inside:

CONFERENCE REPORTS
Conference on Domain Specific Languages
LISA '97

SAGE NEWS & FEATURES
Auditing: The Ugly Duckling of Computers
On Reliability – What About Yourself?
ToolMan: Upcoming Tools and Analyzing Paths

STANDARDS REPORTS
Whither POSIX?

BOOK REVIEWS
UNIX Power Tools, Cisco Routers

USENIX NEWS
1998 Board of Directors Election
Letter from the President
USA Computing Olympiad Team Scores Again
There's Gold in Good Works

ANNOUNCEMENTS & CALLS
COOTS '98 Tutorial Program
Electronic Commerce Workshop
1st International SANE Conference
LISA '98

features:

Musings
by Rik Farrow

Interview with Bill Cheswick
by Rob Kolstad

The ABCs of TPCs and NT Scalability, II
by Neil Gunther

Using Java
by Rik Farrow

Using C++ as a Better C
by Glen McCluskey
### up coming events

<table>
<thead>
<tr>
<th>Event</th>
<th>WHEN</th>
<th>WHERE</th>
<th>WHO program chair</th>
</tr>
</thead>
<tbody>
<tr>
<td>4th USENIX Conference on Object-Oriented Technologies and Systems (COOTS)</td>
<td>April 27-30/98</td>
<td>Santa Fe, NM</td>
<td>Joe Sventek, Hewlett-Packard</td>
</tr>
<tr>
<td><strong>DEADLINES</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Papers</td>
<td>March 17/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3rd USENIX Workshop on Electronic Commerce</td>
<td>August 31-Sept. 3/98</td>
<td>Boston, MA</td>
<td>Bennet Yee, Program Chair</td>
</tr>
<tr>
<td><strong>DEADLINES</strong></td>
<td></td>
<td></td>
<td>Dan Geer, Public Key Infrastructure Session Coordinator</td>
</tr>
<tr>
<td>Extended Abstracts</td>
<td>March 6/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notifications to Authors</td>
<td>April 17/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Papers</td>
<td>July 21/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5th Annual Tcl/Tk Conference</td>
<td>September 14-18/98</td>
<td>San Diego, CA</td>
<td>Don Libes, NIST</td>
</tr>
<tr>
<td><strong>DEADLINES</strong></td>
<td></td>
<td></td>
<td>Michael J. McLennan, Bell Labs</td>
</tr>
<tr>
<td>Paper Submissions</td>
<td>April 8/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notification to Authors</td>
<td>May 11/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Papers</td>
<td>July 28/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>USENIX Annual Technical Conference</td>
<td>June 15-19/98</td>
<td>New Orleans, LA</td>
<td>Fred Douglass, Program Chair AT&amp;T Labs</td>
</tr>
<tr>
<td><strong>DEADLINES</strong></td>
<td></td>
<td></td>
<td>Clem Cole, IT Coordinator, Digital Equipment Corporation</td>
</tr>
<tr>
<td>Full Papers</td>
<td>March 30/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Papers</td>
<td>April 27/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6th Annual Systems Administration Conference (LISA '98)</td>
<td>December 6-11/98</td>
<td>Boston, MA</td>
<td>Xev Gittler, Program Co-chair</td>
</tr>
<tr>
<td><strong>DEADLINES</strong></td>
<td></td>
<td></td>
<td>Lehman Brothers, Rob Koistad, Program Co-chair</td>
</tr>
<tr>
<td>Extended Invited Talks Notification to Authors</td>
<td>June 23/98</td>
<td></td>
<td>BSDI, Inc.</td>
</tr>
<tr>
<td>Final Papers</td>
<td>October 16/98</td>
<td></td>
<td>Netscape</td>
</tr>
<tr>
<td>Paper Submissions</td>
<td>June 23/98</td>
<td></td>
<td>Pat Wilson, IT Coordinator</td>
</tr>
<tr>
<td>Notification to Authors</td>
<td>July 21/98</td>
<td></td>
<td>Dartmouth College</td>
</tr>
<tr>
<td>Final Papers</td>
<td>October 16/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2nd USENIX Windows NT Symposium</td>
<td>August 3-5/98</td>
<td>Seattle, WA</td>
<td>Thorsten von Eicken, Cornell University</td>
</tr>
<tr>
<td><strong>DEADLINES</strong></td>
<td></td>
<td></td>
<td>Susan Owicki, Intertrust, Inc.</td>
</tr>
<tr>
<td>Paper Submissions</td>
<td>March 3/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Break-Out Proposals</td>
<td>March 20/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notifications of Acceptance</td>
<td>April 3/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Papers</td>
<td>June 19/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Large Installation System Administration of Windows NT Conference</td>
<td>August 5-7/98</td>
<td>Seattle, WA</td>
<td>Remy Evard, Argoonne National Laboratory</td>
</tr>
<tr>
<td><strong>DEADLINES</strong></td>
<td></td>
<td></td>
<td>Ian Reddy, Simon Fraser University</td>
</tr>
<tr>
<td>Paper Submissions</td>
<td>March 3/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>All Other Submissions</td>
<td>March 10/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notification to Authors</td>
<td>March 31/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Papers</td>
<td>June 18/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DEADLINES</strong></td>
<td></td>
<td></td>
<td>Paul Leach, Microsoft</td>
</tr>
<tr>
<td>Paper Submissions</td>
<td>July 28/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notification to Authors</td>
<td>October 13/98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Papers</td>
<td>January 6/99</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IN THIS ISSUE . . .

LETTERS TO THE EDITOR
3 In Response to Lee Damon
   from Greg Maple, from Jon Williams
4 Erratum
   from Alan Perry
   To Rik Farrow
   from Mick Carberry

CONFERENCE REPORTS
5 Conference on Domain-Specific Languages
16 Eleventh Systems Administration Conference (LISA ’97)

SAGE NEWS AND FEATURES
31 Worth Repeating
   by Tina Darmohray
32 President’s Letter
   by Hal Miller
33 New Editor for Short Topics Series
34 Auditing: The Ugly Duckling of Computers
   by Phil Cox
38 On Reliability – What About Yourself?
   by John Sellens
40 ToolMan: Upcoming Tools; Analyzing Paths
   by Daniel E. Singer

FEATURES
44 Musings
   by Rik Farrow
47 Interview with Bill Cheswick
   by Rob Kolstad
50 The ABCs of TPCs and NT Scalability, II
   by Neil Gunther
55 Using Java
   by Rik Farrow
57 Using C++ as a Better C
   by Glen McCluskey

STANDARDS REPORTS
60 An Update on Standards Relevant to USENIX Members
   by Nicholas M. Stoughton

BOOK REVIEWS
64 The Bookworm
   by Peter H. Salus
66 UNIX Power Tools, Second Edition
   Reviewed by Reginald Beardsley
66 Managing IP Networks with Cisco Routers
   Reviewed by Nick Christenson
67 Cisco TCP/IP Routing Professional Reference
   Reviewed by Nick Christenson

USENIX NEWS
69 1998 Election for Board of Directors
   by Ellie Young
69 Board Meeting Summary
   by Ellie Young
71 Letter from the President
   by Andrew Hume
72 USA Team Scores Gold
   by Rob Kolstad
72 There’s Gold in Good Works: A Report on USENIX Support of Worthwhile Projects
   by Cynthia Deno
75 Twenty Years Ago in ;login:
   by Peter H. Salus
76 Thanks to Our Volunteers
   by Ellie Young

ANNOUNCEMENTS & CALLS
77 COOTS ’98
78 3rd USENIX Workshop on Electronic Commerce
80 LISA ’98
82 SANE ’98
86 LOCAL USERS GROUPS
88 motd
   by Rob Kolstad
Welcome to a new year of ;login:. You may notice that with this issue the internal order has changed a bit: USENIX News has moved from the front to the back. This decision was based on the realities of print production. News items tend to be the last material to come together, and you can’t lay out the rest of a magazine if the first section isn’t ready! Don’t miss important information in this issue’s News section about the 1998 election for the USENIX Board of Directors.

A “dose of reality” is prevalent in this issue – editor Rob Kolstad resolves in his “motd” to make sure that his business contacts are “dealing with reality and not promises, hopes, or dreams.” Other authors seem to have picked up the theme and want to share with you their recommendations for grounding decisions and behavior in the concrete. (Of course, practical features, such as “Using Java,” “ToolMan,” and “Using C++ as a Better C,” are ;login:’s stock in trade.) Neil Gunther, continuing his series of articles comparing UNIX and NT scalability, explains how misuse of a benchmarking technique can distort reality. In the SAGE section, Phil Cox points out that without good system auditing, the chances of discovering what actually has happened in a security incident are nil, and he provides excellent guidance on how to begin auditing. And John Sellens discusses how system administrators can act more reliably by considering one of the most important realities of their environment – the people they work with.

This issue also features an enjoyable interview with Bill Cheswick and reports from two USENIX conferences held in October 1997. Andrew J. Forrest has provided thorough and informative summaries of the sessions at the Conference on Domain-Specific Languages, and several attendees have contributed reports on LISA ‘97. We hope you’ll like the photo portion of the LISA reports too.

Finally, we’re proud to include a news article about the generous level of support USENIX provides for a variety of worthwhile causes. Find out how your Association makes a difference “out there.”
Two Letters in Response to Lee Damon

[WWWhither(ing) Internet, December, 1997]

Speaking as one of those ex-ARPA research contract guys you applaud, let me say that I couldn’t disagree with you more. One of the whole reasons we worked on the project was to incrementally add accessibility. In the beginning, that was limited to new TIP and TAC nodes, then to variant hosts, then various byte and word order weirdnesses, then to encapsulated protocols. The point was to increase access, and not just for so-called “serious research,” but for contractors, the military, then for schools, etc. Although no one ever talked about a node in every home, that was a failure to project on our part, not something outside the envelope.

The access you complain about sounds an awful lot like the complaining that occurred every time a new load of the clueless crowd hit the nets over the last ten years. One surprising thing about those folks…they all eventually either got a clue, or they gave up and went away. Every time one of them went away, it was not a success for technical Darwinism, but instead a failure of the interfaces and protocols we had built to support them.

So what if the current users tend to view the Web as the whole net? In terms of greatest utility, they’re right. We’re suffering under a USENET not built to scale well, hideous old protocol interfaces, and a software/interface elitism that would stun a hieratic priest.

Your more serious charge is that users will not find sufficient utility and value in the Web to justify an ongoing dollar outlay and time commitment to use it. I hope you’re wrong. There’s not a lot of data out there right now, but the survey data I have seen tend to show that a large cross-section of net users tend to find continuing value, and that although there is a drop-off in use, it’s not nearly as steep as the drop-off in other acquired skills, like bowling or scuba diving.

To hope that the masses will find so little value in a technology that promises egalitarian access to an unprecedented depth of knowledge, history, scholarship, and entertainment that they’ll drop it and go back to watching TV is horrendous, and you ought to be ashamed of yourself for such blatant technoclassism.

For those of us that embrace this change in our user community, the mission should, by now, be clear. It’s our job to make it easier and better, not harder. If people think that the Web is the Internet, then it’s our job to make all those resources available through the Web or Web-like interfaces. If the accepted navigation tool proves to be a thin client with a remote control, then it’s our job to understand that and take it into account when designing resources, not to put up a “come back when you have a clue” sign and hide in net nostalgia.

It’s my personal hope that Joe Six-Pack and his kind will find themselves drawn into a new world of access and resources, not to drive them away. Who knows, some Joe Six-Pack might just be another Rembrandt or even a Lee Damon, just waiting to get turned on to what’s possible out there. Your approach is a sure method to save the technopriesthood, but isn’t it possible that it’s time for that particular religion to fade away?

Greg Maples
<greg@clari.net>

I won’t even try to argue that the Internet is currently caught up in a flurry of media hype even worse than Clear Pepsi, and we all know how well that went, but to suggest that the Internet is “the fad of the decade,” similar to the “CB radio in the 70s” and destined to “collapse under its own weight” is just as myopic as the vision of the supposed neophytes to the Internet who believe that the Web is All There Is.

As a result of all this hype, much of the image that the general public has of the Internet can be broken down into four basic types. First, “The Internet is a smut-ridden cesspool of filth, populated entirely by furry-fisted geeks.” While it is true that a large portion of what we see on the search engines is from the e-sex industry, it is not the majority, just the most visible and most easily criticized. Furthermore, the e-sex community has brought about as many, if not more, advances in the realm of online commerce than any other institution, so their presence, however seedy, has benefited the business community as a whole.

Second, “The Internet is merely a fad, destined to go the way of the hula hoop.” This view comes mainly from those who have little actual experience with the technology. Anyone who had spent even a small amount of time looking at the history of the Internet would know that above and beyond the thousands logging in daily for the first time, there are tens of thousands of us who have been in that quaint small town for several years. I have been a quiet resident for over four years now, and I know that there are others who have been there even longer. How often do you get to meet the founders of a “small town” of twenty million people? Not very often, and on the Internet, many of them will actually stop and chat. With the sheer numbers alone, the Internet has surpassed any of the fads listed without even mentioning the die-hard core who were proselytizing this land of milk and honey years ago.
Third, “The Internet contains only fluff.” This one, I can speak to personally. I have personally located pages upon pages of information that would not have been available to me through other means, including research on my favorite author (Bruce Chatwin), a relatively unknown classical composer named Pavel Haas whose brilliant career was cut short by the senseless Nazi slaughter, the friendship, support, and knowledge that has helped me keep my marriage together, and countless pages of information on health issues. The fact that the Internet provides access to everyone to publish their thoughts is NOT a detriment! The ease at which individuals can publish their information, no matter how seemingly trivial, focused, weird, or pointless, gives us access to a body of knowledge greater than any physical library in history.

Finally, and most annoying to me, “The Internet was great in the good old days, but all these idiot newbies have ruined it.” It is in this statement that the Internet shows its academic roots. Despite what I’ve said, there are those who have been in my town for many years, those whose heads are so filled with the grandeur of their private empire and the glories of their teaspoon Gardens of Eden that they have collapsed into the same xenophobia and isolationism that plagues many rapidly growing communities. To this small group, the media spotlight is a menace because with it comes “Them,” that nameless mass against whom all of us have fought at one point or another, without realizing that without Them, the Internet wouldn’t have nearly the resources, vibrancy, or diversity that it has today, or, more importantly, that we in that small town of the Internet have been viewed as Them by those we wish to keep out.

As frightening as it seems, the broad acceptance of the Internet means an acceptance, and eventually demystification, of our trade. For many, many years, we have been viewed in both the eyes of the world and of ourselves as wizards, possessors of arcane knowledge too strange, too difficult to be grasped by mere mortals. With every newbie, with each coin in AOL’s cup, another person becomes a part of that inner circle and our power is that much more diminished. It’s no wonder that Lee of the Arcane Hat hopes for the Internet to “collapse under its own weight” and become once again “the domain of old timers and the few clued individuals who have discovered that the Internet is much more than just a Web and some Usenet posts.” I, for one, hope that never happens.

Jon Williams
<dragon@revealed.net>

Erratum

Not sure if I should be sending this to you guys, but what the heck . . . .

There is an incorrect reference on page 57 of the special ;login: issue on Windows NT [November 1997], though I am not sure if the error is in the report or in what was presented.

The summary of Michael Frederick’s “Utilizing Low-Level NTFS Disk Structures for Rapid Disk Searching” talk lists two references, but the first reference is incorrect. The reference should be “Inside the Windows NT File System,” not “Inside Windows NT,” both of which were written by Helen Custer.

Though the book is pretty good, as someone who has been trying to use a file system book as a guide to implement NTFS under NetBSD, I think it is missing a bit more than just “the numbers.”

Alan Perry
<alanp@phcnet.com>

Rik Farrow responds:

The fault was mine. I was unaware of the NT filesystem book and had the title changed to the book I knew that Helen Custer had written.

Mea culpa.

To: Rik Farrow

I enjoyed reading Musings in the December ’97 ;login: but would like to make one correction.

When IBM released the first PC in an uncharacteristic spirit of openness, it published the PC Technical Reference which provided all the hardware specifications and a BIOS listing. There was no need to reverse engineer the BIOS but numerous copyright lawsuits were filed or threatened against BIOS clones. Phoenix, AMI and others bypassed copyright infringement by sprinkling NOPs throughout their firmware.

I’m writing this just after Microsoft lost the Internet Explorer anti-trust case. Another case of boundaries is Microsoft’s insistence that all PCs sold contain a copy of one of their operating systems leaving the customer with no choice in the matter.

Mick Carberry
<carberry@toronto.cbc.ca>

Rik Farrow responds:

Thanks for your comments on the December Musings. I really did believe that the code was reverse engineered on a functionality basis using a clean room approach, not simply copied with NOPs added. In my own experience with copyright law (as a writer, not a programmer) only 10% of the original material can remain or the copyright has been violated. You then appear to be suggesting that 91% of the Phoenix BIOS was NOPs. NOPs are fast, but . . .
Conference on Domain-Specific Languages

SANTA BARBARA, CA
October 15-17, 1997

Summaries by Andrew J. Forrest

OVERVIEW
USENIX held its first ever conference on the subject of Domain-Specific Languages (DSLs). The purpose of the conference was to bring together people who are interested in the idea that programming languages are first-class tools to exploit in the creation of software, and that the development of problem-appropriate computer languages is the basis for a valuable approach to software engineering.

Although this conference was organized into seven sessions, each with a specific focus, a number of themes emerged, cutting across sessions and recurring in many presentations.

Domain-Specific Languages – The Ultimate Abstraction
This theme holds that Domain-Specific Languages represent direct support for key abstractions in a programming domain. A DSL can represent an abstraction in a way that offers advantages when compared with other abstraction approaches, such as the use of libraries.

Language as a First-Class Tool
This theme presents language as a fundamental tool; it revolves around the idea that in many circumstances it is appropriate to create a new language rather than rely on the less specialized features of an existing general-purpose programming language.

Domain Analysis and Design for DSLs
A key step in the creation of a DSL, domain analysis is done to varying degrees of formality. The main question is “How does one pick the appropriate abstractions for the DSL to contain?” Is there a process for this? What is the relationship between the language designer and the domain expert(s)? Does the domain already have an accepted notation that suggests itself as a syntax for the DSL?

DSL Implementation Framework
This theme examines the various means by which a DSL can be implemented. Should a DSL be embedded in a larger, more general-purpose language (GPL)? Should it be implemented via a pre-processor? Or should an entirely separate implementation be developed?

DSLs and Rapid Prototyping
This theme occurs in two forms: DSLs support rapid prototyping because they tend to operate at a high level of abstraction; and rapid prototyping supports DSL creation by facilitating iterative design of the language.

Compiler Support and Tools for DSLs
Because DSLs are created more frequently than full GPLs, and because they may be changed more frequently, the need for compiler-compiler and other translator tools is greater with DSLs than with GPLs. It seems that the needs of DSL creators could influence compiler construction technology toward such benefits as more debugging, better debugging, support for specifying semantics, and more visualization.

Advantages of DSLs
Finally, there are many, many advantages to DSLs, such as notational convenience, certain type checking and global optimization opportunities, the ability to make additional safety guarantees, the potential for domain experts to program, the possibility of a variety of analysis, and the ability to capture an abstraction, thereby serving as an example of reuse.

Regarding the conference as a whole: a very high proportion of attendees was...
present for the entire conference despite the pleasant weather and beachfront venue. Interest in the BOFs was so great that initial plans to run them concurrently were shelved in favor of a consecutive schedule, and the papers presented were of such uniformly high quality that the program committee was unable to single out one for special commendation!

Overall, I believe the conference was a great success and would not be at all surprised if USENIX were to repeat it at some point in the future. See you there!

KEYNOTE ADDRESS

The Promise of Domain-Specific Languages
Paul Hudak, Yale University
Department of Computer Science

Domain-Specific Languages: Some Definitions
With help from a motley crew of animated agents residing in his laptop computer, Paul Hudak began his entertaining and thoughtful keynote address on the promise of Domain-Specific Languages. Hudak offered a framework for thinking about and working with DSLs, starting with a definition of the term itself: “A programming language tailored specifically for an application domain: it is not general purpose but rather captures precisely the semantics of the domain, no more and no less.” He quickly followed with a definition of “application domain,” which he accomplished by way of example, citing simulation, lexing and parsing, CAD/CAM, hardware description, text/pattern matching, computer music, and database queries, among others. To add to the list, Hudak said, one merely needs to reflect on the question: “How many papers have you seen with a title such as XXX: A Language for YYY?”

Hudak also offered a second definition of a DSL: “the ultimate abstraction of an application domain; a language that you can teach to the intended user in less than a day.” This definition relies on the observation that the intended user of a DSL is probably already well versed in the semantics of its domain and needs only a suitable notation with which to program.

In case you think DSLs are something new, you could ponder the list of popular and successful DSLs and their domains that Hudak presented:
- Lex and Yacc (for program lexing and parsing)
- Perl (for text/file manipulation/scripting)
- VHDL (for hardware description)
- TeX and LaTeX (for document layout)
- HTML/SGML (for document markup)
- Postscript (for low-level graphics)
- OpenGL (for high-level 3D graphics)
- TCL/Tk (for GUI scripting)
- Macromedia Director (for multimedia design)
- Prolog (for logic)
- Mathematica/Maple (for symbolic computation)
- AutoLisp/AutoCAD (for CAD)
- emacs Lisp (for editing)
- Excel Macro Language (for things never intended)

Advantages and Disadvantages of the DSL Approach
Chief among the advantages of the DSL approach to software development is higher programmer productivity because programs written in a DSL tend to be more concise, quicker to write and maintain, as well as easier to (automatically) reason about. Furthermore, although it sounds oxymoronic, DSL programs can sometimes be written by nonprogrammers. Hudak observed that these motivators are the very ones that drove the adoption of high-level general-purpose languages in the first place!

Of course, the DSL approach is not without challenges, too: performance may be poor because high-level languages are sometimes inefficient; there may be unacceptable start-up costs associated with the development of a DSL; a “Tower of Babel” may result if every domain acquires a specific language; the temptation to add features incrementally to a DSL can lead to bloat; and perhaps most important, designing and implementing languages (well) is a very hard task typically requiring two to five years for a new one. All of these issues represent possible obstacles that we need to overcome in order to more readily enjoy the benefits of DSLs.

A Recommended Approach to DSL Development
Given his experience with implementing and using a number of DSLs, Hudak distilled these recommendations for building software with a DSL:
- Choose your domain.
- Design a DSL that accurately and effectively captures the domain semantics.
- Concentrate on the semantics. Don’t let performance dominate design. Try to keep the end-user in mind at all times and to keep things as simple as possible.
- Prototype your design; refine and iterate. Also build SW tools to support the DSL.
- Develop applications (domain instances) using the DSL infrastructure.
- Success equals a happy customer!

Hudak observed that, although syntax and semantics are well treated in current and prior DSL development, tools often receive short shrift.

The Embedded DSL: An Implementation Approach
To overcome the weakness in tool support and to address some of the earlier noted disadvantages, Hudak advocated an approach to DSL development whereby the DSL is embedded in an existing, more
As advantages of the DSEL approach, Hudak cited rapid DSL design, increased changeability, familiar look and feel, reuse of infrastructure, a reduction in language bloat, and because whatever formal methods are applicable to the host language are applicable to the DSEL, the possibility of using algebraic/denotational semantics.

A DSL tends to share the same limits and face the same problems as the underlying host language, so Hudak cautioned that the choice of host language should be made after the abstract design of the DSL so that it can be made based primarily (if not entirely) on its suitability for hosting the particular DSL rather than on some other considerations.

**The Lightweight DSL: An Implementation Refinement**

Hudak introduced a refinement to the embedding approach called the Lightweight DSEL, which is a “pure” embedding. He noted that this approach requires a fairly powerful base language, one with higher-order functions, automatic memory management, syntactic extensions, flexible evaluation, and a flexible type system. In fact, in his experience implementing DSLs in Haskell, Hudak has made significant use of the higher-order functions, lazy evaluation, type classes, monads, and infix syntax features of Haskell. According to Hudak, the definitive way to embed a DSL within Haskell is to treat the DSL as an abstract data type for which an implicit interpreter represents the semantics. In this way, equational reasoning can be used to verify key algebraic properties of the DSL.

**Conclusion**

Hudak concluded by saying that DSLs are a good thing. Embedded DSLs and lightweight DSELS can be good things, but we need more and better tools to help with the design and implementation of DSLs. He also said that there should be a shift in emphasis in tool design from syntax to semantics, that the science of computer science has a role to play here. Algebraic/denotational semantics, modular interpreters, modular program execution tools, extensible type systems, and program transformation/partial evaluation were all mentioned as fruitful areas for this work.

Hudak never explicitly addressed the teaser that introduced his abstract for the keynote address, namely, “Are domain-specific languages (DSLs) the long-awaited silver bullet of software engineering?” However, he did argue cogently that DSLs have value and that embedding can be an excellent approach to the implementation of DSLs. In addition, he framed the entire conference with his definitions of a DSL and his worthwhile recommendations for the design and implementation of DSLs.

To see more, consult his official and personal home pages:

- [http://www.cs.yale.edu/HTML/YALE/CS/homepage/people/faculty.html#hudak](http://www.cs.yale.edu/HTML/YALE/CS/homepage/people/faculty.html#hudak)

---

**Refereed Papers**

**Session: Domain-Specific Language Design**

Each of the three presenters in this session introduced a problem domain, characterized some of its unique aspects, and described the design goals of an appropriate solution. In each case, the author showed how insight progressed to design and related the language design to the implementation architecture.

**Service Combinators for Web Computing**

Luca Cardelli, Digital Equipment Corporation, and Rowan Davies, Carnegie Mellon University

Cardelli and Davies based this work on observations about both the unique characteristics of the World Wide Web and the way it is used. On the Web, documents may be unavailable or slow to transfer. People compensate with interesting retrieval strategies involving multiple connections and preemptive behavior based on transfer rates. These strategies are not expressible via existing distributed paradigms, such as remote procedure calls. Cardelli and Davies therefore began with the view that the Web is a new and peculiar kind of computer, the “Berners-Lee Machine,” and set out to derive a language for programming it.

The result is a nascent language of service combinators for which a Java-based interpreter exists and a useful look at how one might go about designing a DSL. The language can be used to express typical human Web-browsing strategies because it allows direct references to the important characteristics of the Berners-Lee machine (including transfer rate). The authors offered a succinct formal semantics for the language and proved the correctness of certain optimizing transformations. This language of combinators was implemented as a set of composable Java functions rather than as a full-fledged language with its own unique
Domain Specific Languages for ad hoc Distributed Applications
Matthew Fuchs, Walt Disney Imagineering

The final paper presented in the DSL Design session examined the value of DSLs as intermediate glue between distributed "agents," be they computational or human. Fuchs observed that humans find binary data inconvenient and that many ASCII formats prove tricky for programs to parse. Yet distributed application components may need to interact with both software agents and human ones, and rather than construct each component with two interfaces, it would be nice to find a single format suitable for both.

Fuchs recommended creating a DSL to subsume both of these interfaces — a single language for communicating state, behavior, and sequence to both human and computer-based agents. To represent the language, Fuchs advocated using SGML or its simplified subset, XML, because they are suitable for human use (they can be displayed by a graphical interface) and are easily processed by programs, especially in the latter case. To explore the value of this idea, Fuchs showed how the game of bridge can be represented by an XML object, which is passed from player to player during the course of a game. At each turn, the string is extended, with data representing a new bid or card play, and processed by an agent (human or computer) representing a player.

Fuchs placed great value on the power of separating syntax and semantics in defining a DSL. Interestingly, Fuchs also observed that LL(1) languages are particularly well suited to use in this domain, especially for applications with a high degree of interactivity. This is because LL(1) languages permit top-down parsing, which means that a string in the language can be successfully parsed (i.e., the relevant production can be known) at any point during the string's construction.

Session: Experience Reports

Participants outlined their experiences with DSLs, what benefits they realized, what challenges they faced, and what advice they could offer to other potential DSL developers.

Experience with a Domain Specific Language for Form-based Services
David Atkins, Thomas Ball, Michael Benedikt, Glenn Bruns, Kenneth Cox, Peter Mataga, and Kenneth Rehor, Bell Laboratories, Lucent Technologies

MAWL is a DSL for creating device-independent, form-based services. Such services are characterized by data flows between the service and its users in a series of query/response interactions. A form is the abstraction that describes each interaction between the user and the service. This simple but powerful abstraction is the key to the numerous benefits provided by MAWL. Ball reported that MAWL enabled certain properties of a Web service to be verified at compile time, something that cannot be done in general for CGI scripts. Additionally, MAWL and its corresponding implementation architecture permit certain flexibilities, such as the creation of a standalone service (independent of a Web server), the use of a variety of implementation languages, and even the substitution of different user interface devices. Furthermore, because a declaration of each form's signature is available to the MAWL compiler, a functional Web-based stub can be generated automatically for the application, thereby permitting rapid evaluation of the service before extensive effort is invested in the various aspects of the eventual user interface. Because all of these benefits accrue directly or indirectly...
from the central form concept, it seems safe to say that the experience with MAWL amply demonstrates Hudak’s suggestion that a DSL is the “ultimate abstraction” in a domain.

Experience with a Language for Writing Coherence Protocols

Satish Chandra and James R. Larus, University of Wisconsin, Madison; Michael Dahlin, University of Texas, Austin; Bradley Richards, Vassar College; and Randolph Y. Wang and Thomas E. Anderson, University of California, Berkeley

A veritable gold mine of concrete advice to DSL designers and implementers, this experience report described a language called “Teapot,” which is for writing the coherence protocols found in Distributed Shared Memory (DSM) systems. The goal of the language is to eliminate a variety of programming errors by providing a language that is specific (and restricted) to the applicable domain as well as to obtain both an implementation of a protocol and a source for a protocol verifier from a single protocol description.

This report offered many suggestions to prospective DSL designers:

- A DSL should probably be as small as you can stand.
- The language should directly support programming scenarios that occur commonly in the domain.
- A DSL’s users should not need to know implementation details.
- Compiler optimizations should be explicitly specified and user-selectable.
- Provision of thread support should be considered from the outset.
- Be prepared to assist users in adopting your DSL; otherwise, natural inertia will preclude it (examples help greatly here).

Implementation techniques were also discussed, and implementers were encouraged to (re)use existing source code and tools wherever possible, in the latter case, by combining language processors (i.e., generating a high-level language source). Simple syntax with lexical hints and self-documenting source files should all be used as well. Finally, features of the target and implementation languages may prove valuable to use.

Session: Compiler Infrastructure for Domain-Specific Languages

This session examined language technologies that could provide benefits to DSL implementors. Two of the papers examined techniques that require opening up the language translator itself.

Lightweight Languages as Software Engineering Tools

Diomidis Spinellis, University of the Aegean, and V. Guruprasad, IBM T. J. Watson Research Center

Guruprasad showcased various advantages, disadvantages, and implementation techniques germane to the DSL arena by presenting a survey of representative DSL systems covering user interface specification, applications development process, text processing, multiparadigm programming, and language implementation. The principal advantage cited in this work is that the DSL reduces the semantic gap between specification and implementation, echoing once again the “ultimate abstraction” theme.

Disadvantages do exist, however, and they include a tendency for the ad hoc nature of DSLs to contribute to a lack of suitable skilled personnel, training materials, and appropriate tools; doubts about scalability; and, through proliferation, a computer language "tower of babel." Nevertheless, the overall experience described by Guruprasad has been favorable.
Typed Common Intermediate Format
Zhong Shao, Yale University

In order to facilitate both the ready generation of compilers for DSLs and the interoperability of DSLs and general-purpose languages, a common substrate is required. FLINT, a typed intermediate format that satisfies this requirement, is discussed by Shao in this paper. Translator implementation is simplified by FLINT because a reusable "back end" is provided. Another benefit of FLINT is that it enables code written in multiple languages to interoperate. This is because each language's translator can share the same back-end facilities (e.g., optimizers, verifiers, and generators) as well as runtime conventions (e.g., garbage collector, foreign function calling mechanisms). Finally, unlike other intermediate languages, FLINT is capable of supporting higher-order languages such as ML.

<http://flint.cs.yale.edu>
<http://flint.cs.yale.edu>

Incorporating Application Semantics and Control into Compilation
Dawson R. Engler, MIT Laboratory for Computer Science

What would happen if one stopped viewing one's compiler as a black box and opened it up, even just a little bit? What kinds of things could one do? Engler provided one answer to these questions with a paper describing MAGIK, a system that permits users to hook into the compilation process and provide transformations or verifications that are driven by application semantics and yet still benefit from the optimization phase of the original compiler. Examples of the kinds of transformations and verifications possible with MAGIK include verifying type safety between the format string and other arguments of the C language's printf; determining if system call return codes are examined and, if not, adding code to do so; enforcing adherence to "programming rules" such as restrictions to be observed when coding UNIX signal handlers; and partial evaluation in the context of RPC parameter marshalling.

<http://www.pdos.lcs.mit.edu/~engler/>

Code Composition as an Implementation Language for Compilers
James M. Sticknothen and Thomas Gross, Carnegie Mellon University

Code composition is a technique that promises speedy implementation of compilers that are capable of translating high-level or complex operations with both good quality and efficiency. Catacomb performs code composition through the interaction of a composition system with a compiler. The compiler partitions the source program into two sets of constructs, those for which code composition is offered and those for which it is not. For each construct that permits code composition, the compiler invokes the composition system. The composition system, in turn, processes code templates that are separate source-comprising code constructs and control constructs. The control constructs are "executed" by the composition system producing custom-generated code, which is then passed back to the compiler proper. Once under the compiler's control, the custom-generated code can be combined with the remaining code constructs and processed by all the further downstream processors (e.g., optimizers).

A Domain-Specific Language for Regular Sets of Strings and Trees
Nils Klarlund, AT&T Labs Research, and Michael I. Schwartzbach, University of Aarhus

Schwartzbach presented both FIDO, a high-level programming notation that concisely expresses regular sets of strings or trees, and a thorough analysis of the DSL experience as he and his colleagues see it. FIDO combines standard programming language concepts such as recursive data types, unification, implicit coercions, and subtyping with a variation of predicate logic called the Monadic Second-order Logic (M2L) on trees. M2L has proved very useful to Schwartzbach and his colleagues, but suffers from a tedious notation. FIDO corrects this.
Once a commitment to DSL creation is made, some reflection on the nature of the software problem and a domain analysis (formal or otherwise) takes place. The outcome of these steps is combined with a general knowledge of language concepts and language technology to create an implementation that, if everything goes well, provides relief from the original software problem. As with everything computational, some iteration may be necessary, in part because of imperfect execution of the earlier steps, but, surprisingly, also due to one more issue: the DSL's implementation and its effect on your software problem may actually lead to further insight in the domain! It is this feedback that yields a deeper understanding of the domain that may, additionally, prompt another iteration in the cycle.

Schwartzbach indicated that there is no substitute for domain experience in maximizing the likelihood of creating a successful DSL. Often, such experience will expose a repetitive or error-prone software activity that must be performed when solving problems in the domain. Removing this repetitive activity from the programmer's work and placing a solution to it within a language is a key impetus for the creation of a DSL.

Once a commitment to DSL creation is made, some reflection on the nature of the software problem and a domain analysis (formal or otherwise) takes place. The outcome of these steps is combined with a general knowledge of language concepts and language technology to create an implementation that, if everything goes well, provides relief from the original software problem. As with everything computational, some iteration may be necessary, in part because of imperfect execution of the earlier steps, but, surprisingly, also due to one more issue: the DSL's implementation and its effect on your software problem may actually lead to further insight in the domain! It is this feedback that yields a deeper understanding of the domain that may, additionally, prompt another iteration in the cycle.

### Evolution of a domain-specific language

- **domain**
- **experience** → **software problems**
- **reflection** → **language concepts**
- **knowledge** → **language technology**
- **implementation** → **relief** → **insight**
- **language**

---

**A Modular Monadic Action Semantics**

Keith Wansbrough and John Hamer, University of Auckland

If there had been an award for "Presentation with the Most Audience Participation," Hamer certainly would have won it. After entertaining the audience with an exercise in creating an origami frog, he presented some work done principally by Wansbrough on the fusing of Modular Monadic Semantics (MMS) with Action Semantics (AS) (yielding Modular Monadic Action Semantics). Action Semantics is popular for its highly readable notation, yet it is monolithic, supports only a fixed range of language concepts, and is somewhat difficult to employ in proving properties about a language or its programs. Modular Monadic Semantics is highly extensible due to excellent modularity and, because it is implemented in a functional programming language, its specifications can be directly executed. Modular Monadic Semantics is considered to be at a slightly lower level of abstraction than Action Semantics and, unfortunately, is also considered to be a bit challenging to use. Modular Monadic Action Semantics thus combines the best aspects of Action Semantics and Modular Monadic Semantics to provide an extensible, easy-to-write notation.

---

**Session: Case Studies and Frameworks**

This session discussed work in interesting problem domains. In the first two papers, case studies of DSL creation and use were presented. The third paper documented a survey and analysis of Architecture Description Languages that could provide the kind of domain understanding required for the creation of a new DSL.

**SHIFT and SMART-AHS: A Language for Hybrid System Engineering Modelling and Simulation**

Marco Antoniotti and Aleks Gollü, University of California, Berkeley

The domain in which SHIFT operates is that of Hybrid Systems Analysis and Modelling which can be likened to simulation of situations where both continuous and discrete phenomena are present and interact. The SHIFT language contains terminology, notation, and semantics straight from the domain, making it readily usable by the intended audience: control system engineers. Befitting a case study, SHIFT has received significant use. This paper describes the reimplementation of the Traffic Simulation framework, SMART-AHS, in SHIFT. Users reported a 50% reduction in the size of both libraries and projects and greater ability to reuse code. Credit is given in the evaluation of SHIFT to it containing the "right" abstractions, being somewhat restrictive compared to a GPL, and consequently requiring fewer "programming rules" than other systems. This last aspect is important because, as a DSL, SHIFT can enforce programming rules, so they become language rules and less of a cognitive load for developers.

<http://www.path.berkeley.edu/smart-ahs>

February 1998
Design and Semantics of Quantum: A Language to Control Resource Consumption in Distributed Computing
Luc Moreau, University of Southampton, and Christian Queinnec, Université de Paris

With the advent of mobile agents comes increased interest in the ability to control the resource consumption of computations either because one needs to pay for such consumption or because one is providing the resources for others to consume. The DSL Quantum was developed for users to specify and enforce resource consumption limits and patterns for distributed computing. Quantum is the synthesis of three ideas:

■ Quotas of energy can be associated with computations, and energy is being consumed during every evaluation step.
■ Asynchronous notifications inform (interested parties) of energy exhaustion or computation termination.
■ Mechanisms exist to transfer energy to or from computations; supplying more energy to a computation gives the right to continue the computation, and removing energy from a computation acts as energy-based preemption.

This paper contrasts Quantum with a broad variety of similar work in this field.

<http://diana.ecs.soton.ac.uk/~lavm/>

Domains of Concern in Software Architectures and Architecture Description Languages
Nenad Medvidovic and David S. Rosenblum, University of California, Irvine

Architecture Description Languages (ADLs) abound, but, as often happens when there is not yet agreement on what the problem is, these “solutions” differ widely, especially in their coverage of software architectural concerns. Medvidovic presented a comprehensive framework of architectural concerns and evaluated several major ADLs with respect to this framework. More understanding of the domain(s), how they relate to application domains, and what effect they should have on ADLs are required, noted Medvidovic.

Medvidovic’s framework showed the kind of analysis that one can imagine forming the basis for a new DSL; whether it is arrived at by virtue of a formal process or by intuition, at some point, an understanding of the domain in this depth becomes necessary. Watch this space for a DSL.


Session: Abstract Syntax Trees

This session concentrated on a fundamental building block of language tools: the Abstract Syntax Tree. Papers were presented that discussed superior tools and techniques for representing such trees, searching them, and manipulating them.

The Zephyr Abstract Syntax Description Language
Daniel C. Wang, Andrew W. Appel, Jeff L. Korn, and Christopher S. Serra, Princeton University

One area where DSLs shine is in the creation of a glue or interchange language that can serve as input to various automated program generators. Such meta-programming languages can aid in the integration of tools from diverse sources. The Zephyr Abstract Syntax Description Language (ASDL) was designed to provide a concise notation for describing abstract syntax trees. This notation is processed by several companion tools that generate data structure definitions and procedures in several target languages for converting ASTs to and from a standardized flat representation (pickles). Interoperation of compiler components is greatly facilitated by the capability to exchange ASTs easily. As a proof of concept, Wang described the use of the Zephyr ASDL to respecify the Stanford University Intermediate Format (SUIF) (among others), with impressive reduction in “program” size.

<http://www.cs.virginia.edu/zephyr/>
<http://www.cs.virginia.edu/zephyr/asdl.html>

ASTLOG: A Language for Examining Abstract Syntax Trees
Roger F. Crew, Microsoft Research

Crew described a clever and useful adaptation of Prolog for locating and analyzing complex syntactic constructs in program sources (as opposed to the lexical searching capabilities of tools such as Awk and grep). ASTLOG was inspired by Prolog and its implicit pattern-matching and backtracking capabilities. Program source, instead of being converted to a Prolog fact database, can be directly examined by predicates present in ASTLOG. This feature is the key to making this approach feasible for examining large programs. The principal consequence of the inclusion of these predicates is what Crew termed an inside-out functional programming style, which turns out to be particularly suitable for searching and pattern matching on parse-trees. Performance of the system on substantial real-world examples is comparable to the time taken for compilation.

<http://www.research.microsoft.com/~rfc/default.htm>
KHEPERA: A System for Rapid Implementation of Domain Specific Languages
Richard E. Faith, Lars S. Nyland, and Jan F. Prins, University of North Carolina, Chapel Hill

Check this out if you’re looking for support for DSL development: KHEPERA is a toolkit that creates DSL processors by generating source-to-source translators that rely on sophisticated tree-based analysis and manipulation to provide ease of implementation and superior debugging information. Systems built with KHEPERA not only provide support for debugging DSL translators themselves, but also support debugging the end-user’s DSL program. Faith started with a software problem in which repetitive work was being performed on custom translators for the DSL PROTEUS, and he was thus motivated to find an automated way of generating DSL translators. Such a system is advantageous because of the recognition that DSLs are likely to evolve. Such evolution implies frequent syntax and other translator-affecting changes. KHEPERA itself contains a DSL—the KHEPERA Transformation Language—that describes the fundamental tree matching and manipulation primitives required to specify a source-to-source translator.

<http://www.cs.unc.edu/~faith/khepera.html>

Session: Embedded Languages and Abstract Data Types

This session was the combination of two topics: how fully supporting a useful abstract data type may require a DSL, and what kinds of considerations one faces when using the embedded approach to implement a DSL. The papers covering the former topic showed two cases where supporting an abstract data type required features that aren’t available in general-purpose languages, and therefore a DSL was the best recourse. The latter topic’s papers, based on actual experiences, cataloged many of the choices (and their consequences) that are encountered when implementing a DSL as an embedded language.

DiSTIL: A Transformation Library for Data Structures
Yannis Smaragdakis and Don Batory, University of Texas, Austin

The domain for DiSTIL is that of complex container data structures. The authors argue that these data types should have uniform interfaces that allow the application and implementation to evolve independently. The implementation of DiSTIL was described as an extension of Microsoft’s Intentional Programming system, which is discussed in some detail within the paper. Furthermore, work described in this paper develops a metaprogramming framework above IP called "generational scoping," which the paper concludes might develop into a general-purpose set of primitives for describing program generators. In addition to describing the primary technical contribution surrounding DiSTIL, the paper enumerates many conclusions about the suitability of IP and its potential influence on DSL design, implementation, and use.

<http://www.cs.hmc.edu/~fleck/envision/envision.html>

Modelling Interactive 3D and Multimedia Animation with an Embedded Language
Conal Elliott, Microsoft Research

Fran is a DSL for describing and composing animations. A key insight behind Fran is that an animation can be thought of as a function of time. Elliott suggested that much animation work is frustrated by the level of detail required and the fundamental mismatch between the medium and the computer. For instance, computers operate in discrete steps, whereas time is continuous; or, for example, single-processing computers must be used to implement concurrent animations. The declarative nature of Fran alleviates these problems. Although Elliott’s talk centered on successively complex animation examples, the corresponding paper conveys a wealth of insight into design and implementation techniques other than direct programming relief accruing: easier collaboration and teaching and replication of research results. By codifying various “hard” parts of image processing into language primitives, Fleck hoped to get her colleagues to focus on images instead of programming. She also challenged DSL designers to bring the appropriate level of programming language abstraction (via a DSL) to this field, claiming that previous approaches are either too high or too low. The last half of this paper describes the particular aspects of computer vision that should be supported in a DSL and a proposed instance of this DSL named Envision.

<http://www.cs.hmc.edu/~fleck/envision/envision.html>

Programming Language Support for Digitized Images or The Monsters in the Closet
Daniel E. Stevenson and Margaret M. Fleck, University of Iowa

In this work, Fleck argues that language support in the form of a new abstract data type, the “sheet,” and various associated primitives can dramatically simplify the task of programmers writing in the field of computer vision (image understanding) algorithms. A host of benefits...
for domain-specific languages, particularly in its discussion of modelling and host-language embedding.


**A Special-Purpose Language for Picture Drawing**

Samuel Kamin and David Hyatt, University of Illinois, Urbana-Champaign

Kamin described an experiment in domain-specific language design and implementation. Inspired by PIC, Kamin reproduced a set of drawing primitives implemented within Standard ML. Kamin showed the design and evolution of his language, FPIC, carefully illuminating his design decisions at each step along the way. He also revealed the trade-offs inherent in his choice to use the embedded-language implementation approach. Kamin noted one interesting limitation of the embedded approach: complications can arise because the ordinary language has an environment distinct from that used by the embedded language's primitives. Nonetheless, Kamin concluded that the cost-benefit ratio of the embedded language approach is favorable.

<http://www-sal.cs.uiuc.edu/~kamin/fpic>

**INVITED TALKS**

**Synchronous Languages – An Experience in Domain-Specific Language Design**

Gérard Berry, École des Mines de Paris, Centre de Mathématiques Appliquées: INRIA, Projet Meije

Berry presented ideas about DSLs drawn from his experience in designing and using Esterel, a language for programming deterministic reactive systems (systems that execute indefinitely, reacting deterministically to asynchronous inputs). Berry’s inquiry was quite thorough in both its breadth and detail; however, a few key points can be selected for special mention.

When DSL designers wish to implement their language by the embedded approach, care must be taken to consider whether the host language is too rich or lacks a precise definition. This would complicate rigorous analysis.

For safety-critical applications, a sound mathematical foundation, and therefore a precise semantics, is an absolute necessity for the language. Automated analysis depends upon it.

There should be a compelling reason for introducing a new language. Either useful mathematical properties or the abstraction of complex computation is a suitable reason.

Sometimes the domain on which a designer is focused can be enlarged from the immediate application to a broader, more foundational one, in which case the resulting DSL can be more widely useful.

Orthogonality of language features should be supported.

It is the particular mathematical properties of a formalism and its usefulness, not just its mere existence, that ultimately determines the value of any formalism.

It is a myth that DSLs must be less efficient than general-purpose programming languages. Because DSLs tend to be small and may operate at quite high levels, sophisticated optimizations may be feasible. Or DSLs may perform functions that are impractical for a human, thereby increasing the scale at which a solution can be obtained or yielding efficient solutions without the cost of hand coding.

Berry’s conclusion: “language design never ends.”

For interested readers, much more information, including the latest Esterel compiler, can be had at <http://www.inria.fr/meije/esterel/esterel-eng.html>

**Intentional Programming – An Ecology for Abstractions**

Charles Simonyi, Chief Architect, Microsoft

Charles Simonyi presented an overview of Intentional Programming (IP), which is referred to in Microsoft papers variously as a process, a programming environment, and an ecology. From a DSL-design perspective, a novel aspect of IP is that it seeks to reduce or eliminate the rigidity of syntax in programming languages. Instead of a text-based syntax, intentional programs are created “preparsed” as trees of nodes (the intentions) aimed at capturing the programmer’s intent. Each node can be thought of as an object or component, possibly quite fine grained, that presents an appearance to the programmer and is capable of behavior. Both the appearance and the implementation of the behavior are permitted to change; however, the intention remains the same, as do its relationships with other intentions in the intentional program. Intentions can operate on other intentions to transform them to forms already understood by an IP system. Such intention-transforming intentions are called enzymes, in keeping with the biological metaphors.

Why program this way? The hope is that the adoption of IP will create an environment where the biological principles of evolution will be brought to bear on software componentry in the hopes that the “fittest” will prevail. Simonyi envisions a vast soup of intentions, each vying for a place in your programs.
IP needs more explanation than an invited talk (or its review) can provide; therefore, I strongly encourage interested parties to pursue this topic by visiting <http://www.research.microsoft.com/> and searching for “intentional programming” or perhaps just by starting with these two pages:

- The Death of Computer Languages, the Birth of Intentional Programming  
- Microsoft Research Intentional Programming  
  <http://www.research.microsoft.com/ip/default.htm>

Aspect-Oriented Programming – Improved Support for Separation of Concerns in Design and Implementation
Gregor Kiczales, Xerox Palo Alto Research Center

Concluding the conference was the privilege of hearing Gregor Kiczales, co-author of “The Art of the Metaobject Protocol.” He chose to talk about his most recent work, which he calls Aspect-Oriented Programming (AOP), in which another “domain” (really a view or slice) of programs is introduced. It leads to programs that are much more tolerant of changes in the requirements of runtime behavior and other systemic properties.

Starting from first principles, Kiczales reminded us how (in general) we build complex systems by first partitioning the system into cognitively manageable parts – the separation of concerns, or decomposition – and then by constructing and/or composing implementation elements in a way that is “suggested” by the decomposition. In this way, concerns that are known from the outset tend to appear localized in the implementation and permit relatively easy modification in the face of changing requirements. There is, however, a class of concerns for which this fails dramatically – systemic properties (e.g., runtime behavior). The interaction of software components whose structure was determined largely by static considerations gives rise to what Kiczales terms “emergent entities.” The recognition of these entities provides the key insight that motivates AOP: emergent entities are important, are difficult to manage classically, and require languages and tools for their explicit description and use.

Emergent entities are important because they can represent things like performance concerns, synchronization behaviors, memory usage, and replication properties. They are difficult to manage classically because accommodating any single one could involve changes to many components in a system. This arises because emergent entities involve the interaction of components and therefore cut across component boundaries.

Kiczales allows that there is frequently a way to refactor a design post hoc to accommodate changes in the runtime behavior requirements; however, what is needed is something more accommodating of such changes. Finally, emergent entities need language and tool support precisely because they do not (yet) appear explicitly in the implementation of a system.

The mechanism Kiczales proposes to manage emergent entities is the “aspect,” which he defines as a “modular unit of control over emergent entities.” He further proposes (and has implemented) “aspect languages” and their translators, “aspect weavers.” The translators work by taking a modular program and the expression of one or more aspects and weaving the codes together, yielding a program that reflects the original modular system, modified to account for the aspect. Through this technique, the aspect can be coded in a compact, local form and yet, through the actions of the aspect weaver, have a widely distributed effect on the implementation. Thus, AOP is a “modularity for that which was previously amodular” and is robust in the face of changes in the requirements for runtime behaviors and other systemic properties.

Slides from a similar talk are available at:  

BOFS

There were three Birds-of-a-Feather (BOF) sessions at DSL ‘97, all of which were arranged by interested parties before the conference began. In part, the BOFs were a great success because of this preparation, and this points the way to a potential approach for increasing the likelihood of having interesting and well-attended BOFs.

Musical Languages

Attendees of the Musical Languages BOF session were treated to a number of speakers, some human, some electromagnetic. The organizer, Tim Thompson, brought musical equipment and recordings with him, as did some of the other presenters, including Paul Hudak, the conference’s keynote speaker. Several music and sound composition systems were demonstrated and contrasted. These exposed themes such as the use of interactive or visual DSLs and their relationship(s) to text-based languages and the nature of the interaction between domain experts and the language developers.

Patenting Domain-Specific Languages

At first glance, having a BOF concerned with intellectual property at a Domain-Specific Language conference might strike one as unusual, and it is. But it was certainly useful and intriguing. Christa
Schwartz, a one-time software designer at Bell Labs and now a patent attorney, acquitted herself very well in delivering an interesting, realistic, and thought-provoking presentation about DSLs from the intellectual property perspective. Covering the background of patent law briefly, she went on to describe what elements are involved in both the decision to patent and also the process of patenting a Domain-Specific Language, its concepts and/or its implementation. Drawing on her interdisciplinary knowledge, Schwartz finished with an example in which she patented a language all DSLers should know: YACC.

Program Generators

This discussion began with Samuel Kamin’s proposal that program generators should be considered programming languages that have programs and program fragments as their intrinsic data types instead of the more usual characters, integers, and floating point numbers. These program fragments parameterize a generator and determine the range of possible output programs, effectively defining an application domain. Kamin further framed the discussion by suggesting some points to ponder when considering program generators:

- How can (or even should) the logic that generates the code and the “template” of the code be separated?
- To what extent can ideas and features from other languages and implementations (such as static type checking, recursion, and optimization) be applied to program generators?
- What kinds of languages work well as the target language of a program generator?
- What is a good way to organize, structure, or maintain a program generator?
- When should a program generator, rather than a full-fledged language or an embedded approach, be used to solve a problem?

The popularity of this BOF and the content of the discussion revealed substantial interest in program generators as a technique for creating parameterized solutions in certain problem domains.

Quotable Quotes from the DSL Conference

Satish Chandra: “Avoid frustrating potential users. Avoid potential frustrating users.”

Gerard Berry: “Language Design Never Ends” (just ask B.S.).

“0 is a beautiful number”

Charles Simonyi: “It’s better to have DSL Bugs than bugs bugs.”

“I don’t believe in simplicity. Simplicity is a trap.”

“Haskell – you will really have to commit to it before you don’t like it.”

“I always feel funny when I hear the word ‘simplicity.’”

Samuel Kamin: “Fn Langs: you either love ‘em or you don’t know what you’re doing.”

Gregor Kiczales: “You sound just like someone from PARC: you talk a long time about a simple problem.”

“Exception handling: many many powerful minds have gone there . . . and come back.”

Unknown limo driver: “What’s this Unisex LSD Conference?”

Eleventh Systems Administration Conference (LISA ’97)

SAN DIEGO, CA
October 26-31, 1997

KEYNOTE ADDRESS

Generation X in IT

Randy Johnson and Harris Kern, R&H Associates Inc.

Summary by Carolyn M. Hennings

Randy Johnson and Harris Kern spoke about the characteristics of a portion of today’s workforce referred to as Generation X and the impact it has on traditional IT departments. The challenge to existing IT departments is identifying the nature of the Generation X workforce, clarifying why these characteristics are potentially an issue, and determining how to manage the situation in the future.

In the early 1990s, industry labeled Generation X—persons born between 1964 and 1978—as “slackers”; however, most are entrepreneurial, like change and diversity, and are technically literate. In contrast, the traditional IT organization was built on control and discipline. As technology has moved away from a single machine to a networked computing model, the nature of the IT business has changed. The speakers noted that IT departments had historically relinquished control of personal computers and local area networks. IT management has come to the realization that these are essential elements of the success of mission-critical applications. As a result, there must be some control.

Johnson and Kern suggested IT management focus on the following areas:
Teamwork. Encourage people to work together and rely on individuals to do their jobs.

Communication. Improve communication within the organization and with the customer.

Involvement. Rather than direction from above, involve the team in decisions and planning.

People. Encourage a “can do, be smart” attitude with some discipline.

Process. Institute the minimum and sufficient processes to support the organization.

They suggested that this could be considered “creating Generation Y.” These people and relationships will be needed to build successful IT organizations. The IT department must become a true service organization. To accomplish this, the department must win back the responsibility for technology decisions, reculture the staff to support diversity and change, market and sell the services, train staff members, and focus on customer satisfaction.

The department must communicate within the IT organization and with customers. Defining architectures, standards, support agreements, and objectives will make great strides in this area. The definition and support of the infrastructure from the desktop to the network, data center, and operations is an essential step. Defining “production” and what it means to the customer in terms of reliability, availability, and serviceability goes a long way in opening communication and expectations.

System management processes with standards and procedures modified from the mainframe discipline are necessary steps. The speakers cautioned organizations against bureaucracy and suggested focusing on producing only “minimum and sufficient documentation.” Implementing deployment methodologies and processes was strongly encouraged, as well as developing tools for automating these processes.

**REFEREED PAPERS TRACK**

**Session: Monitoring**

Summaries by Bruce Alan Wynn

**Implementing a Generalized Tool for Network Monitoring**

Marcus J. Ranum, Kent Landfield, Mike Stolarchuk, Mark Sienkiewicz, Andrew Lambeth, and Eric Wall, Network Flight Recorder Inc.

Most network administrators realize that it is impossible to make a network unbreachable; the key to network security is to make your site more difficult than another so would-be intruders find easier pickings elsewhere.

In this presentation, Ranum further postulated that when a network break-in does occur, the best reaction (after repelling the invader) is to determine how access was gained so you can block that hole in your security. To do this, the author presents us with an architecture and toolkit for building network traffic analysis and event records: the Network Flight Recorder (NFR). The name reflects the similarity of purpose to that of an aircraft’s flight recorder, or “black box,” which can be analyzed after an event to determine the root cause.

Further, he postulated that information about network traffic over time may be used for trend analysis: identifying approaching bottlenecks as traffic increases, monitoring the use of key applications, and even monitoring the network traffic at peak usage periods in order to plan the best time for network maintenance. Thus, this information would be useful for network managers in planning their future growth.

The NFR monitors a promiscuous packet interface in order to pass visible traffic to an internally programmed decision engine. This engine uses filters, which are written in a high-level filter description language, read into the engine, compiled, and preserved as byte-code instructions for fast execution. Events that pass through the filters are passed to a combination of statistical and logging back-end programs. The output of these back-ends can be represented graphically as histograms or as raw data.

Ranum can be reached at <mjr@clark.com>; the complete NFR source code, including documentation, Java class source, decision engine, and space manager, is currently available from <http://www.nfr.net> for noncommercial research use.

**Extensible, Scalable Monitoring for Clusters of Computers**

Eric Anderson, University of California, Berkeley

The Cluster Administration using Relational Databases (CARD) system is capable of monitoring large clusters of cooperating computers. Using a Java applet as its primary interface, CARD
allows users to monitor the cluster through their browser.

CARD monitors system statistics such as CPU utilization, disk usage, and executing processes. These data are stored in a relational database for ease and flexibility of retrieval. This allows new CARD subsystems to access the data without modifying the old subsystems. CARD also includes a Java applet that graphically displays information about the data. This visualization tool utilizes statistical aggregation to display increasing amounts of data without increasing the amount of screen space used. The resulting information loss is reduced by varying shades of the same color to display dispersion.

Anderson can be reached at <eanders@u98.cs.berkeley.edu>. CARD is available from <http://now.cs.berkeley.edu/Sysadmin/esm/intro.html>.

**Monitoring Application Use with License Server Logs**

Jon Finke, Polytechnic Institute

Many companies purchase software licenses using their best estimate of the number required. Often, the only time this number changes is when users need additional licenses. A side effect of this is that many companies pay for unused software licenses. In this presentation, Jon Finke described a tool for monitoring the use of licensed software applications by examining license server logs.

This tool evolved from one designed to track workstation usage by monitoring entries in the wtmp file. Because most license servers record similar information (albeit in often radically different formats), the tool was modified to monitor license use.

Information can be displayed in a spreadsheet or as a series of linear graphs. The graphs provide an easy visual estimate of the number of software licenses actually in use at a given point in time, or over a period of time. Analysis of this information can quickly uncover unneeded licenses at your site.

Currently, the tool interfaces with Xess (a commercial spreadsheet available from Applied Information Services), Oracle, and Simon (available from <ftp://ftp.rpi.edu/pub/its-release/simon/README,Simon>). Finke can be contacted at <finkej@rpi.edu>.

**Session: The Business of System Administration**

Summaries by Brad C. Johnson

**Automating 24x7 Support Response to Telephone Requests**

Peter Scott, California Institute of Technology

Scott has designed a system, called helpline, that provides automated answering of a help desk telephone during nonpeak hours and is used for notifying on-call staff of emergencies within a short amount of time (minutes or seconds) once a situation is logged in the system (scheduler) database. This system was designed mainly to be cheap and therefore mostly applicable to sites with low support budgets. The system is comprised of source code written in Perl, the main scheduler information base written in SGML, and two dedicated modems—one for incoming calls (for problem reporting) and one for outgoing calls (for notification).

The rationale for creating helpline is that most other available software that was sufficient to provide automated support cost more than $100,000. Several tools that cost less were discovered, but they did not provide sufficient notification methods (such as voice, pager, and email according to a schedule). Recent entries into this market include a Telamon product called Tel Alert, which requires proprietary hardware, and VoiceGuide from Katalina Technologies, which runs only on Windows. There is also available some freeware software called tpage, but it concentrates on pagers, not on voice phones.

The key to the system is a voice-capable modem. When an incoming call is answered by the modem daemon, it presents a standard hierarchical phone menu—a series of prerecorded sound files that are linked to the appropriate menu choice. Independent of the phone menu system is the notifier and scheduler component. When an emergency notification occurs, the scheduler parses a schedule file (written in SGML) to determine who is on call at the time, determines what profile (i.e., action) is appropriate based on the time and situation, and takes the action to contact the designated on-call person. Multiple actions can be associated with an event, and if the primary notification method fails, alternate methods can be invoked.

Unfortunately, this software may not be completed for a long time (if ever) because funding and staff have been assigned to other projects, although the current state of the source code is available for review. Send email with contact information and the reason for your request to <jks@jpl.nasa.gov>. Additionally, in its current state, there are some significant well-known deficiencies such as data synchronization problems (which require specialized modem software), (over) sensitivity to the load of the local host (a host that is assumed to be reliably available), and virtually no available hard copy documentation.

**Turning the Corner: Upgrading Yourself from “System Clerk” to “System Advocate”**

Tom Limoncelli, Bell Labs, Lucent Technologies

Limoncelli believes that many administrators can be classified in one of two ways: as a system clerk or as a system advocate. A system clerk takes orders, focuses mainly on clerical tasks, and performs many duties manually. A system advocate is focused on making things better, automates redundant tasks, works issues and plans from soup to nuts, and
treats users as customers to create respectful, understanding partnerships for resolving problems. The key to job satisfaction, feeling better, and getting better raises is to make the transition from clerk to advocate.

Making a successful transition to system advocate requires converting bad (subservient) habits into good (cooperative) ones, creating spare time for better communication and quality time for planning and research, and automating mundane and repetitive tasks.

Although changing habits is always hard, it’s important to concentrate on getting a single success. Follow that with another and another, and over time these experiences will accumulate and become the foundation for good habits.

How to Control and Manage Change in a Commercial Data Center Without Losing Your Mind
Sally J. Howden and Frank B. Northrup, Distributed Computing Consultants Inc.

Howden and Northrup presented a methodology to ensure rigor and control over changes to a customer’s computing environment. They (strongly) believe that the vast majority of problems created today are caused by change. When change occurs unsuccessfully, the result can range from lost productivity to financial loss. Change is defined as any action that has the potential to change the environment and must consider the impact from software, hardware, and people. Using the rigorous method that was outlined will lower the overall risk and time spent on problems. They believe that this rigor is required for all changes, not just for significant or complex ones.

There are eight main steps outlined in this methodology: (1) Establish and document a base line for the entire environment. (2) Understand the characteristics of the change. (3) Test the changes in both an informal test and formal preproduction environment. (4) Fully document the change before, during, and after implementation. (5) Review the change with all involved parties before placing it into the production environment. (6) Define a detailed back-out strategy if the change fails in the production environment. (7) Provide training and education for all parties involved in the change. (8) Periodically revisit the roles and responsibilities associated with the change.

The authors were quite firm about testing a change in three physically distinct and separate environments. The first phase includes (unit) testing of the change on the host(s) involved in development. The second phase requires testing in a preproduction environment that, in the best case, is an exact duplicate of the production environment. The third phase is placing the change in the actual production environment.

When pressed on the suitability of using this (heavyweight) process on all changes, the authors stated that the highest priority activities are to fully document change logs and to create thorough work plans. The paper notes, however, that although this process does generate a significant
amount of work by the administrators before a given change, it has (over time) shown to reduce the overall time spent – especially for repeated tasks, when transferring information to other staff, when secondary staff are on duty, and when diagnosing problems.

Session: System Design Perspectives
Summaries by Mark K. Mellis

Developing Interim Systems
Jennifer Caetta, NASA Jet Propulsion Laboratory

Caetta addressed the opportunities presented by building systems in the real world and keeping them running in the face of budgetary challenges.

She discussed the role of interim systems in a computing environment – systems that bridge the gap between today's operational necessities and the upgrades that are due three years from now. She presented the principles behind her system design philosophy, including her extensions to the existing body of work in the area. Supporting the more academic discourse are a number of cogent examples from her work supporting the Radio Science Systems Group at JPL. I especially enjoyed her description of interfacing a legacy stand-alone DSP to a SparcStation 5 via the DSP’s console serial port that exposed the original programmer’s assumption that no one would type more than 1,024 commands at the console without rebooting.

Caetta described points to consider when evaluating potential interim systems projects, leveraging projects to provide options when the promised replacement system is delayed or canceled, and truly creative strategies for financing system development.

A Large Scale Data Warehouse Application Case Study
Dan Pollack, America Online

Pollack described the design and implementation of a greater-than-one-terabyte data warehouse used by his organization for decision support. He addressed such issues as sizing, tuning, backups, performance tradeoffs and day-to-day operations.

He presented in a straightforward manner the problems faced by truly large computing systems: terabytes of disk, gigabytes of RAM, double-digit numbers of CPUs, 50 Mbyte/sec backup rates – all in a single system. America Online has more than nine million customers, and when you keep even a little bit of data on each of them, it adds up fast. When you manipulate that data, it is always computationally expensive.

The bulk of the presentation discussed the design of the mass storage IO subsystem, detailing various RAID configurations, controller contention factors, back-up issues, and nearline storage of “dormant” data sets. It was a fascinating examination of how to balance the requirements of data availability, raw throughput, and the state of the art in UNIX computation systems. He also described the compromises made in the system design to allow for manageable system administration. For instance, if AOL strictly followed the database vendor’s recommendations, they would have needed to use several hundred file systems to house their data set. By judicious use of very large file systems so as to avoid disk and controller contention, they were able to use a few large (!) file systems and stripe the two gigabyte data files across multiple spindles, thereby preserving both system performance and their own sanity.
Shuse At Two: Multi-Host Account Administration
Henry Spencer, SP Systems

Spencer's presentation described his experiences in implementing and maintaining the Shuse system he first described at LISA ’96. He details the adaptation of Shuse to support a wholesale ISP business and its further evolution at its original home, Sheridan College, and imparted further software engineering and system design wisdom.

Shuse is a multi-host administration system for managing user accounts in large user communities, into the tens of thousands of users. It uses a centralized architecture. It is written almost entirely in the expect language. (There are only about one hundred lines of C in the system.) Shuse was initially deployed at Sheridan College in 1995.

Perhaps the most significant force acting on Shuse was its adaptation for ISP use. Spencer described the changes needed, such as a distributed account maintenance UI, and reflected that along with exposing Sheridan-specific assumptions, the exercise also revealed unanticipated synergy, with features requested by the ISP being adopted by Sheridan.

A principal area of improvement has been in generalizing useful facilities. Spencer observed in his paper, “Every time we’ve put effort into cleaning up and generalizing Shuse’s innards, we’ve regretted not doing it sooner. Many things have become easier this way; many of the remaining internal nuisances are concentrated in areas which haven’t had such an overhaul lately.”

Other improvements have been in eliminating shared knowledge by making data-transmission formats self-describing, and in the ripping out of “bright ideas” that turned out to be dim and replacing them with simpler approaches. These efforts have paid off handsomely by making later changes easier.

Spencer went on to describe changes in the administrative interfaces of Shuse, and in its error recovery and reporting. Shuse is still not available to the general public, but Spencer encourages those who might be interested in using Shuse to contact him at <henry@zoo.toronto.edu>

Spencer’s paper is the second in what I hope will become a series on Shuse. As a system designer and implementor myself, I look forward to objective presentations of experiences with computing systems. It’s a real treat when I can follow the growth of a system and learn how it has changed in response to real-world pressures and constraints. Often papers describe a system that has just been deployed or is in the process of being deployed; it is rare to see how that system has grown and what the development team has learned from it.

Session: Works in Progress
Summaries by Bruce Alan Wynn

Service Level Monitoring
Jim Trocki, Transmeta Corp.

Many system and network administrators have developed their own simple tools for automating system monitoring. The problem, proposes Jim Trocki, is that these tools often evolve into something unlike the original and in fact are not “designed” at all.

Instead, Jim presents us with m<on>: a Perl 5 utility, developed on Linux and tested on Solaris. m<on> attempts to solve 85% of the typical monitoring problems. The authors developed m<on> based upon these guidelines:

- Simple works best.
- Separate testing code from alert generation code.
- Status must be tracked over time.

The m<on> tool accepts input from external events and “monitors” (programs that test conditions and return a true/false value). The m<on> processes then examine these data and decide which should be presented directly to clients and which should trigger an alarm.

The authors are currently expanding the functionality of m<on> to include dependency checking of events, severity escalation, alert acknowledgments via the client, “I’m okay now” events, asynchronous events, a Web interface, and a better name.

The current version of m<on> is available at <http://consult.ml.org/~trocki/m<on>>.
Jim Trocki can be reached at <trocki@transmeta.com>.

License Management: LICCNTL – Control License Protected Software Tools Conveniently
Wilfried Gaensheimer, Siemens AG

Gaensheimer presented an overview of a number of tools that can help control and monitor the use of software licenses. The tools can also generate reports of license use over time.

For additional information on these tools, contact Gaensheimer at <wig@HL.Siemens.DE>.

Inventory Control
Todd Williams, MacNeal-Schwendler Corp.

One of the less exciting tasks that system and network administrators are often faced with is that of taking a physical inventory. Typical reasons for this requirement include:

- Maintenance contract renewal
- Charge backs for resource use
- Identifying the type of machinery

Williams began tracking his inventory by including comments in the system’s “hosts” files, but quickly outgrew this
mechanism when devices appeared that did not have an IP address, and when the amount of information desired made the "hosts" table unwieldy.

Instead, Williams developed a database to track this information. He developed procedures to keep this information up to date as machinery moves in and out of the work site.

For additional information on these software tools for tracking inventory, contact Todd Williams at <todd.williams@macsch.com>.

Values Count
Steve Tylock, Kodak

Although it may initially seem a surprising topic for a technical conference, Tylock reintroduced the basic values of a Fortune 500 company:

respect for the dignity of the individual
uncompromising integrity
trust
credibility
continuous improvement and personal renewal

Instead of applying these to the company itself, Tylock suggested that system and network administrators could increase their professionalism and efficiency by applying these basic values to their daily work.

For more information on this topic, contact Steve Tylock at <tylock@kodak.com>.

Extending a Problem-Tracking System with PDAs
Dave Barren, Convergent Group

Many system and network administrators use one type of problem-tracking system or another. But because working on the typical system or network problem often means working away from one's desk, administrators must keep track of ticket status independently of the tracking system. When administrators return to their desk, they must "dump" the information into the tracking system, hoping that they don't mis-key data or get interrupted by another crisis.

To help alleviate this problem, Barren suggests using a PDA to track ticket status. Barren has developed a relatively simple program for his Pilot that allows him to download the tickets, work the problems, track ticket status on the Pilot, then return to his desk and upload the changes in ticket status in one easy step.

This allows Barren to work on more tickets before returning to his desk and increases the validity of the tracking system. Barren hopes to encourage more users to implement this plan so that the increased number of Pilots will allow him to upload ticket status information at virtually any desk instead of returning to his own.

For additional information on this concept and the software tools Barren has developed, contact him at <dcbarr@npd.com>.

Survey of SMTP
Dave Parter, University of Wisconsin

One of the beautiful things about the Simple Mail Transfer Protocol is that it allows people to use any number of transfer agents to deliver electronic mail across the world. The down side is that there is a hodgepodge of versions and "brands" of transfer agents in use, and nobody really knows what is in use these days. Except, perhaps, Dave Parter.

To examine this issue, Parter monitored the incoming mail at his site for a short period of time. For each site that sent mail to his, he tested the SMTP greeting and tried to identify the type and version of the agent. His results:

sendmail: 60%
other/unknown: 17%
SMAP: 3%
PMDF: 2%

Parter was able to identify 140 distinct versions of sendmail in use in this small sampling.

Where, Parter asks, do we go from here with these data? He isn't sure. If you would like to discuss these findings, or conduct your own survey, contact Parter at <dparter@cs.wisc.edu>.

Session: Net Gains
Summaries by Mark K. Mellis

Creating a Network for Lucent Bell Labs Research South
Tom Limoncelli, Tom Reingold, Ravi Narayan, and Ralph Loura, Bell Labs, Lucent Technologies

This presentation described how, as a result of the split of AT&T Bell Labs Research into AT&T Labs and Lucent Bell Labs, they transitioned from an "organically grown" network consisting of four main user communities and ten main IP nets (out of a total of 40 class C IP nets) to a systematically designed network with two main user communities on four main IP nets, renumbering, rewiring, cleaning up, and "storming the hallways" as they went.

Unlike many projects of this scope, the authors planned the work as a phased transition, using techniques such as running multiple IP networks on the same media and operating the legacy NIS configuration in parallel with the new config to transition slowly to the new configuration, rather than make all the changes during an extended down time and discover a critical error at the end. They relate their experiences in detail, including a comprehensive set of lessons learned about strategy, end-user communications, and morale maintenance. ("Yell a loud chant before you storm the hallways. It psyches you up and makes your users more willing to get out of the way.")

Having been faced with a network unfortunately typical in its complexity, and real-world constraints on system down-
time, this group described their thought processes and methodologies for solving one of the problems of our time, corporate reorganization. In the face of obstacles such as not having access to the union-run wiring closets and “The Broken Network Conundrum,” where one must decide between fixing things and explaining to the users why they don’t work, they divided their networks, fixed the problems, and got a cool T-shirt with a picture of a chainsaw on it, to boot.

Some the tools constructed for this project are available at <http://www.bell-labs.com/user/ta1>.

**Pinpointing System Performance Issues**
Douglas L. Urner, BSDI

Urner gave us a well-structured presentation that within the context of a case study on Web server performance optimization presents a systematic model for tuning services from the network connection, through the application and operating system, all the way to the hardware. His paper is a vest-pocket text on how to make it go faster, regardless of what “it” might be.

Urner began the paper by describing an overview of system tuning: methodology, architecture, configuration, application tuning, and kernel tuning. He discussed the need to understand the specifics of the problem at hand – protocol performance, application knowledge, data collection and reduction. He then described tuning at the subsystem level, including file system, network, kernel, and memory. He presented a detailed explanation of disk subsystem performance, then went on to examine CPU performance, kernel tuning, and profiling both application and kernel code. Urner’s paper is about optimizing Web server performance, but it is really about much more. He describes, in detail, how to look at performance optimization in general. He encourages readers to develop their intuition and to establish reasonable bounds on performance. By estimating optimal performance, the system designer can determine which of the many “knobs” in an application environment are worth “turning”, and help set reasonable expectations on what can be accomplished through system tuning.

**Session: Configuration Management**
Summaries by Karl Buck

The first two papers deal with the actual implementations of tools written to handle the specific problems. The third paper is an attempt to get a higher level view of where configuration management is today and make suggestions for improving existing CM models.

**Automation of Site Configuration Management**
Jon Finke, Rensselaer Polytechnic Institute

Finke presented his implementation of a system that not only tracks interesting physical configuration aspects of UNIX servers, but also stores and displays dependencies between the servers and the services that they provide. The configuration management system has an Oracle engine and outputs data to a Web tree, making for a very extensible, useful tool. For instance, if a license server is to be updated, one can find out not only all the other services that will be affected, but also the severity of those outages and who to contact for those services. Source code is available; see <http://ftp.rpi.edu/pub/its-release/simon/README.simon> for details.
Chaos Out of Order: A Simple, Scalable File Distribution Facility for “Intentionally Heterogeneous” Networks
Alva L. Couch, Tufts University

The core of this paper is a file distribution tool written by Couch called DISTR. Using DISTR, administrators of unrelated networks can use the same file distribution system, yet retain control of their own systems. DISTR can “export” and “import” files to and from systems managed by other people. Frank discussion is given to the existing limitations and potential. DISTR is available at <http://ftp.eecs.tufts.edu/pub/distr>.

An Analysis of UNIX System Configuration
Remy Evard, Argonne National Laboratory

This paper is an attempt to step back and take a look at what is available for use in UNIX configuration and file management. Examine a few case studies, and make some observations concerning the current configuration process. Finally, Evard argues for a “stronger abstraction” model in systems management, and makes some suggestions on how this can be accomplished.

Session: Mail
Summaries by Mark K. Mellis

Tuning Sendmail for Large Mailing Lists
Rob Kolstad, BSDI

Kolstad delivered a paper that described the efforts to reduce delivery latency in the <inet-access@earth.com> mailing list. This mailing list bursts to up to 400,000 message deliveries per day. As a result of the tuning process, latency was reduced to less than five minutes from previous levels that reached five days.

Kolstad described himself as a member of Optimizers Anonymous, and he shared his obsession with us. He described the process by which he and his team analyzed the problem, gathered data on the specifics, and iterated on solutions. He took us through several rounds of data analysis and experimentation, and illustrated how establishing realistic bounds on performance and pursuing those bounds can lead to insights on the problem at hand.

Kolstad and his team eventually homed in on the approach of increasing the parallelism to the extreme of using hundreds of concurrent sendmail processes to deliver the list. They also reduced timeouts for nonresponsive hosts. This, of course, required the creation of a number of scripts to automate the parallel queue creation. These scripts are available upon request from Kolstad, <kolstad@bsd.com>.

Kolstad closed by noting that after the optimizations were made, the biggest remaining problem was unavailability of recipients. He expressed his amazement that in a mailing list dedicated to Internet service providers, some one to three percent of recipients were unreachable at any point in time. Also, even with these improvements, the mailing list traffic of mostly small messages doesn’t tax even a single T-1 to its limits.

Selectively Rejecting SPAM Using Sendmail
Robert Harker, Harker Systems

Harker offered a presentation that addressed one of the hottest topics on the Internet today – unsolicited commercial email, otherwise known as spam. He characterizes spam, examines the different requirements for antispam processing at different classes of sites, and offers concrete examples of sendmail configurations that address these diverse needs.

After his initial discussion of the nature of spam, Harker outlined the different criteria that can be used for accepting and rejecting email. His approach differs from others in that he spends sendmail CPU cycles to get finer granularity in the decision to reject a message. He goes on to treat the problem of spammers sending their wares to internal aliases and mailing lists.

The remainder of the presentation was devoted to detailed development of the sendmail rulesets necessary to implement these policies. He discussed the specific rulesets and databases needed, and how to test the results. His discussion and code are available at <http://www.harker.com/sendmail/anti-spam>
A Better E-mail Bouncer
Richard J. Holland, Rockwell Collins

Holland presented work that was motivated by corporate reorganization: how to handle email address namespace collisions in a constructive way.

As email usage becomes more accessible to a wider spectrum of our society, fewer and fewer email users are able to parse the headers in a bounced message. Holland talked about his bouncer, implemented as a mail delivery agent, which provides a clearly written explanation of what happened and why when an email message bounces due to an address change. This helps the sender understand how to get a message through, helps the recipient get a message, and helps the postmaster by automating another portion of her workload.

The bouncer was originally implemented as a simple filter. Because of the diversity in headers and issues related to envelope vs. header addresses, especially in the case of bcc: addresses, the bouncer was reimplemented as a delivery agent. The bouncer, written in Perl, relinquishes its privilege and runs as “nobody.” Many of the aspects of bouncer operation are configurable, including the text of the explanatory text to be returned. A particularly nice feature is the ability to send a reminder message to the recipient’s new address when mail of bulk, list, or junk precedence is received, reminding them to update their mailing list subscriptions with the new address.

Holland concluded by discussing alternatives to the chosen implementation and future directions. Those interested in obtaining the bouncer should contact Holland at <holland@pobox.com>.

INVITED TALKS TRACK

So Now You Are the Project Manager
William E. Howell, Glaxo Wellcome Inc.

Summary by Bruce Alan Wynn
Many technical experts find themselves gaining responsibility for planning and implementing successively larger projects until one day they realize that they have become a project manager.

In this presentation, Howell offered helpful advice on how you can succeed in this new role without the benefit of formal training in project management.

Howell’s first suggestion is to find a mentor, someone who has successfully managed projects for some time. Learn from that mentor not only what the steps are in managing a project, but also the reasons why those are the right steps.

But, as Howell points out, a mentor is not always available. What do you do then? Howell presented a few tips on what you can do if you can’t find a mentor.

For copies of the presentation slides, contact Marie Sands at <mms31901@glaxowellcome.com>; please include both your email and postal addresses.

When UNIX Met Air Traffic Control
Jim Reid, RTFM Ltd.

Summary by Mike Wei
Every once in a while we see reports of mishaps of the rapidly aging air traffic control (ATC) system in the United States. We have also seen reports that some developing countries have ATC systems “several generations newer” than the US system. For most of the flying public, the ATC system is something near a total mystery on which our lives depend. As a pilot and a system administrator, I hope I can lift the shroud of mystery a little bit and help explain the ATC system Reid talked about, how UNIX handles such a mission-critical system, and how this system helps air traffic control.

The primary purpose of air traffic control is traffic separation, although it occasionally helps pilots navigate out of trouble. Government aviation authorities publish extensive and comprehensive regulations on how aircraft should operate in the air and on the ground. Air traffic control is a massively complex system of computers, radar, controllers, and pilots that ensures proper traffic separation and flow. Human participants (i.e., controllers and pilots) are as essential as the computer and radar systems.

Naturally, air traffic congestion happens near major airports, called “terminal areas.” In busy terminal areas, computer-connected radar systems provide controllers with realtime traffic situations in the sky. Each aircraft has a device called a transponder that encodes its identity in its radar replies, so the controllers know which aircraft is which on the computer screen. Computer software along with
traffic controllers ensure proper separation and traffic flow by vectoring planes within the airspace to their destinations.

Outside terminal areas, large planes usually don’t fly anywhere they want. They follow certain routes, like highways in the sky. On-route traffic control centers control traffic along those routes. Traffic separation is usually ensured by altitude separation or fairly large horizontal separation. Some on-route centers have radar to help track the traffic. For areas without radar coverage, on-route centers rely on pilot position reports to track the traffic and usually give very large separation margins.

This system worked fairly well for many years, until air travel reached record levels. Two things happened. First, some terminal areas became so congested that, during some parts of the day, the airspace just couldn’t hold any more traffic. Second, traffic among some terminal areas reached such a level that these on-route airspaces became almost as congested as terminal areas.

A new kind of system was developed to address the new problems. This “slot allocation system” tries to predict what the sky will look like in the future, based on the flight plan filed by airliners. Based on the computer prediction, we can allocate “slots” in the sky for a particular flight, from one terminal area to another, including the on-route airspace in between. Every airline flight is required to have a flight plan, including departure time, estimated time on-route, cruising airspeed, planned route, destination, and alternate destination. With the flight plan, an airplane’s position in the sky is fairly predictable.

This slot allocation system is very much like TCP congestion control in computer networking: when the network is congested, the best way to operate is to stop feeding new packets into it for a while. For the same reason it’s much better to delay some departures than to let planes take off and wait in the sky if the computer system predicts congestion sometime in the future.

The Western European airspace, according to Reid, is the busiest airspace in the world. Instead of a single controlling authority, like the US Federal Aviation Authority, each country has its own aviation authorities. Before “Eurocontrol,” the agency Reid worked at last year, each country managed its airspace separately, and an airliner had to file a flight plan for each country it had to fly over along its route. This led to a chaotic situation when traffic volume increased. According to Reid, there was also a problem of ATC nepotism (i.e., a country favoring its own airliners when congestion occurred).

The Eurocontrol agency has three UNIX-based systems that serve Western Europe. IFPS is a centralized flight plan submission and distribution system, TACT is the real-time slot allocation system, and RPL is the repeat flight plan system.

IFPS provides a single point of contact for all the flight plans in Western Europe. It eliminates the inconvenience of filing multiple flight plans. This is basically a mission-critical data entry/retrieval system.

The TACT system provides slot allocation based on the flight plan information in the IFPS system. It provides slots that satisfy separation standards in the airspace above Western Europe. It controls when an airplane can take off and which slots in the sky it can fly through to its destination. It keeps a “mental picture” of all the air traffic in the sky for all the moments into some future. RPL is the repeat flight plan system. Airlines tend to have the same flights repeatedly, and this system simplifies filing those flight plans. The RPL system is connected with the IFPS system and feeds it with those repeat flight plans.

This must be an awesomely impressive system with equally impressive complexity. According to Reid, it actually works. Ever since the adoption of the system, it has never failed. Furthermore, the increase in traffic delay is much less than the increase in traffic volume. Kudos for our European computer professionals!

The slot allocation system does not provide the actual traffic separation. Realtime traffic separation must be based on actual position data obtained from radar or pilot position report, rather than projected position data based on flight plan. However, this slot allocation system is an invaluable tool to help the realtime traffic separation by avoiding congestion in the first place.

Using UNIX in such a mission-critical system is quite pioneering in an ATC system. Most ATC systems in the US are still mainframe-based. The system is built on multiprocessor HP T90 servers, and the code is written in Ada.

Like most of the mission-critical systems, operation of those UNIX systems has its idiosyncrasies. According to Reid, the system operation suffers organizational and procedural inefficiencies. However, some of them may well be the necessary price to pay for such a mission-critical system. The whole system is highly redundant; almost all equipment has a spare. The maintenance downtime is limited to one hour a month. Change control on the system is the strictest I’ve ever heard of. For new code releases, it has a test environment fed with real data, and there’s a dedicated test group that does nothing
but the testing. Any change to the production systems must be documented as a change request and approved by a change board, which meets once a week. Any kind of change, including fixing the sticky bit on /tmp, needs change board approval. Reid said that it took SA six weeks to fix the /tmp permissions on six machines because each one needed a change request and only one change a week is allowed on the production system. To minimize the chance of system failure, all nonessential service on the system is turned off, including NFS, NIS, and all other SA life-saving tools. This does add pain to SA’s daily life.

This kind of process sounds bureaucratic, and it’s a far cry from a common UNIX SA’s habit. However, for this kind of system, it might be right to be overly conservative. At least when Reid flew to the LISA conference this year, he knew nothing bad would likely happen to Eurocontrol due to a system administrator’s mistake.

Enterprise Backup and Recovery: Do You Need a Commercial Utility?

W. Curtis Preston, Collective Technologies

Summary by Bruce Alan Wynn

Nearly every system administrator has been asked to back up filesystems. Even those who haven’t have probably been asked to recover a missing file that was inadvertently deleted or corrupted. How can a system administrator determine the best solution for a backup strategy?

In this presentation, Preston presented an overview of standard utilities available on UNIX operating systems: which ones are common, which ones are OS-specific. He then explained the capabilities and limitations of each. In many cases, claims Preston, these “standard” utilities are sufficient for a site’s backup needs.

For sites where these tools are insufficient, Preston discussed many of the features available in commercial backup products. Because some features require special hardware, Preston described some of the current tape robots and media available. Once again, he iterated the capabilities and limitations of each.

Copies of Preston’ presentation are available upon request; Preston can be reached at <curtis@colltech.com>.

A Technologist Looks at Management

Steve Johnson, Transmeta Corp.

Summary by Bruce Alan Wynn

Employees often view their management structure as a bad yet necessary thing. Johnson has worked in the technical arena for years, but has also had the opportunity to manage research and development teams in a number of companies. In this presentation, he offered his insight into methods that will smooth the relationship between employees and managers.

Johnson began by postulating that both employees and managers have a picture of what the manager-employee relationship should look like, but it is seldom a shared picture. He further postulated that a great deal of the disconnect is a result of personality and communication styles rather than job title.

Johnson loosely categorized people as either thinkers, feelers, or act-ers. A thinker focuses on analyzing and understanding; afeeler focuses on meeting the needs of others; an act-er focuses on activity and accomplishment.

These differences in values, combined with our tendency to presume others think as we do, cause a breakdown in communication that leads to many of the traditional employee-manager relationship problems.

After making this point, Johnson suggested that technical people who are given the opportunity to move into management first examine closely what the job entails: it’s not about power and authority; it’s about meeting business needs. He suggested a number of publications for additional information on this topic.

Steve Johnson can be reached at <scj@transmeta.com>.

IPv6 Deployment on the 6bone

Bob Fink, Lawrence Berkeley National Laboratory

Summary by Mike Wei

We all know that IPv6 is the future of the Internet; there’s simply no alternative to support the explosive growth of the Internet. However, despite years of talking, we see little IPv6 deployment. According to Fink, the adaptation and deployment of IPv6 is currently well under way, and it’s leading in the right direction.

An experimental IPv6 network, named 6bone, was created to link up early IPv6 adopters. It also serves as a test bed to gain operational experiences with IPv6. Because most of the machines on the 6bone also run regular IPv4, it provides an environment to gain experience in IPv4 to v6 transition.

The 6bone is truly a global network that links up 29 countries. Most of the long haul links are actually IPv6 traffic tunnelled through the existing IPv4 Internet. This strategy allows 6bone to expand
anywhere that has an Internet connection for almost no cost. On the 6bone network, there are some “islands” of network that run IPv6 natively on top of the physical network.

An important milestone was achieved in IPv6 deployment when Cisco, along with other major router companies, committed to IPv6. According to Fink, IPv6 will be supported by routers in the very near future, if it’s not already supported. In addition, we will start to see IPv6 support in major OS releases.

A typical host on the 6bone runs two IP stacks, the traditional v4 stack and the IPv6 stack. The IPv6 stack can run natively on top of the MAC layer if the local network supports v6, or it can tunnel through IPv4. The v6 stack will be automatically used if the machine talks to another v6 host. An important component of the 6bone network will be the new DNS that supports IPv6 addresses. The new DNS supports AAAA record (quad-A record, because a v6 address is four times the length of a v4 address). If a v6 host queries the new DNS server for another v6 host, an AAAA record will be returned. Because the new DNS simply maps a fully qualified domain name to an IP address (v4 or v6), the DNS server itself doesn’t have to sit on a v6 network. It will be perfectly normal for a dual-stack v6/v4 host to query a DNS server on the v4 network, getting a v6 address, and talk to the v6 host in IPv6.

The key to the success of IPv6 deployment is smooth transition. The transition should be so smooth that a regular user should never know when the IPv6 has arrived. Given the fact that the IPv4 network is so far reaching throughout the world, IPv6 and v4 will coexist for a very long time; the transition to IPv6 from v4 will be gradual. Routers will be the first ones that have IPv6 capabilities. Just like the 6bone, an IPv6 backbone can be built by tunnelling v6 traffic through the existing v4 network or run v6 natively on the physical network when two neighboring routers both support v6. Because v6 is just another network layer protocol, it can run side by side with IPv4 on the same physical wire without conflict, like IP and IPX can run together on the same Ethernet. This means that we do not have to make a choice between v6 and v4; we can simply run both of them during the transition period. IPv6 hosts will gradually become IPv6 capable when the new OS versions support it. During the transition, those IPv6 hosts will have dual IP stacks so they can talk to both v4 and v6 hosts. Nobody knows how long this “coexist” will last, but it will surely last for years. When the majority of the hosts on the Internet are doing v6, some of the hosts might choose to be v6 only. One by one, the v4 hosts will fade away from the Internet.

Will that ever happen? The answer is yes. In the next decade, the IPv4 address will be so hard to obtain that IPv6 will be a very viable and attractive choice. We haven’t seen that yet, but based on the current Internet growth, it will happen.

The IPv6 addressing scheme is another topic Fink talked about in the seminar. IPv6 has a 128 bit address space, which allows thousands of addresses per square foot if evenly spread on the earth’s surface. How to make use of this address space in a highly scalable way is a big challenge. IPv4 suffers the problem of an explosive number of routing entries, and this problem arises years before the exhaustion of IPv4 addresses. To address this problem and to allow decades of future expansions, IPv6 uses an aggregation-based addressing scheme.

<table>
<thead>
<tr>
<th>3bits</th>
<th>13bits</th>
<th>32bits</th>
<th>16bits</th>
<th>64 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>TLA</td>
<td>NLA</td>
<td>SLA</td>
<td>Interface ID</td>
</tr>
<tr>
<td>public topology</td>
<td>site</td>
<td>local machine topology</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The best analogy of this aggregation-based addressing is the telephone number system. We have ten-digit phone numbers in US and Canada, with a three-digit area code, three-digit exchange code, and the last four digits for individual telephone lines.

The first three bits are 001. In the great tradition of TCP/IP, other combinations are reserved for future use, in case one day we have an interplanetary communication need that requires a different addressing scheme. The 13-bit TLAs are top-level aggregators, designed to be given to long-haul providers and big telcos that run backbone service. The 32-bit NLAs are next-level aggregators for various levels of ISPs. It can be further subdivided to several levels of NLAs. The 16-bit SLAs are for site topologies. (It’s like getting a class A IPv4 address and use 16-bit for network address.) The machine interface ID is 64 bits.

An important feature of IPv6 is autoconfiguration, in which a host can figure out its own IPv6 address automatically. The 64-bit interface ID is designed so that the host can use its data link layer interface
address as the host portion of the IPv6 address. Ethernet uses a 48-bit address, and it seems adequate for globally unique addresses. Reserving 64 bits for the local machine shall accommodate any future address method used by future physical networks.

Aggregation-based addressing is a big departure from the current IPv4 addressing. Although IPv4 has three classes of addresses, it's not a hierarchical addressing scheme. In IPv4 (at least before the CIDR days), all the network's addresses were created equal, which means they could all be independently routed to any locations they chose to be. This caused the routing entry explosion problem when the Internet grew. Classless Inter Domain Routing (CIDR) was introduced as a stopgap measure to address this urgent problem by introducing some hierarchy in the IPv4 address space. IPv6 is designed at the beginning with a hierarchical scheme. By limiting the number of bits for each aggregator, there is an upper limit to the number of routing entries that a router needs to handle. For example, a router at a long-haul provider needs only to look at the 13-bit TLA portion of the address, limiting the possible number of routing entries to 213.

Another advantage of a hierarchical-based addressing system is that address allocation can be delegated in a hierarchical manner. The success of DNS teaches us the important lesson that delegation of address allocation authority is a key to scalability.

There's a price to pay to use a hierarchical addressing system. When a site changes its providers, all the IP addresses need to be changed. We already experience the same kind of issue in IPv4 when we use CIDR address blocks. IPv6 tries to make address changes as painless as possible, to have a host autoconfigure itself. The host will use its MAC layer address as the lower portion of its IPv6 address and use a Network Discovery protocol to find out the upper portion of the address (routing prefixes). The whole site can be renumbered by simply rebooting all the hosts without any human intervention.

There are still lots of problems to be discovered and addressed in IPv6. That's exactly what the 6bone is built for. IPv6 is the future of the Internet, and the transition to IPv6 will start in the near future.

More information on 6bone can be found on <http://www.6bone.net>

**Joint Session**

**Panel: Is System Administration a Dead-End Career?**

Moderator: Celeste Stokely, Stokely Consulting
Panelists: Ruth Milner, NRAO; Hal Pomeranz, Deer Run Associates; Wendy Nather, Swiss Bank Warburg; Bill Howell, Glaxo Wellcome Inc.

Summary by Carolyn M. Hennings

Ruth Milner opened the discussion by responding to the question with, "It depends." She went on to explain that it is necessary for everyone to define "system administration" and "dead-end career" to answer this question for themselves. In some organizations, "system administration" leaves no room for growth. However, Ruth pointed out that if people enjoy what they do, then maybe it should not be considered a "dead-end."

Hal Pomeranz outlined the typical career progression for system administrators. He described the first three years in the career field as a time of learning while receiving significant direction from more senior administrators. During the third through fifth years of practicing system administration, Hal suggested that even more learning takes place as the individual works with a greater degree of autonomy. Hal observed that people with more than five years of experience are not learning as much as they were, but are more focused on producing results as well as mentoring and directing others. Hal commented that many organizations move these senior people into management positions and wondered how technical tracks might work.

Wendy Nather discussed the question from the angle of recruiting. Those hiring system administrators are looking for people who have dealt with a large number of problems as well as a variety of problems. She pointed out that being a system administrator is a good springboard to other career paths. Wendy outlined some of the characteristics of good system administrators that are beneficial in other career areas: a positive attitude, social skills, open-mindedness, and flexibility.

Bill Howell examined the financial prospects for system administrators. He commented that there will always be a need for system administrators. However, industry may be unable and unwilling to continue to pay high salaries for them, and salary increases may begin to be limited to "cost of living" increases. Bill suggested that growth in personal income and increases in standard of living are the results of career advancement. If salaries
do become more restricted in the future, system administration may become a dead-end career.

Celeste then opened up the floor for questions and discussion. One participant asked about other career options if one was not interested in pursuing the managerial or consultant path. The panel suggested that specializing in an area such as network or security administration would be appropriate. Discussion ranged among topics such as motivation for changing positions, how the size of the managed environment affects opportunities and working relationships, the impact of Windows NT on UNIX administrator’s careers, how an administrator’s relationship with management changes with career advancement, and the importance of promoting system administration as a profession.

**BOFs**

Summaries by Carolyn M. Hennings

**Keeping a Local SAGE Group Active**

This BOF at LISA ’96 and SANS ’97 inspired me to start a group in Chicago. Chigrp’s initial meeting was in early October, and I was anxious to announce our existence. General suggestions for getting a group started and keeping one alive were shared by attendees. If you want more information on how to start a group, see <http://www.usenix.org/sage/locals/>.

**Documentation Sucks!**

As system administrators, we all know how important documentation is, but we hate to write it. This BOF explored some of the reasons we don’t like to write documentation, elements of good documentation, and what we can do personally to improve our efforts in this area. About 50 people attended the BOF. Some professional technical writers participated in the BOF and were interested in the approach sys admins were taking in their struggle to write documentation.
Worth Repeating

by Tina Darmohray
Tina Darmohray, editor of SAGE News & Features, is a consultant in the area of Internet firewalls and network connections, and frequently gives tutorials on those subjects. She was a founding member of SAGE.

<tm@usenix.org>

There were a lot of things to see and do at the recent LISA conference in San Diego. One of the highlights for me was the short, impromptu speech that Paul Vixie gave as he accepted the annual SAGE Outstanding Achievement Award. I wish I could think up such “right on,” heartfelt words on the spot. He did, and they really went to the heart of one of the reasons why it’s so cool to be a part of this technical community. The award reads:

SAGE
The 1997 Outstanding Achievement Award
Presented to Paul Vixie for his work on DNS and BIND and for his leadership in opposing inappropriate use of the Internet

I’ve included them here for those of you who didn’t have the opportunity to hear Paul’s words live. But even if you caught his speech, I think his message is worth repeating.

My first reaction on being told of this was that you guys are really scraping the bottom of the barrel now! Pause for laughter to die down.

I guess I’ve got a couple of remarks to make. One is that at my very first USENIX, I was sitting in the very first tutorial and the fellow next to me was from Alaska, or somewhere else far away, and he was remarking on how it was amazing him to walk through the halls of the conference and see all of the people that he had been hearing about and reading articles from on USENET over the years. And I got to talking to him about where I was from, and what I did, and I told him that I had just started at DEC Palo Alto. Now, he was sitting on one side of me and my badge was hanging on the other side of me, and he said, “Oh! Isn’t that where Vixie hangs out?”

Paul imitated the original delivery of his last name, which was with some hint of disdain or incredulity.

And I said, “Uh huh.” Pause for laughter to die down.

And certainly, on one hand, I was very glad to have been noticed (we all like attention), but I was a little bit concerned at his tone of voice, in that perhaps I had been even more vitriolic than I had intended to be on the various mailing lists and newsgroups where he might have noticed me. And that sort of gave me my first inkling that someday I was going to be a well-known person, and I wondered what that would be like.

What I’ve discovered is that it’s a little bit like parenting: when you’re a kid, you think your parents know everything and have super powers, and that someday you’re going to grow up and you’ll know everything, too. Then you get to be a parent and discover that your parents were shining you on; they didn’t have a clue! So here I am!

Applause and laughter by the attendees and a pause before Paul continues, with a more serious demeanor.

I would not be here if not for the kind attentions of Rick Adams and Brian Reid, both of whom thought that I was worth teaching at various times. And if their work can be an example to me it will be because I have found other people who were worth teaching and I hope that in turn will be an example to all of you.

The way that this industry works is not that we all go to school. Some folks have been lucky enough to work with Evi and other really great instructors. But, for the most part, we’re in this for fun and we’re trying to learn and it’s only to the extent that other people are willing to work with us and tell us the clues so we can prosper and succeed and get better at what we do. So if you’re not mentoring somebody, you should start thinking about how you’re going to start doing that.

I thank you for the award.

Postscript from Paul: At my first USENIX conference, I noticed Henry Spencer’s name tag and the man who went with it and wondered, as described in my award speech, what it must be like to write software that everybody everywhere runs. Since then, I have inherited and written some well-known software. At this conference, I spoke to Henry. We talked about memory leaks in C News and BIND, and now I know the answer to my earlier question. I am very honored to have been recognized by my peers.
One of the issues on my mind recently is the relationship between SAGE and various vendors. Although this hasn’t become a very high priority, it’s still something we have to deal with and will continue to deal with. Here then, are some thoughts on the matter. I hope someone will ponder this more thoroughly and write a proposal on what we ought to do.

As I often do, I broke this down into a series of questions, then put some answers together. Everyone has their own style. My questions this time are:

- What is the problem?
- What is the current “relationship”?
- What do we want the relationship to be?
- What do we not want the relationship to be?

What Is the Problem?

It’s us, or some of us anyway. We as a guild have been conscientiously avoiding the “appearance of impropriety” with regard to tying ourselves to any vendor to the detriment of others. We as individuals often earn our livelihood from these vendors. The best interests of our employers may sometimes conflict with “the greater good” of society. Fortunately, that conflict doesn’t seem to be very serious yet. As our profession continues to grow and mature and as vendors continue to jump on the bandwagon (“Oh wow, thousands of potential buyers, and they’re the ones who actually decide what to buy!”), I expect this to become more difficult to manage.

The problem, then, is how to reconcile the conflicting interests of the individual vendors that supply our industry with those of the community as a whole.

What Is the Current “Relationship”? Things have been, overall, pretty good thus far. We have sponsoring members, an excellent vendor show at LISA, hospitality suites, T-shirts, mugs and toys, and some level of influence on the engineers at those vendors who are building the products we will be purchasing (partly because those engineers are “us” too). However, there are also some negatives. If we are seen to favor one vendor over others, we are opening ourselves to various problems. We have seen in a number of instances that BOFs at LISA can degenerate into sales pitches if we’re not careful. We are beginning to see a proliferation of separate “certification” programs, with no telling where that may lead.

There have always been a few “very big” vendors in our industry and generally lots of “wish we were big” vendors. My view of where SAGE fits into the community does not include helping make anyone profitable, but to help vendors ensure their products meet our needs, for the obvious self-serving interest.

What Do We Want the Relationship to Be?

One of the primary motivating factors in my interest in SAGE has always been the gain we all get from pooling our resources. The more we share our precious time and effort, the more we all win. I see this as the biggest potential value for SAGE members in future relations with our vendor force. We as individuals and as a guild ought to be intimately involved with vendors on future direction, bug reporting, design issues, and so forth. We should be the place vendors turn when they want to know which

---

SAGE STG EXECUTIVE COMMITTEE

President:
Hal Miller <helm@usenix.org>

Secretary:
Tim Gassaway <gassaway@usenix.org>

Treasurer:
Barb Dijker <barb@usenix.org>

Members:
Helen Harrison <helen@usenix.org>
Amy Kreiling <amy@usenix.org>
Kim Trudel <kim@mit.edu>
Pat Wilson <paw@usenix.org>

---

SAGE, the System Administrators Guild, is a Special Technical Group within USENIX. It is organized to advance the status of computer system administration as a profession, establish standards of professional excellence and recognize those who attain them, develop guidelines for improving the technical and managerial capabilities of members of the profession, and promote activities that advance the state of the art or the community.

To achieve its mission SAGE may:

- Sponsor technical conferences and workshops;
- Publish a newsletter, and/or professional short topics series;
- Develop curriculum recommendations and support education endeavors;
- Develop a process for the certification of professional system administrators;
- Recognize system administrators who are outstanding or are otherwise deserving of recognition for service to the professional community;
- Speak for the concerns of members to the media and make public statements on issues related to system administration;
- Promote and support the creation and activities of regional or local professional system administrators.
products or features are of value and which are merely marketing hype, useless to those of us stuck trying to implement them.

I want vendors to “bring” SAGE with them when they visit their client sites. They see many of our “potential” members, particularly those who don’t know we exist and whom we have (thus far) not reached. I want more vendors to join as sponsoring members. And, of course, I like the toys they supply us with. . . .

What Do We Not Want the Relationship to Be?
Clearly, I don’t want to see us align too strongly with any one vendor in any given field. We border on that occasionally as it is, which makes me think we need to consider taking steps to improve competition, but that’s not our role either. I was a little upset with a couple of vendors who put on at the last LISA “BOF”s that were nothing more than sales presentations. I most definitely don’t want any vendor (or series of vendors) influencing SAGE’s direction to any significant degree, even though I recognize that every vendor is likely to influence us to some degree.

What Can You Do?
Think about how our vendors can meet their needs without doing so at the expense of SAGE and how SAGE can continue to press for improvements in computer systems and networks without doing so at the expense of any given vendor. How can we define the roles, and how do we ensure continued good relations?

New Editor for Short Topics Series
SAGE is pleased to announce that William N. LeFebvre is now Editor for the publication series, “Short Topics in System Administration.” This series of booklets is intended to present topics in a thorough, refereed fashion that are of immediate use to the growing community of system and network administrators.

Bill has been a tutorial speaker, program committee member, session chair, author, speaker, and guru at various USENIX and SAGE events since 1992. He is a published author on UNIX topics and has 15 years of experience with UNIX systems, most of them as a system administrator. As editor, he will be responsible for coordinating new content for the series; organizing revised editions of existing booklets; and acting as liaison between authors, readers, and the USENIX staff. “I consider it an honor to be appointed to this position,” Bill says, “I will endeavor to make the SAGE series an indispensable part of every system administrator’s book collection.”

Forthcoming in early 1998 are two new booklets in the series on system administrator education and on hiring and interview practices. A booklet on legal issues for system administrators will follow. If you have a proposal for a new topic that you’d like to develop into a booklet, please contact Bill LeFebvre at <wnl@usenix.org>.
Auditing: The Ugly Duckling of Computers

Auditing is an often overlooked portion of system administration, both in the UNIX and Windows NT world. At least 80% (maybe 95%) of the sites I work with do not utilize the auditing functionality of their systems. In this article I discuss what I consider the minimal amount of auditing that would be of benefit in troubleshooting or security tracking. I then give specific recommendations for Solaris 2.X and Windows NT 4.0.

Most people I talk with cite "resources," both human and computer, as the reason they have not yet implemented auditing. They say that computer resource usage, such as disk, processor, and memory, is extensive, and that the time resource for their staff to maintain the system as well as review the data is not worth the hassle. I understand this mindset. My first experience (1991) with C2 auditing on SunOS was daunting. The log files filled up so fast that I could not maintain them adequately, never mind that I never got a chance actually to look through the logs for information. After about two weeks, I just shut it off and wrote the experience off as "proper motivation, but ignorance of reality." The following is a good introduction into practical settings for auditing, with specifics for Windows NT 4.0 and Solaris 2.X.

Minimal Auditing Requirements

With all the reasons we have for why we don't and why we should audit, there needs to be a starting point. Many people ask me, "What should I be auditing?" To answer this, I have compiled a general list of "auditing categories" that I feel is a good minimal starting point:

- all successful and unsuccessful login and logout events
- all modifications to system specific files (config files, system binaries, and libraries)
- all administrative actions (user adds, host changes, password changes, etc.)
- all system type events (reboots, eeprom changes, etc.)

This set gives you a very good, though not complete, picture of your system at any given point in time. You will find it invaluable in troubleshooting, as well as incident handling.

Setting Up Auditing: The Specifics for Windows NT

First you must "enable" auditing. There are two different types of auditing in Windows NT: system level and file/directory level. The "system level" is what I am most concerned with. To set system level auditing, open the "User Manager" tool. While in the User Manager select "Policies->Audit->Audit These Events." Then set the following policy:

- Logon and Logoff: success and failure
- File and Object Access: none (see Auditing Files and Directories)
- Use of User Rights: none
- User and Group Management: success and failure
- Security Policy Changes: success and failure
Restart, Shutdown, and System: success and failure

Process Tracking: none

**Auditing Files and Directories**

Auditing files and directories allows you to track their usage. For a particular file or directory, you can specify which groups or users and which actions to audit. You can audit both successful and failed actions. To audit files and directories, you must set the audit policy to audit file and object access. You can select the following file and directory events to audit:

- **Read**: display of filenames, attributes, permissions, contents, and owner
- **Write**: creation of subdirectories and files, changes to attributes, change in contents, and display of permissions and owner
- **Execute**: display of attributes, permissions, and owner; execution of file and changing to subdirectories
- **Delete**: deletion of file or directory
- **Change Permissions**: changes to permissions
- **Take Ownership**: changes to ownership

You will have to determine which directories and files are to be audited on your system, but a good option is to audit “Write” attempts to files in the %systemroot%\system32 folder. To do this, you would select the “Properties” option on the folder, then “Security->Auditing->Add.” Then add “Everyone” and select “Write: success and failure,” do “replace on existing files,” do not “replace on existing subdirectories.”

**Viewing the Audit Events**

You can use the “Event Viewer” for viewing audit events, or there are several third party reporting packages available. I am investigating the EventLog module for Win32 Perl. I have heard a lot of good comments on it, but have not used it extensively.

**Setting Up Auditing: The Specifics for Solaris 2.X**

The auditing package that comes with Solaris is part of the BSM (Basic Security Module) package. The audit daemon auditd is the process that performs the auditing on Solaris systems. It is started by default if the /etc/security/audit_startup file exists. The actions that can be audited are defined in the /etc/security/audit_event file. This file can be customized, but it is very in-depth and beyond the scope of this article (the answerbook has a very good description of the whole process and files involved).

Audit flags indicate which classes of events to audit. Systemwide defaults are listed in /etc/security/audit_control. This file is very important and is the basis for the rest of the discussion. This file is similar to the “User Manager->Policies->Audit” setup in NT; it controls will be audited on the system and what will not. Set up improperly, and you will either have too much information or too little. A `man on audit_classes(4)` will give you a large amount of information. In standing with my initial recommendations, here is an audit_control file to start with:

dir:/etc/security/audit
flags:10,ad,-fm,-fc,-nt
nflags:10,ad
minfree:10

You will have to determine which directories and files are to be audited on your system, but a good option is to audit “Write” attempts to files in the %systemroot%\system32 folder.
Let's see what this means. The first line defines the directory for the audit files to be placed. This location must have adequate space; if it fills up it will lock you out.[1] You can have more than one dir flag in the file, and they will be used in the order specified. The second line is for events attributable to a user, and tells us the following: (see the audit_event file for list of actions that fall into a class):

- lo – all login and logout events
- ad – all administrative actions
- -fm – all failed change of object attributes (chmod, flock, etc.)
- -fc – all failed creation of objects
- -nt – all failed Network events (This may be noisy)

The third line is for events that are not attributable to a specific user. The fourth line tells us the minimum free in the dir files before we get a warning message from the audit system. The audit –s command will cause the auditd to reread this file after editing.

The other file of interest is the /etc/security/audit_user. This file allows more specific auditing of individual users. If specified, the flags in this file are combined with the global flags in audit_control to provide a more granular auditing ability.

**Keeping Solaris Audit Files Manageable**

To keep audit file manageable, a cron job can be set up to periodically rotate the audit files. The audit –n command will checkpoint the logs. This process closes the current audit log and opens another. Then you can process the just closed audit file. Figure 1 is a rudimentary script that will process the just closed audit log. Figure 2 is a script to store and rotate audit logs created during the auditreduce portion, a simple modification of the newsyslog script.

Audit maintenance on Solaris has a steep learning curve, but it flattens out pretty quickly. The best documentation is in the answerbook.

**Conclusion**

Now that my primary job is helping those unfortunate individuals or sites with security incidents, I see the errors in not taking the time in 1991 to “finish the job.” In an incident, if you don’t have good logs (i.e., auditing), you’d better have good luck. The chances of figuring out what happened without good auditing are few and far between. If you take one thing from this article, make it this: Take the time, learn your systems, and set up auditing that is adequate and appropriate for your systems.

In the next issue, I will discuss central management of UNIX and NT audit files.

**Note**

[1] If all audit directories fill, either the “cm” policy must be enabled, or an “audit” account that is not subject to auditable events must exist. See the “Administering Auditing” in the Basic Security Module Answerbook.
Figure 1: A rudimentary script

#!/bin/sh

# Checkpoint the logs so we can reduce them
/usr/sbin/audit -n

# Setup path to search for modified files in, ~ says to ignore this
# file or directory
# excluding /dev/mnt might be too permissive, but it generates a lot of
# "noise" otherwise
srch_path="/usr/sbin, /sbin, ~/etc/uucp, ~/etc/syslog.pid, /etc, /usr/bin,
/usr/lib, /usr/openwin/lib, /usr/openwin/bin, ~, /dev, ~/devices, /kernel"

# get the hostname
host='uname -n'

# Want to be able to clobber /tmp/foo if already exists
unset noclobber

# Setup the header in the file echo
"$host Login/out event" >> /tmp/foo
echo " " >> /tmp/foo
echo " " >> /tmp/foo

# Use auditreduce to get the information, and praudit to clean it
# up. Will give us a listing of all login/logout events.
/usr/sbin/auditreduce -C -c lo | /usr/sbin/praudit >> /tmp/foo
echo "================================================================" >> /tmp/foo

echo "$host Prom event" >> /tmp/foo
echo " " >> /tmp/foo
echo " " >> /tmp/foo

# Will give all events associated with PROM events
/usr/sbin/auditreduce -C -m AUR_EXITPROM | /usr/sbin/praudit >>
/tmp/foo

echo "================================================================" >> /tmp/foo

echo "$host Boot event" >> /tmp/foo
echo " " >> /tmp/foo
echo " " >> /tmp/foo

# Will give all events associated with system boot
/usr/sbin/auditreduce -C -m AUR_SYSTEMBOOT | /usr/sbin/praudit >>
/tmp/foo

echo "================================================================" >> /tmp/foo

echo "$host File Mod event" >> /tmp/foo
echo " " >> /tmp/foo
echo " " >> /tmp/foo

# Will list all File modification events for file in $srch_path
/usr/sbin/auditreduce -C -c fm -o path=$srch_path | /usr/sbin
/praudit | grep "path | sort | uniq" >> /tmp/foo

echo "================================================================" >> /tmp/foo

echo "$host Admin event" >> /tmp/foo
echo " " >> /tmp/foo
echo " " >> /tmp/foo

# Will list all administrative events
/usr/sbin/auditreduce -C -c ad -o path=$srch_path | /usr/sbin
/praudit >> /tmp/foo

# Mail the file to the person watching audit logs
/bin/mail logwatcher@audithost.ntsinc.com < /tmp/foo
rm /tmp/foo
by John Sellens

John Sellens has recently joined the Network Engineering group at UUNET Canada in Toronto after 11 years as a system administrator and project leader at the University of Waterloo.

On Reliability – What About Yourself?

In past articles on reliability, I’ve talked about general principles of reliability, computing hardware, networking, and some aspects of system administration. Most of those things are really quite tangible – if you can’t put your hands on them physically, you can at least copy them to a printer or a tape drive and hold them in your hands that way.

Since I wrote the last article (for the December issue, publishing deadlines being what they are), I’ve been to the 11th LISA conference in San Diego, where we spent a lot of time (more than usual) talking about management, motivation, and people issues. Since returning home, I’ve found myself doing some reading on management and people and thinking more about the people issues that we face in our jobs (and other activities). And I’ve spent a heck of a lot of time in meetings, working with people, and thinking about motivation, coordination, and how people can really enjoy their work.

So I find myself here with my laptop on my daily commute on the intercity bus and with the Christmas holidays and a new year looming up before me, composing a reliability article with a different flavor this month. I’m compelled to consider, from a purely amateur point of view, personal reliability. By that I mean to consider how we interact with our co-workers, vendors, customers, and, to a lesser extent, friends and families. How does one act “reliably”?

How is this relevant to system administrators and computing professionals in general? How does this help to make our computer systems and networks run better and more effectively? System administration is very closely tied to personal interaction, with individuals and with groups, and sometimes with people that you will never see or talk to.
directly. I'll try to give a few examples of why I think that is the case and why reliability and trust are important.

System administration is a service activity – we supply the computing resources so that other people can do their work (or play). We solve problems for people, we design systems and software to serve people, and we help people learn to accomplish their computing tasks in the most effective ways. Any time we install a new command, send out a notice or advisory message, or answer the phone on the help desk, the underlying end product is (almost always) a service for some person or group. When we take a system down for maintenance, submit a request for more funding for more equipment, design a mission-critical computing environment, start fixing a computer or network problem, or propose a solution to suit someone's needs, we're asking for trust: trust that we are using good judgment, trust that we are knowledgeable and competent, and trust that our intentions are good. In short (and I'm sure you've been waiting for this), we are asking others to rely on us. And that's where reliability comes into things this time around.

Why is it important to be reliable? Quite simply, if we are to call ourselves "professionals," we must rely on our reputations, and the most important part of a (positive) reputation is the trust that people can place in us, our judgment, and our abilities. If we cannot be relied upon, all of our experience and abilities will be far less valuable to our customers and co-workers. The ability of others to rely on us is the foundation of the value that we bring to the profession of system administration.

How do you demonstrate your reliability? How do you earn the trust of your constituents? I think the most important piece of advice is to avoid the "us vs. them" mentality that we see (or hear about) all too often. Recognize that you and your users are (or should be) working toward the same goals and toward the success of your enterprise. Although the goals and needs of different groups sometimes seem to be at odds, a little goodwill and effort to understand will make it far easier to work together toward the best solutions.

Consider the other people in your organization, and work to understand their concerns and needs. System administration is not done in a vacuum – a system is only as worthwhile as the systems and solutions that it provides. A beautiful, carefully designed, "perfect" computing system is useless if it is conceptually pure but unsuited to solving the problems at hand.

When interacting with customers or others in your organization, be honest and open. If there's a problem, admit it; and if it's a result of something you did (or didn't do), own up to it, and take responsibility. Any short-term pain will be far outweighed by the long-term gain as your users trust and rely on you. Say what the problem is (or was) and what you've done to keep it from happening again. Give advance warning when you're about to change something, and be realistic about expected downtimes. And remember to follow through: do what you said you would do, when you said you would do it. And finally, be proactive: talk with your users, solicit their feedback and concerns, and act on them. Earn their trust, and you'll be far better off in the long run.

And if the word "lusers" is a part of your vocabulary, you might want to reconsider your use of it.

If you're a manager or leader of system administrators, can the people in your group rely on you? Are you supportive, understanding, fair? Do you send people home when they are sick, or do you tell them to "tough it out"? Are you an advocate for your co-workers? Do you defend them if they're being attacked (deservedly or not)? Do you
champion them in interactions with other groups and higher-ups? Do you fight for appropriate conference and training budgets and extra pay or comp time when they work overtime? Can the people in your group rely on you? And finally, allow me to offer some words from Dee Hock, founder and CEO emeritus of Visa: "If you don't understand that you work for your mislabeled 'subordinates,' then you know nothing of leadership. You know only tyranny."

When I started in system administration years ago, I spent a lot more time concentrating on my "relationship" with the machines. These days, I spend a lot less time dealing with machines and a lot more time dealing with the people who surround them. I'm starting to learn which of those relationships is the more complicated and the more rewarding and where the true value and the true satisfaction lies. (The machines really don't care whether I'm reliable or not, so long as I keep the AC power coming and the backup tapes loaded.)

Well, that's enough of that. I suspect that I've been "preaching to the choir" a little bit here. Next time I promise something a little more concrete that you can sink your teeth into: backups, restores, and disaster recovery.

---

ToolMan: Upcoming Tools; Analyzing Paths

It's a new year, a new volume of :login:, perhaps time for new resolutions. A resolution I made late last year was to incorporate tools from other tool makers into ToolMan articles, primarily to keep the series more useful and interesting. Toward that goal I've included tools by two of my co-workers in the previous and current issues. I'll be writing about tools by people from the wider community in future articles, though I've found that working out the details in these situations takes much longer. But a few things are in the works, so please stay tuned.

Some topics I'm considering for future articles include:

- email folder processing
- RCS/SCCS wrappers
- accounts
- tar wrappers
- netgroups
- backups
- disk quotas
- directories/files
- disk space
- processes
- finding
- searching/replacing
- comparing
- sorting
- text manipulation
- printing
- documenting
- email alias parsing
- remote execution
- scheduling/calendar
- Web (of course)

In other words, the possibilities are wide open. Tools relating to these topics might be geared toward system administrators or general users. When possible, I'll survey several tools relating to a given topic. If you have any tools that fit this list or other categories that you would like me to include, please send a note.
Analyzing Paths

This issue I’ll present a couple of tools for analyzing paths. One resolves symbolic links; the other shows status information for each component of a path (and follows links, too). Both are time savers in relevant situations.

**Resolving Symbolic Links: reslink**

As filesystems evolve and grow in complexity at your site, the tangle of symbolic links can become quite intractable. Sometimes you need to see where a path really goes, and how, in order to understand some situation or problem. Getting this information quickly, easily, and reliably would be nice.

Yuji Shinozaki, a fellow sysadmin here in my department, has written a tool named reslink for just such situations. It’ll follow the links to a file (or some other filesystem object) and display various information about the links, depending on which options you choose.

For instance, at our site, `/usr/local/bin/` contains symbolic links to files actually residing in other filesystems. Sometimes these links, or various intermediate components, are themselves symbolic links and, well, you get the idea. It can become difficult to trace and grasp any particular one. reslink is an ideal tool for dissecting this maze of links.

reslink by default will list just the final path to the object specified as an argument. This can be useful in command substitution situations. For example:

```
% reslink /usr/local/bin/latex
/auto/pkg/tetex-0.4/tetex/bin/virtex
% ls -l "reslink /usr/local/bin/latex" -rwxr-xr-x 1 lab lab 201548 Aug 7 1996 /auto/pkg/
/tetex-0.4/tetex/bin/virtex
```

With the `-t` (trace) and `-v` (verbose) options, details of all the links are shown:

```
% reslink -tv /usr/local/bin/latex
/auto/pkg/tetex-0.4/tetex/bin/virtex
/auto/local/bin/latex -> /auto/pkg/tetex-0.4/bin/latex
/auto/pkg/tetex-0.4/bin/latex -> /usr/pkg/tetex-0.4/tetex/bin/latex
/auto/pkg -> /auto/pkg
/auto/pkg/tetex-0.4/tetex/bin/latex -> virtex
```

Another handy option is the `-w` (which) option, which simulates the which command:

```
% reslink -w latex
/auto/local/bin/latex -> /auto/pkg/tetex-0.4/tetex/bin/virtex
% ls -l "reslink -w latex" -rwxr-xr-x 1 lab lab 201548 Aug 7 1996 /auto/pkg/tetex-0.4/tetex/bin/virtex
```

There are a few other options (both real and planned), but I won’t go into the details here. You can pick up a copy for yourself and play around with it.

The O’Reilly book *Programming Perl* by Larry Wall and Randal Schwartz includes a Perl script named `s1` that is similar to reslink, sans options. `s1` is available from your favorite CPAN site (paths vary) at `<ftp://.../scripts/nutsell/ch6/s1>`. It is also described in *UNIX Power Tools* by Jerry Peek, Tim O’Reilly, and Mike Loukides, and is available on the included CD archive and at `<ftp://ora.com/published/orielly/nutsell/learning_perl/examples.tar.gz>`.
Seeing the Components of a Path: seepath

Sometimes you might need to see a little more about what's going on with a path. seepath is a tool for discovering problems with permissions and modes in a path by giving long-listing (ls -l) style status information on each of the path's components. For example, Shirley may tell you that she's running the new sizzle program and it's dying with the message sizzle: cannot open /usr/project/sizzle/fizzle/drizzle. To make a long story short, you could do the following:

```bash
% pwd
/usr/project
% seepath sizzle/fizzle/drizzle
seepath: /usr/project/sizzle/fizzle/drizzle
-dwcr-xx-x 27 root root 1024 Sep 29 16:21 /
dwcrwxr-x 35 root sys 1024 Nov 13 09:43 usr/
dwcrxrw-r 2 root root 3 Dec 6 15:36 project/
dwcrwxrw-r 34 ziggy eng 1024 Dec 2 15:53 sizzle/
dwcrw-x 6 ziggy eng 512 Nov 20 17:46 fizzle/
-rwxrw-r-- 1 ziggy eng 2674 Nov 5 19:14 drizzle
```

% groups shirley
user acctg mgmt

The problem is now apparent: Shirley cannot access the fizzle directory.

seepath can also follow links and in fact will do just that when the -F (Follow) option is chosen. To use the example from the resmlink discussion:

```bash
% seepath /usr/local/bin/latex
seepath: /usr/local/bin/latex
-dwcr-xx-x 27 root root 1024 Sep 29 16:21 /
dwcrwxr-x 35 root sys 1024 Nov 13 09:43 usr/
dwcrxrw-r 16 lab lab 512 Nov 13 09:43 local/
dwcrwrx-x 2 lab lab 16384 Nov 13 17:30 bin/
lrwxrwxxw 1 lab root 29 Nov 13 09:41 latex -> /
/auto/pkg/tetex-0.4/bin/latex
```

%seepath -F /usr/local/bin/latex
seepath: /usr/local/bin/latex
-dwcr-xx-x 27 root root 1024 Sep 29 16:21 /
dwcrwxr-x 35 root sys 1024 Nov 13 09:43 usr/
dwcrxrw-r 16 lab lab 512 Nov 13 09:43 local/
dwcrwrx-x 2 lab lab 16384 Nov 13 17:30 bin/
lrwxrwxxw 1 lab root 29 Nov 13 09:41 latex -> /
/auto/pkg/tetex-0.4/bin/latex

dwcr-xx-x 27 root root 1024 Sep 29 16:21 /
dwcrwxr-x 35 root sys 1024 Nov 13 09:43 usr/
```

```
Time for a little filesystem reorganizing, eh? With no arguments, `seepath` defaults to analyzing the path to the current directory.

**Tools You Can Use**

`xreslink` and `seepath` are tools you can use to help diagnose problems with paths, in regards to both symbolic links and permissions/modes. Because they can also dramatically reduce the number of commands you need to type in certain situations, they might even help delay the onset of carpal tunnel syndrome! (Sorry, this does not constitute a warranty.)

If you see any ways in which these tools can be improved, the authors invite your comments. And remember, if you have tools that are worth sharing, please be sure to contact ToolMan.

---

**SAGE Award to O'Reilly & Associates**

At LISA '97, Tim O'Reilly accepted the inaugural SAGE Vendor Special Achievement Award on behalf of O'Reilly and Associates. This award is presented to a commercial entity that SAGE wishes to recognize for its contribution in the overall field of Systems Administration.

The award reads:

*The Inaugural Award we give, with thanks, to O'Reilly and Associates who for many years have been publishing consistently top-quality books which span the realm of systems administration and have proved invaluable to a generation of sysadmins.*

---

**Got a tool that's useful, unique, way cool? ToolMan will make you famous! Please send a description to <toolMan@usenix.org>**
Ye gods! Not long ago, someone called me an “old geezer.” I’m not even 50 yet, although the gray in my sideburns makes me look older than I am (right?). Nor do I feel as old as other real UNIX old-timers, who actually worked with UNIX in the 1970s. Dave Korn presented two slides during the NT conference that listed all the operating systems he had worked with before he was exposed to NT. My experiences with computers have been much more modest, and even humorous at times.

For example, I was in sixth grade the first time I got to touch a computer. As part of a science fair project, I was taken to a computer center in Rockville, Maryland, and allowed to marvel at the huge machines: disk drives the size of washing machines, whirring drums, refrigerator-sized magnetic tape drives, and a central processing unit with neon display lights showing the current address and data on the busses. As we were leaving, the operator stopped us. He had something he wanted to show us that he felt would really impress us. He loaded a deck of punched cards, and a tinny speaker on the side of the console started to play music. Pretty impressive.

Actually, I was impressed. I imagined having my own computer someday, which would of course take up an entire floor. Later, I think I appreciated the filtered, dry, air conditioned air in the computer room as much as I appreciated the computer. The project I was assigned was to write a statistical program in IBM assembler. I balked. It was the concept of floating point routines that really had me floored.

Bootstrap
When I finished my freshman year of college, I got a summer intern position with General Electric in Bethesda, Maryland. I was the program librarian’s assistant and not required to do any programming. But I did get to operate the mainframe, an early, dual-processor GE 275. It took about as long to boot as my NT system today, but was much more interesting. First, you manually loaded two punched cards that were coded in binary (not Hollerith) and called “lace cards” because of all the holes. These cards contained the program that would then load the rest of a card deck (about 200 cards). That program would start the terminal controller, which read a paper tape and provided a program that could read a magnetic tape. The magnetic tape actually contained the operating system, which, when loaded into core, finished by copying utilities onto the hard drives. This took about five minutes.

Well, yes, NT gets the old, 66 Mhz 486, so it does take a while to load (especially when you include starting up the desktop).

College programming classes in those days required the use of punched cards. Students were required to wait in line in basement rooms until a keypunch, an IBM innovation, was available. Then they could punch in the text of the programs in their assignments. Keypunching required precise typing; you could not backspace, but had to retype any card (one line of a program) with a single error in it. You submitted the completed deck to the priests running the mainframe, and the results, in the form of a 14-inch-wide printout wrapped around the card deck, would arrive several hours later. If you were lucky, this would include a core dump, which would provide you with clues about what went wrong. Then you could go through the process again. A single typo could cost you from three to as many as 18 hours (at semester’s end) of elapsed time.
Among the disadvantages of card decks was the potential for dropping them. Bent cards were a problem, but this was nothing compared to putting a deck of several hundred or more cards back into perfect order.

Crash
I went back to the university for a couple of courses in 1978. People were still using punched cards, but you then had the option of using DECwriters – 300 baud teletype-writers, very advanced. There were one or two “glass terminals,” but they were a bit scary, and there was no hard copy allowing you to review your command history.

My most embarrassing moment came at the beginning of the semester. We were to enter an assembly program for a lab DEC PDP-11 computer from a listing, and I naïvely had entered the octal memory locations along with the assembler code. Duh! Well, I can fix that, I thought, just a little quick substitution using the line editor. I hit return, and nothing happened. Soon other people began to get up and walk away from their terminals, and I began to look closer at the command I had just entered. Instead of deleting the first column of numbers, I had entered a recursive command that would “never” end. I had crashed the mainframe.

I have often wanted a front panel for my computers. Something about being able to enter machine code in binary, and to watch lights flicker as a program executes, still grabs my fancy. Then again, my desktop machine is about a thousand times faster than that 1960s mainframe I remember fondly, and the lights wouldn’t even appear to flicker. Perhaps that PDP-11 emulator that will run under UNIX could use a front panel?

Crystal Ball Redux
Another year has ended. I just reread the column I wrote at this time last year and can’t say I was displeased. As predicted, I have been forced to learn more about NT and still can’t say I like it. Although I am impressed by some aspects of NT, an operating system and applications hegemony written by a “team” of 8,000 programmers suffers, not surprisingly, from a lack of consistency. And can anyone be surprised to hear that the release of NT 5.0 has been delayed, likely to 1999?

I mentioned that I expected microkernels to be on the ascendant. I have learned about how the design of Mach influenced the designers of NT, in particular, in the area of using subsystems to provide support for several APIs. I also feel vindicated in learning that Sun has purchased Chorus, the major microkernel vendor. Although the current code base for Sun’s Network Computer has been Solaris, I fully expect a microkernel design in the near future.

Java has been plugging along, enmeshed in the politics of “standards.” Sun, for its own reasons, wants to maintain ownership of the Java standard – something I really don’t fault, because they have played pretty fairly so far. Microsoft is being sued by Sun to remove the Java branding from Internet Explorer; Netscape has already removed the branding because it fails total compatibility in four small areas in version 4.0. But I can sense the groundswell of support growing for Java among large commercial users who are attracted to its write-once-use-anywhere promise, reusable components, and fear of Microsoft.

Superhighway
IPv6 is in its early implementation stage. The ebone exists, and router vendors are beginning to support IPv6, although I have yet to hear of a large commercial installation using it for anything other than small-scale testing.
I don't need a $5,000 multimedia-faster-than-a-desktop-hunk laptop with a battery life of 40 minutes. But then, I still use vi for "word processing."

The Internet had several meltdowns this year, including Network Systems butchering the root nameservers and UUNET throwing monkey wrenches into its own backbone routing tables. Nobody even talks anymore about the growth of the Internet; it is just accepted as commonplace, without reliable quality of service, and not apt to be replaced by anything anytime soon. One bright spark on the horizon for organizations will be DSL, a means of using pairs of the Telco's copper wire loops to support digital transmission of up to 6 megabits per second.

Intrusion Detection Systems (IDS) have become the rage. Although they will be great at augmenting firewalls and watching internal security, another trend will make them less useful. We are moving away from broadcast-style networks to switched networks. Using switching means that each host has a "private line," instead of a shared media, for communicating, with the switch acting as mediator and buffer. This means that the IDS people cannot attach to a network and listen to all the traffic, looking for intrusion signatures or unusual behavior. At best, they can monitor individual ports or the connection between backbone routers.

I am still waiting for the laptop of my dreams. It will have a real keyboard, decent-sized display, eight hours of battery life, and weigh less than two pounds. And it won't run Windows or Windows Lite (CE) or worry about supporting Microsoft products. I need to respond to email, take notes, and use the Internet while travelling. I don't need a $5,000 multimedia-faster-than-a-desktop-hunk laptop with a battery life of 40 minutes. But then, I still use vi for "word processing."

Speaking of MS products, I was forced last year to use an LCD projector instead of overheads for a course I was teaching. The expectation is that everyone who presents uses a Microsoft product -- the same one that began controlling itself several times during the NT conference, much to my amusement. I decided to convert my troff-formatted course notes into a simple HTML document, which could then be displayed using Netscape. This worked well, although someone complained that he could still see the browser's controls (unlike MS Presents, which hides everything until you need to restart the presentation). I think a great slide presentation program could be based upon classes taken from the HotJava browser or something like it.

One Potato, Two Potato

And while gazing into my crystal ball, I like to muse about the Microsoft Worm. Nope, probably hasn't happened yet. But as the number of installed NT servers reaches a critical mass, another Internet Worm--like incident will become likely. Just like the Irish potato famine was caused by reliance on just two species of potatoes, having lots of identical servers, internally complex beyond accurate documentation, can lead to a very interesting security meltdown.

Life is interesting, I am traveling less (thank God), and I still find myself looking forward to new developments. I hope the new year finds you better off than last year and also looking forward to the future.
Interview with Bill Cheswick

Rob You've been at the labs for many years. How did you get to your current position?

Bill I was hired as a system administrator: what a great place to learn things. The “less-is-more” approach to programming and system design appealed to me greatly. Subsequently, I met Norman Wilson at a Decus conference, and we became close friends. He was an important link to the labs.

By the way, I strongly recommend that engineers insist on attending a couple conferences a year so they can rub shoulders with leaders in their field and get a good perspective of the issues in their area. This can be negotiated with prospective employers during the hiring process.

I was a system administrator and postmaster for the Computer Science Research Group for several years. I relieved Dave Presotto of the postmaster job because I wanted to learn the ropes on email and the emerging Internet. I also took over the firewall he had built. This I redesigned and re-implemented several times.

By the mid-1990s it was clear that I was more useful as a consultant and speaker than as a postmaster. Bob Flandrena and Paul Glick now handle this unenviable task.

Rob And your current position is in Lucent, right? How has the split-up affected your job?

Bill Yes, I stayed with Bell Labs, which is part of Lucent. I stayed in the same office and company after the AT&T/Lucent split. Many of my friends and colleagues went to AT&T Research, and I miss them. I am glad I stayed with the hard scientists, though.

My work is about the same. Lucent has seemed to be much more eager to develop our projects than AT&T was. I think the folks at Basking Ridge (AT&T) may have mistrusted the labs a bit. (For example, I couldn't sell anyone on a firewall product back in 1991.) Lucent management made Murray Hill the corporate headquarters, and it is clear to me that they have been using the labs a lot more. For example, the patent office has snatched up a couple of ideas I gave them a couple years ago.

There's still plenty of basic or long-term research going on, and I think we have given up trimming the Physics staff.

Rob How do the book sales? Are you chasing the wily hacker, new edition, any time soon?

Bill We are at a slow exponential decay on book sales. Steve and I have been working on the second edition, and clearly some stuff is really dated. For example, the first edition says that email is the primary reason many people connect to the net.

The general stuff is still good, and we are focusing a bit more on that. Also, firewalls aren't quite the same thrust now: they are a useful tool, but there's lots of other aspects to Internet security.

So the second edition is coming along, but I wouldn't hold your breath: we both have a lot of other things to do.

Rob How long does it take to assemble a book like yours?

Bill Cheswick logged into his first computer in 1969 and has worked on operating system security for more than 25 years. Since joining Bell Laboratories in 1987, he has worked on network security, PC viruses, mailers, the Plan 9 Operating system, and kernel hacking. With Steven Bellovin, he co-authored Firewalls and Internet Security: Repelling the Wiley Hacker.
Bill Cheswick

"When the book is done, I may work on dnspoxy and its relation to DNSsec. There are lots of things to do. A simpler ssh? Write the old blit games in Java or Limbo? I have far more ideas than I have time to work on them."

Bill

Months and months. It's like an English assignment that never goes away. The good news is that I am usually writing about something I understand, which wasn't true in English class. Sometimes I'll come to a section and realize that I don't know what I am talking about. I have to take a break and spend some time coming up to speed. For both Steve and me, the consequences of the first edition are taking a lot of our time, so progress is slow.

Arno Penizia, our Nobel prize-winning former VP, said that mundane work forces out creative work. I remember this and try to focus on writing, but email is seductive. I have to ignore the world for a while to get work done.

Basically, I have to quit whining and get to work.

Rob

What interesting projects are you working on these days?

Bill

Not much, actually. The book is job one, officially. But I have spent time as a poster boy for Lucent, taking junkets here and there. I am recovering from foot surgery, which took a lot of time, and the physical therapy still does.

When the book is done, I may work on dnspoxy and its relation to DNSsec. There are lots of things to do. A simpler ssh? Write the old blit games in Java or Limbo? I have far more ideas than I have time to work on them. Arno says this is a good sign.

Rob

You've been in and around so many neat projects. What's the coolest technology that you've seen recently?

Bill

Hmm. Submillimeter radar mapping comes to mind. You get an instant topo map of an area. A satellite can take a picture of earth-deformations around an earthquake zone, accurate to about a millimeter.

Genome summaries. The journal Nature has published the source code for two different bacteria in the past year. We understand what only about half of the proteins do right now, but it is way cool to see the summaries so far: these proteins scavenge iron, this one pumps arsenic out of the cell, these are involved in DNA repair, etc., etc. You don't have to be a molecular biologist to find these true nanomachines interesting. If you are tired of amateur hackers, go learn some chemistry and do some real computing.

I'll put in a plug for Inferno here. It took me about two days to get up to speed on it. The cool thing about less-is-more programming is there is much less to learn. The two complete Inferno manuals are smaller than one Idiot's Guide to Using Windows 95.

At the Hackers conference last month, there was a motorized hobby horse that simulates the motion of the Loma Prieta earthquake. I dropped a quarter in to check out the motion before spending another two bits to actually ride it.

GPS still amazes me, and now it is under $100, $125 with CD-ROM map. Who'd have thought that the speed of light could be so manageable?

Chips that perform bulk analysis of DNA sequences and proteins will change the world of diagnostics over the coming years. And yes, I really do want to know if I am prone to a particular disease. The human body needs a good mechanic. I regret that I will never know the results of my own autopsy.
Rob Any observations on the communication industry and where it's headed?

Bill Other than predicting the Y2K problem in 1970, I haven't been very good at prediction. I'll take a couple easy (and obvious?) shots:

- The net will keep growing, but at a moderated rate. Duh. So will most leading network companies, which remain a fine place to invest your savings.

- Internet telephony (and video) are doomed to remain small potatoes as long as the Internet retains its current general configuration and technology. Neither is likely to change quickly. The IP model was not designed to handle "isochronous" data, as the phone system is. An empty Ethernet or backbone can handle voice pretty well, but there is no financial incentive for the ISPs to deploy enough extra capacity to handle it well. You will still be able to call Bolivia on Sunday morning over the Internet, but I don't think you will want to on Wednesday afternoon. It will sound like the mbone.

- I am told that Moore's law is good until around 2010. If so, digital cameras are really going to be terrific in a few years. Sometimes I wonder what it would (will) be like when a $0.02 IC has an IQ of about 70 and a little microphone and speaker. What would an intelligent light bulb do? Would it carry on a discussion with the refrigerator when I am not home? Would we have an ANSI command set for it?

- Crypto will become ubiquitous and strong. Those fast Intel chips are even better for crypto than multimedia.

- The worst effects of the information age won't be drug lords and porn kings with unreadable records. They will be targeted biological weapons. Imagine an airborne HIV with the infectivity of the flu or one that is especially lethal to [your least favorite ethnic group]. These arms races parallel the Internet arms race (and all the other ones), but may turn out to be much nastier. Not a cheerful prediction, and I don't really know how to prepare for this. But I think my children will see it.
the ABCs of TPCs and NT scalability, II

In the special ;login: issue on Windows NT [1], I promised to delve more into my concerns about the comparisons of UNIX and NT scalability that were presented at the USENIX-NT Workshop last August. In this second article, I want to start with the data presented in Figure 1, which purported to show the superiority of NT over UNIX scalability on the common basis of the TPC-C benchmark workload.

![Graph showing throughput vs. processors for Sybase/Solaris and SQLS/Wintel.]

Before doing so, however, I have to assume that most readers are not familiar with the TPC approach to database benchmarking. Unfortunately, there is not enough space to go into great detail about this complex measurement process, so I can provide only the briefest of sketches. The interested reader can find specifics at <www.tpc.org>.

**TPC Road Rules**

Unlike many computer benchmarks (e.g., Dhrystone, Linpack, SPEC), TPC benchmarks do not exist as code that you purchase or download. Rather, TPC provides a (downloadable) benchmarking specification document. Anyone wishing to run the benchmark is free to implement the specification in any way he or she sees fit. You are not free, however, to interpret the TPC rules as you please. In order to report an official TPC result, you must write a corresponding full disclosure report that itemizes how you met each one of the clauses in the TPC specification. In addition, the benchmark runs that produced the result you wish to report must be witnessed and reviewed by an official TPC auditor at runtime. Your disclosure report is also reviewed by members of the TPC council. Any discrepancies that cannot be satisfactorily explained may lead to the result being withdrawn. In other words, TPC benchmarks are a serious and expensive undertaking that come with a high degree of credibility. Any attempt to cut corners is likely to be spotted and dealt with accordingly.

**The TPC Performance Race**

Currently, there are two TPC benchmarks: TPC-C (for benchmarking online database transaction processing: AKA OLTP systems) and TPC-D (for benchmarking decision
support systems: AKA DSS). The TPC-A and TPC-B benchmarks have been retired for two major reasons. First, these workloads corresponded to a relatively simple debit/credit banking transaction. Second, removal stops ongoing attempts to exploit any loopholes in those benchmark designs. Moreover, both the TPC-A and TPC-B were directed solely at OLTP performance. TPC-C is a more complex OLTP benchmark that uses a heterogeneous mix of five transactions accessing a database that models inventory control in a distributed warehouse. TPC-D is the first TPC benchmark to be directed at multi-user, large-scale, query-intensive systems.

Rather than get bogged down in technical details, I've chosen to highlight the difference between TPC-C and TPC-D using the following whimsical analogy with automobile sporting events.

**TPC-C Indianapolis 500**

TPC-C is the Indy 500 of database benchmarking. In the real Indy event, 35 vehicles race around a 2.5-mile circuit and the first car over the finish line on the 200th lap is declared the winner. The sporting focus is on the performance of individual vehicles as measured by their top speeds in miles per hour.

In the TPC-C benchmark, the database transactions are analogous to the Indy race cars, but the performance focus is shifted away from the cars and onto the racetrack itself. For example, a wet track is slower than a dry one. The performance of the track could be measured by the number of cars per minute the track can support over the 200 loops of the Indy 500 race. It is a measure of the raceway's carrying capacity. Technically, this would be accomplished by counting the number of cars that cross the same place (e.g., the starting line) every five minutes (roughly the time it takes a car to make one loop of the raceway) and averaging those counts over the duration of the entire race. Under TPC road rules, any car taking longer than five minutes would not be counted as part of the track's capacity. In the TPC version of the Indy 500, there is another rule that all cars must make at least one simultaneous pit stop (corresponding to a database checkpoint) and then continue again.

In practice, when the checkered flag falls, all the cars take some time to maneuver into position and get up to top speed. In the TPC-C benchmark, this corresponds to the ramp-up period necessary to get the database cache warmed up and the system operating in steady state. This ramp-up period is not included in the performance results. In the real TPC-C benchmark, transactions committed every half minute or so are counted and used to determine the average throughput measured as transactions per minute (or tpmC) over the entire benchmark run. Any transaction that does not commit within a two-second minimum response time is not counted.

**That Transparency Thing**

Furthermore, suppose you wanted to assess the Indy track capability on a worldwide basis (e.g., tracks in the US, Australia, Canada, and Britain). This would be a way to compare Indy racing with other kinds of races (e.g., NASCAR racing). The worldwide Indy performance would be given as the sum of the performance of each Indy raceway.

If you raced only US cars on the US track, Australian cars on the Australian track, and so on, you would be unintentionally optimizing the measurement. The TPC-C version of measuring this worldwide Indy performance does not permit such an optimization. Instead, you must also run some US cars on the Australian raceway, some Australian cars on the British track, and every other permutation in between. Moreover, which car runs on which track must be determined by drawing track-car pairs out of a hat. In other words, you are not allowed to bias the results by knowing beforehand which
In contrast to the TPC-C Indy 500 race, TPC-D is more like a monster tractor pull.

car will race on which track. The selection process is then said to be unbiased or transparent.

Similarly, in the real TPC-C benchmark, you can have four servers with four separate database instances, but TPC-C does not permit you to confine transactions to each database separately and then add the separate throughputs together to give the total capacity. Transactions must be distributed in such a way that any transaction can access any of the four database tables without knowing ahead of time which database it will run against. This adds realism to the benchmark. But transparency can also introduce some performance degradation due to the longer code paths needed to distribute the transactions.

Clearly, it would be much simpler to ignore this transparency requirement and just add up the throughputs of more and more independent servers. That is an easy (but unrealistic) way to generate a big throughput number without any distribution overhead. That's precisely what Microsoft did; but because it violates TPC-C road rules, they could not report it as a bona fide TPC-C result. It would never have gotten past the TPC auditor. Gray claimed this was just a "technicality." [2] Now you can decide. On top of this failure, they didn't use TPC-C transactions either, contrary to the statement in [3]. What did they use? We'll never know because, not running a TPC benchmark, they were not subject to the disclosure rule. Gray used the term "debit-credit transaction," which suggests some kind of banking transaction, but we don't know that.

Instead of saying Microsoft did 1 billion transactions per day, I'd prefer to call it 1 billion diddleleysquats per day just to remind myself that the entire Microsoft claim is beFUDdled.

**TPC-D Monster Tractor-Pull**

In contrast to the TPC-C Indy 500 race, TPC-D is more like a monster tractor pull. In the TPC-D version of the tractor pull, there are 17 vehicles of different weights that the tractor must tow across the arena to complete the competition. For each tow, the elapsed time to get across the arena is measured and used to construct an overall towing capacity for the tractor. There's no constraint on how long it takes to pull all 17 vehicles because only the elapsed time for each pull is measured. In the real TPC-D benchmark, the key performance metric is the time taken to execute each of the 17 queries. Gray did not discuss TPC-D results for SQLServer because there aren't any for SQLServer. You can check for yourself at <http://www.tpc.org/execsum_TPCD.html>. But there are many TPC-D results on UNIX.

**Sensible Scalability Comparisons**

Figure 2 shows a comparison of TPC-C results across a wide variety of results published in 1997. The most important notable difference from Figure 1 is that there are no curves. That is because these are all different platforms running various flavors of UNIX, different RDBMSs, on different hardware. Using curves (as in Figure 1) would erroneously suggest that certain data belong to the same family, when they do not. Recall what I said about the performance analyst's cardinal rule [1]: only change one thing at a time!

There are four CPU categories shown in Figure 2: uniprocessor, two-way, four-way, and six-way multiprocessors. In each CPU category, the UNIX results are grouped to the left while the NT results are grouped to the right. I've selected official TPC-C UNIX and NT results for all of 1997 to give some reasonable definition to my requirement [1] that the data be in some sense contemporaneous. The selected servers have between one and six processors to conform to the range where NT actually tries to compete with UNIX servers.
Things look a little less impressive for NT than in Gray's benchmarking presentation [2]. First, note that there is considerable variance within each UNIX group. This is to be expected because (unlike NT) there is no single UNIX, and the data in Figure 2 include Oracle and Sybase running on various UNIX platforms. Typically, CPUs with larger second-level caches produce higher throughput because they can accommodate a larger RDBMS footprint.

Second, there is far less variation within each NT group. This is to be expected when there is only one RDBMS (viz., SQLServer) tuned to run relatively few Intel-based architectures. Note also that for the six-way configurations, the best UNIX result (HP/Sybase) has more than twice the throughput performance of the NT system, and the next best UNIX result (Sun/Sybase) is more than 30% better than NT. This demolishes Gray's point based on Figure 1 that one needs to go to a more expensive 12-way UNIX system just to match a six-way NT in throughput. How did I arrive at a different conclusion? I didn't bias the data by handpicking aged Solaris/Sybase TPC-C results for making NT comparisons.

Table 1 summarizes the various platform combinations that have been reported. Sequent has announced a parallel query result on a four-node NUMA-Q 2000 cluster with dual-quad CPUs (32 total CPUs). This is not a TPC-D result, however. Also, Compaq has an official TPC-C result with Oracle on NT (third column in Figure 2). There are no SQLServer results on UNIX that I know of.

<table>
<thead>
<tr>
<th>RDBMS \ OS</th>
<th>NT</th>
<th>Solaris</th>
<th>UNIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLServer</td>
<td>✓</td>
<td>x</td>
<td>?</td>
</tr>
<tr>
<td>Sybase</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Others</td>
<td>(a)</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Price-Performance Comparisons

We can use the disclosed price of the TPC benchmark platform expressed as $/tpmC to make the comparisons shown in Figure 3. When it comes to price-performance, Microsoft does indeed have the drop on UNIX, especially at the low end. But it's not so dramatic for larger CPU configurations. In case you're wondering, the expensive outlier in the two-way class is a Fujitsu UNIX box.

Open system hardware is generally cheaper than mainframes, so how can Wintel pricing beat UNIX so convincingly? One way of looking at this is to recognize that history is simply repeating itself. Over the last 20 years, UNIX workstations and multiprocessors have eroded the profit margins that were sacred to selling mainframe big iron. This occurred because UNIX boxes were cheaper to build and became more ubiquitous than centralized mainframes. At the outset, they could not compete with mainframe performance, but gradually, that changed as UNIX systems scaled up.
the PC shall do unto UNIX servers what UNIX servers hath done to mainframes.

Over the last ten years, the PC has become more ubiquitous than UNIX servers. They represent real commodity computers. At the outset, they could not compete with UNIX workstation or multiprocessor performance, but gradually, that is changing as PC-based systems scale up. In other words, the PC shall do unto UNIX servers what UNIX servers hath done to mainframes.

Next time, I'll consider the factors that determine hardware scalability.

Acknowledgments
I am grateful to Kim Shanley (TPC CEO), Francois Raab (TPC auditor), and Mike Brey (Oracle) for various technical discussions.

Notes
Using Java

Although I have tried to avoid it, I find myself writing about Java Beans. The migration to JDK version 1.1 brought many changes and new classes, among them Beans. I was at first overwhelmed by the amount of change in 1.1, as were some people I spoke with. But when I finally looked at Beans, I discovered they can actually be quite simple. In fact, a Button from the AWT could be used as a Bean or even a completely non-graphical class, like the following.

```java
public class ABean implements java.io.Serializable {
    protected int aValue;
    public void setAValue(int value) {
        theValue = value;
    }
    public int getAValue() {
        return aValue;
    }
}
```

Not very exciting, but still a Bean. The Java Bean white paper describes a bean as a reusable software component that can be manipulated visually in a builder tool. I haven't (apparently) done anything special in the ABean class. Actually I have, and it relies on two aspects of Java, one new to 1.1 and both common in object design.

The first trick has to do with reflection – the ability of a programming language to examine itself. The java.lang.reflect package includes classes that permit Java code to explore the structure of a class (under limits imposed by the security manager). Because each class includes all the information needed to be dynamically loaded and used, this should not be too surprising – you can think of it as a type of object debugger. But instead of just debugging, the reflect package lets you list the variables, methods, and constructors of a class. Reflection is also used in Java serialization, the technique used to create persistent Java objects.

The Introspector class, included with the Beans class library, uses reflection to extract information from a Java Beans class. You don’t have to write this class, and every tool vendor can use the same class to analyze a Bean.

What the Introspector abstracts from a Bean relies on design patterns, the second aspect I alluded to. Methods beginning with set (and returning void) and get are extracted, the get/set deleted, and the remainder of the Method name represents a property of the Bean. In my trivial example, the only property is AValue. Properties can be read-only (only a get method) or even write-only. The 1.1 AWT components are all written using this design pattern, so the foreground and background colors and fonts are properties of these components. (Events are also extracted, but more on that later.)

You can ignore these patterns if you wish by building a companion class that implements BeanInfo. The BeanInfo class is separate from the Bean itself. It is used while manipulating the Bean but is not required in a finished application, when it would just be additional baggage.

**Bean Box**

A better way to get a handle on Beans is to download the BDK, the Bean Developer’s Kit (<java.sun.com/beans/software>). You will need a 1.1 version of the JDK (I used JDK 1.1.4, which differs from 1.1 in that it includes various bug fixes). At this time, only the HotJava browser fully supports 1.1, which means that even if you create Bean applets, you can view them only with HotJava or the 1.1 appletviewer. There is also a short
tutorial that guides you through using the Bean Box, a simple visual developer tool.

Part of what’s neat about the Bean Box is that although it is a bare-bones tool, most of the functionality is included in the Beans library. Starting up the Bean Box produces three windows, a palette, an empty container where you can drop the Beans, and a property window. The container window includes a menu bar, so please don’t be surprised when I mention a File menu.

You add Beans to the container by clicking on one in the palette, then picking the center of the component in the container and clicking again. This action selects the new Bean and converts the third window into a display of the editable properties of the Bean. For example, if you choose the first Bean, OrangeButton, four properties will be displayed: foreground color, label, background color, and font. Clicking on the color properties brings up a color selection window, and clicking on the font lets you pick a font.

The properties windows is part of a built-in class that any Beans development tool can use. If you build a Bean with properties that are more complex than can be handled with a simple properties editor, you can build customized property editors and step your developers through the process of editing the properties.

**Events**

Now you have a customizable OrangeButton, but what can you do with it? Well, suppose you have been experimenting with the other beans and have placed a JugglerBean into your container. The JugglerBean begins juggling, which can be quite annoying. You can slow down the juggler by increasing its delay property, but you can’t stop it. But there is a way, which illustrates another important aspect of Java Beans.

Select the OrangeButton, then open the Edit menu (next to File in the menubar), and select Events, button push, actionPerformed. You have now chosen to add a component as the target of this event, and a red line will follow your cursor until you click on another component. Pick the juggler.

Ah. Now a new window appears (the Event Target Dialog), which permits you to select an event listener method on the juggler. Select stopJuggling. The window changes to a message that reports that a new adaptor class is being generated. Once the window disappears, click on the OrangeButton and the juggler will stop.

You use events to communicate between Beans. You can have invisible Beans, like the TickTock, which comes with the Bean Box and fires off TimerEvents. You can create your own invisible Beans, which can listen for events or property changes and fire off events in return.

You can customize the Bean Box by creating your own Beans, writing a manifest, collecting the Beans into a jar file, and loading this jar file using the File menu in the Bean Box’s container window. And if you really like what you have accomplished, you can save the container, along with its customized Beans, by serializing them (saving them as a file).

This column is no more than a basic introduction to Java Beans, just to provide a few of the capabilities. In subsequent columns, I plan on presenting example Beans that can be used in the Bean Box and will actually do something (other than juggle). If you can’t wait, O’Reilly’s Exploring Java (Patrick Niemeyer and Joshua Peck) has a chapter on Beans that is sufficient to get you going, and their Developing Java Beans (Rob Englander) goes into 300 pages of excellent details.
using C++ as a better C

In this column we'll look at C++ replacements for malloc() and free(), the operators new() and delete().

To give an example of how these are similar and how they differ from their C counterparts, suppose that we want to allocate a 100-long vector of integers for some purpose. In C, we would say:

```c
int* ip;
ip = (int*)malloc(sizeof(int) * 100);
...
free(void* ip);
```

With new/delete in C++, we would have:

```c
int* ip;
ip = new int[100];
...
delete [] ip;
```

The most obvious difference is that the C++ approach takes care of the low-level details necessary to determine how many bytes to allocate. With the C++ new operator, you simply describe the type of the desired storage, in this example int[100].

The C and C++ approaches have several similarities:

- Neither malloc() nor new initializes the space to zeros.
- Both malloc() and new return a pointer that is suitably aligned for a given machine architecture.
- Both free() and delete do nothing with a NULL pointer.

malloc() returns NULL if the space cannot be obtained. Many versions of new in existing C++ compilers do likewise. However, the draft ANSI C++ standard says that a failure to obtain storage should result in an exception being thrown or should result in the currently installed new handler being invoked. I assume that NULL is returned.

New Handlers

The idea of a new handler can be illustrated as follows:

```c
extern "C" int printf(const char*, ...);
extern "C" void exit(int);
typedef void (*new_handler)(void);
new_handler set_new_handler(new_handler);
void f()
{
    printf("new handler invoked due to new failure\n");
    exit(1);
}
int main()
{
    float* p;
    set_new_handler(f);
    for (;;) {
        p = new float[5000];   // something that will
        return 0;
    }
}
```
This is an area of C++ that has changed several times in recent years. There are a number of issues to note.

A new handler is a way of establishing a hook from the C++ standard library to a user program. `set_new_handler()` is a library function that records a pointer to another function that is to be called in the event of a new failure.

**Deleting Arrays**

Note that saying:
```
delete ip;
```

instead of:
```
delete [] ip;
```

will work with some compilers in the example above.

This is an area of C++ that has changed several times in recent years. There are a number of issues to note. The first is that `new` and `delete` in C++ have more than one function. The new operator allocates storage, just like `malloc()` in C, but it is also responsible for calling the constructor for any class object that is being allocated. For example, if we have a String class, saying:
```
String* p = new String("xxx");
```

will allocate space for a String object and then call the constructor to initialize the String object to the value “xxx.” In a similar way, the delete operator arranges for the destructor to be called for an object, and then the space is deallocated in a manner similar to the C function `free()`.

If we have an array of class objects, as in:
```
String* p = new String[100];
```

then a constructor must be called for each array slot, because each is a class object. Typically, this processing is handled by a C++ internal library function that iterates over the array.

In a similar way, deallocation of an array of class objects can be done by saying:
```
delete [] p;
```

It used to be that you had to say:
```
delete [100] p;
```

but this feature is obsolete. The size of the array is recovered by the library function that implements the delete operator for arrays. The pointer/size pair can be stored in an auxiliary data structure, or the size can be stored in the allocated block before the first actual byte of data.

What makes this a bit tricky is that all this work of calling constructors and destructors doesn’t matter for fundamental data types like `int`:
```
int* ip;
ip = new int[100];
delete ip;
```

This code will work in many cases because there are no destructors to call, and deleting a block of storage works pretty much the same whether it’s treated as an array of ints or a single large chunk of bytes.

But more recently, the ANSI Standardization Committee decided to break out the new and delete operators for arrays as separate functions so that a program can control the allocation of arrays separately from other types. For example, you can say:
void* operator new(unsigned int) /* ... */ return 0;
void* operator new[](unsigned int) /* ... */ return 0;
void f()
{
    int* ip;
    ip = new int;     // calls operator new()
    ip = new int[100]; // calls operator new[]()
}

and the appropriate functions will be called in each case. This is kind of like defining
your own versions of the malloc() and free() library functions in C.

**Defining Your Own New/Delete Functions**

It is possible to define your own new and delete functions. For example:

```c
void* operator new(size_t s)
{
    // allocate and align storage of size s
    // handle failure via new_handler or exception
    // return pointer to storage
}
void operator delete(void* p)
{
    // handle case where p is NULL
    // handle deallocation of p block in some way
}
```

`size_t` is a typedef, typically defined to mean "unsigned int." It's found in a header
file that may vary between compiler implementations.
The following reports are published in this column:

**What is POSIX ad hoc Meeting**

**Whither POSIX? A Proposal from the Open Group**

Our Standards Reports Editor, Nick Stoughton, welcomes dialogue between this column and you, the readers. Please send any comments you might have to:

<nick@usenix.org>

---

An Update on Standards Relevant to USENIX Members

by Nicholas M. Stoughton

USENIX Standards Liaison

<nick@usenix.org>

---

**What is POSIX?**

Lowell Johnson <LowellJohnson@unisys.com> reports on the November 18, 1997 “What is POSIX” ad hoc meeting in San Jose, CA.

The ad hoc meeting began at 4:30 after various meetings of the IEEE Computer Society were finished. My notes follow because there was no secretary. Considerable discussion had occurred before 4:30, with various collections of people participating.

The problem as I understood it was simply, “What is inside the box we call POSIX?” Various presentations were made by the participants. It is very easy to make generalities that hide the truth. For example, it was claimed that the people working on standards now are not representative of “the community.”

One early proposal was that all existing 1003.x standards be frozen. New work could still be called POSIX, but would have new numbers. Checkpoint Restart might be numbered 1777, but it would still be called POSIX. The frozen set would become the “core POSIX.”

This position seemed surprising to me. There are several standards already published (such as the Realtime extensions) that go beyond the original “core” standards of POSIX.1 and POSIX.2. However, almost everyone agreed that the existing published POSIX standards should be left alone, and new numbers should be used for new standards. Only one person spoke up and said he would “prefer” to roll back to the core standards, but he realized it was impractical to do that because it would be too much work.

---

I think it is safe to say that POSIX is in a period of crisis. Falling attendance and little headway in ballot resolution has put off many of the vendors, and they no longer feel that this is an effort worth following. At the October meeting in Reno, the Sponsor Executive Committee (SEC), which governs the work of the IEEE Portable Applications Standards Committee (PASC), formed two ad hoc committees to examine options for the future, one considering the question “What is POSIX?” and the other looking at possible collaboration with The Open Group (TOG) for future work.

Preliminary reports from both these committees are published in this column.

Whether PASC will survive is at best doubtful. Industry seems to be confused by the existence of two closely related organizations, PASC and TOG, and wants to follow only one. The dilution of the core standards by specialist areas does not help either. I have reported previously on the increase in the proportion of defense-related personnel at POSIX meetings and the building of MIL-SPECs by another name.

If we cannot adapt to our circumstances, we will surely die.
There was a long discussion about how you could have a standalone standard that did something like modify `read()`.
This was thought to be part of the topic of another ad hoc group, but it was agreed that the standalone standards could not change any code in the base standards.

One suggestion was to leave all currently sponsored projects, whether complete or no, in the POSIX circle. However, this was felt to be inappropriate.

A question was directed at checkpoint/restart: how much work would it be to make it a standalone document? The chair said it would break the group, but one member said she did not think it modified any part of POSIX.1. [Editor’s Note: When checkpoint/restart was a part of POSIX.1a, it touched many parts of core POSIX.1, particularly `fork()` and `exec()`]. Several groups need to evaluate how much work it would be to change to a standalone document.

A list of projects to be included in this new core standard was agreed upon:

<table>
<thead>
<tr>
<th>IN</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1003.1</td>
<td>1003.2</td>
</tr>
<tr>
<td>1003.1b</td>
<td>1003.2a</td>
</tr>
<tr>
<td>1003.1c</td>
<td>1003.2d</td>
</tr>
<tr>
<td>1003.1g</td>
<td>1003.1h</td>
</tr>
<tr>
<td>1003.1I</td>
<td>1003.1m</td>
</tr>
<tr>
<td>1003.1n</td>
<td>1003.1q</td>
</tr>
</tbody>
</table>

It was agreed unanimously that a resolution must be based on a list of projects, not on any sort of cutoff date. It was also agreed that a vote will have to be taken at the SEC meeting. Some people need to do some homework to find out how much work it would take to change the proposed standalone projects to be stand-alone documents.

Finally, we debated what the results of this meeting really were, and after considerable debate, the following compromise emerged: I will report on what we agreed and what we did not agree on. At the January SEC meeting, a motion will be made to limit the core POSIX to the above list of standards.

After a reasonable amount of debate, and possible amendments, a vote will be taken. Then I and the appropriate WG chairs will take the necessary action with the affected projects.

All items not included on the list should have sponsorship withdrawn, but they could be resubmitted at the same meeting with a new number (not 1003.x). A final discussion reiterated the requirement that no future standard can break this core POSIX. This was agreed to be an absolute rule, regardless of the result of division issue. This means, for example, that anyone who adds to common functions like `read()` or `write()` must be able to fully document the enhancements without changing anything in the core standards.

**Whither POSIX?**

**A Proposal From the Open Group**

Dr. Petr Janeček <p.janecek@opengroup.org>,
Director of Standards Development at The Open Group, presents a proposal for closer collaboration with The Open Group.

This article is based on a draft proposal for coordination of standardization activities between the IEEE Portable Applications Standards Committee (PASC) and The Open Group (TOG).

The proposal will be discussed by the members of the ad hoc committee established last October by IEEE PASC Sponsor Executive Committee (SEC) and subsequently, if appropriate, by the SEC itself. At this stage, it is merely a proposal for discussion, but views are welcome!

**Objective**

This proposal aims at eliminating duplication of industry standardization efforts in the area of open operating systems and eventually eliminating the need for a single supplier to provide different products complying with two very close but not (yet) identical industry standards, POSIX and UNIX.

**Background**

The interest in standardization of open operating systems has peaked and is rapidly tailing off, with the industry moving its standardization resources into new areas, in particular, the Global Information Infrastructure.

All standards organizations have been feeling a falling interest, meeting attendance, and membership.

The open operating system standards established through the activities of IEEE PASC, in particular the POSIX.1 and POSIX.2 groups of standards, are stable, and there is little industry interest in anything more than their maintenance (i.e., error removal and interpretations). Some niche markets are still interested in profiling and further development of special features of POSIX (e.g., POSIX for embedded systems), but vendors serving the general market do not wish to have to implement such new features as part of their basic operating systems offering.

POSIX has become mature.
The UNIX operating system, the trademark to which is owned by The Open Group (UNIX and the “X” device are registered trademarks in the US and other countries) has become a de facto standard reflecting industry’s choices in practical implementation of POSIX “core” standards. The UNIX definition recognizes the POSIX standards as over-riding ones but is tighter, so that a UNIX-compliant system automatically complies with the POSIX standards while the opposite is not true.

The two standards, POSIX and UNIX, use different style and format, but are maintained by essentially the same limited group of the best of industry experts and are implemented by the same vendors. There is little reason to have the same group of people participating in two different forums carrying out what is essentially the same task.

The time has come to optimize and merge relevant standardization activities of the two bodies, IEEE PASC and TOG, while making sure that no constituency feels disenfranchised.

Industry’s financial support makes it possible for The Open Group to provide professional standardization support services, including full-time managers and editors, as well as Web-based and other publications. Such facilities off-load volunteers from the administrative and routine work and secure a high speed of development.

Industry support also makes possible additional services building on the results of the standardization effort: development of test suites and management of testing services, branding of compliant systems, and professional marketing services. It is therefore felt that TOG could provide a good professional home for future POSIX as well as UNIX standardization activities.

**Options**

Options for collaboration between IEEE PASC and TOG depend on the degree of overlap of the current activities, the two organizations’ plans for future work, and IEEE’s interest in new services. The Open Group can offer IEEE and PASC the following:

- Maintenance services for approved standards (.1 and .2 families)
- Home for ongoing 1003.1/2 projects
- Home for projects in specialized areas
- Testing services Branding services
- Publication Services

**Maintenance Services for Approved Standards**

The following completed standards are in maintenance mode (i.e., error correction and interpretations) and suitable to be handled via an email-based maintenance structure of expert review groups provided by TOG, not requiring physical meetings:

- POSIX.1 System Interface
- POSIX.1b Realtime
- POSIX.1c Threads
- POSIX.1g Protocol Independent Interfaces (Sockets and XTI)
- POSIX.1i Fixes to .1b (Realtime)
- POSIX.2 Shell & Tools
- POSIX.2a User Portability Extensions
- POSIX.5 Ada binding to POSIX.1
- POSIX.9 FORTRAN binding to POSIX.1

The mechanism could be implemented by opening up the membership of TOG base group to interested PASC members as belonging to the existing category of “invited experts.”

**Home for Ongoing 1003.1/2 Projects**

The following core standards are still under development and could be handled via broadened participation in the Base Working Group of TOG:

- POSIX.1a System Interface Extensions
- POSIX.1n Fixes to 1003.1/1b/1i/1c
- POSIX.2b Additional Utilities
- POSIX.18 POSIX Profile

The same mechanism as above, namely opening up the membership of TOG base group to interested PASC members as belonging to the existing category of “invited experts,” can be applied here.

**Home for Projects in Specialized Areas**

The following are some of the 1003 standards under development that cover specialized areas of interest to large and important customers. Although TOG currently has no similar activities, it could handle them through establishing new TOG working groups with open participation. The funding would have to come from the participants paying a meeting attendance fee of the same kind they today pay to PASC.

- POSIX.1d and POSIX.1j Additional Realtime Extensions
- POSIX.1h Fault Tolerance
- POSIX.1m Checkpoint/Restart
- POSIX.21 Realtime Distributed Systems Communications (LIS)
Testing Services
The Open Group has the first complete POSIX Conformance Test Suite family in the industry for the whole of POSIX.1–1996 and POSIX.2–1992.

Further, The Open Group recently introduced Validation Services for FIPS 151–2 (POSIX) in response to the termination of NIST validation services. TOG's service is based on NIST procedures and NIST's PCTS test suite.

Branding Services
The Open Group's UNIX Brand provides to customers a legally binding guarantee of a branded product's compliance with the UNIX definition now as well as in the future. A vendor can obtain as part of the UNIX Brand a certificate that includes the FIPS certification. Branding of POSIX-compliant systems would be possible.

Publication Services
The Open Group makes its specifications publicly and freely available on its Web site. In addition, CD-ROM and electronic and paper publications are provided. The Open Access mechanism currently under development makes access and marking-up of documents over the Web easy and suitable for review by groups of experts.

Summary
Several issues still need to be worked out. In particular, the decision-making process in TOG is based on organization representation, while that in PASC balloting groups is based on individual participation. One possibility to solve this might be that TOG would consider the current PASC members interested in continuing their activities within TOG to be a group with the right to institutional representation (i.e., voting rights) to the Base Working Group. This is exactly akin to the existing system, ISV and customer councils. The members of our Customer, Software Vendor, and System Councils vote in ballot reviews through elected representatives who are charged with representing the respective council’s consensus. My idea is that the POSIX members would become another “council” with one guaranteed vote. Of course, companies that are already voting members do not go through this process.

The other area that requires further thought and resolution is intellectual property rights. All existing material is copyright of the IEEE and is not currently freely available. How do we deal with the issue of TOG modifying this material? How does it get published? I believe these are not insurmountable hurdles.
The bookworm

by Peter H. Salus

Peter H. Salus is a member of ACM, the Early English Text Society, the Trollope Society, and is a life member of the American Oriental Society. He has held no regular job in the past 23 years. He owns neither a dog nor a cat.

<peter@pedant.com>

These last few months there have been several new publications that I considered really outstanding. It may be that books for the thoughtful are being published as an antidote to VMS for Dummies. I don't know the reason, but I'm thankful for it.

I was a speaker at a bar meeting in Phoenix last November. I had thought that the legal community would be interested in porn on the Net, etc. They weren't. They were interested in privacy and in jurisdiction. This last is a clear problem where we're dealing in a many-to-many universe, unlike publishing, radio, or TV, where there is an obvious single source.

Privacy

In the many-to-many world, firewalls can prevent intrusion, and encryption can hinder perusal. But the issue of personal privacy is a much more difficult one. We all know how much information (misinformation?) about every one of us is out there; banking, credit history, motor vehicle, birth-death-marriage-offspring-adorption, property, wills and deeds, etc. - all are among the data available. With a suitable computer, it's easy to suck up stuff and construct a rather elaborate sketch of an individual. Is this new dban invasion of privacy or not? In Europe, it apparently is.

Bruce Schneier and David Banisar have put together a splendid volume on the encryption/public key/Cracker debates. This is a very large (and heavy)

.collection of papers, statements, congressional bills and reports, and newspaper and magazine articles about "the battle for privacy in the age of surveillance." The documents range from Harry Truman's executive order of October 24, 1952, to the present: 45 years of governmental intrusion, rationalized first as national security and more recently as a fight against organized crime. (Our modern Elliot Ness is sitting in front of his keyboards reading messages like "The cash is in the hollow tree.")

There are flashes of humor here, too: Matt Blaze's tale of taking a SecurePhone to Europe is adverting; the FBI's Sensitive Electronic Surveillance Techniques document, four pages of backed-out lines, is inadvertently.

It will take many hours to read all of Schneier and Banisar. Do it. This is an important book. My compliments to the compilers.

Routing

For a long time, I've felt that routing was the neglected child of the Internet. Internet Routing Architectures may not be the end-all of publishing, but it is a very fine beginning. Published by Cisco, this is an excellent presentation of the design considerations of interdomain routing. There is a good deal of space devoted to BGP4 (the Border Gateway Protocol). I may actually have come away with a good understanding. Halabi's opus will be of value in teaching folks about data routing manipulation.

More Programming

The third edition of Knuth's Art, Vol. 2, Seminumerical Algorithms, has plopped itself on my desk. Because I have written earlier that I'm going to wait till all three volumes are out before I devote space to them, this announcement will have to serve for the nonce.
Java
Addison Wesley should get an award for binding the two volumes of *The Java Class Libraries*: over 1,700 pages each! Chan and Lee have served up a detailed, annotated, alphabetic reference with thousands of lines of code examples. This is useful, but at 1,712 pages, volume 2 is not “handy” — nor, at over 2,000 pages, is volume 1. Nonetheless, I consider these indispensable references.

Internet
If you can recall when the Net was free and there were only a few hundred or a few thousand hosts on it, *Coordinating the Internet* may upset you a great deal. Personally, I feel that the chaotic nature of the Internet and the benign anarchy that prevails on it are wonderful. But it has become clear over the past decade that we need coordination, if not governance. The handling of domain names is the least of this. In September 1996, there was a conference at the JFK School of Government at Harvard. This volume represents rewriting the papers delivered there. If public policy and the economics of the Internet interest you, I think you will have to read this. The political and economic future of the Net will depend on these views.

The Net for Managers
Over the years, I’ve reviewed several books by Jeroen Vanheste, because I consider it important to take note of what’s published in languages other than English — or at least those I can read or puzzle my way through. This *Internet Handbook for Network Administrators* is a first-rate job. I hope that the publisher has it translated into English with alacrity. Every aspect of the Net that’s of interest to a manager is covered here: DNS, BIND, TCP/IP, Web servers, etc., etc. The exposition is lucid: I especially enjoyed the descriptions of SLIP and PPP. The final chapters (on firewalls and secure communication) are excellent. The bibliography is extremely brief and should be expanded in the translation or next edition.

Software
When I received WebSite Professional V2.0, I wasn’t quite sure what to do with it. I run UNIX and Linux and nothing from Microsoft. But I know webmasters who run NT. So I asked Steven Katz for his thoughts and sent him the CD-ROM and the docs. He wrote:

My general impression of WebSite is that it is for beginners or administrators of small to medium-sized sites. It has always been and is still easy to get started with. The documentation, support, and interface are very good. The default and unenhanced abilities and configuration of WebSite are probably more than adequate, possibly ideal for most.

I noticed that O’Reilly had dumped Cold Fusion and replaced it with iHTML. This is probably because O’Reilly Allaire isn’t distributing Cold Fusion 1.0 anymore and O’Reilly had a significant hand in developing iHTML. If you’ve looked at WebSite Pro, you may be surprised at the many other changes. The most visible is an increase in speed. The biggest drawback is that WebSite Pro has no remote administration capability. But it is vastly superior to Microsoft’s IIS if you’re involved with commercial public access. If you are creating Web sites that are meant for general public use and you aren’t looking to put massive amounts of security on the system, WebSite Pro is it. If you are looking for control and security and a limited number of people accessing the site, then IIS may be the one to use.

One problem is that IIS takes over completely separate server application permissions as well. When running O’Reilly’s WebBoard as a separate service, IIS decided that it’ll take over some of its permissions as well. It’s quite a nightmare.

*ZD Internet Magazine* had an article comparing Web servers, and remarked:

One thing to note when viewing our performance data: WebSite Pro’s performance suffers due to the fact that the product does not cache its pages. However, there’s a trade-off: WebSite Pro developers have more control over the site.

Katz also noted:

No one has really hit the nntp server for NT market very hard yet. As I have no experience with news servers, this is the sort of thing I’d want to buy from O’Reilly and hope that it had as nice an interface as WebSite and is as easy to get started with.

I hate the notion of an alien program grabbing permissions; I like the notion of having control; I like having robust software. It’s a clear win for WebSite Pro V2.0.
**book reviews**

Jerry Peek, Tim O'Reilly & Mike Loukides

*Unix Power Tools*, Second Edition

**Reviewed by Reginald Beardsley**
<rhb@acm.org>

The second edition of *Unix Power Tools* reminds me of a once popular witticism about ALGOL: the first version was a distinct improvement upon its successors. Some of the differences between the first and second editions:

- The section on password security has been deleted.
- The section on Awk has been substantially abbreviated and is now part of the section on batch editing. No mention is made of the availability of ‘the one true Awk’ from Brian Kernighan’s home page.
- Short scripts included in the text of the first edition must now be retrieved from the CD-ROM (e.g., logerres, at the end of the section on redirecting I/O).
- A page listing the significant changes to Perl w/release 5.0 has been added. (I mention this because the publisher’s blurb cited the importance of Perl as justification for abbreviating the treatment of Awk.)
- Highlighting of key words is now done by printing in medium gray rather than blue. Key words in sidebars are now printed in medium gray text on a light gray background!
- bash and csh have subsections of their own.

There are undoubtedly some other changes I didn’t notice. It is, after all, still more than 1,000 pages. However, several hours spent paging through both editions side by side revealed little change in the content. In some instances, it appeared that sections, such as the discussion of hard and soft links, had been substantially rewritten. Closer examination showed that the changes were really just improved paragraph headings.

The deletion of content and the change from two-color printing to one color suggest to me that reducing production costs was the real focus of the second edition. Otherwise, updating the CD-ROM would have been sufficient.

Normally, I give away my old copy when I get a new edition of a book; shelf space is just too dear for me to keep two copies. In this case, I gave away the new edition and kept the first.

Scott M. Ballew

*Managing IP Networks with Cisco Routers*

**Reviewed by Nick Christenson**
<npch@jgcafe.org>

Today one can go to any well-stocked technical bookstore and easily find dozens of books on just about every hot Internet topic, be it Java programming, WWW site construction, ATM, Linux, or what have you. However, practical information on Internet routing is a topic that has received little attention. This is especially strange because personnel with skills in deploying and maintaining routing systems are among the most sought after in today’s job market. Finally, a few books have begun to close this gap in the market. This book is one of these.

Unlike Chris Lewis’s *Cisco TCP/IP Routing Professional Reference*, Scott Ballew’s *Managing IP Networks with Cisco Routers* is not really a tutorial on configuring routers; this attempt to be more of a source of experience and wisdom in designing networks, selecting routing protocols, and then maintaining the whole system. It succeeds admirably in this regard.

The book starts with an introduction to the basics of IP networking, as one would expect. This is a good introduction and manages to cover nicely the debate over whether one should use IP addresses provided by an IP registry or ISP or whether one should use RFC 1918 reserved addresses. For IP veterans, there’s nothing new here, but that’s not who it’s for.

The next two chapters, covering network design, provide excellent information and advice for those who are looking to make their networks more maintainable, as well as for those who are designing a network from scratch. It’s apparent that the author has considerable experience in doing this, and one would be well advised to follow his advice.

After this, Ballew covers recommendations in selecting network equipment purchase, selecting appropriate routing protocols (focusing on interior routing protocols), and then configuring the router. The selection advice is sound, there is nothing really exciting here. Nonetheless, a lot of folks who have bough thrust into the role of vendor selection could benefit from this advice. There is also good advice on which routing protocol one ought to select under various circumstances, and we’re shown how to configure whichever routing protocol
one selects in a Cisco router. Be warned, though, that there's not enough information here to get complete novices and their routers out of the box and into service. For the complete novice, I would recommend *Cisco TCP/IP Routing Professional Reference* or, of course, the documentation that came with the router itself.

The next two chapters, covering the technical and nontechnical sides of network management, are my favorite in the book. Under nontechnical issues, we hear about the importance of defining the boundaries of one's network, developing staff skills, and establishing a help desk. Under technical issues, we cover network monitoring, troubleshooting, and change management. These last topics are especially well thought out. The issues Ballew raises on network monitoring are very well considered, and just about everyone would do well to read this before pursuing this topic too far. There's also especially good coverage of the use of a version control system for managing changes to router configuration files, a hot button of mine. If you're not doing this, you should be, and this book tells you how.

The final two chapters cover connecting to the outside world and network security. The chapter on exterior connections is decent, though not quite up to the quality of the rest of the book. Exterior routing as a whole is still an unexplored area in the literature. Ballew rightly points out that just the topic of BGP configuration issues could easily fill a book by itself. This is true, and it's a book that would be well received. The network security chapter does a good job defining the issues and a very good job of not trying to be the end-all authority on this. Instead, the basics are presented here, and the reader is then referred to other good sources for the details. Surprisingly few authors do this, and it's very refreshing.

There are four appendices covering configuring interfaces and obtaining RFCs and Internet drafts, and IP addresses. The last three, though certainly appropriate, contain pretty basic information for Internet veterans. The first is a pretty good guide to configuring network interfaces on Cisco routers, although no substitute for the documentation or a more thorough work.

Every goal the author had in writing this book seems to be well realized. It's obvious that Ballew has a lot of experience in designing and maintaining networks, and this experience shows through. A lot of wisdom about all aspects of networking and internetworking is present in this book, and every network professional would probably benefit from reading it carefully. This is probably the best single source of networking knowledge in print. *Managing IP Networks with Cisco Routers* isn't an introduction to Cisco's IOS, so beginners should supplement this work, but as it stands, I strongly recommend it. Also, Ballew's book complements Lewis's book nicely. Anyone thrust into the job of maintaining routers would do well to acquire both.

*Managing IP Networks with Cisco Routers* is a collection of excellent advice on configuring and managing IP networks. Although it is not an introduction to routers, it is a collection of exceptional advice on networking practices from someone who has obviously been there. I recommend it for every network engineer or anyone else interested in these issues.

**Chris Lewis**

*Cisco TCP/IP Routing Professional Reference*  

**Reviewed by Nick Christenson**  
<npc@jetcafe.org>

Despite the dominance of Cisco in the routing market, the incredible number of router devices deployed, and the scarcity of professionals truly skilled in configuring and maintaining these devices, there has been an scarcity of reference works on Cisco routers outside of Cisco's own documentation. In fact, this book is the first work published outside of Cisco's documentation that covers configuration of routers in any detail. Finally, this gap in the literature has been filled.

The book jumps right in to discuss the basics of routers, comparisons to bridges, and how to get the router out of the box and ready for configuration. In the second chapter, we take a step back and review TCP/IP. This introduction is necessary in a book like this, which aims to satisfy the needs of beginners. This delivery is pretty routine and unremarkable.

With Chapter 3, we get right into configuring the router, covering issues such as bootstrapping the config and loading the configuration from flash memory, the network, or typing it in by hand. Also discussed is the importance of setting up a lab or test network for learning and experimentation. This is an important issue, and its presence here is appreciated. Chapter 4 covers basic routing protocols, RIP, OSPF, IS-IS, IGRP, and EIGRP. Also covered is static routing, but I believe this technique is not pursued with the vigor it deserves. Static routing has some distinct advantages (some of which are explored in *Managing IP Networks With Cisco Routers* by Scott Ballew) that deserve to be championed.
Chapter 5 talks about supporting “legacy LANs,” which in this book means “anything that’s not IP.” This is a really good and important chapter, though, because there are still a lot of IPX, SNA, et al. networks out there, and it’s still somebody’s job to support them.

The next chapter discusses WAN technologies, focusing on slower speed networks like Frame Relay, SMDS, and X.25. It would seem that the logic is that folks with, for example, multiple T3s or ATM networks would already have someone on staff to maintain the router who doesn’t need to read this book. I believe this is an unfounded assumption, especially in today’s Internet. One of my biggest criticisms of this book is that it really shies away from high-end concerns. HSSI, ATM, BGP, and security issues are entirely welcome in a volume like this.

Chapter 7 is a big one that’s all over the map. Much of the meat of this book is contained here, and although it is very useful information, we would have been better served if there were more detail split into several chapters. Some of the issues covered are whether to use an RFC 1918 address space or not, configuring a Cisco router as a firewall, data compression, and SNMP. Clearly, each of these is a very deep topic. At the very least, suggestions on where the reader might turn for more detail would be appreciated.

The final chapter is on troubleshooting. Again, this is a very big topic. It is impossible to cover everything, but Lewis does a good job of providing a framework for a general analysis of networking problems.

Overall, *Cisco TCP/IP Routing Professional Reference* is an excellent introduction to the topic of Cisco routing. The topics covered are clear, and the basics are well covered. For someone with little experience who needs to maintain a Cisco router, this book would be extremely valuable, and I’m happy to recommend it as such. Nonetheless, I was disturbed by some of the omissions.

In general, the book seems to cover routing state of the art as of five years ago, but there have been some very significant changes that, in my opinion, are absolutely necessary. The first is the issue of classless routing (CIDR). I couldn’t find this critical topic mentioned once in the entire book.

Another shortcoming is on the topic of firewalls. Even though one is instructed on how to set up access lists, the discussions on methodology are neither sufficient nor state of the art. There is neither mention of IP spoofing filters nor distinctions between input and output filters. The readers would be better served by being given a more thorough description of how to set up various kinds of filters and then strongly referred to a different source for information on general firewall implementation philosophies, such as Chapman and Zwichky’s excellent *Building Internet Firewalls*.

A curious shortcoming, given that the author hails from the United Kingdom, is the US-centric view of networking. Most of the concepts can be applied to E1 circuits in the same way that one would apply them to T1 lines. Nonetheless, I would not have minded a more international viewpoint.

There are some fairly minor editorial problems with the book, nothing extreme, but more than one would like. Most of the problems occur early in the book and have to do with missing spaces in examples, which can be misleading. The reader would be well advised to look out for them.

Despite these criticisms, this is still a very good and useful book. Beginners needing a reference on Cisco routers would be well advised to pick it up and read it carefully, although they should keep in mind that it does not include everything they probably want to know. Matched with Ballew’s *Managing IP Networks with Cisco Routers*, which is long on good design principles but shy on the configuration details, it becomes even stronger.

A very good basic book on configuring Cisco routers, this book fills an important gap in the literature. Despite some significant shortcomings, it is still a worthwhile acquisition, especially for novice router administrators.
The Committee nominates the following individuals:

**President:**
Andrew Hume, AT&T Research

**Vice President:**
Greg Rose, Qualcomm

**Treasurer:**
Dan Geer, CertCo

**Secretary:**
Peter Honeyman, University of Michigan

**At-Large:**
Jon “maddog” Hall, Digital Equipment Corp.
Jordan Hubbard, FreeBSD
Darrell Long, University of California, Santa Cruz
Pat Parseghian, Transmeta Corporation
Hal Pomeranz, Deer Run Associates
Mark Teicher, WITSEC, Inc.
Elizabeth Zywicky, Silicon Graphics
K. Bostic, Chair, Nominating Committee

**Board Meeting Summary**

by Ellie Young
Executive Director
<ellie@usenix.org>

The biennial election for officers and directors of the Association will be held this Spring.

Ballots will be sent to all paid-up members on or about February 18. Members will have until March 27 to return their ballots, in the envelopes provided, to the Association office. The results of the election will be announced on comp.org.usenix, the USENIX Web site, and in the June issue of :login:.

The Board is made up of eight directors, four of whom are “at large.” The others are the President, Vice President, Secretary, and Treasurer. The voting is preferential; those candidates with the largest number of votes are elected. Ties in elections for Directors shall result in run-off elections, the results of which shall be determined by a majority of the votes cast.

Newly elected directors will take office at the conclusion of the first regularly scheduled meeting following the election, or on July 1st, whichever comes earlier.

**Report of the Nominating Committee**
The USENIX Nominating Committee, under the By-Laws of the Association, is charged with ensuring that there is at least one candidate for each of the four officer posts on the Board of Directors and at least four candidates for the four At-Large seats. We are very pleased that such a large number of nominees stepped forward. Each of the following nominees has indicated her/his willingness to serve. The By-Laws further permit nominations by petition. Such nominations must be of the form described by the By-Laws.

Here is a summary of the actions taken at the regular meetings of the USENIX Board of Directors held on October 25 and 26, 1997, in San Diego, CA.

Budget

The assumptions behind the first draft budget for 1998 were discussed, and the budget was later approved as amended.

Conference Fees. It was decided to raise conference registration fees for events with 3-day technical sessions by $15.00 and full day tutorial fees for all conferences by $10.00.

Proposals For Funding

Standards. Stoughton's proposal to wrap up our efforts in POSIX in 1998 and to continue sending a representative to the Open Group's System Management Group was accepted. Stoughton was asked to write a statement regarding USENIX's strategy shift from POSIX to the Open Group.

Student Network Administration Project. The proposal submitted by the Maryland Virtual High School to have USENIX co-fund at a 50% level ($167,000) with the NSF a three-year student network administration project was accepted. This project is designed to develop curriculum and teacher training in order to support the technological needs of schools and vocational skills of students.

USACO. The proposal to once again fund the USA Computing Olympiad, and also travel for the team to participate in the International Olympiad in Informatics was approved. The Board also made a motion to express their gratitude to Rob Kolstad for his dedication to this project over the years.

John Lions Student Prize. The proposal for USENIX to contribute AU $10,000 to create an endowed fund for this prize with the AUUG was accepted.

Affiliations. It was agreed to renew membership in the Computing Research Association with a $10,000 contribution.

Conferences

NUUUG. It was agreed that Young should proceed with plans for USENIX to co-sponsor a conference with the Netherlands UNIX Users Group in 1998.

USENIX Annual Conference '98. Greg Rose was appointed the liaison to the organizing committee that is putting together a third "freenix" track.

Electronic Commerce Workshop. Bennet Yee will serve as program chair for this workshop, and Dan Geer was charged with coordinating a track/program on public key infrastructure.

Tek/Tk Conference. Don Libes and Mike McLennan will serve as program co-chairs for the '98 event.

Windows NT Workshop. Seltzer reported that Thorsten von Eicken and Susan Owicky had agreed to serve as co-chairs or a 1998 workshop and that it would also include tutorials.

System Administration of NT. It was decided that a conference on large installation system administration of NT will be held. Remy Evard and Ian Reddy will co-chair, tutorials will be included, and it will once again be co-located with the Windows NT Workshop.

Domain Specific Languages Conference. Hume reported that the program committee had recommended repeating this in 24 months, and that Tom Ball will serve as program chair.

Academic Acceptance

There was a lot of discussion concerning how USENIX might gain a wider audience and more acceptance in academia. Honeyman was asked to make a proposal to develop and possibly publish a compendium of USENIX papers that have had a major impact on the systems community.

Revisions to the STG and Reserve Fund sections of the Policies Document were made. A record of the STGs committee's deliberations, past action, and reviews was to be kept.

Next Meeting

The next meeting of the Board will be held on March 21, 1998, in Boston, MA.
Letter from the President: Looking Forward

By Andrew Hume
President, USENIX Board of Directors

<andrew@usenix.org>

As I write this in January, it is commonplace to self-assess, to evaluate the year past, and to make some plans for the year to come.

What were some of my highlights of 1997? Professionally, taking an internal project from a blank slate and a several million dollar budget to a system in production in just nine months. (Designing and building such a large system (2.5TB of disk, 150TB of tape, processing 200+GB per day) is a blast, btw.) Musically, this year was good. Concerts were fewer but better than the previous year: Bruce Springsteen at a small theater in Asbury Park, Morrissey, and Philip Glass's 60th birthday concert at Lincoln Center. My current CD rotation is Secret Samadhi (Live), Akhnaten (Glass), Razorblade Suitcase (Bush), Songs from the Capeman (Simon), and Little Jagged Pill (Morrisette). And finally, my wife and I took up sailing.

The year's lowlights? Working too many 18-20 hour days, which caused substantial neglect of home life. Working with too many folks for whom process is much more important than results. There are surely more lowlights, but they're obscured by my fatigue headache.

What about USENIX? We put on some thought-provoking and practical workshops (I quite liked Domain-Specific Languages in Santa Barbara, and had quite an interesting time with the NT crowd in Seattle - which was not so much a clash of cultures as a cognitive dissonance). We felt the eighteen month gap between our annual technical conferences more keenly than I had expected - my thanks to Carl Staelin for filling that gap with the Symposium on Internet Technologies & Systems. We spent $800K on "Good Works" (detailed on page 72), which are mostly aimed at student-related activities. Our staff continues to be outstanding and enthusiastic. They produce a large amount of work with a small, competent group - fortunately they are lean and keen, rather than lean and mean. And last but not least, the USENIX Board of Directors works well together, with effective and productive meetings. I especially commend the Scholastic Committee, chaired by Margo Seltzer, for its work in outlining a charter and overseeing all aspects of our student programs.

What about the year to come? The biggest event on the near horizon is the upcoming election for the USENIX Board of Directors. Three members of the current board are not seeking re-election (Eric Allman, Lori Grob, and Margo Seltzer). We will miss them dearly. The nominating committee, chaired by Keith Bostic, has assembled a good slate of candidates for the election (see p. 69), and they may be joined by others who are nominated by petition. Although it is neither my place nor my intent to tell you how to vote, I would like to share with you some of the issues that I consider when I vote. Hopefully, they will be useful to you as well.

The basic job of the Board is to lead and guide the organization. In order to do this, the Board must be responsible and accountable to the membership, understand the organization's operations, and perform some tasks. For me, the latter is a combination of having the time and energy to take on extra tasks and also being able to work effectively within a group. Besides possessing these qualities, it is also important for potential board members that their experience and background represent the major constituents...
cies of our membership. Let me illustrate this with examples from the previous election on why I voted for Margo Seltzer and Eric Allman. To me, Margo represents our academic and research constituencies, provides good insight and understanding on the academic/student portion of our Good Works program, has a track record with USENIX, and works very well with others. Eric Allman has long term experience with the academic and research community as well as working with a start-up company, is willing and able to perform the significant duties of treasurer, and has participated effectively at meetings as a program chair and board member for many years.

When considering this year’s candidates and how they might best represent our constituencies and goals as an organization, I would want at least two active academics to serve on the Board, since much of USENIX’s focus and money is devoted to student programs and academic research plays a large role in our conferences. Roughly half of our membership are system administrators, and we have SAGE as a special technical group with its own committee to address their specific needs. Yet the USENIX board needs to have its own direct representation of that group as well (such as Zwick on the current board). The free (well, almost free, or at least not terribly expensive) UNIX community, which includes *BSD*, GNU and Linux (in alphabetic order), is a growing and vibrant subgroup; I would want board members familiar with that area (especially its goals and politics). And finally, but certainly not least, we need to represent and understand the commercial side of UNIX; the companies who build and sell the UNIX we use, or sell the products that help us use those UNIX systems effectively, or provide the human resources to help build or run our systems and applications.

This is not the only, nor even the dominant, metric for evaluating candidates. But it is an important one to me, and I hope you’ll consider it as well.

USA Team Scores Gold

by Rob Kolstad

Rob Kolstad, editor of *Login* and president of BSDI, is head coach of the USA Computing Olympiad Team.

<kolstad@usenix.org>

The USA Computing Olympiad team earned one gold, one silver, and one bronze medal at the recent International Olympiad on Informatics (IOI) held in Cape Town, South Africa, November 30–December 9, 1997.

15-year old whiz kid Matthew Craighead, high-school senior from Mendota Heights, Minnesota, scored a gold medal in his second trip to the international championships. Dan Adkins, MIT freshman and three-trip veteran, earned a silver. Russell Cox, Harvard freshman and two-trip veteran, won a bronze medal. Barely missing a bronze (nine points out of 600) was Benjamin Matthews, Dallas, Texas, a sophomore in his first international competition.

Coaches Don Piele (University of Wisconsin/Parkside professor), Rob Kolstad (BSDI president), and Hal Burch (CMU graduate student) accompanied the team to South Africa. Noncompetition events included a visit to an ostrich ranch, a trip to the top of beautiful Table Mountain, an excursion to World of Birds, and a drive to the stormy Cape of Good Hope, where two oceans meet.

This year’s competition sported problems different from most. Instead of problems begging for clever, well-programmed searching solutions, some of this year’s had more of an artificial-intelligence flavor. Scoring was based on a rating of the solution based on a “very good” solution rather than a perfect solution. This did cause a bit of trouble for some of the contestants!

The 1997-1998 USA Computing Olympiad is well underway with three more contests scheduled over the next five months.

Join the <hs-computing@delos.com> mailing list for complete info or see <www.usaco.org>.

There’s Gold in Good Works: A Report on USENIX Support of Worthwhile Projects

by Cynthia Deno

Cynthia Deno directs the USENIX Association’s marketing effort. She is always eager to hear from members with suggestions for outreach or new services for members.

<cynthia@usenix.org>

The USA Computing Olympiad (see previous story) is just one of many “good works” programs USENIX supports, and this is not the first time USENIX has enjoyed the pride of a job well done. Every year USENIX member dues, conference revenue, and other funds are used to give back to and help nurture the development of the advanced computing systems community interpreted in the largest sense. In 1997 alone USENIX spent just under a million dollars on such good works. You, as a USENIX member, can be proud to be associated with such fine projects and pleased that your Association is in the thriving financial position which allows this generous level of support. Here are some details.

Graduate and undergraduate college education is always of the highest priority to the Association. USENIX and its members value students and the research in the computing systems arena that is generated in colleges and universities. Recognizing the importance of this work, USENIX generosity funds a num-
ber of programs for college students. As Margo Seltzer, USENIX scholastic committee chair and professor of computer science at Harvard, says “We are enthusiastically looking forward to providing opportunities to an ever-increasing group of students.”

**Student Scholarships**

First among these programs are the USENIX student scholarships, which typically cover some or all of a student’s expenses including tuition, supplies, and stipend. The proposal by a faculty member is easy, according to Mary Baker, professor at Stanford University, whose student was a recent recipient. Here’s what she had to say when notified of the award: “One of the things I love about USENIX is that there’s so little paperwork involved. A scholarship where the student and advisor don’t have to dig up transcripts back to kindergarten and such is a special thing in this world.” See: [http://www.usenix.org/students/scholar.html](http://www.usenix.org/students/scholar.html)

**Conference Participation for Students**

USENIX strongly supports graduate and undergraduate student participation in our conferences. We offer students very low registration fees for USENIX technical sessions and tutorials. The student stipend program provides grants for travel to our conferences. Student contributions to conference programs are encouraged with best student paper awards; $1,000 cash prizes at the annual Technical and LISA conferences, while $500 prizes are awarded at the other conferences and symposia. See: [http://www.usenix.org/students/stipend.ann.html](http://www.usenix.org/students/stipend.ann.html) [http://www.usenix.org/students/best_paper.html](http://www.usenix.org/students/best_paper.html)

**Student Research and Software Projects**

A generous annual budget provides for funding of student research projects and student software projects (i.e., projects that allow students to perform the software engineering necessary to take an undergraduate course project software package to an actual, robust, and portable software package useful to the greater computing community). With funding, we also pre-approve student travel stipends so the students can attend a USENIX conference and present the results of their work. See: [http://www.usenix.org/students/research.html](http://www.usenix.org/students/research.html) [http://www.usenix.org/students/software.htm](http://www.usenix.org/students/software.htm)

**Reps on Campus**

A more innovative program is the USENIX Reps on Campus. In exchange for an annual free conference registration and a complimentary educational membership, computer science department faculty and staff on various campuses distribute Association materials to students, maintain a library of USENIX conference proceedings, answer questions, and spread the word about USENIX’s activities. See: [http://www.usenix.org/students/outreach.html](http://www.usenix.org/students/outreach.html)

**The College Fund Endowment**

We are particularly proud of the USENIX Association’s endowment in 1997 of a scholarship for the College Fund; the endowment will provide a $10,000 annual scholarship to encourage minority students to study computer science. The College Fund (formerly United Negro College Fund) is one of the nation’s most successful higher education assistance organizations. On the day the scholarship was announced USENIX president Andrew Hume said, “Historically and currently, minorities are underrepresented in the technical community that is the core of USENIX’s membership. USENIX is delighted to make a substantial contribution towards increasing minority participation in the field of computer science.”

**Training for Settlement House Youth**

USENIX’s commitment to providing opportunities in computing to disadvantaged youth was demonstrated again in April 1997 when we granted $65,000 to the Polytechnic University. The grant is to train youth in five United Neighborhood settlement houses in New York City in Internet applications and help them develop skills they will need in the future. It supports a mentor program of Polytechnic students who use their technology skills to provide valuable community service and support access to computers and technology for all residents, as they help disadvantaged younger students. Announcing the program, Dr. Noel Kriftcher, head of Polytechnic University’s David Packard Center for Technology and Educational Alliances, said: “Beyond the computer skills that will be developed in minority and female youths, these young people will expand their employment opportunities, be encouraged to continue their studies, be exposed to Polytechnic mentors as role models and learn about technology related careers.” See: [http://www.poly.edu/pr/usenix.asp](http://www.poly.edu/pr/usenix.asp)

**Women in Computing**

Another group which has been under-represented in the computing professions is women. In efforts to support women’s fuller participation, USENIX has contributed to funding the production of a video targeted at high school and college students. The video “Career Encounters: Women in Computing” will be broadcast nationally on cable and satellite public television networks. USENIX also provided travel grants to enable 32 women students to attend the Grace Hopper Women in Computing Conference held September 1997 in San Jose, California. See: [http://www.sdsc.edu/Hopper](http://www.sdsc.edu/Hopper)

**Pre-college Programs**

As illustrated by our support of the USA Computing Olympiad, pre-college computer education is another area of natural interest with USENIX. We provide funds to support many worthwhile projects designed to further the national goal of getting meaningful access to computers into schools, encourage students in tool-based technology and skills, and enhance the quality of early education in computing.
SAGE and USENIX last year funded Evi Nemeth and Adam Boggs of the University of Colorado to present a twoday seminar to student sysops who are part of the Maryland Virtual High School (MVHS) program. MVHS links high schools via the Internet to share information and computer resources. The program is funded by a $1.5 million National Science Foundation grant and brings to the classroom a team approach to problem solving amid a technology-rich environment. Student sysops independently study advanced computer science topics while keeping school computers and networks running and helping other students, staff and faculty learn computer tools. See: <http://mvhs1.mhs.edu/mvhsprog/sage.html>. We will co-fund the Student Network Administration Project with the NSF in 1998. An outgrowth of MVHS, the project aims to develop a formal curriculum and teacher training to support the technological needs of schools and vocational skills of students. Making the curriculum available over the Web is one of the project goals and we will let you know when it is available.

The CitySpace Project is an ongoing series of award-winning, focused preproduction workshops exploring Internet communications, three-dimensional modeling, as well as fundamentals of system administration and maintenance for students between the ages of 10-16. USENIX provided seed money for this fun, highly interactive way to foster sophisticated software skills among young people. See: <http://cityspace.org/>.

The WebStar Award recognizes public-service Web sites developed by an individual or group in the K-12 age group. Awards are based on a combination of interface design, friendliness, accuracy, and the value of the site to the online community. Dave Taylor, president of Intuitive Systems and columnist for login:, generously donates the annual WebStar Award prize money. Association staff provide the logistics in support of the award and USENIX funds travel for the lead Webmaster and a parent to attend the USENIX annual technical conference. See: <http://www.intuitive.com/webstar/>.

The John Lions Student Prize is an example of both USENIX’s commitment to supporting college students and our ongoing cooperation role with other computing organizations. USENIX provided 50% of the endowment to fund this annual $1,000 student prize, which is administered by the Australian UNIX Users Group (AUUG). See: <http://www.auug.org.au/>.

Support for Other Organizations
Among other organizations which have recently received USENIX support are the Internet Software Consortium (ISC) and the Netherlands UNIX Users Group. USENIX will co-sponsor the System & Network Administration Conference with NLUUG in Fall 1998. See: <http://www.nluug.nl/>.

In 1997, USENIX provided bridge funding for the ISC, which is maintaining and developing publicly available code for key portions of the Internet infrastructure, including widely used implementations of the Domain Name System (BIND), Netnews (INN), the Dynamic Host Configuration Protocol (DHCP), and Kerberos Version 5.0. See: <http://www.isc.org/>.

Lastly, USENIX has recently joined Computing Research Association. The CRA mission is to represent and inform the computing research community and to support and promote its interests. CRA seeks to strengthen research and education in the computing fields, and improve public policy-makers’ understanding of the importance of computing and computing research in our society. See: <http://www.cra.org/>.
Twenty Years Ago in login:

by Peter H. Salus

Peter H. Salus is the author of A Quarter Century of UNIX (1994) and Casting the Net (1995). He has known Lou Katz for over 40 years.

The beginning of 1978 poses several chronological problems for the historian. First, the February login: was mailed before the January issue; second, when it appeared, the January issue was the December/January issue. Should I present these in the order labelled or in the order sent? I pick the latter; it reflects reality. (Anyway, it makes March and April clearer; they were mailed in reverse order.)

The February login: concentrated on the program for the forthcoming meeting (May 24-27) at Columbia University’s College of Physicians and Surgeons; Lou Katz was the program chair. The program outline was printed with the note: “The program is very much subject to change depending upon the schedules of speakers.” These were still the days when, if you had something to talk about, you told the chair and were put on the program.

P&S was where the first UNIX Users’ Meeting had been held in May 1974, with about two dozen attendees. The meeting in Urbana in 1977 brought just over 250 devotees together. Lou and his cohorts had no idea as to how many might register for the 1978 meeting. And this was a real meeting. Sessions ran from 1 p.m. to 11 p.m. on Wednesday, 9 a.m. to 11 p.m. on Thursday and Friday, and 10 a.m. to 2 p.m. on Saturday. After 2 p.m., there were “Visits to Laboratories.” As the meeting was in May, I’ll leave the rest of this to the next article.

The December-January 1978 issue of login: followed. It began with a note from Mel:

Here at last, with many apologies, is the “first” issue of login: for 1978. The “March” issue will go to the printer within a week or so and then “May” will complete the catching-up on old correspondence.

Tom Ferrin’s famed “fix” of the PDP 11’s memory management unit (letter of December 9, 1977) deserves mention here. It was an elegant hardware solution to a software problem. Here’s a part of Tom’s letter:

The memory management unit in the PDP-11/45 and 11/70 computers offers several advantages over those found in the other PDP-11 family computers. Among the more powerful features is the ability to separate programs into instruction segments and data segments . . .

Four PDP-11 instructions facilitate program communication between different addressing modes and instruction/data areas in memory. These are “move from previous instruction/data memory space” (mpfi, mfpi, mtpd, mfpd) . . .

Because of DEC’s desire to “preserve the integrity of proprietary programs,” the “mfpi” instruction does not work correctly when executed with a process status word equal to 17xxxx (i.e., both current and previous modes are USER). This fact prevents the C sub-routine “nargs.s” from operating as intended . . .

There are several solutions to this deficiency . . .

4. Modify the hardware to work more “correctly”. . . After several telephone calls to DEC representatives and a few hours of looking at microcode . . . we have arrived at a simple modification to the PDP-11/70 cpu to allow the “mfpi” instruction to function properly. The modification takes about 15 minutes for an experienced person to implement and involves cutting one foil etch and adding one jumper wire to the M8138-YA memory management board.

The issue also contained a duces notice of great complexity. I’ll let Mel Ferenz’s words speak for themselves:

Enclosed is an invoice for eighteen months for January 1978-June 1979. As was explained in the recently mailed report of the “USENIX committee,” the annual dues starting July 1978 is $50.00. We feel honor bound to charge $10.00 per year for the period before July 1, hence the invoice is for $5.00 for the first six months and for $25.00 for each of the next six month periods. By striking out the appropriate lines you may pay for 6, 12, or 18-months. If you can avoid processing purchase orders we will be most grateful.

There were over 250 members.
Thanks to our Volunteers

by Ellie Young
Executive Director
<ellie@usenix.org>

USENIX continues to be successful and this would not be possible without the volunteers who lend their expertise and support for our conferences, publications, members services, and philanthropic activities. While there are many who serve on program committees, coordinate the various activities at the conferences, work on committees, and contribute to this magazine, I would like to make special mention of the following individuals who made significant contributions in 1997:

The program chairs for our 1997 conferences:

John Kohl, 1997 USENIX Technical Conference
Steve Vinoski, 3rd Conference on Object-Oriented Technologies & Systems
Brent Welch & Joe Konstan, 5th Tcl/Tk Workshop
Mike Jones & Ed Lazowska, USENIX Windows NT Workshop
Phil Scarr & Kev Gittler, Large-Scale System Administration of Windows NT Workshop
Chris Ranming, Conference on Domain-Specific Languages
Hal Pomeranz & Celeste Stokely, LISA XI
Carl Staelin, USENIX Symposium on Internet Technologies & Systems

The conferences’ invited talk/special track coordinators:

Mary Baker & Barry Kercheval for the invited talks at the 1997 Technical Conference

Jon “maddog” Hall & Michael Johnson for organizing the Uselinux track at the USENIX Technical Conference
Doug Schmidt for serving as tutorial program chair for COOTS ’97
Rik Farrow & Pat Wilson for the invited talks at LISA XI
Adam Moskowitz for organizing the Advanced Topics workshop at LISA XI
Rajendra K. Raj for organizing the Advanced Topics Workshop at COOTS.
Evi Nemeth for organizing student volunteers and MBone services

On the SAGE executive committee, I would like to thank the following members for their extra efforts in 1997:

Paul Evans, who helped in the development of the Sys Admin of NT workshop
Hal Miller & Barb Dijker for their efforts in pulling together By-laws and Procedures documents
Helen Harrison & Kim Trudel for their work on the SANS ’97 conference program
Pat Wilson for her efforts in helping our Webmaster to expand the site’s services.

And also:

Keith Bostic for serving as chair of the USENIX Nominating Committee for the upcoming Board of Directors Election.
Steve Johnson for serving this past year as an ex-officio member of the USENIX Board of Directors, as well as being the liaison to the Computing Research Association.
Eric Allman for serving as the USENIX liaison to the SAGE Executive Committee
Margo Seltzer, Brian Bershad, David Kotz, Lori Grob, & Peter Honeyman for serving on the USENIX Scholastic Committee, which initiated the many programs that endow and support student projects.
Greg Rose for his efforts in providing and maintaining the PGP Key signing service to our members.
The authors and reviewers, too numerous to list here, who review most of the articles that appear in this magazine.
And last but not least, the members of the USENIX Board of Directors who spend many of their “free” hours providing leadership and governance. USENIX is grateful.
Keynote: The Shape of Things to Come

Rick Rashid, Microsoft Research
The relentless pace of progress in hardware and software technology will dramatically change computing over the next ten years. Software technologies once considered esoteric such as natural language processing, Bayesian reasoning, computer vision, and speech will dramatically affect not only the way humans and computers interact but also the way humans interact with each other. Moreover, the fundamental relationships between software and hardware will significantly change as software objects become more dynamic and operating systems increase the level of abstraction provided to developers. This talk addresses these coming changes and discusses how they will effect the uses of computing and the kinds of software our industry will be developing in the future.

Richard Rashid heads the Microsoft Research Division where he has focused on operating systems, networking, and multiprocessors, and is responsible for the creation of key technologies leading to the development of Microsoft's interactive TV system, now in test deployment.

Before joining Microsoft, Dr. Rashid was a professor of computer science at Carnegie Mellon University where he directed the design and implementation of several influential network operating systems, including the Mach operating system, and published dozens of papers in the areas of computer vision, operating systems, programming languages for distributed processing, network protocols, and communications security. He is credited with co-development of one of the earliest networked computer games, Alto Trek.

Tutorial Program

Monday, April 27
Morning Session: 9:00 am - 12:30 pm
M1am Designing Concurrent Object-Oriented Programs in Java Part 1
  David Holmes, Microsoft Research Institute, Macquarie University;
  Doug Lea, SUNY Oswego
M2am Understanding COM and MTS
  David Chappell, Chappell & Associates
M3am Building Distributed CORBA Applications In C++
  Steve Vinoski, IONA Technologies, Inc.
Afternoon Session: 1:30pm - 5:00 pm
M4pm Concurrent Java Programming (pt 2)
  David Holmes, Microsoft Research Institute, Macquarie University;
  Doug Lea, SUNY Oswego
M5pm Designing with Patterns
  John Vlissides, IBM Research
M6pm Distributed COM and MTS: The Programming Model, the Protocol and the Runtime Architecture
  Don Box, DevelopMentor

Tuesday, April 28
Morning Session: 9:00 am - 12:30 pm
T1am Framework and Component Modeling for Java with UML
  Desmond D’Souza, Icon Computing, Inc.
T2am Distributed Computing with Java Remote Method Invocation
  Jim Waldo and Ann Wollrath, Sun Microsystems, Inc.
T3am High-Performance C++ Programming
  Scott Meyers, Software Development Consultant
Afternoon Session: 1:30pm - 5:00 pm
T4pm Java/RMI, DCOM, and CORBA Interworking
  Keith Moore, Hewlett-Packard Laboratories
T5pm Three Cool Things in C++
  Scott Meyers, Software Development Consultant
T6pm Java Beans
  Prithvi Rao, KiwiLabs

The technical program was not available at press time.
For more information and updates, see
http://www.usenix.org/events/coots98/
Announcement and Call for Participation

3rd USENIX Workshop on Electronic Commerce
Including Invited Presentations on Public Key Infrastructure

August 31–September 3, 1998
Tremont Hotel
Boston, Massachusetts

Sponsored by USENIX, the Advanced Computing Systems Association

Important Dates
Extended abstracts due: March 6, 1998
Notification to authors: April 17, 1998
Camera-ready final papers due: July 21, 1998

Program Committee
Chair: Bennet S. Yee, UC San Diego
Public Key Infrastructure Coordinator: Daniel Geer, CertCo, LLC
Ross Anderson, Cambridge University
Nathaniel Borenstein, First Virtual
Marc Donner, Morgan Stanley
Niels Ferguson, DigitalCash
Mark Manasse, Digital Equipment Corp.
Cliff Neuman, University of Southern California
Avi Rubin, AT&T Labs
Wintreese, OpenMarket
Doug Tygar, Carnegie Mellon University
Hal Varian, UC Berkeley

Overview
The Third Workshop on Electronic Commerce will provide a major opportunity for researchers, experimenters, and practitioners in this rapidly self-defining field to exchange ideas and present the results of their work. It will set the technical agenda for work in electronic commerce by enabling workers to examine urgent questions, share their insights and discover connections with other work that might otherwise go unnoticed. To facilitate this, the conference will not be limited to technical problems and solutions, but will also consider their context: the economic and regulatory forces that influence the engineering choices we make, and the social and economic impact of network-based trading systems.

Each of the Workshop's three days will have two sessions focused on Public Key Infrastructures (PKI). This series of invited speakers and debates will examine the role and possible mechanisms of PKI in the future of electronic commerce. Emphasis will be on the practical side--actual field cases--and learning from experience. We seek, as engineers, to determine which of the various competing PKI claims are sustainable and practical, and, as business people, to learn what of the available PKI technology is actually correlated with our needs as they truly are.

The Workshop on Electronic Commerce will begin with tutorials which offer in-depth instruction in essential technologies. The one day of tutorials will be followed by three days of refereed papers and panel presentations examining topics in electronic commerce as well as the invited sessions exploring Public Key Infrastructures. A hosted reception on Wednesday evening and evening Birds-of-a-Feather sessions will provide opportunities for attendees to meet together informally.

Tutorials Proposals Welcome
One day of tutorials, on August 31, will start off the Workshop...USENIX's well-respected tutorials are intensive and provide immediately-useful information delivered by skilled instructors who are hands-on experts in their topic areas. Topics for the Electronic Commerce Workshop will include, but are not limited to, security and encryption.

If you are interested in presenting a tutorial, please contact:
Dan Klein, Coordinator
Email: dok@usenix.org
Phone: 412.421.2332

Public Key Infrastructures Sessions
We welcome your suggestions of participants, topics and format. All speakers will be invited. Please contact the PKI Sessions Coordinator, Dan Geer, at geer@world.std.com.

Workshop Topics
Two and one-half days of technical sessions will follow the tutorials. We welcome submissions for technical and position paper presentations, reports of work-in-progress, technology debates, and identification of new open problems. Birds-of-a-Feather sessions in the evenings and a keynote speaker will round out the program.

We seek papers that address a wide range of issues and ongoing developments, including, but not limited to:

- Advertising
- Anonymous transactions
- Auditability
- Business issues
- Copy protection
- Credit/Debit/Cash models
- Cryptographic security
- Customer service
- Digital money
- EDI
- Electronic libraries
- Electronic wallets
- Email-enabled business
- Exception handling
- Identity verification
- Internet direct marketing
- Internet/WWW integration
- Key management
- Legal and policy issues
- Micro-transactions
- Negotiations
- Privacy
- Proposed systems
- Protocols
- Reliability
- Reports on existing systems
- Rights management
- Service guarantees
- Services vs. digital goods
- Settlement
- Smart-cards
Questions regarding a topic's relevance to the workshop may be addressed to the program chair via electronic mail to ec98chair@useunix.org. USENIX will publish Conference Proceedings which are provided free to technical session attendees; additional copies will be available for purchase from USENIX.

What to Submit
Technical paper submissions and proposals for panels must be received by March 6, 1998. We welcome submissions of the following type:

1. Refereed Papers—Full papers or extended abstracts should be five to 20 pages, not counting references and figures.

2. Panel proposals—Proposals should be three to seven pages, together with a list of names of potential panelists. If accepted, the proposer must secure the participation of panelists, and prepare a three to seven page summary of panel issues for inclusion in the Proceedings. This summary can include position statements by panel participants.

3. Work-In-Progress Reports—Short, pithy, and fun, WIP reports introduce interesting new or ongoing work and should be 1 to 3 pages in length. If you have work you would like to share or a cool idea that is not quite ready to publish, a WIP is for you! We are particularly interested in presenting student work.

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, email and fax numbers and, if available, a URL for each author. If all of the authors are students, please indicate that in the cover letter for award consideration (see "Awards" below).

USENIX workshops, like most conferences and journals, require that papers not be submitted simultaneously to more than one conference or publication and that submitted papers not be previously or subsequently published elsewhere. Submissions 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.

Where to Submit Proposals
Please send submissions to the program committee via one of the following methods. All submissions will be acknowledged.

- **Preferred Method:** email (Postscript or PDF formats only) to: ec98papers@useunix.org.

- Files should be encoded for transport with uuencode or MIME base64 encoding.

Authors should ensure that the PostScript is generic and portable so that their papers will print on a broad range of postscript printers, and should submit in sufficient time to allow us to contact the author about alternative delivery mechanisms in the event of network or other failure. If you send PostScript, remember the following:

- Use only the most basic fonts (TimesRoman, Helvetica, Courier). Other fonts are not available with every printer or previewer.

- PostScript that requires some special prolog to be loaded into the printer won’t work for us. Please don’t send it.

- If you use a PC- or Macintosh-based word processor to generate your PostScript, print it on a generic PostScript printer before sending it, to make absolutely sure that the PostScript is portable.

- If you are generating the PostScript from a program running under Windows, make sure that you establish the "portable" setting, not the "speed" setting for PostScript generation.

A good heuristic is to make sure that recent versions of Ghostview (e.g. Ghostview 1.5 using Ghostscript 3.33) can display your paper.

- **Alternate Method:** 10 copies, via postal delivery to:

  EC’98 Submissions
  USENIX Association
  2560 Ninth Street, Suite 215
  Berkeley, CA 94710

  For detailed submission guidelines, send email to ec98authors@useunix.org, refer to the conference Web page at www.useunix.org/events/le98/guidelines.html, or send email to the program chair at ec98chair@useunix.org.

  An electronic version of this Call for Papers is available at:
  www.useunix.org/events/le98/.

Birds-Of-A-Feather Sessions (BoFs)
Do you have a topic that you’d like to discuss with others? Our Birds-of-a-Feather Sessions may be perfect for you. BoFs are very interactive and informal gatherings for attendees interested in a particular topic. Schedule your BoF in advance by telephoning the USENIX Conference Office at 714.588.8649 or sending mail to: conference@useunix.org.

Awards
The program committee will offer awards of $500 for the best paper and the best student paper.

Registration Information
Complete technical and tutorial programs, registration fees and forms and hotel information will be available on the USENIX Web site in June, 1998. The information will also be printable from a PDF file located on the Web site. However, if you would like to receive the printed program booklet, please request it at any time by email to conference@useunix.org.

About USENIX
USENIX is the Advanced Computing Systems Association. Since 1975 USENIX has brought together the community of engineers, system administrators, and technicians working on the cutting edge of the computing world. For more information about USENIX:

- **URL:** www.useunix.org
- Email: office@useunix.org
- Fax: 510.548.5738
- Phone: 510.528.8649
Announcement and Call for Participation

12th Systems Administration Conference (LISA '98)
Sponsored by USENIX, The Advanced Computing Systems Association and SAGE, the System Administrators Guild

December 6-11, 1998
Marriott Copley Place Hotel
Boston, Massachusetts

Important Dates
Extended abstracts due: June 23, 1998
Invited Talk Proposals due: June 23, 1998
Notification to authors: July 21, 1998
Camera-ready final papers due: October 16, 1998

Program Committee
Chair: Xev Gittler, Lehman Brothers
Co-Chair: Rob Kolstad, Berkeley Software Design, Inc.
Eric Anderson, University of California, Berkeley
Melissa D. Binde
Phil Cox, NTS, Inc.
Tina Darmohray, System Experts
Rik Farrow, Independent Consultant
Tim Hunter, KLA-Tencor
David Kensiski, Cisco Systems, Inc.
Kurt J. Lidl, UUNET Technologies, Inc.
E. Scott Menter, ESM Services Inc.
John Orthoefer, GTE Internetworking
John Sellens, UUNET Canada, Inc.
Marc Staveley, Marc Staveley Consulting
Ozan S. Yigit, Secure Computing Corp.

Invited Talks Coordinators
Pat Wilson, Dartmouth College
Phil Scarr, Netscape Communications Corporation

Overview
"Systems Administration in the Real World" is the theme for LISA '98.

LISA, the Systems Administration Conference, is the largest and oldest conference exclusively for system administrators. LISA is unique because the entire program is put together by veteran systems administrators who know first-hand the issues you face, and what factors are important in devising solutions. LISA offers the most comprehensive program for systems administrators at all levels of experience, supporting a variety of platforms and managing sites of all sizes.

Systems administration is a recognized and valued professional skill, and one critically essential in production, mission-critical environments. In addition to possessing sophisticated technical skills and needing to enhance skills to keep up with rapidly changes in technologies and tools, sys admins must be on top of social, business even legal issues that face their organizations. How can you keep up?

Whether you are a novice or senior-level sysadmin, or the manager of sysadmins, this year's program will have plenty to offer. Many different types of learning options are offered—intensive tutorials, refereed papers, invited talks, vendor demos in the trade show, and after-hours Birds-of-a-Feather sessions—with lots of discussion and chances to get together with your peers. LISA is a very informal conference, and nobody—even industry luminaries—stands on ceremony. It's a terrific place to hang out and learn from your fellow system administrators.

Tutorial Program, December 6-8, 1998
Gain mastery of complex techniques and technologies and you'll get immediate payoff within your organization. You can choose from up to 40 full- and half-day classes over three days. Whether you are a novice or senior system administrator, you will be able to find a tutorial to meet your needs. Tutorials cover important topics such as: performance tuning, administering Windows NT, Perl, TCP/IP troubleshooting, security, networking, network services, sendmail, Samba, legal issues, and professional development.

Technical Sessions, December 9-11, 1998
The three days of technical sessions will deliver specific, useful knowledge, derived from experience, and applicable to managing the real-life issues you face daily.

This year the technical sessions will be marked by flexibility in the kind of session formats offered. In addition to the traditional refereed papers, Works-in-Progress (WIP) reports and invited talks by leaders in the field, focused panels, mini-tutorials, jump-start talks, technical overviews and the like will be selected to promote enjoyable and interactive learning environments. They will offer systems administrators a chance to learn of the very latest developments in important technologies, hear about solutions which have worked for your peers, and survey what's happening in topic areas of particular concern.

The refereed papers will provide the latest findings on cutting-edge technologies. Refereed papers may be academic in nature, designed to advance the field of systems administration, or they may report practical solutions to specific problems.
Conference Topics
The Program Committee invites you to contribute to the LISA conference. Submissions of refereed papers or other presentations which address the following topics are particularly timely. Presentations addressing other areas of general interest are equally welcome.

- Innovative system administration tools and techniques
- Distributed or automated system administration
- Integration of emerging technologies in system administration
- Incorporation of commercial system administration technology
- Experiences supporting large sites (1000 users or machines)
- Experiences supporting nomadic and wireless computing
- High availability solutions
- Integration of heterogeneous platforms including legacy systems
- Managing enterprise-wide email
- Disaster recovery solutions
- OS/platform migration strategies
- Performance analysis and monitoring
- Data management
- Security

Invited Talk Track Proposals
If you have a topic of interest to system administrators, but it is not suitable as a refereed paper, please submit a proposal to the Invited Talk coordinators. Please email your proposal to itilisa@usenix.org.

Tutorial Program Proposals
To provide the best possible tutorial offerings, USENIX continually solicits proposals for new tutorials. If you are interested in presenting a tutorial at this or other USENIX conferences, please contact the tutorial coordinator:

Daniel V. Klein
Tel: 1-412-421-0285
Email: dsk@usenix.org
Fax: 1-412-421-2332

Birds-of-a-Feather Sessions
Birds-of-a-Feather sessions (BoFs) are very informal gatherings organized by attendees interested in a particular topic. BoFs will be held Tuesday, Wednesday, and Thursday evenings. BoFs may be scheduled in advance by phoning the Conference Office at 1-714-588-8649 or via email to conference@usenix.org. BoFs may also be scheduled at the conference.

Cash Prizes
Cash prizes will be awarded at the conference for the best paper and the best student paper within the refereed paper track.

How to Submit a Refereed Paper
An extended abstract of two to five pages is required for the paper selection process. Full papers are not acceptable at this stage; if you send a full paper, you must also include an extended abstract. Include references to establish that you are familiar with related work, and, where possible, provide detailed performance data to establish that you have a working implementation or measurement tool.

Submissions will be judged on the quality of the written submission, and whether or not the work advances the state-of-the-art of system administration. For more detailed author instructions and a sample extended abstract, send email to lisa98authors@usenix.org or call USENIX at 1-510-528-8649.

Note that LISA, like most conferences and journals, requires that papers not be submitted simultaneously to more than one conference or publication, and that submitted papers not be previously or subsequently published elsewhere for a certain period of time. Papers accompanied by non-disclosure agreement forms are not acceptable and will be returned unread. All submissions are held in the highest confidence prior to publication in the conference proceedings, both as a matter of policy and as protected by the U.S. Copyright Act of 1976.

At least one author of each accepted paper presents the paper at the conference. Authors must provide a final paper for publication in the conference proceedings. Final papers are limited to 20 pages, including diagrams, figures and appendices. Complete instructions will be sent to authors of accepted papers.

To discuss potential submissions and for inquiries regarding the content of the conference program, contact the program chair:

Xev Gittler
Email: xev@leman.com
Tel: 1-201-524-4160

Where to Submit
Please submit an extended abstract for the refereed paper track by two of the following methods:

Email to: lisa98papers@usenix.org
Fax to: 1-510-548-5738
Mail to:
LISA '98 Conference
USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA USA 94710

Authors: Please include the following (in a separate email message, in ascii format please, if the abstract is submitted electronically) providing:

- The title and authors of the manuscript.
- The name of one author who will serve as a contact, with postal and electronic mail addresses, daytime and evening telephone numbers, and a fax number.
- An indication of which, if any, authors are full-time students.

Products Exhibition, December 10-11, 1998
See demonstrations of innovative solutions which can put you ahead of your systems, network and internet management tasks. The exhibition lets you preview in operation products you've heard about and get the details from well-informed vendor representatives. Compare solutions quickly and save hours of research looking for products and services you need!

VENDORS: You can reach 2000 highly qualified system administrators eager to purchase system administration, network, intra/internet and other solutions. Email: cynthia@usenix.org
Announcement and Call for Papers

1st International SANE Conference

November 18-20, 1998
Maastricht, The Netherlands

Organized by the NLUUG, the UNIX User Group – The Netherlands, co-sponsored by USENIX and Stichting NLnet

Overview

Technology is advancing, the system administration profession is changing rapidly, and you have to master new skills to keep pace. At the International SANE (System Administration and Networking) conference you can join the community of system administrators while attending a program that brings you the latest in tools, techniques, security, and networking. You can learn from tutorials, refereed papers, invited talks, and Birds-of-a-Feather sessions. Visit the Vendor Exhibition for the hottest products and the latest books available. The official language at the conference will be English. The conference will be located at the Maastricht Exposition and Conference Center, MECC.

Tutorial Program & Technical Sessions

On Wednesday November 18, 1998, up to four in-depth tutorials will be presented to you by the most popular and widely acclaimed speakers.

Two days of technical sessions, including keynote address, presentations of refereed papers, and invited talks will follow the tutorial day.

Conference Organizers

Program Co-chairs:
Edwin Kremer, Dept. of Computer Science, Utrecht University
Jan Christiaan van Winkel, AT Computing

Program Committee:
Jos Alsters, C&CZ, KU Nijmegen
Bob Eskes, ASR, Hollandse Signaalapparaten
Peter den Haan, C&CZ, KU Nijmegen
Patrick Schoo, Department of Mathematics, Utrecht University
Michael Utermöhl, Dept. of Computer Science, University of Paderborn
Jos Vos, XIOS Experts in Open Systems
Elizabeth Zwicky, Silicon Graphics, Inc.

Event Organization:
Chiel van Gennip, Hiscom
Mariëlle Klatten, NLUUG
Monique Rours, NLUUG

Important Dates

Extended abstracts due: April 17, 1998
Notification to speakers: May 8, 1998
Final papers due: September 4, 1998

Complete program and registration information will be available in June 1998. To receive information about the conference, please contact: sane98-info@nlrug.nl
or visit the conference Web site:
http://www.nlug.nlevents/sane98/

Conference Topics

Presentations are being solicited in areas including but not limited to: Security tools and techniques. Managing enterprise-wide email (what about UCE?). Experiences with free software, including operating systems, a professional environment. Innovative system administration tools & techniques. Distributed or automated system administration. Incorporation of commercial system administration technology. Adventures in nomadic and wireless computing. Intranet development, support, and maintenance. Integrating new networking technologies. Integration of heterogeneous platforms. Performance analysis, monitoring and tuning. Support strategies in use at your site. Effective training techniques for system administration and users.

Invited Talks

If you have a topic of interest that is not (yet) very well suited for a refereed paper submission, please submit a proposal for an invited talk to the Program Committee at the address:
sane98@nlrug.nl

Refereed Paper Submissions

An extended abstract of up to four pages is required for the paper selection process. Abstracts accompanied by nondisclosure agreement forms are not acceptable and will be returned unread. Authors of accepted submissions must provide a final paper for publication in the conference proceedings. Final papers are held in the highest confidence prior to publication in the conference proceedings. Authors agree with publication of the final paper in the members-only area on the NLU-UGWWW site and/or the conference CD-ROM. Please submit extended abstracts by one of the following methods:
E-mail to: sane98@nlrug.nl
Fax to: +31 20 6950018
Postal mail to:
NLUUG PO Box 22727
1100 DE AMSTERDAM The Netherlands
The Internet Navigator, 2ed
Paul Gilster
1-05260-4 $24.95 member price: $21.20
# of copies: __________

Advanced Topics in UNIX
Ronald Leach
1-03663-3 $24.95 member price: $21.20
# of copies: __________

Introduction to Client Server Systems
Paul Renaud
1-57774-X $34.95 member price: $29.70
# of copies: __________

Portable UNIX
Douglas Topham
1-57926-2 $14.95 member price: $12.71
# of copies: __________

UNIX, Self-Teaching Guide
George Leach
1-57924-6 $19.95 member price: $16.95
# of copies: __________

Object Oriented Programming with Turbo C++
Keith Weiskamp
1-52466-2 $24.95 member price: $21.20
# of copies: __________

Obfuscated C and Other Mysteries
Don Libes
1-57805-3 $39.95 member price: $33.96
# of copies: __________

Finding It On the Internet
Paul Gilster
1-03857-1 $19.95 member price: $16.95
# of copies: __________

Internationalization: Developing Software for Global Markets 1-07661-9 (pub. date: 1/95)
Tuoc Luong $29.95 member price: $25.45
# of copies: __________

Adventures in UNIX Network Applications Programming
Bill Riemen
1-52858-7 $39.95 member price: $33.96
# of copies: __________

UNIX Shell Programming, 3e
Lowell Jay Arthur
1-59941-7 $29.95 member price: $25.45
# of copies: __________

The UNIX Command Reference Guide
Kaare Christian
1-85580-4 $32.95 member price: $28.01
# of copies: __________

Berkeley UNIX: A Simple & Comprehensive Guide
James Wilson
1-61582-X $40.95 member price: $34.80
# of copies: __________

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
AN INTRODUCTION TO NATURAL COMPUTATION
Dana H. Ballard
"This is a wonderful book that brings together in one place the modern view of computation as found in nature. It is well written and has something for everyone from the undergraduate to the advanced researcher."
— Terrence J. Sejnowski, Howard Hughes Medical Institute at The Salk Institute for Biological Studies
304 pp. $45

SOFTWARE AGENTS
edited by Jeffrey M. Bradshaw
A comprehensive survey of the state of the art in the design and use of intelligent software agents and in the creation of communication ability between agents.
450 pp. $40 paper

NOW IN PAPERBACK
ARTIFICIAL MINDS
Stan Franklin
"A highly imaginative and original synthesis of work in the sciences of the artificial and the natural. The book combines the best of cognitive science research with artificial intelligence theorizing, building a bridge between areas usually a chasm apart. I enjoyed it, too!"—Robert Ornstein, psychobiologist and author of The Roots of Self, The Evolution of Consciousness, and Multimind
A Bradford Book • 464 pp., 95 illus. $17.50 paper

SLAVES OF THE MACHINE
The Quickening of Computer Technology
Gregory J. E. Rawlins
Rawlins argues that it is lack of basic knowledge that threatens to make us slaves to computers, and he shows how we can take control of them once more. Amusing, thought-provoking, and packed with information, it will put you "under the hood" of the present-day computer and those of the future.
A Bradford Book • 240 pp. $25

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. $27.50 paper

COMPUTABILITY AND COMPLEXITY
From a Programming Perspective
Neil D. Jones
"This is an introduction to the basic concepts of computability, complexity, and the theory of programming languages. The author knows very well all three subjects, has made important contributions to them, has original insights and delightful personal points of view, and overall has good taste. I know of no previous book that provides a comprehensive introduction to all three subjects."
—Christos H. Papadimitriou, University of California at Berkeley
Foundations of Computing series
454 pp., 25 illus. $40

ORIGINAL IN PAPERBACK
GREAT IDEAS IN COMPUTER SCIENCE
A Gentle Introduction
second edition
Alan W. Biermann
Presents the "great ideas" of computer science that together comprise the heart of the field. This second edition has new chapters on simulation, operating systems, and networks. In addition, the author has upgraded many of the original chapters based on student and instructor comments, with a view toward greater simplicity and readability.
568 pp., 264 illus. $37.50
The Perl Resource Kit—UNIX Edition gives you the most comprehensive collection of Perl documentation and commercially enhanced software tools available today. Developed in association with Larry Wall, the creator of Perl, it's the definitive Perl distribution for webmasters, programmers, and system administrators.

The Perl Resource Kit provides:

- Over 1,800 pages of tutorial and in-depth reference documentation for Perl utilities and extensions, in 4 volumes.
- A CD-ROM containing the complete Perl distribution, plus hundreds of freeware Perl extensions and utilities—a complete snapshot of the Comprehensive Perl Archive Network (CPAN)—as well as new software written by Larry Wall just for the Kit.

Essential Perl Software Tools All on One Convenient CD-ROM

Experienced Perl hackers know when to create their own, and when they can find what they need on CPAN. Now all the power of CPAN—and more—is at your fingertips. The Perl Resource Kit includes:

- A complete snapshot of CPAN, with an install program for Solaris and Linux that ensures that all necessary modules are installed together. Also includes an easy-to-use search tool and a web-aware interface that allows you to get the latest version of each module.
- A new Java/Perl interface that allows programmers to write Java classes with Perl implementations. This new tool was written specially for the Kit by Larry Wall.

Experience the power of Perl modules in areas such as CGI, web spidering, database interfaces, managing mail and USENET news, user interfaces, security, graphics, math and statistics, and much more.

For more information, go to: http://perl.oreilly.com/log or call 800-998-9938.
Local User Groups

UNIX and LINUX Groups

The USENIX Association will support local user groups by doing a mailing to assist in the formation of a new group and publishing information on local groups in login; and on its Web site. Full details can be found at: <http://www.usenix.org/membership/LUGS.html>.

At least one member of the group must be a current member of the Association.

Send additions and corrections to: <login@usenix.org>

California

Bay Area
Bay Area FreeBSD User Group

Orange County
UNIX Users Association of Southern California (UUASC)

Colorado

Boulder
Boulder Linux Users Group
Front Range UNIX Users Group

Connecticut

The Connecticut Free UNIX Group

District of Columbia

Washington
Washington Area UNIX Users Group

Florida

Orlando
Central Florida UNIX Users Group
Western
Florida West Coast UNIX Users Group

Georgia

Atlanta
Atlanta UNIX Users Group

Kansas/Missouri

Kansas City
Kansas City UNIX Users Group (KCUUG)

Massachusetts

Worcester
WPI Linux Association
Worcester Linux User's Group

Michigan

Detroit/Ann Arbor
Southeastern Michigan Sun Local Users and Nameless UNIX Users Group

Missouri

St. Louis
St. Louis UNIX Users Group

New England

Northern New England UNIX Users Group (NNEUUG)

New Mexico

Albuquerque
ASIGUNIX

New York

New York City
Unigrouip of New York City

Oklahoma

Tulsa
Tulsa UNIX Users Group, SUSR

Texas

Austin
Capital Area Central Texas UNIX Users Society (CACTUS)

Dallas/Fort Worth
Dallas/Fort Worth UNIX Users Group

Houston
Houston UNIX Users Group (HOUNIX)

Washington

Seattle
Seattle UNIX Group

Armenia

Yerevan
The Armenian UNIX Users Group (AMUUG) was founded in December 1996. AMUUG is open to all interested individuals and organizations, regardless of affiliation, without any fee.

Canada

Alberta
Calgary UNIX Users Society (CUUG)

Manitoba
Manitoba UNIX User Group (MUUG)

Ontario

Toronto Group
Ottawa Carleton UNIX Users Group (OCUUG)
Ottawa Carleton Linux Users Group (OCLUG)
System Administration Groups

SAGE supports local groups. Full listing of group Web sites and other details can be found at:

http://www.usenix.org/sage/locals/

**ASUQ**
Meets first Wednesday of every third month in Montreal, Quebec.

**AZSAGE**
Meets monthly in the Phoenix area.

**BayLISA**
Serves the San Francisco Bay Area.

**Back Bay LISA (BBLISA)**
Serves the Boston, Massachusetts and New England area.

**Beach LISA**
A group for system administrators in the San Diego area hosts monthly meetings and the occasional social event.

**dc.sage**
Serves the Washington D.C. Area.

**Dallas-Fort Worth SAGE (dfwsage)**
Serves the North Texas area.

**$GROUPNAME**
Serves the New Jersey area.

**EnglishBayLISA**
Serves the Vancouver, British Columbia and the British Columbia lower mainland.

**Houston Area Sysadmins (HASH)**
Serves the greater Houston Area. Join the mailing list by sending a subscribe message to hash-request@tree.egr.rice.edu

**Los Angeles Area Group**
A group in Los Angeles is being formed. Please contact Josh Geller (joshua@cae.rothix.com) if you are interested.

**New York Systems Administrators (NYSA)**
Serves the New York City area.

**North Carolina System Administrators**
Serving central North Carolina, particularly the Triangle Area.

**Old Bay SAGE**
A group for sysadmins in the greater Baltimore Maryland area.

**SAGE-AU**
The System Administrators Guild of Australia

**Seattle SAGE Group (SSG)**
Seattle, Washington area.

**Twin Cities Systems Administrators (TCSA)**
Serving the Twin Cities and surrounding areas of Minnesota.
Reality Check

It’s the new year. Time for resolutions, prognostications, and taking stock. I think it’s time for a reality check.

Most of us work in the fastest-paced industry that has ever existed. Many of us have a technological bent. We watch the industry, toil on our keyboards, in meetings, or just in our heads.

USENIX focuses mostly on those of us who are technical rather than members of sales organizations, executive management, marketing, finance, or human relations (to name a few). This is not to say that many of us don’t work within these other kinds of organizations, but rather to emphasize that our goals are more often the technical.

So what reality should we check?

This month, let’s start with trade magazines and their audience. My goodness but they are full of great stories! The CIO at one company saved zillions in one year; changing to a new database will make your company successful; Windows is the answer; UNIX is the answer; etc., etc.

I regret to say that I’m more confused by most of the articles than I am enlightened. I understand that everyone must make a living, including reporters, but sometimes the articles do tend too much toward product promotion and not enough toward the sort of balanced practical advice that I think they think they’re writing about. At least that’s what it looks like to me.

I am always hoping for a bit more balance in these articles, something like “Well, we changed over to SchlockMeister 2.0 and things got 10% better. We were hoping for 20%.” Instead, articles shout headlines that sound like “SchlockMeister 2.0 Speeds Cure for Cancer.” Cute, but not very enlightening.

We sure don’t hear too many complaints about the articles in various trade magazines, despite the fact that they do tend to hype things a bit. Why is that? I don’t know. Plenty of my acquaintances believe every word printed and never check up on the story-behind-the-story.

Maybe this is because of the dramatic rise in “computer literacy.” Almost everyone in USENIX members’ companies uses a personal computer. Some small fraction of these people has a “knack” for dealing with them and another small fraction has an extremely difficult time. (My frequent racketball opponent bemoans his lack of ability in things technical; he can, however, draw or illustrate almost anything. I can not. Period. He felt much better after learning this.)

Regrettably, those who use word processors, spreadsheets, the Web, and electronic mail are not necessarily those who should be making decisions about scalability, electronic mail servers, and worldwide networks, much less a “commitment to XYZ software company.” And too often they don’t know this. This forms the heart of a big problem, if you ask me.

I’m resolving this year to try to help my customers, my associates, and those people I meet through my business to make sure that they are dealing with reality and not promises, hopes, or dreams. I am going to try to help them evaluate their solutions and hold their solution providers accountable for those solutions. I hope and pray that my “reality” is one that is close to some notion of the “real reality” and that I am not just a polished religious zealot of some sort.

How goes reality in your professional life? Let me know and I’ll publish a summary if it’s interesting.
Linux, FreeBSD, OpenBSD, Samba, NetBSD, and more

FREENIX
The Freely Redistributable Software Track at the USENIX Technical Conference
June 15-19, 1998  New Orleans, Louisiana

FREENIX is the showcase for the latest developments and interesting applications in freely redistributable software including Linux, FreeBSD, OpenBSD, NetBSD, GNU, etc.

A special track within the USENIX Annual Technical Conference, FREENIX attendees may choose among all of the conference offerings and informal get-togethers, including

- FreeBSD, Linux, OpenBSD, NetBSD, GNU, Samba, etc.
- Tutorials for in-depth instruction
- Keynote speakers
- Refereed presentations
- Invited talks
- Works-in-Progress reports
- Birds-of-a-Feather sessions
- “The Guru is IN”
- Products Exhibition

FREENIX Committee Chair:
Jon “maddog” Hall, Digital Equipment Corporation

For the detailed USENIX conference program (available March) including FREENIX, special events and registration online, go to http://www.usenix.org/events/no98/ or telephone for the printed brochure 1.714.588.8649.

Sponsored by the USENIX Association

Co-sponsored by:
The FreeBSD Project www.freebsd.org
Linux International www.li.org
The OpenBSD Project www.openbsd.org
The NetBSD Project www.netbsd.org
CONTRIBUTIONS SOLICITED
You are encouraged to contribute articles, book reviews, and announcements to {login}. Send them via email to <login@usenix.org> or through the postal system to USENIX Association office.
Send SAGE material to <tmd@usenix.org>. The Association reserves the right to edit submitted material. Any reproduction of this magazine in its entirety or in part requires the permission of the Association and the author(s).
The closing dates for submissions to the next two issues of {login} are April 7, 1998 and June 9, 1998.

ADVERTISING ACCEPTED
{login} offers an exceptional opportunity to reach 9,000 leading technical professionals worldwide.
Contact: Cynthia Deno
408 335 9445