EDITORIAL
The User/Product Conundrum .................................. 1

LETTERS TO THE EDITOR .......................................... 2

USENIX NEWS
Board Meeting Summary by Ellie Young .......... 3
Call for Student Research Proposals ................. 5

SAGE NEWS
Back to the Future by Tina M. Darmohray .......... 6
Letter from the President by Paul Evans .......... 7
Where's the Ark? Dealing with SYN Flooding
by Shawn Instenes ........................................... 8
BayLISA 1996 by Bryan McDonald ................. 9
Requests for Proposals for SAGE booklets .... 10

FEATURES
Interview with Mike O'Dell by Rob Kolstad .... 12
Beating Bottlenecks in the Design of Distributed
File Systems by Ahmed Amer and Amr El-Kadi 14
Using Java by Rik Farrow ................................. 24

CONFERENCE REPORTS
Musings from LISA 10 by Rik Farrow .......... 27
Reports on LISA 10 ........................................... 29
Refereed Papers ............................................ 29
Invited Talks .............................................. 35
Works-in-Progress ......................................... 37
BOFs .......................................................... 38
Confessions of a USENIX Volunteer
by Idajean M. Fisher ..................................... 40
The Terminal Room: A Volunteer's View
by David J. Bianchi ....................................... 40
Notes on the LISA '96 Advanced Topics in
System Administration Workshop by Tina M.
Darmohray, John Schimmel, and John Sellens 41
Horror Story Contest ........................................ 47

BOOK REVIEWS
The Bookworm by Peter H. Salus .................. 49

ANNOUNCEMENTS & CALLS
1997 USENIX Technical Conference ................. 51
5th Annual Tcl/Tk Workshop ............................ 54
Conference on Domain-Specific Languages .... 56

ETC . . .
Local User Groups .......................................... 70
Calendar of Events .......................................... 72
Upcoming USENIX Events

USENIX 1997 Technical Conference/USELINUX: Linux Applications Development and Deployment Conference
January 6–10, 1997, Anaheim Marriott, Anaheim, CA

3rd USENIX Conference on Object-Oriented Technologies and Systems (COOTS '97)
June 16–19, 1997, Portland, OR
Program Chair: Steve Vinoski, Hewlett-Packard; Tutorial Program Chair: Douglas Schmidt, Washington University
Tutorial Submissions due: Feb 6, 1997; Paper Submissions due: Feb 12, 1997; Notification to Authors: Feb 25, 1997; Camera-Ready Papers due: May 6, 1997

5th Annual Tcl/Tk Workshop '97
July 14–17, 1997, Boston, MA
Program Chairs: Joe Konstan, University of Minnesota, Brent Welch, Sun Microsystems Laboratories, Inc.
Paper Submissions due: March 11, 1997; Notification to Authors: April 8, 1997; Poster Submissions due: April 22, 1997; Camera-Ready Papers due: June 3, 1997

USENIX Windows/NT Workshop
August, 1997, Seattle, WA. Program Chairs: Mike Jones, Microsoft; Ed Lazowska, University of Washington
Pre-Announcement/Call for Papers available early December, 1996

Large Scale Systems Administration of NT
Pre-Announcement/Call for Papers available early December, 1996

Conference on Domain-Specific Languages
October 15–17, 1997, Red Lion Resort, Santa Barbara, CA
Program Chair: Chris Ramming, AT&T Research
Papers due: June 13, 1997; Notification to Authors: July 10, 1997, Camera-Ready Papers due: September 2, 1997

11th Systems Administration Conference (LISA '97)
October 26–31, 1997, San Diego, CA
Co-sponsored by USENIX and SAGE, the System Administrators Guild
Program Chairs: Hal Pomeranz, Netmarket/CUC International; Celeste Stokely, Stokely Consulting
Extended Abstracts due: June 3, 1997; Notification to Authors: June 30, 1997, Final Papers due: September 9, 1997

Workshop on Internet Technologies & Systems
December 8–11, 1997, Monterey, CA
Program Chair: Carl Staelin, Hewlett-Packard Laboratories
Extended Abstracts due: July 8, 1997; Notification to Authors: August 12, 1997; Final Papers due for Editorial Review: September 24, 1997; Camera-Ready Papers due: October 22, 1997

7th USENIX Security Symposium
Sponsored by the USENIX Association in cooperation with The CERT Coordination Center
Program chair: Avi Rubin, Bellcore; Invited Talks Co-ordinator: Greg Rose, Qualcomm International
Papers due: Sept 9, 1997; Notification to Authors: October 8, 1997; Camera-Ready Papers Due: December 9, 1997

For more information, access the USENIX Resource Center on the World Wide Web. The URL is http://www.usenix.org.
The User/Product Conundrum

Those in academia are too often enamored of the notion that "universities would be a great place if it weren't for all the students." Some in the technical world also too often think this is true in the case of "users."

We can classify customers of computer software into two broad groups: those who are experts on workstations and mainframes - people who can build tools to make their computer jump through hoops; and those who consume tools such as spreadsheets, word processors, web browsers, "groupware," and email, to name a representative set.

By and large, the now-huge group of computer consumers does not have a good understanding of what the technical group people do. Many think of themselves as having vast knowledge of computers, and see no need for "expensive" technical people. Worse, the smaller technical group often harbors ill-will toward the consumer group (based on any number of factors, not limited to technical skill). Yet it is clear that more (and ever more) money is flowing toward the consumers. This motivates software companies to address that group ever more explicitly.

These conditions combine to form an environment that is unfortunate, at best. Why? Here are some reasons.

First, there is a notion that the larger consumer group should take a large part in computer business decisions - and hence product decisions.

Second, purchasing decisions in companies thus become strongly influenced by a group who might not focus in on the scalable, big-picture concerns that the technical people think are most important. Inability to evaluate technology beyond the demo (or marketing hype) quickly leads to the situation best described in the Bill-Gates-visiting-hell joke. The subtle points of computer capabilities and performance become lost.

Third, the inability of both groups to understand each other’s concerns easily leads to a sort of inter-group contempt. Each group may choose to denigrate the other, leading to myths about what each group actually does and needs.

What it all boils down to is: technical products in the style of UNIX (many of which are UNIX people favorites) are not very popular in the consumer world. The consumers like the "other" style because it meets all of their perceived needs.

I like the UNIX-style tools and I am thinking of ways to help other people like them, too. If you know some way to illuminate the "dark side" of tool building and tool using, why not drop me a line? Maybe there’s even an article or a good conference talk somewhere in the discussion.

RK
LETTERS TO THE EDITOR

Correction

In ;login:, October 1996, Volume 21, Number 5, my letter to the Editor entitled “Choosing A Good Password” was published. The letter included a reprint of copyrighted material. Since the material was presented as a reprint of copyrighted material and not as new and/or edited material based on copyrighted material, please publish a statement that the material had been edited and, as such, was not a reprint.

Additionally, please include a statement that the change of “unixsuck” into “dossucks” constituted a content change: unixsuck is a common, bad UNIX password, dossucks is not a common, bad password, UNIX or otherwise.

Regards,

David G. Beausang
<dgb@csn.net> or <dgb@mines.edu>

Done.
The Editors

Notice of Annual Meeting

The USENIX Association’s Annual Meeting with the membership will be held sometime during January 8-10, 1997, at the Anaheim Marriott Hotel, site of the 1997 USENIX Technical Conference. The exact date, time, and room location will be published in the onsite conference directory, comp.org.usenix, and you also may contact the Association’s office in mid-December.
Board Meeting Summary

by Ellie Young
<ellie@usenix.org>

Below is a summary of the actions taken at the meeting of the USENIX Board of Directors held on September 28-29, 1996 in Chicago, IL.


Budget

The assumptions behind the first draft budget for 1997 were discussed. The Board also considered member fees, benefits, and several proposals for funding services and activities in the coming year. The budget was later approved as amended.

Member Dues. Individual membership dues beginning February, 1997, will be decreased by $10 (to $60).

Conference Fees. Conference registration fees for technical sessions at two-day events will not be raised, and the exhibitor booth fees will be increased as follows: LISA '97 to $1,950; USENIX '98 to $1,750; and Security '98 to $750.

Publications. It was decided that since the journal will no longer be published, the number of issues of ;login: will increase from 6 to 8 within the next 12 months. The ;login: editorial committee will meet in the coming months to continue its efforts to improve the quality of ;login:. It was also decided to hold a strategic meeting in early 1997 to discuss the Association’s future goals and member benefits.

Student Programs. The scholastic committee will evaluate proposals from students and faculty to fund student research projects. A discussion about the model for a scholarship program and for helping graduate/undergraduate students (especially focusing on under-represented groups) ensued. It was decided that the scholastic committee should come up with proposals for the Board’s consideration, and Seltzer agreed to chair this committee.

Proposals for Funding

Standards. Stephe Walli was thanked for his services as our Standards Institutional Representative to the IEEE PASC/POSIX meetings these past two years. (Walli is stepping down due to other commitments.) The proposal from Nick Stoughton to serve in this capacity and for our continuing to fund standards activities for 1997 (which includes sending a paid Institutional Representative to the IEEE PASC/POSIX meetings, reporting back to the membership about this activity, and the preparation of IETF area director’s reports from those meetings) was approved. It was also agreed to allocate funds for future standards activities, such as sending a representative to attend an X/Open working group for one year. A proposal from Stoughton will be forthcoming.

Good Works

It was agreed to once again sponsor the US International Computing Olympaid on Informatics ($35,000).
A proposal to provide bridge funding in 1997 for the Internet Software Consortium ($50,000) was approved.

A proposal from the Polytechnic University to support a collaborative program with the United Neighborhood Houses of New York for training youth, encouraging college study in technology, and providing equipment and stipends, was tabled, pending a site visit by two board members.

A proposal from Peter Salus to fund a book on the history of programming languages was declined.

In response to a request from EurOpen to assist them in organizing a conference next Spring, it was agreed that we will sponsor a tutorial (to include speaker honorarium, travel and expenses).

**Conference Proposals**

**Conference on Domain-Specific Languages.** Hume and Johnson will serve as co-liasons to Chris Ramming, who will serve as program chair for this conference, to be held in mid-October, 1997, in Santa Barbara, CA.

**Workshop on Internet Technologies & Systems.** A proposal from Carl Staelin for USENIX to hold an event on this topic in late-1997 was accepted. Seltzer will serve as liaison.

**Windows/NT Workshop.** Honeyman obtained a proposal from Mike Jones and Ed Lazwoska to co-chair a workshop on this topic. It was accepted and Seltzer will serve as liaison.

**Authentication & Public Key Infrastructure.** Rose and Geer will look into the possibility of holding a small workshop on this topic.

**CARDIS '98.** A proposal from Honeyman, Doug Tygar, and Bruce Schneier for USENIX to organize a workshop on smart card technology in the Spring of '98 was accepted, and a formal agreement for presentation to the CARDIS steering committee will be forthcoming.

**Anonymous remailer service.** It was agreed to send a letter to Johan Helsinguis expressing our thanks for his providing this service to the community in the past, and encouraging him to undertake further study on the implications of anonymity and privacy on the Internet.

---

**Webstar Update**

The Webstar contest is now officially over, and by the time you read this the winners will have been announced!

To learn about the winners and other talented young websters, please check out the Webstar URL: [http://www.intuitive.com/webstar](http://www.intuitive.com/webstar).
Call for Student Research Proposals

USENIX has established a new $50,000 fund for student research projects. The first award, totalling $14,000, was given to Professor Victor Yodaiken of the New Mexico Institute of Technology for three students to work on porting Linux to the PowerPC architecture.

"USENIX made applying for this grant extremely easy," said Professor Yodaiken. "It's encouraging to be selected, and an honor to be the first recipient of a USENIX award for student research." The award will fund one graduate student for a year and two undergraduate students for one semester each.

USENIX is seeking others to apply for grants. Please include the following information in your proposal and submit them by December 1, 1996 to Ellie Young via email at: <ellie@usenix.org> or post c/o:

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710

Proposals will be evaluated by the USENIX Scholastic Committee (some of whose members also serve on the USENIX Board of Directors). Applicants will be notified by January 15, 1997.

Proposals should contain the following items:

• Your full name
• Postal address /Email/Phone/Fax/URL (if you have one)
• Program of study
• Year in program
• Department/University
• Name of Faculty Advisor and contact information
• A statement of what the project is about, e.g., its goals and what it hopes to achieve, how it fits within the programs/goals of USENIX, and any other background information (e.g., if this is already in progress, how much has been accomplished, how much time will it take).
• A detailed list of specific resources needed (if any), with projected costs. Resources may be anything you will need to do the project – staff, computers, etc.
• Expected completion date
• Any other resources you have or are applying for, including matching grants

Statement of Ownership
Management and Circulation, 10/1/96


<table>
<thead>
<tr>
<th>Extent and nature of circulation</th>
<th>Average no. of copies each issue in preceding 12 months</th>
<th>Actual no. of copies of single issue published nearest to filing</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Total no. of copies</td>
<td>7872</td>
<td>8700</td>
</tr>
<tr>
<td>B. Paid Circulation, Mail Subs.</td>
<td>7120</td>
<td>8166</td>
</tr>
<tr>
<td>C. Total paid circulation</td>
<td>7120</td>
<td>8166</td>
</tr>
<tr>
<td>D. Free distribution</td>
<td>569</td>
<td>330</td>
</tr>
<tr>
<td>E. Total distribution</td>
<td>7689</td>
<td>8496</td>
</tr>
<tr>
<td>F. Copies not distributed</td>
<td>183</td>
<td>204</td>
</tr>
<tr>
<td>G. Total</td>
<td>7872</td>
<td>8700</td>
</tr>
<tr>
<td>H. Percent Paid Circulation</td>
<td>90%</td>
<td>94%</td>
</tr>
</tbody>
</table>

I certify that the statements made by me above are correct and complete.

Ellie Young, Executive Director.
SAGE NEWS

Back to the Future
by Tina M. Darmohray
<tm@iwi.iwi.com>

It's interesting. I just got back from the USENIX LISA conference where a lot of the hallway, dinner, and less formal meeting discussions centered around another operating system, NT. Here are some pros and cons I heard about NT:

• USENIX will be sponsoring a workshop on NT (It's noted that it will probably be harder to get experienced presenters than desperate attendees!)

• Organizations buy NT because it has the popular desktop “Word” and “Excel” applications.

• Organizations like the commodity hardware on which NT runs.

• High-end networking and applications still end up on RISC machines running UNIX.

• Microsoft is educating future engineers by giving products to universities (much like what happened with UNIX.)

• It’s hard to scale support for NT machines because their administrative paradigm is a single person sitting at the console.

• There was a paper about administering PCs using LINUX. (I understand it was not the only one submitted on this topic. Someone commented “it is telling that LINUX is used to administer the PCs”.)

By mid-conference I found myself trying to decide what conclusions can be drawn from all the PC/NT information. The things that I thought I heard sounded so “backwards” from a historical perspective that it’s hard to believe, let alone use it to prepare oneself for the future. Could the following really be true?

• If UNIX had Word/Excel equivalents, it would “own the desktop” instead (because you can get UNIX on commodity hardware now, too).

• UNIX, which has always been criticized for being “hard to use,” is easier to manage in a networked environment than PCs? Is it possible that UNIX is “friendly” now?

• If UNIX is the OS of choice on large machines running large databases, it’s now the “old,” “established” OS?

• Marketing makes all the difference in the success of an OS?

One of my colleagues commented on the results of the “handouts” at the Synopsys hospitality suite. It appears that, given helium-filled balloons, legos, a little free time, and a hotel with an atrium, UNIX administrators will build floating toys. He felt that spoke to the problem-solving nature of the attendees. I thought about that, and the paper on using LINUX to administer PCs, and USENIX embracing the NT administration problems with a workshop, and I thought that those also speak to the problem-solving nature of system administrators, and maybe UNIX administrators in particular. If that’s the case, then I suppose it doesn’t matter which platform creates the challenge because we’ll find a solution.
Letter from the President

by Paul Evans
<ple@usenix.org>

I have just returned from the LISA 10 Conference in Chicago and am pleased to report that the conference was very successful. We had an attendance of over 1,660, which is just a little short of last year’s attendance of 1,724. This was the first year that the SAGE board’s policy of moving the conference around the country to give more people an opportunity to attend was implemented. There was uncertainty about just how many people would register because it was the first time the conference had been held outside California since 1990, but the attendance surpassed even optimistic projections.

Of course, content is a more important ingredient of a successful conference than location, and on behalf of the SAGE board of directors, I’d like to thank the Program Committee co-chairs, Helen Harrison and Amy Kreiling, the Program Committee, the Invited Talk coordinators, Rik Farrow and Kim Trudel, and the Tutorial coordinator, Dan Klein, for arranging a truly outstanding program. One of the most important measures of the health of a conference is the number of paper submissions the Program Committee receives. This year the Program Committee reviewed 78 submissions, up from 55 last year, which in turn was up from 42 the year before. This trend of increasing submissions bodes well for the future of the conference and also means that next year’s program co-chairs, Hal Pomeranz and Celeste Stokely, and their Program Committee have their work cut out for them!

The most enjoyable duty the president of the SAGE board has is to present the annual SAGE Outstanding Achievement Award at the keynote session on Wednesday morning at the LISA conference. This year’s award went to Elizabeth Zwicky, and it’s hard to imagine a more deserving recipient. Among her many services to SAGE, USENIX, and the UNIX system administration community, she was one of the “original seven” founders of BayLISA, the first of a now large and growing number of local groups serving UNIX system administrators, and served on its board of directors for several years. In 1991, she served as Program Committee chair for LISA 5, a conference that still sets the standard of quality for the refereed paper track, and has frequently contributed to the LISA conferences as an author, a member of the Program Committee, or an invited talk speaker (sometimes all three!). In 1992, she was a member of the working group that got SAGE off the ground and was designated president of the interim SAGE board. She was a member of the SAGE board from 1992 through 1995 and was president of SAGE in 1994 and 1995. In that role, she had more to do than any other single individual with the phenomenal growth and success of SAGE as an organization in its first few years. Most recently, she was elected to a seat on the USENIX board, so our community can continue to look forward to her future contributions in that role. Congratulations, Elizabeth, on this well-deserved honor.

Congratulations are also in order to Barb Dijker, the SAGE policies working group, and the USENIX staff for seeing the Policies pamphlet through to successful completion. If you are a SAGE member, by the time you read this, you will have received a copy of A Guide to Developing Computing Policy Documents, the second issue of the Short Topics in System Administration series and our first publication since 1993. Another publication, entitled Systems Security: A Management Perspective, has been received from the authors, David Oppenheimer, David Wagner, and Michele Crabb, and is currently under review by its editor, Dan Geer. Look for the Security pamphlet in your mailbox sometime after the first of the year.

One of the major themes of hallway conversation at the conference was the challenge posed by increasing numbers of NT systems the attendees are being confronted with in their computing environments. In response to the overwhelming interest in this issue, SAGE and USENIX are working to put together a small workshop in 1997. As I write this, an organizing committee led by Xev Gittler and Phil Scarr are putting together a workshop proposal for review by the SAGE and USENIX boards and a Call for Participation. This is another SAGE- and USENIX-sponsored event we can look forward to.

I’ll close with a reminder about the SAGE board election. Ballots are due to the USENIX office in Berkeley no later than December 10. Many of you will be getting this copy of :login: just a few days before the deadline. If you haven’t yet done so, please take a few minutes now to read the election materials and vote.
Where’s the Ark? Dealing with SYN Flooding

by Shawn Instenes
<shawni@siforest.com>

If you’ve been watching security mailing lists as I have, you’ve heard about the rash of TCP SYN flooding attacks that have been going on a lot lately. As if the Internet didn’t have enough growing pains, now any malcontent with access to a Linux box and a network connection can spam any server they want to. Fortunately, there are things you can do about it, but first I’ll summarize the problem.

The nature of the attack is simple and deeply rooted in the nature of TCP. TCP relies on a three-way handshake to start a connection:

1. The client sends a packet with the SYN (synchronize) bit set. This informs the server what ISN (initial sequence number) is going to be used by the client. The sequence numbers enable TCP to detect dropped and duplicated packets. (Netstat reports SYN_RECEIVED.)
2. The server sends back a packet with the SYN and ACK (acknowledge) bits set. With this packet, the server confirms the use of the client’s ISN, and informs the client what ISN the server will use. (Netstat reports SYNSENT.)
3. The client sends a packet with the ACK bit set, confirming the server’s ISN. (Netstat reports ESTABLISHED.)

In most TCP implementations, the server will keep around data that pertain to the connection (ISN, port numbers, IP addresses) between the SYN_RECEIVED and ESTABLISHED states. These data will be flushed if the connection times out (no ACK from the client within 75 seconds, usually). The server has limited storage to handle these “half-open” connections, with 30 or so being a common maximum number of half-open connections per port number.

If a server receives a new connection packet when it has already reached its limit of half-open connections, the new connection is ignored. This is the goal a SYN flooder is trying to reach: making your server unresponsive to requests from legitimate users.

SYN flooders use a program to generate TCP SYN packets that are addressed to their target that have one special property: the source address listed in the packet is falsified. It is some address to which the target can route a packet, but it will never respond to that packet (e.g., it is behind a firewall). The false source address also makes it hard to track precisely the origin of the bogus SYN packet.

These SYN packets are received by the target, which allocates space for the new half-open connection and responds by sending a SYN-ACK response, as usual. The SYN-ACK packet never triggers a reply, so the server keeps trying until it times out the connection.

Multiply these forged packets by dozens or hundreds per second, and you can see that it is very easy to cause the server to run out of resources to handle new connections.

There are some tricky bits here. If the source address actually refers to a machine that is reachable, that machine normally will respond to the server with a packet that has the RST (reset) bit set, which basically means “I don’t want a connection with you.” The server will then free the resources that the phony packet tied up. This is why it is important for the SYN flooder to pick a source IP address that belongs to a host that isn’t responding.

Why wasn’t this sort of thing used earlier? Well, it didn’t used to be as easy to do as it currently is; modern kernels support a more fully functional sock_raw interface that allows a fully built IP packet to be constructed outside of the kernel space, which is useful for all sorts of things but makes the programming for this kind of attack trivial.

What you can do to harden your computer against this sort of malicious behavior is to modify the way your kernel handles half-open connections. There are patches from some vendors for this; ask for them. The most common solution I’ve seen so far involves expanding the kernel’s ability to handle half-open connections to a much larger value and also reducing the timeout interval to something much less, on the theory that 75 seconds is much too generous. I hope you won’t have to deal with this kind of maliciousness at your site.

On a completely unrelated note, I want to put in my two bits about one of the more useful things I learned about at the LISA conference: the creation of the Perl Institute, a non-profit organization dedicated to making Perl more useful for everyone. Give it a look; having used Perl for years in order to do my job, I signed up right away.

URLs to visit:
The Perl Institute: <http://www.perl.org/>
BayLISA 1996
by Bryan McDonald
<bigmac@baylisa.org>

BayLISA has been around for something like six years now. It was created from an idea at LISA 4, probably over some drinks in a bar. We had Larry Wall come talk about Perl4 when it was the new thing and again when Perl5 was the new thing. We have held our meetings at DEC, SGI, Synopsys, and are now moving on to Cisco ... who is next?

Some random thoughts at a BayLISA board meeting four years ago turned into week after week of meetings between a whole lot of people, and SAGE was born. Last week I went to LISA 10, sponsored by that same organization, and worked with other SAGE people to help bring together other local groups and create new conferences.

The BayLISA community has seen at least one, and probably two, complete personnel turnovers and a few returning friends as careers move people out of the area, then back into it again. We have cycled through enough ideas for our meetings that we worry not only about who to get next month, but if they were here last year as well. Fortunately for us (both the organization and the individuals inside it), the world we work in keeps changing at a tremendous rate, keeping us all quite busy. Some things are always the same, however; we are all still looking for a really fully functional account management tool that does just the thing that we need for that corner case.

About the time that this article gets to you, BayLISA will be holding its board elections. We are fortunate this year to have a crew of new and old people running, which will breathe some new energy into the board. In October, we have Vinay Kumar talking about MBONE. In November, we have Randal Schwartz talking about the stuff of life, Perl and police dramas. In December, we are working on our usual holiday session, which revolves around horror stories and stupid user/admin/vendor tricks, because we run too close to Christmas for most of our regulars to attend. In 1997, we are already looking at talks on new security concepts, new network concepts, the worldwide standards bodies, and NT admin for UNIX system administrators.

We are always looking for backup speakers as well as new topics, and that last one is the perfect example. We lost our NT admin speaker for October during the week that half of the BayLISA board was in Chicago at LISA. Fortunately, we had two (possibly three) people all lined up in a a day or so, and we had the luxury of choosing, even with only two weeks to spare.

Anyways, we have meetings. We also have a relatively low membership count, with a few generous corporate members that enable us to continue to fly in speakers multiple times a year. You don’t have to be a member of the organization to attend the meetings, but we do like to encourage people who are long-term members to join for the sake of a better pool of speakers to choose from.

One of the board’s new actions for 1996 was to create a student membership with a very reduced membership fee. Our next step is to advertise our meetings in the local schools and draw in these people who are learning the ropes. This is probably as self-serving as we can get ... get those people training to do this into those meetings, where they can hear our job announcements, and we get first crack at them. But this also gives them a chance to see both that there is a real profession out there waiting for them and how they can apply what they are learning today directly in the community.

For the future, well, we have our meetings, we have our new membership drive, and we have a new initiative, if I can get it off the ground, to try to get some of the BayLISA people more interested in the standards efforts. There was an interesting panel at LISA 10 about the standards process, and one key point was that there is little input from the sysadmin community about those standards that will affect us most. The USENIX/SAGE groups are thinking about forming some sort of group of “technical experts” (i.e., us) that could assist the USENIX standards representative, Nick Stoughton, in developing input for the committees and give him more influence over their work. I like the idea, so I will try to start a similar or related effort inside BayLISA – my personal goal for the next year.

For more information about BayLISA, you can check out our web pages at <http://www.baylisa.org>, or you can drop me a line.
Requests for Proposals

SAGE is seeking proposals from authors for four separate booklets as part of its “Short Topics in System Administration” series. The four booklets, described in more detail below, encompass legal issues, education, hiring, and site audits. These new booklets will complement the following existing booklets in the SAGE series:

1. *Job Descriptions for System Administrators*
2. *A Guide for Developing Computing Policy Documents*

Legal Traps and Pitfalls for System Administrators

This document will provide an overview of some known (as of the date of publication) legal problem areas that directly affect system administrators. It should serve both to raise the level of caution on the part of system administrators and managers and to spur further discussion between the legal community and the system administration community. This document will not be a comprehensive treatment of the law in any particular jurisdiction. Some topics to be addressed include:

- Censorship
- Fighting computer crime without being caught by “rights” laws in reverse
- Privacy issues
- Ownership of data files, email, etc., on the machine and network
- Copyright issues
- Hiring and firing
- Responsibility to adhere to the code of ethics
- Relative duty owed to users and employers

Educating and Training System Administrators

This document will be a guide for educators developing curricula, managers developing staff training, and system administrators interested in furthering their own education. The booklet should cover both technical and nontechnical areas of education.

This document is not intended to be a step-by-step manual for implementation of educational programs in general, nor is it expected to document an educational certification process. It should be a guide regarding skill requirements specific to system administration. Emphasis should be on imparting actual knowledge and problem-solving skills in practical environments. Some topics that should be addressed are:

- Fundamental prerequisites
- Mandatory and elective subject lists to satisfy the equivalent of a bachelors degree in system administration
- Paths and flow of basic education and specializations
- Course suggestions, including objectives, descriptions, and topic lists
- Methodologies for building problem-solving skills
- Hands-on approaches
- Considerations in building a training lab
- Accompanying textbooks
- How to teach or foster the requisite nontechnical skills
- Case studies of currently used models and their successes and failures: both commercial and education industries
- Continuing education

Hiring Policies and Practices and Interview Strategies

This document will serve as a guide for system administrators, managers, and human resources personnel regarding the need for developing good interviewing practices to prevent poor hiring decisions and the trade-offs between seeking inexperienced and experienced administrators. It is meant to complement the existing SAGE booklet, *Job Descriptions for System Administrators*.

This document is not intended to be a manual for conducting interviews, but rather an issues-oriented guide for hiring decisions and the interview process. Some topics that should be addressed are:

- Trade-offs between low-cost, inexperienced people and high-cost experience
- Training model trade-offs: “growing your own” vs. hiring fully trained
- Internal vs. external hiring
- Effective job advertising
- Selecting and preparing an interviewing team
- Interviewing strategies
- Assessing actual knowledge and competence
- Assessing ethical and interpersonal compatibility
- Expected and typical benefits to candidates during the hiring process
- What is expected of a candidate
- What is expected of a new employer
- What is expected of a new hire
- Common pitfalls and their risks
- Fair and ethical practices

Site Audits

This document will serve as a guide for system administrators regarding the need for an initial and a periodic audit of system and network status. This document is not intended to be a checklist, but should cover reasons and areas to be investigated. Some topics that should be addressed include:
• Existence of a complete and viable set of policy documents
• Security model in place and effectiveness of implementing policy
• Hardware inventory and configuration documentation
• Licensed software control procedures
• Documented procedures for policy-based tasks such as backups and account management
• Accurate network diagram
• Effective problem/job/request tracking system
• Maintenance agreements

Case studies or illustrative anecdotes are encouraged. Each booklet should be 20-50 pages in length. The author(s) will be compensated. Proposals should contain names, curriculum vitaeas, and contact information for all authors, a representative writing sample not to exceed 500 words, a draft outline for the booklet, and an estimate of how long it will take to deliver the manuscript. Proposals should be in either text, html, or a URL reference. The author will not be expected to copyedit, design, typeset, or print the booklet. An agreement regarding copyright and compensation will be executed with USENIX upon approval and acceptance of a proposal by the SAGE Board of Directors.

Please address all inquiries and proposals on or before February 1, 1997 to the USENIX Executive Director, Ellie Young <ellie@usenix.org>.

The SAGE Calendar for 1997 Is Here!

Thanks to ContainCo, the highly fictitious creation of Nick Cuccia, the SAGE Calendar for 1997 will soon be on its way to all SAGE members.

The theme of the 1997 calendar is firewalls, and ContainCo represents “Superior Enhanced Firewall Technology and On-site Integration Services for Almost a 50th of a Century.” As always, the SAGE Calendar does its best to note the most critical dates in the upcoming year such as the date ARPANET was born and when TCP/IP became official ARPANET protocol, as well as important birthdays, many of which will no doubt become national holidays, such as Dennis Ritchie’s, Biff’s, and many others of equal renown.

And this year, as a special plus, the Calendar features the photography of Bill Owens, author of Suburbia, a classic of American photography. If you’re not familiar with his work, we’re sure that – through the filter of the ContainCo public relations staff – you’ll come to enjoy his rich mixture of the absurd and the everyday that his photographs convey.

Premier firewall services since the first one was invented...not too long ago

ContainCo’s firewall solutions come in configurations suitable for every networked environment.
Interview with Mike O’Dell

by Rob Kolstad
<kolstad@BSDI.COM>

Rob: Seems like you’ve been at UUNET for quite a while. Please tell us a bit about your position and your tenure.

Mike: I came to UUNET from three years at Bellcore, where I learned quite a bit about scaling problems and how you want service provisioning systems to work. (Assuming you get a choice!)

I joined UUNET full-time as employee #31 in March 1993. Now there are about 500 people in Fairfax, about 300 in various non-US locations. These are just UUNET numbers!

Rick Adams asked me to join and help mind the store while he worked on putting together the venture funding. As we brought on an experienced management team, I ceded pieces until now I have no direct reports! My primary responsibilities focus on the architecture of our network in the one to five year time frame. This is pretty scary when the load is doubling every four to six months.

Rob: Do you end up, then, spending most of your time thinking about hardware systems or software systems?

Mike: It depends on the day of the week (or maybe the hour). I spend a lot of time working with our current and potential technology suppliers, explaining how our network really works and what we need from them to meet the growth challenges. Some of these discussions revolve around the internal architecture of various network elements and some of them around network element software.

Another large piece, though, is the “information infrastructure” of the company and how we leverage it to improve productivity and responsiveness to customer needs. Dramatically reducing the level of human fiddling currently required to install a new customer is a big requirement for future scaling.

In fact, the entire spectrum of problems is summed up as follows:

The only real problem is Scaling;
Everything else is a subclass.

Rob: Bob Metcalf is widely quoted as predicting the “death of the Internet” within a few months. Do you see that happening?

Mike: As a person living in Colorado Springs, do you plan your shopping trips to the grocery store based on the knowledge that sometimes the traffic on the Washington, DC, Beltway can be pretty grim when it snows?

The Internet is pretty large, and while not impossible, the likelihood of a global meltdown is very, very, very small. There are areas which experience traffic problems sometimes for various reasons, one of the most important is that the “public exchange points” are now anachronisms (MAE-EAST, MAE-WEST, etc.).

Like every technology, the exchange points enjoyed a window of viability which opened and then shut. Today, a diskless SUN seems pretty silly when a 2 gig disk is $200. But there was a time when they made sense. So, too, with the exchange points. Now they are dying a horrible death, and the large providers are installing...
many high-performance bilateral interconnects just as fast as the circuits can be delivered.

Rob: What are the particular design challenges that you see on the horizon for keeping up with the growth of the Internet?

Mike: It doubles every 4-6 months, now. 18 months ago it was doubling every 12 months. Care to speculate about the growth another 18 months from now?

One of the biggest architectural challenges is growing capacity without increasing link bandwidth. In traditional IP-only networks, the only really effective way to increase the capacity of the network was to increase the bandwidth of the links. The problem is that even at DS3 this starts getting harder to do. It is pretty near certain that single-strand link speeds will be stuck at OC12 (622 megabits/sec) and OC48 (2.5 gigabits/sec) for quite a while even as the individual fibers themselves move to carrying multiple OC48 streams using dense wavelength-division multiplexing.

I can see from here when the ability to increase link speed at any price hits the wall, so one had best have a good handle on adding capacity without increasing the link speeds. Luckily, we attacked this problem with the current DS3 network. We believe we have a pretty good handle on an architecture which can manage adding arbitrarily connected links to increase transmission capacity without going crazy messing with IP routing metrics and wasting gobs of bandwidth. Whether the architecture scales to a network where each hop in the fabric is 20-40 OC48 links in parallel will be interesting to discover.

The other big problem is that the bandwidth crunch is starting. The folks writing about how much unused bandwidth is in the ground are simply smoking dope — there are routes right now where you cannot get more capacity, and the list is growing. The telecom’s infrastructure business is undergoing a very fundamental shift in the scaling driver, and it is going to get distinctly unpleasant for a while.

Rob: Do those changes apply to the international scene as well?

Mike: Yes, this shift is fundamental, and it is worse on the international scene because wet fiber is much more rare than fiber covered with dirt.

Growth in the telecoms infrastructure is traditionally driven by how many people call their mother on Mother’s Day. No longer. The “Silicon Cockroaches” have arrived, and they are very hungry.

The first of the Silicon Cockroaches was the fax machine — a computer that got itself a phone line and wouldn’t let go. And it almost single-handedly destroyed the North American Numbering Plan (although its buddy, the cellphone, certainly pitched in to help). Now the PC with a modem is chewing on the switched network bandwidth capacity, and the long-haul Internet links are gobbling up transmission capacity like it’s free lunch.

The Cockroaches have increased their bandwidth appetite 1,000,000-fold in the last 20 years — from a 150 baud acoustic coupler on your terminal to a 155 megabit ATM drop to your SUN. And we still talk at 3KHz audio. Equally important, they breed faster than we do.

As far as communication via human sensory signals, we are an endangered species in the competition for bandwidth with the prospect of the worldwide Voice Minutes business becoming the first $250 billion niche market.

So the demand on the infrastructure is shifting from the Mother’s Day Law to something that grows even faster than Moore’s Law. In fact, the only things that grow as fast as the Internet are kudzu and plagues.

Rob: Would you care to make any predictions about the deployment of technologies that would implement some of the hype we hear so much about? I’m thinking here of video servers to the home, Internet telephone, bandwidth that’s “too cheap to bother to measure.”

Mike: “Bandwidth too cheap to meter!” I guess that’ll happen when we get nuclear-powered routers. High-performance access to the Global Internet is not going to be a lot cheaper any time in the near future. The long-haul capacity is too expensive and will stay that way, sadly.

But high-performance access to your metropolitan Giant Web Cache may well get pretty cheap. But these are very, very different businesses (e.g., compare UUNET with @Home). However, Web designers should think very hard about the implications of this — those who adapt will do well; those who don’t adapt will leave the gene pool.

The Internet is like any other artistic medium — there are works which can be rendered well using it and those that don’t work so well using it. The real artists discover the strengths and limitations and still manage to do compelling, expressive work within the constraints of the medium.

Rob: Any startling predictions about the future?

Mike: I think the current reality of the Internet’s success is so amazing that speculation pales by comparison.

However, I am looking forward to the day that we can install fiber systems based on Optical Solutions, so a 200 fiber cable can run each strand at 20 gigabits across an ocean or continent without repeaters, giving a capacity of 4 Terabits/sec on the pipe.

Now THAT will start to get interesting!
Beating Bottlenecks In the Design of Distributed File Systems
Ahmed Amer and Amr El-Kadi
<amer4, elkadi}@auc.cs12.eun.eg>

Abstract
Many different issues are involved in the design of a distributed filesystem. Requirements vary, but the main issue discussed in this article is how the introduction of a network and multiple computers as an element of the system upon which a distributed filesystem (DFS) is built introduces new bottlenecks to the performance of filesystems and how these issues have been, and are being, handled. This article focuses on techniques to deal with problems of scalability, availability, and performance in the context of distributed filesystems — highlighting key systems and concepts that have tackled these problems.

1. Introduction
Distributed filesystems are simply an evolution of the concept of a network filesystem, or filesystems that enable files to be shared across a network with a large degree of transparency. Functionality varies from mounting remote filesystems locally to providing what can be called implementations of distributed shared memory. These systems satisfy the all-important requirement of providing access to files with no special consideration on the part of the “user” for the intervening network subsystem and thus provide the possibility of location-transparent access to files.

This article does not so much deal with the characteristics of distributed and network filesystems, as concentrate on the concepts applied to beating the various bottlenecks introduced when we start considering the larger picture of network filesystems. With the introduction of Sun’s RPC mechanism, it was not too surprising to see network filesystems appear that provide transparent access to remote files. Contrary to popular belief, Sun NFS was not the first such system. In fact one of the earliest systems was Newcastle, which provided a simple mechanism whereby calls involving files were intercepted (by a software layer between the application and the kernel) and, if they referred to a remote file, were passed to the relevant server through RPC calls.

Naturally, this system had poor performance because there was absolutely no caching of data at the client side. This meant that every call accessing a file would suffer the overhead of network communication. The first truly successful network filesystem to be implemented was probably Sun’s NFS. It has been so successful that there exist implementations of NFS services for systems ranging from workstations running non-Sun versions of UNIX, PCs running UNIX, or even DOS/Windows, to larger systems running such completely different operating systems as VMS.

NFS achieves a degree of location transparency. It does not address migration transparency (i.e., should a remote directory be moved to a new server, the administrator has to update every client with the new server id). As with Newcastle, NFS was based on Sun’s RPC mechanism, but its performance was dramatically improved. In fact, certain mixed read/write benchmarks show NFS is really only 20% worse in performance than a local filesystem [Coulouris et al., 1994]. This is due mainly to the use of caching at both the server and client sides.

In the second part of this article we discuss the problems of massive scaling across WANs. We overview the serverless filesystem, xFS, and summarize how it handles this issue. We will then discuss how the technique of caching can be used to increase performance while maintaining a consistent filesystem. Finally we discuss the concept of data striping and how it applies to filesystems (focusing on distributed filesystems) to achieve both goals of high performance and increased availability.

We conclude with a discussion of how the concepts of striping, caching, and log-structured filesystems can be combined to “beat the disk” and provide both performance and reliability benefits for filesystem performance. These concepts can even be shown to provide levels of performance that not only match but exceed those of local filesystems. In this context, we are referring to performance in the sense of reliability as well as simple “speed.”

2. The Classics
2.1 NFS

With the introduction of Sun’s RPC mechanism, it was not too surprising to see network filesystems appear that provide transparent access to remote files. Contrary to popular belief, Sun NFS was not the first such system. In fact one of the earliest systems was Newcastle, which provided a simple mechanism whereby calls involving files were intercepted (by a software layer between the application and the kernel) and, if they referred to a remote file, were passed to the relevant server through RPC calls [Keeton et al., 1995].

2.2 Coda

NFS, while providing a degree of location transparency, did not address migration transparency (i.e., should a remote directory be moved to a new server, the administrator has to update every client with the new server id). As with Newcastle, NFS was based on Sun’s RPC mechanism, but its performance was dramatically improved. In fact, certain mixed read/write benchmarks show NFS is really only 20% worse in performance than a local filesystem [Coulouris et al., 1994]. This is due mainly to the use of caching at both the server and client sides.

In the second part of this article we discuss the problems of massive scaling across WANs. We overview the serverless filesystem, xFS, and summarize how it handles this issue. We will then discuss how the technique of caching can be used to increase performance while maintaining a consistent filesystem. Finally we discuss the concept of data striping and how it applies to filesystems (focusing on distributed filesystems) to achieve both goals of high performance and increased availability.

We conclude with a discussion of how the concepts of striping, caching, and log-structured filesystems can be combined to “beat the disk” and provide both performance and reliability benefits for filesystem performance. These concepts can even be shown to provide levels of performance that not only match but exceed those of local filesystems. In this context, we are referring to performance in the sense of reliability as well as simple “speed.”

2. The Classics

2.1 NFS

With the introduction of Sun’s RPC mechanism, it was not too surprising to see network filesystems appear that provide transparent access to remote files. Contrary to popular belief, Sun NFS was not the first such system. In fact one of the earliest systems was Newcastle, which provided a simple mechanism whereby calls involving files were intercepted (by a software layer between the application and the kernel) and, if they referred to a remote file, were passed to the relevant server through RPC calls [Keeton et al., 1995].

Naturally, this system had poor performance because there was absolutely no caching of data at the client side. This meant that every call accessing a file would suffer the overhead of network communication. The first truly successful network filesystem to be implemented was probably Sun’s NFS. It has been so successful that there exist implementations of NFS services for systems ranging from workstations running non-Sun versions of UNIX, PCs running UNIX, or even DOS/Windows, to larger systems running such completely different operating systems as VMS. Basically, NFS allows you to mount filesystems from a remote server onto a mount point in the filesystem of your local machine.

NFS achieves a degree of location transparency. It does not address migration transparency (i.e., should a remote directory be moved to a new server, the administrator has to update every client with the new server id). As with Newcastle, NFS was based on Sun’s RPC mechanism, but its performance was dramatically improved. In fact, certain mixed read/write benchmarks show NFS is really only 20% worse in performance than a local filesystem [Coulouris et al., 1994]. This is due mainly to the use of caching at both the server and client sides.
2.1.1 Caching. NFS improved its performance dramatically over Newcastle. It incorporated many performance-enhancing features, including the implementation of the client and server modules as parts of the operating system's kernel and even a modification of the UDP packet size to 9K to allow the inclusion of a full 8K Berkeley "fast filesystem" disk block with an RPC call in one packet. But the single most important enhancement is probably client caching, wherein data is cached at the client.

NFS includes both server-side and client-side caching. The server-side caching is allowed only for read operations. This measure was mandatory to make the NFS server stateless—an important requirement to allow NFS to appear identical to a local filesystem. Because the server can fail independently, any write caching (like the 30 second timer used in many UNIX implementations) could result in serious inconsistencies. The application would be unaware of any crash and would effectively continue running, assuming that its write operation was successful. Admittedly, NFS allows two kinds of remote mounting: hard and soft. In the former, no call will return until it receives a response from the server; there is no possibility of a call completing with a communication failure. Soft mounting allows such a result, implying the possibility of inconsistent behavior from applications that have not been "prepared" to deal with such an exception.

In client-side caching, the client will cache both data and attributes of files. We believe that the design of the client-side caching mechanism in NFS is a small nightmare. The client will have no idea that a copy of a file, from which it is caching data locally, has been modified (the server is stateless and so will not hold any information about the clients), and so it will validate any locally cached data by comparing to the time-stamp of the file at the remote server. This time-stamp checking is performed whenever the client has cause to communicate with the server (i.e., when it is opening a file or fetching disk blocks) and at regular intervals (every 3 seconds for files and 30 seconds for directories). Anywhere between these regular intervals, the locally cached data are assumed to be valid! Caching is also performed for writes, but the data are then written back asynchronously on the close of a file or at a sync operation (possibly sooner through the use of biodaemons) [Coulouris et al., 1994].

2.1.2 Performance, Scale, and Availability. The performance of NFS has been shown to be comparable to that of a local filesystem, but these tests were using light loads (or even single client – single server setups). In practice, NFS is really practical only for five heavy clients or up to 50 light clients. (Heavy vs. light clients are measured in terms of amount of data they pass to/from the server.) These same tests also show that the client’s calls for getting files’ attributes account for 50% of all calls to the server. Performance is further impaired by the write-through behavior of the server. This is tolerable with light loads because writes have been seen not to exceed 5% of the file references [Coulouris et al., 1994], but with heavier loads, this becomes a serious performance problem. In terms of availability, should a server fail, in the case of hard mounting the application on the client system would simply "hang," because the call referencing the remote file will not return until the server comes back online. There is no inherent support for replication in NFS; the failure of a server is serious with respect to the files thereon.

2.1.3 The Pros and Cons. In terms of performance, scalability, and availability, there is room for improvement in NFS. There is also a possibility of cache consistency problems (during a three second window). In a paper describing the major issues in operating systems performance, the author went so far as to state that "the assumptions inherent in NFS (statelessness and write-through-on-close, in particular) represent a fundamental performance limitation. If users are to benefit from faster machines, either NFS must be scrapped (my first choice), or NFS must be changed to be less disk-intensive." [Outerhout, 1990]. And yet, on the positive side, NFS is very practical; it took compatibility with the local filesystem as a primary requirement and thereby achieved rapid acceptance and widespread use. Of all the systems reviewed in this article, it is by far the most widely used.

But when we have different requirements, and higher scalability, reliability, and/or performance become greater concerns, then different designs are required. The development of Andrew and Coda are good examples of this fact.

2.2 Andrew

The Andrew filesystem (AFS) was designed with good scalability as an important requirement; it was designed to be usable for 5 to 10,000 clients. At CMU, it was happily supporting 800 clients accessing shared files supplied by 40 servers. Like NFS, it supplies directories imported from a remote host at a particular local directory. Unlike NFS, this remotely mounted directory represents the entire directory structure of the shared filesystem tree and gives a globally consistent view of this tree. The system therefore has a small degree of location nontransparency, but this is valid only up to a point that affects solely the system administrator. By using symbolic links, the users need never be aware that they are accessing shared files. AFS uses client caching of files to increase performance. AFS assumes that most files are small, that sequential access is more common than random access, and that files are accessed in bursts (where a file that is used once will probably be used again soon). Therefore, AFS was designed to cache entire files locally to the local disk of the client system. These assumptions are not valid for databases. A database that uses files for storing its data would tend to have large files that, by the very nature of a database, will almost definitely be accessed randomly. Therefore, AFS does not claim to provide any support for large database files.
The main technique that enables AFS to achieve greater scalability than NFS is the use of session semantics and call-back promises to deal with the problems of maintaining cache coherency. (A call-back promise is a token issued from the vice server that acts as a guarantee that it will be updated and thus the venus client will be notified should any other client modify the cached file. If any such modifications occur, the promise is placed in an invalid state; otherwise it remains in a valid state. [Coulouris et al., 1994]) When an opened file is found to be in the shared filesystem tree, the call is passed to venus (the client module for AFS), which will attempt to read the file from local cache. If the file is found, and its call-back promise is still valid, then the local file is used; otherwise a call is made to the vice module on the server to retrieve the file. When the client closes the file, the file is checked. If it has been modified, the local copy is flushed back to the server, which will then send calls to all clients who have cached copies of this file, so they may invalidate their call-back promises for that file. To avoid problems caused by losing an invalidation request message, the client modules will automatically invalidate any call-back promises for files from servers with whom there has been no communication for more than period T (usually around 10 minutes). This basically gives a window of T during which a file open-write-close operation may be performed by two different clients, resulting in only the last close taking effect. The AFS solution provides far better scalability than NFS. On the same benchmark with 18 clients, AFS exhibited only 40% server load, versus 100% for NFS. This result is even more impressive when you consider that AFS is implemented using server and client modules (vice and venus) implemented outside the kernel.

AFS allows a higher level of availability than NFS by providing replication, but this is provided solely for read-only files.

2.3 Coda

Coda was basically an adaptation of Andrew, catering to different requirements that appeared as a result of work performed during the Andrew project. One weakness of Andrew was the unavailability of read/write file replication. It was not too unusual to find that a (possibly large) number of users were incapable of using their systems due to the failure of one particular system on campus. This was one issue Coda addressed.

Instead of a certain server providing access to a certain group of files, we have a VSG (Volume Storage Group) that includes all the servers that provide copies of a set of files. This also leads to the definition of an AVSG (Available VSG), consisting of the subset of reachable servers in a particular VSG. With no replication, Coda provided performance comparable to AFS. With 3x replication and five clients, we find a 5% performance penalty, and a 70% performance penalty with 50 clients (Some of this penalty was attributed to performance tuning issues.) Thus we see an example of increasing reliability at the cost of scalability.

Coda's ability to support "disconnected operation and re-integration" is of particular interest for the support of mobile computing. As AFS and Coda cache whole files, it is possible to maintain a working set of files that the user can employ even while disconnected from the network. Coda supports many features that enable the automation of this process and the process of reintegration. In some cases, though, it may require the user's intervention to resolve a conflict if all attempts at automatic reintegration fail. So it provides tools for assisting user intervention in building the working set.

3. To Scale Even Further – xFS

3.1 The Design

A serverless filesystem currently under development as part of the NOW project at the University of California, Berkeley, xFS was designed to meet the challenges introduced by WANs and mass storage [Wang & Anderson, 1996]. It is designed to be effective as a filesystem distributed across LAN and WAN links. It handles terabytes of storage and, such, was designed to be aggressively scalable, while allowing for cache coherency to be maintained for shared files. It includes support for primary storage, in addition to the normal support for secondary storage devices. Direct filesystem support for tertiary storage is already available in commercial products that act as add-ons to existing filesystems, providing the functionality of automatic migration of cold files to tertiary storage (e.g., magnetic tape) and automatic retrieval upon demand. Examples include IBM's FSF/6000 provided for AIX NFS storage setups. xFS goes beyond this by providing this as part of a shared hierarchy spanning multiple systems and domains and including terabytes of data.

The first point about xFS that allows it to achieve this goal is its hierarchical organization of servers, among groups of systems referred to as clusters. Figure 1, on the next page, illustrates an example of such a hierarchy.

Files are served to consistency servers that then handle the management of the files within their domain. As shown in Figure 1, a consistency server would need to exist for each cluster reached via a WAN link. This reduces the overheads for the home server and avoids the need for its dealing with every client that wants files. It effectively delegates responsibility for the serving of files to the consistency servers. This "delegation" occurs for groups of files (basically entire filesystem subtrees) to avoid problems of possible state explosion. The home server is therefore reduced to dealing with the consistency servers, relieving it of all intracluster traffic.
For managing "consistency," the servers use a protocol that is designed to maximize scalability by reducing traffic. xFS attempts to enforce strict consistency on any cached files; to achieve these seemingly contradictory goals, it uses a protocol based on the requesting and granting of "ownership" of files. (Actually, it uses entire subtrees to avoid state explosion, as was just mentioned.) When a user opens a file for reading or writing, the client on that host would request the appropriate ownership (read or write) via the nearest server in the hierarchy. This request would be propagated until an owning server is reached. If multiple clients request read ownership, this is granted, and the file is in the read-sharing state (each client has a copy of the file cached locally). Upon closing the file, there is no write-through to the fileserver; clients never voluntarily relinquish ownerships they have been granted. A revoke-ownership request is sent by the server only when a client requests write ownership for a file (another example of negative acknowledgment), and the clients having previous ownership can opt to keep ownership or relinquish it. If the clients all relinquish ownership, the file would be in the write-1 state, and the client requesting write ownership is granted this ownership. Otherwise no clients are allowed ownership of the file. Ownership implies permission to store locally, and this is not granted when a file being opened for writing is shared. This ownership is managed on the basis of entire file clusters whenever possible, reducing the possibility of state explosion dramatically and allowing for greatly increased scalability.

xFS supports tertiary devices through a technique similar to that provided in handling virtual memory on secondary storage. As tables in primary storage are used for keeping track of the primary memory blocks stored on secondary storage, xFS uses tables on secondary storage tracking data blocks. Instead of a virtual address, xFS has a Block ID of the form fileId, block#. This is mapped using translation tables of the form blockID, #ofBlocks, device address. This cleanly separates disk management from file management. The only problem with the implementation of this technique is the expense of this information: it requires approximately 10 GB of secondary storage for 10 TB of tertiary storage. The designers of xFS consider the cost to be bearable with falling storage costs!

Thus, xFS is an excellent, possibly even extreme, example of a distributed filesystem designed for massive scalability, showing the feasibility of designing systems that beat scalability bottlenecks.

3.2 Open Problems
One possible problem in the previous design is that of being unable to revoke ownership of a file due to communications failure. One of two choices is available for dealing with this: ignoring the problem and assuming the client would probably agree, or, blocking the new client from receiving ownership until the unreachable client responds. The designers chose the former approach, which can be justified on the grounds that experience with Coda has shown that problems with write sharing are rare. (This may be even more true in the case of xFS that offers such a massive amount of files.) It would also be unwise to block clients from receiving ownership of a file (and effectively forcing them to wait), pending communication with another client that may have crashed and be undergoing lengthy repairs or have been rendered unreachable by a lengthy network outage.

A second problem introduced by such a massively scalable filesystem is that of data backups. How do you backup a system that can include massive amounts of data or that is distributed over such a wide area that obtaining a consistent snapshot is impossible? Providing wide-area backups and disaster recovery mechanisms is a topic of future research.

For further details on the xFS serverless filesystem, the reader should consult [Wang & Anderson, 1996], and [Anderson et al., 1996].

4. To Perform Even Faster—Caching
Caching is a well-known and effective technique for improving performance. We saw how it improved performance in both NFS and AFS, and how in NFS it led to scalability problems that were resolved by AFS but at a cost with regards to cache coherency. There are three alternative techniques for gaining the performance advantages of caching, while limiting the problems associated with maintaining cache coherency.

4.1 Sprite
Sprite is a distributed operating system developed at the University of California, Berkeley. Although it is a monolithic operating system, it has had more than one successful change to the filesystem that it employs (as we shall see in the following sections on Zebra and Sawmill). Sprite's file-
system is similar to NFS in that it maintains the UNIX model of files and file operations and its high performance is attributable to its aggressive caching. Unlike NFS (and even AFS or Coda), it provides a globally shared location-transparent filesystem (this extends to the possibility of accessing remote devices). Also unlike these filesystems, it employs a very strict cache coherency protocol.

In terms of caching, Sprite has been described as "aggressively caching" because it will cache very large amounts of data at the client side. In fact, the unusual feature of Sprite's caching is that it will vary the size of the client cache dynamically, depending on the availability of memory. This has enabled diskless clients in Sprite to achieve performance within 0-12% of systems with local disks. This large-scale caching also appears to reduce server load by 50% and network traffic by 75% [Nelson et al., 1988].

The caching in Sprite uses main memory (not the local disks, as in the case of AFS and Coda) to enable diskless workstations (quieter and cheaper). This is driven by the feasibility of larger main memories providing high hit rates (80% read hit rate for 1 MB of client cache). Sprite also manages to maintain UNIX file semantics in spite of the existence of cached copies of data at multiple systems. Under Sprite, for all processes running on all the workstations, the same file semantics apply. As for processes running on a single timesharing system, Sprite achieves this by disabling client-side caching on a file-by-file basis for any files that are concurrently opened by more than one client, and at least one has opened the file for writing. Sprite can do this without incurring too heavy a performance penalty because this situation is fairly rare [Nelson et al., 1988].

Another feature that enables Sprite's caching to be an effective performance enhancement is its handling of writes. A third of all file operations are writes, so write-through policies cannot reduce server load by more than two-thirds, making delayed write-back seem a better choice [Nelson et al., 1988]. Also, 90% of files remain open for less than 10 seconds, so write-on-close policies, as employed by NFS, AFS, and Coda, do not really help reduce the amount of server traffic [Nelson et al., 1988]. Sprite uses a write-back policy for client caches that mimics the UNIX 30 second timer—modified data in client caches that have been unmodified for 30 seconds are written back to the server, and there they will be written to disk 30/60 seconds later. This achieves the benefits of delayed write-back while maintaining a level of reliability compared to that of the local UNIX filesystem. It also reduces server traffic, by avoiding writing modified data and then writing them again if modified immediately afterwards (only modified data that have remained unmodified for the 30 seconds are written back). For more information on Sprite and its filesystem, see Baker & Ousterhout (1990), Welch & Ousterhout (1989), Nelson et al. (1988), and Welch (1992).

### 4.2 Amoeba

Amoeba, like Sprite, provides a globally shared, location-transparent filesystem. But, unlike Sprite, which was influenced by the file-intensive workstation environment, Amoeba is based on the processor pool model, which does not have optimization of file-intensive applications as a primary concern.

Amoeba's standard fileserver, known as the Bullet server, does not implement client-side caching. All accesses to remote files require network transfers, increasing latency and network load. This is surprising considering that Amoeba does not implement UNIX file semantics. It allows the opening of a file, the writing of initial contents, and then the commitment of the file, after which it is available for access by others, but is immutable. The only other operation that can then be conducted to change the file is its complete deletion. This would have allowed simple implementation of client-side caching, which is not implemented in the Bullet server!

Amoeba splits the directory (naming) service from the file (storage) service, allowing the directory service to refer to remote files, or even nonfile objects. It also allows the creation of files without names (for use as a temporary store, for example).

In terms of performance, Sprite with caching enabled can generally outperform Amoeba on almost all primitive file operations, and yet it loses if caching is disabled (except for the create-delete operation of named files, where Amoeba has to update directory entries). Another side effect of Amoeba's file semantics is that it suffers great overheads when attempting to emulate such concepts as append-only files and files opened as read-write. But on the positive side, its semantics greatly simplify file replication. (The Zebra filesystem we discuss below, which addresses increasing availability through data redundancy, was implemented recently under Sprite.) Douglas et al. (1991) provides a more detailed comparison of Amoeba and Sprite, including a discussion of their respective filesystems.

### 4.3 Leases – Soda

Soda is the second distributed filesystem that implements client-side caching, and it is a good example of the use of leases for dealing with cache coherency problems. Leases were first described by Gray, and the concept was independently discovered by the developers of the ECHO distributed filesystem. Basically, a lease is a contract that grants its holder a guarantee of access to some good for a limited period of time. In distributed filesystems, a lease is a guarantee that the server will not update the file without the permission of the holder of the lease for the duration of the lease's validity.
When a client performs a read, the client receives both file data and notification of when its lease will expire. For the duration of the lease’s validity, the client can use the data without worrying about possible modifications to the data. Only when the lease expires does the client need to contact the server to check the validity of its cached data and to obtain a new lease.

Should the server receive a request to modify the data, it will not do so until it has successfully notified all holders of valid leases, or until all outstanding leases expire. This protocol solves the problem of maintaining cache coherency and has the added advantage of being inherently tolerant of server faults. Unlike Sprite and AFS, where server crash may mean loss of valuable state information, the lease protocol will not lose any server state if a server’s reboot can be guaranteed to be longer than the lease duration. In other words, upon reboot, all outstanding leases will expire before the server resumes operation. (It is interesting to compare this behavior of automatic loss of ownership, to the explicit nature of revoking ownership in xFS.) The tuning of performance involves careful selection of the lease duration; leases that are too short increase the requirement to reacquire the leases, but if the lease is too long, we can get substantial delays for file updates when communication fails with a client holding a valid lease.

Soda was implemented as a layer above NFS on PCs running Linux at the University of Sao Paulo, Brazil [Kon & Mandel, 1995]. The issues Kon and Mandel did not directly address include the method for synchronizing clocks and the requirement of implementing write-through to the server.

5. To Beat the Disk

Over the past few years, processor performance has increased at an exponential rate, and in some respects, communication technology has advanced faster than processor speeds. Storage technology has also improved, but any substantial improvement has been mainly in the increase of storage capacities. There have been limited performance improvements in terms of throughput, but even more importantly, there has been limited improvement in seek performance for magnetic media. Without major breakthroughs in storage technology, this situation is not likely to improve over the next ten years.

The use of caching has improved the performance for random reads, but the situation for sequential reads/writes needs other techniques (e.g., striping). For small random writes, the situation would seem to be hopeless. Fortunately, a new filesystem architecture (the log-structured filesystem) promises to resolve even this problem. The description of some modern distributed filesystems will show how these techniques, when combined with networks and distributed systems, can produce some impressive results.

5.1 Striping

The data are written in blocks, and these blocks are written in sequence, one to each storage device. Then we start again at the first storage device. This provides us with gains in terms of performance and availability.

5.1.1 Striping for Speed. The organization of blocks across storage devices is similar to that of cylinders in hard disks. This is how striping provides increases in throughput, and hence improvements to the performance of sequential reads and writes, where caching may not be very effective. By striping the data across multiple storage devices, we can read a set of adjacent blocks in parallel, achieving higher throughput than can be supported by the bandwidth of a single device. This technique is widely applied in disk array subsystems.

5.1.2 Striping for Availability. If we include a slight degree of redundancy with striping, we can achieve tolerance of single node failure at the expense of only one extra node in the array. This is the same concept applied in RAID technology that has become very popular over the last few years. A simple illustration of the use of this technique is shown in Figure 2.

![Figure 2. Example of striping with redundancy for error recovery](image-url)
that the parity is computed for the stripe and overwrites the existing parity; no update is required. Arranging writes to guarantee such writes is not practical for normal filesystems that provide fixed locations for file blocks, but this is not a problem for logging/log-structured filesystems.

![Diagram of RAID architecture](image)

**Figure 3.** An example of the increased (4x) cost of small writes in a RAID architecture.

### 5.2 Log-Structured Filesystems

The use of logs in filesystems is not a new idea. In fact, several commercial operating systems, notably AIX, implement "redo logs" that allow them to do a better job of recovering after a crash by having a record of what they were about to do. But a log-structured filesystem (LFS), as described by Rosenblum and Ousterhout (1990, 1991) is one that is itself structured as a log. When we write a block of data, we write it at the next storage block in sequence. Therefore we do not have a fixed location for a block of data, but every time we modify a block, it is written to a new location. This basically transforms all writes to sequential writes and effectively resolves the problem of random small writes being limited by seek speeds of magnetic media. But such a scheme raises a few obvious questions, not least of which is "Won't we run out of space?" The answer is simple. As blocks are rewritten, their old locations become invalid, so we can reclaim their space.

We have two means of achieving this. One is to thread the new writes between existing valid blocks to use the invalid blocks in between, or we can simply compact the invalid blocks to reclaim contiguous areas of storage. Neither technique is acceptable. The former would lose the benefit of writing to adjacent blocks, and the latter would lead to problems in scheduling this compaction. The LFS, as described by Rosenblum and Ousterhout (1990), uses a combination of these two techniques that is reminiscent of segmentation with paging. In effect, they organize blocks into segments, and they reclaim entire segments when all the blocks therein become invalid.

The implementation of logging filesystems has been found to achieve comparable performance to conventional filesystems for reads but to perform an order of magnitude faster for random writes and considerably faster for sequential writes. This has led to the near-immediate acceptance of this technology for systems that deal with continuous media (e.g., systems with multimedia support). In fact, the log-structured filesystem was the first filesystem architecture to be used in the filesystem for the Pegasus operating system [Bosch & Mullender, 1995]. It was designed to be easily modified to accept new storage schemes, but the first choice was the logging filesystem.

### 5.3 Beating the Disk with a DFS

Through the use of striping and LFS (naturally these are combined with caching, but we focus on the advantages brought through applying striping and LFS in particular), distributed filesystems can be designed to overcome the disk bottleneck entirely. In fact, those techniques put the bound on the communication subsystem, the processor, or simply storage capacity, all of which are increasing at a rate far greater than that of the disk subsystems, offering a better chance of benefiting from these advances.

#### 5.3.1 Swift

Swift is a distributed filesystem whose main goal is high-performance storage of large objects – with increased reliability. It applies the concept of striping with parity, but in this case, the striping is across multiple systems on a network. This achieves increases in performance and reliability. It can tolerate the loss of any single storage node. In tests, the Swift architecture was shown to be capable of providing higher performance than that of the local disks. In fact, it was in the range of a 250% to 280% increase over local disks for reads and writes. (This figure dropped, but only for reads, when SunOS 4.1.1 was used with an improved filesystem.) The limiting factor during these tests was found to be the network [Cabrera & Long, 1991], but again gigabit networks are now on the not too distant horizon.

![Diagram of Swift architecture](image)

**Figure 4.** Components of the Swift Architecture

5.3.2 Sawmill. The Sawmill logging filesystem is described by Shirriff and Ousterhout (1994), and basically it is an implementation of a filesystem on a new storage architecture known as RAID-II. This solution uses specialized hardware to provide a highly efficient server that has very high throughput exceeding that of its local disks and with a data path that goes directly from the controller to the network, minimizing data passing through the server to metadata. The most important feature of this filesystem is that it applied the use of a logging filesystem to this RAID architecture with a segment size equal to the data stripe size of the disks — meaning that all writes are in the optimal size for RAID storage. We can never encounter the problem of the 4x more expensive parity update.

It is therefore not surprising that Sawmill achieves up to 21 MB/s sustained read throughput and 16 MB/s sustained write throughput. In fact, with a single client making 1 KB to 10 KB writes, the Sawmill filesystem is up to 10x faster than the raw disk performance (no seek costs for writing) and 20x faster than RAID performance. Current UNIX filesystems can use only 5% to 10% of the disk bandwidth when writing, whereas LFS can use 70% (even when including segment-cleaning overheads) [Rosenblum & Ousterhout, 1991].

5.3.3 Zebra. The final example that will be presented is more recent than the previous two examples, but in the context of this article, it would seem to be the natural next step.

This is the Zebra filesystem, whose architecture resembles the Swift architecture (compare Figures 4 and 5), but has one very major difference. Zebra, as described in Hartman and Ousterhout (1992), combines the concepts of network striping with the benefits of a log-structured filesystem.

As in Swift, the data are striped across the systems, but the stripe is now equivalent to a segment in a log-structured filesystem. This means there is never a parity update. Clients write out the stripe fragments to the different storage servers in one whole stripe, after calculating the parity locally. This implies no server overhead per file or file block, and the storage servers do not interpret the streams of stripe fragments they are receiving. The file manager and stripe manager handle metadata, and stripe cleaning respectively. (Stripe/segment cleaning is an open issue in LFS design, and a discussion of its policies is beyond the scope of this article. Simply stated, it is the process of reclaiming entire stripes/segments whose data blocks have been invalidated.) They are therefore critical, and Zebra implements backups for both of these services. To conclude, we briefly list the advantages predicted for the Zebra "striped filesystem" as described in Hartman and Ousterhout (1992):

- Scalable performance. File transfer rate is proportional to the number of storage servers used.
- Cost-effective servers. No special high-performance hardware is required, and performance can be improved by adding more servers, without even replacing old ones.
- High server efficiency. No server overhead on a per-file or per-block basis, and writes always in large transfers.
- Simple parity mechanism. The clients calculate parity before writing a stripe, and no expensive parity updates are required.
- Uniform server load. This form of striping should reduce the variance of server load, effectively providing a form of load balancing across the multiple storage servers.

6. Conclusions

We have covered numerous examples of distributed filesystems and the techniques used in them to achieve high performance and availability while still providing the functionality that a user expects of a filesystem. We have described how distributed filesystems can be designed to beat the limits of scale and how they can be designed to handle the problems introduced by client-side caching of data (a technique that dramatically increases performance). We concluded with examples of two techniques that have been applied to filesystems in general, but, when combined with distributed filesystems, can achieve dramatic performance advantages that, in turn, beat the bottleneck imposed by the slow technological advances in secondary storage performance. We then described three filesystems that apply these two techniques, ending with a filesystem that combines in one very elegant design many of the features we discussed. As needs grow for larger, faster, and more reliable filesystems, we will see more
and more of the concepts discussed in this article being applied to more mainstream filesystems. In the shorter term, the increasingly rapid growth of the gap between processor performance and I/O may well drive the adoption of such techniques, possibly combined with solutions such as the use of NVRAM caches to supplement magnetic storage. (See Baker et al., 1992, for more details on this subject.)

7. References


Additional Reading


In the first column, I promised that I'd look for other ports to Java. I did find a few. The OSF has ports to HP/UX 10.01, DEC UNIX 3.2, Sony NeWs, and AIX 4.1 (for Bull). I don't have any of these platforms, but I know that DEC signed a license with Sun at the end of September. Try <www.gr.osf.org/java/javaport/JDK-1.0.1/>.

There is also guava (don't laugh) at <ftp://summit.stanford.edu/pub/guavac/guavac.tar.gz>. Guavac is written for GCC 2.7.2 and will (probably) run under SunOS. There is also a precompiled binary for Linux there. I won't have time to try this before I leave for Iceland.

Networking Made Easy

The topic for this month is networking. I noticed Hal Pomeranz's coverage of Perl networking in his column and realized how much Perl networking looked like C. Not surprisingly, Java doesn't look at all like C.

For one thing, the Java developers were writing with one protocol in mind - TCP/IP. This eliminates the need to specify the protocol type when creating a socket. Also, Java provides input and output streams that make reading and writing, whether from a socket or a file, easy.

On the client side, you must create a socket, which requires the hostname and port addresses. The hostname must be a string and can also be the dotted decimal notation quoted to turn it into a string. Creating the socket will succeed if the host is reachable and has a server running on the given port.

Once the port is open, the client can get the socket's InputStream for reading or its OutputStream for writing. These streams permit reading one or more bytes of data. But wrapping these streams in other Input or OutputStream classes makes life easier.

```java
import java.io.*;
import java.net.*;

public class DateClient {
    public static final int PORT = 13;
    // the daytime port

    public static void main(String[] args) {
        Socket s = null;
        String line = null;
        try {
            // Create a socket
            s = new Socket("localhost", PORT);
            // Create stream for reading this socket.
            DataInputStream sin = new DataInputStream(s.getInputStream());
            line = sin.readLine();
            if (line != null) System.out.println(line);
        }
        catch (IOException e) {System.err.println(e);} // Always be sure to close the socket
        finally {
            try { if (s != null) s.close(); } catch (IOException e2) {
        }
    }
}
```
This example, DateClient.java, creates a socket connected to the daytime port on the localhost. It could be fancier by accepting a command line argument for the hostname, but this example keeps things simple. If creating the socket is successful, you get its InputStream, and use it as the argument for creating a new DataInputStream. The advantage here is that you can read a line at a time from the DataInputStream as a string, then print it to the standard output, which is referenced by the System.out class variable. Being good citizens, we close the socket when we are finished.

The try and catch statements handle Exceptions, the Java (and C++) way of receiving error notification. If any statement within the curly braces following the try generates an IOException, the catch clause is executed immediately. The finally clause always gets executed, whether the catch clause does or not. The Java compiler enforces catching exceptions, so you will be reminded if you call a method that may throw an exception and you have forgotten to catch it.

**DateServer**

UNIX systems have a daytime server built into the inetsd program. On most systems, the daytime server will not be disabled. Other internal servers, such as chargen (the character generator) and echo (pretty obvious) have been disabled on many UNIX systems because they can be abused by a denial of service attack.

If you don’t have a daytime server, you can write your own in Java with a little effort. The server has more work to do than the client. You create a ServerSocket, and start a Thread that listens (using accept()) for connections. On a more complicated server, each connection would run in its own Thread. But all we want to do is return the date, which can be done quickly without a separate Thread.

```java
import java.io.*;
import java.net.*;
import java.util.Date;

public class DateServer extends Thread {
    public final static int PORT = 13;
    protected ServerSocket listen_socket;

    // Create a ServerSocket to listen to;
    public DateServer() {
        try { listen_socket = new ServerSocket(PORT); }
        catch (IOException e) {
            System.err.println("Exception creating server" + "socket: " + e);
            System.exit(1);
        }
        // fire up the Thread's run() method
        this.start();
    }

    public void run() {
        PrintStream out;
        try {
            while(true) {
                Socket client_socket = listen_socket.accept();
                try { out = new PrintStream(client_socket.getOutputStream()); }
                catch (IOException e) {
                    try client_socket.close(); catch (IOException e2) {
                        System.err.println("Exception while getting" + " socket streams: " + e);
                        return;
                    }
                    out.println(new Date());
                    client_socket.close();
                }
            }
        }
    }
}
```

In the constructor for the DateServer class, we create a new ServerSocket, catching any IOExceptions and exiting if they occur after printing an error message. The Thread is started with this.start(), passing execution to the Thread’s run() method.

```java
public void run() {
    PrintStream out;
    try {
        while(true) {
            Socket client_socket = listen_socket.accept();
            try { out = new PrintStream(client_socket.getOutputStream()); }
            catch (IOException e) {
                try client_socket.close(); catch (IOException e2) {
                    System.err.println("Exception while getting" + " socket streams: " + e);
                    return;
                }
                out.println(new Date());
                client_socket.close();
            }
        }
    }
```
The `run()` method of the `DateServer` does the grunt work. The while loop blocks at the `accept()` method until a client connects. Then we get the `OutputStream` and wrap it in a `PrintStream` while checking for `IOExceptions`. Notice that the later catch could be used, but here we are sending a different error notation. Once the `PrintStream` is successfully created, we simply use the `println()` method to send the Date, after it has been converted to a string automatically by `println()` calling the `toString()` method for us. Again, we close the socket, return to the beginning of the loop, and block again listening.

A server that would take more than a few seconds to carry out its work should create another object, which has its own `Thread` to handle its conversation with the client.

```java
// Start the server up, listening on an optionally specified port
public static void main(String[] args) {
    new DateServer();
}
```

The end of the `DateServer` class is a simple `main()` method that calls the `DateServer()` constructor and gets the ball rolling. Running this class will fail if you already have a daytime server. If you still want to experiment with it, change the `PORT` variable in both the client and the server.

**Another Way**

Like Perl, there are many ways to do things. Perl excels in string handling and output formatting, things that are rather weak in Java. Java has yet another way to handle networking. You can create URL objects and use them to talk to servers. But that's a topic for another day.
It’s cloudy, and in the near distance, sirens wail through the high-rise canyons. I can see Lake Michigan as a tantalizing green below the horizon, but I know it is too cold for swimming. And downstairs LISA 10 continues.

In an earlier column, I talked about how a program committee selects papers for a conference. Now I am at the conference, but totally subsumed by the Invited Talks track, for which I am co-coordinator with Kim Trudel. So far, everyone has shown up and done well. Believe it or not, I lost sleep over this before it started. It’s hard to know what others might like to hear and whether the selected speakers will be interesting or boring.

I did get to hear the keynote, from Dick Lampman of HP’s research labs. Lampman gave us an HP view of the future, with ubiquitous computing and networking. He emphasized the availability of high-speed networking to the home via cable. Later, someone asked about problems with cable networking, that is, that current cable providers can barely provide a decent signal to the home, much less support high-speed networking. Lampman agreed, but said that HP is working with cable vendors in hopes of improving the situation.

Cheap, high-performance processors will be used in almost everything, and 60 gigahertz low-power microwave radio will create a home- or small business-wide backbone for networked devices. The Web and HTML would provide the communications protocol at the application level. The term “Java” was not mentioned, but it seemed obvious that something would be happening to generate the data.

Standards

The standards panel went off much better than I had hoped. Lee Damon of Qualcomm had guaranteed some sparks, with adamant opposition to the current standards process. Nick Stoughton championed the POSIX process and encouraged participation. Louis Imershein, of SCO, provided a vendor perspective, although from an “in-the-trenches” software designer’s viewpoint. Nobody went to sleep, and people left realizing that if they didn’t like what was happening in standards, the way to make changes was to participate at some level. Writing white papers about desired features or processes is one way of making yourself heard (or email <nick@usenix.org>).

Web Servers

Dan Klein gave an enlightening (and energetic) talk about scaling Web servers: lots of common sense, plus some not so obvious advice for making a Web server run smoothly, including making sure that your upstream connection to the Internet is not the bottleneck. Automating tasks is important. Klein has a cron job that watches system load, which occasionally heads for deep water. Load averages greater than some “normal” value indicate that it is time for a reboot, which the system will do automatically. Logging, including compressing and summarizing, is all run by cron.
A Technical Mistake

T-shirts showing ATM shredding cute looking IP packets were a hot item at the conference. Characters on the shirt are shown pondering the meaning of ATM: A Technical Mistake or A Tariff Mechanism. Peter Van Epp, of Simon Fraser University (British Columbia, Canada) both wore one of the shirts and talked about SFU’s experience with ATM.

For SFU, ATM was the right thing to do. The local telecom had converted to ATM-ready switching equipment and promoted 10 megabit/second (Ethernet speed) links for wide area networking. Over time, Cabletron hubs were connected to a campuswide ATM backbone. SFU avoided one big ATM nightmare by using a single vendor (Newbridge) for the ATM equipment. They also used VIVID, instead of LANE (LAN emulation), which permits subnetting instead of pretending to be one gigantic LAN.

Do It Yourself

Celeste Stokely talked about the ultimate do-it-yourself project — becoming self-employed. Reading Scott Adam’s The Dilbert Principle recently had provided tremendous insight into why I had made the change so many years ago, besides making me laugh with every page. Stokely focused on the more practical issues of succeeding as a consultant, for example, starting with enough savings to support yourself (and any significant others) for at least a year.

Stokely claims (correctly) that succeeding as a consultant rests 20% on technical skills and 80% on business skills. Although having skills that are very much in demand is important, you must be able to meet with clients, negotiate contracts, keep good records, handle your own accounts receivable, and write good reports. She did not fail to mention politeness, professionalism, and, above all, good hygiene (smelly consultants, whether it be perfume or infrequent bathing, will lose clients).

SunSITE

I had wondered what the connection between Sun and SunSITEs was, and I learned from Stuart McRobert, Imperial College, London. Sun helps by donating equipment, but what the various sites choose to post is up to them. At Imperial College, it sounds like they collected everything and put it up for Web, FTP, ftp-mail, Gopher, and even NFS (so you can upgrade Linux, for example, by simply mounting it).

The site started modestly (a spare drive on a VAX 11/750) as a collection of software and documentation for local use. Over the years, the server has been upgraded, with various experiments, mainly in making filesystems and the disk subsystems work faster. The latest version runs on two Sun servers, an “old” SPARCserver 1000 with eight processors and a new Enterprise Server 6000 with six 167 Mhz UltraSPARC processors. To keep the newer system well balanced, there are 3 GB of memory and a RAID device supporting five SCSI channels to 200 GB of disk.

Keep in mind that this is a hobby, in that Stuart and the other people who keep this running do it in their “spare” time. But it is nice to have such cool toys. A couple of us asked them to publish their results. For example, they experimented with various RAID levels and use the Trans filesystem (a logging filesystem) so reboots don’t require hours of fsck. There are times when fsck (which is still used for some of their filesystems) gets stuck and core dumps. In those cases, the easiest way to move on is to use clri to clear the inode (I wondered if anyone still used clri; be cautious with this one at home).

Whips!

Whoops, I meant WIPs (Works in Progress). This fast-paced session, chaired by Adam Moskowitz, moved almost as fast as the name implied, so fast I missed getting notes about all six speakers (but the USENIX Web site is supposed to have abstracts and email addresses). I do recall a new encryption library providing a higher level API by a student of Dave Patterson’s at UC Berkeley, Eric Anderson, and a scheme for creating “Mail Appliances,” essentially splitting a single email server into many to remove a single point of failure and improve performance.

Mike Richichi of Drew University wants to create Perl libraries or modules for administering to Novell Directory Services from UNIX. A member of the audience suggested using the Java extensions instead. Then Neil Groundwater of Sun (Rocky Mountain) talked about the network management APIs being designed for Java while showing off a cute, pen-based system (not Sun’s) and used Netscape to show off the Java examples.

Friday morning brought Todd Heberlein to talk about intrusion detection. Heberlein gave us a history of the topic from the very beginning (a paper in 1979) to the present. He compared three different mechanisms for intrusion detection, all based on monitoring audit or network logs. The older model was to look for unusual events. The latest version examines the events related to patterns of abuse, for example, using a SUID program to do something other than what it was written to do.

Perl Hacker

Randal Schwartz spoke to a large crowd about his felony convictions. While working as a contractor to Intel in Ore-
gon, Schwartz became enmeshed in an attempt to “do the right thing” and wound up paying lawyers and others about $180,000, plus receiving a sentence of 30 days in jail and 480 hours of community service. His crime? Collecting a password file from a machine he had administered to in the past and suspected of being insecure (it was, but he should have gotten written permission first).

Schwartz pointed out that 47 states passed laws making unauthorized access or modification of computers illegal the year after the “War Games” movie came out. The law in Oregon is very vague, and the use of someone else’s calculator without permission could be construed as a felony. Other states are as bad. Note that the method he used to collect the password file, even though well intentioned, would have been a federal crime also.

Closing Address

Rob Kolstad gave the closing address to a packed house. I brought my wife to this, and she told me that his railings against marketers were right to the point – for example, people who want to use multiprocessing “because they have an extra slot on the motherboard” or the folks who insist on having an NT Web server at UUNET even though it would run twice as slow. To discourage them, UUNET offered the service at twice the going rate – and they accepted. Go figure.

Sunshine

I’m finishing this back home, just before I go off to Iceland and Romania to teach. Here it’s in the low 90s. In Iceland, well, it’s not quite freezing, but rainy. In Romania, the Web says it’s in the 50s, but raining. Wish I could stay home, but there’s a job to do (UNIX and Internet security is not getting any easier).

By the way, if you have ideas about invited talks for next year’s LISA, write to me or to Pat Wilson <paw@phibes.dartmouth.edu> and tell us about them. See you in Anaheim.

Reports on the Tenth Systems Administration Conference (LISA ‘96)

Chicago, Illinois
September 29 - October 4, 1996

Refereed Papers

Summaries by Mark K. Mellis
<mkm@mellis.com>

Security

Priv: Secure and Flexible
Privileged Access Dissemination

Brian C. Hill, University of California, Davis

Brian discussed the challenges faced by aggressive delegation of system administration tasks in the UC Davis UNIX community, consisting of 38,000 users whose systems are managed by a core team of six system administrators assisted by help desk personnel, operators, lab assistants, and lab developers. He presented a survey of mechanisms used to support access restriction and currently available software, both commercial and free, that implement those mechanisms. Brian payed particular attention to the role that control over command line arguments plays in flexible delegation, and pointed out the problems inherent in allowing delegates to run interactive commands with shell escapes. He also described auditing capabilities found in existing implementations. He was motivated to develop priv because no existing system offered the desired set of features over the range of platforms present at Davis and at a price a university could afford.

Priv is implemented as a SUID wrapper program and a policy database, written in a termcap-like format and accessible as a regular file or via NFS. It supports inclusive or exclusive restrictions, restrictions based on username, group membership, net group membership, host type, and OS type. Priv optionally logs via syslog and can capture and log sessions for later administrative review. It also allows control over the umask and environment variables, permits commands to be executed with uids and gids other than root, and uses a template for command rewriting. Perhaps the most powerful feature of priv is its ability to control command line arguments: process ids, files, and login names – each of which can be validated and resolved to forms resistant to exploitation. For example, file names are resolved to their inode and device to prevent spoofing via symlinks, and login names are resolved to uids. Priv currently builds on eight different operating system families.
Priv doesn't attempt to provide network security although it is carefully coded to avoid known vulnerabilities of SUID programs. Future capabilities foreseen for priv include a system call trapping mechanism to allow the secure use of programs with shell escapes, time of day restrictions, one-time-password support, additional shadow password support, and default templates.

Priv is currently available for anonymous FTP at: <ftp://ftp.ucdavis.edu/pub/unix/priv.tar.gz>

**The Igor System Administration Tool**  
**Because Every Mad Scientist Needs an Assistant**  
*Clinton Pierce, Decision Consultants Inc.*

Clinton presented Igor, a tool for executing commands on many diverse computers concurrently, and with a measure of platform independence. Igor is currently being used on more than 850 hosts in Ford Motor Company’s Powertrain Division.

Igor provides, on the one hand, a rich standard execution environment for sysadmin processes on a variety of platforms, and on the other hand, a friendly and productive “console” from which to initiate and control those processes, and on which to receive status. It insulates the system administrator from differences in vendor-preferred root shells, paths, etc., and manages the dirty work of running tasks on many machines concurrently, such as tracking lists of hostnames, monitoring completion status, and so forth.

Igor is a client-server application, with the client GUI written in Tcl/Tk’s wish (supported by Perl), and the server in Perl. The server runs on each of the systems to be managed, communicating with the front end on a designated port. It uses traditional rexec-style authentication. Using the GUI, sysadmins select a list of hosts on which to operate. They then select an existing script to run (either shell or Perl) or enter ad hoc commands. A mouse click on the “Run” button causes the commands or script to be propagated to the selected hosts and executed in parallel. The standard error and standard output from each host are captured for later examination. Output can be summarized with an integral regular expression filter. A status indicator keeps the sysadmins apprised of progress throughout the run. Igor supports duplication of past runs, saving of ad hoc sessions for future reuse, aborting runs in progress, and repeating runs on subsets of the original host list.

Igor has only six commands of its own:

- `do` - runs a shell script
- `EVAL` - runs Perl commands
- `openfile` - begins a here-document
- `closefile` - terminates a here-document
- `id` - reports the version of Igor in use
- `quit` - quits

The rest of the language is native Perl, with the standard runtime Perl environment available.

Although the documentation for Igor is available at <http://www.dcicorp.com/~clintp/igor/igordoc.html>, the software itself is awaiting approval for release.

**Centralized Administration of Distributed Firewalls**

*Mark Miller and Joe Morris, Bell Atlantic*

Mark and Joe described Bell Atlantic’s firewall architecture and management process. They detail the products used, configuration strategies, monitoring techniques, and mechanisms for appropriate delegation of firewall responsibilities.

In November, 1995, Bell Atlantic had five different firewalls being managed by eight people, each with its own security policy and stance. At the time the paper was presented, Bell Atlantic had ten similarly configured firewalls being managed by three people, implementing a common security policy. How did it happen?

The project began by defining a common architecture that would be flexible enough to accommodate business requirements while maintaining desired levels of security. The architects chose Sun and Cisco as their standard hardware platforms for ease of support, availability of tools, and existing user knowledge. Following an extensive survey of commercial software offerings, they selected Firewall/1 from Checkpoint as their firewall product. Firewall/1 is well respected, supports remote administration, has a GUI that facilitates communicating security issues with management, and was already deployed at some Bell Atlantic sites.

They configured the host operating systems for firewall use by following the standard practice of removing unnecessary or dangerous applications and installing the commercial firewall software along with supplementary packages. In addition to Firewall/1, they used ssh to facilitate secure communication with remote firewalls from the central administration point, tiger (from Texas A and M University) for system audits, and Tripwire (from Purdue University) to assure the integrity of system files and permissions. Logging is performed with syslog, modified to reject connections from other systems. Logs are forwarded to the central site and are also printed locally. The entire host software installation process was documented and scripted such that a system can be installed in less than an hour.

Problem notification is facilitated by a custom application that parses the log files on the central error message collector. It performs various actions based upon the significance of the log messages, including identifying and preserving
unusual log messages, and paging operations staff when critical conditions occur. Log file review is carried out on a 7x24 basis by members of the Bell Atlantic Network Management group. The data reduction capabilities of the system allowed 17,000 messages to be reduced to 120 noteworthy events on a recent day.

A comprehensive administration process was developed to ensure the operational integrity of the system, including provisions for physical security, disaster preparedness, and change control procedures.

All in all, Mark and Joe have described a complete, thorough network firewall implementation. The paper and accompanying material are available at <http://www.bell-atl.com/BC>.

**Account Management**

**Shuse: Multi-Host Account Administration**

*Henry Spencer, SP Systems*

Henry discussed the Shuse (pronounced “shoes”) user management system, currently supporting 20,000 user accounts at Sheridan College, located in the suburbs of Toronto. Shuse is written almost entirely in Expect, using Telnet for interprocess control, NFS for bulk data transfer, and only 3 auxiliary programs (about 100 lines) written in C to provide functions not present in Expect.

Sheridan College, expecting large increases in its user population, found account management with NIS was becoming unwieldy. The college also needed functionality not routinely provided with NIS: initializing home directories, tracking office and phone number information, and so forth. Commercial user management packages were costly and limited in availability for diverse platforms. Even the freely available software depended upon commercial databases. So, with Henry’s help, they decided to roll their own.

The Shuse central database lives entirely in memory, managed by a single-threaded daemon. Henry observes that “RAM is cheaper than database packages nowadays” and that this approach yielded “vastly easier debugging.” It has scaled up quite effectively. Since read-only transactions are satisfied from memory, the on-disk representation can be optimized for ease of updates. Each record in the database is contained in its own file, all located in a single directory. Reading the 20,000 user files into memory at system startup takes only a few minutes.

As updates are made, the shuse daemon (shused) communicates with auxiliary programs on remote servers, which then initiate action as necessary to conform to what the central database thinks they should look like. This adds a measure of robustness, since a missed update will be corrected at the next transaction. Rather than invent a new low level communications protocol, the processes communicate via telnet processes spawned by Expect. Likewise, creating directories, adjusting quotas, and so forth are easily managed by driving the normal system administration utilities from Expect.

There is continuing work on Shuse, particularly in managing the modularity of the code and in exploiting new features of Expect. Unfortunately, this testimony to the UNIX tradition of doing unanticipated things with general purpose tools is not freely available.

**The Design and Implementation of a Network Account Management System**

*J. Archer Harris, James Madison University and Gregory Gingerich, Bell Atlantic*

Harris and Gingerich described the Network Account Management System (NAMS), used by the James Madison University Computer Science department to manage user accounts for 300 users on fifty computers. NAMS allows each user to have a single “network login” while accommodating system specific variations, such as differences in home directory, login shell, or password.

In order to manage their user account information, the authors developed the concept of clusters, groups of computers for which items of user information are identical. For example, this model might allow an individual to use a common password on one set of machines (one “cluster”) and a common home directory on another (potentially overlapping) set, while maintaining the same login shell on all the computers, and having a single entry in the central administrative database.

A variety of existing account management systems were surveyed, including NIS, NIS+, Kerberos, and TME from Tivoli. Although NIS allows for exceptions, they are managed on a host-by-host basis by entering exceptions in the individual password files. This approach doesn’t scale well for diverse environments. Likewise, NIS+ doesn’t support exception cases in a scalable manner. Kerberos addresses but a single component of the network login problem, that of authentication. Furthermore, it isn’t widely available. Tivoli’s TME was also examined, and although it supported some of the features desired, it wasn’t supported on the required range of hosts, nor did it fully accommodate the cluster concept.

NAMS was developed with the cluster model in mind. Its database contains records for each network account, along with access permissions for those records, and per-cluster information. Information not duplicated in the per-cluster items is used for all machines. As information changes in the central database, it is propagated incrementally to the individual computers, where it is stored in the traditional UNIX databases such as /etc/passwd and /etc/group. The UNIX

---

**CONFERENCE REPORTS**

December 1996; *login: 31*
CONFERENCE REPORTS

SLINK: Simple, Effective Filesystem Maintenance Abstractions for Community-Based Administration
Alva L. Couch, Tufts University

SLINK was the Best Paper winner at LISA X. Alva Couch outlined how SLINK can aid in fostering community in system administration. SLINK as a tool does not enforce policies, but instead it reinforces policies of the community. Software environments are constructed incrementally, by making changes necessary to install one package at a time without requiring complete control over the environment, but it can easily merge environments. Software is maintained in filesystem hierarchies that can be controlled with recursive commands including (un)link, (un)copy, and destroy.

Because SLINK's commands are very dangerous and powerful, virtual protections are used to maintain the security of the disk image. These virtual protections within SLINK allow/disallow the removal/addition of files or links to any point in the hierarchy. These virtual protections also allow for the delegation of particular administration powers and privileges without using setuid programs.


Managing and Distributing Application Software
Ph. Defert, E. Fernandez, M. Goossens, O. Le Moigne, A. Peyrat, I. Reguero, CERN, European Laboratory for Particle Physics

ASIS (the Application Software Installation Server) was presented in its beta version as a tool to assist in managing and distributing application software to heterogeneous UNIX systems. The goal of ASIS is to centralize the process of obtaining, configuring, generating, installing, and managing software. ASIS uses a repository to handle software packages for multiple platforms and versions. Software stored in the repository is partitioned into two parts: specific and sharable. The specific part contains libraries and binary files for each operating system. The sharable part reduces the storage requirement by maintaining one copy of scripts, fonts, startup files, and man pages that are usable on all platforms.

ASIS maintains a database to track products with their description, support information, platform, and commands. It includes several Perl scripts and a Tk-based editor to customize software distributions for different machines and users. These passive configurations can be configured in an automated way with a Tk-based output interface.

The Future of System Administration
Summaries by Carolyn M. Hennings <cmh@psa.pencom.com>

A New Twist on Teaching System Administration
Raven Tompkins, Indiana University

This talk provided an overview of Web-based training on systems administration topics. Visit the UNIX Systems Administration Independent Learning Project at <http://www.uwsg.indiana.edu/usail> for a look at the subject of Ms. Tompkins's talk. The university was unable to hire a full-time system administrator, therefore students and staff were functioning as novice sysadmins and spent a good deal of time learning and sharing their knowledge with others. The UNIX Workstation Support Group (UWSG) at Indiana University identified this need for a learning resource for their own use as well as for external parties. The Web is the optimum deployment medium for providing this resource.

Institute White Pages as a SysAdmin Problem
Jon Finke, Rensselaer Polytechnic Institute

Jon Finke expanded on his “Manage People, Not Logins” presentation from earlier in the day and focused on using a relational database to generate a telephone directory. This Oracle database manages information about people, rather than simply login accounts. This broad view of the information requirements of an educational institution enabled the school to build additional systems around the data stored. One of these systems generates the telephone directory and Web pages with telephone information. Additional systems enabled students and staff to update their information, increasing the timeliness and accuracy of the data.

New Fangled Phone Systems Pose New Challenges for System Administrators
Snoopy, iXOS Software GmbH

Snoopy discussed his experience installing and administering a PBX system. He enthusiastically related the challenges he faced and some of the more enjoyable aspects of having access to a PBX. An introduction to the terminology used in the telephone industry was helpful. One entertaining part of Snoopy's talk was his sharing of some pranks he was able to play with the PBX, such as ringing all the phones in the office at the same time and watching the confusion that resulted.
Invited Talks

Invited Talk: ATM: Not Just A Type of Bank Machine Anymore
Peter Van Epp, Simon Fraser University

Summary by Carolyn M. Hennings
<cmh@psa.pencom.com>

Peter Van Epp's invited talk was the perfect introduction for system administrators interested in ATM. He began with an overview of Simon Fraser University's computing environment in terms of the number of users and machines. He also outlined the university's involvement with ATM since 1993.

The speaker reviewed the basics of FDDI and switched Ethernet technologies, providing an understanding of the options that were considered. In making the decision to use ATM, they examined the advantages and disadvantages of the communication protocol. Peter Van Epp shared the following conclusions:

- ATM is production ready
- ATM should be considered if network changes are being made
- the protocol is not fully mature and standardized yet
- ATM can be used for wide area, backbone and desktop networks
- owned fiber networks provide bandwidth scalability and flexibility
- flexible billing creates the possibility of new applications
- ATM requires a low error rate transport layer to be successful.

The most powerful conclusion Peter Van Epp shared was that the use of ATM technology means a customer will not hear “but the network can’t do that.” However, the response may be “we can do that, but it will cost this many dollars.”

How to Run a Worldwide Network When You Work in the Center of the Universe
Joel Avery, Nortel

Summary by Mark K. Mellis
<mkm@mellis.com>

Joel Avery gave an entertaining and informative presentation on the administrative and technical challenges of running Nortel's worldwide mission critical network. He started his presentation by describing the size and uses of the Nortel network. Nortel has more than 138,000 IP addresses and spans the globe. It lives and dies by its network. On its ATM backbone, it carries voice, video, and data. In some locations, even the badge readers that control the door locks are on the net. The network links desktops, homes, joint ventures, and connects to the Internet. At one time, the Nortel network was disjoint, carrying many different protocols. Now it is unified and based on Internet standards.

Joel continued by stating Universal Truths of Network Administration, my favorite being “The more choices your users have, the more your phone will ring. If there are n choices, the universe splits into at least n camps.”

Next, Joel addressed tradeoffs between control and autonomy, standards vs. duplicated effort, and the effects of these factors on innovation. Certain things must be controlled at the highest level. Chief among these are network standards and addressing. Other decisions are made in a more consensuss-driven manner, with business units providing input.

Much of the interaction between headquarters and the rest of the corporation is stylized, governed by processes that provide a measure of consistency and fairness to corporate citizens. Joel commented, “The best way to develop processes is to steal them from people who are doing things better than you are.” Complementing the formal, process-driven interaction is “off-premises team building” in the local tavern or restaurant when field personnel visit HQ. Informal interactions help build good will and break down barriers.

Joel continually stressed the importance of communications. In a surprisingly apt observation he pointed out that the World Wide Web is not necessarily a communications tool, because effective communications must be bi-directional.

In summary:

- Controlling the universe requires outposts to keep the peace in their sectors.
- Each sector needs standards for predictable operation.
- When these standards are shared by everyone, the federation remains strong.

What It’s Like To Be Your Own Boss
Celeste Stokely, Stokely Consulting

Summary by Jeff S. Tyler
<jeff@psa.pencom.com>

Celeste’s talk should be mandatory for anyone considering a career in consulting. There are many potential pitfalls in making the jump from full-time employee of an established company to independent consultant, some obvious and some a bit more subtle. She covered all of them and did so in a thorough and entertaining manner.

I was particularly impressed by her coverage of two of the biggest obstacles that trip people up: financial planning and attitude. Setting up a realistic business plan and arranging your finances accordingly is pretty basic, yet many people neglect to do a good job in this area and then fall flat on their faces when they hit a dry spell.
The attitude thing is not so apparent to many: they only see the great potential of knocking down major dollars and overlook what it takes to get from here to there. Celeste advised people getting into the consultants game to have “vision, courage, determination, and persistence,” with heavy emphasis on persistence. She noted that being your own boss is a mixed blessing; you can pick your jobs (many times), but you also have to keep yourself motivated. As one who has been there, I can tell you that’s not as simple as it sounds.

She then outlined the basic business tools one needs to work as a consultant — no, not computers or net access, those come later, rather, an attorney, an accountant, a bank, health insurance, business permits and licenses, and more. Celeste stressed the value of good project planning, written and oral communication skills, and a neat appearance. She had a great line about how companies will put up with smelly employees but not smelly consultants. She urged people to dress appropriately for the job in question, not to wear a suit everywhere (or jeans either), but rather to blend in with the workforce. She put major emphasis on the importance of always working within the protection of a contract, your own if possible. She then talked of the importance of ethics, networking (the people kind), and helping out your fellow consultants. She made the point that the first time a company hires you as a consultant, it’s for what you know; after that, it’s for who you are.

Celeste closed with some advice on how to collect from deadbeats and what sort of tools and toys you might want. As a manager of a large consulting engineering group, I found that all of what she said rang true. I have not done her talk justice in this brief write-up; if you are even considering consulting, then by all means attend Celeste’s talk the next time she gives it.

**Experiences of Running a Large Archive Site**

*Stuart McRobert, Imperial College, London*

*Summaries by Mark K. Mellis*

< mkm@mellis.com>

Stuart McRobert detailed the historical perspective of the software and hardware issues encountered in managing Sun SITE Northern Europe (Sun SITE) with his partner, Lee McLoughlin. The principles discussed in Mr. McRobert’s presentation are applicable to all data storage facilities regardless of size.

Sun SITE serves as an archival repository built entirely on donations from generous contributors, including Sun, Quantum, and Trim. The site was originally run on a VAX 750 and then evolved onto a Sun 4/360 eventually to be replaced by a Sun SS1000 and an SS6000 with large RAID storage devices. Sun SITE, which supports a worldwide community, responded to the incessant demands of its users.

FDDI, ATM, and commercial ISPs were utilized to improve interconnectivity by bypassing known bottlenecks. Limitations of RAID 5 that necessitated logging, such as lengthy fsck, were discussed. Sun SITE implemented hardware changes to RAID 5 performance.

Software that was utilized to provide user services for Sun SITE included Web caches and FTP mail. Exim was the mail transfer agent used for FTP mail access to the archives. Heavy system use during software releases led to the introduction of the Squid/Harvest Web Cache.

Although surrounded by a multitude of disks, Stuart noted the simple truth that one can never have enough disk space.

**Manage People, Not Logins**

*Jon Finke, Rensselaer Polytechnic Institute*

Jon’s talk was really a cram course in tying together all aspects of data management in a university environment. At RPI, almost all data processing is managed by Jon’s group, manipulating data contained in an Oracle RDBMS. Everything from ID cards, food service, parking lot assignments, telephone directories to book store charges, library circulation, and phone switch programming is affected by UNIX system administrators. Of course, when so much of the university infrastructure is dependent upon your system, your administrative practices may vary slightly from those employed in a CAD lab or at an ISP.

The RPI system, called Simon, had its origins five years ago and has been under continuous development ever since. Simon is built on top of an Oracle RDBMS, and is implemented using Oracle SQL tools. As the system has evolved, more and more problems have been solved by adding “another report.” Through careful management of data validity and the use of data-driven policy rather than embedding policy in programs, the system’s flexibility has helped contain costs while raising the standards of service.

Just Another Convicted Perl Hacker
Randal Schwartz, Stonehenge Consulting Services

Summary by David J. Bianchi
<dj@psa.pencom.com>

Randal gave an entertaining and scary talk about how he became a felon while performing his job as a system administrator. He started the talk by describing the evening of November 1, 1993, when the police knocked on his door. The company involved was Intel, a long-term client of Randal. He described the trial and jury process with the end result of being convicted on three felony counts. His jail sentence (90 days) was delayed, but he received five years of probation, 480 hours of community service, and over $67,000 in restitution. It also cost him $180,000 in legal fees. His appeal is ongoing and may take 18 months to hear. He has gotten a lot of press along the way, like articles in Wired, UNIX Review and Dr. Dobbs.

Randal admitted to doing some stupid things, but certainly nothing illegal. He had some words of advice: never talk to the police without a lawyer present, and always maintain good (and frequent) communications with your employer. This is especially important if you are a consultant.

Randal’s purpose in giving the talk was threefold: get the word out to individuals about these laws, give notice to large institutions like Intel about their policies and practices, and get bad or vague laws changed through the legal system. There are petitions to Intel and to the state of Oregon about this case, and additional information can be found at: <http://www.lightlink.com/fors/>.

Works-in-Progress

Summaries by Bruce Alan Wynn
<wynn@psa.pencom.com>

Prop – A Simple User/Group Distribution Program
Trey Harris, University of North Carolina

Trey discovered that traditional distribution methods of system files, including NIS, break down when the number of users becomes large. In his environment, this occurred at around 8,000 users. To work around this, they developed “prop.”

Prop maintains a central database, which is periodically propagated to clients using AFS’s access control lists. An additional field was added to Prop’s “passwd” database in order to allow customization of the propagation. For example, specifying that a given user is in the “ADMIN” working group would cause that entry to be propagated only to computers that were configured to receive entries for that working group. This allows logical groupings within a single database.

This utility consists of three tools: “propclient” to pull subsets of the databases periodically, “prop” to modify the databases and queue changes, and “propserver” to make modifications to the databases.

Host Factory
Brian Bartholomew, Working Version

Similar to the “golden root” concept, Host Factory provides a mechanism of storing multiple copies of your filesystem under revision control. This allows hardware-specific differences in the filesystem to be accessed during the installation of new systems, easing the process.

Discussed in depth was the “PostGRES Filesystem,” “PGFS.” This freely available version-control filesystem provides the functionality described previously. It is available via Working Version’s Web site at <http://www.wv.com>.

Property-Based Message Security
Eric Anderson, University of California, Berkeley

Although security is an important topic in the minds of most system administrators, few take many steps to implement it. To increase the use of security measures, property-based message security was created. In this model, security properties such as “signed,” “encrypted,” and “nonreputable” are selected per message. The system then automatically applies these properties to the message, thus easing the effort on the part of the sender.

Additional information is available from the Web site: <http://www.cs.berkeley.edu/~eanders>.

A Scalable Appliance Model for Email
Strata Rose, Synopsis

As the size of your network and the number of users grow, the traditional model of a single mail server breaks down. A new paradigm must be created to provide reliability, scalability, and maintainability.

In this presentation, Strata described a means of assigning specific roles to distributed computers: spooling incoming mail, queuing outgoing mail, delivering mail, filtering mail, routing mail, expanding aliases, and providing a spooling area for mail that will be downloaded via a POP client. These computers are then dedicated to those services – thus the term “appliance.”

Because these services can be disassociated, treating the computers that provide them as “appliances” can greatly increase the scalability of your mail system configuration.
A Perl Library to Manage Novell Directory Services
Mike Richichi, Drew University

Traditionally, Novell has provided only a GUI interface to their directory services. This is wonderful when you’re adding small amounts of information, but it quickly becomes nightmarish when you are faced with the task of adding hundreds of new entries.

In order to provide the command line interface to which UNIX users have become accustomed, Mike has developed a library of Perl functions. This allows traditional UNIX-like scripting with iteration to perform multiple transactions while reading relevant input from a data file. This is a wonderful alternative to pointing and clicking for hours.

JMAPI - A Java Management API
Neil Groundwater, Sun Microsystems

Although it was presented by an employee of Sun Microsystems, this is not a Sun product. The JMAPI is a set of extensible objects and methods for the development of seamless system, network, and service management solutions for heterogeneous networks. Instead of one vendor creating the ultimate management tool, JMAPI places the management onus squarely on the shoulders of the vendors.

Current partners in this venture include 3Com, Bay Networks, and Cisco. For additional information, examine the Web site at: <http://java.sun.com/products/JavaManagement>.

Closing Session: System Administration – The Last Ten Years and The Next
Rob Kolstad, Berkeley Software Designs, Inc.

Summary by Carolyn M. Hennings
<cmh@psa.pencom.com>

Certainly, the reason conference attendees stayed for the final session of LISA ’96 was Rob Kolstad’s presentation. Rob’s animated speaking style engaged listeners in an overview of factors shaping the future of system administration. Rob illustrated how marketing and the opinions of individuals with little or no understanding of UNIX and technology have a major influence over our industry. Some examples of public misperception included:

- UNIX is difficult to use and learn
- there is an abundance of bandwidth on the Internet
- Linux is not UNIX
- the Web and the Internet are really the same thing

Recent legislation and court cases have also opened our eyes to how technology is misunderstood by the mainstream public. The Communications Decency Act of 1995 was presented as an example of how public perceptions can impact the direction of government action.

BOFS

Summaries by Carolyn M. Hennings
<cmh@psa.pencom.com>

Women in System Administration (WISA at LISA)
Moderator: Idajean M. Fisher, PSA-Pencom Systems Administration

As a first time attendee at a USENIX Conference, I was pleasantly surprised at the involvement and visibility of women in the field of system administration. To start off the discussion, Idajean asked the group if there are any particular things women look for when considering employment opportunities. An immediate gut level response was “one of ‘those’ men,” indicating a strong aversion to organizations where prejudicial employees are allowed to continue. It was a refreshing and open response that was only possible in an informal setting. Additional items that were mentioned included the composition of the team, work structure, mentoring projects, number of women in management, women’s initiatives, and women’s programs. However, the way this information is shared with a candidate determines the interpretation, thus shaping an opinion about the company.

Various other topics were discussed, such as relationships with former peers after a promotion, feeling pressure to join in social activities where you are not comfortable, and ways to help young high school and college women understand this career opportunity.

BOF: Education and Systems Administration
Moderator: Bruce Alan Wynn, PSA-Pencom Systems Administration

This two-hour BoF covered a wide variety of topics related to education and systems administration. Topics included the availability and usefulness of vendor training, university programs in systems administration, viable presentation media, curriculum and course development guidelines, metrics, and using educational goals as part of career planning.

The value and usability of vendor training was discussed. The group was in general agreement that most training from vendors does not provide the maximum value for its cost. This was followed by discussions of how sysadmins learn and elements of good Web-based training.
In light of the growing interest in working with universities to define a formal program for sysadmins, the question “What should be included in a college curriculum?” was posed. Here are some of the suggestions:

- Technical writing, hands-on and focused on real-life projects, including documentation, proposals, and lab reports
- Business and management skills, accounting, negotiations, market research, teamwork, and presentation skills
- Technical skills such as good system design, scalability, and algorithms
- Some computer science classes like operating systems
- Technical writing, hands-on and focused on real-life applications
- Problem solving, philosophy, and logic
- English and communication skills
- Experience supporting computer labs
- Learning how to learn and how to teach others
- Psychology
- Listening to users, determining what is needed, putting the needs in a vendor framework, and implementing the solution

Attendees and other interested parties are encouraged to join the majordomo mail group <sage-edu@usenix.org>. Instructions are available at: <http://www.usenix.org/sage/mail/majordomo-user.html>

**PERLS**
Jeff Okamoto with Larry Wall, Tom Christiansen, Randal Schwartz, Jon Orwant

Summary by Steven P. Potter <spp@psa.pencom.com>

Tuesday night’s Perl5 BOF was a good opportunity for interested parties to find out what is happening (and has happened) in the world of Perl. Perl is rapidly becoming a “real” language, with a solid user backing, a shelf full of books, and now its own journal and institute.

The BOF was originally scheduled for a small room, but when ten minutes before the BOF started the room was filled (before any of the main participants even arrived) and people were milling around the balcony outside, a larger room was quickly found. Approximately 125 to 150 people were in attendance, and the hot topics of the night were database interfaces, Oracle and Sybase interfaces, *The Perl Journal*, the Perl Institute, the newly released “Programming Perl,” and database interfaces. Another really hot topic was database interfaces.

Copies of *The Perl Journal* were handed out to as many people as possible, as well as information about the Perl Institute. Also handed out were camels for the first annual “Perl Camel Races.” More information about the journal can be obtained from <http://orwant.www.media.mit.edu/the_perl_journal/>.

More information about the institute, and what you can do to save the world, can be obtained from <http://www.perl.org/>.

**SAGE Local Groups**

Summary by Rob Brian Jensen <robjen@psa.pencom.com>

The SAGE Local Groups BOF was moderated by Kim Trudel of MIT, a member of the SAGE board, and Rob Jensen of Pencom Systems Administration. Kim is very active in growing local SAGE groups and is also interested in the sustainability of existing groups, such as Back Bay Lisa, once they have been running for a while. Rob is an active member of dc.sage, and is interested in starting a SAGE local group in Baltimore.

We started out with a fairly solid list of topics to be covered and enough slack to allow for plenty of discussion. Of course, a BOF never runs that way: soon the room had control of the discussion, and the moderators were just there for the ride. The intended topic list was:

- What can a local SAGE group do for you as a sysadmin?
- What does it take to start a SAGE local group?
- What support does SAGE provide to the local groups?
- Where are there active SAGE local groups now?
- Where will new groups be forming soon?
- What do we have to do to get our own local group going?
- Now that we have a local group, how do we keep it alive and active?

The BOF was well attended by representatives from most of the existing SAGE locals, including many of the founding members of Back Bay Lisa, $GROUPNAME, NC*SA, NYS, and SSG, to name a few. There was also a healthy crowd of sysadmins from Chicago, interested in forming a local group. Christine McCormick is going to take a stab at starting a Chicago group in the near future. In theory, a group could flourish in any metropolitan area with more than a dozen system administrators interested in networking. In practice, it helps to have at least four highly motivated people to get a local group started and keep it running. The following regions with existing local groups (or groups that are currently forming) were represented: Baltimore, Boston, Chicago, Minneapolis/St. Paul, New Jersey, New York City, Research Triangle NC, Seattle, and Washington, DC. There were also interested people from the following areas who wanted to have a SAGE local in their area: Berkeley, Chattanooga, Central Iowa, Dallas/ Ft. Worth, Denver, Detroit, Houston, Los Angeles, Pittsburgh, Phoenix, Ottawa, Toronto, and London, UK. We collected the names and email addresses of everyone attending, and we plan to start a mailing list in the near future for people interested in starting SAGE locals.
I think the BOF was reasonably successful. We put some more information into the hands of those who would have their own local group. Kim received some good solid feedback from people about the type of support that the SAGE board could provide to the local groups. We introduced a few local people who are interested in getting a group started. We hope there will be several more existing groups at LISA '97.

If you are interested in helping to start a SAGE local group in any part of the world, including those listed above, here are some pointers to more information about getting started:

Start with the SAGE page listing the existing locals: <http://www.usenix.org/sage/locals/sage-localgroups.html>.

Join the sage-locals mailing list: <sage-locals-request@usenix.org>.

Or, contact Kim or myself for more details from the BOF, including a checklist of guidelines to get started (still in draft form): <kim@usenix.org>, <robjen@pencom.com>

Confessions of a USENIX Volunteer

Commentary by Idajean M. Fisher
<ides@psa.pencom.com>

I have a confession to make: I am a USENIX volunteer. I don't know if there is a 12-step program for this, but if there is, I don't think that I really need to know about it. You see, I really like being a USENIX volunteer.

At LISA this year I spent much of my time as a "Booth Babe." I met lots of nice people who were also inhabitants of the Booth. The Booth People are little-known inhabitants of the USENIX and SAGE communities. They congregate in small wooden cages. Passersby typically bring them offerings of cash and small plastic religious objects. The Booth People in turn provide their followers with clothing, pins, and SHARED SECRETS. I am not sure what is really at the root of all this, but by the nature of the reverence for the Booth People and the proliferation of followers who come to bring them tribute lead me to believe it must be based in religious ritual.

The first day that the sacred Booth was set up several neophyte Booth People (myself included) were placed in the folding chamber where various objects were prepared for dispersal to the followers at the Booth itself. There was much talk about exactly how this should be done. The faithful came by on occasion to give instruction to the neophytes regarding appropriate observance of the folding ritual. There appeared to be two different sects: the three-way fold (followers of Toni) and the four-way fold (followers of Evi).

Over the course of the week, the neophyte Booth People each got their turn at serving the needs of the followers and protecting the Booth. There was much exchanging of tribute for clothing as well as much cracking wise. We were labeled by brightly colored ribbons to alert people from a great distance that we were indeed inhabitants of the Booth.

All the Booth People at LISA also had to pass the Test of Many Puzzles. The first part of the test was very straightforward. Neophytes were expected to assemble the puzzles into a series of small cubes. Later in the ritual the Booth People began stacking the cubes and building elaborate things with them. In the final phases of the Test of Many Puzzles, the Booth People began throwing the puzzles at passersby and finally wearing the puzzles as adornment.

A wily breed these USENIX Booth People.

The Terminal Room: A Volunteer's View

by David J. Bianchi
<djb@psa.pencom.com>

The terminal room at LISA continues to be a success. Gretchen Phillips did a great job coordinating equipment, communications and volunteers; and the conference attendees made good use of the room.

There were a dozen Sun workstations (thanks to Sun Microsystems), as many HDS X-terminals (thanks to HDS Inc.), and more than a dozen PCs. There were also connections for laptops and a handful of dialout modems. In addition, you could dial the terminal room from your hotel room and gain access to the Internet. Secure connections via SSH and Kerberos were also available from the terminal room.

FaceSaver was back, and judging by the number of times that they ran out of labels and luggage tag covers, it was very popular. The NeXT computer had problems periodically. There were times when it had to be rebooted, and no one in the room had the login and password. I would like to see a donation of a new workstation (with an attached camera) from some well-known computer manufacturer. Someone reading this must have connections.
Notes on the LISA '96 Advanced Topics in System Administration Workshop
by Tina Darmohray, John Schimmel, and John Sellens
<tdarmohray@iwicom>
<jes@sgi.com>
<jmsellens@uwaterloo.ca>

It's 9:00 am and we've already turned the tables on this workshop (or at least the tables in the meeting room, so that they'd face each other). The continental breakfast is rapidly disappearing. But all is not lost – John Schimmel made sure that there was plenty of Diet Coke available for the break.

The workshop was attended by 36 people, with roughly equal representation from universities and commercial organizations, with a handful of government types thrown in for good measure. Most attendees labeled themselves as managers, and there were no students in the group.

The attendees:

This is the second Advanced Topics Workshop held at LISA – the first was last year at LISA 95 in Monterey. The workshop was organized by John Schimmel of SGI with the goal of providing a forum for a focused discussion on the most important current issues in system administration. Participants were expected to submit at least one topic for discussion, and the workshop was designed to be wide-ranging and free-wheeling.

The Advanced Topics Workshop was created because back in the olden days, LISA was 100 people in a room throwing stones at each other. Now it's over 1,600, and one of the questions is: how do you go to a conference that size and actually learn anything outside the organized sessions? So, our hand-picked, select group (i.e., anybody that sent John mail expressing interest) got together to see what we could learn.

9:30 am: John says "Let's start." By some quirk of fate, the room ended up with almost all the university people on one side and almost all the commercial people on the other, with a smattering of research and government people mixed in. A short review of operating system use revealed a predominance of SunOS and Solaris, with almost everything else represented (AIX, Sequent, DEC Ultrix and Digital UNIX, BSD variants, Linux, Plan 9 [1 person], NT, VMS, MacOS, HP/UX, IR1X, Windows).

John wrote down a list of every topic that had been suggested for discussion. Helen Harrison suggested that we vote for three (or so) topics to discuss today. John said, “You're a manager, right?”

Tina Darmohray tried to suggest to John how to run the vote on the topics, and she learned a valuable lesson: Tina was now running the vote. Very efficiently. Most topics turned out to get either exactly 1, 5, 10, or 20 votes, until Tina started getting hassled about it, and we had votes of 12, 7, and 4. Windows NT and its associated implications was the hot topic at over 25 votes. Software distribution, networking, and other topics followed further down in the voting. We decided to try to tackle some of the other topics first, before diving into Windows NT issues, and so started on the topic of software distribution.

Part 1 – Software Distribution

Andrew Hume started the discussion by asking “What about rdist to Windows 95?”

Paul Evans related that there had been two or three paper submissions to this year's conference with the general approach of using dual boot PCs with Linux and Windows installed, and rebooting into Linux every night, using rdist and other commands to (re-)configure the Windows side of things, and then rebooting into Windows. General laughter ensued, with the feeling that this was not a very pleasant solution.

Laura de Leon observed, “Maybe we should have started with NT.” The discussion quickly centered on NT administration and integration in a large network (and NT and PC issues continued to surface regularly throughout the morning). Many attendees reported that their organizations are getting NT-based machines rapidly. The main reason behind their acquisition is to run popular applications (like Word and Excel) and to provide “easy” file and print service for PC users. The problem that NT servers (and PCs themselves) create for system administrators is that it is hard to provide large-scale support for them because their typical administration paradigm assumes that they are administered from the console, rather than remotely or through some sort of automated scripting language. We agreed that a command line interface to hook into NT and PCs to manage the machines remotely would be a dramatic improvement over the current state of affairs, and that it would be even better if...
there were a well-defined API for all the administrative functions.

People had come up with a large number of utilities to handle the PC problem, but none of these was particularly earth shattering. Probably the most useful tool discussed was Samba. The biggest problem seemed to be Lotus Notes.

Do GUI interfaces to UNIX (or NT) system administration get in the way of larger scale administration? Pat Wilson suggested that "surely there are command line alternatives for every GUI." Andrew Hume pointed out, as a counterexample, that the interface to NCR's RAID setup is through the GUI only. There was discussion of a few examples of administrative tasks that are more easily done through a GUI. Some systems have a command line alternative, but it isn't always obvious to the casual (or not so casual) administrator.

The question was raised: why don't vendors make the command line alternative as easy to find as the GUI? Some of the suggested/postulated reasons were:

- Most sites are small shops, and their administrators appreciate the GUI interface
- Sites (and administrators) with large installations are the exception, rather than the rule
- Marketing is easier with a GUI interface
- A GUI interface makes telephone support easier and often provides the useful alternative of grabbing the other person's interface over the network.

Paul Evans threw out "a fundamental philosophical issue" in the UNIX/Windows discussion: from the PC/Windows point of view, the fundamental unit is the isolated machine on a person's desk, and much of the administration of the machine is done by walking up to the person's desk or doing it one at a time over the network. Paul Anderson pointed out that automatically administering a network of 300 or 400 UNIX workstations is doable, and adding one more new machine is not a linear increase in the required effort. But in the PC world, the effort required is much more often linear.

The question arose: how can we protect users from themselves? This is, of course, the age-old dichotomy of centralized vs. distributed administration.

Why don't OS vendors provide a "large installation administration kit" as a package that could be installed on your machines? Elizabeth Zwicky mentioned SGI's "inst", which will, in the next version, have a programmatic interface, so that it will be (theoretically) much easier to automate.

Andrew Hume asked for opinions on the use of encrypted data distribution within an organization. His example was AT&T - can AT&T really trust the 200,000 people inside the AT&T firewall with "confidential" and "need to know" information available on the internal network?

The discussion turned to standards and formal methods. What about the "sysman" standards effort?

Mark Verber asked: is the issue data distribution or configuration management? Do formal specifications and theoretical methods work? Can you apply theory if you don't deal with it in practice? Andrew Hume believes that formal specifications will work in system administration and that the current action is in Europe.

Greg Rose asked: when was the last time you moved your car's clutch pedal to the middle? Is reconfiguring your machine/environment necessary or even a good idea? Xev Gittler pointed out that if machines were as standardized as the automobile interface is, we wouldn't have nearly as many problems. Paul Evans suggested that it's a function of how mature the system or industry is and that there weren't a lot of standards in the early days of the automobile. Brent Chapman wondered if turning on the ignition in your car and setting up your computer for your needs are comparable tasks. Is the latter an inherently more complicated task? Strata Rose pointed out that the utility of a car is already present in the car, but the utility of a computer is in how much you customize/modify it to suit your particular needs.

Does pressing your vendor's sales rep cause change to happen? Paul Evans observed that organizations behave in a way that makes sense internally. Sun never would have had Solaris if it was just driven by marketing or customer desires. The conclusion: pressing your sales rep may have no effect at all.

And once again we veered back into the PC arena, as indicated by Paul Evans' observation that people want things for a combination of rational and irrational reasons, stated and unstated needs. The PC "culture" thing means that there is a cultural backlash to any attempt at central control of "my desktop." The user's perfectly reasonable assumption is "I do it at home, why can't I do it here?" - which leads to the obvious conclusion that something that is okay to do when you have one system is not necessarily a good thing to do when you have 10,000 spread across the globe.

What about charging for support? Will this help to cause people to toe the company line? This is quite an ironic idea, given that we are the people who set out to topple the tyrannical regime of MIS with UNIX, and now we are the tyrannical regime. Can we influence people in the desired direction by offering good, standard support for free, but charging money to fix someone's "customized" installation? Andrew Hume observed that there are two types of users - one is happy with the standard setup, and the other needs the "latest and greatest" all the time.
Paul Evans: the cost of truly supporting people is far higher than typical perceptions suggest. It’s an economic decision. We know how to do very good system administration. What we don’t know is how to do whatever is the most “cost-effective” for us.

One approach is to build a “baseline” distribution that is good for 80% of the users, and then make it easy to add the few extra things that people need (or want), in a standard way. Elizabeth Zwicky mentioned that SGI Europe has 93% compliance to their baseline distribution. They used convenience (it’s really easy to use the baseline distribution), and bribery – extra disk space lured in a few people, but free T-Shirts lured in many more people. And posting a daily compliance statistics list helps people want to comply – an example was given of one office that increased its compliance simply so that it would have higher compliance than its rival office in another country.

The topic of software distribution traversed all the extremes. On UNIX the rdist/track tools are still the most widely used methods for distributing software and configuration information. There are a number of commercial products now on the market to act as wrappers around these protocols, but none of these seemed to cleanly fill the needs for either software distribution or configuration management. The number of large sites represented that used a commercial tool to handle configuration management was still very small. Most sites are still creating homegrown tool sets to handle their unique circumstances.

Attempting to handle the filesystem layout in a large environment is still a significant problem. It has been long recognized that maintaining a single software repository, or at least a standard layout, is beneficial for large sites, but no one has yet worked out a clean way to do this. Many papers have been delivered at past LISA conferences on ways to control the layout of software to be best managed in a large site. This is still not a solved problem in the UNIX arena. In the Windows-based site, this is pure chaos.

The day’s first session closed with some observations. Strata Rose: Microsoft is just there – it’s like a glacier. Xev Gittler: It’s not a glacier, it’s an avalanche.

Part 2 – After the morning break

John Schimmel wrote down a list of things we had talked about:
- File distribution – UNIX has known solutions, but Windows/NT solutions are not nearly as obvious
- NT/Windows administration
- Administration models
- User customization/modification
- Baseline software installation
- User mentality (UNIX vs. PC)

Is it part of the PC problem that people are expected to differentiate between the consumer electronics that they have at home and the centrally administered and supported business tool that they have at work? Is that a reasonable expectation to have?

The conclusion at this point was that the biggest issue or concern that we have now is PC support, integration, and Windows/NT administration and control. There was some concern that management was being unduly swayed by Microsoft’s marketing power and the idea that Microsoft is the answer to all questions.

There followed a discussion of the UNIX development environment as compared to the PC development environment. It was felt that the best characterization was that UNIX provided a collection of tools that work together, while the emphasis in the PC world was on the one tool that could do it all. The conclusion was that each approach has its place, but the “one tool” approach was more likely to limit the things you can do.

Just before we broke for lunch, we agreed (unanimously!) that we wouldn’t say anything at all about PCs or Windows NT in the afternoon session.

Part 3 – After lunch: Networking

This discussion started as an informal survey of who is using what and continued on into a few questions and hints back and forth.

- Everyone is 10baseT Ethernet; some old “legacy” thin and banana cable is still in use
- 100baseT is gaining ground – some have 100Mb to the desktop
- Some FDDI, some ATM, a little HIPPI, no token ring
- Boeing has OC48 ATM between campuses
- About half have ISDN – most are pure data rather than data/voice
- Leased lines 56K, T1, frame relay
- Andrew Hume has “remote extension” ISDN to his house (50 miles), which works well
- There is some use of wireless networks, both within and between buildings
- A little Novell
- No IPV6

Q. If you’re 100Mb to the desk, what are you to the server?
A. This is not always a problem, since the desktop is sometimes running video or similar applications directly to another desktop, bypassing the server.

Q. Trends?
A. 100VG is dead
A. FDDI appears to have a limited future, with little expansion
A. Any organization that’s growing is installing more fiber
A. Switched Ethernet is expected to be very useful and more widely deployed in the future.
A. There’s an expectation that wiring will be either category 5 UTP or fiber.

Q. What do you see three years out? Is anyone envisioning anything other than 10/100 baseT to the desktop?
A. Not really . . .

Q. Routing protocols?
A. Some RIP, almost no RIP2, some OSPF, IGRP, static routes are remarkably common.

Q. Network monitoring?
A. SNMP is used for monitoring – there is limited use of SNMP for configuration since telnet is usually more convenient.
A. A little OpenView, a little Spectrum, tkined/scotty.
A. Freeware tools are about as common as commercial tools.
A. Capacity monitoring is used more as a diagnostic tool, rather than an ongoing planning tool.

A few points:
There was a discussion of effective capacity in leased lines, frame relay and so on.
A fair number of people are doing central syslog collecting – primarily ignored until there’s a problem – reactive rather than proactive.
Lots of people (almost everyone) are using console concentrators in some form, for central access to and logging of console activity.
Gigabit networking (1000baseT) is coming and is expected to be available in non-standard implementations by the end of the year.

There seemed to be a general consensus that networking is typically “solved”, i.e., we know how to deal with it.

Security

Computer and network security is a significant concern for most organizations. As we did for networking, we fell into the informal survey/short question and answer mode for our discussion of security-related issues.

Q. How many people are using ssh?
A. Over half

Q. PGP?
A. Quite popular

Q. Anyone using stel?
A. Pretty much no one

Q. Kerberos?
A. About the same as last year

Q. DCE?
A. Far fewer sites had plans for using DCE this year than last.

Q. Anyone using a commercial firewall right out of the box?
A. A few people are

Q. TIS Firewall Toolkit?
A. The majority

Q. No firewall at all?
A. About 1/3 of the sites

Q. Using socks to get through the firewall?
A. A few

Q. Doing Java applet filtering?
A. A few, but it’s a popular concept

Q. Doing virus scanning on the firewall?
A. One or two

Q. Using one time password tokens?
A. Roughly 1/3 of the sites

Q. Running a key signing service for your users?
A. A few (Boeing, SGI, USENIX . . .)

Q. What do we think about Java?
A. We’re leery . . .
A. Elizabeth Zwicky: “I think sendmail will be safe before Java will.”

Simon Fraser University just signed a site license for Timestep, which is secure, encrypted, virtual private networking (SVNP).

Many sites are still looking at Kerberos implementations. Some attendees lamented that vendors could help us out by offering kerberized versions of their applications.

The current state of the art in interdomain system administration was discussed (e.g., filesystem sharing, account creation, etc.). There was some interest in SSL. Neutral zones (“shared project networks”) are in use by a number of organizations to join private networks.

Part 4, after the afternoon break: Human-to-Human Communications

We started this topic with a discussion of wireless communications for support staff. Synopsys uses radios to help the 70 or so sysadmins keep in touch:

- They can be dangerous things, since it’s a very public mechanism.
- An interesting management tool (since you can hear everything).
- Radio is “out of band” and “multi-party.”
- Very powerful for coordinating multiple people at the same time, e.g., when moving to a new building, or recovering from a major outage.
- Allows all the admins to go offsite to nearby restaurants and still be in easy 2-way contact.
- Radios help with team building, because there's more sharing of activity information (e.g., "I'll check on that since I'm here anyway.").

There was general agreement that radios can be a very useful tool.

Problem-tracking software is in much wider use today than it has been in years back, and now many of the larger sites are using commercial packages instead of the homegrown solutions that were presented at earlier LISA conferences.

How can we deal effectively with new "short" tasks? It’s typically two things – find the time to do the short task, and make it obvious that you have had to not do something else. Strata Rose suggests that “TTM lists” (task, time, money) are a good tool for dividing a large task up into subtasks that people can understand. Assign time and monetary costs to each subtask, and then it’s much easier to make the decision of what to stop doing in order to accomplish this task.

For many of us, email is one of the primary means of communication. It is important to remember that email is an imperfect medium. It’s easier (and faster and better) to solve conflicts in person rather than through email. Email lacks facial impressions, tone of voice, stance, and is very fast, which makes it much harder to “edit yourself.” Current version humans tend to be much less “ept” at writing than speaking.

Remy Evard pointed out that being able to write effective email is a critical skill for system administrators. There is a tendency for technical people to think that everyone else is like themselves and that people who disagree with you are either evil or stupid. It is likely that, when it comes to email, we (as system administrators) are the weirdos, because we tend to deal with larger amounts of email and have more experience with it. Our attitudes toward email, and the way we use it, are likely to be atypical – it’s important to remember that not everyone looks at email the same way that we do.

The discussion of email slid into a mention of voice mail, and the question was raised: who’s having trouble dealing with the combination of email and voice mail? There was general agreement that we find email more effective and less disruptive/annoying than voice mail. Bill LeFebvre opined that "Voice mail is the fax of the 90's."

How do you communicate with your users? Brent Chapman makes it a habit to write explanatory documents and put them on the Web – Brent does his project plans as HTML documents. It helps if you've got strong Web indexing tools on your server. Laura de Leon said that at her site the users like the Web pages, but they love the bimonthly paper newsletter (which also ends up on the web). At SGI, everything is Web-based. "Silicon Junction" is the SGI internal default Web site and is essentially a daily newspaper.

Are Web pages good for announcements and discussions? The consensus was that they are just about as good as the alternatives (mailing lists, newsgroups). Sometimes it is appropriate to send an email pointer to a URL with a two- or three-line description. Mailing lists used for announcements need to be very low volume.

One person suggested that important announcements should just "pop up on your screen." This was not a popular option – it was felt that this would be too annoying. In addition, there would be problems with choosing which screen to pop it up on, and there is the problem of what happens to those people who aren’t signed on when the announcement is posted.

Paper notices are appropriate too – server down notices get posted on the washroom doors at Boeing, and Synopsys posts notifications of the quarterly downtime on every external door.

The important point is: determine what the appropriate method is for your site, document it, and use it. Whatever channel you choose, it must be dedicated to important announcements for it to be effective.

**Publications**

As SAGE publications editor, Tina Darmohray is always looking for printed material. She wants/expects ;login: articles from workshop attendees. Can we put together a document for vendors demonstrating the need for configuration APIs? SAGE would like to have a legal issues document, and is interested in hearing about relevant issues. Two of the questions to be addressed are what you actually do and what you are legally obligated to do.

Legal issues were discussed very briefly. One common thread was that many sites have a policy NOT to back up email, specifically so that they cannot recover it if asked to do so.

**Backups**

It was not a real surprise that backups are still a big issue for workshop attendees. The average site was dealing with hundreds of gigabytes of backed-up data. The most popular backup technology appears to be DLTs this year, but several people are actually beginning to use the higher speed DST tapes from Ampex to deal with extreme data stores. For example, Andrew Hume reported using 4 DST drives to back up his organization’s 960GB SGI server. The rapid growth in
the size of disk technologies and the incredible demand for online storage continue unrestrained.

Various network backup systems are in use, with the most common being Legato Networker and IBM’s ADSL.

Other related questions/problems are the use of archive servers, network bandwidth, hierarchical storage management, and the use of onsite and offsite disk mirroring.

**Quality, Service and Metrics**

Remy Evard asked: how can you tell that the system administration in your organization has improved over time?

There was a discussion of metrics for evaluating system administration performance. Some of the problems and issues raised were:

- Measuring the number of support requests is one obvious metric, but it’s hard to measure problem reports as distinct from requests for enhancement.
- If one of the goals of system administration is to make it possible for the users to be more effective, how can you measure whether or not the users are more effective in doing their jobs? This may be especially difficult at research or educational sites.
- Can you measure user satisfaction? Are surveys an effective measurement tool? Or would they be skewed by current events or problems outside the system administrator’s control?
- If you have machines submitting complaints automatically, are your machine-generated complaints going down?
- How permanently do you solve a problem? Are you solving the same problems over and over again? Is that a good metric?
- How much of what you’re doing today were you doing last year? How much have you automated, or passed on to lower-paid people?
- If you can spend your time planning the future, is your site well-run?

If you base the evaluation of system administration effectiveness on user satisfaction, you need to survey the users. Does it matter if the users are rational and know what’s best for them? It’s the old question of what your goal is – to have a well-run system, or to make your users happy? Are those necessarily conflicting goals? Paul Evans pointed out that trying to make your users happy is sometimes the road to ruin.

Should our goal be to keep them at the same level of happiness? Is user satisfaction/happiness beyond the system administrator’s control? If the users want more file space, and there’s no money for more disk, that’s not (usually) the sysadmin’s fault, or a problem that he/she can solve. Or is this a “site” problem – the site is not being well-run (for whatever reason) because the “appropriate” resources are not there?

Strata Rose asked: do we consider users to be competent people in their own right? Or do we treat them as children who don’t know what they need to get their job done? It is important to treat users as mature professionals.

Do the goals, values, and satisfaction levels of a site differ between (for example) commercial sites, universities, and ISPs?

A measurement problem arose: does the system administrator get blamed for vendor problems that are beyond his/her control?

Elizabeth Zwicky suggested that a good metric might be: do the users think that things are getting better? Her site offered users a virtual $100 and asked what part of IS would the user prefer to spend it on. This kind of survey can easily be biased by the interaction that users have with particular IS groups.

A question was raised: how many sites have system administration and networking as disjoint groups? Quite a few hands went up. Do the disjoint groups typically work well together? (apparently not.)

And finally, there was some discussion of chargeback for system administration services. Should system administration be a profit center, or just another component of a company’s infrastructure?

**Conclusions**

It’s 4:30 pm and we’re done.

The obvious questions are: Do we have any useful conclusions? Have we accomplished anything? Or is the benefit of this workshop the sharing of ideas and experiences and meeting people with similar or different problems? Can we change the world? Can we fix anything substantial as a result of this exercise? Perhaps not.

Strata Rose mentioned that perhaps one of the primary accomplishments of this workshop is that of “morale” – if my peers are also facing the same problems that I am, maybe I’m not incompetent after all. We networked today – people got some good ideas about what other people are doing, and hopefully have a better understanding of which way the solutions might lie.

A final consideration is, of course, planning for next year. Is there some different structure which can really accomplish something in this kind of forum? Or is the sharing of knowledge, problems and experiences the most important benefit of this workshop? Something to think about.
Horror Story Contest

This year at LISA, we added a little levity to the festivities by proposing a Sysadmin "Horror Story" contest. "Don't spare the gory details," we implored. And "fabulous prizes" were promised to the victorious.

The stories poured in on bright orange forms and were dutifully read by a board of judges whose primary reason for being selected was that they were not easily amused. (No easy winners for us!)

The results are in, and you can judge the First and Second Prizes for yourself:

- First Prize $125: “The Last Game of Cricket” by Andy Haxby, Systems Gynecologist, Shell International Exploration and Production Research Technical Services, Rijswijk, Netherlands
- Second Prize $75: “Pentagon” by Greg Maples, ClariNet Communications
- Third Prize $35: “4mm Pickup” by Jeffrey A. Gilman, UB NB Networks, Inc.
- Best “Out of Band” $75: “Smart House” by Todd Williams, MacNeal-Schwendler Corp.
- Honorable Mention: Mike Fisk, Los Alamos National Laboratory; Allen Peckham, Texas Instruments; Dan Stephans, SBCW; Doug Hughes, Auburn University.

The Last Game of Cricket

by Andy Haxby
<a.haxby@siiep.shell.com>

Back in the Dark Ages of IBM 3090’s and monster VAX clusters, I worked for a Nuclear Power company in England. We were a research lab and the main data center for the whole company. The machine room was huge – great halls filled with blue and grey cabinets, disk farms, rows of tape drives which looked like windmills on a distant hill. Pride of place in the middle of all the dull boxes was given to a Cray-2 which bubbled its freon through perspex pillars back-lit with colored fluorescent light to impress the managers and justify the exorbitant price.

To operate this vast array of machinery an army of operators were employed. The main console area was named “The Bridge” since it resembled the Starship Enterprise with rows of green screens and keyboards, and hordes of people running about pressing buttons and speaking in a strange language with words like “IPL,” “VTAM” and “TSO.” These strange words were often followed by “is down.”

Anyway, in the quiet evenings after “prime shift,” the teams of operators had very little to do other than watch cryptic messages scroll up a screen and wait to see if they could find a red one. To while away the hours people used to do things like driving their car into the machine room. The suspended floor could be rm’ed to make a temporary “pit” while you made a few repairs. One day creativity and the need for physical exercise after so many hours of sitting at consoles got the better of them, and they decided to pass the time enjoying the traditional English game of Cricket.

The halls in the machine room were a suitable size for a miniature cricket pitch, and the side of some of the slimmer machinery made a good wicket. The tube from a roll of graph plotting paper was an adequate bat and unwanted printout rolled into a ball and wrapped in Duck Tape was a reasonable substitute for a cricket ball, albeit rather lighter. There were long gaps between the machinery through which you could run, and on countless evenings the machine rooms echoed to the sound of laughter and, if not the crack of leather on willow, at least the dull thud of Duck Tape on cardboard. This was a popular pastime, with most of the operators involved, and great rivalry between the different ops teams. While managers and day staff were unaware of the nocturnal activities, managers actually commented that they had noticed a greater team spirit amongst Ops of late.

For those of you not familiar with the rules of cricket, basically someone throws the ball and the guy with the bat hits it. He then runs up and down a short pitch while the fielders try to retrieve the ball and throw it at the wicket while the batsman is in mid run. The object of one side is to get as many runs as possible, the object of the other side is to get all of you opponents “out” by hitting the wickets (or in this case a 3090 or similar) with the ball. When you’ve done this you all shake hands and drink tea and then swap sides.

The details of scoring are as convoluted as writing a C compiler in awk. The major win when playing cricket is for the batsman to hit the ball so hard that it reaches the boundary of the pitch, in which case you don’t have to run anywhere and four runs are awarded. This is called a “four.” When this happens everyone cheers and shouts “four.” In a machine room filled with cabinets it is extremely difficult to get a “four” because all the machines get in the way. A variation on the “four” is the “six.” A six is when the ball reaches the perimeter without touching the floor, and this is considered a major major win. This is easier in this particular variety of cricket due to the relatively light weight of the ball.

Obviously the ratio of the weight of the bat to the weight of the ball influences how much energy you can transfer from the bat to the ball. This led the Ops to experiment with different bats, made from cardboard tubes filled with a variety of materials to add strength and weight. All of this was to chase the highly desirable but relatively elusive “six.” As the
design of the bats improved, the frequency of sixes increased.

One evening a particularly skillful batsman took a long swing at an easy ball. The ball sailed effortlessly over the top of a variety of mainframes, power supplies, HSM machines etc., and everyone got ready to shout “six.” However the batsman’s aim was remarkable, and the ball headed directly, almost as if attracted by magnetism, towards a large red button on the wall labelled “Emergency Powerdown.” The ball struck the button with a dull thud, almost inaudible above the hum of the great machines. This was instantly followed by relays clicking and then a sickly silence. The entire data center had powered down . . . . This was the last game of cricket.

Pentagon
by Greg Maples
greg@clari.net

It was 1982, and I was working in the Pentagon. I had just completed the world’s first port of UNIX V7 to a PDP 11/25 flown in from DEC. The serial number was 0007. The 11/25 had shipped with an exciting new technology, a small removable 5 MB (wow!) removable disk pack in addition to the internal 20 MB (wow!!) disk. Imagine, no front panel keying of the bootstrap loader! At that time, we had a 24 by 7 support policy from DEC with a mandatory four hour escalation policy for each level of support. On the third day of operation, the mainboard failed. When the field service engineer arrived, he had “the spare” – not “a spare,” but “the spare” – mainboard. The mainboard was connected to memory and peripherals as well as power by two ribbon cables. The cables were carefully labeled 1 and 2, but not “up” and “down,” nor were the cables keyed. So, of course, when the FSE put the replacement board in, he attached the cables upside down, and the board smoked. Not failed, caught fire. Not only did he blow the only spare, but now the memory was dead. Under the escalation procedure, DEC had to get us another board in four hours. So, somewhere in Maynard Mass, a field SUC manager hopped on a plane with two new cards. He got a taxi straight from the airport and just made the four hour deadline. He arrived, slammed the new boards into the machine, attached the ribbon cables, took a deep breath, and fired it up. Yes, he got it wrong and smoked both boards. A pile of dead boards was forming. The next tier of support was now within engineering at DEC. An engineer was found, parts gotten, and he jumped on a plane. When he got in, he analyzed the problem and announced: “I see the problem, the cables aren’t keyed and you’re putting them in upside down.” He replaced the now melted ribbon cables, put in new boards, etc. Result? You guessed it, he smoked another set. At this point, total frustration was beginning to give way to comedy.

End of the story: The principal design engineer at DEC responsible for the 11/25 was put on a plane with a boxload of parts. He arrived and entered the huddle of gathered DEC employees. We were up to five or six by then, I happened to notice that all the connectors were now labeled with tiny little red dots on one side, as were the cables. We had been down 20 hours when we got a good boot.

Postscript – The next revision of the hardware service manual devoted ten (10!) pages to orienting ribbon cables on the 11/25.
BOOK REVIEWS

The Bookworm
by Peter H. Salus
<peter@pedant.com>

In the late summer, the representative of a major publishing company informed me that C and C++ were "passe," and that "only Java" was "relevant now." He was wrong . . . and still is. Analogously, I can recall reading of the demise of both FORTRAN and COBOL nearly 30 years ago and of ALGOL and Lisp more recently. I thought of this a lot since Labor Day. While Java books keep pouring onto the floor, books on a number of other languages have not disappeared.

(Oh yes, my Christmas/Chanukah/New Year list is, as every December, at the end of this column.)

C

Hanson's C Interfaces and Implementations is a great example of supplying a resource in one of the "passe" languages. Here, a C programmer will find the necessary support for building and employing reusable modules. Each chapter details one interface and its implementation (for example, 13 is Bit Vectors and 20 is threads). The explanations are good; the software examples are adequate. But all the source code is available on the Web (http://www.cs.princeton.edu/software/ciil), and that's really neat.

C++

I got three really interesting books on C++ in a short period of time, each from a different publisher. Stan Lippman's anthology is a fine example of "best of . . ." volumes. It contains a rich selection of "programming pearls" from the first seven years of C++ Report. All the big guns are here: Stroustrup, Waldo, Tiemann, Vlissides, Schmidt, Cargill, Koenig, etc. And you can learn a lot from these brief essays and notes. When he was at AT&T, Lippman's manager was Barbara Moo.

That enables me to transition "gracefully" to Koenig and Moo's Ruminations on C++. This coherent and well-written volume is the result of Barbara's reading and editing nearly 100 articles and columns that Andy published in JOOP, C++ Journal and C++ Report over the years. Amazingly, they have produced a valuable book in under 400 pages.

Third, there's Terrence Chan's volume on system programming. Chan covers ISO C++ programming techniques and the ISO 1003.1, 1003.1b, and 1003.1c APIs. Having read Gray's book on IPC a few months ago, I turned to Chan's chapter 10. It's a fine job. The one significant shortcoming is references: only chapters 1 and 2 have references, and there's no bibliography.

Java

There were two Java books this time round that I thought worthy of one's attention. Harold's Java Developer's Resource is certainly one way of just jumping into Java, learning how to use the AWT, and getting on with the job. The tutorial (I admit that I didn't got all the way through it) looks OK. The canned and preapproved two pages of pseudohistory (23f.) should be skipped.

Flanagan's JavaScript may well be obsolete by the time you read this. O'Reilly plans to have a new edition out toward the end of the year, covering JavaScript

Books Reviewed in this Column:


3.0 and MS Internet Explorer 3.0. This is Flanagan's fourth or fifth O'Reilly book. He has done his usual, competent job. The volume is conceptually divided into three parts: an overview of the language, a long list of bugs, and a reference manual. The bug list is worth the price of the book.

**Dylan**

I got two books on Dylan. Dylan was developed by Apple. It is "both object-oriented, like C++ and Java, and dynamic, like Smalltalk." There's a public-domain version (Mindy) available from CMU. I read about 75 pages in each book. It's not a very interesting language per se, but some of you might find it useful.

**Underlying Facts**

Some folks use languages; some are actually interested in languages. There comes a time when how-to books aren't what I need. I think of this as a time to get into the basics . . . or maybe I should think of them as the meta-level. The next two books are for those of you who really want to get into things.

If you're interested in programming languages, then Mitchell's massive work is for you. I read it in relatively small pieces over a period of six weeks. In brief, Mitchell uses lambda calculus to study the axiomatic, operational, and denotational semantics of programming languages. I found Chapter 3, "Universal Algebra and Algebraic Data Types," really impressive. I'm not certain that Chapter 9, "Polymorphism and Modularity," would be everyone's cup of tea. But if you want to understand C++ and Java and Dylan, it's important to recognize the relevance of this to data abstraction in general and C++ templates in particular.

Back in 1985, the first edition of Abelson and Sussman was the ideal volume for anyone who wanted to understand programming in the deepest sense. The new edition, now by Abelson, Sussman, and Sussman, is a pleasure to read. It is thorough, witty, and fascinating. It is also about 100 pages longer. It's never dull.

**Methodology**

Nayeem Islam was the author (or co-author) of several articles in *Computing Systems*. His *Distributed Objects* is an attempt at providing a flexible methodology for OS development. Nayeem both describes how O-O programmers can customize OSes to optimize performance and covers the strategies of a C++ framework in facilitation implementation choices.

**Ghosts of Christmas Past**

Campbell-Kelly and Aspray have turned out a solid, interesting history of computing from Babbage to Bill Gates. It's a narrative history of wide compass. The post-Apple period is whizzed through in about 25 pages, but 1981-1996 is more familiar to most of us than 1819-1980, so it doesn't matter. From Babbage and Hollerith to Licklider and the Whirlwind, it's a cracking good read, Gromit.

**Peter's gift list**

1. McKusick, Bostic, Karels, & Quarterman, *The Design and Implementation of the 4.4BSD Operating System* (Addison-Wesley)

and a bonus:


Have a happy and safe holiday; see you in the New Year!
USENIX 1997 Annual Technical Conference

January 6-10, 1997
Anaheim Marriott Hotel, Anaheim, CA

Mark your calendar! Our Annual Technical Conference will provide the latest information and tools to keep you on top of technology. Plus, the first Linux Applications Development and Deployment Conference, USELINUX, will take place at the same time. One fee covers both USENIX and USELINUX conference programs and you can switch freely between them. (Tutorial fees are separate for both.) The full program is available at our Web site, <http://www.usenix.org>. You may also send email to <conference@usenix.org> or phone 714 588 8649.

Early Registration Discount Deadline: November 22
Hotel Discount Deadline: December 20

Tutorial Program
Monday, January 6

Beginning Perl Programming for UNIX Programmers—Updated for Perl 5
Tom Christiansen, Consultant

The Kerberos Approach to Network Security—Updated
Daniel Geer, Open Market, Inc; Jon Rochlis, System Experts

An Introduction to Java
Ken Arnold, Sun Microsystems Laboratories

Secure Java Programming—New
Marianne Mueller and David Brownell, JavaSoft

Windows NT and Windows 95—The Win32 API—New
Joseph M. Newcomer, Consultant

UNIX Network Programming
Richard Stevens, Consultant

Selected Topics in System Administration—New
Trent Hein, XOR Network Engineering; Evi Nemeth, University of Colorado, Boulder

How Networks Work—The Limits of Modern Internetworking—Updated
Vincent C. Jones, Consultant

System and Network Performance Tuning—New
Hal Stern, Sun Microsystems

Inside the Linux 20 Kernel—New
Stephen Tweedie, Digital Equipment Corporation

Tuesday, January 7

UNIX Security Tools: Use and Comparison
Matt Bishop, University of California at Davis

CGI and WWW Programming in Perl—New
Tom Christiansen, Consultant

Security on the World Wide Web—New
Daniel Geer, OpenMarket, Inc; Jon Rochlis, System Experts

Creating Effective User Interfaces—New
Joseph A. Konstan, University of Minnesota

Java Applets and the AWT—New
Nataraj Nagaratnam, Syracuse University

Setting Up And Administering A Web Server—New
Bryan Buus, XOR Network Engineering

Security for Software Developers: How to Write Code that Withstands Hostile Environments—New
Marcus J. Ranum, V-ONE Corporation

Solaris System Administration—New
Marc Staveley, Consultant

IP version 6: An Introduction
Richard Stevens, Consultant

Writing Device Drivers Under Linux—New
Theodore Tso, Massachusetts Institute of Technology

Technical Program
Wednesday, January 8

9:00 am - 10:30 am
Opening Remarks
John Kohl, Pure Atria Corporation

Keynote Address
Developing on “Internet Time”
James Gosling, Sun Microsystems

USELINUX
Linux: What It Is and Why It Is Significant
Mark Bolzern, Work Group Solutions; Tom Miller, X Engineering Software Sy
ANNOUNCEMENTS & CALLS

11:00 am - 12:30 pm
Performance I
Session Chair: Carl Staelin, Hewlett-Packard Laboratories

Embedded Inodes and Explicit Grouping: Exploiting Disk Bandwidth for Small Files
Gregory R. Ganger and M. Frans Kaashoek, Massachusetts Institute of Technology

Observing the Effects of Multi-Zone Disks
Rodney Van Meter, University of Southern California, Information Sciences Institute

A Revisitation of Kernel Synchronization Schemes
Christopher Small and Stephen Manley, Harvard University

Invited Talk
Nomadicty and the IETF
Charles E. Perkins, IBM T.J. Watson Research Center

USELINUX
The Sparc Port of Linux
David S. Miller, Rutgers CAIP; Miguel de Icaza, Instituto de Ciencias Nucleares, Ciudad Universitaria, Universidad Nacional Autonoma de Mexico

2:00 pm - 3:30 pm
Interface Tricks
Session Chair: Rob Gingell, Sun Microsystems

Porting UNIX to Windows NT
David G. Korn, AT&T Laboratories

Protected Shared Libraries—A New Approach to Modularity and Sharing
Arindam Banerji, John M. Tracey, and David L. Cohn, University of Notre Dame

A Novel Way of Extending the Operating System at the User-Level: The Ufo Global File System
Albert D. Alexandrov, Maximilian Ibel, Klaus E. Schauser, and Chris J. Scheiman, University of California, Santa Barbara

Invited Talk
If Cryptography Is So Great, Why Isn’t It Used More?
Matt Blaze, AT&T Laboratories

USELINUX
Advanced Device Drivers
Alessandro Rubini, Università di Pavia

4:00 pm - 5:00 pm
Client Tricks
Session Chair: Fred Douglis, AT&T Research

Network-Aware Mobile Programs
Mudumbai Ranganathan, Anurag Acharya, Shamik Sharma, and Joel Saltz, University of Maryland

Using Smart Clients to Build Scalable Services
Chad Yoshikawa, Brent Chun, Paul Eastham, Amin Vahdat, Thomas Anderson, and David Culler, University of California, Berkeley

Invited Talk
The Inktomi Web Search Engine
Eric Brewer, University of California, Berkeley

USELINUX
4:00 pm - 5:30 pm
Future of the Linux Kernel
Linus Torvalds, Helsinki University

Thursday, January 9
9:00 am - 10:30 am
Clustering
Session Chair: Clem Cole, Digital Equipment Corporation

Building Distributed Process Management on an Object-Oriented Framework
Ken Shirriff, Sun Microsystems Laboratories

Adaptive and Reliable Parallel Computing on Networks of Workstations
Robert D. Blumofe, University of Texas, Austin, and Philip A. Lisiecki, Massachusetts Institute of Technology

A Distributed Shared Memory Facility for FreeBSD
Pedro A. Souto and Eugene W. Stark, State University of New York, Stony Brook

Invited Talk
The AltaVista Web Search Engine
Louis Monier, Digital Equipment Corporation

USELINUX
Real Time
Victor Yodaiken and Michael Barabanov, New Mexico Institute of Technology

11:00 am - 12:30 pm
Tools
Session Chair: Matt Blaze, AT&T Laboratories

Libcfd: A General and Efficient Container Data Type Library
Kiem-Phong Vo, AT&T Laboratories

A Simple and Extensible Graphical Debugger
David R. Hanson and Jeffrey L. Korn, Princeton University

Cget, Cput, and Stage-Safe File Transport Tools for the Internet
Bill Cheswick, Bell Laboratories
Invited Talk
IPv6: The New Version of the Internet Protocol
Steve Deering, Xerox Palo Alto Research Center

USELINUX
/proc
Stephen Tweedie, Digital Equipment Corporation

The Pluggable Authentication Modules (PAM) Framework
Ted "T"so, Massachusetts Institute of Technology

2:00 pm - 3:30 pm
Works in Progress
Session Chair: John Schimmel, Silicon Graphics, Inc.

Invited Talk
Highlights from 1996 USENIX Conferences and Workshops

USELINUX
Standards
Heiko Eissfeldt, Unifix Software

4:00 pm - 5:30 pm
Joint Session Inferno
Rob Pike, Bell Laboratories

USELINUX
Connecting Legacy and Open Systems
Michael Callahan, Stelias Computing, Inc.

Friday, January 10

9:00 am - 10:30 am
User Something
Session Chair: Nathaniel Borenstein, First Virtual Holdings

WebGlimpse—Combining Browsing and Searching
Udi Manber, Michael Smith, and Burra Gopal, University of Arizona

Mailing List Archive Tools
Sam Leffler and Melange Tortuba, Silicon Graphics, Inc.

Experience with GroupLens: Making Usenet Useful Again
Bradley N. Miller, John T. Riedl, and Joseph A. Konstan,
University of Minnesota

Invited Talk
Measuring Computer Systems: How to Tell the Truth with Numbers
Margo Seltzer and Aaron Brown, Harvard University

USELINUX
Linux: What It Is and Why It Is Significant
Mark Bolzern, Work Group Solutions; Tom Miller,
X Engineering Software Systems

Linux and Distribution Channels: Ways to Enter the Commercial Market
Don Rosenberg, Stromian Technologies

11:00 am - 12:30 pm
Performance II
Session Chair: Mike Karels, Berkeley Software Design

Overcoming Workstation Scheduling Problems in a Real-Time Audio Tool
Isidor Kouvelas and Vicky Hardman, University College London

On Designing Lightweight Threads for Substrate Software
Matthew Haines, University of Wyoming

High-Performance Local-Area Communication With Fast Sockets
Steven H. Rodrigues, Thomas E. Anderson, and David E. Culler, University of California, Berkeley

Invited Talk
Stupid Net Tricks
Bill Cheswick, Bell Laboratories

USELINUX Business
Using Linux in Your Business: A Business Justification
Presented by Linux International

2:00 pm - 3:30 pm
Caching and Stashing
Session Chair: Bill Bolosky, Microsoft Research

An Analytical Approach to File Prefetching
Hui Lei and Dan Duchamp, Columbia University

Optimistic Deltas for WWW Latency Reduction
Gaurav Banga, Fred Douglish, and Michael Rabinovich,
AT&T Laboratories

A Toolkit Approach to Partially Connected Operation
Dan Duchamp, Columbia University

Invited Talk
Finding Bugs in Concurrent Programs
Gerard J. Holzmenn, Bell Laboratories

USELINUX Business
2:00 pm - 4:00 pm
Member of the Board, Linux International

4:15 pm - 5:45 pm
Joint Closing Session
Severe Tire Damage's Stupid Mbone Tricks—A Lecture/Demonstration
Announcement and Call for Participation

Tcl/Tk Workshop '97

July 14-17, 1997
Boston, Massachusetts

Sponsored by theUSENIX Association

Important Dates
Paper, demonstrations, and panel
proposals: March 11, 1997
Acceptance notification: April 8, 1997
Poster submissions: April 22, 1997
Camera-ready copy: June 3, 1997

Conference Organizers
Co-chairs
Joe Konstan, University of Minnesota
Brent Welch, Sun Microsystems Laboratories, Inc.

Program Committee
Dave Beazley, University of Utah
Mark Harrison, DSC Communications
Jeffrey Hobbs, University of Oregon
George Howlett, Bell Labs Innovations
Ray Johnson, Sun Microsystems Laboratories, Inc.
Kevin Kenny, General Electric Corporate R&D
Gerald Lester, Computerized Processes
John LoVerso, Open Group Research Institute
Michael J. McLennan, Bell Labs Innovations
Brian Smith, Cornell University

The fifth annual Tcl/Tk workshop, sponsored by the USENIX Association, will be held July 14–17, 1997 in Boston, Massachusetts. The workshop is a forum to:

■ Bring together Tcl/Tk researchers and practitioners
■ Publish and present current work involving Tcl/Tk
■ Learn about the latest developments in Tcl/Tk
■ Plan for future Tcl/Tk related developments

The workshop program will include formal paper and panel presentations, poster and demonstration sessions, works in progress (WIP) sessions, birds of a feather (BoF) sessions, and tutorials.

The workshop program will include formal paper and panel presentations, poster and demonstration sessions, Works-in-Progress (WIP) sessions, Birds-of-a-Feather (BoF) sessions, and tutorials.

Forms of Participation
All forms of participation provide an opportunity to report on original Tcl/Tk research. Topics include, but are not limited to, system extensions, novel Tcl/Tk based applications, experience reports on building applications in Tcl/Tk, comparative evaluations of Tcl/Tk and other languages or toolkits for building applications, use of different programming paradigms in Tcl/Tk, and proposals for new directions. The audience for all submissions is practitioners and researchers who are experienced users of Tcl/Tk. For this reason, reports on experiences and applications must draw out lessons for other Tcl/Tk developers.

Papers
Papers are limited to ten pages, and authors of accepted papers will be given twenty minutes to present the paper at the workshop. A full version of the paper must be submitted for review. Papers must be written in the English language and paper authors are encouraged to include black-and-white figures in their papers.

Papers will be reviewed by the program committee and evaluated by the following criteria:

■ Quantity and quality of novel content
■ Relevance and interest of content to the Tcl/Tk Workshop audience
■ Quality of presentation of content in the paper
■ Suitability of content for presentation at the workshop

Papers should present a cohesive piece of work. Shorter papers are encouraged when the research covered can be adequately described and discussed in fewer pages. Papers may report on non-commercial or commercial systems. Papers with blatant marketing content, however, will not be accepted.
In prior workshops, authors of papers describing applications built using Tcl/Tk and experiences using Tcl/Tk often encountered difficulties targeting the content to the workshop attendees. Application and experience papers need to strike a delicate balance between giving too little background on the application domain so that the audience cannot follow the paper, and devoting too much space to the application domain so that the relevance of Tcl/Tk is not addressed adequately. Application and experience papers should clearly explain how the application or experience illustrates a novel use of Tcl/Tk and what lessons the Tcl/Tk community can derive from the application or experience to apply to their own development efforts.

This workshop, like most conferences and journals, requires that papers not be submitted simultaneously to another conference or publication and that submitted papers not be published elsewhere. Papers accompanied by non-disclosure agreement forms are not acceptable and will be returned to the author(s) unread. All submissions are held in the highest confidentiality prior to publication in the Proceedings, both as a matter of policy and in accord with the U.S. Copyright Act of 1976.

Posters
Poster submissions are new for the 1997 Tcl/Tk workshop. They provide an opportunity to present interesting results, including preliminary results, with less time and effort than is typically needed for a paper. They are also the ideal submission category for submitting material that is better presented to small groups than as a large-group presentation.

Posters will be displayed during one day of the workshop and a poster session will provide an opportunity for workshop attendees to interact with poster authors individually and in small groups. The workshop will provide display space of approximately 3 feet wide by 4 feet high on which to display the poster. Poster authors should submit a draft of the poster contents along with a one-page abstract. Abstracts of accepted posters will be published in the conference proceedings.

Demonstrations
The Tcl/Tk Workshop is experimenting with a new format for demonstrations for 1997. We will be holding a demonstration reception on one of the evenings of the workshop at which demonstrations will be held in parallel, allowing attendees to more closely interact with the demonstrators. Space will be available for reviewed and informal demonstrations.

Reviewed demonstrations will be given a demonstration station for the entire session and will have an abstract published in the conference proceedings. Submissions should include both a one-page abstract and six copies of a videotape (VHS) showing the demonstration. Demonstrations judged to be of wide interest may also be asked to present a version of the demonstration during a conference session.

Informal demonstrations will be assigned a specific time during the demonstration session. Authors of accepted papers as well as those with demonstration-ready works in progress are encouraged to sign up for informal demonstration time slots. More information on the facilities available for informal demonstrations will be provided in the registration packet.

Demonstrations of commercial products of interest to the Tcl/Tk community are encouraged. The abstract for the proceedings, however, should avoid commercial content (i.e., it should not include pricing and sales information or marketing content).

Panel Proposals
The program committee is responsible for organizing panel discussion of up to 90-minutes on topics of interest to the workshop attendees. We invite panel proposals that include a list of panelists (who have agreed to serve on the panel), a topic and format, and a panel description for the proceedings with position statements from each panelist. Panels should have no more than four speakers, including the panel moderator, and should include substantial interaction with workshop attendees. Panels should not simply be presentations of related research papers — papers should be submitted individually and the program committee will group them into sessions of related material.

WIP Presentations and BoF Sessions
Work in progress presentations and birds of a feather sessions are not reviewed. Both are available on a first-come, first served basis starting in June 1997. Specific instructions for reserving WIP and BoF time slots will be provided in the registration brochure in April 1997. Some WIP and BoF time slots will be held open for on-site reservation, so we encourage all attendees with interesting work in progress to consider presenting that work at the workshop.

Detailed Submission Instructions
Please see http://www.usenix.org/tcl97 or send email to tclauthors@usenix.org

More information on the workshop will be posted to comp.lang.tcl, comp.org.usenix, and placed on the World Wide Web at http://www.usenix.org as it becomes available.

Registration Materials
Materials containing all details of the technical and tutorial programs, registration fees and forms, and hotel information will be available in April, 1997. If you wish to receive the registration materials, please contact:

USENIX Conference Office
22672 Lambert Street, Suite 613
Lake Forest CA 92630
Phone: 714.588.8649
Fax: 714.588.9706
Email: conference@usenix.org
URL: http://www.usenix.org
Announcement and Call for Papers
Conference on Domain-Specific Languages

October 15-17, 1997
Red Lion Resort, Santa Barbara, California

Sponsored by the USENIX Association in cooperation with ACM SIGPLAN (pending)

Important Dates
Papers due: June 13, 1997
Author notification: July 10, 1997
Camera-ready final papers due: September 2, 1997

Program Committee
Program Chair: Chris Ramming, AT&T Labs
Thomas J. Ball, Lucent Bell Laboratories
Gerard Berry, CMA, Ecole des Mines de Paris
Jon Bentley, Lucent Bell Laboratories
Peter Buneman, University of Pennsylvania
Luca Cardelli, Digital Equipment Corporation
Steve Johnson, Transmeta Corporation
Takayuki Dan Kimura, Washington University
Todd Knoblock, Microsoft Research
Speaker Chair: David Ladd, Spyglass
Adam Porter, University of Maryland
Jan Prins, University of North Carolina at Chapel Hill

Introduction
Language is central to the discipline of software engineering. Programmers use a variety of languages in their daily work, and new languages appear frequently. This proliferation is not gratuitous: each new language offers specific solutions to genuine software problems. However, not all languages address the problem of general-purpose computing; domain-specific languages (DSLs) are explicitly designed to cover only a narrow class of problems, while offering compelling advantages within that class. This conference is dedicated to the discussion of the unique aspects of DSL design, DSL implementation, and the use of DSLs in software engineering.

Domain-specific languages give rise to a number of questions. What are the design principles for the creation of new DSLs? How can the process of DSL design be codified and structured? What roles can domain-specific languages play in software engineering? How does the use of domain-specific languages affect software engineering process? What are the tools, environments, and techniques needed to support the use of domain-specific languages? What are the concrete technical advantages and disadvantages of domain-specific languages? What are the economic costs and benefits of domain-specific languages? These and other questions are the focus of this conference on domain-specific languages.

The conference seeks to advance the practice of DSL design, DSL implementation, and software engineering generally by:

- eliciting examples of successful domain-specific languages
- highlighting the spectrum of benefits which domain-specific languages can provide
- discovering design principles and methodologies for creating DSLs
- eliciting design techniques and tools for working with domain-specific languages throughout the software engineering lifecycle
- providing a framework within which language designers from different domains can easily communicate
- establishing the practical value of domain-specific languages through the publication of empirical data concerning productivity, quality, and maintainability
- creating a community that will continue to study and refine the practice of software engineering through domain-specific languages
Conference Topics

The technical sessions will include refereed papers, invited talks, and Birds-of-a-Feather (BoF) sessions. We seek papers that draw on experience in a wide variety of areas, including but not limited to the following topics.

- formal methods
- software design and architecture
- declarative languages
- software engineering
- software process
- database languages
- program analysis and automated transformation
- computer architecture
- design process and languages
- visual languages and environments
- hardware specification languages
- parallel computing languages
- type theory
- distributed computing languages
- testing
- prototyping

Paper Criteria

Papers will be judged on the depth of their insight and the extent to which they translate specific experience into general lessons for domain-specific language designers, and implementers, and software engineers.

Papers can range from the practical to the theoretical; papers should refer to actual languages, tools, and techniques with pointers to full definitions and implementations where possible. Empirical data on results should be included where possible.

How to Submit a Paper

Technical paper submissions must be received by June 13, 1997. Full papers are requested and should be 10 to 15 pages (around 5,000–6,000 words).

All submissions will be judged on originality, relevance, and correctness. Each accepted submission will be assigned a member of the program committee to shepherd preparation of the final paper. The assigned member will act as a conduit for feedback from the committee to the authors. Camera-ready final papers are due September 2, 1997.

Each submission must include a cover letter stating the paper title and authors along with the name of the person who will act as the contact to the program committee. Please include a surface mail address, daytime and evening phone number, an email address, and fax number for the contact person.

If you would like to receive detailed guidelines for submission send email to dilauthors@usenix.org. An electronic version of this document is available at http://www.usenix.org.

The DSL conference, like most conferences and journals, requires that papers not be submitted simultaneously to another conference or publication and that submitted papers not be previously or subsequently published elsewhere. Papers accompanied by “non-disclosure agreement” forms are not acceptable and will be returned to the author(s) unread. All submissions are held in the highest confidentiality prior to publication in the Proceedings, both as a matter of policy and in accord with the U.S. Copyright Act of 1976.

Please send one copy of a full paper to the program committee via one of the following methods. All submissions will be acknowledged.

- Preferred Method: email (Postscript) to dlpapers@usenix.org
- Alternate Method: postal delivery to DSL Conference c/o Chris Ramming USENIX Association 2560 Ninth Street, Suite 215 Berkeley CA 94710 Phone: 510.528.8649

Invited Talks

There will be several invited talks at the conference. If you have suggestions for possible speakers, please send them to the speaker chair, David Ladd (dladd@spyglass.com).

Registration Materials

Materials containing all details of the technical and tutorial programs, registration fees and forms, and hotel information will be available beginning in August, 1997. If you wish to receive the registration materials, please contact USENIX at:

USENIX Conference Office
22672 Lambert Street, Suite 613
Lake Forest, CA USA 92630
Phone: 714.588.8649
Fax: 714.588.9706
Email: conference@usenix.org
URL: http://www.usenix.org
<table>
<thead>
<tr>
<th>General Technical Conferences</th>
<th>Price: Member/List</th>
<th>Overseas Postage</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1996 San Diego, CA</td>
<td>$32/40</td>
<td>$18.</td>
<td></td>
</tr>
<tr>
<td>Summer 1994 Boston, MA</td>
<td>25/33</td>
<td>20.</td>
<td></td>
</tr>
<tr>
<td>Summer 1993 Cincinnati, OH</td>
<td>25/33</td>
<td>18.</td>
<td></td>
</tr>
<tr>
<td>Winter 1993 San Diego, CA</td>
<td>33/40</td>
<td>25.</td>
<td></td>
</tr>
<tr>
<td>Winter 1991 Dallas, TX</td>
<td>28/34</td>
<td>18.</td>
<td></td>
</tr>
<tr>
<td>Summer 1990 Anaheim, CA</td>
<td>22/22</td>
<td>15.</td>
<td></td>
</tr>
<tr>
<td>Summer 1989 Balimore, MD</td>
<td>20/20</td>
<td>15.</td>
<td></td>
</tr>
<tr>
<td>Winter 1988 Dallas, TX</td>
<td>26/26</td>
<td>15.</td>
<td></td>
</tr>
<tr>
<td>Summer 1986 Atlanta, GA</td>
<td>37/37</td>
<td>20.</td>
<td></td>
</tr>
<tr>
<td>Summer 1985 Portland, OR</td>
<td>45/45</td>
<td>25.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Systems Administration (LISA) Conferences</th>
<th>Price: Member/List</th>
<th>Overseas Postage</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Systems Administration (LISA X)</td>
<td>September 1996</td>
<td>30/38</td>
<td>12.</td>
</tr>
<tr>
<td>Systems Administration (LISA I)</td>
<td>September 1995</td>
<td>30/38</td>
<td>11.</td>
</tr>
<tr>
<td>Systems Administration (LISA VII)</td>
<td>November 1993</td>
<td>25/33</td>
<td>12.</td>
</tr>
<tr>
<td>Large Installation Systems Administration (LISA V)</td>
<td>September 1991</td>
<td>20/23</td>
<td>11.</td>
</tr>
<tr>
<td>Large Installation Systems Administration (LISA IV)</td>
<td>October 1990</td>
<td>15/18</td>
<td>8.</td>
</tr>
<tr>
<td>Large Installation Systems Administration (LISA III)</td>
<td>September 1989</td>
<td>13/13</td>
<td>9.</td>
</tr>
<tr>
<td>Large Installation Systems Administration (LISA II)</td>
<td>November 1988</td>
<td>8/8</td>
<td>5.</td>
</tr>
<tr>
<td>Large Installation Systems Administration (LISA I)</td>
<td>April 1987</td>
<td>4/4</td>
<td>5.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Security</th>
<th>Price: Member/List</th>
<th>Overseas Postage</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX Security</td>
<td>August 1988</td>
<td>7/7</td>
<td>5.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operating Systems</th>
<th>Price: Member/List</th>
<th>Overseas Postage</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mach I Workshop</td>
<td>October 1990</td>
<td>17/20</td>
<td>9.</td>
</tr>
</tbody>
</table>
Programming Languages

- 4th Tcl/Tk Workshop: July 1996, 22/28
- 3rd Tcl/Tk Workshop: July 1995, 29/34
- Conf. on Object-Oriented Technologies and Systems II: June 1996, 20/30
- Conf. on Object-Oriented Technologies and Systems I: June 1995, 18/24
- Very High Level Languages: October 1994, 23/30
- C+ Conference: April 1994, 24/28
- C++ Conference: August 1992, 30/39
- C++ Conference: April 1991, 22/26
- C++ Conference: April 1990, 28/28
- C++ Conference: October 1988, 30/30
- C++ Workshop: November 1987, 30/30

Other Symposia & Workshops

- Electronic Commerce Workshop: November 1996, 20/26
- Electronic Commerce Workshop: July 1995, 20/26
- Mobile and Location Independent Computing II: April 1995, 18/24
- Mobile and Location Independent Computing I: August 1993, 15/20
- High-Speed Networking: August 1994, 15/20
- UNIX Applications Development: April 1994, 15/20
- File Systems Workshop: May 1992, 15/20
- Graphics V Workshop: November 1989, 18/18

USENIX CD-ROM

- LISA X: September 1996, 65/90

Discounts are available for bulk orders of 10 or more of the same proceeding.

SAGE Publication: Short Topics in System Administration Series

<table>
<thead>
<tr>
<th>Quantity</th>
<th>List/Member</th>
<th>Overseas Postage</th>
<th>Subtotal</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1: Job Descriptions for System Administrators</td>
<td>5/7.50</td>
<td>$3.50</td>
<td></td>
</tr>
<tr>
<td>#2: A Guide to Developing Computing Policy Documents</td>
<td>5/7.50</td>
<td>$3.50</td>
<td></td>
</tr>
</tbody>
</table>

Reprints:
Reprints of individual papers from all proceedings are available for $5.00 each. (To get a complete listing of conference papers, please refer to our Online Library Index, available through the WWW at <http://www.usenix.org>.)

Payment Options

☐ Check enclosed payable to USENIX Association. ☐ Purchase order enclosed ☐ Visa ☐ MasterCard

Account# ____________________________ Expiration Date ____________ Signature ____________________________

Outside the USA? Please make your payment in U.S. currency by one of the following:
☐ Check — issued by a local branch of a US bank ☐ Charge (Visa, MasterCard or equivalent) ☐ International postal money order

Total Price of Publications ____________________________
Calif. res. add sales tax ____________________________
Overseas Postage ____________________________

TOTAL ENCLOSED $ ____________________________

*If you are paying the member price, please include member's name and/or membership number:

☐ If you are not a member and wish to receive our membership information packet, please check the box.

Please return this order form and your payment to:
USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710  Fax: 510/548-5738  Phone: 510/528-8649  office@usenix.org  http://www.usenix.org
Visit the Vendors!
USENIX 1997 VENDOR EXHIBITION

Wednesday, January 8
Noon - 7 pm

Thursday, January 9
10 am - 4 pm
Anaheim Marriott, Anaheim, CA

The popularity and importance of the USENIX Annual Technical Conference is once again demonstrated by our “sold out” accommodations for our vendors.

A broad spectrum of products and services will be on display representing the newest tools and most cutting-edge technologies. In the relaxed environment, attendees have time for in depth discussion with technical representatives. You will find the product and services of some 58 vendors on display.

Companies with reserved space at USENIX:

- Addison-Wesley, Booth 58
- Advanced Digital Information Corp., Booth 19
- Aspen Systems, Inc., Booth 62
- Auspex Systems, Inc., Booth 1
- Caldera, Inc., Booth 61
- Centon Electronics, Inc., Booth 53
- Clarity Software, Inc. Booth 10
- Competitive Automation, Booth 33
- Cosmos Engineering Company, Booth 42
- CrossWind Technologies, Inc., Booth 66
- Digital Equipment Corp., Booth 63
- Devcom Mid America, Inc., Booth 49
- Enhanced Software Technologies Inc., Booth 43
- Enterprise Systems Management Corp., Booth 7
- Falcon Systems Inc., Booth 34
- Fastlane Systems Ltd., Booth 30
- Fujitsu Microelectronics, Booth 52
- GraphOn Corporation, Booth 36
- InfoGain, Inc., Booth 12
- Info Magic Inc., Booth 31
- Legato Systems, Inc., Booth 18
- Lexa Software Corporation, Booth 27
- Linux Journal, Booth 26
- Lucent Technologies, Inc., Booth 3
- MTI, Booth 18
- McAfee Associates, Booth 8
- MaX Enhancement Group, Booth 64
- NEC Technologies, Booth 13
- NET-Community, Booth 16
- Network Appliance, Inc., Booth 21
- O’Reilly & Associates, Inc., Booth 38
- Parity Systems, Inc., Booth 67
- PDC, Booth 20
- Pencom Systems Administration, Booth 41
- Platinum Informatics, Booth 25
- Prentice Hall PTR, Booth 54
- Pure Atria Software Inc., Booth 44
- QMASTER Software Solutions Ltd, Booth 45
- Raima Corporation, Booth 65
- RDI Computer Corp., Booth 28
- Red Hat Software, Inc., Booth 47
- San Diego Technical Books, Booth 55
- Solfower Computer, Inc., Booth 9
- Spire Technologies, Inc., Booth 37
- Storage Computer Corporation, Booth 29
- SunSoft, Inc., Booth 22
- Syntax, Booth 11
- Transitional Technology Inc. (TTI), Booth 51
- Unisolutions Associates, Booth 50
- UniTree Software, Inc., Booth 23
- UNIX Review, Booth 32
- Vanguard Technology, Booth 17
- Walnut Creek CDROM, Inc., Booth 15
- Western Scientific, Booth 5
- WRQ, Makers of Reflection) Software, Booth 14
- Wyatt River Software, Booth 59
- Yggdrasil Computing Inc., Booth 46

ADMISSION TO THE EXHIBITION IS FREE

If you cannot make it to the conference, but would like to visit the exhibition, please contact Cynthia Deno, 408 335 9445, Fax: 408 335 5327, or email to <cynthia@USENIX.org>.
We've just released new, updated editions of many of the books in our popular collection of guides for UNIX developers. With lots of new insights and new information—much of it prompted by our reader's comments and suggestions—these books are now more useful than ever. The new 2nd edition of *Programming Perl* also unveils the many new secrets of Perl version 5, which has become very popular among Web developers. **SAVE 25%!** If you have the previous editions of these books, we'll give you a 25% discount when you upgrade. To receive the discount, complete the order form below and mail it to us along with your payment and the title page from the edition you own (no photocopies, please).

---

## What a Deal!
Attached is the title page from my well worn, O'Reilly books. Please send me the revised and updated 2nd Editions of the following titles, and take 25% off the list price for me.

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Title</th>
<th>List price</th>
<th>Discount</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>sed &amp; awk, 2nd Ed.</em> #2255</td>
<td>$27.95</td>
<td>$20.95</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>sendmail, 2nd Ed.</em> #2220</td>
<td>$32.95</td>
<td>$24.95</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>Programming Perl, 2nd Ed.</em> #1496</td>
<td>$39.95</td>
<td>$29.95</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>DNS and BIND, 2nd Ed.</em> #2360</td>
<td>$32.95</td>
<td>$24.95</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>Software Portability with make, 2nd Ed.</em> #0554</td>
<td>$32.95</td>
<td>$24.95</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>Using &amp; Managing UUCP</em> #1534</td>
<td>$29.95</td>
<td>$22.95</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>Learning GNU Emacs</em> #1526</td>
<td>$29.95</td>
<td>$22.95</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>Practical UNIX &amp; Internet Security</em> #1489</td>
<td>$39.95</td>
<td>$29.95</td>
<td></td>
</tr>
</tbody>
</table>

---

Due to the special nature of this offer, we cannot accept phone, fax or online orders. Please mail your check or credit card info to: O'Reilly & Associates, Inc. 101 Morris St. Sebastopol, California 95472. Special offer code ALO2G (25% Discount with title page of 1st Edition). This discount is not valid in conjunction with any other offer.
Computer Publications from John Wiley & Sons, Inc.

USENIX members receive a 15% discount

<table>
<thead>
<tr>
<th>Title</th>
<th>ISBN</th>
<th>Member Price</th>
<th>Number of Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Internet Navigator, 2nd</td>
<td>1-05260-4</td>
<td>$24.95</td>
<td></td>
</tr>
<tr>
<td>Paul Gilster</td>
<td></td>
<td>member price: $21.20</td>
<td></td>
</tr>
<tr>
<td>Advanced Topics in UNIX</td>
<td>1-03663-3</td>
<td>$24.95</td>
<td></td>
</tr>
<tr>
<td>Ronald Leach</td>
<td></td>
<td>member price: $21.20</td>
<td></td>
</tr>
<tr>
<td>Introduction to Client Server Systems</td>
<td>1-57774-X</td>
<td>$34.95</td>
<td></td>
</tr>
<tr>
<td>Paul Renaud</td>
<td></td>
<td>member price: $29.70</td>
<td></td>
</tr>
<tr>
<td>Portable UNIX</td>
<td>1-57926-2</td>
<td>$14.95</td>
<td></td>
</tr>
<tr>
<td>Douglas Topham</td>
<td></td>
<td>member price: $12.71</td>
<td></td>
</tr>
<tr>
<td>UNIX, Self-Teaching Guide</td>
<td>1-57924-6</td>
<td>$19.95</td>
<td></td>
</tr>
<tr>
<td>George Leach</td>
<td></td>
<td>member price: $16.95</td>
<td></td>
</tr>
<tr>
<td>Object Oriented Programming with Turbo C++</td>
<td>1-52466-2</td>
<td>$24.95</td>
<td></td>
</tr>
<tr>
<td>Keith Weiskamp</td>
<td></td>
<td>member price: $21.20</td>
<td></td>
</tr>
<tr>
<td>Obfuscated C and Other Mysteries</td>
<td>1-57805-3</td>
<td>$39.95</td>
<td></td>
</tr>
<tr>
<td>Don Libes</td>
<td></td>
<td>member price: $33.96</td>
<td></td>
</tr>
<tr>
<td>Finding It On the Internet</td>
<td>1-03857-1</td>
<td>$19.95</td>
<td></td>
</tr>
<tr>
<td>Paul Gilster</td>
<td></td>
<td>member price: $16.95</td>
<td></td>
</tr>
<tr>
<td>Internationalization: Developing Software for Global Markets</td>
<td>1-07661-9</td>
<td>$29.95</td>
<td></td>
</tr>
<tr>
<td>Tuoc Luong</td>
<td></td>
<td>member price: $25.45</td>
<td></td>
</tr>
<tr>
<td>Adventures in UNIX Network Applications Programming</td>
<td>1-52858-7</td>
<td>$39.95</td>
<td></td>
</tr>
<tr>
<td>Bill Rieken</td>
<td></td>
<td>member price: $33.96</td>
<td></td>
</tr>
<tr>
<td>UNIX Shell Programming, 3e</td>
<td>1-59941-7</td>
<td>$29.95</td>
<td></td>
</tr>
<tr>
<td>Lowell Jay Arthur</td>
<td></td>
<td>member price: $25.45</td>
<td></td>
</tr>
<tr>
<td>The UNIX Command Reference Guide</td>
<td>1-85580-4</td>
<td>$32.95</td>
<td></td>
</tr>
<tr>
<td>Kaare Christian</td>
<td></td>
<td>member price: $28.01</td>
<td></td>
</tr>
<tr>
<td>Berkeley UNIX: A Simple &amp; Comprehensive Guide</td>
<td>1-61582-X</td>
<td>$40.95</td>
<td></td>
</tr>
<tr>
<td>James Wilson</td>
<td></td>
<td>member price: $34.80</td>
<td></td>
</tr>
</tbody>
</table>

Please send all orders to:
John Wiley & Sons, Inc.
Attn: Karen Cooper, Special Sales
605 Third Avenue
New York, NY 10158
Phone: (212) 850-6789
Fax: (212) 850-6142

Payment enclosed, plus sales tax
VISA
Mastercard
American Express
Card No.
Name
Firm
Address
City/State/Zip
Signature
(order invalid unless signed)
GLOBAL NETWORKS
Computers and International Communication
edited by Linda M. Harasim
Global Networks takes up the host of issues raised by the new networking technology that now links individuals, groups, and organizations in different countries and on different continents. The twenty-one contributions focus on the implementation, application, and impact of computer-mediated communication in a global context.
340 pp $29.95 hardcover HARNH

THE NETWORK NATION
Human Communication via Computer
Revised Edition
Starr Roxanne Hiltz and Murray Turoff
"The Network Nation... contained a fascinating vision. ...it is a place where thoughts are exchanged easily and democratically and intellect affords one more personal power than a pleasing appearance does. Minorities and women compete in equal terms with white males, and the elderly and handicapped are released from the confines of their infirmities to skim the electronic terrain as swiftly as anyone else." — Teresa Carpenter, Village Voice
580 pp $24.95 paperback HILWP

THE EVOLUTION OF C++
Language Design in the Marketplace of Ideas
edited by Jim Waldo
This collection of articles traces the history of C++, from its infancy in the Santa Fe workshop, to its proliferation today as the most popular object-oriented language for microcomputers. Waldo notes in his postscript that in the process of evolving, the language has lost a clearly articulated, generally accepted design center, with no common agreement about what it should or should not do in the future.
279 pp $24.95 paperback WALEP

THE DIGITAL WORD
Text-Based Computing in the Humanities
edited by George P. Landow and Paul Delany
This book explores the larger realm of the knowledge infrastructure where texts are received, reconstructed, and sent over global networks.
Technical Communication and Information Systems series 384 pp $39.95 hardcover LANDH

SOCIOMEDIA
Multimedia, Hypermedia, and the Social Construction of Knowledge
edited by Edward Barrett
Sociomedia continues the assessment of hypermedia systems begun in Text, Conference, and Hypermedia and The Society of Text. It examines the use of integrated multimedia to support social and collaborative research, learning, and instruction in the university, one of the best environments for developing and analyzing the effects of computing technologies on our understanding of complex sets of information.
Technical Communications and Information Series 360 pp $50.00 hardcover BARRH

CONNECTIONS
New Ways of Working in the Networked Organization
Lee Sproull and Sara Kiesler
"...Sproull and Kiesler raise crucial questions about our technical and particularly our human strategies as a producing society." — Howard Webber, Sloan Management Review
228 pp $21.95 paperback SPRCP

TECHNOBABBLE
John A. Barry
288 pp $12.50 paperback BARCP

Please send me these titles

Payment enclosed [ ] Purchase Order Attached [ ] Charge to my: [ ] Master Card [ ] VISA

Card #: exp. Signature

3.00 Total for books(s)
Postage for North American addresses
Canadian customers add 7% GST
Total for books(s) & postage

Make checks payable and send order to:
THE MIT PRESS
55 Hayward Street, Cambridge, MA 02142-1399 USA
To order by phone, call (617) 625-8569 or (800) 356-0343. E-mail order # mitpress-orders@mit.edu. The operator will need this code: UNIX1.
Prentice Hall PTR is pleased to recommend the following titles to USENIX members...

USENIX members receive a 15% discount on orders

(15105-0) List: $48.00 Members: $40.80

Object-Oriented Modeling and Design, James Rumbaugh, 0-13-629841-9
(62984-0) List: $54.00 Members: $45.90

Zen and the Art of the Internet, Third Edition, Brendan Kehoe, 0-13-121492-6
(12149-1) List: $23.95 Members: $20.36

The Magic Garden Explained, Bernard Goodheart/James Cox, 0-13-098138-9
(09813-7) List: $38.00 Members: $32.30

(47224-1) List: $61.33 Members: $52.13

SCO Performance Tuning Handbook, Gina Miscovich/David Simons, 0-13-102690-9
(10269-9) List: $42.00 Members: $35.70

Object-Oriented Programming, Peter Coad/Jill Nicola, 0-13-032616-X
(03261-5) List: $48.00 Members: $40.80

(47422-1) List: $53.00 Members: $45.05

(47423-9) List: $53.00 Members: $45.05

The Internet Message: Closing the Book with Electronic Mail, Marshall T. Rose, 0-13-092941-7
(09294-0) List: $50.00 Members: $42.50

The Standard C Library, PJ Plauger, 0-13-131509-9
(13150-8) List: $37.80 Members: $32.13

(30957-5) List: $42.00 Members: $35.70

The Simple Book: An Introduction to Internet Management, Marshall T. Rose, 0-13-177254-6
(17725-3) List: $55.00 Members: $46.75

Networking Operations on UNIX SVR4, Mike Padavano, 0-13-613555-2
(61355-4) List: $50.00 Members: $42.50

Solaris Porting Guide, Sunsoft ISV Engineering, 0-13-030396-8
(03039-5) List: $52.00 Members: $44.20

Multiprocessor System Architectures, Ben Catanzaro, 0-13-089137-1
(08913-6) List: $44.00 Members: $37.40

The HP-UX System Administrator's "How To" Book, Marty Poniatowski, 0-13-099821-4
(09982-0) List: $32.00 Members: $27.20

UNIX System V Performance Management, Edited by Phyllis Eve Bregman and Sally A. Browning, 0-13-016429-1
(01642-8) List: $29.95 Members: $25.45

SCO* UNIX* Operating System System Administrator's Guide, Santa Cruz Operation, 0-13-012568-7
(01256-7) List: $39.00 Members: $33.15

HERE'S HOW TO ORDER:

CALL 800-880-6818

OR WRITE: CompuBooks Route 1, Box 271-D, Cedar Creek, TX 78612

OR INTERNET: 70007.1333@CompuServe.com (GO CBK on CompuServe)

WE SHIP ANYWHERE!

FOR MORE INFORMATION, OR QUANTITY ORDERS, PLEASE CALL 201-592-2657
A UNIQUE OFFER
ON THE BEST IN UNIX
FOR USENIX MEMBERS

□ THE INTERNET
GUIDE FOR NEW
USERS
D. Dern
hardcover, 016510-6, $40.00,
MEMBER PRICE $32.00
paperback, 016511-4, $27.95,
MEMBER PRICE $22.36

□ INTERNET FOR
EVERYONE
R. Wiggins
hardcover, 067018-8, $29.95,
MEMBER PRICE $23.96
paperback, 067019-6, $45.00,
MEMBER PRICE $36.00

□ THE ESSENTIAL
INTERNET
INFORMATION GUIDE
J. Manger
707905-1, paperback, $27.95,
MEMBER PRICE $22.36

□ 20% DISCOUNT FROM
McGRAW-HILL

□ THE INFORMATION
BROKERS
HANDBOOK,
Second Edition
S. Rugge
911878-x, paperback, $34.95,
MEMBER PRICE $27.96
Available December 1994

□ UNIX DEVELOPER’S
TOOL KIT
K. Leininger
911836-4, $65.00,
MEMBER PRICE $52.00

□ UNIX SECURITY:
A Practical Tutorial
N. Arnold
002560-6, $24.95,
MEMBER PRICE $19.96

□ THE UNIX AUDIT:
Using UNIX to Audit
UNIX
M. Grottola
025127-4, $32.95,
MEMBER PRICE $26.36
Available November 1994

□ UNIX: A Database
TOUNIX
H. Hahn
025511-3, paperback, $28.00,
MEMBER PRICE $22.40

□ A STUDENT’S GUIDE
TOUNIX
S. Das
015745-6, $29.95,
MEMBER PRICE $23.96
Available November 1994

I am a member of USENIX Association. Please
send me the books I have indicated at the
member special rate. I have added $3.00 postage
and handling for the first book ordered, $1.00
for each additional book, plus my local sales tax.

Check or money order is enclosed—
payable to McGraw-Hill, Inc.
Charge my □ Visa □ Mastercard
□ Discover □ Amex

Bill & Ship To:

Name __________________________________________
Street __________________________________________
City, State, Zip __________________________________
Daytime Phone # ____________________________

Account # ______________________________________
Expiration Date ____________________________
Signature ______________________________________

USENIX Membership # __________________________

If I am not completely satisfied, I will return the book(s) within 15 days for a full refund or credit. Satisfaction unconditionally
guaranteed. Prices subject to change without notice. We can only accept orders within the continental USA.
Web Server Technology
The Advanced Guide for World Wide Web Information Providers

Nancy J. Yeager and Robert E. McGrath
National Center for Supercomputing Applications

This authoritative presentation of web server technology takes you beyond the "how-to" guides to provide an understanding of the underlying principles and technical details of how Web servers really work. It explains the current state of the art and suggests how the technology will evolve. Topics covered include: performance, caching, search, security, and digital commerce.

High-Performance Communication Networks
Jean Walrand and Pravin Varaiya, both of the University of California, Berkeley

Meet your evolving needs with this in-depth presentation of emerging technologies used to build high-speed, high-performance communication networks. It explains how the converging telephone, data, and CATV technologies are combined into high-performance networks, and how to plan, manage, and control these networks. The perspectives of the communications engineer, the computer scientist, and the economist are combined to provide a system-level understanding of the core networking principles and technologies.

"The experience of reading this book is almost as if I had spent a three day weekend seminar in a general discussion with an internetworking and engineering expert."

Computer Networks: A Systems Approach
Larry L. Peterson, University of Arizona and Bruce S. Davie, Cisco Systems, Inc.

A systems-oriented view of computer network design that goes beyond current technology to help you grasp the underlying concepts and build a foundation for making sound network design decisions. By providing an understanding of the components of a network and a feel for how these components fit together, this book empowers you to design real networks that are both efficient and elegant. It uses the Internet to illustrate how network protocols are designed.

"I was pleased with both the depth and clarity of the book. This is a winning combination. The discussions start with the basics, motivate...through examples and descriptions of the actual problems being solved, and end up with lively discussions of some of the current open problems."

Jeffrey Kipnis, Ameritech

Gary Delp, IBM Corporation

Additional Titles of Interest

Introduction to Data Compression
Khalid Sayood, University of Nebraska, Lincoln
1996; hardcover; 475 pages; ISBN 1-55860-346-8; $69.95

Object-Relational DBMSs: The Next Great Wave
Michael Stonebraker, Informix—with Dorothy Moore
1996; paperback; 216 pages; ISBN 1-55860-397-2; $39.95

MORGAN KAUFMANN PUBLISHERS

15% Discount Available to all USENIX members Please reference code ALOG
New Editions, New Covers, and More

GNU's Not Unix

USE 'GNU

The Free Software Foundation presents new editions of three of our bestselling manuals
Now with full-color covers and lay-flat binding

GNU Emacs Manual
12th edition, Updated for Emacs Version 19.33
By Richard M. Stallman
12th edition August 1996
506 pages, with reference card
ISBN 1-882114-05-1
$25.00

Using and Porting GNU CC
for Version 2.7.2
By Richard M. Stallman
August 1996
512 pages
ISBN 1-882114-36-1
$50.00

Debugging with GDB
for GDB Version 4.16
by Richard M. Stallman & Cygnus Support
September 1996
194 pages, with reference card
ISBN 1-882114-09-4
$20.00

Also available:
SOURCE 8 CD-ROM
Binaries 3 CD-ROM

Free Software Foundation

+1-617-542-5942
Weekdays 9am-5pm EST
Fax: +1-617-542-2652
or harass your local bookseller

59 Temple Place
Suite 330
Boston, MA 02111

Electronic orders:
fsforder@gnu.ai.mit.edu
For an online order sheet:
http://www.gnu.ai.mit.edu
FINALLY...
Cross-Platform Scheduling/Calendaring
that's as easy to administer
as it is easy to use!

SYNCHRONIZE™

Synchronize gives you the scheduling, task and resource management features your users want -- without the installation and administration headaches you may have encountered with other workgroup software.

Enterprise-wide scalability to support thousands of users
Today, employees work in groups that cross and transcend departmental boundaries and geographic locations. Synchronize's client/server architecture with distributed databases technology enables diverse workgroups in multiple time zones to work together as a tight, close-knit team...with peak efficiency and effective communication. And Synchronize scales transparently allowing you to easily add new users as your enterprise grows.

Real-time data access
Because Synchronize communicates directly across TCP/IP to transfer data, information access is virtually instantaneous. The latency associated with file-based or email dependent access methods and slower transport protocols is eliminated.

Cross-platform support
No scheduling software supports more platforms than Synchronize. Designed for cross-platform deployments, Synchronize runs on over 20 commercial UNIX and NT server platforms and includes desktop clients for Windows, Macintosh, X.11/Motif and ASCII terminals.

Take Synchronize for a test drive--FREE!
Can Synchronize solve your scheduling and task management problems...increase workgroup productivity...and enhance communication across your organization? To help you decide, we'd like to send you a FREE 30-Day Evaluation.

Try Synchronize Risk-Free
No Obligation--Nothing to Return
Call Today!
(800) 335-4988
info@crosswind.com

See Synchronize at:
LISA '96 - Booth #23
October 2-3, Chicago
UNIX EXPO - Booth #830
October 8-10, New York

CrossWind Technologies
Software for the Cooperative Workplace
http://www.crosswind.com
### CALENDAR OF EVENTS

This is a combined calendar of conferences, symposia, and standards meetings. If you have an event that you wish to publicize, please contact <login@usenix.org>. For complete USENIX conference and symposia listings see URL <http://www.usenix.org/events/usenixcal.html>. For an up-to-date, comprehensive, and easy-to-access information resource on the Internet, covering events all over the world, consult the WWW Virtual Library on Conferences at Fraunhofer-IAO. <http://www.rpd.net/Info/Conferences>

* = events sponsored by the USENIX Association.

<table>
<thead>
<tr>
<th>Year</th>
<th>Month</th>
<th>Event Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1996</td>
<td>December</td>
<td>IETF</td>
<td>San Jose, CA</td>
</tr>
<tr>
<td>1997</td>
<td>January</td>
<td>*USENIX, Anaheim, CA</td>
<td>6 - 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USELINUX Conference, Anaheim, CA</td>
<td>6 - 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IEEE Networks and Distributed Systems, Phoenix, AZ</td>
<td>12 - 15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>POPL '97</td>
<td>20 - 24</td>
</tr>
<tr>
<td></td>
<td>February</td>
<td>ISOC Symposium on Network &amp; Distributed Systems</td>
<td>San Diego, CA</td>
</tr>
<tr>
<td></td>
<td>March</td>
<td>ACM '97</td>
<td>San Jose, CA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SUG West</td>
<td>San Francisco, CA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UniForum</td>
<td>San Francisco, CA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFP '97</td>
<td>Burlingame, CA</td>
</tr>
<tr>
<td></td>
<td>April</td>
<td>ACM '97</td>
<td>San Jose, CA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IETF, Memphis, TN</td>
<td>7 - 11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>World Interop '97, Singapore</td>
<td>7 - 11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SANS, Baltimore, MD</td>
<td>21 - 26</td>
</tr>
<tr>
<td></td>
<td>May</td>
<td>HotOS-VI</td>
<td>Chatham, MA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>International Conference on Distributed Computing Systems</td>
<td>27 - 30</td>
</tr>
<tr>
<td></td>
<td>June</td>
<td>SIGGRAPH '97</td>
<td>Los Angeles, CA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IETF</td>
<td>Munich, Germany</td>
</tr>
<tr>
<td></td>
<td>July</td>
<td>USENIX Security</td>
<td>San Antonio, TX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>*USENIX, New Orleans, LA</td>
<td>21 - 29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SIGGRAPH '98</td>
<td>Orlando, FL</td>
</tr>
<tr>
<td></td>
<td>August</td>
<td>USENIX Windows/NT Workshop</td>
<td>Seattle, WA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Large Scale Systems Administration of NT, Seattle, WA</td>
<td>7 - 11</td>
</tr>
<tr>
<td></td>
<td>October</td>
<td>ACM SOSP</td>
<td>St. Malo, France</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OOPSLA '97</td>
<td>Santa Barbara, CA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>*WITS, Monterey, CA</td>
<td>8 - 11</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>POPL '98</td>
<td>St. Petersburg Beach, FL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7th USENIX Security</td>
<td>San Antonio, TX</td>
</tr>
<tr>
<td></td>
<td>June</td>
<td>USENIX, New Orleans, LA</td>
<td>15 - 19</td>
</tr>
<tr>
<td></td>
<td>July</td>
<td>SIGGRAPH '98</td>
<td>Orlando, FL</td>
</tr>
<tr>
<td></td>
<td>October</td>
<td>OOPSLA '98</td>
<td>18 - 22</td>
</tr>
<tr>
<td></td>
<td>December</td>
<td>LISA '98</td>
<td>Boston, MA</td>
</tr>
</tbody>
</table>

**Note:** The information is from the USENIX Association and the USENIX Virtual Library on Conferences.
The Andrew User Interface System has been a part of the campus computing environment of Carnegie Mellon University since 1986 and has continuously evolved to remain at the forefront of multimedia information technology.

The Andrew User Interface System is among the few software packages available which turns a Unix computer into a user-friendly workstation. The system offers simple, menu-driven interfaces for electronic mail, text, spreadsheets, equations, graphics, animations and images. Moreover, the interfaces work together to allow one to "mix and match" this large array of document types into single files.

As a forerunner in compound document architectures, the Andrew Consortium maintains and extends the Andrew User Interface System, a portable, extensible set of compound document applications for X11. These include "ez"—the word processor— and various objects which can be embedded and can also serve as standalone applications: spreadsheet, drawing editor, system monitoring tool, and an editor-based shell interface.

AMS, the Andrew Message System, provides a MIME-compatible, multimedia interface to mail and bulletin boards. AMS features include authentication, return receipts, automatic mail sorting, vote collection and tabulation, enclosures, audit trails of related messages, and subscription management.

Underlying Andrew applications and object editors is ATK, the Andrew Toolkit architecture, which supports the creation of new tools. Like most user interface toolkits, ATK provides a library of objects and supports sharing of the screen between objects. ATK additionally supports sharing file storage among objects, cut/paste across windows, an application construction environment, an extension language, and printing.

Participate in the development of the Andrew Consortium!
Utilize the latest advances by our technical staff, and undertake commercial exploitation with the active cooperation of the developers. Memberships supporting the work of the Consortium are offered at four grant levels:
- Participating
- Full
- Contributing
- Associate
You are cordially invited to join us!
Email <info-andrew-request@andrew.cmu.edu> to discuss the opportunities that the Consortium has to offer you.