Publishing a Magazine with Linux

Phil Hughes

<phil@ssc.com>

ABSTRACT

Since 1994, *Linux Journal* has been published using primarily Linux-based tools. In this presentation, I will look at how we have developed this Linux-based solution to a real business problem. The presentation will include design trade-offs, what we have learned from our mistakes, what Linux does for us and where Linux falls short.

Specific topics to be covered include: editorial preparation using Linux, Linux (and non-Linux) in layout, producing ancillary products, Linux and the web and general office use of Linux. I will also explore what needs to be added to the Linux tool chest in order for it to be a complete solution.

This talk will not be extremely technical. It is intended to look at how to address problems with Linux tools rather than look at the code that solves the problem. It will also help someone understand the production process of a print and web magazine.

1. Brief History of SSC

1.1 SSC's Traditional Products

SSC has been publishing UNIX documentation since 1983. Prior to *Linux Journal* that documentation consisted of:

- Pocket-sized reference cards
- Training manuals
- Books

Virtually all the reference cards and training manuals were done using troff. A laser printer served as the proofing printer for years.

Once we purchased an ink jet printer, we started proofing color products such as reference cards on the ink jet. In order to produce proofs on the ink jet printer and color separated film to send to the printer without having to make source file changes we use a make file and different macro files for the composite and separated output formats.

Today we produce pocket references using this same method—the only change being that we now have a color laser printer.

To simplify the process we use key maps in vi for the most common formatting commands such as font changes between bold, italic and roman.

1.2 Computer Evolution

Our first computer platform was a Codata 3300 running BSD 4.2 UNIX. Over the years we have used an IBM AT running Xenix and various 386- and 486-based systems running SCO UNIX and UnixWare.

We made our transition to Linux in 1993 and then began replacing dumb terminals with Linux-based workstations. Today, there are Linux workstations on all but two desks. We also have a handful of Linux-based web servers and file servers plus a firewall running Linux.

1.3 The SSC Philosophy

Our company philosophy has always been ‘‘Don’t make a product that you wouldn’t use yourself’’. Have we always made this check before developing a product? Well, no. And, we have always paid the price. For example, we decided the world must need MS-DOS reference cards. We started with DOS 3.1 and kept updating, changing and attempting to sell them.

The bottom line is that we have sold so many more UNIX-related references into a much smaller market. We didn’t use them ourselves and, apparently, DOS users didn’t know how to read.

2. Introducing Linux Journal

When *Linux Journal* entered the picture it was time to add staff, upgrade some internal equipment and generally get up to speed with producing a monthly publication.

The good news was that we already were using Linux to do much of our work and producing *Linux Journal* passed the company philosophy check. The bad news was that there is a lot more to do to produce a monthly magazine and we had no idea if Linux could do the work we needed done. But we were going to give it a try.

Producing the magazine includes a whole bunch of parts. In the following sub-sections I will discuss the normal production process. Then, in the following section, I will detail the processes necessary to put the whole picture together.

2.1 Editorial

When many people think of a magazine, they just think about the editorial content. As it turns out, this is a fairly small part of the whole production process. The basic steps here are:

- Establish the editorial direction and outline specific topics where articles are needed
- Seek the articles
- Edit the articles
- Send them to layout along with a report telling layout about the articles

2.2 Advertising

Advertising is the primary source of revenue for a magazine. (Your subscription money pays to get you the magazine.) The major pieces of advertising production are:

- Produce advertising promotional materials such as rate cards and editorial calendars and send them out
- Track advertising leads
- Secure advertising contracts and insertion orders
- Create a booking report for use by layout
- Handle advertising art (mechanicals and electronic)

- Bill for advertising

2.3 Production

This is the actual layout process of the magazine. The inputs to this process are the materials from Editorial and Advertising. The output is a CD with PostScript page files on it which is sent to the printer and color proof prints. The steps are:

- Place ads into a magazine template
- Wrap articles around the ads
- Produce a printed version for review
- Make changes and produce a printed version for the proofreader
- Make changes and produce a final printout and PostScript files on CD

2.4 Circulation

There are two types of circulation: subscriptions and distribution. We have currently outsourced subscription processing but we used to do it in-house. We continue to handle the e-mail and make subscription information available on our web pages.

Distribution is the wholesale sales of magazines. We handle some resellers directly and others through distributors. In either case, the process is the same. We collect orders and produce shipping paperwork so the printer can send the magazines directly to the distributors and resellers.

2.5 Web

Our web sites include an assortment of information. Some of it is relatively static, other parts are dynamic. It is not all *Linux Journal* specific. The major components are:

- Linux news and articles that appear only on the web site
- Linux resources that offer general information and links about Linux
- Subscription forms and information including subscription status information
- *Linux Journal* article archives, advertiser's indices and tables of contents
- Author and advertiser information

News is updated daily, subscription status weekly and articles monthly. The other information is updated as needed.

2.6 Related Products

In addition to the monthly magazine, we sell some related products including our ''famous in Atlanta'' caffeinated peppermints, t-shirts, caps and such. All of these products are available on-line (only). To support this operation we maintain the store software, update products and deal with processing orders. This is currently done by manually entering the orders.

3. System Details

3.1 Editorial

The internal editorial department is totally Linux-centric. As most of our authors are running Linux there is virtually no problem using Linux systems for this part of the process. I will talk about the one exception later.

The most popular tool in the Editorial department is the vi editor. Number two is mutt for handling e-mail.

3.1.1 Article Management System When an article idea comes up, it is entered into our Article Management System (AMS). This is part of our Progress database (see *Linux Journal* #29, September, 1996). The system includes descriptive information, status and tickle dates. As the article progresses through the steps, this database is used to track it. When we want to run an article in a specific magazine, its status is changed to reflect that. This information is then used to produce the run sheet for layout.

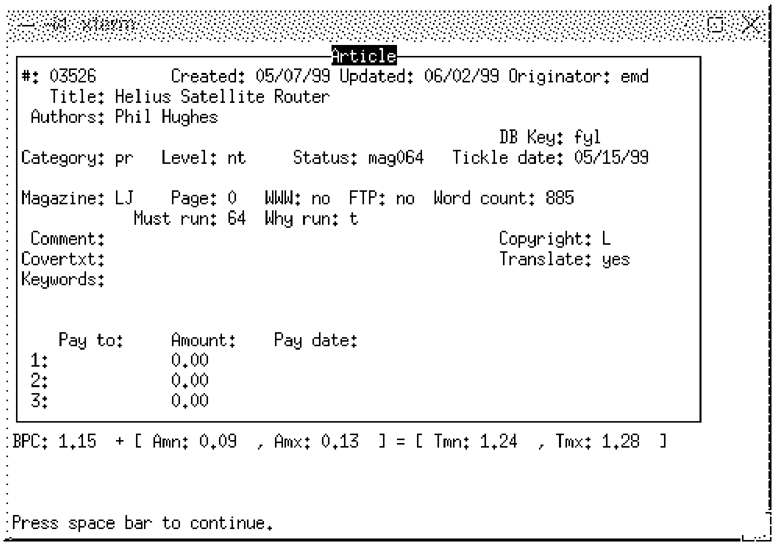

Figure1: AMS Main Article Screen

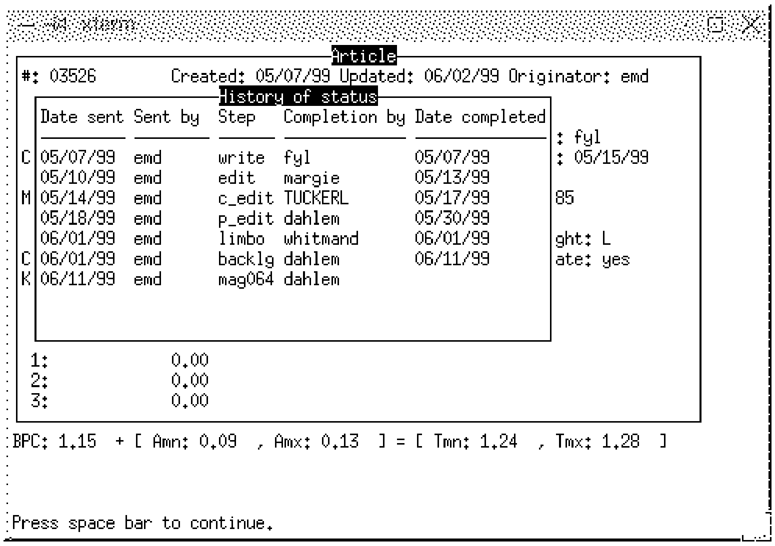

Figure2: AMS Article Status History Screen

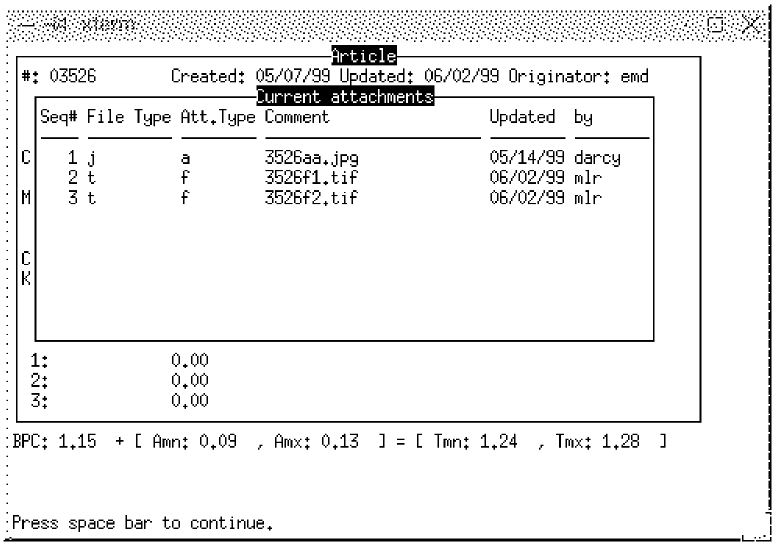

Figure 3: Article Current Attachments

When the layout department completes Layout 1, article page numbers are inserted into AMS. Two formats of the table of contents are produced: one for the web page and one for inclusion in the magazine.

Additional functions of AMS include:

- Article status reports
- Print author contracts and author payment requests
- Article tickle report
- Various article search, enter and update forms

3.1.2 Article Preparation The article preparation process consists of:

- Tagging the articles into the production format we use
- Running the articles through multiple editing phases
- Checking figures and scanning photographs

We reviewed various types of mark-up to use for production. They included HTML, RTF and even our own creation. Our final decision was to work in the mark-up language that our layout program, QuarkXpress, uses. It is a fairly simple format with style tags that you apply to paragraphs and some local formatting if you need to, for example, put a word in italic.

This decision left us with two problems to solve:

- how to easily produce the required mark-up and
- how to proof our files before they went to layout.

The tool of choice for doing the mark-up is vi. The people doing this work use vi all day so, while vi may be harder to learn than a WYSIWYG editor, they get up to speed quickly and, once they are vi-competent, they can do editing faster.

To make the routine task easier, we use a set of vi abbreviations and key maps for common expressions. Note that there are both Quark and HTML abbreviations as there is manual work involved in moving articles from their Quark format to HTML for presentation on our web site.

The basic scoop on their meanings is:

Function Key Bindings for vi:
F1 — Set to font to Courier
F2 — Return to the previous font
F3 — Italic
F4 — Bold
F5 — ASCII 151
F6 — ASCII 150
F7 — ASCII 147
F8 — ASCII 148
F9 — Bullet tag followed by a tab
F10 — Number 1 list tag followed by a tab
F11 — ASCII 34
F12 — NEWLINE

Numeric Key Pad Bindings for vi:
1 — Boxed head tag
2 — Abstract tag
3 — Author tag
4 — Bold w/drop cap tag
5 — Body tag
6 — Courier example tag
7 — Subheading tag
8 — Non-indented body tag
9 — Sidebar heading tag
0 — Sidebar body tag
/ — <I>Linux Journal</I>
* — Layout comment tag
- — Letters heading tag
+ — Letters body tag
Enter — LJ replies tag

The actual content of the .exrc file is:

```
set nobackup
set showmode
set wm=5
map! ^[[11~ <f"Courier">
map! ^[[12~ <$f$>
map! ^[[13~ <I>
map! ^[[14~ <B>
map! ^[[15~ <\#151>
map! ^[[17~ <\#150>
map! ^[[18~ <\#147>
map! ^[[19~ <\#148>
map! ^[[20~ @bb:<\#149>
map! ^[[21~ @bb:1.
map! ^[[23~ <\#34>
map! ^[[24~ <\n>
map! ^[Oq @hdbx:
map! ^[Ow @abs:
map! ^[Os @a:
map! ^[Ot @bdc:
map! ^[Ou @b:
map! ^[Ov @cx:
map! ^[Ow @subhd:
map! ^[Ox @bni:
map! ^[Oy @sidehd:
map! ^[Op @side:
map! ^[Oo <I>Linux Journal<I>
map! ^[Oj @lay:
map! ^[Om @lte-hd:
map! ^[Ok @lte-body:
map! ^[OM @ljr-body:
ab ljslug <f"LJ">L<$f$>
ab fx [fixme: ??]
ab ilj <I>Linux Journal</I>
ab doctyp <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
ab hr  <HR>
ab br  <BR>
ab bhd  <HEAD>
ab ehd </HEAD>
ab ht  <HTML>
ab eht </HTML>
ab bbd  <BODY>
ab ebd </BODY>
ab h1  <H1>
ab eh1 </H1>
ab h2  <H2>
ab eh2 </H2>
ab h3  <H3>
```

```
ab eh3 </H3>
ab h4  <H4>
ab eh4 </H4>
ab bp  <P>
ab ep  </P>
ab bpr  <PRE>
ab epr </PRE>
ab pac <P ALIGN=CENTER>
ab bi   <I>
ab ei  </I>
ab sb   <B>
ab eb  </B>
ab ba  <A HREF="
ab ea  </A>
ab bq <blockquote>
ab eq </blockquote>
ab bt <title>
ab et </title>
ab btt <tt>
ab ett </tt>
ab lst <li>
ab cq %s/<\#34>/"/g
ab cga %s/<\#096>/`/g
ab caa %s/<\#180>/'/g
ab chy %s/<\#150>/-/g
ab cgt %s/<\>>/>/g
ab clt %s/<\<>/</g
ab dn %s/<\n>//
ab cbs %s/<\\>/\/g
ab addq %s/"/<\#34>/g
ab addh %s/-/<\#150>/g
ab addl %s/</<\<>/g
ab addn %s/$/<\n>/
ab addsl %s/\/<\\>/g
ab addsn %s/\n/<\\>n/g
ab rlr 1234567890123456789012345678901234567890
```

3.1.3 Pre-Proofing Articles Displaying the marked up files easily required a little more work. Sure, we could ask layout to import the files into Quark and print them out but this would result in the interruption of work flow for editorial and layout. The best solution would be to make it possible for the person doing the mark-up to proof her changes on her Linux workstation.

Well, it was time for some programming. Linux comes with tools to produce typeset-quality output so all we had to do was convert our files with Quark mark-up into something that one of the included text processors would work with. Considering I have been working with troff since 1983, it was my text processor of choice.

The Quark file is a plain text file and the task is to convert one set of mark-up characters into another set. Specifically, change from what Quark uses to what troff uses. This was a task for awk, another UNIX/Linux tool that has been available for tens of years. Programming was straightforward. Wrapping the pieces in a shell script offers an easy to use program we call qt2.

The script includes on-screen (using GhostView) as well as print output. Below is the beginning of the script which documents its features:

```
#!/bin/sh
# qt2  -- print quark tag files in a reasonable fashion
# things to deal with:
#    paragraph breaks
#    tables and such
#    paragraph styles
#
# Updated: July 21 1998

DATE=`date`
LPR='lpr';

if [ $# -eq 0 ]
then
      echo "Usage  $0 [-p] [-g] file"
      echo "       -p gives double-spaced, printed PostScript output"
      echo "       -g shows file in ghostview"
      echo "       -1 single space output (forces p)"
      echo "       -2 double space output (forces p)"
      echo "       -s save troff file only"
      exit
fi
```

3.1.4 Editorial Odds and Ends Someone is sure to ask why not MS-Word? After all, Quark can directly import MS-Word files. In 1994 I investigated using RTF as a format to work with. It's (sorta) a standard and there are Linux tools (specifically, the Andrew System) available to work with it. Experimentation showed that MS-Word would create bad RTF (according to their own standard) and that good RTF would cause MS-Word to crash. With no control over the code, this looked like a bad approach.

Now, let me talk about that one non-Linux editorial situation. Our Senior Editor, Doc Searls works from his home in California. He is and has been surrounded by Macs for years. Without having him in our office it is hard to infect him with the vi virus. Or, is it?

Early this year we installed a Linux system in his home office. It is generally hidden inside a cabinet but it is there and talking on his network of Macs. Except for one reliability problem, his two-year-old son pushing the power button, it has worked well.

In June I spent a couple of days at Doc's house. At that time I started a little more Linux infiltration. The Linux box now acts as a Mac file server and can talk to his Mac printer. I set up Apache on the Linux box as well and now Doc can drag-and-drop files on his Mac desktop and have them appear on the web server. He isn't using vi yet but we are encroaching on his Mac territory little by little.

3.2 Advertising

Another part of our Progress database handles advertising much like AMS handles articles. When an ad is booked, information about the ad including advertiser, size, color and art disposition are logged. Art disposition means whether it is existing art (mechanical or EPS) that we have or is coming in from the advertiser.

A booking report is produced for Layout to use. Ad page numbers are added to the log once the ads are placed in the magazine and the database is used to output the advertiser's index information in Quark format.

Other functions in the advertising system include:

- Enter/Update rate card
- Update contract, transfer contract
- Various contract status reports
- Update inserts
- Various insertion order reports
- Print advertiser mailing labels (for tear-sheet mailing)
- Print bingo response labels

Working with electronic art (EPS files) requires a tool chest. These files may come in via FTP, in e-mail or on various disk formats including DOS and Mac. To make things more complicated, files are regularly compressed using any of a number of compression tools. The following tools are regularly used to deal with EPS files:

- bbcheck — a locally written script (in awk) that displays bounding box information
- HFS tools for dealing with Mac disks
- Mtools for dealing with DOS/Windoze disks
- file for determining file type
- gunzip, hexbin and such for unpacking the incoming files
- gv for displaying EPS files
- dd for removing Mac and Windoze crud from the front of EPS files
- od to figure out what is really in a file

3.3 Layout

When we started LJ, layout was outsourced and, as much as we hate to admit it, was done on a Mac but still with Quark. In these early days, the most important utility was suntar for the Mac. This allowed the Mac to read and write tar format Syquest 88MB cartridges which we used to transfer the files.

As we started to grow it became obvious that we needed to move layout inside. We elected to continue to use Quark (which is a good choice—it really is the right program for magazine layout) but selected Windoze as our platform. This wasn't out of love of Windoze, just that at the time it was somewhat more compatible with Linux than the Mac.

There isn't anything Linux-centric about what we do here. We continue to use Windoze but it talks to the real world—printers, file servers and the Internet—through Linux systems and Samba.

3.4 Imagesetting

In the early days of *Linux Journal* we also outsourced producing the film from the PostScript files. We quickly realized that the work was costing us time and money. With additional imagesetting work for the SSC publications it became clear that it was cost-effective to purchase an Imagesetter.

We managed to locate a used Agfa 9400 RS/EL Imagesetter and Agfa MaxPlus Raster Image Processor (RIP). We added a new film processor and all the equipment was in place to produce film. In order to manage the PostScript files we spooled them on a Linux system and, with a home-brew shell script, queued them for processing on the RIP/Imagesetter. Nothing fancy here but it did the job. Or, it did for a while.

As the content of *Linux Journal* became more complicated, the RIP became a bottleneck. While the Imagesetter could output the film for a page in about two minutes, the RIP was taking five minutes on simple pages and hours to process a graphics-heavy page. The solution was a faster RIP, but commercial RIPS were in the $30,000 range.

Enter another Linux solution. A local company had produced an ISA card that would interface to the Agfa imagesetter. They were selling the card with some MS-Windows software as an alternative to a $30,000 RIP. We talked to them about the product and it turned out that they were using GhostScript as the software.

Armed with the specs for their card and a commercial version of GhostScript that we purchased, we were off to write a device driver for Linux. From the driver side, the card looked much like a parallel printer card with some extra functionality so it was fairly easy to get it up and running. We went out and bought the fastest computer we could find at the time and had a workstation and RIP, all in one, for about $6000 including the interface card and software license for GhostScript.

How did it perform? We didn't run extensive benchmarking as it wasn't even a fair comparison but I do remember that one page that took over an hour on the MaxPlus RIPed on the new system in 27 seconds.

We used this system for about two years. We changed printers and the format of choice for the new printer is PostScript on a CD. Our Linux-based RIP served us well and could have become a product but we were busy publishing a magazine instead of developing Linux products.

4. Ancillary Products and Steps

There is a lot more to producing a magazine than collecting articles, putting them on a page and getting them printed. And there is a lot more to the Linux part of SSC than *Linux Journal*.

4.1 Subscription Processing

We initially did subscription processing in-house but decided it could better be done by a professional fulfillment service and outsourced it. Twice. What we discovered was that the subscription fulfillment business was caught in the 1980s. They thought six to eight-week turnaround was acceptable and that people would rather mail a card or make a phone call than use e-mail and Web-based solutions.

In late July, 1999 we took subscription processing back in-house because of a large number of subscriber complaints with our then current fulfillment service. We have plans to implement a close to amazing fulfillment system but, in the interim, we have put together a system that serves the needs of our subscribers.

The biggest subscriber complaint was slow response to e-mail inquiries along with many "I didn't get my magazine" complaints. In order to address these problems we put e-mail processing at the top of our list along with adding a descent subscription status inquiry capability for subscribers.

When we took over subscription processing we had just mailed a magazine. Thus, we had about 30 days to get our new system written and working. This limited our choice of solution to something that could be implemented quickly. Here is what we did.

First, we implemented a new subscription entry system using the Progress database. This includes new entry screens optimized for subscription fulfillment only. This minimizes data-entry keystrokes and eliminates common data entry errors.

The database consists of three record types:

- **Name/Address** - a generic name and address record used for subscribers as well as accounts used to pay for subscriptions.
- **Subscription** - a record that shows the current status of a particular subscription. It includes pointers to the associated name/address records.
- **Transaction** - a record that records a change in status of a subscription. Status changes include renewals, payments, suspends and address changes.

This database structure was chosen to minimize implementation time and contain enough history to satisfy both our auditing requirements and allow us to retrieve any information that we left out of the subscription record due to a design error. (Yes, there have been some and the transaction records have bailed us out.)

While Progress is capable of doing reports, we have elected to use awk as our report writer. There are three reasons for this:

1. Awk is portable. That is, we can salvage awk code to include in another system that does not include the Progress database.
2. Testing is easier. If you select records in Progress and then test in awk you have quicker turn-around.
3. What we call reports are such things as:
    - Invoicing and renewals via e-mail
    - Paper invoicing and renewal mailings

   Working at the awk and shell level makes it easy to customize our output to, for example, send e-mail and then print forms using the same record selections.

Another important part of the system was the ability to let the subscriber check their own subscription status. We do this by running boa, a mini-webserver on our database machine. From the *Linux Journal* web page, a subscriber can enter their subscriber number and last name and get the current status of their subscription, including address information, immediately. This means fewer "is my address correct" inquiries in e-mail or via telephone and a higher subscriber confidence is us.

The interface between the inquiry form and getting the information back to the inquirer is as follows:

- cgi-script on www.ssc.com formats a query for the subscription database and sends it along to the database server

- boa on the database server runs a cgi-script that builds a progress database query in the Progress database
- results of the query are displayed on www.ssc.com

This project isn't complete. We still have automation work to add as well as web interface changes. For example, a reader can subscribe, renew a current subscription and pay an invoice from the web page. Unfortunately, while automated from the point of view of the user, we still process these transactions manually.

All that said, here are the types of reports that we currently do:

- Subscriber count
- Subscription obligation (how many magazines we owe subscribers)
- Subscriber invoicing
- Labels and bag tag information
- Subscription renewals

4.2 Distribution

The distribution system keeps track of bulk orders for magazines. These orders fall into three categories:

1. Shipments to retailers and advertisers
2. Shipments to distributors
3. Shipments to promotional events

The first two items, from the system's point of view, are the same. These are standing orders that need to produce shipping paperwork and invoices every month. The default action is to produce orders with the same quantity as the previous month. The distribution manager can modify quantities and also must enter returns.

Promotional events are when we send magazines to some sort of gathering. This could be a conference, user group meeting or trade show (such as this one). The only difference between this item and the first two is that it is a one-time shipment. It is flagged as such when it is entered and requires no manual intervention after its initial entry.

4.3 www.linuxjournal.com and interactive.linuxjournal.com

We started our web site in 1994 on a 486DX4/100 with 16MB RAM. Running Linux and Apache, of course Our plan was to see how well it did up to 10,000 hits/day and then decide how much hardware we needed to handle additional load.

Well, we got busy and just weren't watching the stats very closely. Before we got back to it, we were getting well over 100,000 hits/day and the machine was very happy. The only reason we upgraded at that time was because we needed to install a larger disk.

Recently, we made a major change to the *Linux Journal* web site. The new site uses a database for the articles so the update process is much easier. Additionally, we have moved the site to a co-located AMD K6 system to improve performance. The pieces that make the site work are:

- Stronghold

- MySQL
- Perl
- Python

4.4 Linux Journal Store

There are a few products related to *Linux Journal* that we want to make available. As we no longer do retail for the SSC products we wanted something that could be self-contained and let the fulfillment be done by someone else.

We decided to use mini-Vend to do this task. We realize the store needs some work at this time but all the pieces we need are offered by mini-Vend—it is just a matter of prioritizing.

5. General Office Use of Linux

Besides the expected general correspondence and e-mail, we have specific areas where we are using Linux.

5.1 Accounting

In the past, accounting was done with a combination of the order entry and shipping code I wrote in Progress and then QuickBooks to do the actual accounting. Between these two systems were a lot of printouts and manual entry.

We are now in the process of installing Appgen's accounting packages. We expect this will integrate all the functions together and get rid of one of those two Windoze systems.

Spreadsheets are also very popular with accountants. Our accountant's spreadsheet of choice is StarOffice. She has managed to produce a spreadsheet the size of a wall if you elect to display all the detail in the company budget.

5.2 Forms

All of our forms have been done in troff. Many are 10 or more years old and there is no reason to change the way we produce them.

5.3 Internal Web Server

While we have a history of close to 20 years of troff-based documentation, we are attempting to get the ordinary stuff up as HTML pages on an internal web server. In general, we don't want everyone to have to learn HTML or troff. That meant we needed a documentation format that would result in HTML for the internal pages but could be written easily by anyone.

We selected a program called YODL (Yet Oneother Documenting Language). It was written up in *Linux Journal* #43 from November, 1997. It's claim to fame is that it is extremely easy to learn and offers translators to HTML, troff (ms macros), LaTeX and even ASCII text.

For the documents that already exist in troff and don't need re-writing, the program Python pymm2html does a good of translation.

6. Where Linux Falls Short

All in all, Linux has offered an amazing set of tools for us to work with. While some programs may have been available for other platforms to address our specific needs (subscription fulfillment, for example), the ability of Linux or any UNIX-like system to roll with the punches has made all the difference.

We have the source code to everything we use on Linux except the Progress database. This has made it possible to build the right solution from available pieces.

In the following list, I am not picking on Linux—I am picking on vendors that have not yet ported their products to Linux. Linux is capable and, in most cases, should be the platform of choice for vendors.

- Scheduling with resource allocation—We currently have a program called Turbo Project that runs on Windoze. It isn't exactly what we need. We will either find something for Linux or write what we need for Linux.
- Image editing—While the GIMP is very powerful, my layout person says there is stuff that PhotoShop will do that it won't. One, probably the most important, is produce CMYK output.
- No magazine layout program—Quark or Adobe's new InDesign needs to be ported to Linux.

7. The Future

We are here with Linux for the long run. The tools that didn't exist anywhere we built. While there are still some additional tools we need, sticking with Linux and either waiting for them to appear or writing them ourselves is clearly the best path for us.

The tools we still need are:

- Subscription Fulfillment
- Layout
- Vector Graphics
- CMYK support in bitmap graphics

That's about it. As more and more software vendors take Linux seriously, the remaining limitations will melt away. With Linux's inherent reliability and the ability to customize it, it as to be the platform of the future.

CONTENTS

Publishing a Magazine with Linux

Phil Hughes, Publisher
Linux Journal
P.O. Box 55549
Seattle, WA 98155
<phil@ssc.com>

September 13, 1999