas, which can be done), whereas StarCalc is well-endowed, which can also mean a higher success rate in importing formats. Applixware is also quite inaccurate in its current incarnation in dealing with discrete floating-point operations. Worth considering is the excellent Xess spreadsheet from Business Logic, very fast and full of formulas, which I will review next month. If spreadsheets are crucial to you, check this one out before deciding.

The graphics on both spreadsheet packages are fine. The computer will generate color graphs of your data, rotate it, label it, and let you decide all the details of how it will look. This is typical of spreadsheets, but these actually do a good job of it. StarCalc is more thorough and offers some nice 3-D functions.

Applixware is a bit funny; it uses chi as the formula icon, but then doesn't have chi stat formulas.

E-mail—Applix Mail and StarMail

There is little to be said about e-mail. It's not hard to implement, so it works. Both Applix Mail and StarMail are fine for sending and receiving e-mail, supporting MIME, Sendmail, POP3, folders, etc. These actually seem a bit dull, and I'd prefer to use mutt with all its colors, but they're integrated components and that's what' important. StarOffice even has its own threaded newsreader called Discussion.

Presentation—Applix Presents and StarImpress

Presentation software is important for people who make presentations. If you are one of these people, you can use one of these packages. As is typical in the Applixware/StarOffice comparison, Applixware is simpler while StarOffice is more thorough. I found Applix Presents to do well for informal affairs (probably just on account of the nature of its templates), while StarImpress would be better suited for formal presentations, although this is a bit categorical. StarImpress has a nice collection of templates and a well-designed drawing program, so with some resourcefulness, you can put together quite a decent presentation. Applix Presents will also do for most circumstances and it's quite easy to use, but seems a bit less flexible and the included templates are not as beautiful. Since presentation software is so bureaucratic and businesslike, it fits in with the office-culture scene, but I have my doubts as to its value. Still, it's fun to play around with.

Database—Applix Data and StarBase

In the *LJ* Readers' Choice poll, many people asked, “Who needs databases?”, and honestly, they can be a bit dull. In any case, StarOffice has a database with drag-and-drop and relational-link capabilities between several tables. Applixware in turn has more or less the same thing, to spare you the effort of learning SQL. ODBC (for example, MySQL and ADABAS) should work fine, so if you feel the need to bother with databases, you can choose either of these tools. Not being a database connoisseur, I don't have a preference.

Development—Applix ELF

A key element of Applixware Office is ELF, the Extension Language Facility, a scripting language with interactive interpreter, compiler and debugger. The idea is to allow rapid prototyping, so there are over 3,300 macros included, as well as typical user-interface design tools such as a menu editor, bitmap editor, drag-and-drop/dialog editor, a TCP/IP socket interface and an SQL database

interface. Applix says the Applixware Office suite is written largely in ELF, so clearly it's capable of something. Although it was once proprietary, ELF has been detached from Applixware and is available under LGPL as SHELF. Inside of Applixware, the giant advantage of ELF is that it rapidly allows users to extend the functionality of the suite. Businesses each have their own peculiar demands, and this kind of flexibility and extensibility is said to have popularized Applixware. ELF is a composite of Elisp and BASIC (sounds promising, eh?), but apparently ELF is more like an evolved awk or sed.

Applix Builder is the integrated development environment for designing and debugging ELF programs. The ELF library is LGPL and can be dynamically linked. The applixware.org web site is devoted to the open-source exchange of ELF software. It's an interesting attempt at utilizing the open-source phenomenon to increase the value of a product, and it may actually be working. In any event, you can incorporate C, C++ and CORBA into your ELF code, so it's fairly powerful.

Office Planning—StarSchedule

StarOffice has a unique offering, an office-scheduling program which is a lot like those big white calendar sheets people put on their desks—only digital and a bit more organized. Control freaks can chart out their days, weeks and months and keep track of projects, meetings and the usual business events. PDA enthusiasts may find this quite useful, as StarOffice Schedule will synchronize data with PDAs such as the PalmPilot. I think **plan** is good enough, but at least you've got this scheduler around if you feel inclined to use it.

Applix Filters and StarTools

By including so many components, Applixware and Sun haven't made it easy to quantify their office suites. First, the filters. There are a ton of them, and they work well enough to get a file loaded into your office suite. However, there are far more filters than importers, and filters take things away. That means you can bring a document, spreadsheet or presentation across, but unprocessable data will be removed and you'll have to fill it in by hand. Exporting is a bit easier to implement, especially for simple files. In general, the less fancy the file, the easier it is to move. The more advanced the formatting, the more likely it will be mangled, er, filtered.

StarTools are components of the individual programs in the office suite. StarOffice Image, for example, is part of StarDraw; Chart is part of the spreadsheet; Math is a formula editor; Gallery is for navigating clip art and other multimedia bits; HelpAgent is the on-line help; Navigator is your typical GUI; Stylist manages the templates. Ultimately, they're just necessary components. For advertising purposes, Sun catalogs its accessories while Applix

catalogs its filters, although both packages have components. It's generally true that StarOffice is better on accessories and Applixware is better on filters, but both are fairly well-balanced.

Which One to Choose?

The question on so many minds is how to set up a productive Linux desktop, and preferably, how to do it inexpensively. StarOffice is free of charge right now, so if you have the bandwidth, you might as well download it. It has a slightly steeper learning curve than Applixware, but once you have it down, you should be able to do anything. The point is, it's free, so if it's good enough for you, there's little sense in spending money unless you are worried about Sun gaining market dominance and not trusting what will happen afterwards. If you plan to spend most of your time writing, I'd opt for WordPerfect. Likewise, if you are going to be dependent on a spreadsheet, at least consider Xess before making a decision.

Applixware has its advantages. It's much faster than StarOffice (which has a habit of crashing and leaving tracers), and the interface is easier to negotiate. The feel is quite UNIX, and it's been a standard for a long time. Probably most, if not all, programs in Applixware are easier to use for anything basic, but when you find yourself wanting those weird, hard-to-do things, you'll probably have a higher success rate with StarOffice. Businesses could definitely take advantage of ELF, so I would recommend they consider Applixware Office, while I would suggest a download of StarOffice for individuals who just want to save money.

Conclusion

Linux office suites are not quite as easy as on other platforms. The features aren't all there, but if you're the slightest bit clever, you can do anything. A large part of the fun of using Linux is roughing it. “I'm not supposed to be able to do this, but look—I did it anyway!” is a defiant thrill, a key part of the Linux experience. The time when everything will work perfectly and there won't be any obstacles to office work is not far off. In the meantime, enjoy the mild challenges that come up when you have to do something weird that isn't quite supported; in the future, it's a recreational activity we might not have.

The market itself is a curious issue. Software's free nature has enabled the Open Source movement, but has also delivered to the commercial software industry the incantation for Armageddon, the ultimate price war. Applix is a strong competitor with unique offerings and intelligent strategies and should survive this full scale assault. Applix launched smartbeak.com, a web site devoted to the open-source exchange of ELF programs, and more recently acquired CoSource.com, a web site that matches free source developers with funding (truly a more modern development model). Nevertheless, the free-of-

charge tactic exposes another fundamental defect in the already failing Cathedral-style, proprietary model. Without intervention, the commercial software industry might burn itself up, but with intervention it could strangle itself with more and more regulations imposed to support an arbitrary and inefficient development model, especially considering the impracticality of enforcing arbitrary “intellectual property” laws which are already considered illegitimate by many. With the new weapons of free-of-charge price wars, look-and-feel patents and reinforced hard-core digital copyright laws, this is a trip-wire and land-mine industry.

One thing we do know is this: Linux is completely ready to be your one and only desktop OS. We've got two excellent office suites, with more on the way, as well as individual components such as Corel WordPerfect and the Xess spreadsheet. Abolish your DOS partition, and put Linux on your secret Macintosh. What was once a hacker's project is now a completely viable system for home users, corporate users, government and public institutions, scientists, researchers, high school and college students; and yes, it's still the strongest magnet of brainpower in the computer world.

Jason Kroll (info@linuxjournal.com) is Technical Editor of *Linux Journal*. He is fond of animals, especially dogs, while he admires the independence and aloofness of cats who often remind him of misfiring computer programs.

Archive Index Issue Table of Contents

Advanced search

LaTeX for Secretaries

Jacek Artymiak

Issue #70, February 2000

How to survive without Microsoft Word.

Life is not fair, is it? You learned how to use Microsoft Word or Corel WordPerfect, only to find yourself at a job where your boss says you must use LaTeX. What is this thing? Can it be used for anything practical, like writing letters, memos, reports, sending faxes or printing price lists? These are some of the first questions that come to mind when you start using it. Fortunately, with LaTeX, it is just as easy to typeset a 1,000-page book filled with mathematical formulae as it is to prepare a short letter or a price list. These kinds of real-world office applications are what we are going to discuss below.

Not a Word Processor

Probably the most common misconception about LaTeX is thinking of it as some kind of text editor or word processor. It is neither of those tools. Let me explain. When you start Word or WordPerfect in Microsoft Windows or on a Macintosh, it opens a window, displaying buttons, menus and a white space where what you type shows up immediately on-screen and can be edited at will. To display or print documents, word processors use an “engine” of some sort that changes simple keystrokes into nice-looking type according to your choice of font and style. LaTeX can be thought of as such an engine, reading plaintext files on one end and changing them into professional-looking documents and saving in its own format called DVI on the other end.

Although LaTeX is only a *pure* typesetting system (it is not a visual DTP package like Microsoft Publisher, Adobe PageMaker or QuarkXPress) and the documents you process must be created using a plaintext editor, the quality of the documents generated in LaTeX are often much superior to Word's own efforts. However, learning to typeset documents in the former might be harder at first than doing it in the latter because of the need to manually add control commands to the text, which some people find confusing. You might think of it

as an unnecessary burden, but you should remember that in reality, all word processors and text editors add control commands to the text you type—they just don't make them visible to us.

Typing Your First Letter

The best way to learn LaTeX is by example, so log in to your system (in case you do not know what *logging in* means, try some beginner Linux books) and type the following command:

```
emacs businessletter.latex
```

File extensions are optional and could be just about anything you like, although using .latex or .tex is good practice, as it makes documents easier to find.

After pressing the **ENTER** (or **RETURN**) key, you should see Emacs in all its glory —two toolbars and an empty space waiting to be filled with text. The \ character marks the beginning of LaTeX commands, so remember to put it in as well, and do not get confused by a **** showing up from time to time in the rightmost column of the Emacs window—it's there to show you that a particular line of text is longer than the width of the window and has been wrapped. Let's type in the lines shown in Listing 1.

Listing 1

To save what you have just typed in, press the **CTRL-X** and **S** keys. You should see a message at the bottom of the window saying **Wrote .../ businessletter.latex** to inform you that your document has just been saved. If you accidentally press the wrong keys and Emacs starts complaining and beeping at you, pressing **CTRL-G** will almost always get you out of trouble.

Basic LaTeX Commands

As you can see, the example letter is sprinkled with many strange commands and curly brackets. Besides making a document less readable, they tell LaTeX how to format it. Unfortunately, you will need to learn at least a few of them. While you read the following paragraphs, refer to Listing 1 to see how each command is used in practice.

First comes the obligatory **\documentclass[...]{...}** command, which is not as scary as it looks and is just an obscure way of telling LaTeX what kind of document you are trying to print. It could be one of the following: **article**, **book**, **letter**, **report** and **slides**. Whatever you choose, put the appropriate word between curly braces.

The square brackets surround more specific options used to set the size of paper (available sizes include **a4paper**, **a5paper**, **b5paper**, **letterpaper**, **legalpaper** and **executivepaper**).

Additional information that can go there includes page orientation (**landscape**, useful for printing presentation slides, or **portrait**), main font size (e.g., **10pt**, **12pt**, etc.) and many others.

The **\frenchspacing** command solves some of the typesetting problems related to setting the amount of space between a full stop and the next word after abbreviations like *Ms.* or at the end of a sentence. I suggest you always put it somewhere at the beginning of a file, perhaps just after the **\documentclass[...]{...}** command. Purists will surely complain about this advice, but using **\frenchspacing** will automatically make documents look better without causing you to worry how it happened. However, if you want to be "politically correct", always put a **~** between an abbreviated word ending with a full stop and the next letter, number or word, e.g., **Ms.~Green** instead of just typing **Ms. Green**.

After setting those options, you will need to specify the sender's address with the **\address{...}** command; every line should be separated by ****--this method is used in many other commands as well. If you are using official company letterhead, you may leave this out.

LaTeX automatically puts the current date into a letter, but you can use the **\date{...}** command to specify a different one.

The signature text should go into the **\signature{...}** command.

To let the program know where your letter begins, use the **\begin{document}** and **\begin{letter}** commands. Just after the latter, insert the recipient's address in curly brackets. Then, begin the greeting with the **\opening{...}** command and start writing your letter. Write as much or as little as you need, separating each paragraph with a blank line (you will need to press the **ENTER** key twice).

At the end of your letter, use the **\closing{...}** command and, if needed, add the **\cc{...}** for "copies to:", **\encl{...}** for "enclosed items:" and **\ps{...}** for PostScript.

You should end a letter file with the following commands: **\end{letter}** and **\end{document}**. That's it. You can save it as described above.

There is one trick which you can use to save yourself a bit of work when you need to type several letters. It is possible to begin another letter in the same file, just put the **\begin{letter}{...}** and the other commands mentioned above between the **\end{letter}** and the **\end{document}** commands. LaTeX will

automatically use the signature, date and sender's address you specified at the beginning of a file (that's why you had to put your signature text at the top of the file).

Changing Text into Type

To start LaTeX, type this at the prompt:

```
latex businessletter.latex
```

After you press the **ENTER** key, LaTeX will print some cryptic messages, two of which are most important: **Output written on businessletter.dvi...** and **Transcript written on businessletter.log.** Your letter is output into a DVI (device independent) file, a universal format, which you can turn into fax, PostScript or printer files.

If there are problems, LaTeX will sometimes stop in the middle of its work and ask you for help. When that happens, make a note of the line number at which the error has been found, then keep tapping the **ENTER** key until you see the "Transcript written on" message or the command prompt. Then run **xdvi**, and after visually finding the place where the error occurred, go back to Emacs and correct your mistake. Usually it will be a missing bracket or a misspelled command.

Previewing Documents

The best way to find errors and check the document's look before you print it is to use the xdvi previewer. It works only in the X Window System. To start it, you either choose it from a menu or type **xdvi businessletter.dvi &** on a command line in a terminal window and press the **ENTER** key.

Using xdvi is easy. The buttons on the right-hand side of a window are fairly self-explanatory, and you can press the **PAGEUP**/**PAGEDOWN** keys to flip pages back and forth or use the arrow-up/arrow-down keys to scroll the page you are looking at up and down (the scroll bars work just like in MS Windows, although with a two-button mouse, you will have to press both buttons to use them). Pressing the **Q** key exits xdvi.

Some Unbreakable Rules

The most common errors that can be found just by looking at a page in xdvi are the *overfull boxes* and sudden changes in font and style. These errors usually

happen when you forget to obey some of the following rules of typesetting in LaTeX:

- Each paragraph must by separated from another by a blank line (just tap the **ENTER** key twice at the end of a paragraph).
- No matter how many spaces you put between words, LaTeX will treat them as one space and will format it according to its own typesetting rules. (You might change this behaviour, as described in the section "Overriding LaTeX Rules" below.)
- Quotation marks are made using **`** and **'** or **“** and **”** instead of **"**, so to typeset **"funny"**, you'll need to type **"Funny"** instead of **"funny"**.
- Ellipsis (...) is printed with the **\ldots** command; consequently, to typeset *bye...*, you ought to type **bye\ldots** and not **bye...**.
- LaTeX commands should be separated from the actual text. We achieve that in one of the following ways:

that begins above]

- bye, bye\ldots my love (note: the **** character starts a command and therefore does not need to be separated with a space, also see the list of special characters below).
- bye, bye\ldots{ }my love.
- LaTeX interprets some characters as special, and to print them literally, we need to use a special command instead. See list in "Special LaTeX Characters".

Special LaTeX Characters

- Commands can be nested, i.e., you can put one or more commands within curly brackets of another, e.g., **extsf{life is \emph{wonderful}!}**.

Getting Rid of Overfull Boxes

LaTeX is very good at documenting its work and puts a lot of information, including error messages, into a log file. For businessletter.latex, it will be called businessletter.log. Every error is described there as best as LaTeX can, together with a line number, which helps to quickly find the right place in a file.

Although the visual method of finding errors with the help of xdvi is probably best for a beginner, there is one category of bugs that requires a different approach—the famous *Overfull box*. LaTeX produces that message when it cannot properly break a line of text or hyphenate a word, which happens quite often when trying to print a long web page address.

The quickest way of finding those messages is by using **grep**:

```
grep Overfull businessletter.log | less
```

After issuing the above command in an Xterm window, you will see either an empty window (a good thing) or a list of lines where LaTeX has put the ominous word. Go back to Emacs, find the offending lines, then use xdvi to judge how to best fix your document.

Overriding LaTeX Rules

LaTeX is very good at typesetting, but it still needs our help from time to time, especially in the case of the *Overfull box* error. Helping LaTeX usually means manually breaking a particular line of text. Don't worry, it will not hurt; just follow the rules given below:

- To correct bad hyphenation, put **\-** inside a word, at a place where you think it should be hyphenated, e.g., **lab\-yrinth** or **laby\-rinth**.
- To break a line after or before a word, without inserting a hyphen, use the **\linebreak[4]** command. (Try it on a line from the earlier example to see how it works for yourself.) The text before the command will be set to fill the whole width of the paragraph.
- To break a line without filling the width of a paragraph, use ****.
- To end a page and start a new one, use **\pagebreak[4]** or **\newpage**.
- To keep a part of text together (such as a phone number), use the **\mbox{...}** command; e.g., **\mbox{+0 (11) 123 456 789}** will be moved to a new line if it does not fit on the current one, but it will not be broken in half.

After you insert one of these commands into the text, save it, run LaTeX, see how it looks in xdvi and use grep to find out whether any problems remain. Repeat until you get rid of all *Overfull boxes*.

Some Useful Formatting Tips

So far, you have read about formatting letters, but you can typeset all sorts of documents with LaTeX. For example, using **\documentclass[...]{report}** will put LaTeX into a nice report formatting mode. Brochures can be typeset with **\documentclass[...]{book}**; memos with a **\documentclass[...]{article}** and presentation slides with **\documentclass[...]{slides}**.

To make those documents look truly professional, there are a few additional commands like **\author{...}**, **itle{...}**, and if necessary, **\date{...}**. To generate a title header or a title page, use **\maketitle** after the last one. You should place those commands before **\begin{document}**.

In all cases of documents other than a letter, we do not need the **\address{...}**, **\signature{...}**, **\begin{letter}**, **\opening{...}**, **\closing{...}**, **\cc{...}**, **\encl{...}**, **\ps{...}** or **\end{letter}** commands.

To divide long text into parts, sections and chapters, use the **\part{...}**, **\section{...}** and **\chapter{...}** commands.

Lists

Listing 2

There are two kinds of lists you can typeset with LaTeX: bulleted (see Listing 2) and numbered (see Listing 3). Numbered lists are especially handy for all sorts of agreements, contracts, instructions, etc.

Listing 3

Fonts and Styles

Here is a list of additional useful commands for formatting text:

- *italics*: enclose the text you want printed in italics with the **\emph{...}** command, e.g., **I wish to \emph{emphasize} this!**
- alignment of text—left:right:center: **\begin{center}...\end{center}**;
- **typewriter text**: **exttt{...}**;
- sans-serif font: **extsf{...}**;
- font sizes: main document font size **\normalsize{...}**, and (in order of decreasing size) **\small{...}**, **\footnotesize{...}**, **\scriptsize{...}**, **iny{...}**, or (in order of increasing size) **\large{...}**, **\Large{...}**, **\LARGE{...}**, **\huge{...}**, **\Huge{...}**.

Templates

Any LaTeX file can be turned into a template with the following command:

```
chmod 444 businessletter.latex
```

These permissions will ensure that no one (including you) can overwrite the file you created. Anyone can open it, but to save the changes, it must be given a different name. In Emacs, you can do that with **CTRL-X** and **W**. It is also a good idea to place all templates in a separate directory.

Printing

Before you send files generated by LaTeX to a printer or a fax modem, they need to be converted to the right format. Fortunately, all distributions of Linux come with a set of conversion utilities that can handle this job without too much human intervention—you just give them the name of a file and they do the rest:

- **dvi2fax**: turns DVI files into fax format, making it possible to send them via a fax modem.
- **dvilj**: turns DVI files into Hewlett-Packard LaserJet format. Even if your printer is not made by HP, it can probably still understand or *emulate* LaserJet commands. All you have to do is switch it into HP *LJ* emulation mode, which should be described in the printer's manual. Some printers can switch into that mode automatically.
- **dvilj2p**: the DVI to Hewlett-Packard LaserJet 2p format converter (similar to dvilj).
- **dvilj4**: the DVI to Hewlett-Packard LaserJet 4 format converter (similar to dvilj).
- **dvips**: turns DVI files into PostScript format, allowing them to be printed on an Adobe PostScript compatible printer.

If you want to print the example letter, type:

```
dvilj businessletter.dvi
```

and then:

```
lpr businessletter.lj
```

If nothing happens (you did remember to switch the printer on, didn't you?), type **lpq** and see if the name of your document is on a list there; if you get a “no entries” message, then your machine is probably not set up properly for printing. In that case, ask the system administrator or the service person for help. Sometimes the printer receives the document, but waits after receiving each page for you to press the *Print* or *On-line* button. This depends on the printer, but a good rule of thumb is if the *Data* light is on or blinking and the *Print* or *On-Line* lights are off or blinking, then pressing one of them will solve the problem.

Also, on some printers you will need to feed an additional page at the end—it will come out blank, and you can reuse it.

Converting Files between Microsoft Word and LaTeX

So far, converting Microsoft Word files to LaTeX is a bit difficult. There are several strategies, but none of them will make you perfectly happy. To convert a Word file to LaTeX, you can:

- Save the MS Word file as a TEXT or an ASCII file. You will lose all formatting, pictures, drawings, linked and embedded documents, but you will at least get the text in a format that can be read by Emacs and LaTeX.
- Save the MS Word file as an RTF file and use either **catdoc** or **word2x** command to turn a DOC file into a LaTeX file. (Type **man catdoc** or **man word2x** to learn more about their use and options. If **man** does not work, try **info**.) You will lose some of the formatting information, but most of it will be preserved, making your job a little easier.

If you need to convert a LaTeX document into a DOC file, try the following method:

- Use **delatex** (or **detex**) to convert a LaTeX file into a plain text (ASCII) file:

```
delatex businessletter.latex >\
businessletter.txt
```

- Compare the output with the original file, and if necessary, remove some of the commands and add missing text. These converters do not yet work as we need them to, unfortunately.
- Open the TXT file in Word; you will then be asked if you want to convert it from a text format. Click *Yes*, apply formatting as it is needed and choose the *Save as...* item from the *File* menu. In the *Save as* dialog, choose the *Word document* option and click *Save*.

So far, these are the most common ways of exchanging files between MS Word and LaTeX, or any other word processor like Corel WordPerfect or Lotus AmiPro/WordPro, even if some of them are available for Linux. There are some interesting developments on the horizon, but none of them can be recommended for use in a typical office yet.

That is just about everything you will need to start using LaTeX in an office. Of course, there is more to learn and I recommend you use the **locate 'lshort2e.dvi'** command to see if you can find the “Not So Short Introduction to LaTeX2e”. It is a well-written LaTeX manual filled with many interesting examples.

All examples from this article can be found on the Internet at the site www.wszechnica.safenet.pl/archiwum/lfors.htm.

If any of the commands or programs mentioned above aren't available on your system, ask the administrator to install them and give you the necessary permissions to use them. All of those tools should be available for any Linux distribution.

Jacek Artymiak is a consultant specializing in helping companies and individuals use Linux as a desktop or personal system for common, everyday jobs. His other occupations include being a writer, journalist, web designer, computer graphics artist and programmer. Readers are welcome to send their comments via electronic mail to artymiak@safenet.pl or visit http://www.wszechnica.safenet.pl/.

Archive Index Issue Table of Contents

Advanced search

Matlab—A Tool for Doing Numerics

Tobias Vancura

Issue #70, February 2000

An introduction to a command-line program for matrix manipulation.

The first time I heard about Matlab was when DOS 3.3 was popular and my dad received a demo version on his PC. I played around with it a bit, but soon lost interest because there was no manual and I did not have any real use for it.

The second encounter was about three years ago when I attended a lecture on numerical mathematics. Some of the exercises the students had to hand in were intended to be solved with Matlab. There was even a handout describing the basic functions of the program. As is usually the case with exercises, numerical homework tends to be very time-consuming, so again I did not spend much time on it. Soon thereafter, I installed Linux on my then new PC.

The third time I came into contact with Matlab was during my diploma thesis. All serious data evaluation in the group where I worked was done using Matlab —not on UNIX, but on a Macintosh.

I asked whether I could bring my until-then “unplugged” Linux box to the lab and network it. I got an IP number and a weekend later the first Linux computer of our group was up and running.

Campus licenses are available for various software and I was quite happy to find Mathematica, Maple and Matlab for Linux. Shortly afterwards, I was running the already-existing Mac-Matlab scripts on my computer.

Introduction to Matlab

Matlab is a command-line-driven program specializing in all types of matrix manipulation. Everything in Matlab is expressed using matrices; even a scalar can be thought of as a 1x1 matrix.

The first noncommercial version of Matlab was based on LINPACK and EISPACK routines. Since then, much has changed and Matlab is now one of the fastest packages available for numerical computation.

When Matlab is started in an xterm, a prompt appears after a brief display of the logo in a separate window. Basic editing is possible, though the spoiled Linux user might miss tab completion. (A sophisticated tab-completion feature recognizing file names would be great.)

A nice feature of Matlab is the ability to cluster data into compounds similar to structs in C, but it is not necessary to define them in advance. One simply adds fields to a variable, i.e., *data.Temp* might be the temperature at which a measurement was taken, and *data.B* is a vector containing the values of the magnetic field at which the Hall resistivities *data.rxy* were measured. Adding a string **data.date='July-4th-1999'** is not a problem, either. Going a step further, it is even possible to use object-oriented features.

For complex tasks involving more than two or three commands, one can write scripts or functions. The difference between them is that scripts are executed as if they were typed at the command prompt, whereas functions use private memory with local and global variables, and of course, can have multiple return values. The syntax loosely follows C and makes scripting relatively straightforward.

It is even possible to implement graphical front ends for your Matlab application with all different kinds of buttons, sliders and levers, as shown in Figure 1.

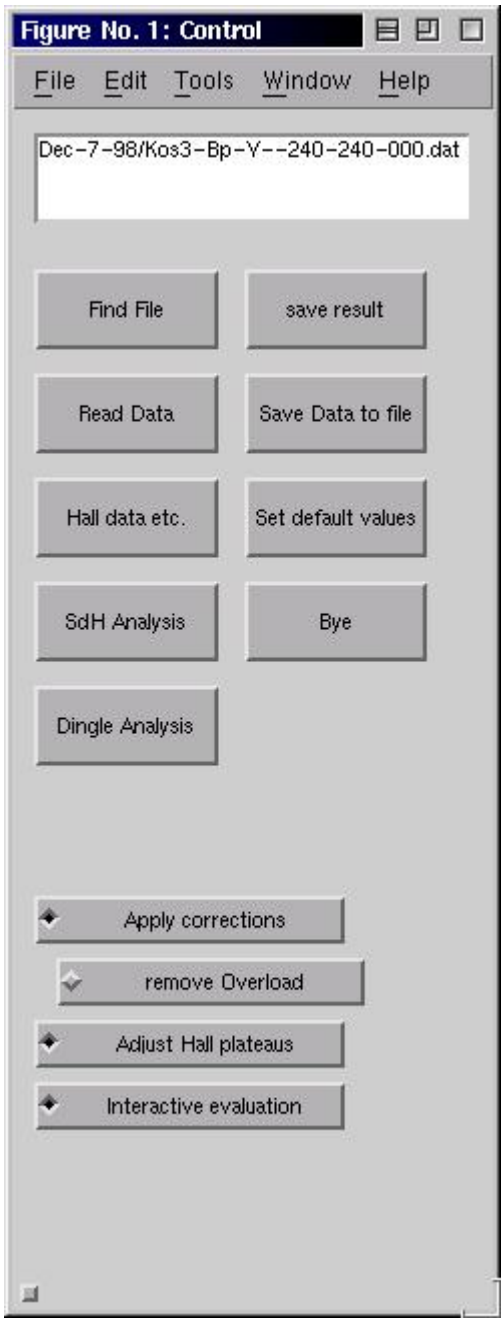


Figure 1. A front end programmed in Matlab. More complicated dialogs are possible by combining buttons, sliders and graphs.

User Interface

The interface used in the Linux version is extremely simple compared to the Windows or Macintosh version. No editor is provided for scripting, so the user has to stick to Emacs or some other editor. I personally use XEmacs in Matlab mode with syntax highlighting. The mode was written by Matt Wette and can be found at ftp.alumni.caltech.edu/pub/mwette/matlab.el. As far as I know, there is even a mode for GNU Emacs which makes it possible to run a Matlab session within the Emacs window.

Debugging scripts is not nearly as comfortable as in the Windows or Macintosh environment, where the editor windows have buttons for running the script

stepwise. Instead, debugging has to be done by issuing commands for setting breakpoints, etc., on the command line. Mathworks could make big improvements by adding, for example, a window with Step In/Over/Out buttons with a display showing the next command and offering the option of adding breakpoints.

In my opinion, it should be rather straightforward to implement this in Tcl/Tk with the script sending the specific command to the command line. An inspection window for variables would also be a neat feature, one that, to my knowledge, does not even exist on other platforms.

Graphics

Separate windows are used to display graphics. There is a vast variety of different kinds of plots ranging from simple bar charts to color-shaded 3-D plots with different light sources. Every aspect of a plot can be controlled by setting the appropriate variables. Since 5.3, it is possible to edit a graph directly with a simple point-and-click interface.

A very handy feature is the support of LaTeX-like syntax for text-in-graphics windows.

Exporting images to encapsulated PostScript is possible, although importing the file into, e.g., a (La)TeX document might not lead to the desired result, especially if there is extra text in the figure. Matlab uses fixed-size fonts, so scaling the picture can result in odd-looking tick labels. Therefore, an export mode where text and graphics are written into separate files as in XFig and Xmgr would definitely find friends in the (La)TeX community. For publications, I still export the processed data and import it into Xmgr.

3502f2.gif

Figure 2. Nice measurement of the Quantum Hall Effect and the longitudinal resistance in a two-dimensional electron gas. The subscripts and Greek letters are produced with commands similar to LaTeX.

Figure 2 shows the trace of a hall measurement in the two-dimensional electron gas of a GaAlAs heterostructure. The plot was obtained from the commands issued as shown in Figure 3.

3502f3.gif

Figure 3. The startup screen of Matlab with some simple computations and the two commands producing a plot.

Portability

Platforms in Matlab do not matter. The only problem that might arise when copying scripts from one platform to the other is CR/LF conversion.

A colleague of ours recently had a problem with a Macintosh at work. It did not have enough memory to display a large matrix, so she issued the command **save** which stores the current state in the file matlab.mat, copied the resulting file to our Linux server, loaded it under the Linux version of Matlab, and continued her work on Linux.

Performance

After the uproar in the Linux community concerning the Mindcraft report, I could not resist running a benchmark with Linux and MS Windows 98. The command **bench(***N***)** is a Matlab script that times five different tasks from different fields of numerical math and graphics. Data structures and general math are tested by solving ordinary differential equations (ODE). Floating-point values are the main issue of the Linpack part (LU), sparse matrices mix both integer and floating-point calculation (Sparse), 3-D graphs test z-buffering, and 2-D graphics test line drawing. The parameter N gives the number of times a test is performed. The higher the number, the more reliable the test.

Table 1.

The results of **bench(100)** are shown in the first two lines of Table 1. The system the test was run on is a 133MHz Pentium with 32MB of RAM. Linux is a bit slower in all cases except when it comes to 2-D graphics, where it is faster. The reason for this might be that the graphics driver for Linux is better than the generic Windows driver. I did not bother to install all possible drivers since VMWare (see “VMWare Virtual Platform” by Brian Walters, July 1999) is now available, and I won't need to reboot the machine any more in order to read an Excel spreadsheet. I installed this PC emulator on our server which has a 400MHz processor and 128MB of RAM. Just for fun, I ran the same benchmark in the emulator and directly on Linux. The results are listed in the remaining lines of the table.

Conclusion

Features in the user interface could be drastically improved—a user-friendly debugger would be great.

I was amazed by the quick response I got from the Matlab newsgroup (comp.soft-sys.matlab). Matlab engineers seem to frequent the forum quite often and provide immediate help and support.

To put it all together, Matlab for Linux is a very useful tool for doing numerical mathematics with a wealth of toolboxes, including signal processing, symbolic computation, financial mathematics and others.

Tobias Vançura (tvancura@solid.phys.ethz.ch) studied physics in Kaiserslautern and Zurich where he is now working on his Ph.D. in semiconductor physics. In his group, he is responsible for a heterogenous computer network consisting primarily of Macintoshes, one NeXT cube, some PCs running most available versions of Windows, and of course, Linux. In his spare time, he loves snowboarding and skiing in winter and tennis in summer.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Remind: The Ultimate Personal Calendar

David F. Skoll

Issue #70, February 2000

If you have trouble remembering where you are going, this clever program can help you find your way.

Remind is a calendar and reminder program for Linux and most UNIX Systems. I started writing Remind in 1989 because I was fed up with the limitations of the UNIX calendar program. In the last ten years, it has accreted features and has become one of the most sophisticated calendar programs I've seen. I'd like to take you on a journey through the history of Remind, demonstrating some of the features which lead me to immodestly call it the “Ultimate Personal Calendar”.

In the Beginning: Calendar

First, let's start with the venerable UNIX **calendar** program. This program simply scans a text file for lines containing likely-looking dates. It prints any lines containing today's or tomorrow's date (where Monday is considered “tomorrow” relative to Friday). You can also arrange to have reminders mailed to you.

Well, this is fine for very simple things. However, there is no provision for repeating reminders, so you can't (for example) remind yourself of something on the first of every month.

The next step up in sophistication is **cron**. This lets you specify dates in a relatively sophisticated format, so you can remind yourself of daily, weekly or monthly events. Still, cron won't handle fairly simple things like events which happen on the first Wednesday of every month.

The Birth of Remind

To remedy these shortcomings, I started work on Remind. In its simplest form, Remind is similar to calendar in that it reads a text file and sends reminders to standard output. However, Remind uses a sophisticated date-specification language which allows very complex reminders to be issued. Although the language is sophisticated, it is still quite intuitive for simple cases. Let's look at a few examples from a Remind script file:

```
REM 6 January MSG David's birthday.
```

That example is fairly straightforward. On January 6 each year, it reminds me of my birthday. But suppose I need advance warning of my wife's birthday, so I have time to buy her a present. Try this:

```
REM 20 December +7 MSG Norine's birthday is %b.
```

A little more complex, but still understandable: The **+7** means Remind starts warning me seven days ahead of time. The **%b** is a special substitution sequence, and it works like this:

- On 13 December, **%b** is replaced with “in 7 days time”
- On 19 December, **%b** is replaced with “tomorrow”
- On 20 December, **%b** is replaced with “today”

So, already we have some nice features: a selectable amount of advanced warning and a message body which changes depending on the “trigger date” of the reminder relative to today's date.

More Complex Reminders

How about some more tricky reminders? My local Linux users' group meets on the first Wednesday of the month. How do we express this in Remind?

```
REM Wed 1 MSG OCLUG Meeting.
```

How does it work? The **Wed** is recognized as a weekday token, and the **1** as a day-of-the-month token. If both of these tokens are present, the trigger date is the first weekday on or after the day-of-the-month. If you specify multiple weekdays, the first matching one is used. Here are a few more illustrations:

```
REM Wed MSG Issued every Wednesday.
REM Mon Tue Wed Thu Fri MSG Issued every working\
   day.
REM Mon Tue Wed Thu Fri 1 MSG Issued first\
  working day of the month.
REM Sat Sun 1 June MSG Issued first weekend day\
  in June.
```

Seemingly more difficult things like the first Tuesday after the first Monday in a month are, in fact, easy: a little thought shows that the first Tuesday after the first Monday in a month is simply the first Tuesday on or after the second of the month:

```
REM Tue 2 MSG Presto! First Tuesday-after-Monday.
```

The last Monday in a month is handled like this:

```
REM Mon 1 --7 MSG Last Monday of a month.
```

This one is rather tricky: the **Mon 1** part is the first Monday of a month, but the **--7** means “go back 7 days”. So that reminder is triggered seven days before the first Monday of every month—which happens to be the last Monday of the previous month. The **--n** sequence can solve some thorny problems related to the “last something in a month”.

Holidays

So far, any commercial calendar package can keep up with Remind, but now we start pulling ahead of the pack.

One of the most annoying things about most calendar programs is how they handle holidays. Suppose you have a meeting every Thursday, but not if it's a holiday. The typical calendar program will go ahead and remind you anyway. Here in Canada, July 1 is a holiday and July 1, 1999, is a Thursday. Look at this script snippet:

```
OMIT 1 July MSG Canada Day.
REM Thursday SKIP MSG Meeting.
```

The OMIT line tells Remind that 1 July is a holiday, and it also prints a nice message on that date. The SKIP token in the REM line tells Remind to skip the reminder if it falls on a holiday. The reminder will thus be triggered on 24 June 1999 and 8 July 1999, but not 1 July 1999.

There are other flavours:

```
REM Thursday BEFORE MSG Meeting moved to preceding\
   Wednesday if Thursday is a holiday.
REM Thursday AFTER MSG Meeting moved to next\
   Friday if Thursday is a holiday.
```

Remind has fairly sophisticated mechanisms for adjusting reminders because of holidays and weekends; please read the manual for more information.

GUI, GUI, GUI!

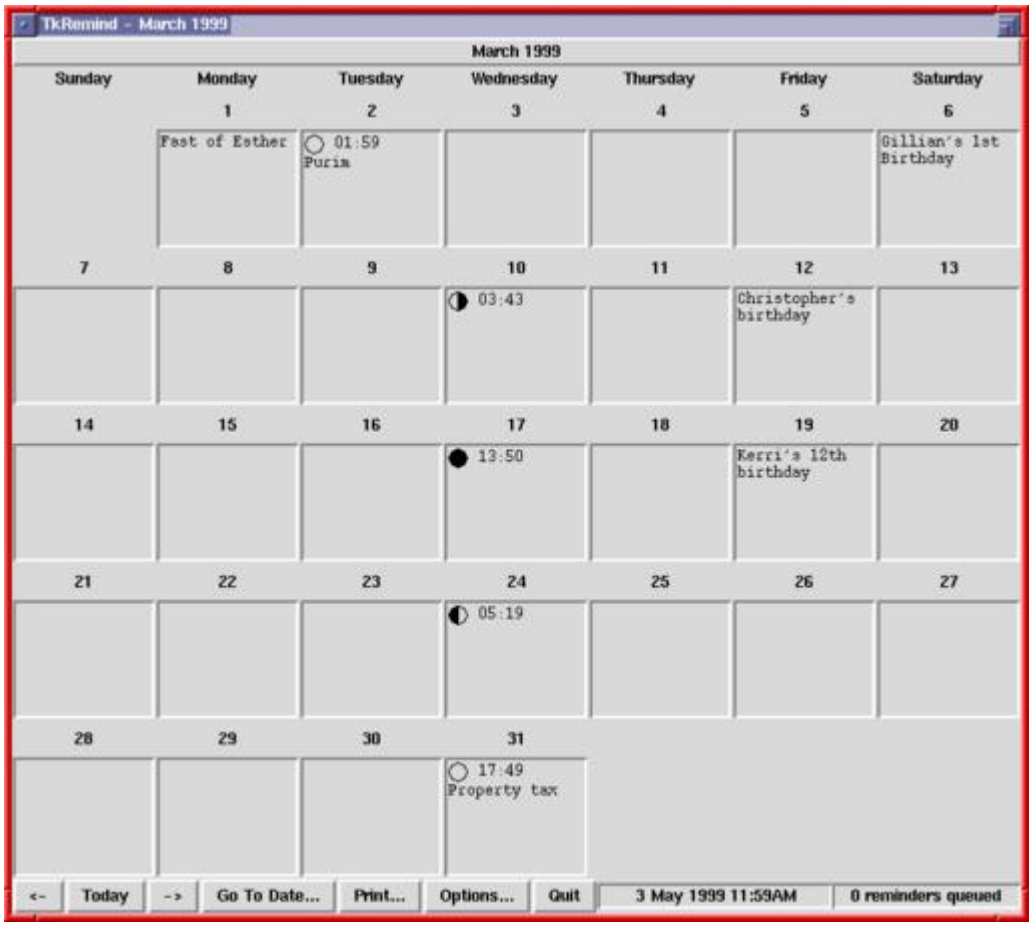


Figure 1. Main TK Remind Window

At this point, your head may be spinning. You don't want to learn yet another command language or obscure configuration-file format. You pine for the GUIs your Microsoft colleagues use. No problem; Remind comes with a graphical front end called TkRemind, written in Tcl/Tk. TkRemind presents a graphical calendar and lets you enter reminders with a simple graphical entry box. Figure 1 shows the main TkRemind window, and Figure 2 shows the reminder entry box.

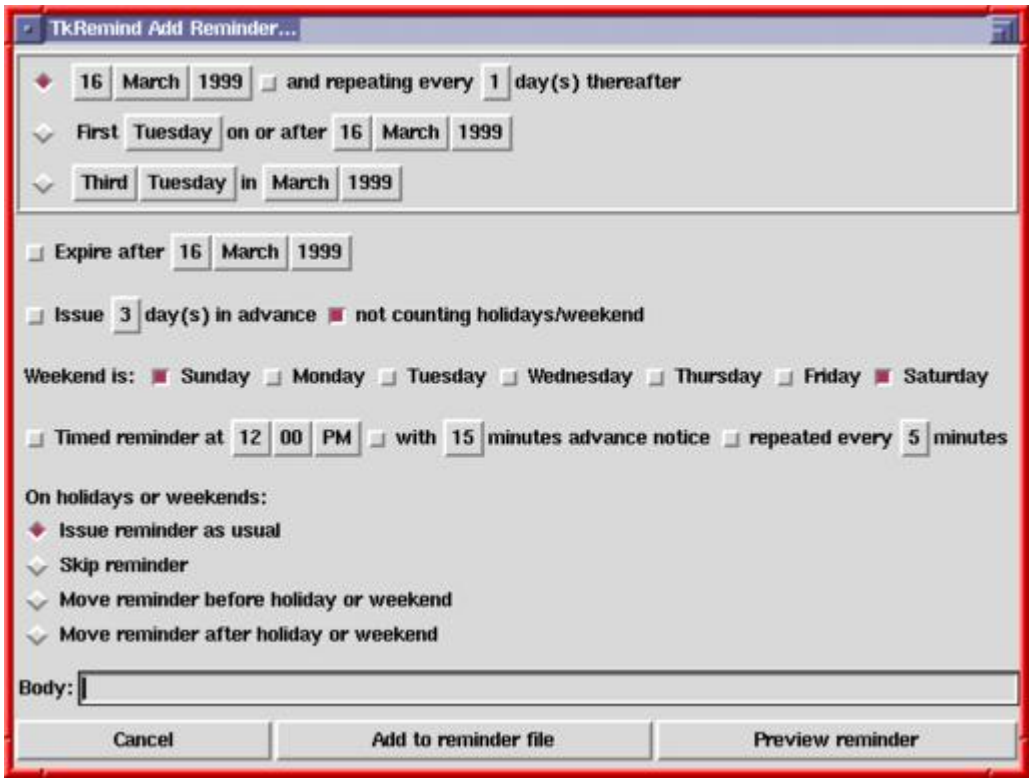


Figure 2. Reminder Entry Box

With TkRemind, you never have to learn Remind's scripting language, as long as you can express all the reminders you need with the GUI. However, you are encouraged to learn to script Remind; from the GUI, just click “Preview Reminder” to see the Remind code which will implement your reminder.

Timed Reminders

The GUI also hinted at the existence of something called a “timed reminder”. This is a reminder with a time of day specified. You can arrange to have Remind pop up reminders just before an important meeting, or more importantly, remind you to go home.

Here's an example of a timed reminder:

```
REM Mon Tue Wed Thu Fri AT 17:00 +15 *3 MSG Go home!
```

The AT keyword introduces an “AT clause”. The **17:00** means that the trigger time is 5:00 PM. The **+15** means Remind starts carping at you fifteen minutes ahead of time, and the ***3** means it annoys you every three minutes.

The TkRemind front end runs Remind in a special “daemon mode” so that timed reminders like the previous example are popped up in X windows.

Super Advanced Scripting

While what we've seen so far is quite cool, there is still the stubborn oddball reminder which requires a much tougher piece of scripting to handle. Consider the 4th of July in the U.S. If this falls on a Saturday, the previous Friday is a holiday. If it falls on a Sunday, the next Monday is a holiday. Otherwise, the 4th itself is the holiday. I won't even attempt to explain this bit of scripting; get yourself the manual, and become a hard-core Remind programmer.

Listing 1 illustrates several features of Remind scripting: Remind has built-in functions (66 of them, to be precise) and allows user-defined functions (e.g., FSET). It also has conditional tests (e.g., IF/ENDIF). A bit of clever scripting can express reminders which prove too tough for most calendar programs.

Astronomy

Ultimately, all calendars are derived from astronomical observations. Remind includes routines to calculate sunrise and sunset times for where you live, as well as moon phases. The moon phases were illustrated in the GUI calendar. These astronomical calculations are available as built-in Remind functions.

PostScript and HTML Output

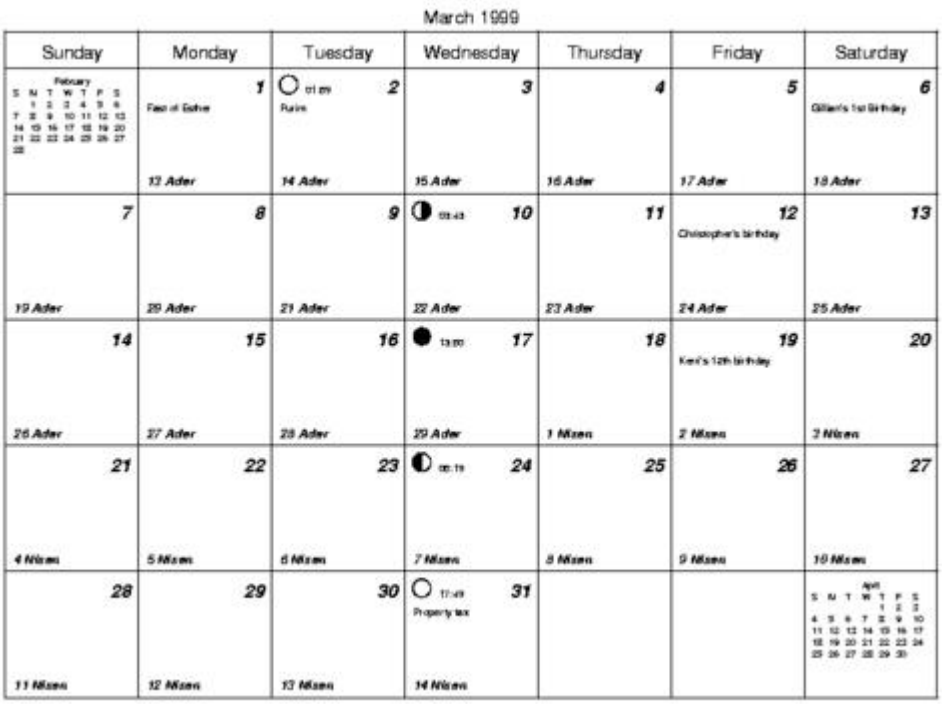


Figure 3. PostScript Output

In addition to issuing reminders on standard output or with pop-up windows, Remind can create high-quality PostScript and HTML calendars. The actual Remind “engine” knows nothing about PostScript or HTML. Rather, if invoked with a command-line option, it prints out reminders in a format convenient for back ends to process. The **Rem2PS** back end produces PostScript output (Figure 3) and **rem2html** produces HTML output. Remind itself can produce a passable text-only monthly calendar.

Remind uses classic UNIX pipes to communicate with back ends. In fact, TkRemind is a pure Tcl script which uses pipes to communicate with a background Remind process. In this way, all the hairy date-calculation code is contained in Remind, and the pretty GUI and formatting code in the appropriate back end. It should be fairly straightforward to write GNOME and KDE equivalents to TkRemind.

In addition to sending normal reminders to back ends, Remind can transmit “out-of-band” data which makes the back end do something magical. Currently, special mechanisms are defined for drawing moon phases and shading calendar blocks. The PostScript, Tk and HTML back ends all respect these mechanisms. Additional back ends can easily extend the mechanism for special purposes.

Francais, Español, Deutsch...

Not everyone speaks English. While you wouldn't know this from most software, things are changing and software authors are internationalizing their software. Remind is no exception; it has been translated into twelve different European languages. Unfortunately, this was done with a customized mechanism which does not recognize or respect the POSIX locale functions. You have to specify a language for Remind at compile time.

Different languages have different rules for forming plurals, expressing times and expressing time intervals. This makes simple message translation impossible; in some cases, the code for a language is specific to that language alone.

Remind has been designed so translators can port it to another language fairly easily; please see the source code for details. If anyone would like to make Remind recognize and respect the LC_* locale environment variables, that would be a great project.

The Kitchen Sink

Because I like to know when Jewish holidays fall, I included Remind functions for dealing with the Hebrew calendar. If anyone would like to contribute code for the Chinese and Muslim calendars, I'm open to input.

Two last bits of scripting. Suppose you want to be reminded of something every Friday the 13th. This does not work:

```
REM Fri 13 MSG Black Cat
```

because it would be issued on the first Friday on or after the 13th of every month. Try this instead:

```
REM 13 SATISFY [wkdaynum(trigdate()) == 5] MSG\
Black Cat.
```

The **SATISFY** keyword causes Remind to iterate through all possible "13ths of the month" until it hits one where the weekday is Friday. This powerful mechanism makes very complex reminders quite simple.

Finally, here's some Remind code to figure out when a blue moon occurs. A blue moon is the second full moon in a calendar month. (Blue moons are quite rare.)

```
FSET isFirstFull(date) \
           monnum(moondate(2, date)) == \
                monnum(moondate(2, moondate(2, date)+1))
REM 1 SATISFY isFirstFull(trigdate())
        set blue moondate(2, moondate(2,\
        trigdate())+1)\
MSG Next blue moon is [trigger(blue)]
```

Running this script through Remind shows that the next blue moon will take place on October 31, 2001. The one after that is July 31, 2004.

I defy you to get Microsoft Schedule to warn you of an upcoming blue moon.

And Still More…

I've barely scratched the surface of Remind. For more exciting esoteric things like system variables, priority, SCHED and WARN, TAG and DURATION, the substitution filter, safe movable OMITs, security features, the OMIT context, expressions, functions, debugging features, and so on, please download Remind. It's free—covered by the GPL—and can be found at ftp://ftp.doe.carleton.ca/pub/remind-3.0/. Although Remind is quite full of features, it's a rather slim 20K lines of code and should compile and install easily on Linux and any other UNIX-like system.

May you never forget another birthday.

David F. Skoll (dfs@doe.carleton.ca) is the founder of Roaring Penguin Software Inc., a Linux consultancy firm (www.roaringpenguin.com). He spends so much time tinkering with Remind that he often forgets his appointments.

Archive Index Issue Table of Contents

Advanced search

LinuxPPC 1999

Stephane Morvan

Issue #70, February 2000

How to install Linux on your Power Macintosh to gain a robust alternative to the MacOS.

There are some very significant things about LinuxPPC and the Macintosh that make installing LinuxPPC a challenge for Mac users. For one, no matter how hard you look, MacOS (the Macintosh Operating System) does not feature a command line or anything remotely resembling one. Fortunately enough, with this release of LinuxPPC 5.1, LinuxPPC, Inc. has done a great job of making the intrinsically complicated installation process easy with their X-based installer. Although the process is not perfect or free from any defects, it does a good job of bridging the gap between a 100% graphical OS and a 100% fundamentally command-line OS.

About LinuxPPC

LinuxPPC, along with MkLinux and NetBSD, are the three flavors of Linux that can be found for Macintosh computers. LinuxPPC supports some very exotic hardware, with various degrees of functionality: BeBoxes, Motorola Powerstacks (a PowerPC-based machine capable of running a defunct WindowsNT), IBM AIX machines (PowerPC-based, such as the Thinkpad 8x0 series) and Apple's discontinued Network Servers (which ran only AIX). Most users run LinuxPPC on Macintosh hardware, whether from Apple or from the late clonemakers (Umax and PowerComputing). A wide array of platforms are supported from old Performas (6360, 6400 series) to the latest B&W G3, Powerbooks G3 and all iMac flavors.

The name LinuxPPC 1999 refers to the most recent version of LinuxPPC, in this case version 5.1, which is a minor update to v5.0. Thus, the name LinuxPPC 1999 and LinuxPPC R5.1 may be used interchangeably.

Essentially, LinuxPPC R5.1 is based on Red Hat 6. This may sound like old news for x86 users, but LinuxPPC was almost an entire year in gestation. On June 10, 1999, the CDs shipped out to patient users. LinuxPPC 1999 follows from LinuxPPC 1998, another Red Hat-based distribution (Red Hat 5). This last release brings some very significant improvements over the previous version, including XFree86, glibc 2.1-based, USB support with 2.2 kernels, a wider array of window managers (WindowMaker, Enlightenment), **linuxconf** support and Netscape Communicator 4.6 to name a few. Furthermore, switching from the MacOS to LinuxPPC has never been easier, due to the introduction of Benjamin Herrenschmidt's BootX 1.1, a utility that makes flirting with the buggy OpenFirmware settings a thing of the past.

I have successfully installed LinuxPPC on a variety of platforms I have owned over the past years: a Performa 6360 (PPC 603ev, 160 MHz), a PowerMacintosh 6500/275 (PPC 603ev 275 MHz, ATI RageII), a Powerbook G3/3500 (PPC 750 250 MHz, Chips and Technologies 65554) and on my latest acquisition, a Powerbook G3/266 (PPC 750 266 MHz, ATI Rage Pro LT).

Installing LinuxPPC 1999

Installation has never been so straightforward: there's the choice of an X-based installer, running on a minimum live file system (with shells and WindowMaker), and for more confident users, the Red Hat installer.

The most difficult step to be carried out during the installation process is partitioning the hard drive. Although this is by no means a complicated operation, it has to be carried out cautiously to prevent any loss of data on MacOS partitions. It should be noted that, although LinuxPPC is becoming a mature distribution, users are expected (read *required*) to polish the edges by manually tuning their installation and resolving some minor problems. First-time users are often frustrated at this point: between the use of **vi** and the large number of configuration files to be edited, their MacOS experience falls short of helping them conquer the final hurdles.

LinuxPPC, however, and for that matter most Linux distributions, are built on the premise of experience capitalization: from the resolution of a problem, so much is learned of the system itself that the next problem is tackled efficiently.

Installing with the X-based Installer

The installation process starts by launching the LinuxPPC 1999 Installer, having the live file system on a local volume and rebooting the machine into LinuxPPC (using BootX). The installation can access the distribution whether it is located on a CD-ROM, HTTP, FTP or NFS. An excellent 68-page installation guide is

provided to cover the common pitfalls, and prospective users should consider reading it several times before attempting the installation.

By far, the simplest and safest installation process is with the $40 CD-ROM sold by LinuxPPC (or any custom-made CD-ROM with long file name support). Although the FTP installation is equally easy, it requires a relatively fast and reliable Internet connection and support present in the kernel for the Ethernet hardware. The local installation, which involves downloading the RPM files from one of the FTP sites to a MacOS volume, can be equally challenging: MacOS does not support long file names (in excess of 32 characters) and the name truncation is often FTP client-dependent. Furthermore, HFS+, the updated version of the MacOS file system HFS, is not supported in LinuxPPC. Hence, a local distribution should be on an HFS volume only. A custom CD-ROM can be recorded on any PC in the ISO9660/HFS Format with Rock Ridge Extensions (with long file name support). The DOS version of **mkhybrid** can be used to generate the ISO image file supported by most CD mastering software.

Nevertheless, once the source installation issue is addressed, the installation process involves six steps:

1. Partition the hard drive (1GB as root and 200MB as swap are recommended).
2. Choose and configure the installation method.
3. Choose the packages to be installed (minimum installation is 400MB).
4. Enter the root password.
5. Configure the network.
6. Reboot the machine.

Although there aren't any major issues to be resolved during the installation, I noticed the output shell is overly verbose. It reports errors which appear to be are serious, but which do not prevent the installation process from concluding successfully.

Once the machine is restarted, the user may be faced with the first problem as the screen cycles a number of times displaying a hieroglyphic message stating that a process is respawning too fast, and as a result, another process was murdered mysteriously. Fortunately, the cycling pauses for five minutes after a number of attempts, which is enough time to log in as root and solve the problem. The resolution is described at the LinuxPPC web site, under the updates section. Once this problem is resolved, the system boots in runlevel 5, and it presents us with GNOME and Enlightenment and an Xlogin screen. From there, customization of the environment and installation of new packages are simple matters common to most other Linux systems.

Installing with the Red Hat Installer

The Red Hat-based installer is geared somewhat more toward the command-line user, though other users should not anticipate any difficulties with it. Perhaps the most stringent problem arising with this installer is the difficulty of going back and making changes to adjust for a problem encountered during the installation.

Nevertheless, booting in the Red Hat installer is very simple. Either add an option to the kernel arguments in BootX (**; redhat**) or remove the live file system from the MacOS (renaming it is sufficient). The installation process then prompts for various configuration information: keyboard type, type of installation (upgrade or new install), source installation (FTP, NFS, local or CD-ROM), and the partitioning step. Rebooting the computer after the partitioning step is required rather than suggested as per **pdisk** instruction, otherwise the file system may be corrupted. Then the file system is configured, the mount points are set, and the package selection is proposed. Once installation of all packages is finished, the installer will try to configure the X Server (e.g., by editing /etc/X11/XF86Config). Unfortunately, this operation will likely fail in a frustrating manner; the workaround to this problem is proposed later.

Then, the root password is set and the system is rebooted. Since the X Server configuration failed, the system will boot in runlevel 4 until the X Server is properly configured.

About the X Server: XF68_FBdev

The current implementation of XFree86 is somewhat peculiar to LinuxPPC: the incantation of the X Server is XF68_FBdev, written by Geert Uytterhoeven. This X Server uses a hardware abstraction for the frame buffer by creating a special device file in /dev/ (usually **/dev/fb0**) and using it to drive the graphics. As with any devices/special files, kernel support is required. The default driver for the frame buffer device is called the open firmware frame buffer (**offb**). This driver is common to all Macintosh computers, which makes it inherently slow, hence the poor graphics performance of a basic installation (or the total failure thereof).

Fortunately, patches are available to build a kernel to support a particular frame buffer. Such is the case for Apple Powerbooks (1998: G3/233, G3/250, G3/266, G3/292 and G3/300) sporting an ATI Rage controller (either RageII LT or RagePro LT) with 4MB SGRAM. Booting with such a patched kernel will provide users with accelerated 2-D graphics, and an overall improved experience with the X Server (often mistakenly associated with the overall system speed, by first-time users).

To configure the graphics for such a kernel, do the following:

- Pass the correct kernel arguments in BootX to indicate the use of another frame-buffer device, screen size, screen depth and refresh rate. For instance, to boot in 1024x768x32@60Hz on a Powerbook G3/266, use the following:

```
video=atyfb:vmode:14,cmode:32,pll:135,mclk:60
```

- Once the booting process is finished, run **Xconfigurator**. Note that the correct amount of VRam is now detected. Select a monitor which suits your needs and the resolutions to be used.
- Let Xconfigurator try to start the X Server; in the event it fails, be sure to select at least one configuration at 8bpp. In the event it keeps failing to start the X Server, select another monitor.
- Start the X Server using **startx**.

Xconfigurator edits the /etc/X11/XF86Config file for specific hardware. Admittedly, this is more work than most users are willing to do, and there is no guarantee that the list of monitors provided will match a particular monitor. Fortunately, some alternate X Servers are available. Such X Servers are **Xpmac** and **Xpmac_mga** (accelerated), which are essentially software-only X Servers (although Xpmac_mga offers a degree of acceleration on some hardware). Installing them requires simply copying them to /usr/X11R6/bin in an existing XFree86 installation and linking X to either Xpmac or Xpmac_mga.

Conclusion

All in all, while this version of LinuxPPC certainly has its flaws, once the problems are resolved, the system is rock-solid and very responsive. Furthermore, the LinuxPPC community is fairly responsive. The support newsgroup is a great source for help on most common problems, as are the mailing lists. Hardware support is currently at its best, as most devices do function properly (even PPP works). For Mac users on the search for a robust alternative to the MacOS, whether for development or server applications, LinuxPPC certainly delivers the goods.

Resources

Stephane Morvan can be reached via e-mail at smorvan@ces.clemson.edu.

Archive Index Issue Table of Contents

Advanced search

Open Source/Open Science 1999

Stephen Adler

Issue #70, February 2000

Mr. Adler tells us about a first-of-its-kind conference.

Brookhaven National Laboratory (BNL), Long Island, NY, hosted the first conference to deal directly with open source and science. The conference was appropriately titled “Open Source/Open Science”.

There are many similarities between the Open Source (or Free Software) movement and the scientific method, having in common the free exchange of ideas. Publishing one's theories and experimental results to confirm or rebuke a given idea on the mechanisms of Mother Nature is one of the methods used to advance science. Posting your software on an FTP site for others to download, criticize (or compliment) and return enhancements and bug fixes to be included in the source-code base follows the lines of the open aspect of scientific research.

Because of this close parallel and the fact that scientists strive to use the latest technology available and often invent new technologies to advance their research, a conference on the subject of open-source software and its use in science was considered appropriate. The fact is, scientists have been relying on GNU/open-source/free software for a long time now. Growth of its use and reliance on this software is escalating each year.

The conference was held on October 2, 1999 on the Laboratory campus. The first goal was to highlight the use of open-source software in science; the second was to encourage the private domain to contribute their software technology to the open-source code base. Finally, it was deemed appropriate to couple this event with a public relations outreach to the local and national community by informing them of the exciting work going on at BNL. Since open-source software can be used by anyone from grade-school kids running Linux on their PCs to weather-forecasting supercomputers, this proved to be a

great opportunity for the public to relate to the science done at BNL and other national laboratories and universities nationwide.

During the day, a single track of invited talks was given. Complementing these talks were tours of the research facilities where open-source software plays a major role. They include two of the four RHIC (Relativistic Heavy Ion Collider) detectors, the Information Technology Division, the Neuroimaging Research Facility and the National Synchrotron Light Source. A call for abstracts was issued for open-source software projects used in research; 14 abstracts were submitted. A room full of X terminals served off a Linux PC was used to demonstrate each. abstracts.

The invited talks were broken into four groups. The first one was called “Introduction to Open Source and Open Science”. Bruce Perens talked about open source, and Dan Gezelter, a chemist who directs the Open Science project at Notre Dame and is in charge of the http://www.openscience.org/ web site, gave an introduction to his open-science project.

The second set focused on large-scale computing. Yuefan Deng started this section by describing the modest Galaxy Beowulf cluster built at the State University of New York, Stony Brook. This was followed by Tom Throwe of BNL with a talk on the RHIC Computing Center (RCF). The goal of the RCF is to amass 1000 Intel nodes to process in “real time” the data generated by RHIC's four detectors at a continuous rate of 60MB/second. Kent Koeninger, from SGI, gave a talk on the open sourcing of the XFS journaling file system—a key component needed for large-scale computing facilities like the RCF, which will be processing petabytes (a million gigabytes) of data on a yearly basis. Malcolm Capel of BNL's biology department talked about the use of Mosix to manage a 16-node Linux farm. These PCs are used to decode genetic structures from data collected through X-ray diffraction techniques at the National Synchrotron Light Source.

The third set was dedicated to analysis and visualization software. Bill Horn from IBM talked about OpenDX and how it became open-source software. Jon Leech of SGI presented the open-source efforts on OpenGL and GLX. Mark Galassi of Los Alamos National Laboratory presented his work on the GNU Scientific Library. Finally, Bill Rooney of BNL closed this section by presenting the open-source efforts in medical-imaging research.

The final section focused on the political arena. Jon “maddog” Hall titled his talk “It ain't Open'Til It's Open”. Immediately following, he moderated a panel discussion on “overcoming the obstacles faced by the Department of Energy (DOE), the National Science Foundation and national facilities like BNL in using and contributing to open-source technologies”. Panelists included Larry Augustin of VA Linux, Oggy Shentov of Pennie and Edmonds, a NYC law firm

specializing in intellectual-property law, Michael Johnson of Red Hat, Bruce Perens of technocrat.net and Fred Johnson from the DOE. The panel discussion proved to be a lively one. Fred Johnson stated he was taking close notes of all that transpired and would be reporting back to the DOE.

The conference was deemed a success by all. Over 200 people were in attendance. Vendors displayed their computer equipment, software and services. Red Hat and VA Linux provided major sponsorship funds. Several members of the Laboratory directorate attended the meeting. One of them told me a few days later that he thought the event would be looked back on as a turning point in the Open Source movement in science.

When not building detectors in search of the quark gluon plasma at BNL with the PHENIX experiment, **Stephen Adler** (adler@ssadler.phy.bnl.gov) spends his time either 4-wheeling around the lab grounds or writing articles about the people behind the Open Source movement.

Archive Index Issue Table of Contents

Advanced search

Profile: TiVo

Craig Knudsen

Issue #70, February 2000

The ultimate in recording television programs, TiVo is a set-top box that does everything for you.

TiVo, Inc., headquartered in Sunnyvale, California, is the creator of the Personal TV Receiver. Michael Ramsay and James Barton founded TiVo in August of 1997. Ramsay and Barton were senior executives at Silicon Graphics, involved with early generation interactive video-on-demand systems.

Personal TV Receiver

The Philips Personal TV (PTV) Receiver is a joint effort between TiVo and Philips Electronics. Philips Electronics manufactures the PowerPC-based hardware, and TiVo designs and develops the software. The appliance uses a modified version of Linux for PowerPC and provides an amazing set of features. It truly is the next generation of VCRs, allowing you to watch what you want, when you want.

TiVo's PTV Receiver does have competition in this emerging market, such as Replay TV. However, the PTV Receiver is currently the only Linux-based system.

You can still do all the things you do with a VCR with the PTV Receiver—except play a VHS tape, of course. However, with DVDs becoming increasing popular, this may not be a significant limitation for you.

TiVo records digitally using MPEG II compression at one of four selectable video qualities. Picture quality at the best setting is as good as the original broadcast, an improvement over VCR technology. Although you can't exchange your recordings with others without copying them from TiVo to a VCR, you won't have to manage a handful of two-hour tapes, either.

Figure 1. TiVo Central

TiVo takes the concept of on-screen programming a step further. Rather than just recording what you schedule, TiVo allows you to rate, with either a thumbs up or down, programs you watch. TiVo will then record shows you've scheduled and shows it finds that match your preferences (see Figures 1 and 2.)

If you happen to watch *Friends* on Thursday night and give it a thumbs up, TiVo will also record syndicated reruns without being scheduled. What if the TV network switches your program to another night? TiVo will record it automatically at its new time slot, with no programming required. This is made possible by TiVo's built-in program listings, updated nightly by an automated toll-free phone call.

TiVo also enhances your TV watching. TiVo maintains 30 minutes of what's currently being watched, so you can pause, fast-forward or rewind live TV.

For example, if you're watching your favorite NFL team on Sunday, push the instant replay button on TiVo's remote and the action will skip back eight seconds and begin playing. Push the jump button, and you're back to live TV.

If you miss the start of a program, you can watch from the beginning while TiVo records the rest of the show.

Not sure what to watch? TiVo will provide you with recommendations based on your preferences. The interactive on-screen listing means you'll never have to pick up another printed TV guide.

A limitation of the current system is that it is unable to establish individual profiles for each viewer. Instead, TiVo creates a preferences profile for the household as a whole.

Having won a “Best of What's New” award for 1999 from *Popular Science*, the Personal TV Receiver is getting positive reviews from both audio/video and computer professionals. Ziff-Davis' *Equip* product reviewer summarized it like this: “TiVo rocks! I've been testing it for a few weeks and can't imagine living without it.”

Figure 2. TiVo Screen

Linux Internals

Source code access, zero cost, high functionality, real-time capability and developer tools were among the factors that prompted TiVo to select Linux. The availability of a PowerPC port of Linux was crucial, since TiVo runs on a PowerPC-based system.

The current system is based on the 2.1.24 kernel. The GNU toolset and Tcl were used to develop all software.

TiVo had to modify the Linux kernel and some open-source tools in order to develop their product. In particular, they added an unbuffered scatter/gather API for disk access, real-time disk scheduling capability, an advanced DMA management subsystem, and support for demand paging and real-time processing. The unbuffered disk access allows the system to handle power outages gracefully. Users won't ever have to worry about waiting for **fsck** to repair the file system.

The TiVo receiver can update its software automatically. This includes both the TiVo application as well as the Linux kernel that drives the system.

During one of its nightly calls to get TV programming information, it will automatically download and install software updates. No user interaction is required.

A recent upgrade improved the picture quality during fast-forward and improved the algorithm for recommending programs. While certainly not new to PC software (America Online and others have been doing this for years), self-updating software is fairly new to consumer devices.

Product Specifications

- PowerPC-based Linux
- Stores 14-30 hours
- Inputs: cable-ready tuner, S-video and composite video support all U.S. standards
- Outputs: RF, S-video and two composite video outputs
- Channel Control: channel and power control of cable and satellite boxes with serial or IR
- Dimensions: 17-1/8-inch width by 12-5/8-inch depth by 4-1/8-inch height
- Compatible with all cable, satellite (DBS) and terrestrial broadcast TV systems in the U.S.

Where to Get One

TiVo is available for $499 in the model that stores up to 14 hours on 13.5GB and $999 for the model that stores 30 hours on 27.2GB. You'll also need to sign up for TiVo's monthly service, which is $9.95 per month or $199 for a lifetime subscription. TiVo is available in a variety of retail stores including Best Buy, Circuit City and Sears. If you prefer to shop on-line, TiVo is available at 800.com, Amazon.com, CircuitCity.com and ROXY.com.

Resourses

Craig Knudsen (cknudsen@radix.net) lives in Fairfax, VA and telecommutes full-time as a web engineer for ePresence, Inc. of Red Bank, NJ. When he's not working, he and his wife Kim relax with their two Yorkies, Buster and Baloo.

CommuniGate Pro Mail Server

Scott Wegener

Issue #70, February 2000

Stalker Software has received numerous awards for their CGP product.

- Manufacturer: Stalker Software
- E-mail: James_Bond@stalker.com
- URL: http://www.stalker.com/
- Price: $499 for 50 users
- Reviewer: Scott Wegener

With the Internet continuing to grow by leaps and bounds and more people becoming exposed to it, a proliferation of new network and Internet-specific software is coming out, giving potential buyers a confusing array of software from which to choose. Free web hosting and “free e-mail for life” sites now number in the hundreds at least, and seem to be multiplying daily. Most companies now require Internet access, including at least a corporate web site and employee e-mail. Many sites are either running NT and the Microsoft Solution for mail and web services, or using UNIX systems such as Sendmail along with a POP3 or IMAP server. After all, Sendmail has been with us for ages, and it's something most UNIX administrators are familiar with. MS Exchange and Sendmail pretty much dominate most mail-server markets today, so why change? Stalker Software (http://www.stalker.com/), a company with a background in Macintoshes and mail software, intends to change all that with their CommuniGate Pro (CGP) mail server.

While Stalker appears to have their work cut out for them, they are also off to a good start. The company has received numerous awards for the CGP product, ranging from *UNIX Review* to the *Linux World* Editor's Choice. Not being

content to blindly listen to others' reviews, I sat down to install CGP on my mail server running Red Hat 6.0. If you don't use UNIX for your mail server, they've got you covered there as well. CGP runs on a broad range of platforms, including most UNIX variants, Mac OS X Server and Windows, and is currently in version 3.1 as of this writing, with a 3.2 beta available for testing.

Installation, Configuration and Administration

Installation is fairly straightforward for all versions I've tested, those being Red Hat 5.2 and 6.0 (RPM and tgz) and Windows NT 4 Server. For review purposes, I'll be concentrating on the Linux version, although everything aside from the installation itself should apply across all platforms. The install script adds its init scripts for startup and shutdown, replaces the mail binary with a CGP-compatible version, and you're off and running. We're not done yet, though. Stalker realizes many users will be migrating from another system, such as an IMAP or POP server. CGP/Stalker provides some simple tools to import users and mailboxes, which can mean the difference between a relatively quick and easy installation, or an all-day (or longer) process for companies with hundreds or thousands of users. You can, of course, also add users manually. Users can even be imported/created via a simple text file, which can easily be made from your password file.

Once the server is up and running, administration is normally handled via a web interface. For the more security-conscious, a nice feature is that with versions 3.1 and later, this can be done via an SSL web connection. For the truly stubborn, it can also be configured via text files. While I am not a fan of web interfaces, this one is quite good. On-line HTML help is part of the system and just a click away. Earlier versions had pointed to the Stalker web site, but now all documentation is available locally. Mail domains and routing, access rules, log levels and all configuration is easily done. It does take a while to become used to getting around the interface the way it's laid out, as well as learn some of their terms, so the help does come in handy for the more unusual features.

Features, Features, Features

After the initial configuration is done, you have a large number of choices to make. For the most part, the defaults may be acceptable and are enough to get the system up and running, but the fact that this is certainly no trimmed-down mail server becomes readily apparent once you spend some time in the Admin screen. You have quite a few options, such as allowing web-based mail access, using SSL, allowing IMAP, POP, ACAP and PWD clients, maximum number of connections allowed, log rotation and levels, mailing-list configuration, blacklisted domains and more. Most of these will already be familiar to mail administrators, and others may be new to some. Administrators may also limit

system resources, message size, allowable services, and web space on a per user/list or per domain basis, as well as setting new defaults.

A *very* nice feature is server-wide rules, which allow us to bid adieu to fighting with procmail recipes. Or perhaps not—the rules also allow the execution of external scripts, if you so choose. They allow anything from simple redirections to auto-replies using the sender's information inside of the reply message.

An LDAP module is included as part of CGP. CGP also provides its own web server module, allowing users to create their own web sites and update them via the Web Mail interface. The list goes on and on—after using CGP for several months, I'm sure there are yet more pleasant surprises to be discovered.

Customization

One of the nicer features about CGP is the ability to customize. You decide which services you want to provide or allow, based on per domain or per user. You could allow only secure connections, or restrict connections via IP or domain. And the web interface, including the web actions dictated by each link, may be customized to suit your taste. The default Web Mail configuration is acceptable for most intranets, but ISPs and free e-mail sites would certainly want to add their own layouts. This seems to be easily accomplished, based on feedback I've seen on the CGP mailing list, including the use of custom CGI scripts, or choosing from ones available on the Stalker web site. Future upgrades will leave your changes intact, but use common sense; always make a backup.

The Web Interface

It's inevitable—everyone has seen it. Most of us have one somewhere or another. Some people love it, some hate it—web mail is here to stay, and can make life much easier for those on the road or otherwise remotely accessing e-mail. The default web mail configuration is usable, but has some quirks and annoyances. To be fair here, so do Yahoo, Hotmail and the like. The web interface can be customized almost to your heart's content, and it does provide some nice features. I have not done any customization, but several commercial customers on the CGP list seem to have had success with it. Therefore, any comments here are about the default interface only.

Users are able to configure their web-mail settings to include sort order, number of messages to display per screen, which mailboxes to display, signature and reply formats and too many other things to cover here, but suffice to say it's quite nice. Users also have the option of polling external mailboxes via POP3, and as an added bonus, can use the web interface to upload and maintain their personal web site. A user may also define his own

rules, using the same options as the server-wide rules—an option any user subscribed to mailing lists will definitely appreciate.

All is not perfection, however; a few annoyances remain, most notably the default “Delete” and “Next Unread” operations. Upon deleting a viewed message, a user is brought back to the Mailbox, rather than the more usual behavior of moving on to the next unread message without requiring user intervention. The “Next Unread” seems to ignore any sort order a user has specified and moves on to the next unread message in its mail file rather than using the sort order for displaying messages. This may sound minor, but it can be quite frustrating with a folder containing a lot of unread messages. The Delete behavior can (thankfully) be fixed easily by making a minor change to the HTML.

The last remaining quirk relates to the way some mail clients send web pages as attachments. Under some circumstances, the user is confronted with a message about “Embedded Web Pages” and his browser being unable to display in-line frames. So far, Netscape Messenger and Outlook Express produce the same result. When this happens, the user must click on a link to launch another browser window in order to view it. Hopefully, this will be fixed shortly.

Users, Lists and Normal Operations

I've used everything from Pine to Netscape and Outlook to retrieve mail from CGP and have yet to run into a problem using these clients via IMAP or POP3. Mailing lists are easy to set up, and also allow accessing them via the Web. They can be configured to automatically digest and support the typical features under traditional list servers such as majordomo, allowing for both moderated and unmoderated, public and private lists. The help files, list notifications and bounce processing are easily configured and customizable.

The Good

- easy installation & administration
- many tools and features
- highly customizable
- thorough administration package
- mail transport agent and web e-mail service
- retrieves POP3 or IMAP

The Bad

- next unread operation is disorganized

- delete operation exits folder
- web page attachments are problematic

Day-to-day administration is minimal, since the log rotation and most tasks are handled for you (once configured). A note of caution is in order, however; the logs can get quite large if all data is being logged, so after determining the server is in working order, logging policies should probably be altered. A “Monitors” section of the Admin interface allows you to watch all active connections via the Web and other protocols and view any logs, including filtered logging by content or time. One minor complaint is the log is displayed in reverse order with the most recent events displayed at the bottom of the page, which may be off the screen. A nice addition would be to send a notification if more connections are being requested than are configured, to enable system administrators to determine more easily when they may need to allow more connections or upgrade hardware. Aside from that, most administrators should rarely need to use the monitors section as the system practically runs itself, but it's good to know it's there.

Target Audience

With the abundance of features packed into CGP, it may sound a bit overwhelming to some system administrators at first, but let me assure you this is most certainly *not* the case. Most features are easily configured, and if you don't need a particular service, don't run it. Click a checkbox and it's off—it's that simple. The Web Mail interface obviously targets free e-mail services akin to Yahoo! and Hotmail, but it is much more than just Web Mail and should be equally at home in small companies, ISPs or large enterprise networks. For those interested in using it as a free web mail server, header and trailer fields are allowed to be set. I've seen people quote anywhere from a dozen to over 100,000 users on the mailing list so far, and just in case you want to support more users, CGP also has a “Cluster” option that allows you to use multiple servers in tandem.

Customer Feedback—Some First-Hand Comments

Software evaluations and reviews often leave out an important part—real-world use. A quick e-mail to the mailing list yielded several responses from different companies currently using CGP, ranging from a school with 1200 or so accounts to several companies using the web interfaces for free e-mail services supporting tens of thousands of users.

Ofer Tanenbaum of Zipmail (http://www.zipmail.com/), which offers a free web-based e-mail service, says Zipmail already has over 20,000 users and is growing daily, running on an Intel-based FreeBSD server with less than .07 CPU load and over 100 concurrent connections. Zipmail had tried EIMS and Appleshare IP

previously, and Tanenbaum had some personal experience with MS Exchange. His overall impression? “Bottom line, nothing to compare.”

MauiMail (http://www.MauiMail.com/) is also using CGP for web-based e-mail and has been running CGP for about a year, on an Apple G3/450. An employee of MauiMail, Darly Hansen, claims it to be “much smoother than our previous system. Very stable.” Most other responses I've gotten from the list tend to agree.

Pricing and Support

Pricing is free for a trial version and starts at $499 to license 50 users, 1000 accounts at $1999, on up to $30,000 for unlimited accounts and lists. Support packages vary, but free support via e-mail is included, with additional packages ranging from five incidents through unlimited 24x7 with guaranteed on-site support at varying costs. I've monitored the mailing list for several months, and for the most part problems are answered the same day, sometimes within hours, including on weekends.

Some of the list members have been doing heavy customization to the system, mainly the Web Mail configuration. While not always able to resolve an issue immediately or the first time in some cases, there almost always seems to be a way to do what the customer wants.

The documentation does a good job as well, but can be a bit overwhelming in size. On-line help is available to the administrator and users alike through the web interface. The list also serves as a place to make feature requests, and staff members at Stalker are very motivated to keep their customers happy.

Conclusion

Many products in the past have tried to be all things to all people, ranging from Windows and all its permutations to the different groups working on making Linux alternately either a desktop OS or a server OS. Most have mixed results, giving up something in trying to be too many things. Surprisingly, this is not the case with CommuniGate Pro. With its abundance of features and ease of administration, it practically runs itself once configured and is up to any task you may ask of it. If you need a feature, chances are there's already a way to do it, or it's coming soon. So, is it perfect? No. However, while it could use a few tweaks here and there, most are minor, which is all the more impressive considering the scope of the product. So who needs Exchange or cryptic Sendmail and procmail configuration? Not me, I'll take my CommuniGate Pro!

Scott Wegener started programming in BASIC on a TRS-80 CoCo way back when, and has been using Linux since before it came on CDs. He is currently

employed as a software engineer and sometime system administrator by VERITAS Software, where he develops cross-platform reusable components. He can be reached at wegster@mindcore.net.

Archive Index Issue Table of Contents

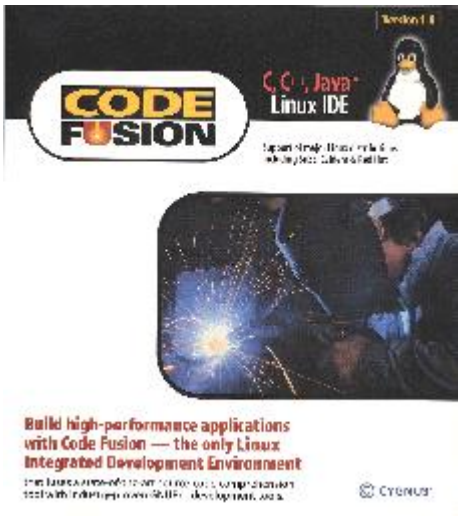
Advanced search

Code Fusion Version 1.0

Daniel Lazenby

Issue #70, February 2000

The Code Fusion environment includes an ANSI/ISO C compiler, an ANSI Tracking C++ compiler and a Java compiler.



- Manufacture: Cygnus Solutions
- E-Mail: info@cygnus.com
- URL: http://www.cygnus.com/
- Price: $299 US
- Reviewer: Daniel Lazenby

Code Fusion is Cygnus' next step in providing Linux developers with an integrated C, C++ and Java tool suite. The Code Fusion environment includes an ANSI/ISO C compiler, an ANSI Tracking C++ compiler and a Java compiler. A linker, build utility, debugger, source browser, an interface to version control systems, plus some other utilities are also included in this IDE (integrated development environment). Could this be an IDE for the masses?

Code Fusion is an integration of two other Cygnus products: Source Navigator and Cygnus Insight (also known as GNUPro Toolkit. Each of these products has

been reviewed in *Linux Journal*.) The integration of these two products provides a very usable integrated development environment. A primary advantage of the Code Fusion IDE is its quick visual access to project files, commands and build configuration files. Another positive attribute of Code Fusion is the compiler. It has been optimized for the Intel Pentium processors. Some benchmarks indicate a significant improvement in compile speeds. Compiles of my sample C, C++ and Java code went very quickly. Unfortunately, my code was too short to get a good measure of the compiler's true speed improvement.

In addition to the C and C++ compilers, Code Fusion includes a GNU Compiler for Java (GCJ/gcj). GCJ is a front end for the gcc compiler. GCJ accepts .java, .class, .jar and .zip file types. Using the **-C** option, **gcj** makes .java files into .class files. GCJ is capable of natively compiling both Java source and bytecode. More information on the gcj compiler is available at http://sourceware.cygnus.com/java/.

Code Fusion version 1.0 GCJ is compatible with JDK 1.1 and includes some JDK 1.2 functionality. Several pages of the manual are devoted to listing the specific Java 1.1 and 1.2 classes/methods/fields that are/are not included in the distribution.

Opening a Java project with Code Fusion provides the same kinds of access to Java information available for C and C++ projects. For example, the Source Navigator's Symbol Browser provides a visual representation of a Java project's files, classes and methods. Selecting a file, class or method opens the Source Navigator's Editor Window. Within this window are four browsers, an editor and search tools. A GUI compiling function is included within the Source Navigator portion of Code Fusion. Insight, Cygnus' GNU debugger, may also be used to debug compiled Java programs. I found this tool very useful in understanding some Java code I had acquired.

Code Fusion's target audience seems to be the experienced individual developer. I feel the Code Fusion IDE code-building conventions may not make a lot of sense if you are not already familiar, or comfortable, with building applications from the command line. The example application-build tutorial provides adequate guidance on what IDE dialog boxes to use, how to access the IDE dialog boxes and what data are needed to build the example code. There is little guidance as to why you are using a dialog box or why you are entering the requested data. I feel this understanding can come only from having learned to do it the old-fashioned way.

As with Source Navigator, Code Fusion's Version Control is based on RCS, CVS, SCCS and the Clear Case product. There were some version control functionality bugs in my review copy of Code Fusion. I could only check project

code in or out. I got error messages whenever I tried to use any of the other IDE version control functions, so I contacted Cygnus about the symptoms I was receiving. Cygnus promptly informed me they were aware of the symptoms and were in the process of testing a fix. This should be fixed by the time this review is published.

Installation

Code Fusion v1.0 supports Caldera 2.2, Red Hat 5.2/6.0 and SuSE 6.1 Linux distributions. This product requires a Pentium with memory—Cygnus recommends 64MB. This product also needs about 200MB of free disk space to install. My Caldera distribution mounts the Code Fusion CD-ROM as a non-executable device. A README file containing guidance on mounting an executable CD-ROM device is included.

Installation is divided into loading the application and configuring it. There is a CLI and a GUI install script; I chose to use the GUI script. Code Fusion came with a preprinted CD asset key. The asset key-based install means you need one licensed copy of the product per developer. Installation involves entering the asset key, typing in the installation directory and choosing the type of install.

After entering the asset key, you are presented with a graphic of the file systems and told to select your install directory. At the bottom of this graphic is a text field displaying the default path. The release I reviewed would not respond to a directory selection—I had to type over the contents shown in the directory text field. A custom install option provides the means of selecting which Code Fusion files or components to install. Two status bars are presented once the install begins. One bar indicates which of the seventeen files is being installed, the other indicates the file's install progress. The install is not complete until a couple of environment variables are configured.

I once saw the phrase “Here there be dragons” on a museum map. The “Here there be dragons” phrase truly applies to the environment-variable configuration step. These variables *must* be configured prior to running Code Fusion. In addition to adjusting the standard **PATH** variable, two other environment variables need to be established to ensure Code Fusion uses the proper libraries in the proper sequence: **LD_PRELOAD** and **LD_LIBRARY_PATH**.

The **LD_PRELOAD** variable points to a library that must be loaded before any other shared libraries are loaded. There are two ways of establishing the **LD_PRELOAD** path: one is to include the variable in one of the traditional environment setup files, the other is to use the /etc/ld.so.preload file. Here is where the dragon lived. A simple typographical error in this /etc/ld.so.preload file raised havoc with my system, the likes of which you don't want to experience. I tried both maintenance and shell boots; neither of them were of

much use. I found it easier to restore my system from a backup I had made just prior to starting the install. The next time I got to this environment variable configuration step, I placed the **LD_PRELOAD** variable into a test user's .profile. This way, I could localize any problems to a single user ID. Needless to say, I was triple cautious when it came to setting up the **LD_LIBRARY_PATH**. The test user ID .profile was used for the first implementation of this variable as well. Once I had proven to myself it was safe, I moved the variables to an environment file, where they had a more global effect. I did not use the /etc/ld.so.preload file. Be sure you can recover your system before you start configuring these environment variables.

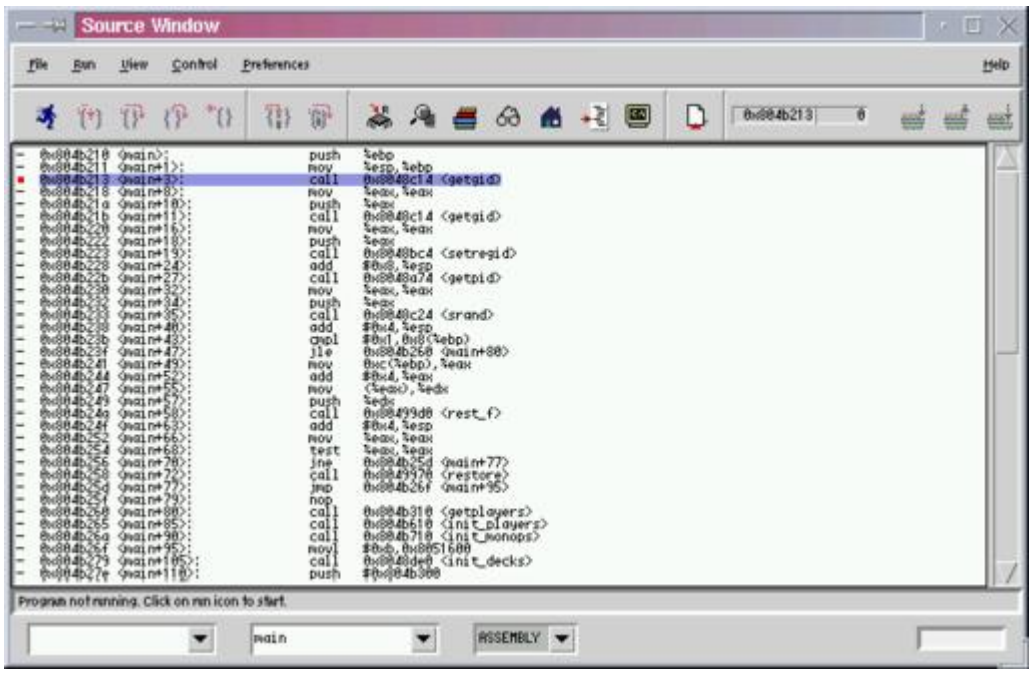


Figure 1. Example Window

Documentation

In general, the Code Fusion documentation is clear and concise. I do have a couple of minor complaints, though. The manual is primarily a composite of extracts from the Source Navigator “User's Reference Guide” and the GNUPro Toolkit “Getting Started” manual. Some new pages cover the integrated and new features. The manual highlights basic Source Navigator interface characteristics, and after highlighting the characteristic, the reader is then referred to the Source Navigator's “User's Reference Guide” for more details. Other parts of the Code Fusion manual directed the reader to Source Navigator's “Programmer's Reference Guide”. The author of this manual seemed to be working under the assumption that a Code Fusion owner already owned a copy of Source Navigator. This premise may be true for a short while after the initial release of Code Fusion. Once Code Fusion has been on the market for a while, this premise will no longer be valid. Code Fusion documentation should be able to stand on its own. The author shouldn't count

on the buyer of Code Fusion already owning a copy of GNUPro or Source Navigator.

The on-line Code Fusion documentation resembles the manual information. I was unable to locate the on-line Source Navigator or GNUPro documentation that was often referenced in the Code Fusion manual. Within the on-line documentation are references to unavailable user guides. For example, the "Launching the Insight Debugger" topic referred the reader to the GNUPro Getting Started Guide. No hyperlink to the referenced Guide was present, and I was unable to locate the document in the product install directories. At a minimum, I feel the proper reference documentation should be made available in either hard copy or on-line.

The Code Fusion product contains GNUPro and other GNU-licensed products. Chapter 2 has eighteen pages of general license and terms for use and distribution of applications created with the Code Fusion IDE. Reviewing the various licensing terms and conditions is worth the time it takes. The rest of the manual is devoted to describing how to use the basic Source Navigator and GNUPro Toolkit interfaces.

Six demonstration projects are included with Code Fusion: one for COBOL, FORTRAN, C++, Assembly, Java and a program called **monop**. The monop program is used for Code Fusion demonstration purposes. There are supposed to be README files for the other demonstration projects, but I was unable to locate several of them. The README files I did locate did not provide any useful information.

Support

As with other Cygnus products, 30 days of installation support is included with your purchase. Cygnus also maintains a Code Fusion support web site. The web site was still under construction at the time of this writing. When I visited the site, I found a FAQ, bug list and a patch database. A developer discussion forum and a couple of other features were listed as "coming soon" and may be implemented by the time this review is published.

Conclusion

Overall, I feel Cygnus did a great job of integrating the two products into an application IDE. I found Code Fusion easy to use and easy to navigate around. The interface between Source Navigator and GNUPro worked very smoothly. Code Fusion's management of project files takes only a couple of mouse clicks to add, delete or apply revision control. Please note, there is a slight IDE learning curve if you have not used either the Source Navigator or GNUPro Toolkit products.

Before paying list price, check on what upgrade paths are being offered. At the time I wrote this, Cygnus was offering three upgrade paths. Each offered a significant savings over the product's list price.

Daniel Lazenby (d.lazenby@worldnet.att.net) first encountered UNIX in 1983 and discovered Linux in 1994.

Archive Index Issue Table of Contents

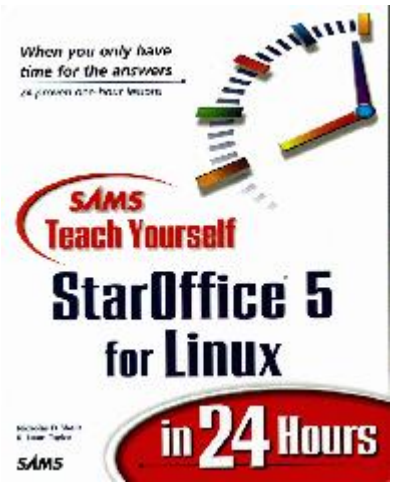
Advanced search

Teach Yourself StarOffice 5 for Linux in 24 Hours

Ben Crowder

Issue #70, February 2000

You will probably use StarOffice mostly for word processing, spreadsheets and presentations, and the sections are appropriately organized.



- Authors: Nicholas D. Wells, R. Dean Taylor
- Publisher: Sams Publishing
- URL: http://www.mcp.com/publishers/sams/
- Price: $19.99 US
- ISBN: 0672314126
- Reviewer: Ben Crowder

Teach Yourself StarOffice 5 for Linux in 24 Hours is a well-written book that claims to teach you, in 24 one-hour lessons of a chapter each, how to use StarOffice, a complete office suite from StarDivision.

The book is split into five parts. You will probably use StarOffice mostly for word processing, spreadsheets and presentations, so the sections are appropriately organized. Part One, “Preparing to Use StarOffice”, goes over the basics of installing StarOffice, getting started with the program, using the built-

in StarOffice Explorer and Desktop, configuring StarOffice, importing and exporting documents, and creating graphics with StarDraw. The last chapter of part one, on StarDraw, seems a little out of place in this section and would probably have worked better in Part Four, “Working with Presentations”. Except for that, however, these six chapters give a thorough enough introduction to StarOffice, one that will definitely help you on your way.

StarWriter, the StarOffice word processor, is covered in Part Two, “Creating Documents with StarWriter”. Topics in this section include creating new documents, formatting (fonts, margins, etc.), advanced formatting tools (footnotes, columns and text styles), tables and indexes, inserting graphics and the spell checker/thesaurus. The areas covered are good choices, going over a wide area of StarWriter's capabilities. This means, of course, that most topics aren't delved into very deeply, but that's to be expected when each chapter is designed to take up only an hour. For one-hour lessons, some of the chapters cover quite a bit—definitely enough to get you started.

Part Three goes over StarCalc, the StarOffice spreadsheet program. This section could very easily serve as an introduction to other spreadsheet programs as well (with the slight differences inherent in each program added). Six chapters are devoted to StarCalc: creating spreadsheets, entering spreadsheet data, using formulas and functions, formatting, adding charts and graphics, and using StarCalc's database functions. Again, these areas are well-chosen, covering both the basics and some of the not-so-basics.

StarImpress, the StarOffice presentations program, is given three chapters in Part Four. You learn how to create presentations, add graphics and charts to those presentations, and then pretty them up for general consumption. Given that presentations probably aren't used as much as word processing and spreadsheets, the smaller amount of space devoted to StarImpress is acceptable. It serves well as a short introduction to StarImpress, helping you get started. Noticing a trend here? This book isn't an in-depth discussion of StarOffice—rather, its main purpose is to give you a quick overview, brief enough to be digested in a day.

Part five, “Using Internet and Scheduling Features in StarOffice”, also devotes three chapters to these functions. There is a chapter on creating web pages, one devoted to e-mail and newsgroup features, and finally a chapter on StarSchedule, the StarOffice scheduler. These also introduce their topics, giving you just enough information to have an idea of what you're doing.

Does the book succeed in its title claim? Yes, it does. If you want a quick introduction to the many different facets of StarOffice, one you can get through in just a number of hours, buy this book.

Ben Crowder is a young Linux aficionado living in Utah. In addition to fiddling with the insides of computers, Ben enjoys reading, writing and music. He can be reached at mlcrowd@enol.com.

Archive Index Issue Table of Contents

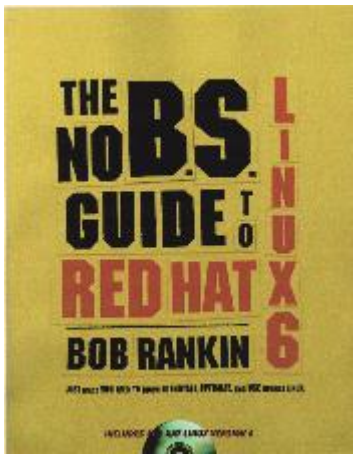
Advanced search

The No B.S. Guide to Red Hat Linux 6

Harvey Friedman

Issue #70, February 2000

The writing is concise and in places even terser than an uninformed reader might desire. Overall, I think it succeeded.



- Author: Bob Rankin
- Publisher: No Starch Press
- E-mail: info@nostarch.com
- URL: http://www.nostarch.com/
- Price: $34.95 US
- ISBN: 1-886411-30-1
- Reviewer: Harvey Friedman

No B.S., really? was my first question when picking up this book. After reading it, I can say that the writing is concise and in places even terser than an uninformed reader might desire. Overall, I think it succeeded.

The book contains 15 chapters and some useful appendices. Chapter 1, “Installing LINUX on your PC”, starts off by assuming the user has a Microsoft OS on his computer, has a net connection and web browser, and wants to run more than one OS on the computer. The instructions are fine if one does not

run into any problems, but if one does, the recommendation is to go to a web site or Usenet newsgroup to read and ask questions. How many MS users have the patience to do that? Other chapters are as follows:

- “GNOME, the Linux GUI” describes the GNU Network Object Model Environment and Enlightenment.
- “Connecting to the Internet” via a serial port and modem.
- “Living in a Shell” is an overview of bash, piping and redirection.
- “The Linux File System”
- “Important Linux Commands” contains brief descriptions of commands and configuring a printer with X.
- “Text Editors” covers vi, Emacs, pico, gnotepad+ and gEdit.
- “Slicing and Dicing” describes filter commands such as sort, uniq, grep and find with examples of using them with pipes.
- “Rolling Your Own: Linux Programming”
- “Managing your E-mail” is mostly about Pine.
- “Compression, Encoding and Encryption” introduces tar, gzip, compress, shar, et al.
- “Linux does DOS and Windows” discusses mtools, **mount**, DOSEMU, WINE, VMware and Macintosh tools.
- “Tweaking Linux” describes using the setup utility for configuration, X, rc.d directory, linuxconf and more.
- “Updating your Linux System” explains RPM in great detail.
- “Learning More About Linux” covers web sites to visit for more information.

Appendix A lists all software packages on the Red Hat 6.0 CD-ROM included with the book. One particularly good feature is an indication of how much disk space each will take. This could be quite useful for someone trying to put together a fun setup on a small hard drive. Appendix B contains “The GNU General Public License”. Appendix C, “DOS and UNIX Equivalencies”, lists a couple of dozen parallel commands with examples of usage.

I do have a few criticisms of this book. In Chapter 1, several examples of hard drive partitions and/or CD-ROM drives referred to by their /dev file names are given, but I don't believe the average Microsoft software user could configure a system with just the /dev information in this book. It was a serious omission not to list the conventions for IDE, SCSI, etc. Luckily, the CD-ROM installation procedure may help the uninitiated. Another problem was the statement “...but you can't really hurt anything if the installation isn't successful....” Without qualifying that your original system absolutely must be backed up for this to be true so that one can restore if necessary, I don't understand how using DOS'

fdisk can't help hurting something. It would have been nice for the author to guide the user through **fips** and not just fdisk.

I also had a problem with the facetious sentence in Chapter 13:

> If you think you did everything right and the mouse still doesn't work, sometimes whacking it on a hard surface while shouting, “You stupid mouse!”, works wonders.

In my experience, the “wonder” is the mouse ball mashes a sensor and one must then obtain a new mouse.

Although Rankin emphasizes using GNOME and describes how quite well in several chapters, he does not neglect the command-line interface. Particularly good were chapters 2, 4, 7, 8, 10 (the description of Pine), 11 and 14 (the detailed explanation of RPM).

For those whom I think of as typical Microsoft users, I recommend they follow the step-by-step installation procedures in *Linux for Dummies, 2nd Edition*, then move on to this book after they have a working Linux system. The installation process described there will work with the CD-ROM that comes with this book as well. For the Microsoft user who is willing to read first and needs little guidance to solve problems easily, I also recommend it. For someone who has already installed Red Hat 6.0 Linux and wants to learn more about its new capabilities, I wholeheartedly recommend this book.

Harvey Friedman can be reached via e-mail at fnharvey@u.washington.edu.

Archive Index Issue Table of Contents

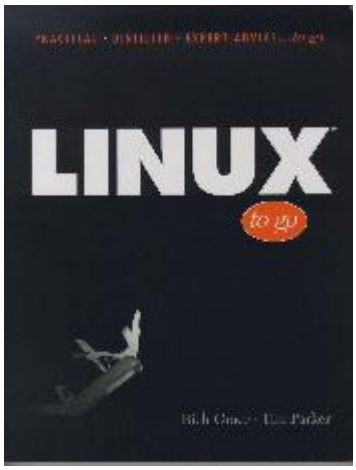
Advanced search

LINUX to go

Marjorie Richardson

Issue #70, February 2000

The presentation of the information in the book as well as the layout is designed to give the reader just what he is looking for and no more.



- Authors: Rich Grace and Tim Parker
- Publisher: Prentice Hall PTR
- E-mail: corpsales@prenhall.com
- URL: http://www.phptr.com/
- Price: $34.99 US
- ISBN: 0-13-999269-3
- Reviewer: Marjorie Richardson

LINUX to go is billed as a book for the intermediate Linux user. It skips all the installation procedures, assuming you already have Linux installed. Also, the information is presented without any background material or newbie-type instructions, with the assumption you have been using Linux a while and know the first steps. Still, the information is not too advanced. The target audience seems not to be a truly intermediate user, but someone who is just far enough along to no longer consider himself a newbie—perhaps a “newly intermediate”

user. For example, sections are included on the basics of Linux command-line entry, learning path names and setting permissions. However, for the most part, the authors do assume you have some elementary knowledge.

The presentation of the information in the book as well as the layout is designed to give the reader just what he is looking for and no more. It is easy to find any subject of interest. Much is presented in the “if you want to make this event happen, do these commands”, a format I like very much.

The authors go on to more advanced subjects, such as basic network configuration, connecting to the Internet, INN and NNTP, configuring DNS, Apache, Samba and rebuilding the kernel. There are also chapters on using Linux and Windows together, using Linux on a laptop, and the all-important security chapter.

There was no “about the authors”, so I cannot tell you their credentials. I did notice that Rich Grace is also the author of *WINDOWS 98 to go*. I always like to know something about the authors of a book and why I should trust their instructions. It is nice to know if the author is an expert on the subject or just a good writer who has taken the relevant information from others and put it into a readable form. Anyway, I consider the omission of this section the book's most serious lack—not too bad, when you think about it.

The book is well-written and well-designed; the information accurate, easy to read and understand. If you are past the newbie stage and ready to go to the next level, this could be the book for you.

Marjorie Richardson is Editor in Chief of *Linux Journal*. She is enjoying all the attention Linux has been getting lately and looks forward to watching it grow even more in the coming years.

Archive Index Issue Table of Contents

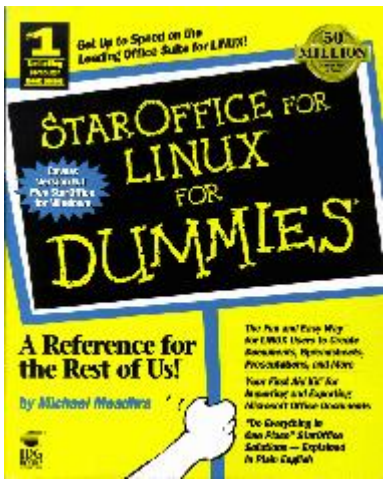
Advanced search

StarOffice for LINUX for Dummies

Sid Wentworth

Issue #70, February 2000

The quick summary is that this is a somewhat strange book.



- Author: Michael Meadhra
- Publisher: IDG Books Worldwide, Inc.
- E-mail: siteemail@idgbooks.com
- URL: http://www.idgbooks.com/
- Price: $19.99
- ISBN: 0-7645-0576-9
- Reviewer: Sid Wentworth

The quick summary is that this is a somewhat strange book. I had intended to say something like “good” or “bad”, but neither word fits—“strange” does.

The book is organized into eight parts with multiple chapters plus an appendix on installing StarOffice. The first part has general information on StarOffice. The next five parts cover StarWriter, StarCalc, the graphics pieces of StarOffice (StarDraw, StarImage and StarPresents), StarSchedule and StarBase. The last

two parts include how to configure StarOffice for an Internet connection and a number of tips.

On the good side, there is a reasonable amount of useful information to help someone new to StarOffice get up to speed. The ideas are presented clearly, with illustrations where needed. It is also very easy to skip to the section of interest, StarCalc for example, and just start reading. As long as you either knew the basics or read Part I, you should be fine.

Now, about the strange stuff. First, in spite of the book's title, it also covers StarOffice for MS Windows. I find this a little strange, and more importantly, distracting. When you are reading about how to use a new application—particularly a large, complicated one—you don't need this sort of distraction.

The next weird thing is the “cheat sheet” in the front of the book. This is very common for this series of books, but the content doesn't make sense. With all the possible things a reader might want on a reference card, IDG chose to define what the icons mean on the various toolbars. If you have ever used StarOffice, you probably know that if you leave the mouse pointer over an icon, the function of the icon is displayed.

The system configuration chosen by the author is also a little odd. It is Red Hat Linux 5.2 with KDE 1.1 and StarOffice 5.1. The default Red Hat 5.2 system didn't run KDE. While StarOffice should run with any window manager, this deviation from the standard distribution is not explained. Also, the installation appendix covers installing StarOffice from StarDivision, not Sun. Things have changed significantly since Sun bought the product.

While I am on the subject of installation, I should mention StarOffice 5.1 automatically adds StarOffice to the KDE panel. The author, however, has you bring up a terminal window to start StarOffice. This makes no sense, as there is no need to associate a terminal with the program. Even if the author didn't want to use the icon, using the **ALT-F2** key sequence to bring up a command line would be more appropriate.

The author's background will shed some light on this subject. While he has written other books and clearly has software experience, there is no indication that he has any Linux experience.

I have other nits to pick with the book. For example, the “Entering Chart Data” for StarChart first insists that you must manually enter all your data. Then, it goes on to show how you can embed a chart in a StarCalc spreadsheet using the spreadsheet data. An introductory sentence indicating an alternative was coming would have made the reader feel much better.

In the chapter titled “Creating and Editing Images”, a “Linux-only” note talks about file and directory permissions. Unfortunately, it is written backwards (mixing up cause and effect) and is also incorrect.

That's about it from my end. The book does cover a lot of what you need to know about StarOffice, but it certainly isn't perfect. My recommendation is to look at the other book(s) available or wait for the second edition, which hopefully will have been technically edited by a Linux person.

Sid Wentworth is enjoying life on the Olympic Peninsula and resists any and all efforts to entice him back to Uzbekistan.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

SCSI—Small Computer System Interface

Keith de Solla

Issue #70, February 2000

Successfully installing a SCSI device on a Linux PC.

The SCSI (small computer system interface) bus is a system commonly found on workstation-class machines, but is available for other systems such as personal computers (PCs). The advantage of SCSI is that once the controller is set up, you can just add new devices (typically up to seven) with a minimum of fuss. The new device is merely plugged into the existing “daisy chain” bus. The user must ensure that the last device in the chain is properly terminated. We will discuss termination in a later section.

The idea to write about SCSI came out of a project I was working on to provide Linux-based web and network servers for a local materials science company, Fibics, Inc. (http://www.fibics.com/). The network server was to include hardware for performing file-system backups. I am not a SCSI expert and had to learn the hard way, so I thought I'd share the basics in hopes that others might benefit from my experience.

Where Do I Start?

You will need a computer (obviously), a SCSI controller card to match the bus type of your computer and one or more SCSI devices (disk drive, tape drive, etc). Note that different SCSI devices may require different SCSI bus types (SCSI-1 vs. SCSI-2 vs. ultrawide, etc.) and this may affect your choice of controller card. For more information on the different SCSI buses, I recommend the booklet *Basics of SCSI* from Ancot Corporation (http://www.ancot.com/).

Before purchasing hardware, you should review the relevant HOWTO documents (SCSI, hardware) and the hardware compatibility list for your Linux distribution. The HOWTO documents I read are from the Red Hat distribution and are installed in /usr/doc: /usr/doc/HOWTO/SCSI-Programming-HOWTO and

/usr/doc/HOWTO/unmaintained/SCSI-HOWTO. The HOWTO documents provide some useful information, although they are somewhat dated.

If your PC has no SCSI devices or controllers, you should start by determining what device you need. For my project, a high-capacity 4mm DAT drive was required. A review of available drives suggested a Seagate Scorpion (12/24GB) would meet the requirements. Once the device is selected, you can narrow down your choice of controller cards based on the type of SCSI bus the device requires.

The following steps were used to determine what controller card I would require:

- The target machine was an AMD K6-2-based PC with PCI and ISA buses.
- A 4mm DAT drive was needed for backup purposes.
- The selected drive required a SCSI-2 bus.
- An Adaptec 2910 controller supports SCSI-2 and is a PCI card; it is not specifically listed as incompatible in the hardware list.
- A driver for the 2910 was needed to run under Red Hat 5.2.

The most difficult task is determining the correct driver, as such information was not documented anywhere I looked. Seagate has an automated phone system (1-800-SEAGATE) which provides considerable information about their hardware. Unfortunately, on the software side only Windows 95/98/NT information was available.

The solution was to post a message to the local Linux mailing list. Many thanks to the individuals on the Ottawa Carleton Linux Users Group (OCLUG) mailing list for indicating that the “aic7xxx” SCSI driver module was the proper choice. The OCLUG URL is listed in Resources.

Another post to the OCLUG mailing list and some looking around led me to more detailed information. README files for the various SCSI drivers reside in /usr/src/linux-2.0.36/drivers/scsi/. Here, you can determine what cards or chip sets are supported by each driver. It would be worth looking at these files prior to purchasing a SCSI controller card.

To add the required driver, I started up Red Hat's control panel. I selected “Kernel Configurator” and clicked on “add”. A “Module Type” window opened; there I selected “scsi_hostadapter” from the “Module Types” pulldown menu. I clicked “OK” to open the “Module Definition” window, then clicked the “Which module” pulldown and selected the required driver. Finally, I clicked “OK” and then “Restart kerneld” in the “Kernel Configurator” window.

Hardware

Having determined what was needed, the hardware was purchased and installed. The Seagate DAT manual was not as clear as it could have been about installing a single SCSI device, but all the required information is there. Listening to the information on their automated telephone help line answered my remaining hardware questions. I'm providing sufficient information here (I hope), so you won't have to make the phone call.

Two rules must be followed when dealing with a SCSI system. One, every SCSI device must be set to a unique SCSI ID number. ID #7 is normally used for the SCSI controller card and ID #0 is often used by the boot drive. Two, a SCSI bus (cable) must be terminated at both ends in this manner:

- Internal devices use active termination on the last device in the chain, at the end of the ribbon cable.
- External devices require a terminator plugged into the unused connector on the last device in the chain.
- The controller card provides automatic active termination. If the card is used only for internal devices, the external connector need not be terminated.

In my particular case, I wanted to install a single internal device. The 2910 controller card was plugged into a free PCI slot on the motherboard. The DAT drive was configured for active termination via the appropriate DIP switch. Other DIP switches were used to set the drive to SCSI address #2. Then the drive was installed in a free 5-1/4-inch drive bay. Since the DAT drive was the only SCSI device being installed, it was plugged into one end of the SCSI ribbon cable. The other end was plugged into the SCSI controller card. The Adaptec 2910 “kit” came with the ribbon cable that allows two devices to be connected to the card. The card also has an external SCSI connector which was not used (nor terminated).

Backup Software

While my plan was to cover SCSI, I thought a listing of backup software would also be relevant, since I installed a tape device. Options range from using basic Linux utilities (tar) to expensive commercial solutions.

Some of the commercial backup solutions available are listed in Table 1. The list is only a sample, and I am sure many other solutions are available. To any companies I may have left out: speak up in the letters section of *LJ*. Some of these commercial products are free for personal use. Both BRU200 and PerfectBackup may be included with some Linux distributions. Arkeia was reviewed in the April '99 issue of *Linux Journal*.

For KDE users, there is a backup utility called KDat which is documented in the KDE on-line help. For further information, see the web pages shown in Table 1 or the Linux applications page listed in Resources.

Conclusion

To summarize, my intent was to help the beginner answer the questions:

1. What do I need?
2. Where do I start?
3. Where do I find more information?

Sufficient information is provided to enable the beginner to install one or more SCSI devices in a Linux-based PC. Pointers to other sources of information will hopefully answer any questions that this article did not.

Resources

email: kdesolla@shield.com

Keith P. de Solla, P.Eng is a VLSI CAD Applications Engineer and Linux supporter. When he's not sitting in front of the computer, he can be found fishing, hunting, or target shooting. He can be reached via email at kdesolla@shield.com

Archive Index Issue Table of Contents

Advanced search

HFS Utilities

Marjorie Richardson

Issue #70, February 2000

Data on Macintosh disks can be read into Linux quite easily with this tool package.

Macintosh users may be a minority, but they are quite loyal to their operating system and loathe to give it up. Mac OS stores its files in HFS (hierarchical file system) format, which uses a return character to separate lines in text files, rather than a newline character as in Linux/UNIX. If you have files on a Macintosh floppy disk that you wish to get on your Linux system, several options are available.

Just as “m” commands deal with DOS disks, “h” commands work for Macintosh disks. These commands are named and behave just as any Linux user might expect. For example, to change directories on your Linux system, you type **cd *path***; to do so with a Mac volume, you type **hcd *path***. In other words, you just add an “h” as the first character of the command name. In some cases, the base command name is taken from DOS rather than UNIX—I guess to make the h tools consistent with the m tools. The most frequently used h utilities are:

- **hcd *path***: change working directory on the disk; default is the root (/) directory
- **hdel *names***: delete a file or files on the disk
- **hcopy *source destination***: copy a file to or from an HFS disk
- **hdir**: list the files on the disk; **hls** also works for this purpose
- **hmkdir *path***: create a directory on the disk; **hrmdir** removes a directory on the disk
- **hmount *device_name***: mount the HFS disk at this location; **humount** unmounts the disk

Other available commands are hformat, hpwd, hrename, hvol and hatrib. For details on these HFS utilities and their options, see the man pages.

Wild cards are permitted in specifying path names, but be aware that if you use "*" to mean "every file in the directory", it will not be interpreted that way. For example, typing **hcopy * .** will look for the filenames on the HFS floppy in the current directory , instead of getting all the files off the HFS floppy as you would expect.

Another method for manipulating Mac disks is the **hfs** shell. This shell, based on Tcl, provides the same type of options as the h commands: mount (m), umount (u), cd (c), dir (d), mkdir (m), rmdir (r), etc. The difference is only in the way the command is specified. For example, to mount the disk with hfs, you type:

```
hfs m
```

instead of

```
hmount
```

Finally, Robert Leslie's program **xhfs** provides a graphical user interface to the hfs commands. It is easy to use and quite intuitive. Just type:

```
xhfs
```

and up pops a window displaying available options (see Figure 1). A click of the mouse on the button next to the option you want, and you are on your way to having those files on your system in Linux format.

If a friend sends you a Macintosh file, don't forget you need to get rid of those pesky return characters. The **tr** command (see "A Little Devil Called tr" by Hans de Vreught, *LJ*, September 1998) is the best tool for doing this. Typing

```
tr '\015' '\012' <
```

will do the trick.

Another good resource for these commands and others is *Linux for Dummies Quick Reference, 2nd Edition* by Phil Hughes.

Marjorie Richardson is Editor in Chief of *Linux Journal*. A movie aficionado, she is currently enamored of DVD technology and can't bear to watch movies "that have been formatted for your TV screen".

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Automated Imaging Microscope System

Jason Neudorf

Steven A. Garan

Issue #70, February 2000

Come up to the lab and see what's on the slab—I mean, slide.

The Aging Research Centre (ARC) is developing an Automated Imaging Microscope System (AIMS) to enable researchers to view the three-dimensional cellular organization of a region of tissue from a series of microscope slides. AIMS employs a computer-controlled microscope that can move a microscope slide along the X and Y axes, so that the field of view can be moved in equal increments. The microscope also has an autofocus mechanism. A CCD camera captures images, one field of view at a time. Using a magnification of 400 power, a typical slide contains 25,000 fields of view; perhaps 7000 of those fields contain something of interest. By using the thousands of individually captured images, the system is able to reconstruct a very high-resolution image of the entire slide.

AIMS analyzes each of the thousands of images and identifies and stores the following attributes for each cell it was instructed to identify: X,Y and Z coordinates for each desired cell type, amount of each stain in the cell, type of each stain in the cell, color of each stain in the cell and type of cell.

When AIMS has processed multiple layers of a tissue block, the system will be able to reconstruct a three-dimensional map of the cells in that block. In an experiment where there are two groups of animals and the researcher wishes to compare tissues from the two groups, AIMS will be able to compare three-dimensional structures as well as cell counts.

The Implementation

The system has two very different functions: acquisition and analysis. When we designed the system, we started on both ends simultaneously, attempting to

analyze images captured from other sources, then working on our own data. Because we hope for the system to be used by biology researchers, not UNIX gurus, a decent user interface is one of the primary goals. We chose Tcl/Tk for the ease with which a GUI can be developed. The easily modifiable prototypes are useful. Various commands, written in C (but with Tcl/Tk interfaces) are used to control physical devices.

Physically, the system consists of an optical microscope, a CCD camera and three stepper motors hooked to the parallel port. The stepper motors move a microscope slide around the stage under computer control; basically, we have a 60,000bpi scanner.

Stepper Motors

The stepper motors are driven by Darlington-pair transistors (see http://www.doc.ic.ac.uk/~ih/doc/stepper/ for a discussion on stepper-motor control). This allows for great control, at the expense of hefty timing requirements. In the Linux kernel, we can find floppies driven every three to eight milliseconds. Empirically, given the load on our motors, we can sometimes move more than twice a second. We've hooked them up to the parallel port. The parallel port has eight data lines and four control lines which can be used for output. This is just sufficient to control three stepper motors at four controls/motor. If we needed more motors, we would have to use a different type of controller. Figure 1 shows the microscope looking at the focus mechanism. Figure 2 shows the circuit driving the motors. It uses two ULN2003A chips and is powered by a spare computer power supply.

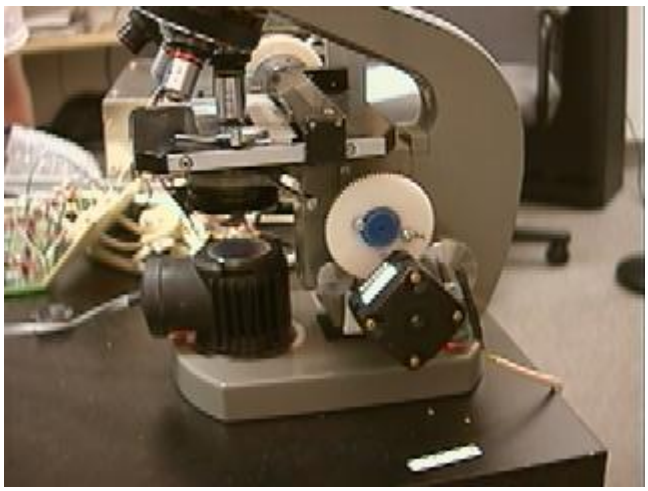


Figure 1. The Microscope

We need to do a lot of moving, so speed is very important. Using **nanosleep** seems the simplest, if not the best, alternative. Combined with real-time priority, this causes the motors to move with a nice smooth hum. The other alternative, using the real-time clock at 2048KHz, doesn't allow as precise a control over speed. The major problem with this approach is the way

nanosleep handles its delays. A busy-wait prevents any other task from running. RTLinux does seem like a better solution, although we haven't investigated that yet. The older **usleep** call is a poor choice, as it has 10ms granularity.

Figure 2. Circuit that Drives Motors

The greatest problem is figuring out the speed to move the motor. The speed at which a stepper motor is capable of moving depends on the load placed on it, and this load varies depending on the friction in the stage. A speed that works at one location may not work at all at another. Empirical experimentation seems necessary.

We can move in three dimensions. Not only can we view the entire slide, we can change the focus. A jury-rigged solution seems to work—move up and down, and pick the image with the most detail. We assume the one with the most detail is the most focused. Technically, “detail” is based on a “busyness” function. For each pixel, find the difference in intensity between each of its neighbors, then sum the absolute value of those differences.

Frame Grabber

Because this project started before Linux 2.2, which has frame grabbers built into the kernel, we are using the Matrox meteor frame grabber, which has support at www.gnofn.org/~marksu/meteorman.html. The **mvid** program which comes with the driver was a useful starting point. We integrated it with Tcl/Tk. This allows us to make snapshots, view real-time video at variable frame rates and sizes over our network and get measures such as “how dark is it?” or “how much detail is there?” by directly accessing frame-buffer memory.

Real-time video is useful for manual focus to check whether autofocus is working, for setting boundaries of the scan, and of course, for just joking around by taking pictures of staff members. The “meteor” driver sends out a signal whenever a new image is available. If we're ready, we will send the image

out using **XPutImage** and **XSync**. If the previous image isn't done, we ignore the frame entirely.

Analysis

While shape is important, size and color are simpler to use as heuristics. We take a single image, then use sliders to select the colors which we consider to be a cell. If it is big enough and the right color, it must be a cell. This isn't a very sophisticated technique; it isn't much of a refinement over “thresholding”, where anything sufficiently dark is counted.

Currently, we use Tcl/Tk to select the ranges of RGB color which will be allowed. In the future, it may be useful to select regions in HSV color space.

The simplicity of the algorithm means cells can be counted “on the fly”; during the scan, the algorithm is performed on each field of view. The cells on the boundaries are counted multiple times, but we know where the boundaries are and can ignore them.

It would theoretically be possible to do this job without any computer at all. A technician could look at each slide, 0.2 mm at a time, and count every cell he saw. Looking at 2mm by 2mm sections, this would require exhaustive work for the 100-odd fields covered by a typical mouse hypothalamus. Fatigue could introduce bias. It would be easy to count a given marginal case one way when wide awake and another when tired—but people are good at image processing, computers aren't. People make mistakes when they are tired; computers make mistakes all the time. Still, even if absolute numbers are biased, we hope that relative numbers will still show useful differences.

User Interfaces Scanner

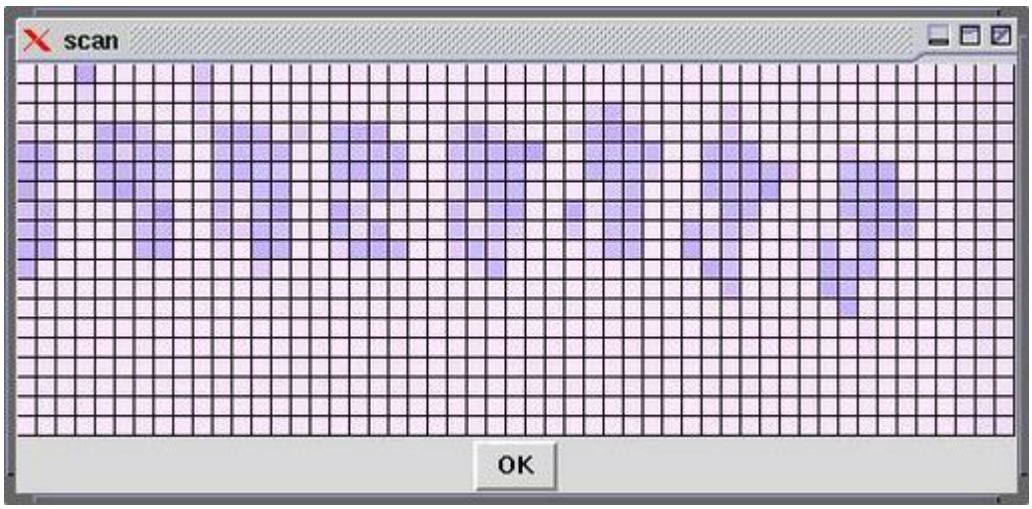


Figure 3. Overview of Slide

There are currently two interfaces to the physical “scanner”: one for grabbing an overview (see Figure 3) of the entire slide, at 25 bits per inch (i.e., the microscope's objective is moved 1mm at a time, and the average color at that point is saved), and another for grabbing a specified region on the slide. In the second case, because of the low speed of directory listings (**ls** takes quite a bit of time if there are 2000 files), a directory is created for every column scanned. Figure 4 shows the interface used to scan in a rectangular region. The user can use the cursor keys to move the slide, and then select the boundaries.

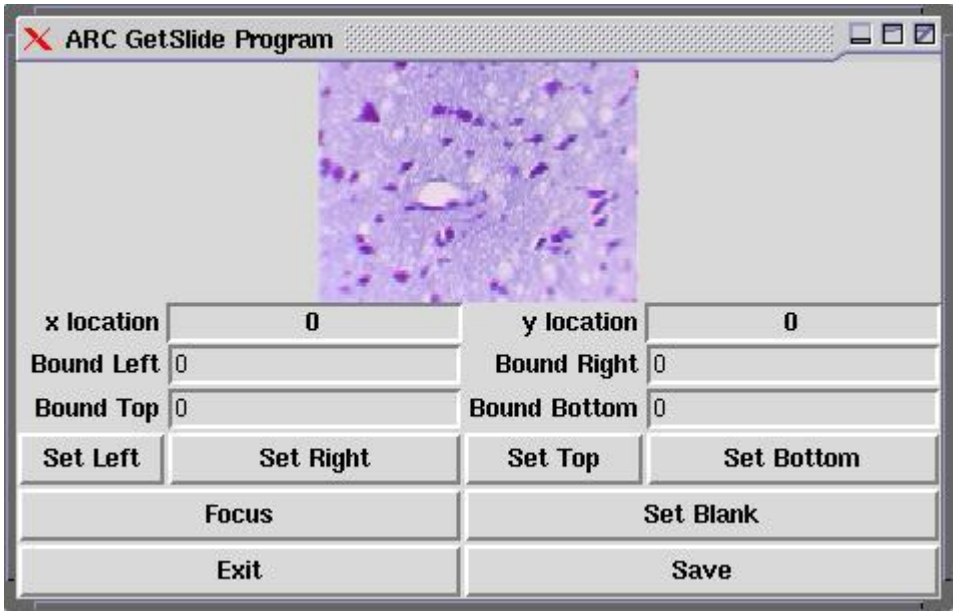


Figure 4. Scan Interface

One planned refinement is to scan in only those areas which we think may have useful content. If locations (x,y), (x+1mm,y), (x,y+1mm), (x+1mm,y+1mm) are all blank, it is reasonable (given the size of our samples) to ignore (x+0.5mm, y+0.5mm).

The optimal refinement would be to store only the regions which actually have useful content. In our case, we are interested in only the hypothalamus. An empty area is near this, which could conceivably be automatically recognized; if so, we could discard thousands of frames of less-important data.

It would be nice to store the entire slide in a standard image format such as JPEG or TIFF, but for some reason, 12,5000x50,000-pixel images are difficult to process at 24 bits per pixel (18GB per image seems a little excessive). Storing each frame individually using JPEG uses 10-50KB per frame; more for detailed ones, less for blanks. If only images with useful detail are saved, it should get under 650MB/slide, in which case each slide might be stored on a CD-ROM.

Panner/Zoomer

Given the non-standard format, we will need a way to view individual segments on a slide. A multi-resolution display tool uses preprocessed images from the slide to create a mosaic, which shows multiple frames at the same time. By saving each frame at multiple resolutions, we do not need to decompress hundreds of JPEG images, only to throw 9999/10,000ths of the detail away.

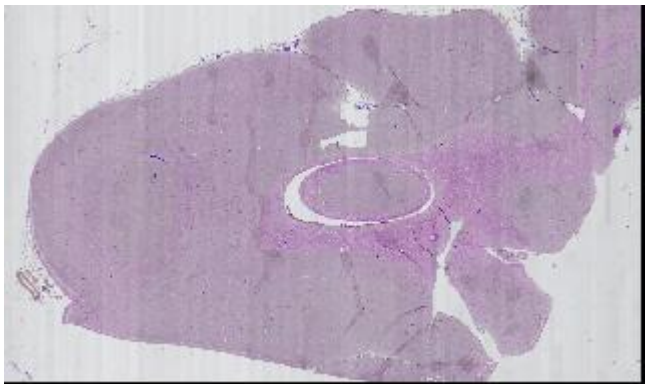


Figure 5

Using this tool, we can zoom out to view the entire slide, zoom in on a specific region of a single sample, then view a specific field of vision. Figure 5 is a mosaic of one portion of one slide.

Refinements would allow integration with the counting algorithm. By drawing a border around a region of interest, counts and densities of cells in that region could be displayed.

When successive sections of the same tissue sample are placed in a known sequence, those successive sections can be merged to form a single three-dimensional image of the original tissue. Distortions in the sample are likely to cause some difficulty, but standard feature-recognition techniques should be able to compensate.

A further refinement is possible. With cells ~25 microns in size, 4 micron slices span several cells. By staining successive sections with different factors, it is possible to determine several different facts about each cell.

The Automated Imaging Microscope System (AIMS) will be used by my colleague Lee R. McCook at the University of California at Berkeley. He will use the system to compare the 3-D cellular densities in the hypothalamus of normal mice and calorically restricted mice. I have downloaded his Ph.D. proposal to the Aging Research Centre's web site at www.arclab.org/linux/leephd.html.

Final Thoughts

ARC has two research facilities in North America: the first is located in Berkeley, California and the second in Waterloo, Ontario, Canada. The near-term objectives of the Aging Research Centre are to conduct experiments that shed light on the underlying mechanisms that cause the aging process to occur and to develop tools which will help researchers better understand why humans and other organisms undergo this process. Our long-term objective is to use the information that has been accumulated on this subject and develop intervention procedures that will dramatically slow down the aging process. The aging of an organism is a very complex and multifaceted process that encompasses many disciplines in biology such as neuroendocrinology, histology, genetics, enzymology, biochemistry, molecular biology and so on. At this time, there is approximately 100GB of information in journals and books that relates to the aging process, and more and more information is being added every day to this body of knowledge. One of the tools we are developing will allow a researcher to see this massive quantity of data via a computer using graphical and interactive methods in order to represent this information in a more comprehensible and coherent manner.

The other major tool is the focus of this article: an Automated Imaging Microscope System to be used for an experiment at the University of California at Berkeley. It will be used to study the effects of calorie deprivation on neural tissue. Mice on calorie-reduced diets often live longer, and this may be based on changes in the hypothalamus.

Jason Neudorf (jcjneudo@calum.csclub.uwaterloo.ca) started on computers with the Commodore PET and is quite happy that the hacker spirit continues in the Linux community. He recently graduated with an M.Math from the University of Waterloo.

Steven A. Garan founded the Aging Research Centre (http://www.arclab.org/) in 1994 and has been in the computer field for 22 years. Steven has worked on machines from TRS 80s to IBM 360/75 to HP3000s to Linux/Intel PCs, and has worked in Canada, UK, Italy and the USA. Of all the systems he has worked on, Linux has to be the one that is the most flexible and powerful environment he has seen thus far.

Archive Index Issue Table of Contents

Advanced search

Mark's Mega Multi-Boot Computer

Mark Nielsen

Issue #70, February 2000

Mark talks about his crazy multi-boot computer, which does have some practical value.

After I started going to computer conferences, I realized I needed to be able to demonstrate Linux working with other operating systems as well as different distributions of Linux. Having one computer for each operating system seemed a bit much, so it became important to have a multi-boot computer to reduce the number of computers I would need. The goal was to put several distributions of Linux, along with other operating systems, onto the computer and be able to choose which operating system I wanted when I turned on the computer. The other operating systems were Windows 98, Windows NT, DOS and FreeBSD.

The computer was originally configured on an IDE system as follows:

- Primary Partition 1: Windows 98 (2GB)
- Primary Partition 2: 100MB of Linux swap space
- Primary Partition 3: Red Hat Linux 5.2 (> 1GB)
- Extended Partition 4
- Logical Partition 5: Windows NT 4.0 (2GB)

The goal was to install more distributions of Linux and FreeBSD onto this computer. The problem was, all the disk space was used up. It was going to be hard to add more systems. A crazy solution had to be used, and this was where the beauty of Linux shone forth.

When the kernel for Linux starts, it doesn't have to reside on the same partition it will use for the “root” partition. The root partition for Linux is just like the root partition for UNIX systems: it contains all files and directories for the operating

system to use after the kernel gets started. That is, the kernel can reside on a different partition than the one used for its files and directories after it starts.

So, put every non-Linux operating system on the first hard drive, as well as one small Linux installation. Then, install all other distributions of Linux onto a second hard drive. The small Linux installation on the first hard drive will contain the kernels for each of the Linux installations (including its own) that reside on the second hard drive. In addition, the small installation on the first hard drive will configure LILO, so that you can choose which version of Linux you want (off of the second hard drive) when the computer starts. The trick is, LILO will be configured to grab a kernel from the small installation on the first hard drive, then switch to the second hard drive and use the appropriate partition for that kernel.

In order to install FreeBSD and a small Linux installation on the first hard drive, both the original swap and Linux partition on the first hard drive must be deleted. In place of the swap partition, install a 100MB installation of Red Hat. In place of the original Linux partition, install FreeBSD. Now the configuration should look like this:

- Primary Partition 1: Windows 98
- Primary Partition 2: Red Hat Linux 5.2 (using only 100MB of space)
- Primary Partition 3: FreeBSD
- Extended Partition 4
- Logical Partition 5: Windows NT 4.0

Listing 1 is the configuration of LILO in /etc/lilo.conf within the Linux partition.

Listing 1

Now, two steps are needed to install the rest of the Linux distributions on the second hard drive. First, all Linux distributions need to be installed without modifying how the computer boots up. The partition table on the second hard drive is set up as shown below. My second hard drive was an IDE hard drive set to be the master on the secondary IDE channel (which becomes “/dev/hdc” in Linux).

- Primary Partition 1: spare Linux partition (unused)
- Primary Partition 2: Red Hat 6.0 Linux
- Primary Partition 3: Debian Linux
- Extended Partition 4
- Logical Partition 5: Linux swap (usable by all Linux distributions)
- Logical Partition 6: Slackware Linux

- Logical Partition 7: Caldera OpenLinux
- Logical Partition 8: SuSE Linux
- Logical Partition 9: Mandrake Linux
- Logical Partition 10: shared partition among all Linux distributions located at /Shared.

After each of these Linux distributions is installed on the second hard drive, the hardest part is setting it up so that any of these operating systems or Linux distributions can be chosen at boot time. Next, do the following:

1. Turn on the computer and choose the small installation of Linux by typing in **linux** when the LILO prompt appears.
2. Copy each kernel or /boot directory from each Linux installation to a directory under /lilo.
3. Edit the /etc/lilo.conf file. Add each kernel for each Linux distribution to the file, including the option to change to the appropriate partition for the root directory after the kernel starts.
4. Execute the command **lilo**.

An example lilo.conf file is shown in Listing 2.

Listing 2

Because LILO was installed, when the computer reboots the LILO prompt will come up. From here, pressing the **TAB** key gives a list of the various options from which to choose. Typing in one of the following will start the corresponding operating system: “linux”, “linuxkernels”, “MicroSoft”, “FreeBSD”, “Debian”, “Slackware”, “Caldera”, “SUSE” or “Mandrake”. Choosing nothing would mean that the first option “linux” would be chosen. Also, if “MicroSoft” is chosen, another menu will pop up which will let you choose “Windows98” or “WindowsNT”. This other menu is the NT Boot Loader.

Conclusion

There are different ways to set up a multi-boot system using Linux and other operating systems. If you do it right, you can use one kernel for all the Linux distributions. In the method shown, the swap partition is shared by all Linux distributions.

If I had started from scratch, I would have used one 20GB IDE hard drive. As a suggestion, if you have to duplicate the configuration above, I would install Windows 98 first, then Windows NT, then FreeBSD, then the Linux distributions.

Personally, I think it is very cool and exciting how there are so many options associated with LILO. In my opinion, both LILO and the kernel are very well-designed. LILO makes it easy to do the weird stuff like I did above. Thanks to all the guys who developed LILO and the Linux kernel, and to Paul Hostetler and Phil Hunter for their help.

Resources

Mark Nielsen works at The Computer Underground, Inc., http://www.tcu-inc.com/, as a Linux geek and enjoys doing silly things and making up silly projects, because hey, computers are supposed to be fun. Mark also works at http://www.800linux.com/ as a professional consultant. During his spare time, he writes in Perl (mostly SQL and object-oriented), HTML, JavaScript, SQL for PostgreSQL and UNIX shell scripting. His long-term desires are to conquer the Ramsey numbers and help spread information about Linux software and Linux-compatible hardware to the galaxy in the best and coolest ways.

Archive Index Issue Table of Contents

Advanced search

Linux 2.4 Spotlight: ISA Plug-and-Play

Joseph Pranevich

Issue #70, February 2000

If you are tired of the complexity of configuring PnP devices for Linux, you can look forward to some relief from the 2.4 kernel release.

An amazing number of features are new or improved under (what will be) Linux 2.4. In my article last month, “Bullet Points: Linux 2.4”, I described a number of these features. The one feature which I feel will most change the face of distributions in the years to come (if only in terms of their support for legacy devices) is ISA Plug-and-Play support.

Back in the good old days (before PCI became the standard bus architecture for Intel-class PCs), buses weren't very smart in and of themselves. Plug-and-Play features were largely not present in these older machines. It was expected that if you owned the machine, you more or less understood exactly what hardware was in it and exactly where each device “lived” in the computer innerspace. Adding a new sound card, for example, was often an exercise in “put in—reboot—pull out—change jumpers—repeat”, as the vast majority of ISA cards required hardware jumpers or something similar to be configurable. This situation was not very good for consumer users of PCs, whose understanding of their machines' internals was slowly degrading to the point at which new users were hardly expected to know where to plug in the mouse.

In conference rooms of computer manufacturers around the world, meetings were held and standards debated over how this user barrier could be overcome. (At least, that's how it happened in my personal universe.) Sure, having a better bus would be nice, but that would require things to change. Some bright soul got the idea to band-aid the situation via a standard in configurable cards which could be configured by a wise BIOS or operating system. The ISAPNP “standard” was thus born.

ISAPNP filled in all those Plug-and-Play gaps we suffered from in the ISA world. No longer would we have to blindly probe for devices, as they would happily

announce themselves (more or less) to anyone who would listen. No longer would we be forced to change jumper settings, as instead we had a neat DOS utility to do that for us. But that was the real crux of the problem: PnP-compatible quickly came to mean “DOS/Windows only”, as other operating systems of the time found they could not speak the magic language of the PnP specification. Linux, too, fell into this trap; it was often advised that one should initialize his or her cards under DOS and (soft) reboot into Linux to make things work.

Fortunately, if you give the Linux community a problem, it isn't long until a solution of some kind can be found. For some, **loadlin** and a DOS partition were still the only answer, but for others there were the ISA PnP utilities which were, until recently, *the* way to configure PnP devices under Linux and elsewhere. This utility, handy as it is, is a pain for many users to figure out. It requires users to resolve resource conflicts on their own. It requires drivers to be compiled as modules so they can be loaded after the user space utility had run. Over time, the interface to this utility improved; it could even do a decent job of autoconfiguring cards. Distributions started supporting it and masking its functionality under a protective shield of pretty dialog boxes so that even the clueless could stand a chance at getting it to work. Still, it was not a perfect solution.

Linux 2.4 will, for the first time, support ISA PnP devices internally. No longer will a user space utility be required to configure cards to be used; the kernel itself can now do it. Generally, this is to be done transparently: the serial driver or the soundblaster driver will simply do a search for PnP devices in the same way they now search for and configure PCI devices. When a compatible device is found, the kernel can configure and activate it and pass the resources it uses on to the driver responsible. The kernel can even handle resource conflicts. Of course, there are probably settings and configurations which the kernel will not get right, and there will always be the option to “fall back” to the old user-space-by-hand configuration. But this, in my opinion, is a great step forward for desktop Linux.

Now, the warnings: Linux 2.4 isn't even released yet, so how can you take advantage of these remarkable new features in your machines? You can download a snapshot of the latest Linux 2.3 (developers only) kernel and compile it for your system. Will it work? Probably. Will it support your cards? Well, maybe not. If you're a programmer, there isn't a better time to get involved with Linux 2.3 development and help the mainstream kernel hackers squash the bugs to make Linux 2.4 the best Linux ever. If you're not a programmer, you can help just by downloading, compiling and installing a recent kernel and reporting the results. Linux is developed by the community, and as we approach the next stable milestone, it is the community members

who can make a difference. Next time you are sitting in front of your computer trying to tweak your isapnp.conf to work with your new modem, think of those brave souls behind their keyboards who work so hard to make Linux the best damn operating system it can be and give them a hand.

Joseph Pranevich (jpranevich@lycos.com) is an avid Linux geek, and while not working for Lycos, enjoys writing (all kinds) and working with a number of open-source projects.

Archive Index Issue Table of Contents

Advanced search

Emacs Macros and the Power-Macros Package

Jesper Pedersen

Issue #70, February 2000

Writing Emacs macros doesn't have to be hard—Mr. Pedersen helps you get “more power”.

People sometimes tend to forget that computers are tools that can make their lives much easier. One of the things computers are especially good at, and which is easy to teach them is monotonous, repetitive work. It gets even better. This kind of work also seems to be the work humans are worst at doing; that is, monotonous, repetitive work tends to be very error-prone. Emacs can eliminate repetitive work with a very useful concept called macros. Macros are basically keystrokes that Emacs types for you.

This article will describe Emacs macros and show you a number of useful examples. Furthermore, it will discuss an Emacs package I have written called power-macros, which makes it very easy to bind macros to keys and save them to a file for use in later Emacs sessions.

Defining an Emacs Macro

Defining an Emacs macro is done by pressing **CTRL-x (**. That is, press **CTRL**, hold it down and press x, release both, then press the open parenthesis. The subsequent keystrokes will be part of your macro; that is, when you ask Emacs to execute your macro, these keystrokes will be typed for you. When you are done defining the macro, press **CTRL-x)**.

When a macro has been defined, you can ask Emacs to imitate your keystrokes as often as you want simply by pressing **CTRL-x e**.

Two Cent Tip

If you need to repeat a macro several times, it might be quite annoying to have to press two keys to execute it. A solution to this is to bind a command to

“execute-last-defined-keyboard macro” to a single key press. For example, you could bind this command to **SHIFT-F1** by inserting the following code into your .emacs file:

```
(global-set-key [(shift f1)] 'call-last-kbd-macro)
```

Example: Making the Current Word Bold

Those are the macro basics. I'm fairly sure you don't yet have the feeling this would change your world much, right? Therefore, here is a small example to whet your appetite. More will follow later.

Imagine that you often want to make the current word in boldface. In HTML documents, you could do that simply by inserting and around the word. That's no big job, but if you are copy-editing a book and need to make words in boldface hundreds of times each hour, a macro to do this can save you a lot of time.

The macro is easily recorded: press **CTRL-x (**, go to the beginning of the word, type , go to the end of the word, type , **CTRL-x)**, then press **CTRL-x e** at the beginning of each word you wish to bold in the document.

There is one very important point to notice about this: you are not allowed to go to the beginning or end of the word by pressing the arrow key a number of times! Why not? Well, if you do, the macro will fail to find the border of the word if your word is of a different length than the word used when defining the macro. You must instead use the commands forward-word and backward-word. These commands are bound to **CTRL** and the arrow keys. Thus, to go to the end of a word, simply press **CTRL** and the right-arrow key.

Basically, there are two kinds of macros: those used infrequently, and those used many times in a row and then never used again. The “make word bold” example is a macro of the first kind. The description of the second kind is outside the scope of this article, but one example could be a macro that added **/* REMOVE**: to the beginning of a line, and ***/** to the end of a line. You may use such a macro a number of times in a row to comment out a whole function in C for later removal.

Making Macros More General

In some C++ programs, you will often find constructs which resemble the following:

```
for (bool
...
}
```

The only difference from one occasion to the next is the set of names: *cont*, *iterator*, *value* and the content in between the curly brackets.

If you insert the above code often, you may wish to build a macro to help you with this. Your first attempt may be to define a macro, which simply inserts:

```
for (bool =.First(); ; =.Next()) {
}
```

That is, a macro that simply leaves out all the parts that may change from time to time. This is, however, not as useful as it could be, simply because you would need to type *cont* three times and *iterator* and *value* two times each. What you really would like is to have Emacs ask you which names to use. You can do that with macros. The trick is called “recursive editing”. With recursive editing, you can tell Emacs to stop at a specific place in the macro, do some editing, and when done, tell Emacs to continue with the macro.

When you record macros, you tell Emacs to enter recursive editing by pressing **CRTL-u CTRL-x q**. Then whenever you execute the macro, Emacs will stop macro execution at that point to let you do some editing, and then the macro will continue when you press **CTRL-META-c**. (If there is no **META** key on your keyboard, it is most likely the **ALT** key instead.)

While you record the macro, Emacs will also enter recursive editing at that point. That is, the editing you do from the point you press **CTRL-u CTRL-x q** until you press **CTRL-META-c** will not be part of the macro.

We are almost ready to develop a neat and useful macro, but first let's exercise what we've learned so far with a simple example. Type the following: **CTRL-x (** Type a word ==> **CTRL-u CTRL-x q**.

Now type Hello World, and when done, continue typing the following: **CTRL-META-CTRL** <== **CTRL-x)**

The above inserted the following text into your buffer:

```
Type a word ==>Hello World<==
```

Furthermore, it also defined a macro, which inserts this text except for the words “Hello World”. Whenever you execute the just-defined macro, Emacs will pause after having inserted **Type a word ==>**, and when you press **CTRL-META-c**, it will continue with the macro, which means it will insert the text **<==**.

Can you see where we are heading? Now we have the tools to ask the user for the three names needed, so now all we need is a way to fetch the information he typed and insert it at the appropriate places.

Fetching the information could be done several ways. The simplest (that is, the one which requires the least knowledge of Emacs) would simply be to switch to a temporary buffer, let the user type in the information there, and whenever one of the words is needed, simply go to this buffer and fetch it there.

A much smarter way is to use registers. A register is a container where you may save the text of the current region for later use. To insert text into a register, mark a region and press **CTRL-x r s** and a letter (the letter indicates the register in which to save the information). Later, you may insert the contents of the register into the buffer by pressing **CTRL-x r i** and pressing the letter you typed above.

Listing 1

Listing 1 shows all the keystrokes needed to record this macro. Text in between quotes should be typed literally, and text in italics is comments, which should not be typed. It may seem like a lot of typing to obtain this, but on the other hand, when you are done, you will have a very user-friendly interface to inserting the given for-loops.

Power-Macros

Power Macros is an Emacs package, which I developed out of frustration at not being able to define a macro, bind it to a key, and have it bound for future Emacs sessions (or rather, not being able to do so easily).

To use this Emacs package, download the file from its home page at http://www.imada.sdu.dk/~blackie/emacs/. Copy the Lisp file to a location in your load path, and insert the following into your .emacs file:

```
(require 'power-macros)
(power-macros-mode)
(pm-load)
```

If you do not know what a load path is, or do not have one, create a directory called Emacs in your home directory, copy the file to this directory, and insert the following line into your .emacs file before the lines above:

```
(setq load-path (cons "~/Emacs" load-path))
```

When that is done, you may simply press **CTRL-c n** when you have defined a macro, and Emacs will ask you the following questions in the mini-buffer.

Which key to bind the macro to? First, Emacs must know to which key the macro should be bound. When you are finished answering these questions, the macro will be available simply by pressing this key. By binding to different keys, you can have several macros defined at the same time.

How should the macro be accessible? With power-macros, you may make the macro accessible in one of two ways:

1. Global: it is accessible in every buffer.
2. A major-mode-specific macro: the macro is accessible only in buffers with a given major mode.

As an example of a mode-specific macro, think about the for-loop macro from the example above. This macro is useful only when writing C++ programs. Furthermore, you may need a similar macro using Java syntax for programming Java. With power-macros, you may bind both the macro for C++-mode and the macro for Java-mode to the same key (say **CTRL-m-f**); then the correct one will be used for the given mode.

Which file should it be saved to? By default, Emacs saves the macros defined with power-macro to the file named ~/.power-macros. If that is okay for the macro you are defining, simply press **ENTER** at this question. If you do not want to save the given macro to a file for future Emacs sessions, remove the suggested text (i.e., answer the question with an empty string). Also, you can name another file. The section below has a description of when doing this can be of special interest.

What is its description? Finally, you have to write a description for the macro just defined. This will make it much easier for you to identify it later, when you have forgotten which key it is bound to, or when you are searching for a key to bind a new macro.

As part of binding the macro to a key, Emacs will also check if the given binding will override an existing binding. If this is the case, it will warn you and ask for confirmation to continue the definition.

Local Macros

Some time ago, I was going to give a speech on Emacs. I have previously made that a number of times, so I hadn't done any special preparation for this specific speech. While I was traveling to the event by train, I decided to go through my presentation. I was terrified to see that the presentation program suddenly didn't work on my machine. What should I do? The answer was obvious: why not make the presentation using Emacs? Fortunately, the input to the other presentation program was ASCII, and the only construct I used in the

presentation was enumerated lists, so it was very easy to rewrite the presentation so it looked good in an Emacs buffer (with a slightly enlarged font).

Now there was only one problem: how could I easily go forward/backward one presentation page? The answer was to create two macros: one going forward one page, and another going backward one page.

Going forward one page was done in the following way:

- Search for a line starting with a number of equal signs, namely the second line of each presentation page (just below the title of the page).
- Press **CTRL-1 CTRL-l** (that is, control-(number)one control-(letter)l). This would locate this line as the second line of the screen, and consequently, the title of the page would be the first one.
- Go to the beginning of the next line. This was necessary so that the subsequent search would not find the current page.

The two macros just defined are useful only for the given file, and later for all files which contain a presentation made for viewing with Emacs. Therefore, it would be a bit annoying to have these macros defined and bound to keys all the time, especially given that there might be several months before my next Emacs presentation.

The two macros were therefore saved to a separate file, and whenever needed, I can simply load them. Loading a power-macro is done by using the function **pm-load**. Thus, I could load the macros by pressing **META-x**, typing **pm-load**, pressing **ENTER**, and typing the name of the file to load. Loading the macros for the presentation could be done even more automatically by inserting the following lines as the last lines of the file:

```
Local Variables:
eval: (pm-load "presentation.macro")
End:
```

Here, it is assumed that the name of the file containing the macros is called presentation.macro. Now, Emacs automatically loads the presentation macros whenever the file is opened.

Managing Power-Macros

When you have defined a number of macros, you might want to perform various functions to manage your macros. This is done by pressing **CTRL-c m**. It will bring up a buffer like:

```
the one shown in Figure 1.
```

What you see in this buffer is your power-macros, each separated with a line of dashes. Many keys have special meanings in this buffer (just like the keys have special meanings in the buffer-managing buffer or in the dired buffer).

Pressing the **ENTER** key on top of one of the fields allows you to edit the given field. Editing a field means either to change its content or copy the macro to a new one with the given field changed. You specify whichever of these meanings you intend, when you have pressed **ENTER** on the field.

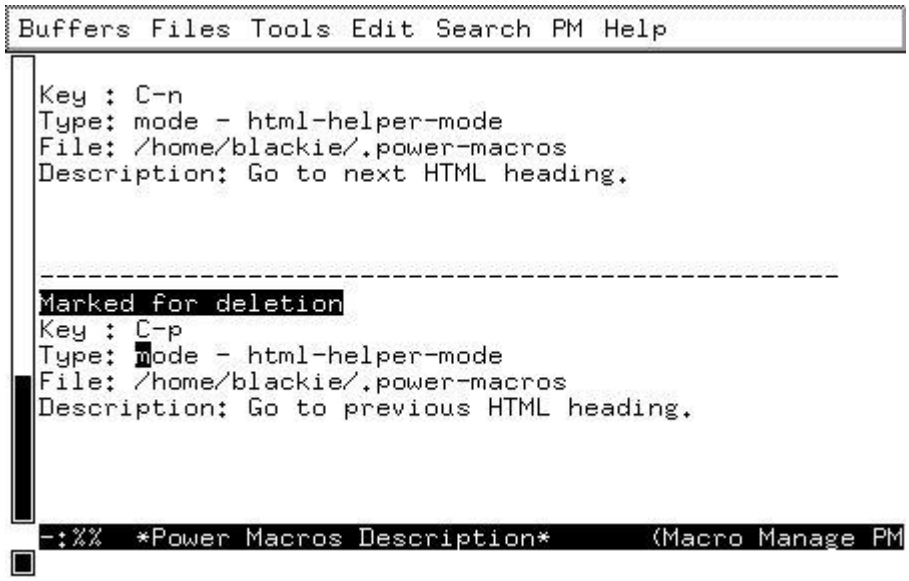


Figure 1. Emacs Power-Macro Buffer

Deletion of macros is done in two steps. First, you mark the macros you want to delete, and next you tell Emacs to actually delete them. If you know either the buffer-managing buffer or dired-mode, you will be familiar with this two-step process.

If you are now ready to learn more about Emacs, visit my home page at the URL mentioned earlier.

This article was first published in Issue 47 of LinuxGazette.com, an on-line e-zine formerly published by Linux Journal.

Jesper Pedersen Jesper Pedersen is the author of the book *Teach Yourself Emacs* in 24 Hours, the program “The Dotfile Generator”, the Emacs package

“Power Macros”, and is chairman for the Linux User Group on Funen in Denmark.

Archive Index Issue Table of Contents

Advanced search

Tasty KDE Desktop Themes

Marcel Gagné

Issue #70, February 2000

Bonjour, mes amis! Welcome back to my restaurant, “Chez Marcel”. Please, sit down. Francois has prepared your usual table and will be along with your wine tout de suite.

This month's menu features “Linux on the Desktop”. Your desktop, like any well-dressed table, should reflect something of your personality. Do you use gold-edged silver cutlery or stainless steel? Is candlelight a must for your dinners? Are your napkins linen, cotton or paper? Most important, of course, is to simply enjoy your meal. The well-dressed table, like the well-dressed Linux desktop, simply *enhances* the experience.

Okay, I admit it; I am stretching the metaphor, but what's a meta for if not for stretching? François! Better bring me a glass, as well. As for you, my welcome guests, let me show you how you, too, can develop a desktop theme that reflects your personality.

On your K Desktop (I am running 1.1.2), you will find the theme manager under the Settings/Desktop menu. Using this tool, you can easily import new themes (see Resources for links), delete old themes, create and modify them. Despite great strides in the development of the theme manager, creating and modifying a theme is still largely a command-line job. The theme manager is used primarily to manage existing themes, a job it does very well.

Using the theme manager, I created the skeleton for a “Cooking” theme by clicking on “Create”. Imagine that, non! This created a default structure for my theme in the directory /home/marcel/.kde/share/apps/kthememgr/Work/Cooking. As I mentioned, this is primarily a command-line (or editor) job. The theme manager allowed me to save my settings and apply changes on the fly as I worked out the details of my theme. A note of caution, though: when you click “Save”, your current working directory will be erased and rebuilt. Before clicking on “save”, you need to go back a directory (**cd ..**) before continuing. Once you

have saved your theme and applied it, you can safely go back into the Work/Theme_name directory.

There are several components in a theme. They include icons, colors and fonts, borders and so on. Each of these is defined in a .themerc file for your theme. In my case, I have a Cooking.themerc file. The default built by the theme manager includes only the title and author information, in the **[General]** section of the file. Here's what mine looked like immediately after the Theme Manager created it:

```
# KDE Config File
[General]
Author=Marcel Gagné
Homepage=http://www.salmar.com
Version=0.1
description=Chez Marcel chef Marcel Gagné brings you his Cooking with Linux theme
Email=mggagne@salmar.com
```

This is only the beginning of the template; it requires several more sections, each with its own details: Display, Colors, File Manager and Window Border, to name a few.

If you feel so inclined and have a lot of wine to drink, you can write every section of your .themerc manually. My method involves copying the system .themerc template to the one created by the theme manager. You can find the system template under your KDE share directory. On an RPM system like mine, it is called /usr/share/apps/kthememgr/Themes/Template.themerc. I use the **cat** command to do the copying with the following single line (ignore wrapping):

```
cat /usr/share/apps/kthememgr/Themes/
Template.themerc >> /home/marcel/.kde/share/apps/kthememgr/
Work/Cooking/Cooking.themerc
```

This gives me a complete template to work with. Using **vi** to edit the file, I removed the template's **[General]** section and left my own in place. For my *Cooking with Linux* theme, I used only the Display, Window Border, Window Titlebar and Window Button Layout sections.

Getting Creative

So, you want to create a theme, eh? Before we begin, let me give you some quick technical notes. All of my graphics were either created or modified using the GIMP. You could use another package to do this, but why would you? Other than the desktop wallpaper, which is a .jpg file, all images are in .xpm format and indexed before saving.

In creating this recipe, I started with a background I knew had to be food-related. A few years ago, at a very strange New Year's party in our home, a couple of friends offered to bring a veggie tray. This veggie tray was a joy to

behold, with hard-boiled egg islanders, carrot palm trees and mushroom huts. We were so impressed, we took a picture of it. This was my starting point. With a little modification in the GIMP, I added Tux, gave him a glass of wine and sat him under a celery palm. Here are the specifics from the **[Display]** section.

```
CommonDesktop=true
Wallpaper0=aveggietray.jpg
WallpaperMode0=Centred
```

Next, I created buttons. The way I see it, when you minimize a window, you are essentially offering it a glass of wine and giving it a rest. What better icon than a glass of wine, non? Then came maximizing a window to the desktop—sort of like “spooning” it out onto the desktop, n'est-ce pas? Et voil<\#224>, a spoon! The sticky button was obvious as well. You stab the window with a fork and anchor it, pointing down to stick and up to unstick (also the default position). Finally, I needed a close button, the great *coup de gr<\#226>ce* that dispatches a window. Ah, oui! A knife of the finest steel.

Figure 1. Icons

All these icons form the **[Window Titlebar]** section. They can be up to 20 pixels in height and width. Before finishing here, I created a pixmap for the titlebar, a *salad* look and feel. The difference between the active titlebar and the inactive one is brightness. Finally, I right-aligned the text. Here is the resulting section:

```
[Window Titlebar]
CloseButton=knife.xpm
MaximizeButton=spoon-up.xpm
MaximizeDownButton=spoon-down.xpm
MenuButton=
MinimizeButton=wine.xpm
StickyButton=fork.xpm
StickyDownButton=forkdown.xpm
TitlebarPixmapActive=saladbar.xpm
TitlebarPixmapInactive=inactivesaladbar.xpm
PixmapUnderTitleText=yes
TitleFrameShaded=no
TitleAlignment=right
```

Buttons can be positioned anywhere you wish. There are six possible button positions: three on the left and three on the right. Since I tend to click before I think, I decided to stay with the default positions. However, if you are feeling dangerous, you could switch the Iconify and Close positions. Ah, the romance of living on the edge! Here is the section in question:

```
[Window Button Layout]
ButtonA=Menu
ButtonB=
ButtonC=
ButtonD=Close
ButtonE=Maximize
ButtonF=Iconify
```

Window borders are a strange thing to me—I both *love* and *hate* them. On one hand, they decorate; on the other, they take up valuable desktop real estate on my notebook's panel display. For those times I feel wild and foolish, I decided to throw in a flash of color to my borders. For corner icons, I brought Tux and his wine glass back. The resulting entry from my ".themerc" follows.

```
[Window Border]
shapePixmapTop=horizborder.xpm
shapePixmapBottom=horizborder.xpm
shapePixmapLeft=vertborder.xpm
shapePixmapRight=vertborder.xpm
shapePixmapTopLeft=tuxwine.xpm
shapePixmapTopRight=tuxwine.xpm
shapePixmapBottomLeft=tuxwine.xpm
shapePixmapBottomRight=tuxwine.xpm
```

The border sections, like the window titlebar, need not be terribly long. KDE will automatically tile your image to fill any window. There is a catch, though, and let me tell you that I ached over this one for quite some time. You need to make sure the borders can line up with the corner. If you create a 20-by-20-pixel corner icon, make sure you line it up with a 20-pixel width border.

You know, as the Perl motto goes, "there is more than one way to do it". Clicking "Save" each time you make changes in the theme manager is just one way to do it. If the constant refreshing of your Work directory gets tiring after awhile, resort to your old friend, the command line, and create your own lovely tar bundle. Gzipped, bien sûr. To simplify the image here, I will step back a directory and back up my bundle from there. Remember, I was in the Work/Cooking directory.

```
cd ..
tar -czvf Cooking.tar.gz Cooking
```

This will create my Cooking.tar.gz theme bundle. The catch, if you do it this way, is that you then need to copy the bundle back to your ~/.kde/share/apps/kthememgr/Themes directory.

In any case, when you are satisfied with your work, you can upload it to your favorite theme site for all the world to savor. It can then be installed with the theme manager using the **Add** function. For your viewing pleasure, Figure 2 is a picture of the resulting theme. Should you be so inspired, you may download a copy from the *Linux Journal* web site or from my own. Surely, fame and fortune are then only a heartbeat away, non?

Figure 2. Cooking Theme

Well, mes amis, it is once again closing time. Do come back soon. You are always welcome at *Chez Marcel*. Bon Appétit!

Resources

Marcel Gagné lives in Mississauga, Ontario. In real life, he is president of Salmar Consulting Inc., a systems integration and network consulting firm. He is also a pilot, writes science fiction and fantasy and edits TransVersions, a science fiction, fantasy and horror magazine. He loves Linux and all flavors of UNIX and will even admit it in public. He can be reached via e-mail at mggagne@salmar.com. You can discover lots of other things from his web site at http://www.salmar.com/.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

More About Searching

Reuven M. Lerner

Issue #70, February 2000

Learning to be more efficient in searching by pre-indexing files.

Last month, we looked at a simple search engine for web sites. The program was little more than a CGI program strapped to the **File::Find** Perl module: each time a user would enter a search term in the HTML form, the search program would dutifully open and examine each of the files under the web hierarchy.

While this sort of search engine works, it is exceedingly inefficient. A site containing several dozen files will not feel too much of a hit when its documents are searched repeatedly by a CGI program, but a site with hundreds or thousands of files, attracting thousands of hits per day, will watch its server's load average skyrocket without very much difficulty.

This month, we will explore ways of making a search engine more efficient. In the end, we will have a search engine which might not work as efficiently as other software, but is simple to install and use. Most importantly, we will get a chance to explore an interesting type of software with inner workings usually invisible to us.

The Secret: Pre-indexing

Searching through files sequentially, trying to find matches for a user's input, is an inherently inefficient business. Each file must be opened, read, scanned and closed, which takes time. Perl programs tend to consume a fair amount of memory, so the slow execution speed means more copies of the CGI program will be running at once. This in turn increases the risk that the web server will have to use virtual memory, rather than physical RAM. Slow web servers make for unhappy users, and often convince users not to return at all.

To solve this problem, we must reduce or remove the need for the search program to read through files. If the CGI search program did not have to open each individual file, things would speed up quite a bit.

A tried-and-true solution is to divide it up between two programs. Once or twice each day, an indexing program traverses the web-document tree, reading through each document and analyzing its word use. This program runs behind the scenes without user intervention or knowledge. Rather than sending its results to a user, the indexer dumps all information it has about word frequency and usage and places it in a file on disk.

This means the search program the user invokes via CGI does not actually have to search. Instead, the search program merely opens the index file, finds those files where the user's search term appears the greatest number of times, and displays that list in the user's browser.

Indexing a page is not difficult in Perl, because of its rich library of regular expressions. The **m//** operator normally matches the regular expression between its delimiters. When invoked with the **/g** modifier and when operating in list context, it returns all matches it can find. Thus, in the expression

```
my $found = join ' ',
  ("encyclopedia" =~ m/[aeiou]/g);
print "$found\n";
```

the first statement finds all vowels in “encyclopedia” and returns them as a list to the caller. In this case, the caller is Perl's **join** operator, which pushes the elements together, separated by spaces. Executing the two lines of code above displays the following on the user's screen:

```
e o a e i a
```

Using the built-in character class for non-whitespace characters, **\S**, we can apply the same algorithm in order to extract words from a text string. For example:

```
my $sentence = "The rain in Spain falls mainly\n\n on the plain.";
my $found = join '|', ($sentence =~ m/\b\S+\b/g);
print "$found\n";
```

The code above prints the following results:

```
The|rain|in|Spain|falls|mainly|on|the|plain
```

Notice how using **\b** (which matches a word boundary) means our program need not worry about newline characters, extra spaces or punctuation.

Indexers have to consider whether to keep case relevant. My personal preference is to ignore case, since users do not necessarily remember, and it

also removes an obstacle to finding desired text. We can thus turn all of the words into lowercase letters:

```
my $sentence = "The rain in Spain falls mainly\n\n on the plain.";
my $found = join '|', map {lc $_}
   ($sentence =~
   m/\b\S+\b/g);
print "$found\n";
```

Storing the Index

To store index information, we will use a hash, **%IGNORE**. The keys will be words we wish to ignore when indexing. Any non-zero value will indicate this word should be ignored when indexing:

```
%IGNORE = ("the" => 1, "in" => 1, "on" => 1);
my $sentence = "The rain in Spain falls mainly\n\n on the plain.";
my $found = join '|',
   grep {!$IGNORE{$_}}
   map {lc $_} ($sentence =~ m/\b\S+\b/g);
print "$found\n";
```

Notice how we can stack different items together: **m//** returns a list, which is passed to **map**, which returns a list, which is fed to **grep**, which is in turn fed to **join**, and which is in turn assigned to **$found**.

Finally, we will index the words by creating a hash (**%index**) in which the collected words are the keys. The value will be a hash reference, where the key is the name of the current file, and the value is the number of times this word appears in the file. In other words,

```
$index{spain}->{foo.html} = 5;
```

means the word “spain” appears in foo.html five times. Here is some code that performs the indexing in this way:

```
%IGNORE = ("the" => 1, "in" => 1, "on" => 1);
my $sentence = "The rain in Spain falls mainly\n\n on the plain.";
my @found =
  grep {!$IGNORE{$_}} map {lc $_} ($sentence =~
m/\b\S+\b/g);
    foreach my $word (@found)
    {
        $index{$word}->{$filename}++;
    }
```

Traversing the Web Hierarchy

Now that we know how to index the words from a string, we will use File::Find to perform this task on the entire web hierarchy. File::Find comes with the standard Perl distribution, and exports a subroutine named **find**. This subroutine takes a subroutine reference and a list of directories as arguments. It invokes the named subroutine (known as a “callback”) on every file it finds in

the named directories. It is the subroutine's responsibility to open the file, retrieve its contents, and index it as necessary.

Listing 3 (see Resources) contains a simple program (index-files.pl) that traverses the web hierarchy, searches through the contents, and stores word frequency in the **%index** hash. The callback subroutine ignores files in any directory named .noindex, as well as any subdirectories of such directories. This is accomplished with another hash named **%ignore_directory**. It also ignores non-HTML files and removes HTML tags from the contents of each file before indexing the words it contains. Rather than removing the HTML tags altogether, index-files.pl turns them into whitespace. This means that two words separated by an HTML tag (but no whitespace) will not be jammed together, but will remain separate words.

To avoid problems with HTML entities such as **>**, **&** and other items needed for symbols and accented Latin characters, we use the **HTML::Entities** module. This module exports a **decode_entities** subroutine that turns the entities back into their regular characters. By running this and then removing the HTML tags, we are able to index a clean document containing only text.

While hashes are convenient and fast, they tend to consume a lot of memory. A hash that points to another hash consumes even more memory, and storing each individual word as a key in the primary hash means the memory usage will skyrocket even more. Running index-file.pl is thus not for the faint of heart or for computers without a significant amount of RAM. This particular version of index-file.pl caused my Linux system (with 72MB of RAM and another 64MB of swap) to run out of memory, particularly when piping the output through **less**.

Writing %index to Disk

While index-files.pl performs an admirable job of indexing the files in a web hierarchy, it does not write the information to disk. Once index-files.pl exits, all indexing information is lost forever.

The solution is clearly to write the index to disk. But how? The most obvious answer is to use a DBM file. DBM files are most easily described as an on-disk version of hashes. Each entry in a DBM file has a key and a value, both of which may be any character string. As with hashes, DBM files are optimized for speed, making them much faster to use than straight text files. There are several different kinds of DBM files out there, ranging from plain old DBM to the GNU Project's GDBM to Berkeley DB, which has gained quite a bit of popularity in the last few years partly as a result of its integration into Sendmail.

Perl supports many types of DBM files through its “tie” interface, each of which has its own object module. Tying is a means of making a variable magical, attaching additional behavior every time a value is stored or retrieved. Thus, tying a hash to a DBM class means that every time a value is stored to the hash, the value is written to a corresponding DBM file on disk. By the same token, every value retrieval reads the information from disk.

There is one problem here, though: DBM files can store text strings, but nothing more complicated than that. Because each value of **%index** is a reference to another hash, the normal DBM classes will not work correctly. Attempting to store references in a normal DBM file will actually store their printed representation, which is not at all useful.

The solution is to use the **MLDBM** class, which is designed precisely for this purpose. MLDBM works in conjunction with another DBM class to store references in a format that can be retrieved later.

To use MLDBM, import it at the top of a program with **use**, specifying the type of DBM file to use with it:

```
use MLDBM qw(DB_File);
```

In this particular case, we will use **DB_File**, which uses the Berkeley DB module that comes with Perl. (Most Linux systems come with Berkeley DB, as well.) It is also possible to specify the underlying method used to store the references; we will use the default, which takes advantage of the **Data::Dumper** module available on CPAN. Other options include **FreezeThaw** and **Storable**, which perform similar tasks in different ways.

Our hash can then be tied to a DBM file with the **tie** command:

```
# In what file should the index be stored?
my $index_file = "/tmp/lj.db";
# Create the word index hash
my %data;
tie %data, "MLDBM", $index_file, O_RDWR|O_CREAT
or die "Error tying %data: $! ";
```

From this point onward, any value stored to **%index** will be stored in the file /tmp/lj.db. Any value retrieved from **%index** will actually be retrieved from /tmp/lj.db. Storing index data in /tmp is a mistake on a production web server, since the file can be deleted by the system.

While MLDBM attempts to function as transparently as possible, it will sometimes cause confusing errors. For example, there is normally no problem with the following Perl code:

```
$data{$word}->{$url}++;
```

As we saw earlier, this will increment the counter for a particular word in a particular file. When **%data** is tied to MLDBM, the above code will no longer work. Instead, the assignment must be performed in two stages, retrieving the hash reference, assigning to it, and placing it back inside of **%data**:

```
my $hashref = $data{$word};
$hashref->{$url}++;
$data{$word} = $hashref;
```

The index should be regenerated on a regular basis to ensure it is as fresh as possible. Consider using **cron**, which comes with all Linux and UNIX systems, to run index-files.pl every day at 4 AM or at another convenient time when people are unlikely to be using the computer.

While our version of index-files.pl does not support such functionality, it would not be difficult to add a hash key indicating when the file system was last indexed. index-files.pl could then be changed to index only those files that were created or modified after a particular date.

Reading the Index

Now that we have created an index, we will need a program to retrieve results from it. Indeed, that's just the beginning—if we want, we can write multiple programs that read through the DBM file, searching in a variety of ways.

Our first and simplest search will accept input from an HTML form. The form will allow users to specify a single word to search for. Unfortunately, the way we have indexed the files makes it easy to retrieve single words, but impossible to retrieve phrases. A different data structure would make it easier to perform such searches, but would undoubtedly increase the size of the index.

Listing 1

Listing 1 is a simple HTML form we can use for searching. This form expects to submit its results to a CGI program called dbsearch.pl, which is in Listing 4 (see Resources). dbsearch.pl opens the DBM file—again, using MLDBM and DB_File—and retrieves the keys from the hash associated with the search term. By sorting the keys (which contain URLs) by their values (which are integers indicating how many times a word appeared in a file), it's easy to create a simple frequency chart. In this particular case, dbsearch.pl displays all of the results, but it would not be difficult to display only the top 20.

Boolean Search Terms

Searching for a single term might be easy to implement, but the best search engines allow for AND and OR searches. Listing 2 is an HTML form that differs

from the previous version by adding a radio button. If the radio button is set to the default OR position, any one of the search terms may appear in the document. An AND search requires that both or all of the search terms appear in the document in order for it to match.

Listing 2

A version of better-dbsearch.pl, which allows for AND and OR searches, is in Listing 5 (see Resources).

If the user chooses OR, dbsearch.pl will have to find the highest total for a combination of hashes, rather than just one. A simple example would be the hashes **%odds** and **%evens**, as follows:

```
%odds =  (one => 1, three => 3, five => 5);
%evens = (two => 2, four => 4, six => 6);
```

We can turn this into a simple hash by mixing them together:

```
%numbers = (%odds, %evens);
```

This technique will not work in our case, because it is quite possible that both words will appear in the same file. If that happens, there will be duplicate keys —unlike **%odds** and **%evens**, where no key appears in both hashes. For our OR search to work, we will need to combine the values in a master hash:

```
foreach my $word (@terms)
{
my $frequency = $data{$word};
foreach my $filename (keys %$frequency)
{
$full_results{$filename} += $frequency->{$filename};
}
}
```

The above code is not very different from its predecessor in dbsearch.pl. The main change is that it introduces a new hash, **%full_results**, where the keys are file names and the values reflect the total scores for each of the search terms. This algorithm means a file containing one copy of both search terms will be ranked the same as one in which the same search term appears twice.

To handle AND searches, we must keep track of how many search terms appear in each file. If a file contains the same number of search terms as are in **@terms**, the file is acceptable. If it contains fewer matches than **@terms**, the file should be dropped from consideration.

We can implement this by using the **%match_counter** hash, where the keys are file names and the values are integers representing the number of search terms contained in a file. By adding a single line to the foreach loop, we can keep track of how many search terms appear in each file:

```
$match_counter{$filename}++ if ($boolean eq "and");
```

Once we have exited from the foreach loop, but before we announce the results to the user, we prune file names from **%full_results**. Because the keys for **%full_results** are the same as for **%match_counter**, the code is relatively simple:

```
foreach my $filename (keys %full_results)
{
delete $full_results{$filename}
unless ($match_counter{$filename} == @terms);
}
```

Notice how we use **delete** to remove the hash element. Using **undef** on **$full_results{$filename}** would make the value undefined, but would leave the hash key unaffected. **delete** removes the key and its corresponding value from the hash completely.

Also notice how we can get the size of **@terms** simply by naming the array itself. The **==** forces scalar context on both of its operands, so there is no need to use the **scalar** keyword before **@terms**.

Downsides to Pre-Indexing

Our pre-indexing solution is, indeed, faster than the search program we examined last month. I did not perform any serious benchmarks on this set of programs, but the difference was readily apparent to the naked eye. Not only that, but our pre-indexed implementation made it easy to rank files according to the number of matches.

However, pre-indexing is not a panacea. For starters, it requires a good deal of disk space, weighing in at 2.6MB for an index of fewer than 400 files. However, given the rapidly dropping price of disk space, even a 10MB index should not prove to be much of a deterrent for most systems.

A bigger problem is the lack of flexibility that pre-indexing forces on a search system. For example, our index programs all used Perl's **lc** operator to force all of the letters to lowercase. Now that we have removed any trace of case from our index, how can we offer a case-sensitive search? The only real answer is to build the DBM file in a slightly different way, perhaps by storing an additional **literal** element in the hash for each file. Then we would know that **abc** appeared five times in a particular file, but that two of these were **ABC** and the remaining three were **abc**.

Pre-indexing also means we can no longer offer a phrase search, in addition to AND and OR searches. There are ways to solve this problem; the easiest might be to store “next word” information in the database hash. Then we could

search for the first word of a phrase and use the “next word” information to see if the other words were found, one by one.

Finally, pre-indexing always means the index will lag behind other content on a web site. If the index is generated once every 24 hours, it might take up to one day for a new document to be read into the index. One solution is to run the indexer more often, such as every three or six hours.

Conclusion

Searching is an essential part of every good web site. It means users can find what interests them quickly, and can perform analyses that the site's administrators never expected. But as we saw last month, a straight search through a site's files can take a long time to execute. Pre-indexing is the standard solution to this problem, trading additional disk space and a slightly out-of-date index in exchange for faster execution speed. Understanding the trade-offs involved in writing a search engine makes it easier to evaluate free and commercial offerings, and thus make your site a more enjoyable place for users to visit.

Resources

Reuven M. Lerner is an Internet and Web consultant living in Haifa, Israel, who has been using the Web since early 1993. His book Core Perl will be published by Prentice-Hall in the spring. Reuven can be reached at reuven@lerner.co.il. The ATF home page, including archives and discussion forums, is at http://www.lerner.co.il/atf/.

Archive Index Issue Table of Contents

Advanced search

Focus on Software

David A. Bandel

Issue #70, February 2000

WebCal, tkballistic, gbase and more.

A few years ago, software licenses were simpler. We had the GPL (General Public License), now in version 2.0 for those suspicious of 1.0 releases. We had the BSD license. And there was a “free but restricted” license, which stipulated that if you used the program to make money, the developers wanted some too (sounds fair to me). Last week, I was looking through some of the licenses we have now; some I can't understand at all. There's the Open Source license, the Sun Open Source license, the Corel license, several variations on the free but restricted license, an Artistic license and a few others I don't recall. For those of us who can't afford to keep lawyers around, we'll just need to read and comply with these licenses the best we can and hope it's sufficient to keep us out of court.

WebCal: bulldog.tzo.org/webcal/webcal.html

WebCal is a nice, easy-to-use web calendaring tool. Users can create their own private calendar, allow others to read it and perhaps a select few to maintain it. Calendars can be created for conference rooms, for scheduling, etc. It is very well-done and doesn't require anything extraordinary. The calendar will even e-mail you a reminder if you need one. It requires Apache or another web server that can handle permission files à la **htpasswd**, and a frames-capable browser.

tkballistic: http://members.xoom.com/joshua_weage/ballistic/

Gun buffs out there can have a good time with **tkballistic**. If you know the bullet diameter in inches, its weight in grains and the muzzle velocity in ft/sec, you can find out almost anything you want to know about the flight path of the projectile. If you also have the wind velocity in ft/sec and the angle in degrees, you can save a trajectory table. This application will also calculate ballistic coefficient and maximum point blank range. It requires Python and tkinter.

gbase: http://www.hibernaculum.demon.co.uk/

gbase is just what I wanted for Christmas! Working on networks, I find there's always a need to convert between decimal, hexadecimal, octal and binary. This little utility does it extremely well. The four text boxes are placed one below the other. Enter a number in the box of the type you know, and as you type, the numbers in the other boxes will increment. I doubt this application will leave the workspace of my laptop any time soon. It's simple and efficient. It requires libgtk, libgdk, libgmodule, libglib, libdl, libXext, libX11, libm and glibc.

xipdump: http://www.epita.fr/~lse/xipdump/

For those familiar with **tcpdump**, **xipdump** is similar, but makes packets readable using a box containing the packet information, rather than the stream-type output of tcpdump. The author chose key bindings that remind me of Sun OpenWin bindings; not very intuitive, but easy to get used to. The program allows you to change and reinsert packets, etc. Since most work, including reading the packet stream, is performed on raw sockets, only root can run this, but you wouldn't want non-privileged users playing with this program anyway. It requires libnet, libXaw, libXmu, libXt, libXext, libX11, glibc, libSM, libICE and libpcap.

MyAddressPHP: http://rob.current.nu/MyAddressPHP/

MyAddressPHP is notable for the ease with which it can be installed. The author put some effort into making the installation as painless as possible for this type of package. While he does assume you've configured your web server to handle PHP documents, the rest (setting up the software and database) is easy if you follow the instructions carefully. It is what it says it is: an address book, but one that allows you to add pictures. This is handy as a database for any organization that uses picture badges or wants to maintain individual photos. Some features are not yet implemented, but the shells are in place. It requires MySQL, PHP3, a web server and a web browser.

mygde: http://ringil.stampede.org/mygde/

mygde is used for accessing a MySQL server and is patterned after xMySQL. However, this package uses gtk rather than the xforms library. It does similar operations, allowing the creation of and access to tables. Queries can be created using point and click rather than entering SQL commands, although not all SQL commands are available. For example, if you need to perform complex outer joins, this tool won't help yet, but adding those query types shouldn't be difficult. It requires libm, libgtk, libgdk, libgmodule, libglib, libdl, libXext, libX11, glibc and libnsl.

FreeS/WAN: http://www.xs4all.nl/~freeswan/

At last, the FreeS/WAN package is ready for the 2.2.*x* and above kernels. This package allows you to create encrypted tunnels between Internet-connected systems. You could use it to create encrypted tunnels between any two hosts, but there's little sense in doing so on a “trusted” network. This is primarily aimed at providing encrypted tunnels between two locations connected only via the Internet. What makes this package so good, or at least better than SSH? The license—this one comes with no strings. That's not true of SSH v1 and even less of SSH v2. The FreeS/WAN package compiles into the kernel in a manner simple enough for relatively inexperienced administrators, yet doesn't prevent experienced ones from adding other patches and custom configurations. It requires working kernel sources, libraries and tools to compile the kernel.

gtktetris: http://www.sudac.org/~napolium/linux/

What can be said about yet another Tetris clone, except perhaps that **gtktetris** compiles easily on any system with the gtk libraries? The version I looked at works well, but I was unable to find a “drop” key to just let the pieces fall. This is one of my wife's favorite games, and she enjoys nice-looking color graphics, which this program has. It requires libgtk, libgdk, libgmodule, libglib, libdl, libXext, libX11, libm and glibc.

David A. Bandel (dbandel@pananix.com) is a Linux/UNIX consultant currently living in the Republic of Panama. He is co-author of Que Special Edition: Using Caldera OpenLinux, and he plans to spend more time writing about Linux while relaxing and enjoying life in the “Crossroads of the World”.

Archive Index Issue Table of Contents

Advanced search

The Open Road Ahead

Stan Kelly-Bootle

Issue #70, February 2000

Over the years, we've seen many computer publications falter or fade as they sought faithful readers and fickle advertisers in the face of an ever-volatile industry.

It's a delight to debut column-wise in one of my favorite magazines, a journal that has remained finely focused since March 1994. Over the years, we've seen many computer publications falter or fade as they sought faithful readers and fickle advertisers in the face of an ever-volatile industry. There has always been a broad spectrum of editorial targets with “content-addressed” titles, ranging from the cosmic general (*Wired*) to the minutely specialized (*printf() Daily*). By “content-addressed”, I mean the magazine title aims both to capture (figuratively) the themes to be addressed and to capture (factually) a viable market segment. The old signifier-signified problem has oft raised its intractable transient-fashion semantic head.

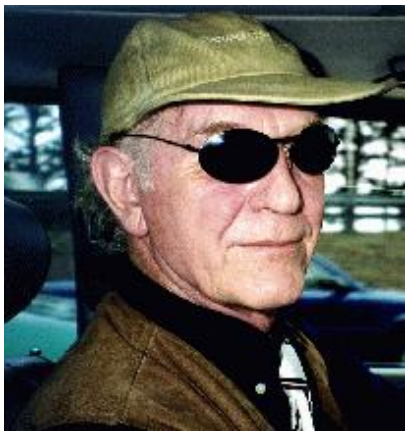
Consider the demises of *Datamation* (why does this sound archaic, in spite of recent data-centric resurrections?) and *BYTE* (the once “in” term, with a matching chain of Byte shops, encapsulating the 1980s thrill of affordable, personal computing, but now worth a few mundane bits). Some publications have survived with brash, broad names such as *Computer Weekly*, a title that was quite content-specific in the 1960s but nowadays reveals only its frequency. Others have gone for “insider” titles. Thus, unless you already knew, *Dr. Dobb's* might at first glance appear to be yet another paternal self-health advisory. (Some Dobb's fans may say “indeed”.) Mostly, though, it's an upfront WYSIWYG world (What You Subscribe to Is What You Get). If you pay through the nose for *MSDN [Microsoft Developer Network] News*, that's what you'll get: the current designation, security vulnerabilities, APIs-du-jour, and ETAs for NT 5.0. (To be fair, Dr GUI does offer “work-arounds” to the baffled with non-corporate aplomb.)

There's no shortage of drop-dead titles that strutted bravely but briefly and are heard no more. I wrote evangelically, no one more, for *OS/2 Magazine* until it succumbed-- not for want of reader devotion (they screamed “Stan, say it isn't so...”) but for IBM's own lack of commitment and the natural ensuing reluctance of the advertisers.

Then we have the fascinating name-change phenomenon. Reacting to those restless, highly paid “demographers” who filter your blatant reader-response fibs through a statistically stunted spreadsheet, the publisher-suits are often convinced that the titular temperature needs a tweak. (Consultant Axiom #1: It's difficult, alas, for the client to ignore expensive advice.) Whether the content should change to reflect the “hotter” title is a tricky problem. Too dramatic a shift in both is rather like a new magazine launch and everything that entails. Instructive exemplars abound: *Computer Language* moved fairly gracefully to the less “technical” *Software Development* with an appropriate content-drift toward the bland. (Its “Peopleware” articles are notorious for their vacuous “Cook Until Done” recipes: “Create a well-formed plan before you start coding, and don't forget to select the best-possible team for your project...”) Similarly, following the trend set by Clemens Szyperski's *Component Software—Beyond Object-Oriented Programming* (Addison-Wesley/ACM Press, 1998) [Reference 1], SIGS's *Object Magazine* traveled “beyond” to become *Component Strategies*. This advance was cut short by another popular name-changing influence: the publishing trade's agglutinative propensity. Following one of those hard-to-keep-up-with takeovers (not to be confused with those equally baffling “strategic alliances”), *Component Strategies* has been newly “folded” into *Application Development Trends*. Presumably, this title allows the tracking of SE (Software Engineering) fashions beyond mere components. An interesting counter-example, where the content has been fashionably “enhanced” without a title change, is *C/C++ Users Journal* that now carries a regular column and even detached supplements on...shock-horror...drum-roll...Java! Recall that the earlier post-incremental title update from *C Users Journal* still triggers irate reader mail. Likewise, for complex “space-precludes” reasons, we've seen the long-going (1983) *UNIX Review*, outliving all its UNIX rivals [Reference 2], moved by market forces to (1998) *UNIX Review's Performance Computing* and now, as I write, to simply *Performance Computing*.

Meanwhile, Linux revives the old UNIX “Live Free or Die” spirit of the 1970s. The only Open Road beckons. *Linux Journal* leads the way. And lest we become too hyper-serious, browse http://www.gnu.org/fun/ and my next column!

References



Stan Kelly-Bootle (skb@crl.com) has been computing on and off since his EDSAC I (Cambridge University, UK) days in the 1950s. He has commented on the unchanging DP scene in many columns (“More than the effin' Parthenon”--Meilir Page-Jones) and books, including The Computer Contradictionary (MIT Press) and *UNIX Complete* (Sybex).

Archive Index Issue Table of Contents

Advanced search

Letters

Various

Issue #70, February 2000

Readers sound off.

Correction

In the November issue of *Linux Journal* on page 18, Terra Soft Solutions, Inc. is presented with the URL of www.blacklablinux.com, which is *not* our corporate web site (http://www.terrasoftsolutions.com/), and again, one sentence later, our primary distribution Yellow Dog Linux is presented with the URL of www.yellowdog.com, which is a completely nonassociated design firm on the East Coast who then called us and complained that Yellow Dog Linux customers are now calling them asking for sales and support information. The correct URL is http://www.yellowdoglinux.com/.

—Kai Staats, Terra Soft Solutions, Inc.kstaats@terraplex.com

I apologize most sincerely for this faux pas —Editor

Merger of the Month

November 15 marks a significant day in the history of Cygnus, Red Hat, free software and open-source software. Both Cygnus and Red Hat have long admired each other's organization and innovative leadership in engineering, promoting and maintaining Linux software. We are both proud of the fact that the software developed and maintained by our companies has become fundamental to the free software and open-source communities and is becoming fundamental to the larger commercial markets as well. We are also mindful of the fact that we did not get to this point alone.

Having spent a lot of time over the past few months contemplating and discussing a possible merger, we have amicably and willingly signed a definitive agreement to merge. To say that we are excited about the possibilities is the understatement of the decade. We believe by enabling developers from both

companies to work together more closely, with a common and larger purpose, we can drive the open-source revolution faster and further than would otherwise be possible.

We hope you will continue your support of both organizations as one and help us move open source even further. Please don't hesitate to e-mail us or anyone else you know at either company if you have any questions, concerns or suggestions about how we can make this merger a “Good Thing” for everybody.

—Donnie Barnes and Michael Tiemann djb@redhat.com

Ads, Ads, Ads

I have been receiving *Linux Journal* for almost a year now, and it has drastically changed in this short period. I am referring to the annoying ads on *every* page. It seems your magazine is no longer content-oriented, it is now *stuff* oriented. I know you're just trying to make money, but when do you have enough? This really bothers me, and I will probably not resubscribe to your magazine.

—Kyle E. Wright tachyon9@purdue.edu

You are right: we do have a lot more ads now—with the growth in Linux popularity and products, more companies are placing ads with us. We are a major advertising spot for them. However, we have not decreased our content pages. Instead, we have expanded the number of pages in each issue. We are now at 132 pages, where just a few months ago, we had only 100. (This issue, February, will have 156.) Advertising money pays for us to grow and offer you more and better content and services for your subscription —Editor

USB and Linux

I noticed an error in a letter from a Mr. Ellerby in the November *LJ*, which mentioned a lack of Linux USB support.

Last Thursday (October 21, 1999) evening in Austin, TX, Mr. David Nelson (davido@austin.rr.com) demonstrated to the Austin Linux Users Group (ALG) a USB HP scanner and a USB camera on an i386 Red Hat Linux machine. The camera was supplied by Jason Cohen of Photodex Corp., producer of Compupic 4.6 beta 20.x Linux. It worked! David's home page lists information on this topic. The URL is http://www.jump.net/~dnelson/.

—Donn Washburn n5xwb@inetport.com

One thing to remember is the November issue came out in mid-October before this announcement, and had been put to bed more than a month earlier —Editor

Too Much Red Hat?

I've been following your journal for about a year, and had a look at older issues from the Web. All went well until #66. News about Red Hat shares was okay, but now in #67, I started feeling that these are advertisements, rather than news. If *Linux Journal* is a "journal" rather than a magazine, I think these writings should be classified as advertisements. You may say that these help the advocation of Linux, but as long as you don't advocate the other companies, it will look as if you're advocating only Red Hat Linux.

I have been using Red Hat Linux for about two years, both for servers and my desktop machine, so I have no antipathy toward Red Hat. But I am not buying your journal with the dream of making money on Wall Street, and I am not interested in how profitable investing in Linux can be.

—Can Bican bican@metu.edu.tr

The upFRONT section was created so that we would have a place to put information we felt was of interest to our readers that was not necessarily technical. Being the first and only Linux IPO so far, we thought people would be interested in the numbers —Editor

Praise Indeed

"Hacking an Industry" by Doc Searls in the November issue is a very well-written and appropriate article—it is the stuff I tend to look for and reference in *Linux Journal*!

—Russell McOrmond russell@flora.ottawa.on.ca

May I blush? Thank you. —Doc Searls, info@linuxjournal.com

The Ties that Bind

I just wanted to correct one error in the interview with Guido van Rossum in your December issue. Guido thought the KDE/Qt Python bindings weren't being actively maintained. That's just not the case. Phil Thompson very actively develops the PyKDE/PyQt bindings—the PyQt bindings even work with the Windows Qt. It's a very slick tool kit, very easy to use, and since it's based on an object-oriented tool kit, fits very neatly into the Python way of programming.

The bindings are at http://www.river-bank.demon.co.uk/software/, and you can find a tutorial on my web site: www.xs4all.nl/~bsarempt/python/tutorial.html. There's also a mailing list at http://mats.gmd.de/mailman/listinfo/pykde/.

—Boudewijn Rempt boud@rempt.xs4all.nl

Getting Ready for the Millennium

Best article (“Millennial Musings”, Peter Salus, December 1999) I've seen so far on the subject—reset that one VCR, and it will be good for another century—that is what a lot of the answers will be; just reset whatever needs it.

There is one more thing: in January 2000, we will start building the Y2.1K problem. Really, it's because two-digit dates cover a man's lifetime.

—Dan Tillmanns tillmanns@earthlink.net

Hot *Damn*

I was just reading Doc Searls' “Linux for Suits” column in the December *LJ*. That's it; it's over; I can go home now—he is generating ESR-like sound bites better than mine. In particular:

> Freedom is an efficiency that drives value.

That's so good, it gives me goosebumps. I wish I'd thought of it first.

—Eric S. Raymond esr@thyrsus.com

Interesting

Thanks so much for the great interview with Guido van Rossum (Dec. '99) which has sparked my interest in Python.

Not only was the article entertaining, but he and writer Phil Hughes confirmed my own first impressions of that other language, Perl: “All the speed of BASIC combined with the readability of FORTH.”

A quick visit to the newsgroup comp.lang.python this morning also proved interesting: there's an attitude of helpfulness there that is a real breath of fresh air after all the smoke and flame in the *perl groups.

I hope *LJ* will continue to publish articles featuring a wide variety of programming languages. No single language is right for everyone, so when it comes to choosing one, a few examples from actual users with real applications

can be far more valuable than any number of man pages written by ace programmers who list their second language as “human”.

—Irv Mullins irv@ellijay.com

Archive Index Issue Table of Contents

Advanced search

Big, Bad LINUX

I thought you might (just might!) be interested in the impressions of one who has earlier experience of UNIX, but has only recently started experimenting with LINUX. (Since taking early retirement, I have continued to fool around with PC/IX, MINIX and similar, but the need for everyday applications forced me to grit my teeth and face up to W9x.)

I'm disappointed. I have a range of computers all of which are capable of running one or other of the variants of UNIX I have lying around. But not LINUX.

Only the newest, fastest machine is capable of running it at all, and then so slowly (by comparison with W95) that it's just not practical. Of course, new machines are getting faster/bigger all the time, but I do worry about an operating system which will not perform at all satisfactorily on machines which run other System V distributions perfectly well.

LINUX is just too big and cumbersome and the machine configuration threshold too high. None of the distributions I have are anything like modular enough, so huge amounts of what gets installed are likely to be redundant in many specific environments. Some hand-pruning is possible, but requires skills which the average user shouldn't be expected to have.

The development labs. where I used to work produced cut-down variants of their in-house System V UNIX for a whole range of experimental machines, right down to hand-held data-collection devices with relatively slow processors. Is it not time that some thought was given to the needs of machines other than big servers and the like? There are a lot of people “out there” who would just love to dump W9x, just as soon as someone offers a practical alternative. LINUX is not that alternative. Nor, if my assessment of the way it is heading is correct, is it ever likely to be.

—Chris. Aubrey-Smith, Cas194@aol.com

Product of the year

I'm a little disappointed in your choice for product of the year, Caldera OpenLinux 2.3. While I can't say that I have a better idea, I do know that I've had significant problems with this product.

While installing it on a system with an onboard Adaptec aic7890 SCSI Controller, Lizard will just hang. Using the console based installer, Lisa, provides the same result. It was only with Rhat 6.0 and an updated install

disk could I get linux installed on this system (I'm a newbie, so this is all relative).

I would like to say that I look forward to your magazine every month, and generally find it very useful and informative. Please though, in evaluating products, don't be so kind. The community can only benefit by a little prodding. Thank You,

—Paul, pkwoods@pop.nwnexus.com

> *We ran across the hanging problem with the Adaptec too. The fix, given to us by Caldera support, was quite easy and was printed in the magazine in the review. At the boot prompt type* `install er=cautious`*—Editor*

Why I still use Windows 95...

I'm a big fan of Linux, and have been since 1994, when I first installed Slackware on my 8MB, 75MHz Pentium system. Since then, I've moved to Slackware “96”, 3.6 and 4. I've used Linux to prototype applications I am working on at work. It has been a marvelous learning tool for me.

Linux doesn't blow up. Individual applications do, and Linux, like Solaris, and other systems, simply removes the offending application from the system. That I don't have to re-start Linux every time I use leads me to conclude that it is a high quality product.

I want to use it as my main operating system, and have been trying to move to it for the past 28 months.

I simply can't.

Currently, I have an ALPS MD-1000 printer, and an A-Open software based modem. Linux flat won't work with either of them. I've tried. I've searched Caldera, RedHat, Debian, and assorted distribution vendor sites. They all indicate their distributions won't work with these products. The standard “Linux Guru/Advocate” cry seems to be, “Buy real hardware.”

Again, I can't. I have children, a house payment, two car payments. I simply don't have the money to drop a new printer and modem in, when the ones I have work fine, and accomplish everything I need them for.

And that's just the hardware side of the problem. The software side is far more complicated.

In my work, I am required to maintain project schedules in a format that is compatible with Microsoft Project. To date, I have been unable to find project managmement software for Linux that would provide this capability at a price a mortal can afford. Hence, I use IMSI's Turbo Project

for Windows 95/98/NT. At $90.00, it works, and it does what I need. Linux doesn't.

The documents I write have to be compliant with Word 97 file formats. I have tried Star Office 5.1, and found it doesn't meet the requirement. More accurately, if I stick to simple tables, and text, with no graphics in the document, Word 97 can read an exported document, even though there are formatting problems (font sizes, page margins, etc. The problem is of a scale that the page counts even change...). WordPerfect 8 has similar problems, although not as sever.

Hence, I'm stuck with Word 97, under Windows 95.

Having limited cash resources, and looking toward the next year, I will have to upgrade my computer system. However, this will be mostly software upgrades. I expect to obtain a new Linux distribution, hopefully with Star Office and Word Perfect (maybe even Corel's Linux?). Printing will continue to be a headache (print to file, split that hummer up across a zillion floppies, take it to the office, re-assemble it, and dump the postscript file to the network printer... Real user friendly...), and using Linux for internet access will remain a wish.

I will obtain a Windows upgrade, and Word 2000. I have no viable alternative in the land of Linux. And this will remain the case until Linux works with the hardware I own, and the hardware I can afford.

Thanks for publishing such a solid magazine. Keep doing so, and I'll keep subscribing. It's been a fun 4 years.

—lurch, lurch@picusnet.com

Re: MSC.Nastran on Linux

While musing over the November/1999 issue of *Linux Journal* we noticed on page 9 (items 28-30) that you had converted Bill Gates III into ASCII and got 666. We tried this using the convention B=66, i=105 (ASCII codes) and summed the digits. We did not get 666. Can you please tell us how you got 666?

Thanks,

Penguins Rule, NT drools,

Joe

—Joe Griffin, joe.griffin@mscsoftware.com

> *Someone sent this in as e-mail and unfortunately we did not check the figures. Sorry. —Editor*

Religion and LJ

After reading the January '00 issue of *LJ* and seeing in the *LJ* Index that there were 7 complaints about your interview with Linux Torvalds I thought I would go ahead and make it 8. I was going to respond before but it escaped my mind and since I'm at work now and reading the Jan. issue I thought while its fresh in my mind I would make the comments I was going to a couple weeks ago.

I think the comment you had about religion based on one of Linus' answers was uncalled for not to mention incorrect. Our country was founded on the premise and beliefs of Christianity. No where in our Constitution does it state there should be a separation of church and state although our government likes to tell us that statement exists. Our country was founded upon the beliefs of Christianity and upon the belief in God whether atheists want to acknowledge that or not. God exists whether they care to believe in Him or not. I find it odd that a lot of people in the CS and IT fields are atheist. I haven't yet figured out the cause of that; maybe it's just a weird coincidence. Who knows? I for one though am a Christian and I am in the IT field. I am probably one of a select few though but I don't mind. I think you need to be more aware of the subjects you talk about in your interviews before you make comments that aren't true. <begin sarcasm> And try not to be too biased if you can. <end sarcasm>

Thank you for listening and with the exception of that article, keep up the good work on your great magazine.

—Brandon McCombs, bmccombs@dns.pulsenet.com

Re: Comments about Jan. 2000 issue #69.

I must admit I was disapointed with two articles in this issue.

On page/44/45, Marcel Gagné wrote about getting connected using ppp. He chose to use a text baased login, that is getting pretty rare, because windoze can't do this without extra software from the ISP. The PPP-HOWTO hasn't been updated since 1997, and is really obsolete.

The two helper tools are also biased towards the text based login. If, as is fairly common, the newbie even sends a newline after seeing the modem connection stanzaa, sending the ISP into a misconfigured text script, which does not permit connection. If you really want to see a good page, see

http://axion.physics.ubc.ca/ppp-linux.html

On page 86/87 Gene Hector makes an invalid statement that the first and last subnets are not usable. While this was true for older versions of Novell, it is not true for any form of *nix.

Hope this helps,

—Bill Staehle, staehle@netvalue.net

Name left out?

I read with interest the article “Audio and Video steaming..” in the Jan 2000 issue of the *Linux Journal* (page 36). I would like to bring your attention to the photograph of the Network Engineering team (Figure 1). There are 11 people on the team. But I see only 10 names mentioned. Someone has been left out.

—Makarand Kulkarni, makarand_kulkarni@usa.net

Linux Journal Letters Page - Linux should go commercial

Firstly, I think it's really great that your mag is available in Scotland now, as you lot seem more in touch with what's going on than the single UK effort. The reason I write is to state that I think that Linux should become a commercial interest, and be sold, or at least KDE and GNOME should be sold. My reason is thus: Imagine is Red Hat discovered a way to make Linux so easy to use, as easy to use and install as say, a PalmPilot. There would not be much demand for their boxed Linux distros as the only reason to buy their's is the support, which would no longer be necessary. This means that it is against Red Hat's and Caldera's interests to make “their” products easy to use. Similarly, if either company found a way to make a full Linux distro which was very small, and a tempting download, they would be reluctant to produce it, as we'd all download it and not buy it on a CD.

On the same subject, we keep heearing that the GPL software is better than commercial, I would like to see one, single, lone example of this. People state the Gimp, excellent software, as good as Painter, or Photoshop? No. Worse still, KDE and GNOME, GNOME is cool buy very sluggish on my 366MHz Celeron, KDE is not as fast as my brother's 233MHz iMac in general reponsiveness (don't get me started on how it compares to my 36MHz RISC OS computer!). I think the best option for the dedicated Linux user is Solaris/CDE with LXRUN, only problem is that in the UK, the free Solaris offer is over, so we have to pay for it, and take the risk that we will prefer what we had. Here is what I think, KDE 2 should be sold for about $30, this would raise rather a lot of money for the KDE group, an allow them to hire some more coders to make development of KDE 3 faster.

That's all. Keep up the good work.

—Garry, gaz@popidea.co.uk

Correction to "Networking Oddities"

I wanted to comment on one of the answers to your question printed in the December issue of The *Linux Journal*.

The second answer, from Marc Merlin, says:

There is another possible reason [...for the connection refused messages, etc...]: the connections are being denied by tcpwrappers...

In fact, this sort of error does *not* indicate rejection by tcpwrappers. Tcpwrappers work through inetd; thus, before the wrapper has a chance to accept or deny a connection, the connection must already have been accepted by inetd. Tcpwrappers can choose to drop the connection or pass it on to the appropriate service.

If the connections are rejected, this will result in telnet connections that are accepted but disappear quickly, or by ftp errors similar to the following:

421 Service not available, remote server has closed connection

The only way you'll get the errors you've described is if (a) inetd is not running, (b) inetd is running but those specific service shave been commented out of /etc/inetd.conf, or (c) there is a firewall configuration on your system that is rejecting the connections.

—Lars Kellogg-Stedman, lars@toad.bu.edu

Correction to "Boot Process Question"

In the December 1999 *Linux Journal*, a questions of yours was printed regarding a problem with bringing up your network interfaces at boot time. What you describe is a classic configuration problem, and I'm afraid that both of the printed answers missed the point entirely.

A properly configured Red Hat system doesn't require you to go around modifing the order of your rc files. The error you describe is usually caused by the simple fact that your network configuration is set to enable your ethernet interface at boot time, generally via an entry similar to the following in /etc/sysconfig/network-scripts/ifcfg-eth0:

ONBOOT=yes

For interfaces corresponding to PCMCIA devices, the ONBOOT paramter should be set to “no”:

ONBOOT=no

The initialization of the device will be handled by the PCMCIA software when the pcmcia rc script runs later in the init process.

Modifying the order of your rc scripts can be a bad idea unless you are sure that you won't be breaking any dependencies. While it might fix the symptoms in this situation, it's not really fixing the problem, and is ultimately an unnecessary hack.

Cordially,

—Lars Kellogg-Stedman, lars@toad.bu.edu

Re: Correction to "Boot Process Question"

> On Wed, Dec 08, 1999 at 11:55:24AM -0500, Lars Kellogg-Stedman wrote: A properly configured Red Hat system doesn't require you to go around modifing the order of your rc files. The error you describe is usually

Actually before 6.0, that wasn't true, and I filed a couple of bugs about this. If you re-read my answer, you'll see that I said that what the user saw was normal and expected behavior. I merely explained how to change the boot order should there be some other reason to do it (RH can't get all the possible cases right, for instance, they used to run routed before my PCMCIA interface was brought up).

> caused by the simple fact that your network configuration is set to enable your ethernet interface at boot time, generally via an entry similar to the following in /etc/sysconfig/network-scripts/ifcfg-eth0: ONBOOT=yes

> For interfaces corresponding to PCMCIA devices, the ONBOOT paramter should be set to “no”: ONBOOT=no

This is true but it's not relevant to the current point. If you look at the code, you'll see that it says:

```
# is this device available? (this catches PCMCIA devices for us)
/sbin/ifconfig ${REALDEVICE} 2>&1 | grep -s "not found" > /dev/null
if [ "$?" "0" ]; then
    echo "Delaying ${DEVICE} initialization."
    exit 1
fi
```

So what RH does is to try to bring the interface up, but that fails for PCMCIA as it's not running yet.

> The initialization of the device will be handled by the PCMCIA software when the pcmcia rc script runs later in the init process.

That's correct, PCMCIA brings the interface up later.

> Modifying the order of your rc scripts can be a bad idea unless you are sure that you won't be breaking any dependencies. While it might fix the symptoms in this situation, it's not really fixing the problem, and is ultimately an unnecessary hack.

I said that he didn't need to change the order for his problem, but I explained how to do it in case there was some other good reason to do it. Since you don't seem to believe me, I'll give you examples:

PCMCIA is started by /etc/rc.d/rc when it reaches S45. That's already too late if you plan to mount SMB or NFS filesystems as netfs starts at S14. Another example is syslog (S30) as you may be sending syslog output to another machine, or atd (S40) as /var/spool could be NFS mounted...

—Marc Merlin, marc_f@merlins.org

Re: Multifax Article in Recent Issue

I read with interest your article on Multifax in the December issue of your magazine. But I found one obvious thing missing from the article. Where are you going to get the fax numbers? Are you going to purchase them? I recently dealt with this issue with a company for which I work.

I run a small Linux consulting company on the side so we did something about it. We fired up the old brain and wrote a utility that will obtain all the fax numbers by area code or prefix. I told the guys at OCLUG (Orange County Linux Users Group) about it. I have already been flamed once. I feel our utility is a good thing which allows you to market to people who leave their fax machines on. Anyway, if you are going to write an article about faxing, Linux can provide tools that make development much easier than under Windows. If you want to do an article on our software, I would be happy to share the source code with your magazine. If you don't want to write an article about it, but publish this letter, I leave this as an exercise for the more advanced readers of *Linux Journal*. I think we should all remember that goverment, organized crime, most marketing organizations and many other people have these numbers, and as Linux users, we might as well have them too.

—Gary Baker, gb9999@usa.net

Archive Index Issue Table of Contents

Advanced search

upFRONT

Various

Issue #70, February 2000

Stop the Presses, *LJ* Index and more.

POLITICS AND THE LINUX PLATFORM

Let's talk about a campaign platform that really matters: the operating system that supports each presidential candidate's official web site. Can you guess who runs on what? Let's take a look. (Cue the drum roll...)

- Running on Windows NT or Windows 98: Republicans Gary Bauer and George W. Bush, Democrat Al Gore and Reform Party candidate Pat Buchanan. All serve pages with Microsoft IIS.
- Running on Solaris: Democrat Bill Bradley and Republicans Steve Forbes, Orrin Hatch and Alan Keyes. All but Forbes use an Apache web server. Forbes uses Netscape-Enterprise.
- Running on BSD: Libertarian Harry Browne. Uses Apache.
- Running on IRIX: Republican John McCain. Uses Rapidsite.
- Finally (intensify that drum roll), running on Linux: Independent candidate Bob Smith and Reform Party candidate Donald Trump (arguably the poorest and richest guys in the race). Both use Apache.

While it shouldn't count, the domain squatter who owns johnmccain.com and patbuchanan.com runs Apache on Linux.

Our source for this trivia is Netcraft (http://www.netcraft.com/whats/). If you have time on your hands, it might be interesting to see if any of these guys have swapped servers since we took this survey.

—Doc Searls

THE INCENDIARY

Back when I did market consulting, I filtered client candidates by asking them to agree with my marketing logic:

1. Markets are conversations.
2. Conversation is fire.
3. Marketing is arson.

I've never met a marketer with a better instinct for arson than Donald B. Marti, former proprietor of Electric Lichen and now a Technical Marketing Manager with VA Linux Systems. Don is a gonzo marketer of the highest order. Talk about starting fires; Don is the guy who discovered the silly Unisys patent on the .GIF compression algorithm, made a lovely stink about it, then staged "Burn all GIFs Day", along with a web site (http://www.burnallgifs.org/) to coordinate and commemorate the event.

Don was also a prime mover behind "Windows Refund Day", and he's the skilled hacker behind one of the Web's great memes: the operating system sucks-rules-o-meter (http://srom.zgp.org/).

More than a great incendiary, Don is a revolutionary thinker. Here are just two lines he dropped in a recent conversation:

"Now it's time to hack the real world and let other people write web sites about it."

"The Sucks-Rules-O-Meter is the first crude attempt to do the opposite of advertising—in which the customers do the writing and the supplier does the reading."

Here at *Linux Journal*, we are adopting the sucks-rules system to keep tabs on what people really say about the various Linux distributions Hey, it's too good a hack not to use.

—Doc Searls

THE NEXT OPEN SOURCE VICTORY: INSTANT MESSAGING

While Microsoft and AOL were joined in an Instant Messaging (IM) "war" last fall, the open-source development community did what it does best. It hacked together a working alternative that outdoes both rivals by doing what neither party seems to know how to do: work with everybody else for the benefit of not just the customer, but the whole marketplace.

The project is Jabber. Think of Jabber as the Linux of Instant Messaging. Then think of Jeramie Miller as the Linus Torvalds of Jabber. About two years ago, Miller became annoyed with the popular but inflexible and proprietary messaging systems from AOL and Mirabilis (since bought by AOL), and came up with the idea for an instant-messaging system that would be open and able to do things the other systems could not, such as keep up with multiple clients running at once.

Just as it happened with Linux, a devoted group of developers and users quickly joined in and got to work. Using XML, they created a "transport" between any and all IM platforms. Among other things. *osOpinion* calls it "the end of instant messaging as we know it".

At the most practical level, this means users of AIM (AOL Instant Messaging), ICQ (AOL's own alternative) and Microsoft's new messaging system will all be reconciled by a new, independent, open-source IM platform. It also means Linux users, still ignored by AOL and Microsoft, can not only participate in the instant-messaging movement, but clear its evolutionary path as well.

Looking ahead on that same path are at least two commercial companies: Webb Interactive Services and Corel. Webb's president is Perry Evans, perhaps best known for creating Mapquest a few years back. Mr. Evans liked Jabber so much he hired Mr. Miller for similar reasons as Transmeta hired Mr. Torvalds. Webb and Corel are now partnering to include Jabber in Corel's new Linux distribution, among other things.

To join the Jabber development conversation, or to participate any other way you like, visit http://www.jabber.org/.

—Doc Searls

LJ INDEX—February, 2000

1. Amount invested by venture capitalists in Silicon Valley during the first nine months of 1999: **$7.7 billion US**
2. Amount spent during the third quarter alone: **$3.36 billion US**
3. Percentage increase over the preceding quarter: **27**
4. Largest single investment in the same quarter (for Webvan): **$275 million US**
5. Amount lost by Webvan in the first six months of 1999: **$35.1 million US**
6. Sales by Webvan during the same period: **$395,000 US**
7. Valuation of Webvan's total outstanding shares on November 29, 1999: **$7.7 Billion US**

8. Number of attendees at Comdex Fall '99: **400,000**
9. Number of attendees at Comdex Fall '99's Linux Business Expo: **38,000**
10. Positions of Linus' keynote and the LBE as attractions at Comdex Fall: **#1 (CBS News)**
11. Number of stores in which AutoZone will install Linux terminals: **2,800**
12. Increase in one share of Red Hat stock after announcing a support deal with AutoZone on November 19, 1999: **22 1/2 points**
13. Red Hat's share price at close of the same day: **$236 US**
14. Red Hat's market cap at that share price: **$16.279 billion US**
15. Percentage increase in Red Hat's share price since its August 1999 IPO: **1600**
16. Cobalt Networks' annual revenues in 1998: **$3.537 million US**
17. Cobalt Networks' revenues for the first nine months of 1999: **$13.849 million US**
18. Cobalt Networks' IPO share price: **$22.00 US**
19. Cobalt's share price on November 29, 1999: **$156.50 US**
20. Cobalt's market cap at that price: **$4.272 billion US**
21. Average price per month of DSL service: **$30-60 US**
22. Total number of DSL subscribers in the country: **300,000+**
23. Total number of subscribers to cable high-speed Internet access: **2 million**
24. Amount of Red Hat stock sold to Cygnus for the takeover: **6.7 million** shares
25. Value of that stock on the day of the takeover: **$674 million US**
26. Current value of the stock: **$1.5 billion US**
27. Price of feeding a penguin for one year: **$700 US**

Sources:

- 1-15: *San Jose Mercury News*, TheStreet.com, *Linux Today*, CBS News, Hoovers
- 16-20: Cobalt numbers from company filings with the SEC
- 21-23: US West Communications
- 24-26: CNN, NASDAQ
- 27: Sea World (http://www.seaworld.org/)

BARREL SCRAPINGS REVISITED

Several months ago, we defied the conventional wisdom that says there are no good domain names left. We found earwig.com, bizfloss.com, stoptalking.com, halfcat.com and fubar.mil were all untaken. With good reason, all but

earwig.com remain on the block, which means their negotiable value is somewhere south of $35/year.

Maybe you guys can actually use a few from this round. Hey, if they don't work for your domain, maybe you can use one to name your band!

- coapathetic.com
- bedkill.com
- barfwash.com
- linkfetus.com
- mailpail.com
- linuxgoddess.com
- domainpain.com
- petsurface.com
- buttcramp.com
- fallinghope.com
- bilespike.com
- luckfarmer.com
- umoo.edu
- nterior.com
- beerleer.com
- neithersex.com
- toygod.com
- cashbird.com
- possuminnards.com
- gyrosnooze.com --Doc Searls

VENDOR NEWS

Compaq Computer Corporation has partnered with **Sair, Inc.**, a provider of Linux training material and certification examinations, to become a Sair Internal Training Organization. Compaq will train its key personnel worldwide with Sair Linux and GNU Certificq ASE Linux certification track.

Progressive Systems announced its Phoenix firewall has received certification from the International Computer Security Association, the foremost independent evaluation and certification facility for network security products worldwide. The Phoenix is the first ICSA-certified firewall to support Caldera, Red Hat, TurboLinux and SuSE Linux distributions, further assuring business customers and value-added resellers that Linux is a viable platform for enterprise network security.

SuSE Linux AG, Europe's leading Linux distributor, announced it has entered into a partnership with VA Linux Systems to co-develop a SuSE Linux software load for VA Linux workstations and servers. The partnership will help extend SuSE Linux's presence on pre-installed Linux systems, while expanding the range of fine-tuned Linux distributions offered and supported by VA Linux Systems in its solutions next year. Co-development is set to begin in December.

Red Hat, Inc. announced the promotion of Matthew Szulik to CEO and president. Szulik, who joined Red Hat as president in November of 1998, has been instrumental in the success of Red Hat Linux, the execution of Red Hat's IPO and the global expansion of the company's open-source service offerings.

Antarctica IT, Inc. and **Caldera Systems, Inc.** will work together to provide Linux services in Boston and elsewhere in New England. Antarctica IT will provide systems integration and front-line support for Caldera Systems' OpenLinux operating system. New England businesses using OpenLinux will now benefit from on-site support and services obtained from a local service provider. Caldera Systems will provide technical resources, 24x7 telephone support, and experience in Linux.

Stormix Technologies, Inc., a provider of Linux-based software and solutions, announced that Storm Linux 2000 will ship with the full version of Sun Microsystems' StarOffice 5.1a, a leading office suite on the Linux platform. StarOffice is full-featured, interoperable and multi-platform. It is available for free download at http://www.sun.com/staroffice/.

Magic Software Enterprises announced it has merged Canada's Open Sesame Systems with its U.S. operations to more effectively deliver e-commerce and other enterprise-level business solutions and services to a larger North American audience. The merger, which is effective immediately, will enable each company to leverage the other's resources and strengths. The acquisition of OSS' IT business is expected to have a positive effect on Magic's bottom line.

Digi International, Inc., a provider of server-based communication adapters, announced it has joined the Enterprise Computer Telephony Forum (ECTF), publishers of interoperability specifications for the computer telephony industry. Joining as a principal member of the forum, Digi* will take an active role in helping to define computer telephony hardware and software interoperability standards, including the ECTF's computer telephony services platform architecture.

Magic Software Enterprises announced the appointment of Israel Teiblum as president. Teiblum, who has served as chief financial officer of the company since February 1997, will retain his position as CFO while assuming this new

role. Jack Dunietz, the company's chief executive officer, has been appointed co-chairman of Magic.

Red Hat, Inc., a developer and provider of open-source software solutions, announced it will be providing on-site consulting, services and support to **AutoZone** as part of the auto parts retailer's program to base its chain-wide Intranet systems infrastructure on Red Hat Linux. Red Hat's services organization will provide consulting and support services for a network of approximately 3,000 Linux-based Intranet terminals in AutoZone's store locations throughout the United States.

The Linux Internationalization Initiative, also known as Li18nux, has established the initial set of subgroups toward Linux Internationalization: written specification, internationalization system architecture, API/application development environment, graphical user interface, text tools, web technology, input method, typography, globalizable document, inter-application collaboration, heterogeneous interconnectivity, web site administration, localization of web contents and printing. Detailed information can be obtained at http://www.li18nux.org/.

Linux Events

- LinuxWorld Conference and Expo, February 1-4, 2000, New York City, at the Jacob Javits Convention Center
- LinuxWorld/LinuxExpo, February 3, 2000, Paris, France

STRICTLY ON-LINE

The following articles are posted on *Linux Journal On-Line*, our web site at http://www.linuxjournal.com/. We wish we had room to print every article in the magazine, but infinite space is just not available. Also, all articles in the current issue and past issues are posted on the *LJ* interactive site at http://interactive.linuxjournal.com/ for subscribers. Non-subscribers can find all articles for issues 1-32 (1994-1996) on the on-line site.

T/TCP: TCP for Transactions by Mark Stacey is a discussion of the operation, advantages and flaws of an experimental extension for TCP. Learn more about TCP as you read about this new protocol designed to address the need for a transaction-based transport protocol in the TCP/IP stack.

POSIX Thread Libraries by Felix Garcia and Javier Fernandez is a look at five libraries which can be used for multi-thread applications. The results of the authors' studies are discussed in this article. They also give us an in-depth look at threads: how they can be used and controlled in your applications for greatest benefit.

Laptops for Linux! by Jason Kroll is a review of the two laptop products currently available for Linux: the Attache from LinuxLaptops and the AS-LT300 from ASL Workstations. From ergonomics to software, find out all about these two products on-line.

Linux and Open-Source Applications by Peter F. Jones and M. B. Jorgenson provides us with a look at system security and how to build a secure and trustworthy computer platform. Learn about viruses, worms and Easter Eggs and what they can mean to your system. The authors answer the question, "Is open source the best way to get a truly secure system?"

GRAPHON INTERVIEW

by Doc Searls

GraphOn's stock doubled in November. Early that month, the company took a tack toward Linux with its new Linux Bridge product, and acquired a technology patent protecting its whole Bridge series of products. These products allow users to operate applications on other platforms—essentially using those platforms' workstations as terminals for applications run on servers elsewhere. GraphOn also announced an OEM deal with Corel, which was a big hit at Comdex with its new Linux distribution and application suite. One could almost see the stock rise as the implications (in particular, Windows-to-Linux migration) became apparent.

While at Comdex, I spoke with Robin Ford and Walt Keller, the couple who founded GraphOn. Robin is Executive Vice President, Sales and Marketing, and Walt is CEO and President. The talk was recorded, and this is the edited transcript.

Doc: What's going on between you guys and Corel?

Robin: Years ago, Corel started developing a technology called J-Bridge, which allows you to access a Windows application from any desktop over any kind of connection. They were doing it because they needed to web-enable their existing applications. GraphOn has that technology for the UNIX and Linux market. We allow people to access a UNIX or Linux application from any desktop over any connection. This is great because you can run an X application over a low-bandwidth line, and it's like running an X server on your desktop. Corel ran into some challenges in their technology about this time last year. They had the core technology—the server part was coming along well—but they needed to put together the protocols and the client. They were already partners of ours for other reasons, and as they became more familiar with our technology, they realized we do this sort of thing for a living. It made more sense for us to carry this forward, as we had the low-bandwidth protocol. Thus,

we acquired Corel's technology, which was unfinished at the time, and integrated it with our software.

Doc: And where are you with it now?

Robin: We call it Win Bridge, and we launched it here at the show. We've also "OEMed" our technology to Corel. Thus, any Windows-based application can be served to any desktop, and any Linux-based application can be served to any desktop—over any connection.

Walt: The exciting news for Corel is they can now insert very strong support into the Linux desktop. They can take any of their Windows applications and run them on those Linux desktops. By that, I mean the application is still running in an NT environment, but you can view and manipulate it from your Linux desktop as if it were running locally. That's the beauty of this bridging technology. Complete cross-platform capabilities.

Doc: So this has all kinds of implications for support, migration...

Walt: Yes. It lets people easily migrate into the Linux world. It's a difficult thing for most enterprises. They can't say, "Hey, we're going to switch to Linux", and it's done. You need a migration path, and this technology provides that path.

Doc: How is this playing out in enterprises you know?

Walt: Well, the most interesting stuff at the moment—to us, at least—is in China. They don't want to deal with sole-source suppliers. They truly like the idea of Linux, and are very committed to it. Yet when you visit the schools, they're training on Microsoft. But the bridge between one and the other is a migration path that we pave. They're figured out ways to do that with server-based technology.

Doc: So you've got three bridges here.

Robin: Linux Bridge, Win Bridge and UNIX Bridge, each a component of a product called Bridges.

Doc: How do you see it playing out in the Linux movement in general?

Robin: Everybody agrees the next step for Linux is the desktop. To be successful, Linux has to be able to run Windows applications. Corel knows this, and they've been very smart about their strategy. That's why they've OEMed the Linux Bridge technology, and why it's very hot already, even though we aren't shipping until the end of December. When we were on ZD-TV, it was the most outrageous thing. Here was ZD-TV talking about server-based computing

—serving up Linux applications. They invited people to come to the playpen part of our site and download a Java applet that allows them to run WordPerfect running on Red Hat on our server. It was amazing. We had thousands of people registering to download the Linux server portion of our product.

Walt: What knocked people out was they could sit there on their PCs and run Linux without ever loading it. An interesting concept, and a great way to start down the road to Linux or teach it over the Internet.

Doc: I want to ask you guys about this patent that seems to be a source of some controversy.

Walt: We discovered the controversy entirely by surprise. Obviously, it is not our intention to be out there stifling innovation, especially in the Linux community. We became aware of the patent because some of the people we have in Seattle were developing this technology a long time ago. We thought it was in the best interest of our shareholders and customers that we acquire this patent. It covered taking a Windows technology to a UNIX desktop using the X server—in other words, right in the path of our own product development. So our goal was just to acquire it and remove it as a potential problem, for the good of everybody.

Doc: That's interesting. I wonder sometimes if the reason to get a patent like this is as prophylaxis against the Jay Walkers of the world.

Walt: It is. These things are like baseball cards. You've got to trade them. If you don't have any, you don't get to trade.

Doc: You give yourself the right to be the alpha developer in this space.

Walt: It's a very strong form of protection. The truth is, you can't work in this industry without tripping over patents, and a patent of this type is a very strong one. We had to have it—better us than someone else.

Doc: So you want the Linux community to trust you to use it well.

Walt: That's right.

Doc: Let's go back to origins here. For years, you guys were known as a hardware company. What happened?

Walt: About three years ago, we saw the light. Before that, we were in the terminal business, selling to the X community. We did quite well, but the PC won the desktop. What we actually discovered was thin-client computing. We

developed this technology for UNIX in the late '80s; the desktop, the client side, was very thin and all the heavy lifting happened on the server. But we were swimming against the tide in those days, very much against the flow of what people thought client-server should be.

Robin: Desktop-centric.

Walt: And we were going the opposite way. So finally, we decided to get out of the hardware business, take this technology, and make it work on the desktop. We found we were right in the middle of this new Internet conversation which is much more server-centric. Sun snapped it up, then IBM, and we were off and running. We funded ourselves last year, then went public last July, and it's been a real ride ever since.

Doc: I've always been interested in the soul of a company—where it comes from.

Robin: We're a family-oriented company. In case you didn't

notice, Walt and I are married, and the place is very much a family operation.

Doc: Are your kids involved?

Robin: Yes, our daughter works at the company. We welcome people bringing their kids in during the day. It's not unusual to hear little ones in the background. Our people work very hard, often for long hours, so we try our best to integrate having a life with being a Silicon Valley company. Although we've been around for a long time, we behave very much like a start-up. That's our energy—very entrepreneurial—very open-door—not very hierarchical. Walt is involved in almost everything that's going on at the company. It's not a strict reporting structure—just very entrepreneurial and very productive.

Doc: You know each other. That must count for something.

Walt: Everybody here knows each other. This value system we have applies to acquisitions too. We picked up this group in Seattle. The people there had been together for years and wanted to be in on what we have going here. The fit has been excellent.

Doc: You've been around since what, 1982?

Walt: 1982, yeah—a long time.

Doc: Were you married when you started the business?

Robin: No. We got married in 1985. But we were together for a long time before we started out. We just finally admitted to ourselves that it would work out. (laughter)

Doc: That's a great story.

Robin: It is a great story. We've had fun, and we're having more fun than ever now.

Resources

GraphOn site: http://www.graphon.com/

Corel OEM deal: www.graphon.com/News/pr-corel991025.html and www.graphon.com/News/pr-corellinux.html

Technology patent: www.graphon.com/News/pr.patent.html

LJ piece on Graphon in China: http://www.linuxjournal.com/articles/briefs/031.html

GAMES FOCUS

Have you ever heard six-year-olds reminisce about having been five? I confess I am guilty of this sort of thing, feeling nostalgic for computer games that were state-of-the-art just a few years back. Old is not necessarily bad. Modern software is technically impressive, but how useful is it? Still, I have yet to find a computer game that rivals chess in excitement, or Go in elegance and subtlety, while both games are at least 1,500 years older than computers. You'd think 3-D shooters would bore people to death, or maybe we could do something more creative with the amazing virtual-reality techniques we've got. Anyway, Linux remakes of classic games are cultural oddities; I hope some anthropologist catalogs them. In keeping with the theme of “Linux on the Desktop”, here are some games you can play on your desktop!

Defendguin

Linux gamers tend to have a soft spot in their hearts for the disturbingly popular blast-Bill-Gates-to-pieces games, the family of games in which Bill Kendrick's Defendguin is the newest member. Yes, it's the classic Defender game, only the graphics aren't hollow line drawings. The trick is to save the Penguinoids from being mutated by little flying Gates saucers. In honor of the new journalistic tradition of pulling random quotes and holding them up as though they represent the entire community, here's a stellar line from Digital Ebola on http://happypenguin.org/: “It's quite satisfying to blow Bill Gates up.

That alone makes anything worth 5 stars." Check it out at http://www.newbreedsoftware.com/defendguin/.

Black Penguin

Beetle enthusiast Holger Priebs has delivered the goods for Q*Bert addicts. Black Penguin ("rotund" would be more accurate, and anyway, it's a blue penguin) is a fellow who hops around collecting happy things while avoiding his opponent, the Evil Window. It uses Qt, the sources are well-commented, and it's a good game for Linux newbies to get used to compiling software from the source as well as learn from the source. I should warn you that Qt is not LGPL, but I imagine you've already heard. The home page lives at www.priebs.de/blackpenguin.html.

Fleuch

Fleuch, a Linux remake of Thrust from the C64, comes to us by way of Karsten Goetze. It's available only as a binary for SVGAlib, but maybe we can convince Karsten to open up the source if it turns out the game is popular enough. Pilot a ship by turning and thrusting, pick up a ball, and fly away. According to an anonymous quote from happypenguin, Fleuch "is unmatched in supremacy". It can be found at http://start.cgirealm.com/meuch/.

Insane Odyssey X

"The greatest game... we've ever made" reads the Insane Odyssey Episode I Trailer. Aaron P. Matthews (coding) and Seth B. Peelle (music and graphics) of Rival Entertainment team up to bring you this futuristic escape from a mental hospital. The scrolling is smooth, the graphics are outstanding, and the game play is fun. It's a cohesive, polished production, one of the nicer-looking Linux games out there. While it's not a remake of a classic per se, it is a remake of a game the author started years ago and didn't finish. With so many 3-D shooters, don't you miss a good platformer? Find this game at http://rival.clan.net/.

—Jason Kroll

STUPID PROGRAMMING TRICKS

Last episode, we got a scrolltext going in SVGAlib, which looked pretty cool. Pair it with some Beethoven and an artsy, scribbled logo, and you've got a happening intro for your next act of brilliance. Ah, but how *do* we get the music playing? Well, assuming Beethoven is what you want and not some frantic dance pop, we've got to figure out a way to get music into the program.

Years ago, this was tricky. We had to program in assembly code and call a play routine every vertical blank. Before that, pitched beeps had to be carefully inserted into the program, and any deviance from the loop would cause the whole program and/or the music to change speed. Fortunately, today we have multi-tasking machines, so we don't have to bother with anything difficult; we just fork a process or call a library routine. Now the question is, what do we want to do? How to do it is easy.

For music in a Linux program, there are several options. One is to use the sound card's built-in midi synthesizer, which is extremely resource minimal. Xavier XOSXE uses this approach in his excellent, high-resolution escapade SpaceBoom, reserving digital audio exclusively for sound effects. This way, the CPU's resources are left over for the horizontal scrolling landscape and trigger-happy space aliens. Playing midi music is very simple; all you have to do is **fork** to the program **playmidi**. And no, it's not cheating—fork is an integral part of the UNIX model. Consider that by using it, you pay homage to the same function that got us past the first **init** routine. Listing 1 is simple code for starting up your tune.

Just keep tabs on the process ID (PID) so you can kill the playmidi program when you want to change tunes, or when the game is finished. (The SpaceBoom source code, albeit C++, has a more elegant example of how to do this.) With on-board synthesizers improving dramatically, midi files are really a viable option for music. They're easy to create, resource-minimal to play, have small footprints, and are fun.

Another option is to use the time-tested, scene-approved, MODs! From their birth on the Amiga, these things just won't die. MODs are digital audio music files, which basically include audio samples of the instruments, loop points and note data (newer formats include more instrument parameters).

The advantage of MODs is they are digital audio, so you get sample playback, as opposed to midi files which use preset instruments on the sound card; MODs give you more flexibility. The downside is that your CPU needs to dump a *lot* of data to /dev/dsp, so MODs are resource intensive. For example, at 44,100Hz of 16-bit stereo (CD quality), you'll be writing 16 bits of data to each channel 44,100 times every second—that's 176,400 bytes of writing to /dev/dsp every second, not to mention the calculations that go on within the mod-playing routine (volume, mixing, effects, pitch slides, etc.).

These days, processors are incredibly fast, so MODs are more practical than they used to be, but sound cards have better synths than ever before. Wouldn't you like extra CPU for more intensive graphics? While I definitely opt for midi, most people prefer MODs. If you're making a game on CD-ROM, just record a

whole album. Maybe when we all have supercomputers, we can use mpegs in real time. You can choose between MikMod and Midas for MOD playing; MikMod is GPL while Midas is not quite, so in the spirit of GNU, let's try out MikMod. Listing 2 is code for starting and stopping a MOD with mikmod.h.

Next month, we'll finish up on sound for now and take a look at how to use generic digital audio for sound effects.

—Jason Kroll

Listings are available by anonymous download in the file ftp.linuxjournal.com/pub/lj/listings/issue70/3798.tgz.

Figure 1. midi.c

```
#include <signal.h> /* for kill */
#include <unistd.h> /* fork & execlp */
#define MID "YOURMIDIFILE.mid"
int play_midi(void) {
  int pid; /* process id of playmidi */
  if ((pid=fork()))
    return(pid); /* return process id */
  execlp("playmidi","playmidi",MID,00);
  return 0; /* to placate gcc */
}
int main(void) {
  int pid; /* playmidi process id */
  long int c; /* just a counter */
  pid=play_midi(); /* here we go! */
  for (c=1; c>0; c++) {
    /* this counts for a while as the music
     * plays. you could do anything here,
     * ie a scrolltext and artsy animation!
     */
  }
  kill(pid,1); /* kill playmidi process */
  return 1; /* and exit our program */
}
```

Listing 2. MikMod

```
/* Check out the MikMod web site
 * for mikmod and libmikmod
 * documentation and tutorials
 * http://mikmod.darkorb.net/
 *
 * gcc -O2 filename.c -lmikmod
 */
#include <unistd.h>
#include <mikmod.h>
#define INTROMOD "YOURMUSIC.s3m"
int main(void)
{
  MODULE *module;
  MikMod_RegisterAllDrivers();
  MikMod_RegisterAllLoaders();
  if(MikMod_Init("")) {
    fprintf(stderr,
       "Could not initialize sound, reason: %s\n",
       MikMod_strerror(MikMod_errno));
    return 0;
  }
  module = Player_Load(INTROMOD,64,0);
  Player_Start(module);
  while (Player_Active()) {
```

```
      usleep(27182);
      MikMod_Update();
    }
    Player_Stop();
    Player_Free(module);
    MikMod_Exit();
    return 1;
  }
```

THE BUZZ

During the month of November, people were talking about:

- Red Hat's takeover of Cygnus and what it will mean to the Open Source community. Who will be next? Talk is, it will be Corel. By the time this prints, we may know the answer.
- Sun Microsystems recently released version 1.2.2 of their JDK (Java Developers Kit). They initially failed to give credit to the Blackdown Team for its early development work. Sun later apologized. Oh, and Inprise helped Sun with the JDK, too. Sorry, I forgot!
- Everyone talks about the money to be made with Linux. I learned the VA Linux IPO date from my morning barista. Point being, everywhere I turn, people want in on the “Linux” stock. On opening day, VA stock soared and the jubilation was heard worldwide.
- Speculation as to when kernel 2.4 will be released, along with much discussion of its new features and changes.

—Jason Schumaker

STOP THE PRESSES: SUPPORTING LINUX AND OPEN SOURCE

During the past two years many companies have come to support the Linux operating system, but most do not come to open source—their software products remain proprietary. Even those who do open source their software usually do it only for Linux, keeping Windows and UNIX versions closed. On December 7, this trend was reversed in a big way.

Matra DataVision, a French company, announced it would be making its product open source, not just for Linux, but for every platform it supports. And this isn't some little do-nothing product, either. It is an enterprise level product for geometric modeling, ranging from CAD to 3-D geological mapping. Also, Matra is not a small company looking for publicity for their product; its CAS.CADE product accounts for 10% of the total market.

Matra has taken a good look at the Open Source movement and seen the advantages that can be had with an open-source product. The company believes this move will enable them to extend their market outside its current limits with gains, not losses, in profitability. They intend to concentrate their

efforts in support and development of technical applications for specific needs of customers. These are exactly the areas most mentioned by advocates of open-source business models. It is quite refreshing to see a major company take the movement so seriously that they are willing to base their whole business model on it.

Matra also intends to give its developments back to the community. They have a team of 50 people dedicated to furthering open-source development in this area and a web site at http://www.opencascade.org/ that is dedicated to open source. I talked to Sana Abou-Haidar, Matra Marketing Manager, on December 10 and she told me, "We have decided to base the business model on the service part, so the development part can be true open source. The license is completely LGPL-compliant." Our conversation can be found on the *LJ* web site at http://www.linuxjournal.com/articles/conversations/009.html.

Linux has needed an enterprise-level CAD application, and now it has one—one that not only supports Linux, but is also open source. I wish Matra success and hope more companies will follow their example in the coming months.

—Marjorie Richardson

Potential: Red Hat went from a *Linux* company to an *Open Source* company just before the Cygnus merger announcement. This seems to open up a whole new area of "merging" potential.

Fact: The African Penguin is the most endangered of penguin species. This millennium has seen the numbers drop from several million to less than 50,000 pairs. To help, contact SANCCOB (South African National Foundation for the Conservation of Coastal Birds (http://www.sanccob.org.za/).

Penguin Fact: A penguin's eyes are adapted for underwater vision. In air, penguins are nearsighted. --Sea World, www.seaworld.org/Penguins/senses.html

Quote: "I am a wandering anthropologist and trouble-making philosopher." --ESR at Fall Internet World, NYC.

Fact: A DSL connection is 10 to 25 times faster than that of a 56K modem. Most DSL service performs at a minimum level of 256Kbps. (US West Communications)

Rumorville: Word has it that The Big Red Machine (i.e., Red Hat) may be buying Corel. What affect would this have on Linux competition?

Questions: My repeated attempts to have a short e-mail interview with Bill Gates continue to be thwarted. What questions do you have for “the master”? Send ideas to info@linuxjournal.com.

Fact: VA and Loki are partnering to distribute Debian/GNU Linux in the retail market. Profits will go to Software in the Public Interest. This looks like a win/win situation for everyone.

Predictions: VA Linux and Transmeta will announce a partnering relationship during the first quarter of 2000. —Jason Schumaker, 12-2-99

Archive Index Issue Table of Contents

Advanced search

Desktops of the Future

Peter Salus

Issue #70, February 2000

Anthony Trollope, the Victorian novelist, invented a box with a sloping hinged lid which was his “travel desk”. The surface he wrote upon was clearly his desktop.

I have two desktops. They are at right angles to each other. One is littered with papers, books, CDs, a phone, a stapler, several pens, a box of business cards, etc. The other one has four rectangular work areas (called “windows”) and toolbars.

I'm not sure that the “virtual desktop” is my desktop at all.

On the other hand, Anthony Trollope, the Victorian novelist, invented a box with a sloping hinged lid which was his “travel desk”. The surface he wrote upon was clearly his desktop.

In his seminal article “As we may think” (*Atlantic Monthly*, January 1945), Vannevar Bush described Memex: a desktop personal computer. At that point, there was no electrical or electronic computer in existence. Thirty years later, we were on the brink.

Forty years later, in 1985, you could buy a Macintosh, an Apple, a Kaypro, and several versions of the IBM PC (AT or XT). I recall being amazed at the four-inch screen on the Osborne. The MacSE I bought early in 1987 had a 40MB hard disk.

Over this past decade, the concepts of locational computing, the size of hardware, and the necessity for cables have undergone a tremendous metamorphosis. The computer itself has shrunk from many tons to a few ounces. Cellular phones and infrared have cut into the wires; with that cut, the location of a computer is no longer fixed in any way.

Palm Pilots and notebooks of under three kg are ubiquitous.

So Where Will We Go?

We've got the size problem licked, and wires are on their way to vanishing. The actual power problem (battery weight and durability) remains with us. Still, we are well on the way to the Dick Tracy wrist radio—which we might think of as a "Wrist Pilot".

But stop!

The MacSE didn't run windows. You could have more than one file open at a time, but the one you were working on sat on top, and keeping more than a few items open meant your machine ran very s-l-o-o-o-w-l-y. But I loved that trash can in the corner.

It was W, then X, that taught us about multiple windows, and versions of SunOS and DOS in the middle and late '80s that brought home the notion of reliable GUIs (the first GUI I actually used was that of the Apple Lisa in 1983). Motif and the Motif Window Manager came out a decade ago in 1989/90.

Could I actually put multiple windows on something the size of a Palm Pilot? Not if I want to read what's written. How about alternate screens, like on the old IBM PCs? Possibly.

Voice input? Sure. We've come a long way in the last 35 years, but we're not there yet.

Voice output? We're a long way from HAL.

I've been told that most folks want editing capabilities, e-mail, a spreadsheet and a database. None of those take up much disk space, and none needs much more space than the 5x7x1-inch spiral-bound notebook I use when traveling. That's the sort of workspace I want.

Add cellular phone capability, and I can use voice and also browse the Web. I'll bet Motorola, Ericsson and Nokia are working on that already. However, although I'd like to say that a three-inch diagonal screen would do, I think the minimum we'll find usable is 2.5x3.125 (6.3cm x 7.8cm), with a diagonal of four inches (10.16cm).

The shape and flexibility may change, but what it's used for doesn't change as rapidly. Business letters have been around for over 4000 years; double-entry bookkeeping was invented during the Renaissance.

Cell phone, e-mail, appointment book, phone and e-mail directories, memos, database, spreadsheet. And under a pound (450 gm). I'll take two, please. And an extra battery pack.

Peter H. Salus, the author of *A Quarter Century of UNIX* and *Casting the Net*, is Editorial Director of SSC. He can be reached at peter@usenix.org.

Archive Index Issue Table of Contents

Advanced search

The Real Meaning of Markets

Doc Searls

Issue #70, February 2000

“In just a few more years, the current homogenized 'voice' of business-the sound of mission statements and brochures-will seem as contrived and artificial as the language of the 18th-century French court.” --The Cluetrain Manifesto

Seven seconds is the life expectancy of a detail. Phone numbers, names, street signs, the exact words in the last sentence, all gone in seven seconds, give or take a few. We can trick our minds into carrying more of the very recent past, but only one fact remains: memory for detail is wickedly, blessedly, short. Decay is the rule.

Yet meaning persists, and if it has any value, it grows.

Meaning is the valued cargo in our trains of thought. The trains come and go, but their purpose is to leave goods on the platform. That's what we *get* from what the other person says—or what they *deliver* when their words *carry* meaning.

What do we mean when we talk about markets? Do we mean populations of consumers we hope will buy our products? Do we mean fields of battle where companies fight to grab or protect territory? Do we mean invisible hands and other animal forces—bulls and bears—that push stock prices up and down? Do we mean a large drop zone at the end of the value chain?

How about when we use the word “market” as a verb? Do we mean air support for sales troops on the ground? Do we mean selling by another name? Do we mean “serving the customer, no matter what”, as Theodore Levitt put it? Or do we mean nothing, as Tom Hanks' character suggested in the movie *Big*, when he asked “What's marketing?” and his boss replied, “Exactly”?

The truth is, we mean all those things—each one is a metaphor. But here's what we forget: none of them were in use before the Industrial Revolution. None of them have anything to do with what markets were in the first place or how they really work.

The first markets were places, not *targets*, *demographics*, *seats*, *eyeballs* or other abstractions. In the first markets, producers and consumers were a handshake apart—and so were all the other reciprocal market nouns: producer and consumer, vendor and customer, supply and demand. They were all embodied in seller and buyer.

In the first markets, most sellers were also producers; most buyers were also consumers. Our surnames still express the personal roles our ancestors played in the markets they made: Weaver, Potter, Hunter, Smith, Shoemaker, Baker, Fisher, Carpenter.

The first markets were central to culture itself. No setting had a more civilizing effect on our species. The Greek word for market, *agora*, means “gathering place”. The Agora of classical Athens was where citizens gathered to make civilization. Politics, philosophy and democracy—all Greek words—were born in The Agora amidst the noise of commerce: vendors and patrons arguing the merits and prices of good bread and bolts of cloth.

For thousands of years, we knew exactly what markets were, but when industry arrived, we began to forget. We made markets into battlefields for competitors, populations of consumers, targets for messages, collections of numbers, forces with animal natures, economic demons and deities, and verbs for actions done to people rather than with them. Why?

Industry pushed production and consumption far apart, then reconceived business as an activity that begins with factories and ends with consumers, interposing a vast distribution delta between the two. As Alvin Toffler put it in *The Third Wave*, industry drove an “invisible wedge” between production and consumption. By rending the two, this wedge “ripped apart the underlying unity of society, creating a way of life filled with economic tension, social conflict and psychological malaise.”

The movement of materials from production to consumption—from flax to linen and from ore to musket—was a long one, but this distance made room for business at every stage along the way. Thus the wedge Toffler talks about is the “value chain” that runs like a conveyor belt from supply to demand, producer to consumer. For two hundred years, we have been thinking in terms of that chain and the metaphor it requires. In business, that metaphor is shipping.

We literally conceive business in shipping terms. We make *content* that we *address* for *delivery* through *distribution channels*. What we now call *markets* (populations of tastes, demographics or characterizations like eyeballs) are so far removed from their suppliers that we need a new professional concern, marketing, to understand and influence them. To do its work, marketing uses military versions of business' shipping metaphor. It *addresses* goods called *messages*, but deploys them through *campaigns* that are *aimed* or *targetted* to *deliver impact* or obtain *penetration*.

In the Industrial Age, these metaphors made perfect sense, but now that age is ending. It is customary to say we now live in the Information Age, but we have also customarily conceived information as an industrial good. In fact, we betray our industrial thinking when we re-characterize information as content. The popularity of the word “content” betrays a monstrous irony: we live in an age of information, while continuing to think in industrial terms.

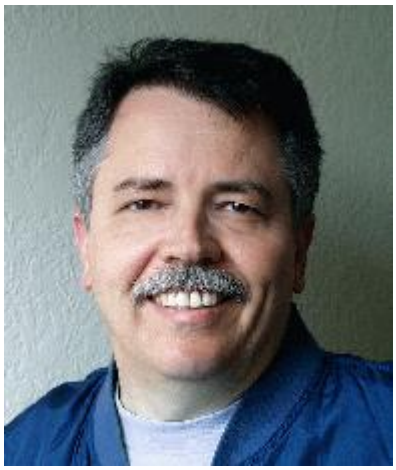
For example, speaking to *PC Week* last year, Lycos CEO Bob Davis said, “We're a media company. We make our money by delivering an audience that people want to pay for.” Note the two different species: audience and people. And note the roles: the audience is cargo; the people are customers. You can see Toffler's wedge sticking out of Davis' head like an axe. Look in the mirror of our own prose, and we see the same axe wedged in our own heads.

With that axe still stuck in our heads, we misconceive what's really going on in the marketplace. We don't see that the Third Wave is not the next stage after the Second Wave, but a return to the First.

The Internet is not just a way to ship content. It is the new agora. It restores markets to what they were in the first place: settings where people can meet, talk about “Stuff that Matters” (thank you, Slashdot), and sell goods to each other. It allows us to reconceive markets as conversations—just as they were before we began to forget, before the long interruption we call the Industrial Age.

What does all this have to do with Linux? It's us—the Linux community—along with like-minded carpenters building the new agora with materials and methods (especially open source) that we create and share. We do it for the same reason our ancestors did it: it's good for everybody's business.

More than any other professional category, open-source developers are demonstrating the truly conversational nature of markets. That means we are the ones in the best position to remember and demonstrate what “market” actually means. It's our job to start yanking those axes out of everybody's heads.



Doc Searls (info@linuxjournal.com) In addition to serving as Senior Editor of *Linux Journal*, is co-author of *The Cluetrain Manifesto*, the web site (http://www.cluetrain.com/) that is now a book by the same name, due out this month from Perseus Books.

Archive Index Issue Table of Contents

Advanced search

Best of Technical Support

Various

Issue #70, February 2000

Our experts answer your technical questions.

Configuring PPP

When I installed PPP for Internet connection, I changed some properties by using the **linuxconf** command in Linux configuration. Now whenever I start Linux, it gives me an error like:

```
starting system
loggers:ypbind[187]:clnt_create for server 127.0.0.1 fasiled starting
NFS Services:rpc.mountd rpc.nfsd YPBINDPROC_DOMAIN: Domain not bound.
```

Please help me to solve this problem. —Kalpesh Vakharia, suryaksh@hotmail.com

You haven't hurt your PPP configuration; you've just activated a tool (Yellow Pages or **yp**) that you haven't fully configured. The error is harmless, but you can remove it by commenting out any entries that point to yp in your boot scripts. —Chad Robinson, Chad.Robinson@brt.com

Your message shows that you enabled yellow pages, but that you have no server to talk to. Run **chkconfig --del ypbind** and you should be fine. —Marc Merlin, merlin@varesearch.com

Crash After Booting

My system just crashed and left me with the following message:

```
checking root filesystems parallelizing fsck version 1.04 (16 may 96)
[/sbin/fsck.ext2] fsck.ext2 -a /dev/sda1 /dev/sda1 contains a file
system with errors check forced Block 4294967295 of inode 131128 >
Blocks (1208304) /dev/sda1: UNEXPECTED INCONSISTENCY; Run fsck manually
an error occurred during the file system check. Dropping you to a shell;
the system will reboot when you leave the shell (repair filesystem) #
```

The system is completely stalled at the moment. What should I do from here? Another person was using the computer when it crashed. What did they do to get it in such a situation? —Nathan Cutter, NCutter@ricegrowers.com.au

Like most modern operating systems, Linux uses write caching. Turning the computer off (it's hard to get it to crash without an actual hardware problem) without properly shutting it down can cause data errors on your hard drive. To solve this problem, you should do exactly what it says: run **fsck** manually on your hard drive. You should do this from a boot disk. The boot disks that you used to install Linux are usually fine for this purpose. Simply run **fsck /dev/sda1** (the partition that is showing the errors). The program will prompt you to fix each error in exactly the same way that—Chad Robinson, Chad.Robinson@brt.com

No, it is not stalled, it just isn't yet booted. You must run fsck by hand (**e2fsck /dev/sda1**) and reply to questions. Most likely, you'll just answer yes to any questions, so you might even add the **-y** switch to e2fsck, although this is considered unsafe. When you are done, exit from the shell (**exit**) and the system will reboot. Since not every file system repair can be performed automatically in a fail-safe way, human intervention can be required when bad errors are detected. I can't tell what caused the problem, but I dare say the most likely reason is some hardware failure (the disk itself or a RAM chip), as fsck found an all-one word (0xffffffff, or 4294967295) where real data was expected. —Alessandro Rubini, rubini@linux.it

Kernel Won't Upgrade

I am trying to upgrade the kernel from 2.2.7 to 2.2.12. I have downloaded the kernel and it compiles fine. I installed the zImage and System.map files where required. However, when I try to boot up again, the kernel version is still 2.2.7. Subsequently, the system tries to load the 2.2.7 modules and not the 2.2.12 modules I require. —Michael Hoegen, m_hoegen@yahoo.com

Copying the image files to the correct disk locations isn't enough. In fact, your system may stop booting shortly. You need to tell your boot loader that you've done this. Linux isn't running yet when your system needs to find that file, so to work around that problem, your boot loader records its physical disk location, and copying one file over another always changes this location. If you are using LILO as your boot loader, you can simply type **lilo** at your shell prompt (as root) to force it to see this file. If you are using another boot loader, consult that program's documentation. —Chad Robinson, Chad.Robinson@brt.com

I don't think you reran /sbin/lilo after copying the kernel, and most likely you didn't even copy the kernel to the right place. If you truly overwrote the kernel

and only forgot to run lilo, your system will stop booting very soon (I won't dig into the technical details here; please check LILO documentation and my article about booting in the June 1997 *LJ*). When upgrading the kernel on a working box, you should always keep a copy of the previous (working) kernel, to recover the computer in case the new kernel image doesn't work for you. To do that, you should add another “image=” stanza to /etc/lilo.conf and rerun lilo. If your lilo.conf is not well-commented, you'll need to refer to proper documentation (such as man lilo.conf). —Alessandro Rubini, rubini@linux.it

Odd Booting Message

When Linux boots, I receive the message “LIL-”. Documentation said that it is a description table error. How can I fix this problem? —Adrian Lasso, alasso@baufest.com

LILO is looking for your kernel and can't find it. Usually this happens in one of two cases. The first is when you install a new kernel and forget to tell LILO it's there by running **lilo** as root before rebooting. The second is when LILO simply can't cope with your hard drive format. You can solve it either way by booting from a set of boot disks and rerunning LILO. If running it alone doesn't help, try running it as **lilo -l**. This often lets LILO work around certain hard drive formats that it otherwise might not be able to read. —Chad Robinson, Chad.Robinson@brt.com

Recovering from such problems is not trivial, as you need an alternate way to boot. I'd suggest you rerun /sbin/lilo after entering your system on booting from CD or floppy. Also, adding a “linear” keyword to the /etc/lilo.conf file is usually beneficial. These problems, however, are usually very hard to track down; you can find a lot of information on architectural problems related to system boot in Andries Brouwer's pages, at http://www.win.tue.nl/~aeb/. —Alessandro Rubini, rubini@linux.it

The Headless Serial Console

I would like to set up a serial console. I would like to be able to issue LILO boot commands on the serial console. Eventually, if all goes well, I'd like the machine to be totally headless. If you could give me a hands-on “how to do this”, it would be great. I have already read your “Serial Terminal as Console” (Issue #36, April 1997) article, but I don't get the LILO prompt on my terminal. —Rick McFarland, mbsrick@ctel.net

This is not a problem at all. If you run kernel 2.2 or 2.3, just configure serial console support and read Documentation/serial-console.txt. If you run 2.0, you must apply the serial-console patch (ftp://ftp.cistron.nl/pub/people/miquels/kernel/) and fall back to the previous case. To use LILO on the serial port, just

add **serial=0,9600n8** or an equivalent line to the /etc/lilo.conf file. In order to interact with LILO, you'll need to send a “break” character. —Alessandro Rubini, rubini@linux.it

Here are the relevant two lines from my lilo.conf file for serial port 1:

```
append="panic=40 console=ttyS0,38400n8 console=tty0"
serial=0,38400n8
```

On my machine, I had to disable hardware flow control for **minicom** to interact with the lilo prompt over the serial port. —Marc Merlin, merlin@varesearch.com

Workstation and Server Connection

I am having problems understanding the concept of setting up a Linux box as a workstation and connecting it to the server. I have set up a server: DOMAIN:server.dungarvin.com IP:192.168.100.1, and a box with DOMAIN:w1.dungarvin.com IP:192.168.100.2; **netmask** is 255.255.255.0 on both machines. My network cards are working, and I can have them ping themselves, but I can't ping one box to the other—I'm stuck there.

Once I do get connectivity, I'd like to run StarOffice from w1 off of the server. I'm wondering how I go about creating a simple network between the two and the concepts involved. —Nick Anderson, neekolai1@hotmail.com

You are slightly confused when you use the DOMAIN term; what you are giving are FQDNs (fully qualified domain names or host names with the domain attached).

Being able to ping yourself doesn't mean much; it can work even if your network is non-functional. You may have a bad cable or some other link-layer problem. You should check the link light on both of your Ethernet cards to make sure it's lit. Maybe you connected the machines directly without using a crossover cable.

To run an application on a remote display, you need to do the following:

```
w1:~$ xhost server
w1:~$ rlogin/telnet server
server:~$ export DISPLAY=w1:0
server:~$ soffice &
```

This is not completely secure, since you're allowing anyone on the server to take over your X display and snoop on your keystrokes. Instead of going into detail about exporting an MIT-MAGIC-COOKIE to allow only one user (you) to access the display, I'd recommend using SSH (secure shell) version 1.2.x (not 2.x

which isn't free), as SSH takes care of X security and the display exporting for you. —Marc Merlin, merlin@varesearch.com

Every computer can ping itself, even if the Ethernet card is not working or not there at all. If you try running **ifconfig** and **route**, you'll be able to see what the problem is (errors transmitting, receiving, or just no route at all). I suspect you have no routing associated to the interfaces; try: **route add -net 192.168.100.0 dev eth0**. —Alessandro Rubini, rubini@linux.it

Archive Index Issue Table of Contents

Advanced search

New Products

Ellen Dahl

Issue #70, February 2000

PerfectBACKUP+ 6.1, Linux Driver for HIPPI 800, Linux by Libranet and more.

PerfectBACKUP+ 6.1

Merlin Softech released its first commercial product, PerfectBACKUP+ 6.1. PerfectBACKUP+ is a Linux-based utility and crash-recovery tool. Users can back up and verify their systems totally unattended. Enhancements include a new Backup Wizard for ease of use, new Search for location of backed-up files, an improved Device Testing Dialog, File Inclusion/Exclusion dialog and remote network backups.

Contact: Merlin Software Technologies, Suite 420, 6450 Roberts Street, Burnaby, BC V5G 4E1, Canada, 888-414-3311, 604-320-7277 (fax), info@merlinsoftech.com, http://www.merlinsoftech.com/.

Linux Driver for HIPPI 800

Essential Communications announced the availability of a Linux driver for its PCI-host bus adapter based on the High Performance Parallel Interface (HIPPI 800) standard. It extends the HIPPI 800 network advantages of very high-speed (800MB/sec) data file transfer capability and low CPU utilization to Linux platforms. HIPPI technology is ideal for organizations requiring the transfer of large amounts of information at high speeds, including video and film archiving, scientific computing, data mining, storage management and transaction processing.

Contact: Essential Communication Corporation, 4239 Balloon Park Road, Albuquerque, NM 87109, 505-344-0080, 505-344-0408 (fax), essential@ods.com, http://www.ods.com/.

Linux by Libranet

Libra Computer Systems announced the release of Linux by Libranet, based on the Debian distribution of Linux, by packaging the most commonly used applications onto an easy-to-install CD. This CD installs over 750MB of software including industry standards such as Netscape. The Libranet release brings Debian to the desktop, making it available to users with little or no Linux experience. Linux by Libranet may be used on any standard PC and will co-exist with Microsoft Windows. Standard hardware components such as sound cards are supported.

Contact: Libra Computer Systems Ltd., 1860 Langworthy Street, North Vancouver, BC V7K 1N8, Canada, http://www.libranet.com/.

Programming Development Kit

Macmillan USA announced the release of *Programming Development Kit: Linux Operating System 6.5*. Users can learn C, Python and Tcl/Tk programming for Linux without having to use a command line. The PDK 6.5 will give Linux programmers an easy, user-friendly way to write programs for the Linux operating system. The software includes essential features such as an IDE optimized for Pentium processors, visual development tools (debugger, build managers and a source editor) and Java, C and C++ libraries.

Contact: Macmillan Computer Publishing, 201 West 103rd Street, Indianapolis, IN 46290, 317-228-4366, orders@mcp.com, http://www.macmillansoftware.com/.

Linux Anti-Virus Solution

DOLFIN.COM now offers anti-virus software for Linux. While Linux is almost immune to virus attacks, it is often used as a file and mail server for Windows clients. DOLFIN assists in providing its clients with the optimum anti-virus solution that meets their business needs, growth targets and budget. DOLFIN's anti-virus solution allows one to scan their FTP upload area as well as e-mail attachments before potential infected files become a problem.

Contact: DOLFIN.COM Inc., 1320 Tower Road, Schaumburg, IL 60173, 847-884-7600, 847-884-7612 (fax), sales@dolfin.com, http://www.dolfin.com/.

OpenDesk.com version 1.0

OpenDesk.com is a free open-source workspace delivered over the Internet. It is 100% open source and works on any computer with a current (version 4.0 or higher) browser, giving resource-strapped small businesses, non-profit and community groups access to the collaboration tools they need to get their work done. Even organizations with members distributed across the globe can come together in this secure virtual workspace. Version 1.0 features include a multi-lingual user interface and compatibility with Macintoshes and PCs running Linux/UNIX or Windows. OpenDesk.com version 1.0 offers e-mail, calendar, contacts, voting/polling, memo editor, bookmarks and customizable themes, with more to come.

Contact: HBE Software, 1030 St-Alexandre, Suite 710, Montréal, Quebec H2Z 1P3, Canada, 514-876-7881, 514-876-9223 (fax), info@hbesoftware.com, http://www.hbesoftware.com/.

UnForm v4.0

Synergetic Data Systems, Inc. announced the release of UnForm v4.0. Significant in this release of UnForm is the ability for users of UNIX applications to integrate electronic documents into their processing and provide access to those documents via web browsers, e-mail or Adobe Acrobat Readers. UnForm is a server-based solution used primarily for laser forms and electronic

document generation. With UnForm, UNIX/Linux sites can enhance existing documents with images, logos and other graphical elements and eliminate the need for preprinted forms.

Contact: Synergetic Data Systems, Inc., 2195 Talon Drive, Latrobe, CA 95682, 800-446-7374, 530-672-9975 (fax), sdsi@synergetic-data.com, http://synergetic-data.com/.

Max for Linux

PlugSys International announced a new compiler, Max for Linux, a development tool for compiling and running Xbase code under the major Linux distributions for Intel processors. Max for Linux compiles and runs character-based applications. It is a CGI-like Xbase engine for NT- and Linux-based web servers and was written using today's C++ in a 32-bit code base.

Contact: PlugSys International LLC, 1636 Graff Avenue, San Leandro, CA 94577, info@plugsys.com, http://www.plugsys.com/.

PizzaBox Linux Distribution

KYZO released the commercial version of its PizzaBox Linux distribution, where a prototype server was built in a Pizza Hut take-out box. The commercial version is the same basic software as the free version, but is shipped pre-installed in a bootable Flash-ROM and comes with the circuit board needed to make it boot in any 486 or above PC. The system includes file, print and CD sharing, remote access, UPS monitoring, tape backup, hardware monitoring and APM. It will automatically accept both SCSI and IDE hard drives and comes with a sophisticated, JavaScript-enabled, web management interface.

Contact: KYZO Ltd, Little Streams, The Abbotsbrook, Bourne End, Bucks. SL8 5QY, United Kingdom, +44-0-1628-526886, +44-0-1628-526030 (fax), sales@jrcs.co.uk, http://www.kyzo.com/.

Appgen Linux Java Client and PowerWindows Applications

Appgen Business Software announced the release of their Linux Java Client. All the benefits of GUI in a pure Linux environment, with Appgen's eleven general business and accounting applications, are available on a CD-ROM. Appgen's applications run natively on Linux. The applications are also the first accounting applications to be validated for IBM Netfinity servers running Linux. The PowerWindows Applications are shipped to VARs with source code.

Contact: Appgen Business Software, Inc., 1300 Veterans Memorial Highway, Hauppauge, NY 11788, 800-231-0062, 631-471-3291 (fax), info@appgen.com, http://www.appgen.com/.

IVR Server

Open Source Telecom announced their IVR Server, the first open-source interactive voice response system. The initial IVR Server solution automates Internet service provider support functions with telephony interface boards from Pika Technologies. It is a web-enabled interactive voice response platform with templates for common business functions, based on the Adjunct Communication Server (ACS) and Debian GNU/Linux.

Contact: Open Source Telecom, 1030 Maude Avenue, Suite 511, Sunnyvale, CA 94086, 650-964-4678, services@ostel.com, http://www.ostel.com/.

Photogenics for Linux

Paul Nolan Ltd. announced that the graphics/art package Photogenics will soon be available for Linux. It was first released on the Amiga five years ago. Photogenics allows one to modify existing pictures or create new images from scratch. It can be used for a multitude of purposes from simple file conversion to advanced photo manipulation and retouching. It supports many different media, including chalk, pencil, watercolors and the airbrush.

Contact: Paul Nolan Ltd., MB #407, 5663 Balboa Avenue, San Diego, CA 92111, 619-839-3803, 619-839-3803 (fax), pnolan@cix.co.uk, http://www.paulnolan.com/.

Contact Information

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

T/TCP: TCP for Transactions

Mark Stacey

Ivan Griffin

John Nelson

Issue #70, February 2000

A discussion of the operation, advantages and flaws of an experimental extension for the TCP protocol.

T/TCP is an experimental extension for the TCP protocol. It was designed to address the need for a transaction-based transport protocol in the TCP/IP stack.

TCP and UDP are the current choices available for transaction-based applications. TCP is reliable but inefficient for transactions, whereas UDP is unreliable but highly efficient. T/TCP sits between these two protocols, making it an alternative for certain applications.

Currently, several flavours of UNIX support T/TCP. SunOS 4.1.3 (a Berkeley-derived kernel) was the very first implementation of T/TCP, and made available in September 1994. The next implementation was for FreeBSD 2.0, released in March 1995. For my final-year project, I implemented T/TCP for Linux at the University of Limerick in April 1998. The source code is available at http://www.csn.ul.ie/~heathclf/fyp/.

In this article, I discuss the operation, advantages and flaws of T/TCP. This will allow application developers to decide when T/TCP is appropriate for networking applications. I present my results from a comparative analysis between T/TCP and TCP, based on the number of packets per session for each transaction. I also give my conclusions from a case study I conducted into the possible impact of T/TCP on the World Wide Web.

Introduction

The TCP/IP reference model is a specification for a networking stack on a computer. It exists to provide a common ground for network developers. This allows easier interconnection of the different vendor-supplied networks.

The most popular implementations of the transport layer in the reference model are Transmission Control Protocol (TCP), a connection-oriented protocol, and User Datagram Protocol (UDP), a connectionless protocol.

Both of these protocols have advantages and disadvantages. The two main aspects of the protocols make them useful in different areas. Being a connectionless protocol, UDP is unreliable but fast and useful for applications, such as DNS (Domain Name System), where speed is preferred over reliability. TCP, on the other hand is a reliable, connection-oriented protocol. As a result, TCP is a slower protocol than UDP.

With the explosion of the Internet in recent years, the need for a new specification arose. The current transport protocols were either too verbose or not reliable enough. A protocol was needed that was faster than TCP but more reliable than UDP. This new protocol could reduce bandwidth and increase the speed of transmission of data, which is very much needed at the moment.

TCP for Transactions (T/TCP) is a possible successor to both TCP and UDP. It is a transaction-oriented protocol based on a minimum transfer of segments, so it does not have the speed problems associated with TCP. By building on TCP, it does not have the unreliability problems associated with UDP. With this in mind, RFC1379 was published in November 1992. It discussed the concepts involved in extending the TCP protocol to allow for a transaction-oriented service. Some of the main points the RFC discussed were bypassing the three-way handshake and shortening the TIME-WAIT state from 240 seconds to 12 seconds. Eighteen months later, RFC1644 was published, with the specification for Transaction TCP. T/TCP cuts out much unnecessary handshaking and error detection done by the current TCP protocol, and as a result, increases the speed of connection and reduces the necessary bandwidth.

Transaction Transmission Control Protocol

T/TCP can be considered a superset of the TCP protocol. The reason for this is that T/TCP is designed to work with current TCP machines seamlessly. What follows is a brief description of T/TCP and how it differs from the current TCP standard in operation.

What is a Transaction?

The term “transaction” refers to the request sent by a client to a server, along with the server's reply. RFC955 lists some of the common characteristics of transaction processing applications:

- Asymmetrical Model: the two end points take different roles; this is a typical client-server role where the client requests the data and the server responds.
- Short Duration: normally, a transaction runs for a short time span.
- Few Data Packets: each transaction is a request for a small piece of information, rather than a large transfer of information both ways.

Background to T/TCP

The growth of the Internet has put a strain on the bandwidth and speed of networks. There are now more users than ever, and a more efficient form of data transfer is needed.

The absolute minimum number of packets required in a transaction is two: one request followed by one response. UDP is the one protocol in the TCP/IP protocol stack that allows this, but the problem is the unreliability of the transmission.

T/TCP has the reliability of TCP and comes very close to realizing the 2-packet exchange (three in fact). T/TCP uses the TCP state model for its timing and retransmission of data, but introduces a new mechanism to allow the reduction in packets.

Even though three packets are sent using T/TCP, the data is carried on the first two, thus allowing the applications to see the data with the same speed as UDP. The third packet is the acknowledgment to the server by the client that it has received the data, which is how the TCP reliability is incorporated.

Basic Operation

Consider a DNS system, one where a client sends a request to a server and expects a small amount of data in return. A diagram of the transaction can be seen in Figure 1. This diagram is very similar to a UDP request with a saving of 66% in packets transferred compared to TCP. Obviously, in cases where a large amount of data is being transferred, there will be more packets transmitted and thus a decrease in the percentage saved.

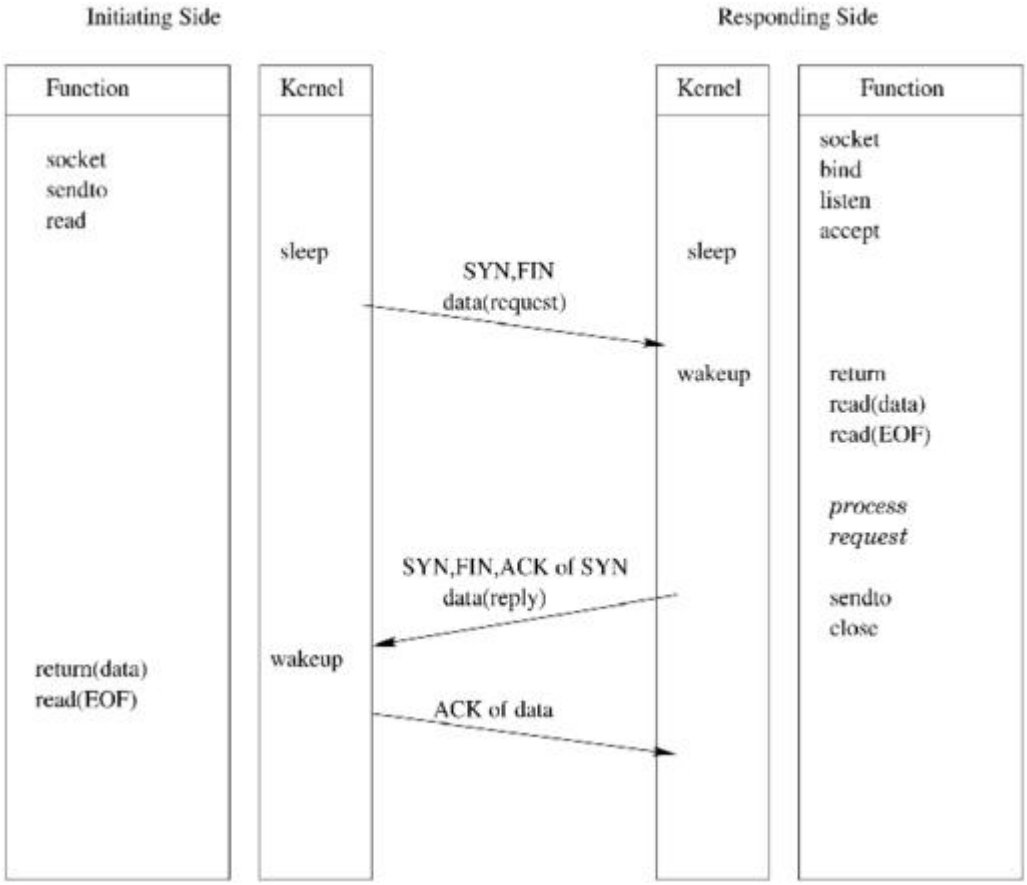


Figure 1. Time line of T/TCP Client-Server Transaction

Timing experiments have shown that there is a slightly longer time required for T/TCP than for UDP, but this is a result of the speed of the computer and not the network. As computers get more powerful, the performance of T/TCP will approach that of UDP.

TCP Accelerated Open

TCP Accelerated Open (TAO) is a mechanism introduced by T/TCP designed to cut down on the number of packets needed to establish a connection with a host.

There are a number of new options that T/TCP introduces. These options allow the establishment of a connection with a host using the TAO. T/TCP uses a 32-bit incarnation number, called a connection count (CC). This option is carried in the options part of a T/TCP segment (see Figure 2). A distinct CC value is assigned to each direction of an open connection. Incremental CC values are assigned to each connection that a host establishes, either actively or passively.

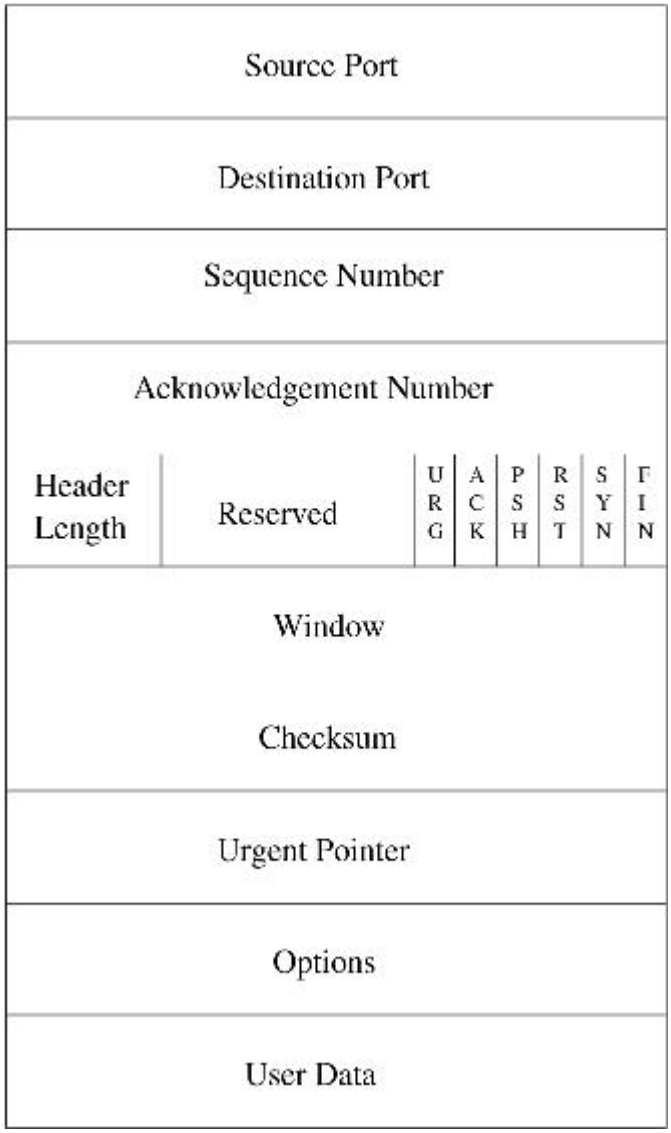


Figure 2. TCP Header

The three-way handshake is bypassed using the CC value. Each server host caches the last valid CC value it received from each different client host. This CC value is sent with the initial SYN segment to the server. If the initial CC value for a particular client host is larger than the corresponding cached value, the property of the CC options (the increasing numbers) ensures that the SYN segment is new and can be accepted immediately.

The TAO test fails if the CC option arriving in the SYN segment is smaller than the last CC value received that was cached by the host, or if a CCnew option is sent. The server then initiates a three-way handshake in the normal TCP/IP fashion.

Examples

T/TCP can be beneficial to some of the applications that currently use TCP or UDP. At the moment, many applications are transaction-based rather than connection-based, but still have to rely on TCP, along with the overhead. UDP is

the other alternative, but not having time-outs and retransmissions built into the protocol means the application programmers have to supply the time-outs and reliability checking themselves. Since T/TCP is transaction-based, there is no set-up and shutdown time, so the data can be passed to the process with minimal delay.

HTTP and RPC

Hyper Text Transfer Protocol is the method used by the World Wide Web to access web pages. T/TCP can be used to reduce the number of packets required.

With TCP, the transaction is accomplished by connecting to the server (three-way handshake), requesting the file (**GET *file***), then closing the connection (sending a FIN segment). T/TCP would operate by connecting to the server, requesting the document and closing the connection, all in one segment (TAO). It is obvious that bandwidth is saved by this method.

Remote procedure calls (RPCs) also adhere to the transaction style paradigm. A client sends a request to a server for the server to run a function. The results of the function are then returned in the reply to the client. There is only a tiny amount of data transferred with RPCs.

DNS

The Domain Name System is used to resolve host names into the IP addresses that locate the host.

To resolve a domain name, the client sends a request with the IP address or a host name to the server. The server responds with the host name or IP address where appropriate. This protocol uses UDP.

As a result of using UDP, the process is fast, but not reliable. Furthermore, if the response by the server exceeds 512 bytes of data, it sends the data back to the client with the first 512 bytes and a truncated flag. The client has to resubmit the request using TCP.

The reason for this is there is no guarantee that the receiving host will be able to reassemble an IP datagram exceeding 576 bytes. For safety, many protocols limit the user data to 512 bytes.

T/TCP is the perfect candidate for the DNS protocol. It can communicate at speeds approaching that of UDP, and it has the reliability of TCP.

Testing and Analysis

In order to investigate the benefits of this implementation of T/TCP, it is important to test its operation, and also to compare its operation to the original TCP/IP operation. I performed these tests using the Linux 2.0.32 kernel with T/TCP modifications and FreeBSD version 2.2.5, which already implements T/TCP.

Performance Analysis

To investigate the performance of T/TCP in comparison to the original TCP/IP, I compiled a number of executables that returned different-sized data to the client. The two hosts involved were elendil.ul.ie, running Linux, and devilwood.ece.ul.ie, running FreeBSD 2.2.5. The tests were performed for ten different response sizes in order to vary the number of segments required to return the full response. Each request was sent 50 times and the results were averaged. The maximum segment size in each case is 1460 bytes.

The metric used for performance evaluation was the average number of segments per transaction. I used **Tcpdump** to examine the packets exchanged. Note that Tcpdump is not entirely accurate, since during fast packet exchanges, it tends to drop some packets to keep up. This accounts for some discrepancies in the results.

Number of Packets per Transaction

Figure 3 shows the results from testing for the number of segments for T/TCP versus the number of segments for normal TCP/IP. It is immediately obvious that there is a savings of an average of five packets. These five packets are accounted for in the three-way handshake, and the packets sent to close a connection. Lost packets and retransmissions cause discrepancies in the path of the graph.

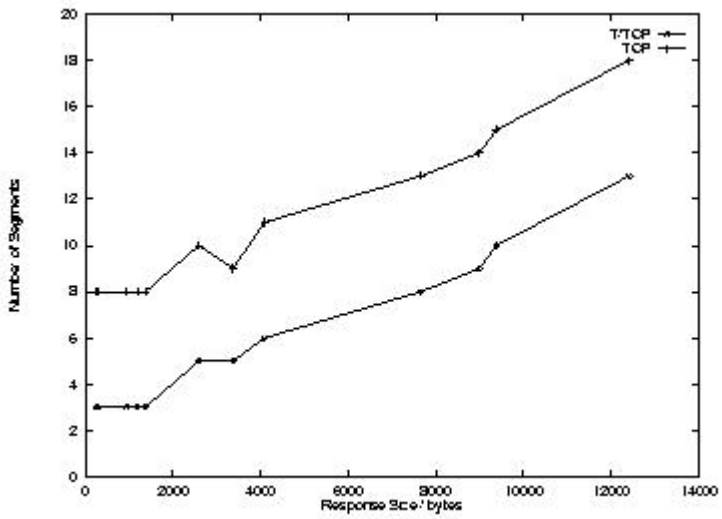


Figure 3. Number of Segments versus Size of Data Transfer

One interesting point about the average number of segments when using a TCP client and a T/TCP server is that there is still a saving of one segment. A normal TCP transaction requires nine segments, but because the server was using T/TCP, the FIN segment was piggybacked on the final data segment, reducing the number of segments by one. This demonstrates a reduction in segments, even if only one side is T/TCP-aware.

Figure 4 shows the percentage savings for the different packet sizes. The number of packets saved remains fairly constant, but because of the increase in the number of packets being exchanged, there is a decrease in the overall savings. This indicates that T/TCP is more beneficial to small data exchanges.

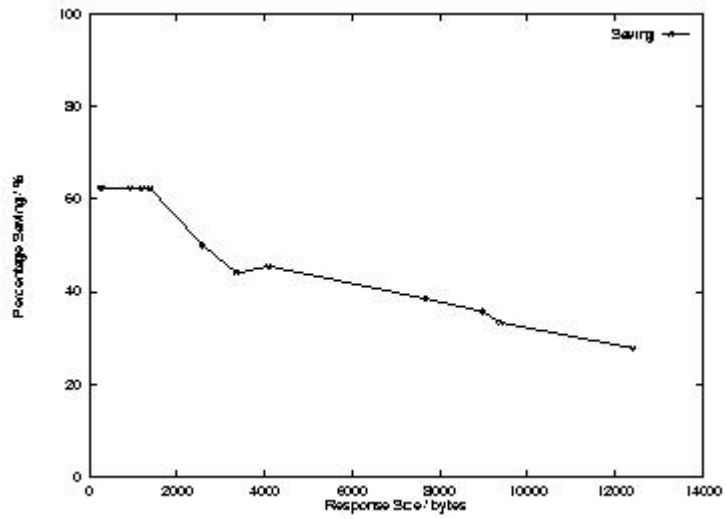


Figure 4. Percentage Savings per Size of Data Transfer

Memory Issues

The main memory drain in the implementation is in the routing table. In Linux, for every computer with which the host comes into contact, an entry for the foreign host is made in the routing table. This applies to both a direct connection and to a multi-hop route. This routing table is accessed through the rtable structure. The implementation of T/TCP adds two new fields to this structure, CCrecv and CCsent.

The entire size of this structure is 56 bytes, which isn't a major memory hog on a small stand-alone host. On a busy server, though, where the host communicates with maybe thousands of other hosts an hour, it can be a major strain on memory. Linux has a mechanism where a route no longer in use can be removed from memory. A check is run periodically to clean out unused routes and those that have been idle for a time.

The problem here is the routing table holds the TAO cache. So anytime a route which contains the last CC value from a host is deleted, the local host has to re-initiate the three-way handshake with a CCnew segment.

The benefits of leaving the routing entries up permanently are clear. The most likely use would be in a situation where a host talks to only a certain set of foreign hosts and denies access to unknown hosts. In this case, it is advantageous to keep a permanent record in memory so the three-way handshake can be bypassed more often.

Protocol Analysis

The original protocol specification (RFC1644) labeled T/TCP as an experimental protocol. Since the RFC was published, there hasn't been an update to the protocol to fix some of the problems. From the previous sections, the benefits over the original TCP protocol are obvious, but is it a case of the disadvantages outweighing the advantages?

One of the more serious problems with T/TCP is that it opens up the host to certain Denial of Service attacks. SYN flooding (see www.sun.ch/SunService/technology/bulletin/bulletin963.html for more information) is the term given to a form of denial-of-service attack where the attacker continually sends SYN packets to a host. The host creates a sock structure for each SYN, thus reducing the number of sock structures that can be made available to legitimate users. This may eventually result in the host crashing when enough memory has been used. SYN cookies were implemented in the Linux kernel to combat this attack. SYN cookies cause problems with T/TCP, as there are no TCP options sent in the cookie, and any data arriving in the initial SYN can't be used immediately. The CC option in T/TCP does provide some protection on its own, but it is not secure enough.

Another serious problem discovered during research was that attackers can bypass rlogin authentication. An attacker creates a packet with a false IP address in it, one that is known to the destination host. When the packet is sent, the CC options allow the packet to be accepted immediately and the data passed on. The destination host then sends a SYNACK to the original IP address. When this SYNACK arrives, the original host sends a reset, as it is not in a SYN-SENT state. This happens too late, as the command will already have been executed on the destination host. Any protocol that uses an IP address as authentication is open to this sort of attack. (See geek-girl.com/bugtraq/1998_2/0020.html.)

It should be noted, however, that the use of T/TCP in conjunction with protocols such as HTTP have fewer security problems, due to the inability to run any server commands with HTTP.

RFC1644 also has a duplicate transaction problem. This can be serious for applications that are non-idempotent (repeat transactions are very undesirable). This error can occur in T/TCP if a request is sent to a server and the server processes the transaction, but before it sends back an acknowledgment, the process crashes. The client side times out and retransmits the request; if the server process recovers in time, it can repeat the same transaction. This problem occurs because the data in a SYN can be immediately passed to the process, rather than in TCP where the three-way handshake has to be completed before data can be used. The use of two-phase commits and transaction logging can eliminate this problem.

Case Study: T/TCP Performance over Suggested HTTP Improvements

With the World Wide Web being the prime example of client-server transaction processing today, this section will focus on the benefits of T/TCP to the performance of the Web.

Currently, the HTTP protocol sits in the application layer of the TCP/IP reference model. It uses the TCP protocol to carry out all its operations, UDP being too unreliable. Much latency is involved in the transfer of information, i.e., the three-way handshake and explicit shutdown exchanges.

Web Document Characteristics

In a survey of 2.6 million web documents searched by the Inktomi web crawler search engine (see http://inktomi.berkeley.edu/), it was found that the mean document size on the Web was 4.4KB, the median size was 2.0KB and the maximum size encountered was 1.6MB.

Referring to Figure 4, it can be seen that the lower the segment size, the better the performance of T/TCP over normal TCP/IP. With a mean document size of 4.4KB, this results in an average savings of just over 55% in the number of packets. When taking the median size into account, there is a savings of approximately 60%.

Suggested Performance Improvements for HTTP

At the moment, all web pages are transferred in plaintext form, requiring little work from either the server side or the client side to display the pages.

Compression

In their paper “Network Performance Effects of HTTP/1.1, CSSI and PNG” (see Resources), the authors investigated the effect of introducing compression to the HTTP protocol. They found that compression resulted in a 64% savings in

the speed of downloading, with a 68% decrease in the number of packets required. Over normal TCP/IP, this brings the packet exchanges and size of data down to the level where T/TCP becomes beneficial. Thus, a strategy involving both compression and T/TCP can result in enormous savings in time and bandwidth.

Delta Encoding

In this situation, a delta refers to the difference between two files. On UNIX systems, the **diff** command can be used to generate the delta between two files. Using the changed file and the delta, the original file can be regenerated again and vice versa.

J. C. Mogul, et al. (see Resources) investigated the effect that delta encoding has on the Web. In their testing, they not only used delta encoding, they also compressed the delta generated to further reduce the amount of information transferred. They discovered that by using the **vdelta** delta generator and compression, they could achieve up to 83% savings in the transmission of data.

If this method was used with T/TCP, there could be as much as a further 66% savings in packets transferred. This is a total of 94% reduction in packet transfer.

It should be noted, however, that this is a best-case scenario. In this situation, the document will already have been cached on both the server and the client side, and the client and server will previously have completed the three-way handshake in order to facilitate the TAO tests.

Socket Programming under T/TCP

Programming for T/TCP is slightly different using socket programming. As an example, the chain of system calls to implement a TCP client would be as follows:

- **socket**: create a socket.
- **connect**: connect to the remote host.
- **write**: write data to the remote host.
- **shutdown**: close one side of the connection.

Whereas with T/TCP, the chain of commands would be:

- **socket**: create a socket.

- **sendto**: connect, send data and close connection. The sendto function must be able to use a new flag **MSG_EOF** to indicate to the kernel that it has no more data to send on this connection.

Programming under T/TCP is much like programming under UDP.

Conclusion

Analysis of T/TCP shows that it benefits small, transaction-oriented transfers more than large-scale information transfers. Aspects of transactions can be seen in such cases as the World Wide Web, Remote Procedure Calls and DNS. These applications can benefit from the use of T/TCP in efficiency and speed. T/TCP reduces on average both the numbers of segments involved in a transaction and the time taken to complete the transaction.

As T/TCP is still an experimental protocol, there are problems that need to be addressed. Security problems encountered include the vulnerability to SYN flood attacks and rlogin authentication bypassing. Operational problems include the possibility of duplicate transactions occurring. Problems that occur less frequently would be the wrapping of the CC values on high-speed connections, thus opening up a destination host to accepting segments on the wrong connection.

Many people recognize the need for a protocol that favors transaction-style processing and are willing to accept T/TCP as the answer. Security considerations lead to the conclusion that T/TCP would be more useful in a controlled environment, one where there is little danger from a would-be attacker who can exploit the weaknesses of the standard. Examples of enclosed environments would be company intranets and networks protected by firewalls. With many companies seeing the Web as the future of doing business, internal and external, a system employing T/TCP and some of the improvements to HTTP, such as compression and delta encoding, would result in a dramatic improvement in speed within a company intranet.

Where programmers are willing to accept T/TCP as a solution to their applications, only minor modifications are needed for the application to become T/TCP-aware. For client-side programming, it involves the elimination of the connect and shutdown function calls, which can be replaced by adding the **MSG_EOF** flag to the sendto command. Server-side modifications involve simply adding the **MSG_EOF** flag to the send function.

Research into T/TCP suggests it is a protocol that is nearly, but not quite, ready to take over transaction processing for general usage. For T/TCP alone, more work needs to be done to develop it further and solve the security and operational problems. Security problems can be solved using other

authentication protocols such as Kerberos and the authentication facilities of IPv6. Operational problems can be dealt with by building greater transaction reliability into the applications that will use T/TCP, such as two-phase commits and transaction logs.

Applications can be easily modified to use T/TCP when available. Any applications which involve an open-close connection can use T/TCP efficiently, and the more prominent examples would be web browsers, web servers and DNS client-server applications. To a smaller extent, applications such as **time**, **finger** and **whois** can benefit from T/TCP as well. Many networking utilities are available that can take advantage of the efficiency of the protocol. All that is needed is the incentive to do it.

Perhaps a more immediate task, though, is to port the T/TCP code to the new Linux kernel series, 2.3.*x*.

Resources

Mark Stacey (Mark.Stacey@icl.ie) graduated from the University of Limerick, Ireland, in 1998 with a first class honors degree in Computer Engineering. His interests include Java programming and Web development. He currently works for ICL in the Information Technology Center based in Dublin, Ireland.

Ivan Griffin

John Nelson

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

POSIX Thread Libraries

Felix Garcia

Javier Fernandez

Issue #70, February 2000

The authors have studied five libraries that can be used for multi-thread applications and herein present the results.

Recent years have seen an increase in the popularity of threads, because there are many applications in which threads are useful. In many aspects, threads operate in the same manner as processes, but can execute more efficiently. All modern operating systems today include some kind of support for thread management. Moreover, threads have been standardized by the IEEE Technical Committee on Operating Systems. This standard allows users to write portable multi-thread programs.

As with other operating systems, Linux includes multi-threading capability, and some multi-threading libraries are available for Linux. We will describe a comparative study of five threads packages for Linux: CLthreads, LinuxThreads, FSU Pthreads, PC threads and Provenzano Pthreads. All libraries evaluated make use of POSIX-compliant functionality. The main objective of this study is to evaluate and compare the performance of some multi-thread features to analyze the suitability of these libraries for use in multi-thread applications. Also, we make a comparison with Solaris threads.

The Notion of Threads

A thread is an independent flow of control within a process. A traditional UNIX process has a single thread that has sole possession of the process' memory and other resources. Threads within the same process share global data (global variables, files, etc.), but each thread has its own stack, local variables and program counter. Threads are referred to as lightweight processes, because their context is smaller than the context of a process. This feature makes

context switches between threads cheaper than context switches between traditional processes.

Threads are useful for improving application performance. A program with only one thread of control must wait each time it requests a service from the operating system. Using more than one thread lets a process overlap processing with one or more I/O requests (see Figure 1). In multiprocessor machines, multiple threads are an efficient way for application developers to utilize the parallelism of the hardware.

This feature is especially important for client/server applications. Server programs in client/server applications may get multiple requests from independent clients simultaneously. If the server has only one thread of control, client requests must be served in a serial fashion. Using multi-thread servers, when a client attempts to connect to the server, a new thread can be created to manage the request.

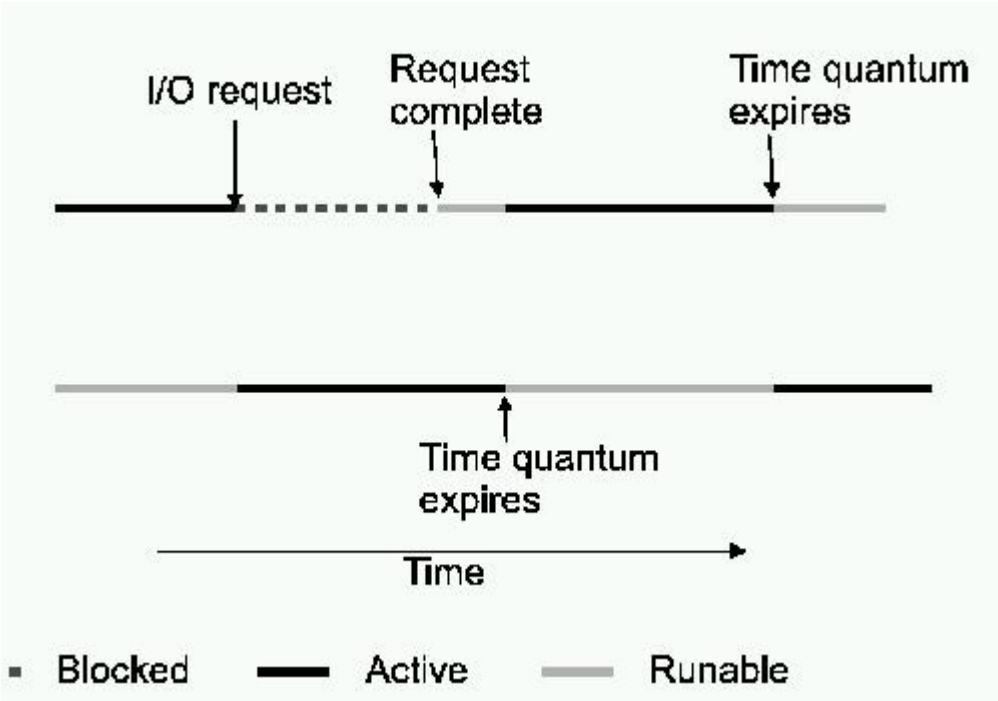


Figure 1. Overlapping Processing and I/O Requests

In general, multi-threading capabilities are of great benefit to certain classes of applications, typically server or parallel processing applications, allowing them to make significant performance gains on multiprocessor hardware, increase application throughput even on uniprocessor hardware, and make efficient use of system resources. Threads, however, are not appropriate for all programs. For example, an application that must accelerate a single compute-bound algorithm will not benefit from multi-threading when the program is executed on uniprocessor hardware.

There are two traditional models of thread control: user-level threads and kernel-level threads.

User-level thread packages usually run on top of an existing operating system. The threads within the process are invisible to the kernel. Threads are scheduled by a runtime system which is part of the process code. Switching between user-level threads can be done independently of the operating system. User-level threads, however, have a problem: when a thread becomes blocked while making a system call, all other threads within the process must wait until the system call returns. This restriction limits the ability to use the parallelism provided by multiprocessor platforms.

Kernel-level threads are supported by the kernel. The kernel is aware of each thread as a scheduled entity. In this case, a set of system calls similar to those for processes is provided, and the threads are scheduled by the kernel. Kernel threads can take advantage of multiple processors; however, switching among threads is more time-consuming because the kernel is involved.

There are also *hybrid models*, supporting user-level and kernel-level threads. This gives the advantages of both models to the running process. Solaris offers this kind of model.

POSIX Threads

Threads have been standardized by the IEEE Technical Committee on Operating Systems. The base for the POSIX standard (POSIX 1003.1), *The Portable Operating Systems Interface*, defines an application program interface which is derived from UNIX but may as well be provided by any other operating system. This standard includes a set of *Thread Extensions* (POSIX 1003.1c). These thread extensions provide the base standard with interfaces and functionality to support multiple flows of control within a process. The facilities provided represent a small set of syntactic and semantic extensions to POSIX 1003.1 in order to support a convenient interface for multi-threading functions.

The interfaces in this standard are specifically targeted at supporting tightly coupled multitasking environments, including multiprocessor systems and advanced language constructs. The specific functional areas covered by this standard and their scopes include:

- **Thread management**: creation, control and termination of multiple flows of control in the same process under the assumption of a common shared address space.

- **Synchronization primitives**: mutual exclusion and condition variables, optimized for tightly coupled operation of multiple control flows within a process.
- **Harmonization**: with the existing POSIX 1003.1 interfaces.

Linux Threading

Multi-threading capability is included in the Linux 2.0 kernel. There is an ongoing effort to refine this and make the kernel more reentrant. The multi-thread capability is offered by the **clone** system call, which creates a new *context of execution*. This call can be used to create a new process, a new thread or one of a range of possibilities which doesn't fit into either of these categories. The **fork** system call is actually a call to clone with a specific set of values as parameters, and the **pthread_create** function call could be a call to clone with a different set of values as parameters.

The clone system call has several flags to indicate how much will be shared between threads. Figure 2 lists each flag.

Flag	Description
CLONE_VM	Share data and stack
CLONE_FS	Share file system info
CLONE_FILES	Share open files
CLONE_SIGHAND	Share signals
CLONE_PID	Share PID parent

Figure 2. Flags of the clone System Call

Threading capabilities are given by different threads packages. Several threading libraries are available for Linux. All libraries referred to in this article make use of POSIX-compliant functionality; however, at the time of this writing, there are no fully POSIX-compliant multi-threading libraries available for Linux.

The libraries we have evaluated are the following:

- **Provenzano threads (PT)**: this package offers a user-level POSIX threads library with thread-blocking system calls (**read**, **write**, **connect**, **sleep**, **wait**, etc.) and a thread-safe C library (stdio, network utilities, etc.). This implementation currently supports basic functionality, synchronization primitives, thread-specific data and thread attributes.
- **FSU_Pthreads (FSUT)**: this is a C library which implements user-level POSIX threads for different operating systems: Solaris 2.x, SCO UNIX, FreeBSD, Linux and DOS. This implementation supports thread management,

synchronization, thread-specific data, thread priority scheduling, signals and cancellation.

- **PC threads (PCT)**: this is a user-level POSIX threads library that includes non-blocking select, read and write. This library has runtime configurable parameters, such as clock interrupt interval, default thread-scheduling policy, default thread-stack size and the I/O polling interval, among others.
- **CLthreads (CLT)**. CLthreads is a kernel-level POSIX-compliant library on top of Linux. CLthreads uses the clone system call to take full advantage of multiprocessor systems.
- **LinuxThreads (LT)**. LinuxThreads is an implementation of POSIX threads for Linux. LinuxThreads provides kernel-level threads: they are created using the clone system call, and all scheduling is done in the kernel. This approach can take full advantage of multiprocessor systems. It also results in a simpler, more robust thread library, especially for blocking system calls.

Solaris Threading

Solaris supports a hybrid model of threads—user-level and kernel-level threads—and a POSIX-compliant library. User-level threads are supported by the library for their creation and scheduling, and the kernel is not aware of these threads. Solaris defines an intermediate level of threads as well, the lightweight process (LWP). LWPs are between user-level threads and kernel-level threads (see Figure 3). LWPs are manipulated by the thread library. The user-level threads are multiplexed onto the LWPs of the process (each process contains at least one LWP), and only threads currently connected to LWPs accomplish work. The rest are either blocked or waiting for an LWP on which they can run.

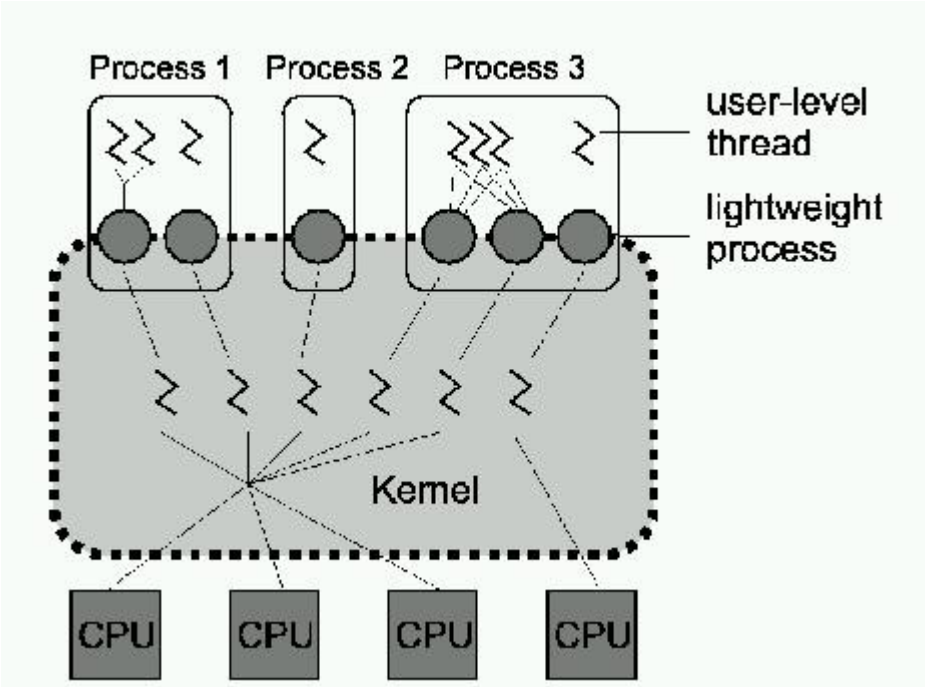


Figure 3. Threads in Solaris

There is a kernel-level thread for each LWP, and some kernel-level threads run on the kernel's behalf and have no associated LWP. Kernel-level threads are the only objects scheduled within the system.

With this model, any process may have many user-level threads. These user-level threads may be scheduled and switched among kernel-supported lightweight processes without the intervention of the kernel. Each LWP is connected exactly to one kernel-level thread. Many LWPs are in a process, but they are needed only when threads need to communicate with the kernel; if one blocks, the others can continue to execute within the process. In Solaris, users can create new threads permanently bound to an LWP.

Figure 4 summarizes the user-level and kernel-level features of the evaluated libraries.

ds	Solaris	PT	FSUT	PCT	CLT	LT
evel	√	√	√	√		
l-level	√				√	√

Figure 4. Kinds of Libraries

Performance

This section describes performance metrics used to evaluate the POSIX-thread libraries, the results obtained for all, and the hardware platform used in the evaluation.

Performance Metrics

Performance metrics are an essential element in the evaluation and use of a system. To this end, a set of performance metrics was defined to evaluate and compare the two main features of all POSIX thread libraries: thread management and synchronization management.

Thread Management

The thread management metrics were aimed at evaluating the efficiency of the creation and termination of threads. These metrics are:

- **Thread creation**: measurement for thread creation time, e.g., time to perform the **pthread_create** operation.
- **Join a thread**: time needed to perform the **pthread_join** operation on a terminated thread.

- **Thread execution**: time needed to execute the first instruction of a thread. This time includes the thread creation time and the time to perform a **seched_yield** operation, as shown below:

```
thread_1()
{ . . .
start_time();
pthread_create(...);
sched_yield();
. . . }
thread_2()
{
end_time();
... }
```

- **Thread termination**: time interval from when a **pthread_exit** operation is performed until the pthread_join operation on this thread is finalized, as shown below:

```
thread_1()
{ . . .
start_time();
pthread_exit(...);
. . . }
thread_2()
{ . . .
pthread_join();
end_time();
... }
```

- **Thread creation versus process creation**: compares the time needed to create a process with the time needed to create a thread within a process.
- **Join a thread versus wait for a process**: compares the time needed to perform a wait operation on a finalized process with the time needed to perform a pthread_join operation on a finalized thread.
- **Granularity of parallelism**: this is the minimum number of iterations of a null loop to be executed in *n* threads simultaneously before the time needed by the *n* threads is less than the time needed for a single thread to execute the total number of iterations by itself. The time for the *n* thread case has to include the time to create all *n* threads and wait for them to terminate. This number can be used by a programmer to determine when it might be advantageous to divide a task into *n* different pieces which can be executed simultaneously. This metric is provided for *n*, with *n* being the number of processors on the machine.

Synchronization Management

These metrics are concentrated in the mutex and condition-variables operations performance.

- **Mutex init**: time interval needed to perform the **pthread_mutex_init**.
- **Mutex lock**: time interval needed to perform the **pthread_mutex_lock** on a free mutex.

- **Mutex unlock**: time interval needed to perform the **pthread_mutex_unlock**.
- **Mutex lock/unlock with no contention**: time interval needed to call pthread_mutex_lock followed immediately by pthread_mutex_unlock on a mutex that is being used only by the thread doing the test. This test is shown below:

```
thread()
{ . . .
start_time();
pthread_mutex_lock(...);
pthread_mutex_unlock(...);
end_time();
. . . }
```

- **Mutex destruction**: time needed to perform the **pthread_mutex_destroy** operation.
- **Condition init**: time interval needed to perform the **pthread_cond_init**.
- **Condition destroy**: time needed to perform the **pthread_cond_destroy** operation.
- **Synchronization time**: measures the time it takes for two threads to synchronize with each other using two condition variables, as shown below:

```
thread_1()
{ . . .
start_time();
pthread_cond_wait(c1,...);
pthread_cond_signal(c2);
end_time();
. . . }
thread_2()
{ . . .
pthread_cond_signal(c2);
pthread_cond_wait(c1,...);
. . . }
```

- **Mutex lock/unlock with contention**: time interval between when one thread calls pthread_mutex_unlock and another thread that was blocked on pthread_mutex_lock returns with the lock held.

```
thread_1()
{ . . .
pthread_mutex_lock(...);
start_time();
pthread_mutex_unlock(...);
. . . }
thread_2()
{ . . .
pthread_mutex_lock(...); < Blocked >
end_time();
pthread_mutex_unlock(...);
. . . }
```

- **Condition variable signal/broadcast with no waiters**: time needed to execute **pthread_cond_signal** and **pthread_cond_broadcast** if there are no threads blocked on the condition.

- **Condition variable wake up**: time from when one thread calls pthread_cond_signal and a thread blocked on that condition variable returns from its pthread_cond_wait call. The condition and its associated mutex should not be used by any other thread.

```
thread_1()
{ . . .
pthread_mutex_lock(...);
start_time();
pthread_cond_signal(...);
pthread_mutex_unlock(...);
. . . }
thread_2()
{ . . .
pthread_mutex_lock(...);
pthread_cond_wait(...);
end_time();
pthread_mutex_unlock(...);
. . . }
```

Performance Results

All results were obtained by running the benchmarks on a PC with a dual Pentium Pro Processor. The Pentium Pro Processor is a 32-bit processor with the RISC technology. This processor uses dynamic execution, a combination of improved branch prediction, speculative execution and data flow analysis. The clock speed of the computer was 200MHz, and it was equipped with 64MB of memory and a 2GB hard disk.

The tests were all performed ten times and the mean of the measurements was taken as the result for the test. This result is an indication of the performance of the function being evaluated. We have considered average values, because they are more representative of the performance the user can obtain from the machine. Other authors consider minimum values, because they are supposed to be free from the influence of the operating system and other users. The tests were taken with only one user on the machine. All tests compared the performance obtained for Solaris threads, Provenzano threads (PT), FSU_Pthreads (FSUT), PC threads (PCT), CLthreads (CLT) and LinuxThreads (LT). Threads created in Solaris are permanently bound to an LWP to take full advantage of the hardware platform used.

The numbers presented in Figure 5 are the results of the thread-management measurements. All values are given in microseconds, except for granularity of parallelism where values are given in number of iterations. In general, it can be seen that user-level packages are more efficient than kernel-level packages and Solaris threads, since the threads are created on top of the operating system and are invisible to the kernel; however, these libraries are not useful for multi-threaded applications running on multiprocessor systems. This is true for Provenzano threads and FSU_Pthreads, although PC threads present more time-consuming results. Results for granularity of parallelism are shown for kernel-level libraries (Solaris, CLthreads and LinuxThreads); user-level libraries

cannot execute multiple threads in more than one processor. Figure 5 shows how Solaris can take better advantage of multiprocessor architecture. Comparing thread execution and granularity of parallelism results, we can see that context switching is more time-consuming for Linux threading (CLthreads and LinuxThreads) than for Solaris threading. LinuxThreads can take better advantage of multiprocessor systems than CLthreads can.

Figure 5. Thread Management Results

Figure 6 depicts the results of synchronization management measurements. PC threads (PCT) is less efficient, although it is a user-level library. Results show that Provenzano threads is the best user-level library evaluated, and LinuxThreads is a good kernel-level library for use in Linux machines.

Figure 6. Synchronization Management Results

Conclusions

Our objective was to evaluate and compare the performance of five POSIX thread libraries available for Linux and how they compared with other operating systems, such as Solaris. Results were concentrated in thread-management and synchronization-management measurements. Primary results show Provenzano threads to be the best user-level library, and LinuxThreads is a good kernel-level library. Moreover, results show that context switching is more time-consuming for Linux threading (CLthreads and LinuxThreads) than for Solaris threading.

Resources

Felix Garcia is an associate professor in the Department of Computer Architecture at the Polytechnical University of Madrid, Spain. His research interests include operating systems, file systems and parallel and distributed systems.

Javier Fernandez is a student member in the Department of Computer Architecture at the Polytechnical University of Madrid, Spain. His research interest is operating systems. He received his MS in Computer Science in 1998.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Linux and Open-Source Applications

Peter Jones

M. B. Jorgenson

Issue #70, February 2000

The building blocks for a secure and trustworthy computer platform.

Can you trust your computer? This question is becoming increasingly important as both consumers and businesses adopt the Internet as a medium for financial transactions. Even more worrying is the number of computers with Internet access that are also used to store sensitive technical or corporate information. We feel that by using 128-bit encryption and similar techniques, many users are simply deluding themselves into thinking their data is secure.

The proliferation of Internet connections in the last few years is the source of a major security concern. The real threat is not to data traveling over secure connections, but rather comes from an inability to safeguard the data on the machine itself. By using open-source software as well as multiple software distribution channels, we believe potential network-related security problems can be largely eliminated.

Introduction

Credit-card information is often transmitted between users and commerce servers using 128-bit encryption provided through the “secure socket layer” in many web browsers. Information within financial organizations is usually transmitted through private links, relying on the integrity of the communications carrier to help ensure privacy, or through virtual private network links where network encryption devices protect the data.

While these methods of ensuring the secrecy of transmitted data have proven to be quite practical, they overlook what we consider to be a potentially far more serious security leak: direct Internet connection of computers at either

end of the secure link, combined with the presence of untrustworthy application or operating-system software on those machines.

Worms, Viruses and Easter Eggs

The world's first major publicized experience of the damaging potential of a worm—a virus-like program that propagates through a network by taking advantage of security loopholes—was on November 2, 1988 (see Resources 1). A graduate student whose worm accidentally went wild managed to bog down thousands of Sun 3 and VAX computers that made up the backbone of the early Internet.

What is not so well-known is that the worm included a high-speed password-cracking algorithm, designed to allow it to gain access to more privileged operating-system functions. Had the worm been designed for espionage and operated at a much lower CPU priority level, it could well have scanned thousands of machines for “interesting” information and quietly sent data back to its author without anyone even noticing.

Computer users today are all too aware of the risks that viruses pose. Virus checkers are now commonly used to search out and destroy viruses that could do harm to a computer system. However, viruses are usually detected because of some action which they initiate to call attention to themselves. A virus developed for espionage would likely be explicitly designed to avoid doing anything that might call attention to its existence. As with the Internet worm, when armed with the appropriate technology to scan a system for interesting information, a virus could take advantage of an Internet-connected computer to “call home” with its findings and await further instructions.

What if the snooping software was intentionally loaded by the user and thus could never be detected by a virus checker? Easter eggs are sections of code within most common application software packages that can be activated by a series of undocumented commands. Eggs generally do something completely unrelated to the host application and unintended by the manufacturer. That isn't to say that eggs occur accidentally—they owe their existence to programmers who secretly add sections of code without any authorization from their employer to do so.

Most known Easter eggs are small and harmless, but as applications and operating systems have gotten larger over the last few years, it has become possible for rogue employees to include much more elaborate eggs and have them released with the finished product. An Easter egg that provides a good example of a significant amount of code being included in a popular product is the flight simulator in Microsoft Excel 97 (see Resources 2).

Word 97 has a surprising dictionary entry: if a sentence includes the word “zzzz”, the auto-spell-checker underlines it and offers “sex” as the corrected spelling. It's clear that Microsoft did not know about that entry when they released Word 97, and demonstrates the inability of the manufacturer to adequately monitor its programmers.

Virtually all versions of Microsoft Windows and application programs from other vendors also contain a plethora of Easter eggs (see Resources 3).

Back Doors and Other Intentional Security Problems

Back doors can easily be embedded in large programs. Occasionally they serve the legitimate function of allowing a manufacturer to perform remote maintenance. But what if a manufacturer embedded a secret door to be used for devious purposes? Would the user even notice?

Programs have certainly become too big for inspection of their executable code for possible security loopholes. More often than not, we don't actually know what a program such as a word processor is doing at any one time—perhaps it is saving a backup copy of the document being typed; perhaps it is scanning the hard disk for credit-card numbers.

Are any of the programs you are running doing tasks you aren't aware of? While writing this article, a prompt appeared on one of the author's screens, informing him that MS Explorer had committed an illegal operation and would be shut down—only he had never explicitly launched Explorer. When starting Visual C++ at home, Windows 95 tries to connect to an ISP. Visual C++ on a machine at work starts without incident, presumably having made the connection through its permanent Ethernet connection to the Internet. If it were not for the first machine, we would never have suspected any network connections were being made.

Early versions of Windows 98 also had an interesting feature. As a result of a programming error, the network registration section passed on system and personal-identification information to the operating system's manufacturer, even if the user explicitly elected not to do so (see Resources 4).

Some versions of Netscape Navigator had an unintended, back-door-style bug first discovered by Cabocomm, a software company located in Aarhus, Denmark (see Resources 5). Web site operators could exploit this error to allow them to upload the contents of any file on the Netscape user's hard disk, making anything on a machine running Netscape 2.*x*, 3.*x* or 4.*x* world-readable to even inexperienced web page creators.

Where Do You Want to Steal Data from Today?

Given the various options discussed so far, what would be the best way to infiltrate as many computers as possible with a data-gathering agent? An ideal vehicle for such a program would be a large application, like Microsoft Office. The overwhelming success of this product has led to its installation in a very high percentage of computer systems. The trick, of course, would be inserting the rogue code into the host program in the first place. Like most corporations, Microsoft would never approve of something like this. However, from the Easter egg examples, we know there are sections of common software packages that definitely did not get corporate approval and can contain substantial functionality.

The unauthorized code could be further hidden by encrypting large portions of it and having some small code fragment decrypt and activate it on demand. An even more flexible technique would be to have a small Easter egg determine if the computer is connected to the Internet, and if so, open a connection to some foreign host. By downloading code from a remote site at runtime, the Easter egg could be tailored to do something to a specific computer or group of computers that wasn't even thought of at the time the original code was created. Perhaps the code would look for computer schematics if the egg was running on a machine inside the ibm.com domain, or automotive sales figures inside gm.com. With browser functionality becoming more and more embedded in operating systems and applications, one more web connection would appear as harmless as the thousands of other web connections continually being made from the victim computer during the course of the day.

The amount of code used by modern programs prevents any manual scrutiny by a few programmers from providing meaningful verification that a program is “safe”. Expert systems, such as those used to track down Y2K problems and conventional viruses, could be used to try to uncover rogue code, but encrypting the implant could render this approach ineffective. Our conclusion is that there is no way a user could effectively scrutinize the object code of an application to determine that it is “safe”. Neither can any software manufacturer.

Proprietary Operating Systems

One possible solution is to make our operating systems more secure. Microsoft Windows NT is a substantial improvement in security from its Windows cousins. NT provides good password security and the ability to regulate access to system resources by different categories of users, and it has generally acceptable network-security features.

What if the OS itself is not safe? We have already suggested that large programs cannot be screened for security violations by programmers or expert systems. The latest operating systems certainly fall into that category, with the result that we cannot be sure the OS itself is not the source of a major security risk. Indeed, most operating systems also contain Easter eggs of one form or another. Thus, there is little point in being concerned about the security risks of application programs if the operating system is suspect.

Open-Source Operating Systems as a Solution

Open-source software offers a way out of this dilemma. If the source code is open, it can be inspected, and security holes can be found and fixed. Intentional security violations become much harder to hide and will almost certainly be discovered by the thousands of amateur and professional programmers on the Internet. If the source is truly open and widely distributed, flaws of all kinds will be discovered and announced on web sites and newsgroups. This form of interaction has proven remarkably effective in making Linux one of the most (if not *the* most) stable operating systems available.

With this type of public code review, can users be reasonably sure that Linux is trustworthy? Can users be sure that, if sufficient safeguards are incorporated into the OS, their data is secure? Linux had to be compiled using a compiler—what if the compiler was corrupted? It appears we now have to insist that even the compiler used to compile the OS should be open source. For truly concerned users, even that will not be enough, and a procedure involving multiple compilations on different platforms using different initial compilers would be required to produce the object code of the open-source compiler used to compile Linux.

Creating a Secure Linux Operating System and Compilers

In order to create a Linux build you can trust with sensitive information, you first need a compiler known not to insert hidden code as it compiles the operating system. How do you create a trusted compiler when starting off with compilers and operating systems that are not trustworthy? We propose the following as a possible sequence of steps.

- Have thousands of programmers on the Internet inspect the source code of the compiler/linker, GNU C++.
- Create an executable of the compiler/linker by compiling the source on a number of different platforms using different compilers and linkers.

- Use the newly compiled compiler/linker executable on each of the different platforms to cross-compile themselves, as well as a number of different test programs, to a single platform such as x86.
- The cross-compiled compiler/linker and sample program executables on each of the different platforms are then compared. If they are not identical under a byte-by-byte comparison, one or more of the newly generated compilers/linkers is probably subject to a security problem, and the system(s) and compiler source code should be investigated.
- Assuming all the newly generated compiler and sample executables are identical, it can be asserted with a large degree of confidence that both the intermediary compiler/linker executables and the re-compiled compiler/linker executables are trustworthy.

Now that a safe version of GNU C++ has been created, the next step is to repeat the process to create a secure Linux build:

- Have thousands of programmers on the Internet inspect the source code of the operating system, key system libraries, utilities and scripts.
- Cross-compile executables of the operating system and its libraries/ utilities to a common architecture (x86) using the trusted compilers.
- Perform a byte-by-byte comparison of the executables and proclaim them trustworthy if they match.
- Now, build a minimal Linux system installation using the trusted components, and gradually expand on the system's functionality by certifying all additional components.

Creating a Trustworthy Browser and Other Applications

As soon as a trusted Linux platform has been created, a similar process could be used to create a trusted browser. Because the browser is the application used to download other applications as well as communicate securely for e-commerce, it deserves special attention. To create a trusted browser:

- Have thousands of programmers on the Internet inspect the source code.
- Compile executables using a trusted platform and compiler.
- Using a trusted compare utility (on CD-ROM), periodically compare CD-ROM versions of the executables of the operating system, compiler, utilities and applications (including the browser) with what is currently on the hard disk, just to ensure that an application hasn't used some hidden code to corrupt the platform.

Since Netscape Corp. has taken the initiative to open their browser code, Netscape would be the logical choice as a trusted browser. Ideally, all other

applications to be used should be made trustworthy too, so the set of steps listed above should be carried out to create each new trusted application.

Banks of Trustworthy Software

Of course, getting users to carry out this certification process would be impossible. What is really needed is a system of software repositories—or "banks"--from which users can obtain certified versions of Linux and associated applications.

A national organization, such as the U.S. National Security Agency, could verify open-source programs and place both source and binaries on the Web for immediate download. However, this approach would be subject to the same concerns that make closed-source software insecure. A disgruntled employee could add some extras to the certified code, or perhaps a government organization will decide that having a back-door might be useful for national security reasons.

Clearly, no single testing organization can be trusted. A better approach would be to have three or more certification organizations, each with its own download site. The National Security Agency in the U.S., the Communications-Electronics Security Group in Britain and the Communications Security Establishment in Canada could each independently verify and make certified binaries available. A user could then download the same binaries from all three sites and be sure they are trustworthy if, and only if, no differences are found. While there is a potential security problem in downloading over the Internet (after all, a devious ISP could intercept the FTP request and divert it to a rigged server), the likelihood of that is small and the chances of it being discovered are high.

For even greater security, each of the major certifying sites would also make certified CD-ROMs available, preferably each with a simple file-comparison program directly bootable from the unalterable CD. That way, one could order certified CDs from two or more certifying agencies and do a quick file comparison between them as a final verification. The write-only nature of CDs would also prevent any corruption on one from contaminating the other CDs.

Of course, trusting the U.S., Britain and Canada's electronic espionage agencies might leave something to be desired. By requiring each certifying agency to make not only its certified binaries available but also the original source code, it would be possible for other countries, companies or individuals to set up their own complementary certifying sites. Presumably, millions of Internet users would be continuously watching the various sites offering certified applications and operating systems, and a sudden discrepancy at one of them would be noticed, investigated and exposed. By having each certification organization

keep its own set of confidential source-code examples for testing the output of compilers being certified, one could dramatically reduce the already small chance of a clever compiler recognizing test code and producing sanitized executables during certification.

At this point, it is also worth emphasizing that a proliferation of independent certifying sites for open-source software located around the world would not only be an excellent safeguard against any sort of Easter egg or back door, but would also ensure that bugs—particularly the security-sensitive ones—are exposed and quickly corrected.

Conclusions

There are potentially severe security problems arising from the inherent nature of closed-source software and its use on Internet-connected computers. While the chances of someone planting a globally or even nationally destructive section of code in a popular operating system or application program is low, the consequences of such an event are potentially too disastrous to ignore. Indeed, a well-orchestrated Easter-egg attack could make the Y2K problem look miniscule in comparison. To safeguard against these problems, the solution is the replacement of closed-source applications and operating systems with certified open-source programs. Organizations providing banks of certified trusted applications and operating systems could provide a vital public service.

Resources

Peter F. Jones is a research engineer at Neptec Communications in Ottawa, Canada. He received a B.Sc. (1986) and a Ph.D. (1993) from the Department of Electrical Engineering at Queens University, Kingston, Ontario, Canada and is also a licensed engineer (P.Eng). Peter has worked on a variety of software projects including writing SVGA card graphics drivers, creating a Java web search engine, and developing a Linux-based multiple-sound card interface library for an adaptive antenna phased-array HF modem. He is currently working on two projects: developing a miniature single-board Linux computer for home and office applications and studying the characteristics of the Space Shuttle's TV cameras for the purposes of developing algorithms to reduce image distortions. Peter can be reached via e-mail at pjones@neptec.com.

Mark B. Jorgenson is at Neptec Communications in Ottawa, Canada. His B.Sc. (1984) and M.Sc. (1989) are both in Electrical Engineering from the University of Calgary and he is also a licensed engineer (P.Eng). Mark's main research focus is in wireless communications, with emphasis on link-layer aspects. Mark has recently led the development of a software radio prototype and is currently leading a team designing an advanced HF radio modem. He can be reached via e-mail at mjorgenson@neptec.com.

Laptops for Linux!

Jason Kroll

Issue #70, February 2000

At the time of this writing, the two Linux-specific laptop providers are LinuxLaptops and ASL Workstations.

- Products: LinuxLaptops Attache ASL Workstations AS-LT300
- Manufacturer: LinuxLaptops ASL Workstations
- E-mail: ncm@linuxlaptops.com sales@aslab.com
- URL: http://linuxlaptops.com/ http://aslab.com/
- Price: $2500 US $2800 US
- Reviewer: Jason Kroll

laptop *n*: a computer in graceful, self-contained form. Laptops are small, wireless, resource-minimal and portable. You can take them wherever you go and hide them when not in use—no ugly computer to occupy space. The LCD flat screen displays are clearer and cleaner than cathode ray tube monitors (with less brain-wave-zeroing effect), and the keyboards are small and quiet. Even without portability, laptops have many advantages over desktop machines.

Laptops also have their shortcomings, the most obvious of which is price. Laptops are *very* expensive and difficult to repair or upgrade, although it *can* be done. They also offer inferior performance compared to desktop machines, specifically in disk access, processor speed and hardware support. Hard disks on laptops spin slowly and are loathe to accelerate or decelerate; hence, disk operations can be very slow, especially random accesses. Take a look at the Bonnie benchmarks (see Table 2) to see just how slow. Processors are also slower and more expensive. One reason is laptops have relatively poor ventilation (making them quiet), so if you have too many megahertz/gates/volts on the chip/CPU, it will get too hot. (Notice the popularity of StrongARM processors in machines without powerful fans, for example the NetWinder and Empeg.) Also, laptop processors are physically different from standard

processors because they must be small and low-profile, although smaller chips require less voltage so they don't generate as much heat.

LinuxLaptops Attache

The Good

- Debian GNU/Linux
- Entirely free software
- Flexible, loose
- Superb console mode
- Highly tuned, fast
- Well customized
- 14.1" LCD screen is clear, flicker-free

The Bad

- GNOME/Enlightment is a tad slow, sometimes problematic
- Too much swap
- Missing some software
- Very expensive

At the time of this writing, the two Linux-specific laptop providers are LinuxLaptops and ASL Workstations. LinuxLaptops is a specialty shop run by Nathan Myers, which deals exclusively in laptops for Linux and has three models at the moment. It is highly focused on optimization and tuning, working within the interesting limitations of laptop hardware, to maximize Linux performance. ASL Workstations, on the other hand, is a successful Linux workstation builder which provides many excellent workstations and makes its AS-LT300 laptop as a logical component in a complete product line of high-powered, server-oriented machines. Penguin Computing is also developing a laptop, but it is still in beta and was not available for review. VA Linux Systems, the best-known Linux machine maker, has discontinued laptops for the time being.

The Laptops

Since only enormous firms have their own laptop factories, laptop providers for Linux buy standard laptops and configure them for Linux. Incidentally, the AS-LT300 and Attache are both ChemBook 7400s. Internally, they have slightly different configurations, but the most significant difference is between their respective Linux setups. Table 1 shows the more relevant features of the laptops sent to *LJ*, but keep in mind that you have much flexibility in configuration if you order a laptop for yourself.

Table 1. Laptop Features

	LinuxLaptops	ASL
Processor	Pentium II 333 MHz	Pentium II 366MHz
L1/L2 Cache	32K / 256K	32K / 256K
RAM	128M SDRAM	96M SDRAM
BUS	66MHz AGP/PCI	66MHz AGP/PCI
Hard Drive	6.4GB Ultra DMA/33	6.4GB Ultra DMA/33

Linux	Debian GNU/Linux 2.1	Red Hat 6.0
X Server	XFree86 3.3.2.3	Accelerated X-Server 5.0
Desktop	GNOME/Enlightenment	KDE or GNOME
Console	128x48	80x25
Video	ATI 3D Rage Pro LT 4MB	ATI 3D Rage Pro LT 8MB
Audio	awaiting driver	Open Sound System
Kernel/Size	2.2.11 / 715K	2.2.12 / 636K

Hardware and Ergonomics

Once upon a time, the important issue with a computer was how well it worked. Nowadays, people have these funny ideas; they'd like to be comfortable while using their machines and not get headaches, neck cramps, back pain, carpal tunnel syndrome, radiation sickness, etc. Laptops win on most counts, the only problem being that tall people may be made to suffer.

Monitors

The LCD flat screens on these laptops are pristine and clear: images are sharp and detailed, without glare, fuzziness, or flicker. There are exactly 1024x768 pixels on the screens (unlike CRT monitors, which often haven't got enough phosphors for the high resolutions they “support”). While laptop monitors are not as lovely as the extraordinarily expensive LCD flat panels, they have remarkable clarity and accurate color. And, they consume very little power while emitting hardly any electromagnetic radiation. Ever since these laptops arrived, I haven't touched my desktop computers, preferring instead to use **ssh** to log in from the laptops.

Both monitors are identical, 14.1-inch Active Matrix TFT (thin film transistor) panels. The ASL looked a tiny bit better in X, while the Attache looks significantly better in console mode. The reason for this is the Attache has been configured with a frame-buffer device, so the console shows up in 1024x768 instead of the usual 640x400. This means the screen can fit 48 rows and 128 columns of crystal-clear text (literally), as opposed to the usual 25 rows by 80 columns in 640x400 mode (which will appear ghosted on a laptop). LCD screens look proper only in their ideal resolution, so a monitor made for 1024x768 will be clear only in this mode and should stay there all the time to take advantage of

the excellent image quality. If you are a console enthusiast, LinuxLaptops' frame-buffer-enabled kernels which run the console in 1024x768 are the only way to go.

The main disadvantage of LCD screens, other than expense, is they have particular viewing angles; if you look from the wrong angle, the colors and brightness will be off. Also, the screens are very delicate, so you can't get mad and punch your monitor. One shouldn't press on the screen at all, so it takes a concerted effort to clean. If you buy a laptop off the shelf and try to install Linux, configuring X will be a challenge. Finally, if you are slightly taller than average, you might develop neck and back pain from hunching over, since the keyboard is fixed so close to the monitor.

Keyboards

The ChemBook 7400 keyboard is a compact arrangement of full-size keys which are perky and shallow. You don't have to push down very far, so typing should conceivably be faster. One problem with these keyboards is they sit far back in the laptop, leaving a big surface in the way, and this usually results in the user resting his or her wrists on the laptop. After about thirty minutes of typing away while resting my wrists on the laptop, I feel a horrible throbbing pain which follows me around all day; I suspect this is what carpal tunnel syndrome feels like. So during the time I have been using only laptops for their nice monitors, I have also been using an external keyboard. Still, the key action on the laptop keyboard is fast, and LinuxLaptops even took the effort to remap **CAPS LOCK** to serve as the **CTRL** key.

Pointer Device

Truthfully, I think track balls and little red joysticks are neat, but these days the typical pointer is a pad which one runs a finger across, inspiring the pointer to move accordingly. The pad is easy to get accustomed to, and the movement is logical and straightforward. The mouse buttons are two wide buttons situated below the pad. Clicking both at once will emulate the third button, but these days we should just have three buttons. Also, tapping on the pad often acts as a mouse click, and this leads to some funny accidents, especially on the Web.

Software

The fundamental difference between the AS-LT300 and the Attache is that the AS-LT300 is a Red Hat system, whereas the Attache is a Debian system. Although distributions are usually fairly similar, especially when one gets past installation, these two are essentially polar opposites. Red Hat, as we know, is the *market* leader, the epitome of commercial Linux, while Debian GNU/Linux is a vanguard of the free-source philosophy, and typically the hacker's favorite

setup. Herein lies the difference in the feel of these laptops: ASL went the commercial route, while LinuxLaptops took the expressly non-commercial route.

X

ASL includes the Accelerated X-Server, while LinuxLaptops runs the standard XFree86. According to Xi Graphics, there is usually a 40% to 60% performance improvement gained from running the commercial servers, as well as support for many more cards. I have not noticed any visible advantage; GNOME/Enlightenment is slow on both laptops. Nevertheless, if you do graphics-intensive work, there may be a benefit here. Both laptops use the ATI 3D Pro LT card, so the graphics performance is similar. Neither in X nor in console mode did I notice the lagging, slow updates associated with LCD screens of the past. Graphics on both are superb.

The Good

- Well stocked, fully loaded
- KDE or GNOME
- Faster X server
- Complete, functional laptop for Linux
- Clear, flicker-free 14.1" LCD monitor

The Bad

- Console mode is ghosty
- Disk access a bit slow
- Has some unnecessary software
- Very expensive

The window manager situation is a bit different. The AS-LT300 offers a choice between KDE and GNOME, and the Attache uses GNOME/Enlightenment. While the Attache's devotion to GNOME/Enlightenment (which *is* well configured) keeps good faith with the open-software movement, the window manager and desktop environment are a tad too resource-intensive for a mid-range laptop, not to mention being unstable and slightly buggy.

Sound

As is typical of Linux, audio support is a mess. The Attache awaits the release of the free sound driver, so other than beeps (which you can mercifully turn off with a volume control), you don't have audio support. The AS-LT300 has OSS audio support, which means you can play CDs, but the audio devices are not

set up properly so you'll miss out on mpegs and the like (I expect ASL has fixed by now). Still, the microphone appears to be working, because at full volume, the machine starts generating horrible feedback when I type. What to do about MIDI? Might as well get a hardware sequencer.

Network

Network support on both laptops is fine and simple. All you have to do is plug an Ethernet/Modem card into the PCMCIA slot, edit the network files (five minutes tops), reboot, and your system will be completely on-line. Networking is transparent and I even swapped the network card in and out of the computers while running and without any disastrous effects. The network card gets *very* hot, though; I worry it will melt. It would be preferable to have Ethernet/modem built into the laptop instead of using a PCMCIA card, since it looks less graceful to have this gizmo sticking out of the laptop's side.

Benchmarks

Benchmarks are good general indicators of system performance, but they're often misleading and not entirely relevant. It is true these laptops do not compare well performance-wise to desktop machines or servers, especially in terms of disk access, but then, these laptops are not servers and servers are not particularly portable. Laptops are generally one- or two-user machines and will not be called on to do anything resource-intensive. As for disk access, you don't need a super fast drive if you won't have several dozen users reading and writing all at once, and you don't need too many megahertz or that much RAM to run vi or Emacs. Multimedia would be the one area for which a single user would need supercomputing power, but then multimedia support on Linux is not outstanding. So, Table 2 shows the Bonnie and BYTEmark results. The AS-LT300 shipped with a faster processor (and higher price tag) so its processor results are slightly better, while LinuxLaptops has spent much effort tuning for hard-drive performance, hence the better results here. I set both laptops to “Suspend to RAM” and “Large Filesystem”, and ran the benchmarks several times for best results. Bonnie's results fell into a broad range, due partially to variance in access speed depending on the location of the data on the disk. Laptop drives tend to be single speed, so the farther in you go, the slower your access speed.

Table 2. Benchmarks

	Attache	AS-LT300	

Memory Index	1.361	1.566							
Integer Index	1.145	1.324							
Floating-Point	2.277	3.616							
	Sequential Output	Sequential Input	Random						
Machine	MB	K/sec	%CPU	K/sec	%CPU	K/sec	%CPU	K/sec	%CP
Attache	100	5608	98.7	14571	22.6	3173	9.3	4577	75.
ASLT300	100	5659	98.9	13442	21.7	2947	8.7	4306	72.

File System Notes

Each laptop partitioned its drive differently, and the Attache even includes a tiny DOS partition for the emulator, but included neither the correct file-system module for the new kernel nor an entry in fstab or mtab. I think both laptops have a slightly odd partition setup, with too much swap (256MB on the Attache, 128MB on the AS-LT300). Table 3 is a short file-system table so you can see how they're set up for Linux, a bit systematic for what is presumably a single-user system.

Table 3. Laptop File Systems

File System	1024-blocks	Used	Available	Capacity	Mounted on
/dev/hda2	204945	50155	144206	26%	/
/dev/hda6	1492311	436562	978639	31%	/usr

/dev/hda8	2901830	5228	2746538	0%	/home
/dev/hda9	965801	13	915894	0%	/other
File System	1024-blocks	Used	Available	Capacity	Mounted on
/dev/hda2	161027	75501	77210	49%	/
/dev/hda3	1565171	1228067	256212	83%	/usr
/dev/hda5	4054179	64890	3779501	2%	/home
/dev/hda6	117087	13	111028	0%	/tmp

Linux Specialty

It is difficult to put together a top-quality laptop for Linux. There are too many peculiarities and quirks of the hardware, and drivers are hard to come by. The best anyone can hope for in the near future is that everything will work after a bit of fixing. I would expect LinuxLaptops to lead the way, with ASL and the others adopting those ideas which turn out well. We don't have Linux-specific laptop factories, and we don't have perfect support for all the hardware. Laptops for Linux are coming, and hopefully some day they will be as completely functional as desktop boxes. Right now, they lack audio and some video support (svgalib, for example). These laptops do have PCMCIA slots, infrared ports, disk and CD-ROM (optional DVD) drives, and some potential for expansion. Check the web sites for the full technical details.

Now that players have entered the Linux laptop market, we're going to see quality go up and price go down, though there are many options to buying a Linux laptop. If you need just a portable machine with little computing power, you could get an inexpensive or used laptop and install Linux. If you want the flat screen monitor, you could buy one (I think they are worth the reduced eye strain and radiation); Xi Graphics already has some drivers available. If you want a small computer, pair your flat panel display with a NetWinder and Happy Hacking Keyboard Lite. If you need a laptop but don't like either of these, you could try one of LinuxLaptops' other models (the Paquet looks exceedingly cute). Also, you could wait until we move on to 64-bit RISC technology (one x86 is enough, eh?). If you are buying only one computer, and you live in an apartment which you'd like to look nice (i.e., no ugly CRTs), or if you travel and need your computer, these laptops have much to offer. The

choice between our review subjects should be easy. By now, you must know if you're a Red Hat (AS-LT300) person or a Debian (Attache) person, although realistically, LinuxLaptops has made a more highly tuned machine for the price. Either way, how can you go wrong putting Linux on a computer, especially one as cool as a laptop?

Jason Kroll (info@linuxjournal.com) is Technical Editor of *Linux Journal*; he gets a new bio and photo each month. He likes animals, though sometimes cats remind him of misfiring computer programs.

Archive Index Issue Table of Contents

Advanced search