news@UK

The Newsletter of the UK UNIX Users Group
Published electronically at `http://www.ukuug.org/newsletter/`

Volume 11, Number 1 ISSN 0965-9412 March 2002

Contents

News from the Secretariat

Jane Morrison

Firstly I have to thank everyone who kindly sent in the subscription payments so promptly. We have received a record number of early payments. Those remaining outstanding will be chased next month and any still outstanding after March will not receive the June Newsletter.

The Winter conference and tutorial was very successful. Attendance numbers were high and from the feedback on the questionnaires the event was well received. The papers from the conference are now available on UKUUG web site.

The next event is the Linux 2002 Developers' Conference being held at Bristol University from 5th - 7th July, details, including a web form for potential speakers' submissions is available at `http://www.ukuug.org`

Your Council is planning a meeting in early March, probably in London and hopefully avoiding any possible train/tube strikes!

UKUUG Secretariat
PO Box 37
Buntingford
Herts
SG9 9UQ
Tel: 01763 273475
Fax: 01763 273255
`office@ukuug.org`
`http://www.ukuug.org`

The 2002 Usenix LISA Conference

Paul Anderson

The 16th Usenix conference on "Large Installation System Administration" is due to take place in November this year, in Philadelphia. This is the largest international Systems Administration Conference, with over 1700 attendees, and four or five parallel tracks last year. The conference starts with three days of workshops and tutorials, presented by some of the most famous names in their field. This is followed by three days of refereed paper presentations, invited talks, BOFs, and other sessions.

It is always good to see papers from the UK and two of those from last year were presented at the recent UKUUG Winter Conference. The LISA programme committee is currently looking for submissions of papers for this year, and the deadline for abstracts is April 29th. Unlike most academic conferences, many LISA papers are from working system administrators who are simply sharing the ideas that they have used and developed in the course of their own work. Anyone who thinks they may have an idea for a paper is welcome to email me, if they would like to discuss what is involved.

The LISA web site contains some points designed to demonstrate to managers the value of sending their staff to the conference. Full-time students are eligible for a Usenix stipend which usually pays the full cost of attending - student papers are particularly welcome, and acceptance of a student paper normally includes the offer of a stipend.

Details of the event can be found at:
`http://www.usenix.org/events/lisa02/`

Who Says Computers Are Becoming More Intuitive?

Stephen L Talbott

It's amazing to see the popular credence still given to the notion that (in the words of a news story about computers and education): "Computers are getting more and more intuitive".

Not only is this untrue; it never will be true, pretty much on principle.

What people who say this seem to have in mind is that they themselves have become more comfortable with computers over time, or that some particular thing they used to do with difficulty can now be done quite simply. This is common enough, but has little to do with computers becoming more intuitive. Particular tasks may be getting easier, but it's also the case that overall tasks are becoming vastly more complex. The old, now-more-intuitive particulars are just a tiny part of a much greater and less intuitive enterprise.

Take someone off the street with no computer experience and try putting him to work using such basic tools as Word, Excel, Access, and Internet Explorer. Does this require less training than was required for a new user to come up to speed on the computer tasks typical of fifteen years ago? When something weird happens during his web browsing, ask yourself whether he could have any clue as to whether the problem originates with his operating system, windowing system, shell, keyboard or mouse, web browser, ISP, or the web site he is currently viewing. I know:"He shouldn't have to worry about any of that. It should all work together transparently". Sure.

Since I left the high-tech business several years ago, I have been concerned with matters other than the tools I am required to use, and have come to begrudge the time put into wrestling with the tools. Generally, I just avoid the time and "make do"; I'm much more interested in my work. But it has been dismaying to see how rapidly this stance makes a technical "dinosaur" out of me, putting me at the mercy of complexities I don't understand.

Once upon a time, an authentication problem on my Unix system was routinely solvable by editing a single line in the `/etc/passwd` or `/etc/group` file. By contrast, I recently encountered an authentication error that shut me out of essential operations on my computer for no apparent reason. But now – thanks to a vastly more complex security environment in the world of computing – there are not only "shadow password" mechanisms, but entire suites of authentication modules. Glancing through the (inadequate) documentation for these, I quickly realized that dealing with the problem in my old fashioned way – by figuring out exactly what was going on internally – was not something I was willing to spend the next couple of days doing. Instead, I wasted a few hours flailing about, doing anything and everything I could think of as a possible cure or workaround until, somehow, I was able to get back to my real work.

Ask yourself: is it more intuitive to design what passes for a first-class web site today than it was in 1995? Or how about designing what passes for a first-class oral presentation with visuals? Yes, you can add this or that feature to your talk much more easily than in the past. But are the technical demands now placed on you for the presentation as a whole easier to fulfil than before? Are you spending less time on the form of the presentation relative to the depth of its content than before?

Again, ask yourself: is there less need for technical-support organisations today than in earlier years? And is the gap between the support people feel they need and what they actually get any smaller than before?

The dynamic in all this isn't hard to grasp. The whole glory of the high-tech industry is the speed at which it turns the latest powers at its disposal toward ever more complex and sophisticated achievements. And the prevailing tendency of society is to re-shape our activity around these latest achievements. So the governing rule is this: we will always find ourselves struggling with the maximum amount of unfamiliar (read: counter-intuitive) complexity we can endure, if not a little more. This rule is less related to technology as such than it is to the values that drive our lives.

In sum, we will continue to face intuitively opaque digital technologies for reasons of our own choosing, whether because we think they are cool, or because we have reconceived the challenge of our own lives in the kind of technical terms that require us, for progress' sake, to embrace whatever is newest and most technically advanced.

You are free to reply, "Well, at least our tools are getting much more powerful, so we get more done with less effort, even if the general level of technical challenge in our work remains unchanged".

But I wouldn't press this if I were you. We certainly get more computation done than ever before, but the relation between this computation and economic productivity remains widely debated. More importantly, whether we are "getting more done" depends on what we want to accomplish in the first place. And our powers of computation themselves tend to bias us toward certain (computable) sorts of accomplishment. But what if this bias runs counter to our deepest needs? That, however, is another essay.

Related articles:

See articles listed under "Fundamental deceit of technology" in the NetFuture topical index: `http://www.netfuture.org/inx_topical_all.html`

"Speeding toward Meaninglessness": `http://www.oreilly.com/~stevet/meditations/speedup.html`

This article is reprinted by permission from NetFuture issue 128.

This article is reprinted by kind permission of NetFuture, and was brought to our attention by Mick Farmer.

Clustering

Owen Le Blanc

Clustering is a hot topic nowadays, though of course the term refers to a broad range of software which performs many different tasks: some clusters provide high performance, others high availability, and others just try to share existing resources which might otherwise go unused.

The third ACIS conference on a variety of software topics will be held in Madrid, 26-28 June; this year it will concentrate on clustering.

Beowulf is perhaps the most widely know software for creating high performance clusters; the Beowulf project was started by Donald Becker, who did an enormous amount of work writing Linux drivers for various network cards. Beowulf is now about 8 years old. It has a very active user community, with mailing lists, a HOWTO, books, and other documentation. Recent developments in the Beowulf world include software which is intended to simplify the management of large clusters.

Mosix is a different project which helps you construct a cluster in which resources from several machines are shared. It comes in two versions: a patch for Linux kernels (K-Mosix) and a user-level version (U-Mosix) which can build a heterogeneous cluster including, for example, Linux, BSD, and Solaris machines. K-Mosix allows processes to migrate transparently across machines, while U-Mosix can be tuned for GRID computing and other cluster applications.

Linux Virtual Server (LVS) is software which enables a group of machines to appear as a single large server; this allows high availability in a very scalable server. The software can be used in three different ways: it can use Network Address Translation (NAT), IP tunneling, or direct routing; each of these methods has its own strengths and weaknesses.

One difficulty in creating a working cluster is arranging to share filestore across machines and platforms. The well-known Network File System (NFS) has a number of weaknesses and limitations, and has sometimes been described as 'past its sell-by date'. Some clustering software (e.g., Mosix) comes with its own file system, and there are shared file systems which can provide better performance and better security than NFS. Some of these, such as Sistina's Global File System (GFS), are best in a cluster where access is controlled, while others are designed to be able to operate more securely in an open network environment: so, for example, OpenAFS or Coda.

Among the tools which can be useful in setting up and managing clusters are heartbeat from the High-Availability Linux project and the Logical Volume Manager (LVM) from Sistina.

The company Alinka, which sells software for installing and managing Linux clusters, sponsors a weekly Linux clustering newsletter, which contains weekly summaries of developments in many important projects related to clustering. You can view a copy of the latest newsletter on the company's web site, which also contains instructions for subscribing and unsubscribing.

References:

ACIS conference
`http://www.ls.fi.upm.es/snpd02/`

Beowulf
`http://www.beowulf.org/`

Mosix
`http://www.mosix.org/`

Linux Virtual Server (LVS)
`http:/www.linuxvirtualserver.org/`

Sistina
`http://www.sistina.com/`

Open AFS
`http://www-openafs.central.org/`

Coda
`http://www.coda.cs.cmu.edu/`

High-Availability Linux project
`http://linux-ha.org/`

Alinka
`http://www.alinka.com/`

Perl tutorial review

Roger Whittaker

As in previous years, a technical tutorial was run in conjunction with the Winter Conference. This year's tutorial was given by Simon Cozens, a noted expert on Perl and is the author of 'Beginning Perl' and a co-author of 'Professional Perl Programming', both published by Wrox Press.

Simon Cozens proved to be a confident and amusing speaker who was clearly in command of his material. The tutorial notes were well produced – the presentation slides were also very readable and elegant (created incidentally in magicpoint: an excellent tool for presentations of this kind).

(One small negative criticism of the notes: some of the examples were intended to process data from a file or from the output of a command: examples of the data to be processed were shown in the slides but not in the notes.)

The tutorial was billed simply as a ‘Perl tutorial’: there were one or two gasps from members of the audience as they opened the notes and, seeing the code, feared that the standard would be beyond them. In a way, some of these fears were partly justified. Certainly it would have been impossible in a tutorial of this kind to instil in all participants the kind of understanding necessary to create the code in the examples again, or even to be clear afterwards about the working of all those examples. Clearly there were very widely differing prior levels of Perl knowledge among the audience.

What the tutorial could do, however, was to give a good introduction to what Perl can do by exhibiting practical examples of various different kinds. At the same time the examples, although at different levels of difficulty, were of the type to encourage in participants the confidence to go home and try them, attempt to modify them and gain the real hands-on knowledge which so often needs this kind of kick start to get under way.

Whether the speaker convinced all those present that such hands-on knowledge of Perl is really worth the effort is perhaps another matter. This writer, for one, would need a good deal of convincing that for all but the most specialised tasks there is any need for Perl’s obscurities and unreadability now that a comparable scripting language with the crystal clarity and elegance of Python is available. This personal view of course is a matter of taste, if not a cause of religious warfare, and I await the inevitable responses with some trepidation.

Simon concluded with a question and answer session: among other things he discussed the changes and features to be expected in the forthcoming perl 6 release.

UKUUG LISA Winter Conference 13th and 14th February 2002

Roger Whittaker

The Winter Conference, together with the associated tutorial, were held this year in the School of African and Oriental Studies in London.

The Conference was preceded by the Perl tutorial given by Simon Cozens: the Conference proper began after lunch on the first day.

It would be superfluous to summarise the talks in detail here as the complete proceedings are included on the CD issued with this issue of the Newsletter.

The location was a good central one (given that a London venue was chosen this time) and close to bookshops and the hardware suppliers of Tottenham Court Road. As usual, we had our own bookshop in the shape of O’Reilly’s Josette Garcia, whose cheerful presence is so often a very welcome addition to our events.

The lecture room itself, I am afraid left a little to be desired in that it proved difficult to control the heating, which meant that the room was rather too hot most of the time. The seats were not the most comfortable lecture theatre seats I have come across – but probably very good for keeping students of Arabic and Chinese awake during their classes.

The standard of the speakers was good throughout, both in content and delivery: all were audible and clear. The slides and notes were uniformly well produced.

The topics covered were all (apart from Wayne Pascoe’s BSD talk) descriptions of particular practical projects. The first two talks by Paul Anderson and Jonathan Hogg described two current approaches to the problems of large scale system configuration and administration. Paul Anderson described LCFG: the configuration and installation system which has been under development at Edinburgh since 1993. Jonathan Hogg described the perhaps more ambitious

Arusha project, which aims to provide a collaborative Internet-wide framework upon which local system administration setups can be based.

The second pair of talks on the first day were both descriptions of good practice applied to a particular environment and set of needs. Peter Polkinghorne described his experience in integrating Windows and Unix in the particular circumstances of a legal environment with an interesting historical story. He evaluated and described the various possible solutions which he had considered and explained his reasons for making certain particular choices. Alain Williams' talk (subtitled "Making it easy, getting it right") was a description of how good practice and clear thinking in a production environment can reduce unreliability and support requirements by an order of magnitude.

Thursday's first two talks were interesting case-studies of solutions to problems in the academic environment. Mike Wyer described the problems involved in setting up a secure and reliable system for invigilation of paperless examinations to be taken on standard Linux workstations in a university department. Jim Davies described a powerful web-based system which is in use at Oxford to administer the Software Engineering Programme. Jim showed how a careful design philosophy had led to a fundamentally original way of solving the problems involved, including the development of a powerful new query language.

Wayne Pascoe's talk was a powerful piece of advocacy for FreeBSD, with interesting comparisons made between FreeBSD, the other forms of BSD and Linux.

Stuart McRobert entertainingly described various improvements in hardware and networking technology, particularly in the field of Gigabit and 10 Gigabit ethernet and wireless networking. He also spoke about the future of SunSITE Northern Europe at Imperial College.

The conference concluded with Andrew Findlay's talk on LDAP and security: a very clear description on the use of openldap as an information and authenication service on a large network.

Eric Raymond: talk at City University, London, 27 February 2002

Immo Hueneke

The audience voted mainly for two topics of interest (out of the five on offer):

Anthropology of the Open Source developer community
Business models and how to make money out of all this

Introduction

The most revelatory experiences in your life occur when the universe confounds your deeply-held convictions. In 1993, this happened to Eric. He had absorbed traditional software engineering "wisdom", which dictated that developers needed to be strictly controlled. At that time, the first Linux CD hit his doorstep (because he had already contributed something to Free Software, which was included in the distribution).

He was somewhat dubious about it, because he knew that it was the result of the collaboration of thousands of people. Also he had to buy a CD-ROM drive before he could install it! He was immediately impressed by what was evidently a world-class OS (and this from someone who was a daily user of System V Unix). All the familiar tools were there. No crash-recovery tools - evidently not needed!

Commercial software works on slow release cycles, because there are only a few people available to fix bugs. Yet the aim of exposing zero bugs to the user is never achieved. Here, on the other hand, was Linux, created in an uncontrolled process, yet evidently producing a world-class product.

Eric dived into Linux development and, over the space of three years, assembled the ideas that went into the paper "The Cathedral and the Bazaar". Basic principles embraced by the Open Source community:

Be open to input at all times
Release early, release often
Acknowledge contributions generously

Essentially, this encourages as much exposure as possible of the code, in order to improve it. This winning strategy is based on source inspection as well as intensive testing by a large user community.

The developer community working on Linux, Apache, Perl and similar projects rediscovered the power of peer review. This mirrors what goes on in any other engineering discipline.

Questions from the audience

Q: Does the Linux kernel still follow these principles, five years down the line?

A: The development model for Linux as a whole is indeed well decentralised, but the kernel itself is centralised around Linus Torvalds - and he has reached his scalability limit!

Q: The last Debian release was WHEN?

A: On the one hand, there hasn't been one in two years - but apt-_get largely overcomes the problem. At the package level of granularity, there is no need for version increments.

Q: In 20-50 year time-frame, will all software be Open Source?

A: There are unusual cases, in which closed source actually makes business sense - but in less than 20 years, Eric expects the majority of software to be Open Source.

Q: At what point does the main developer of a project decide whether to take a patch?

A: The open and closed source development models are both bottlenecked by the availability of geniuses!

Q: Does the Open Source model really extend to unsophisticated users?

A: In one respect, developing for non-technical users is different. We have no very good mechanisms for collecting quality feedback from end-users who do not think like hackers. Most technical users are able to deal with interface complexity and baroque bug-reporting interfaces. Secondly, we need idealistic determination that it's worth collecting the user feedback.

Q: Would Linux benefit from being controlled by a foundation (like Apache)?

A: There are some proposals - such as the patch panel penguin proposal(?) initiative. Some scaling issues were foreseen about four years ago, but nothing has been achieved to alleviate it. Linus is not good at delegating and that is something he has to learn. In fact, he has managed to alienate some key collaborators by taking apparently arbitrary decisions.

Q: What is really meant by scalability?

A: Take aeroplanes as an example - performance depends on the ratios between wing area, engine power and fuselage volume. They don't all scale at the same rate. In terms of software, not all factors scale linearly with project size. Problems are usually resolved by arguing a lot!

Q: Are there other practices from engineering (apart from peer review) that could usefully be copied by software practitioners?

A: Some practices are already being copied inherently - educational structures, career structures, professional societies - but there may be some. Suggestions from the audience?

Destructive testing perhaps

Formal mentoring

One of the transitions observed over the past 25 years is from implicit to explicit culture. Development practices were observed and absorbed by a sort of osmosis. Nowadays, ideas and attitudes are reinforced by explicit models (e.g. the Hippocratic Oath for medical profession). Analogously, normative documents such as manifestos (the GPL for example) have to some extent replaced the informal mentoring process.

Q: As a major developer figure, do you really want "normal" people involved in the process? If so, what can you do to make it easier for them to become involved?

A: Yes. Find a project that requires user interface improvement and which welcomes input from non-technical users. For maximum impact, find someone doing a basic productivity application (e.g. word processing, presentation software) and offer them your services as a UI designer.

Q: Could the Open Source community come up with multimedia tools comparable to Flash or Director?

A: Certain that this will happen sooner or later. In the same way as toy programs grew into serious communications and developer tools over a period of six or seven years beginning in the early 1980s. These addressed the working needs of programmers - so in the early 1990s it became "let's build an operating system". Over the last four or five years, the community has begun turning its attention to applications (KDE, Gnome, ABIword and so on) because the earlier problems have been essentially solved.

Q: Isn't a great application or operating system the result of a clear vision, a pervading paradigm, controlled by a small number of minds? Example - Mozilla has an inconsistent UI.

A: You cannot easily originate coherent architectures by an Open Source process. Pick an architect - this is essentially achieved by electing a project leader. Architecture can be delegated to some extent to a small committee. In a Bazaar process, one person has to have ownership of the UI. For example, KDE decided to provide its developer community with toolkits that reinforce the guidelines. Another example: an audio editor called audacity. It has a beautifully simple, powerful user interface, yet it was developed by a Bazaar process. There was a UI "czar" to safeguard the integrity of the UI. Incidentally, learn to love domain experts with deep intuitive understanding of the application area in order to improve UIs. For example, the sound application was heavily influenced by a pianist.

Q: How does it all work - what sustains the process? See next part of the talk.

Anthropology

"Homesteading the Noosphere" was written in response to an interesting contradiction in the behaviour of Open Source developers. Eric was applying the principles of social anthropology fieldwork to the Open Source community. What motivates developers, what makes it psychologically sustainable (as opposed to economically sustainable, which is covered in the next part)?

These principles guide researchers towards identifying apparent contradictions. The one spotted by Eric was this: on the one hand, all forms of Open Source licence posit a world in which everyone is free to change any software at any time. This would lead to a world consisting of code bases that continually fission in different directions. In reality, this rarely happens - projects are rarely cloned and taken in different directions, at least not without first making strenuous efforts to persuade the "lead developer" of a project to incorporate the changes in the original codebase. In effect, there is a kind of unwritten ownership. The three rights of ownership appear to be:

You can own a project if you found it.
You can own a project if the owner transfers ownership to you publicly.
You can own a project if the owner has disappeared and you announce a willingness to acquire it (typically through newsgroups) followed by a suitably long interval for responses.

Ownership confers on you the right to make canonical releases of the software (in fact, it is a requirement to do so on a fairly regular basis).

Eric considered that this pattern was strangely familiar. It is very closely similar to John Locke's theory of common law property rights - land tenure in particular. The ownership of land is asserted in the following ways:

By homesteading it - putting a fence around it and defending your title if need be.
By transfer of title, formalised in a deed.
By "adverse possession", in which you take over a piece of land that has evidently been abandoned and occupy it for seven years and one day.

Land is tangible, software projects are not. Land is exhaustible, the potential number of software projects is infinite. Land is a "rivalrous good", while software can be copied indefinite numbers of times. So why should land tenure be so closely analogous to software project ownership?

Humans are adaptive organisms, so they tend to re-use known mechanisms. The question to ask is "what function does property serve"? Can this be generalised to the function of project ownership?

An insightful observation is that "your dog knows where your property is" - and evidently acquires this knowledge through observation of the master's behaviour, because dogs can't read deeds! There is a strong homology between human property and dog territoriality. The function of the latter is well known. Dogs have a property system in order to minimise the amount of intra-species violence.

This is common to all fierce predator species. Every individual requires a certain area to sustain itself and reproduce. Predators who wander around at random will tend to fight each other. This reduces the survivability of the species as a whole. Territoriality is the system that nature invents to circumvent this. Rules are simple:

Individuals have to mark their boundaries in some way.
Any individual crossing the boundary will be attacked.

It is possible to demonstrate that this is the optimal strategy for reducing intra-species violence. Hence human property systems, which have evolved in just about every society, have a clear function and cannot be arbitrarily modified.

Property systems also have the function of protecting a yield of a critical survival good. A great deal of nonsense has been written about cultures that have no land tenure system. Bushmen, for example, don't protect real estate in the desert, which yields nothing. However, they have a tenure system for waterholes! So property systems are only ever constructed around commodities whose benefit exceeds the cost of defending them.

So, if hackers have a quasi property system around projects, what is the "critical survival good" being yielded by the projects? Repute among the peer-group! This explains why it's such a taboo to remove author labels and history lists from source modules. It's such a taboo that we are not even aware of it.

Why is repute among the peer group important? It spills over into the real world - your reputation as a developer opens doors to better jobs, book deals, lecture tours and so on. Another important use of reputation is an effective way to get others to collaborate with you on problems you want to solve.

Human beings are deeply wired to play "social status" games. Anthropologists call relate this the to "environment of ancestral adaptation". In prehistory, human beings existed in small groups of up to two dozen individuals with few technological aids to access resources - so the successful people were those who were good at enlisting the help of others: building and maintaining collaborative networks.

Questions from the Audience

Q: To what extent does Open Source resemble a market economy, to what extent is it a gift economy?

A: Please read “Homesteading” and “The Magic Cauldron”.

Q: When is the fourth paper coming out?

A: “The Great Brain Race” is coming - but no release date has been set.

Q: Why has there been disagreement between Eric and the Free Software Foundation, when both seem to be arguing in favour of the same thing?

A: Differences, broadly speaking, are based on the fact that Richard Stallman is on a moral crusade, while Eric isn’t. Richard has actually stated that he would rather use inferior free software than superior proprietary software. Eric is interested in engineers getting good engineering results, people achieving their own purposes in the least inconvenient way.

Q: Many commercial projects are less successful than the original community projects - Mozilla for example is producing results more slowly than Apache.

A: In many cases, the sheer scale and complexity of the project might be to blame, rather than the motivation and cohesiveness of the developer team.

Q: Does the transition to an explicit culture defeat innovation?

A: Eric hasn’t seen any evidence of this. For example, adopting a penguin emblem hasn’t slowed innovation in Linux. In fact, creativity may be liberated by explicitly codified rules of behaviour.

Q: Selling the concept to “suits” is difficult in the face of a mindset that knows what it likes because it likes what it knows.

A: You want my “advocacy” talk!

Q: Why does it matter?

A: If you want to change the world, you have to co-opt the people who write the cheques! For example, it was possible to persuade the people with money and power that it was not in their interest to tie down the Internet with all kinds of proprietary IPR.

Q: In what cases is software better off being proprietary?

A: Read “The Magic Cauldron”. Important indicators are: when the secrecy value of the exclusive use of an algorithm exceeds the value of peer review in debugging the software - typically very early in the life cycle of software technologies. Example: biometric verification. Currently, the engineering challenges of making those algorithms work on a large scale and reliably are formidable. In future, a palette of available algorithms will come into the public domain and so the challenges will be those of software quality rather than algorithm performance. Other examples are certain game titles.

(no more time)

News from O’Reilly

Josette Garcia

Python News

Frank Willison Award: this year’s Frank Willison Award for Contributions to the Python Community was presented to Andrew Kuchling at the Tenth International Python Conference. Guido van Rossum, one of the award’s judges, said that “besides being a prolific programmer who has contributed to many corners of the Python implementation, Andrew has done numerous writing projects related to Python.” For more details about this award see the Press Release:

http://press.oreilly.com/willison_award.html

Tenth International Python Conference
http://www.python10.org/

New book announcements

Building Wireless Community Networks

Implementing the Wireless Web
Rob Flickenger
ISBN 0-596-00204-1
138 pages
Building Wireless Community Networks offers a compelling case for building wireless networks on a local level: They are inexpensive, and they can be implemented and managed by the community using them, whether it's a school, a neighbourhood, or a small business. This book also provides all the necessary information for planning a network, getting the necessary components, and understanding protocols that you need to design and implement your network.

Physics for Game Developers

David M. Bourg
0-596-00006-5
344 pages
Colliding billiard balls. Missile trajectories. Cornering dynamics in speeding cars. By applying the laws of physics, you can realistically model nearly everything in games that bounces around, flies, rolls, slides, or isn't sitting still, to create compelling, believable content for computer games, simulations, and animation. Physics for Game Developers serves as the starting point for enriching games with physics-based realism.

Web Security, Privacy and Commerce, 2nd Edition

Simson Garfinkel with Gene Spafford
ISBN 0-596-00045-6
786 pages
This much expanded new edition explores web security risks and how to minimise them. Aimed at web users, administrators, and content providers, Web Security, Privacy and Commerce covers cryptography, SSL, the Public Key Infrastructure, digital signatures, digital certificates, privacy threats (cookies, log files, web logs, web bugs), hostile mobile code, and web publishing (intellectual property, P3P, digital payments, client-side digital signatures, code signing, PICS).

Python and XML

Christopher A. Jones and Fred L. Drake, Jr.
ISBN 0-596-00128-2
384 pages
Python is an ideal language for manipulating XML, and this new volume gives you a solid foundation for using these two languages together. Complete with practical examples that highlight common application tasks, the book starts with the basics then quickly progresses to complex topics, like transforming XML with XSLT and querying XML with XPath. It also explores more advanced subjects, such as SOAP and distributed web services.
http://www.oreilly.com/catalog/pythonxml/

Designing Large-Scale LANs

Kevin Dooley
ISBN 0-596-00150-9
400 pages

This unique book outlines the advantages of a top-down, vendor-neutral approach to network design. Everything from network reliability, network topologies, routing and switching, wireless, virtual LANs, firewalls and gateways to security, Internet protocols, bandwidth, and multicast services are covered from the perspective of an organisation's specific needs, rather than from product requirements. The book also discusses proprietary technologies that are ubiquitous, such as Cisco's IOS and Novell's IPX.
`http://www.oreilly.com/catalog/lgscalelans/`

Free BSD 4.5

Sam Smith

If this was just a review of FreeBSD 4.5, and I just mentioned that it comes on DVD, then the review would have missed the point. There is a significant difference, when it comes to using the discs, between the CD and DVD distributions.

Since 4.5 is a release from the stable branch of FreeBSD, there's nothing radically different from 4.4. There's support for IDE drives bigger than 137Gb, various improvements in different filesystems, the linux emulation is better, and IPF now supports both IPv6 and storing NAT state across a reboot. It's better, more functional, but nothing big and new has been added.

However, while the install kernel on the DVD has more devices included than the floppy install kernel, FreeBSD itself is identical. The installer is the same, and once installed it's the same quality people have come to expect from previous FreeBSD releases, there's just more of it available while you're installing from the DVD. But that said, while the DVD is good at what it does (giving you access to lots of data if you want it), it generally stays out of your way and lets you get on with the install, with the media being largely irrelevant.

The package itself contains 2 discs, a single sided dual layered (7.6Gb in size) DVD, and a double sided disc with a 4.3Gb DVD image on one side and a standard CD image on the other. The first DVD image is a superset of the CD image, with the additions mainly being additional packages. This DVD is everything you need to install (including all 5,683 packages) and is also a emergency "fixit" disc. The sheer number of packages included on the DVD you boot from allows almost anything to be installed very quickly. The second DVD image contains a complete ports tree, including all distributable distfiles. This is intended to be mounted as /usr/ports; and makes the installation of ports a lot faster once you have installed any packages you want. You don't have to swap DVD's when installing packages, they're either on that DVD, or they're not available. Similarly with the ports tree, once mounted, everything that could be there, is.

Even more BSD

On 14th January 2002, Caldera, who own the copyright to the original AT&T UNIX source code, released it under a BSD-style license. These old versions, are once again unencumbered and are freely available both from The Unix Heritage Society Archive, and are included on the first DVD. While their practical utility may be limited, they are of interest to all those who have interest in BSD and BSD history. Additionally, on the DVD, but not widely available is the FreeBSD repository and code from version 1.1.5 of FreeBSD – the final release before FreeBSD moved over to the 4.4BSD codebase, on which all of Free, Open and Net BSDs are based.

As bonus extras, the DVD contains the complete FreeBSD, OpenBSD and NetBSD CVS repositories detailing all changes since the repositories were set up, along with archives of FreeBSD mailing lists going back to 1993, which is a useful resource to have access to. The CVS repositories can be unpacked and updated using the usual methods of anonymous cvs and cvsup to get a fully populated and up to date repository. Making the set not only extremely comprehensive for FreeBSD, but containing significant interest for those of other BSDs as well.

Overall, while it is unlikely that the majority of people will use all of the content of the disc, there is a lot of useful content there for everyone, not just FreeBSD users, to get something out of the set. The CVS and list archives are useful to keep on the shelf until the next time they're needed.

Resources

FreeBSD Services Limited
`http://www.freebsd-services.com`

FreeBSD
`http://www.FreeBSD.org`

The Unix Heritage Society (for versions 3 and 4 of BSD)
`http://www.tuhs.org`

Perl for Web Site Management

John Callendar

O'Reilly & Associates

ISBN 1-565-92647-1
528 pp.
£ 24.95

reviewed by Joel Smith

I was rather confused when I started reading this book. The first chapter starts off at an extremely basic level covering such things as how to use Telnet, how to use vi and how TCP works. I really do not think that the book benefits from such a low starting point. If people are starting from such a low level, are they really going to buy a book entitled "Perl for Web Site Management"?

Again, in the second chapter, even though throughout the book John Callendar assumes that the reader will be using Unix hosts, the perl scripts are always named with a `.plx` extension. Now I know that under Windows/DOS perl scripts tend to have a three letter extension of `.plx` , but I found that this grated somewhat. Surely it would have made more sense to conform to the more common practice of perl scripts having a `.pl` extension? Since the readers are expected to follow the details of unix file permissions, they should be able to cope if they did need a `.plx` extension. The author does make plain that Unix systems do not require an extension, but I find it a surprising omission that he does not even mention that common practice is to name a perl script with a `.pl` extension.

The next few chapters deal with constructing fairly simple CGIs and scripts. As I read this, I felt that the book seemed to be covering the territory of "CGI Programming on the World Wide Web" and "Learning Perl". Where was the web site management? I was feeling less well disposed towards this book.

In the middle of the book, chapters 8, 9 and 10 dealt with writing scripts to parse web server logs. Why? I can't imagine that people are really going to roll their own scripts to analyse their access logs, when there are tools such as Analog out there `http://www.analog.cx/` . Although this was web site management, I was not overly impressed with its real world application.

From here, things got better. The book moved on to cover running a search engine for your site (SWISH-E), creating pages using templates, automatically generating links through the use of meta-information and writing your own perl modules.

The book finishes up with examples of generating web pages through cgi scripts, user registration and management initially via `.htaccess` and `.htpasswd` files and then expanding to using DBM files to hold the user data. Although throughout the book there are various discussions on security, I think that these chapters should have a far greater security emphasis. The scripts used as examples are probably safe enough, but since this book is encouraging relative novices to delve into writing their own scripts, I feel that the security implications should be discussed in much more depth, particularly where the scripts are likely to be exposed to external users.

I read the blurb on the back of the book in more detail, and finally noticed the statement "Assuming no prior programming background, this book teaches you how to use Perl and other open source tools". This explains my earlier confusion. I still do not think that the book really addresses its market. A perl novice is not going to be attracted to the title "Perl for Web Site Management" as there is no indication that the book is a beginner title. Yet at the same time someone with experience of perl is going to find much of the book irrelevant.

If you accept that the book is teaching the reader to use perl from scratch, then it accomplishes that task quite well, and also brings in CPAN frequently. Indeed, many of the examples are followed by information about CPAN modules which can accomplish the task in a better, more secure or more efficient way. The example scripts are given to illustrate perl concepts, rather than as implementations to follow. Early scripts cover simple HTML parsing which can lead to a host of problems. These potential pitfalls are mentioned, and `HTML::Parser` is introduced in later scripts. The danger is that someone dipping into the book for examples to use might pull out one of the less secure scripts to modify.

My biggest criticism about the book is that it does not really know who to address. I do not think I would recommend this book to someone as the best way to learn perl, as there are better books for that purpose. I also think that too much of the book is given over to basics to recommend it for a more experienced perl user seeking extra ideas for web site management (you would be better off browsing CPAN). You might be happy buying a book for 10% of the content, but I believe you could get that content elsewhere and spend your money more profitably. I suppose the best market for the book would be as a companion to "Learning Perl" for someone wishing to rapidly get up to speed with both perl and web site administration.

DNS on Windows 2000 2nd Edition

Matt Larson and Cricket Liu

O'Reilly & Associates

ISBN 0-596-00230-0
334 pp.
£ 28.50

reviewed by Raza Rizvi

I recently reviewed the fourth edition of the classic 'DNS and BIND' (also co-authored by Cricket Liu) which includes a quite reasonable section on Windows 2000 DNS and dynamic update features. This book, although focused on the Microsoft provided DNS server, pays the complement back by leaning heavily on direct transplantation of text from 'DNS and BIND'. Whilst this is initially annoying, the target audience is likely to be different and the quality of the Windows specific parts is high enough to avoid leaving a bitter taste!

After a run through of DNS basics, chapter 4 starts on the setup of domains using the Microsoft DNS server. The example is based on the familiar 'Movie University' domain? and shows the DNS console and how the wizard is used to create the zone file. The contents of the file, `SOA` , `A` , and `MX` records together with the `PTR` are explained well and the many screenshots make it simple to follow.

The function of `MX` records and mail delivery are covered in chapter 5, and then we return to new text with a chapter on the configuration of the resolver on the Windows clients, 2000 obviously but also 95, 98, and NT. The server maintenance is covered in chapter 7 with explanations of the logging and the `named.ca` hints file.

WINS integration and co-existence is covered very well in chapter 10 together with other advanced features such as DNS Notify. Active Directory, arguably the biggest new feature, gets a half chapter that is concise and well written. The latter part of the chapter deals with Dynamic Updates.

Now in the last third of the book, nslookup and troubleshooting in covered in enough detail for the majority of likely scenarios. To finish the appendices cover the DNS RFC (1035), installation of the Microsoft DNS server, and conversion from BIND to Microsoft DNS.

All in all a useful book for Windows administrators but not required if you already own the fourth edition of 'DNS and BIND'. The concepts covered are the same and the DNS console makes it easy to work out what is required to set up files and their hosts.

Web Design in a Nutshell, 2nd Edition

Jennifer Niederst
O'Reilly & Associates

ISBN 0-596-00196-7
618 pp.
£ 20.95

reviewed by Daphne Tregear

Editon one of 'Web Design in a Nutshell' came out (in 1999) just as I was getting seriously interested in web design. It's been my constant companion ever since and my copy is looking rather battered. It was (and still is) perfect for the task given on the cover. It is comprehensive, compact and yet full of helpful discourse (Ms. Niederst talks so much common sense). It's very much a 'dip-into-when-you-need-it' reference book. It isn't the book to teach a novice how to write HTML or JavaScript, but does have useful reference pages for these languages (and for CSS) scattered throughout. There's no sign either of the ploys of less reputable publishers who pad their texts out to brick-sized proportions with reference material to help justify their price tag. Nor is it full of references to long-dead links (another of my pet peeves on web books).

'Web Design in a Nutshell' is useful whether you are designing conservatively or designing for the latest and greatest in browsers (Netscape 6 and Internet Explorer 5.5 at the time it went to press) and wish to make use of Flash, Shockwave and dynamic HTML. I'm usually found in the conservative end of the spectrum since I am employed in a department where Sun workstations with 8-bit graphics cards predominate. It's all very well her saying 'only about 5-7% of web users have 8-bit displays' but when they make up the majority of your immediate audience they have to be pleased.

The lack of pages printed in colour does make the 'Designing Graphics with the Web Palette' chapter more difficult to follow than it needs to be; a few colour pictures here would aid clarity greatly. Jennifer Niederst generously makes reference to Lynda Weinman's books, but not to her very useful web site `www.lynda.com` which has the benefit of colour illustrations of the principles Jennifer addresses.

'Web Design in a Nutshell' is particularly strong on using tables for design which, let's face it, are still the cornerstone of the majority of pages. I also found the illustration of how differently form elements appear on the major browsers invaluable. The whole book is peppered with useful tips, although the later edition suggests more caution with nonstandard web tricks.

Edition one was written during 1998. Edition two reflects changes which have occurred in the web during the last three years. These include mentions of the capabilities of later browsers and the requirements of changing web standards. The order of some of the material has been re-jigged, some material has been expanded and new bits have been added. Are the new bits worth buying the book again if you already have edition one? There's a useful chapter on printing from the web (covering printer-friendly pages and PDFs) and others on accessibility and internationalisation (topics increasingly important in web standards). More emphasis is placed on CSS. Character entities have been moved to an appendix. Flash and Shockwave get a chapter to themselves as does SMIL (Synchronized Multimedia Integration Language, apparently – hands up who's using it), XHTML (which we should all be using) and WAP and WML (oh dear...). On balance, I would say yes.

If you were only going to buy one book on web design, this would have to be it. Use it and Jakob Nielsen's thought-provoking essays on usability at `www.useit.com` and you can't go far wrong.

Shell Programming Puzzle: Prime Numbers

James Youngman

This issue's puzzle is to write a shell script (Bourne shell or Korn shell) which prints out (on stdout) all the prime numbers between 1 and 350, separated by whitespace. Entries may not use an external program whose main purpose is to produce prime numbers (one is installed in /usr/games/primes on the machine I'm writing this on, for example). Entries emitting non-prime numbers will be disqualified. There are no other restrictions on the contents of your shell script. The winner will be the shortest correct entry received (length being measured in bytes). The time taken to produce the result will not be taken into account, as long as it is finite! Entries by email to `puzzle@ukuug.org`, please.

Results of the previous puzzle - Christmas Tree Picture

James Youngman

The seasonal challenge of last issue's puzzle was to write a script which draws a picture of a Christmas tree.

There was an initial problem with the `puzzle@ukuug.org` email address; apologies to anybody affected by this - it's fixed now.

The winning entry for the Christmas tree puzzle is by Mike Jones, who in fact provided a black-and-white entry and a colour one; both appear below. Mike wins his choice of book from O'Reilly; thanks to O'Reilly for the donation of the prize.

```
plaintree.sh
------------

#!/bin/sh
cat>p<<+
%!PS
/l {rlineto} def
/m {rmoveto} def
/t {144 0 l -72 72 l closepath 0 50 m} def
newpath
```

```
216 360 moveto
t t t
fill
showpage
+


treestar.sh
-----------
#!/bin/sh
cat>ps<<+
%!PS
/d {def} def
/S {setrgbcolor} d
/c {closepath} d
/n {newpath} d
/f {fill} d
/l {rlineto} d
/M {moveto} d
/m {rmoveto} d
/t {144 0 l -72 72 l c 0 50 m} d
/s {5 0 l -5 20 l -5 -20 l c 72 rotate} d
/b {n rand 44 mod 265 add rand 100 mod 380
add 3 0 360 arc c 1 0 0 S f} d
n 216 360 M t t t 0 1 0 S f n 288 532 M
s s s s s 1 1 0 S f realtime srand b b b b b
showpage
+
```

Line breaks have been inserted in the above entries to allow them to fit within a single column.

The CD

David Hallowell

This issue's CD contains the proceedings of the Winter Conference, and the UNIX Heritage Society archives.

Contacts

Charles Curran
Council Chairman; Events; Newsletter
Oxford
Tel: 07973 231 870
`charles.curran@ukuug.org`

James Youngman
UKUUG Treasurer
Manchester
`james.youngman@ukuug.org`

David Hallowell
Website
Tyne and Wear
`david.hallowell@ukuug.org`

Alasdair Kergon
Events
Reading
`alasdair.kergon@ukuug.org`

Dr A V LeBlanc
Newsletter
Manchester
`owen.leblanc@ukuug.org`

Roger Whittaker
Schools; Newsletter
Borehamwood
`roger.whittaker@ukuug.org`

Jane Morrison
UKUUG Secretariat
PO Box 37
Buntingford
Herts
SG9 9UQ
Tel: 01763 273 475
Fax: 01763 273 255
`office@ukuug.org`