# RELEASE 1.0

11-98 | ESTHER DYSON'S MONTHLY REPORT | 19 NOVEMBER 1998

## OPEN MIND, OPEN SOURCE

*By Esther Dyson*

*At a time when Microsoft's business practices are under attack in the courts, its development and business models are under attack at home. Much as IBM's big problems two decades ago were in its business rather than in the antitrust courts, Microsoft should probably be paying more attention to Open Source than to Open Courts.*

*Open source (described at length below) is basically software developed by uncoordinated but collaborating programmers, using freely distributed source code and the communications facilities of the Net.*

Of course, history repeats itself, but never exactly. We doubt Microsoft will cede to its competition the way IBM did. More likely, just as Microsoft did with the Internet, it will recognize the threat of Open Source to its business model and figure out how to embrace and exploit it. (Or perhaps open source is just one more piece of the Internet "threat," and we're only in the middle of that war.)

In the case of open source, Microsoft is perhaps lucky that the trial is drawing attention from what otherwise might be big news: the Halloween Document. This is an internal memo from Microsoft that somehow found its way onto the Web (at www.opensource.com/halloween1.html). It is one of the great artifacts of software history, a found document laying out some of the fundamental questions of business and development models. Please read it along with the issue below.

In this document Microsoft employee Vinod Valloppillil lays out lucidly the threat posed not by Linux in particular but by the open-source approach in general. As noted by open-source pioneer Eric Raymond in his annotations, what Microsoft keeps missing is that open source is not just about bug-fixing, which can be done in a massively parallel fashion. It's about competing innovations, which live or die in a market of other people deciding to adopt/enhance them, or ignore them. Interestingly, this is an information market, unencumbered by prices on the one hand or by bundling deals or marketing on the other. ======>

**PC FORUM INVITATIONS IN THE MAIL!**

**INSIDE**

(Although probably a new feature from, say, BIND creator Paul Vixie would draw more trials than one from Juan or Alice.) The Microsoft approach keeps asking: Where is the leader? And the OS approach keeps answering: The leader is whoever is ahead, by acclamation.

Now the Microsoft approach is clearly the best one for building a company; Bill Gates has done that par excellence. But it might not be the best approach for building commodity software.

And here is the crux of the matter: One major way to make money is to lead a market into commodity-hood — sell something at a value-based premium while your cost structure matches that for a commodity. As other people enter the market, the leader can enjoy high profits while competitors scramble to catch up. But over time the market does turn into a commodity one. For now, the open source still has a shortage of enterprise-oriented development tools, infrastructure and support. But as attention and capital flow into the open-source world, that will change.

That's the challenge that Microsoft is now facing in systems software — and perhaps what was once known as applications but is now increasingly user tools for manipulating an information-rich environment on the Web.

One approach is to fight by fostering proprietary protocols (a.k.a. integration), as outlined in the Halloween document. But Microsoft in general and Bill Gates in particular have shown a unique ability to learn. So the real question is: How could Microsoft apply the open-source business model? Perhaps the company will break itself up, without action by the DOJ, into a collection of content-focused businesses that use OS software as a base. They may never reach the vast economies of scale and mass production that Microsoft has enjoyed recently, but that era is over. Alternatively, it could extend its consulting and services business — a reliable source of revenue, but one that depends almost linearly on the number of employees and is not as profitable as selling copies of software.

Microsoft shows a unique ability to recognize reality. With its accumulated wealth and talent pool, it is probably in a better position than any other company in the world to succeed in the future if it can successfully leave the past behind.

**About this issue**

A note on the context for this issue: We are grateful to the work of Tim O'Reilly and his editorial team for the content of this issue. As a big promoter in the concept and an investor in some of its instantiations, O'Reilly is clearly partisan, but he is partisan for the concept rather than for particular products or companies. We believe he did a fair job here of focusing on business issues, and we can't imagine finding anyone more qualified. Consider it an experiment in editorial approach much as Open Source is an experiment in business and development models. Yes, Open Source may be a moral issue to some, but for most it is also a matter of taste and business models. Read below, and judge for yourself.

*(Some adherents prefer to capitalize Open Source, as in the Open Source movement. We use lower case in order to make it clear we see it as the open-source model, although indeed it is both.)*

**THE OPEN-SOURCE REVOLUTION**
*By Tim O'Reilly**

**Free software becomes open source**

*It starts like an X Files episode, with a series of puzzling statistics:*

*A university student in Finland writes an operating system kernel and gives it away. Within a half-dozen years, Linux boasts more than 7 million users, and is the fastest-growing server operating system.*

*An administrator at a Website development company starts a mailing list to share patches to a free server no longer being maintained by its creators. Within a couple of years, the "patchy server" runs more than 50 percent of all visible Websites, and has dominant market share despite hundreds of millions of dollars spent by commercial vendors hoping to unseat it.*

*Starting from their dorm room, using only freely-available software, a couple of college students build not just a new business, but a new multi-billion dollar industry.*

*The most important computer standards body in the world consists of whoever participates in the mailing lists and shows up at meetings held three times a year.*

*Something strange has been going on right under our noses. Just what it is and why it's important is the subject of this issue of Release 1.0.*

A year ago, if you had asked the IS manager at a large company about free software, he'd have told you he didn't use it. It's unsupported, he might say. Not robust enough. Not commercial quality.

Suddenly, in a flood of mainstream activity, that perception has changed.

In January, Netscape shocked the industry by deciding to make the source code for its flagship Communicator product freely available. While there will be an official Netscape version of the product, anyone who wants to build and sell competing products on the code base is free to do so. Netscape's decision was inspired by Eric Raymond's ground-breaking paper, *The Cathedral and the Bazaar* (www.tuxedo.org/~esr/writings/cathedral-bazaar), which argues that a key to both the technical and market success of Linux is a distributed software development methodology based on freely redistributable source code. Raymond claims that open-source software is not only more robust than comparable proprietary code, it is better supported and more innovative.

In April, O'Reilly & Associates (owned by the author of this issue) hosted a high profile meeting of leading free software developers to discuss the implications of Raymond's paper, and to launch an effort to raise the profile of free software. A key outcome of the meeting was the developers' agreement to use Raymond's term "open-source software" in order to escape the baggage associated with "free software." (Despite the dual meaning of

* Contributors to this issue include Mark Jacobsen, Stig HackVän, Mark Stone and Dale Dougherty.

free — “Think ‘free speech,’ not ‘free beer,’” says Richard Stallman, originator of the term — people have come to think that free software must necessarily be non-commercial, rather than simply non-proprietary.)
In relatively quick succession, Corel, Informix and Oracle announced that they have already or will soon be porting their products to Linux.
In July, IBM announced that it was joining the Apache Group, making the Apache Web server the core of its WebSphere product line. It is dedicating a team of programmers to helping to develop Apache on the NT platform.

In September, Intel, Netscape and leading venture capital firms Greylock and Benchmark Partners invested in Linux distributor Red Hat Software.

Meanwhile, people are beginning to notice just how much open-source software is at the heart of the Internet infrastructure: The DNS runs on BIND, the Berkeley Internet Name Daemon; sendmail touches virtually every piece of e-mail sent over the Net; languages such as Perl, Tcl, and Python are heavily used at such Websites as Yahoo!, C|NET and Amazon.com.

Suddenly open source is the Next Big Thing.

But just what is it and how do companies take advantage of it?

**The native development methodology of the Internet**

In the early days of UNIX and the Internet, programmers routinely shared their source code. AT&T was prohibited from competing in the computer industry, so Ken Thompson and Dennis Ritchie sent out UNIX source tapes from Bell Labs for a nominal copying fee. Hundreds, then thousands of programmers built on their work, contributing individual utilities, porting the software to additional platforms, and in general, doing whatever it took to make the software meet their own unique needs.[1]

Among the most urgent of those needs was for easier and faster ways to share their work.
A network, originally based on a dial-up protocol called UUCP (Unix-to-Unix Copy Protocol), grew up to facilitate communication and code-sharing. Anyone could join: All you needed was an existing site that was willing to connect to you.

Meanwhile, universities and research institutions connected via dedicated high-speed links subsidized by the US government used another protocol, TCP/IP (Transmission Control Protocol/Internet Protocol). The UUCPnet and the TCP/IP-based Arpanet shared a massively redundant message-passing system called Usenet. The original purpose of Usenet was to distribute computer source code and related information and discussions. Quickly, it became a social medium as well.

In 1991, two events happened which transformed the computer industry:

[1] *Meanwhile, in Russia and other places without communication, programmers operated one to an installation and virtually all applications and much system software forked into thousands of unique variations unsupportable by anyone but their creator. In other words, the source was open but little of the derivative work ever converged back or were shared.*

Rick Adams, a system administrator at the US Geological Survey, which hosted the largest Usenet news hub, decided that the time was ripe for a commercial Usenet infrastructure. He founded UUNet to provide commercial UUCP and TCP/IP network services.

Tim Berners-Lee, a programmer at the CERN high-energy physics lab in Geneva, came up with a hypertext-based client-server system for information distribution, which he called the World Wide Web.

Everyone knows the history of the commercial Internet and the Web from that point on. A team at the University of Illinois built the Mosaic browser based on Berners-Lee's work; the team was scooped up to found Netscape. Microsoft got the Internet religion, commercial Websites proliferated, and by 1997, the battle between Netscape and Microsoft for control of the Web made many think the Internet was somehow a product of these two companies.

In fact, the Internet was created by a community of independent developers who built the tools they needed to collaborate more and more effectively.

A couple of snippets from Raymond's paper The Cathedral and the Bazaar capture the flavor of the movement:

> "The best hacks start out as personal solutions to the author's everyday problems, and spread because the problem turns out to be typical for a large class of users."
>
> "The [open source] world behaves in many respects like a free market or an ecology, a collection of selfish agents attempting to maximize utility which in the process produces a self-correcting spontaneous order more elaborate and efficient than any amount of central planning could have achieved."
>
> "Good programmers know what to write. Great ones know what to rewrite (and reuse).... They know that you get an A not for effort but for results, and that it's almost always easier to start from a good partial solution than from nothing at all."
>
> "Release early. Release often. And listen to your customers.... Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging.... Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone.... Or, less formally, 'Given enough eyeballs, all bugs are shallow.'"

There are thousands of open-source projects, each incorporating the work of many developers. Here are some of the most important:

**Linux.** Built on top of Linus Torvalds' kernel, Linux distributions typically include hundreds of other open-source packages. (However, most of those packages also run on other platforms, including proprietary versions of UNIX, Windows and Windows NT, MacOS, and many others.)

**FreeBSD, OpenBSD** and other Berkeley UNIX derivatives. While Linux gets the lion's share of the attention, BSD UNIX systems also have a significant following.

**Programmer's Tools.** The Free Software Foundation's GNU project has created a high-quality set of programmer's utilities, including the gcc C compiler, the g++ C++ compiler, the emacs editor, the gdb debugger. Two other programming tools developed outside the GNU project that are an absolutely indispensable part of the open-source culture are Larry Wall's patch program, which allows developers to exchange small fixes to programs rather than having to ship around the source code for the entire program, and CVS, the Concurrent Versioning System, which allows developers to maintain multiple versions of the source tree.

**Languages.** Larry Wall's Perl language is the undisputed king of the open-source programming languages, but John Ousterhout's tcl and Guido van Rossum's Python language also have thriving communities. (Wall is now an employee of O'Reilly Associates.)

**Apache.** As noted earlier, the Apache Group has dominant web server market share. The most recent Netcraft Web server survey (www.netcraft.co.uk/survey) shows Apache with 53 percent of all visible web servers, followed by Microsoft's IIS at 23 percent and Netscape at 7 percent. The Apache Group was formed in 1995 by a group of Website administrators to share their modifications to the original NCSA Web server, which had been abandoned by NCSA after its creator, Rob McCool, left the agency. (NCSA is the National Center for Supercomputing Applications at the University of Illinois, which also created the Mosaic browser.) Apache continues to be developed and maintained by a group of about twelve core developers, plus a large and active user community.

**Samba.** Developed by a worldwide team headed by Andrew Tridgell in Australia, Samba allows UNIX and Linux systems to act as file and print servers on NT and Windows 95/98 networks. This is a "stealth" technology that has allowed administrators to work Linux into their networks without the knowledge of management. Silicon Graphics recently hired one of the key Samba developers, realizing that Samba was a great tool to help sell SGI server hardware.

**Sendmail.** Originally developed as part of Berkeley UNIX, sendmail is the backbone of the Internet's e-mail server infrastructure.

But looking at just the best-known packages misses the point. When you scrape the surface, open-source software is everywhere. Virtually every vendor's TCP/IP stack (including Microsoft's) is based on one originally developed as part of the Berkeley UNIX networking package. The Internet infrastructure and its standards process are prime exemplars of the open-source movement. And at the current cutting edge, most of the commercial XML packages build on the open-source xml parser written by James Clark, an independent programmer living in Thailand who also wrote many of the GNU text-processing tools.

### Sizing the Open-Source Community

Estimating the market size of the different open-source user communities can be challenging since there are no definitive surveys or studies. To make our calculations we started with the generally accepted estimate
of 7 million Linux users. (See Red Hat's "Sizing the Linux Market" white paper at www.redhat.com and Software Magazine of September 1998.)

We then compared the number of users who made postings in the different newsgroups associated with each open-source community (as measured by the Netscan analysis tools located at netscan.sscnet.ucla.edu for the period from September 2, 1998 until October 2, 1998). From the size of these groups, we extrapolated the number of active users for each community using as a base the 7 million Linux users number.

**Projected Size of Open-Source Communities**

| | Number of Posters | Estimated Size of User Community |
|---|---|---|
| **Linux** | 13,231 | 7,000,000 |
| **Perl** | 2,023 | 1,000,000 |
| **BSD** | 1,820 | 960,000 |
| **Apache** | 738 | 400,000 |
| **Sendmail** | 670 | 350,000 |
| **Python** | 612 | 325,000 |
| **Tcl/tk** | 563 | 300,000 |
| **Samba** | 309 | 160,000 |

Many of these projections are in the ballpark of estimates made by each open-source community. In addition, attendance at O'Reilly's Perl Conference, the Apache Group's ApacheCon, Usenix's Tcl/Tk Conference, and the PSA Python Conference has been consistent with these figures. In short, while our estimates are not statistically defensible, they are surprisingly predictive.

The one anomaly in these figures is the relatively low numbers for the Apache and sendmail user communities. After all, the automated Netcraft survey has found over 1 million Websites running Apache. Similarly, extrapolating from the number of Internet hosts found by Network Wizards' domain name survey (www.nw.com/zone/WWW/top.html), Sendmail, Inc., projects a minimum of 600,000 sendmail servers on the public Internet, with perhaps twice as many more on internal networks.

These numbers make more sense when you consider that Apache and sendmail aren't end-user products. For example, Apache is the server of choice for Internet service providers and Web-hosting companies. A ratio of 2.5:1 in the number of Websites to the number of administrators actually running those sites is not surprising. Similarly, a small number of administrators at a large company can be responsible for dozens of mail servers.

## BUSINESS MODELS

For Release 1.0 readers, this may all be very interesting, but one big question remains: How can you make money with open source?

As we'll argue later, many companies are making money because of open-source software. A smaller number are making money actually selling open-source products or derivatives. We have identified three basic models:

### Branding and distribution

One way to make money with open source is to deliver and charge for the latest software release under a trusted brand name and an existing or newly created distribution channel. Coupled with the trusted brand name is usually a promise of technical support for a short installation period and additional support packages for problems that may surface thereafter.

Red Hat Software (www.redhat.com) is the most prominent example of this business model. In order to understand Red Hat's business, you first need to understand that Linux is really an aggregate of many different open-source products, built on top of Linus Torvalds' kernel. As Linux became increasingly popular, there were many different distributions, each incorporating different open-source packages, and including drivers for different hardware devices.

In 1994 Marc Ewing, a graduate of Carnegie Mellon's computer science program, was working at IBM. In his spare time building an advanced development tool, he used a couple of Linux workstations. They were running a version of Linux that he had had to patch and modify substantially. When he found that he was not making the progress he expected on his project, he discovered that it was because he was spending substantial time maintaining his Linux workstations. He then decided that the world did not really need another development tool, but it did need a better Linux distribution. So he began the Red Hat Linux project to address the limitations in Linux that he personally had experienced.

Shortly thereafter, Bob Young, president of a small Connecticut-based free software distributor that was Red Hat's biggest customer, the ACC PC UNIX and Linux Catalog, made the pitch that he'd like to do more than just sell the product. He proposed that they merge their efforts, adding his marketing expertise to their development team. Young served as Chairman, ceo and chief evangelist since then, only recently replacing himself as president with Matthew Szulik, who was most recently President of Relativity Software. Red Hat completed its first financing round from angels in August 1997. In a second round in September 1998, it raised an undisclosed amount from Greylock, Benchmark Partners, Intel and Netscape. Today, Red Hat's annual revenues exceed $10 million.

Red Hat manufactures shrink-wrapped software, including versions of the Linux OS and tool sets, under the "freely distributable" GPL license (see box, page 12). The cost of the Red Hat Linux/Intel basic package is $49.95. It can be ordered online or purchased from its resellers. In addition, Red Hat offers "standard" and "gold" support packages.

Red Hat is a strong proponent of the GPL (see page 12). Young points out that part of Red Hat's success is due to lowering the transaction costs of adapting software to specialized needs. Proprietary licenses create a barrier to entry because of the requirement for negotiation and agreement.

Young says: "Open source gives our users real control over the operating system layer in the technology they are using. This is much like the control that IBM gave computer buyers when they published the specs to the original IBM PC. Consumers will take control and increased choice over closed systems and limited choice every time. This control enables them to build more reliable, stable, and secure systems at a lower cost than the proprietary binary-only alternatives."

Young encourages people to copy and resell Red Hat's entire distribution. In fact, it's downloadable from www.cheapbytes.com for only $1.99. He considers this an advertising expense rather than a loss of revenue. His strategy is founded on the principle that people buy name-brand products, even though generic products (often provided by the very same manufacturers) are widely available at lower cost.

Young actually credits Esther Dyson, editor of this newsletter, with being a key influence on Red Hat's business strategy. He recalls a 1993 issue as follows: "Her thesis was that as technology cycles shortened, the value of owning the technology decreased, and the value of owning the brand around the technology increased."

Young's most provocative statement is that his goal is to "shrink the size of the operating system market," commoditizing what he feels is currently an overpriced product category. He's not after Microsoft's multi-billion dollar business; instead, he thinks he can own a big piece of it if he can shrink it down to a $500-million market.

Young also believes that services and support, not product sales per se, are key to his business. He says: "In effect we give away the technology in order to generate the services and support revenues that all large technology companies generate. Digital Equipment Company earned more revenue from support and services than they did selling computers, and they were a $15 billion/year company."

Other open-source packagers include Walnut Creek CD-Rom, which distributes a $39.95 version of the FreeBSD OS, and S.u.S.E., the largest distributor of Linux in Europe, which offers the latest version of its Linux OS distribution, a reference book and an administration tool for $49.95. It's important to all of these companies that they contribute back to the community. Both Red Hat and S.u.S.E. develop Linux code and contribute it back to the open-source community. Walnut Creek supports the development of FreeBSD by funding several FreeBSD developer positions.

Under the branding and distribution model, purchasers, in effect, pay for three features: (i) the source code on a CD-ROM; (ii) a company's stamp of approval on the code as the (temporarily) latest and most stable version; and (iii) usually a corporate commitment to support the packaged distribution, sometimes at an additional fee. In some way, they are paying extra for open source in the way other consumers might pay for the environmental benefits of "green" products.

**Addition of proprietary value to open source**

The second business model for open source is for a company to develop proprietary products that add value to open source and then sell these products primarily to the commercial community. As proprietary efforts, these products are generally not contributed back to the open-source community.

Sendmail, Inc. (www.sendmail.com) is the best-known example of a company formed to add proprietary value to an open-source product.

Eric Allman originally wrote the sendmail program as a staff member at the University of California at Berkeley. This is a classic example of the "scratch your own itch" origin of many open-source projects. Allman's team at Berkeley had a coveted Arpanet connection; other researchers wanted access, and Allman thought it would be easier to create a program to route their mail for them than to give them accounts on his machine.

Allman's program was released under the name "delivermail" in 1979 as part of Berkeley UNIX. As a new protocol was developed for transporting mail called SMTP (Simple Mail Transport Protocol), Allman evolved delivermail into sendmail, which quickly became the dominant mail transfer agent on the Arpanet and then the Internet. (When you send an e-mail message over the Internet, your desktop mail client doesn't deliver it to its destination. Instead, it is passed off to a series of mail servers, which find the most direct route to its intended recipient.) Approximately 80 percent of all Internet sites use sendmail as their mail transfer agent; because mail often makes several "hops" in going from sender to recipient, virtually every e-mail message sent over the Internet is handled somewhere by a sendmail server.

In late 1997, Allman decided to commercialize sendmail. Recruiting former Integrated Systems and Sybase vice president Greg Olson as ceo, he quickly raised $1 million from a group of angel investors, including Sun cofounders Bill Joy and Andy Bechtolsheim, John Funk (ceo of Infobeat, an operator of opt-in e-mail marketing lists that is the world's largest sendmail user, sending out over 6 million e-mail messages per day), and publisher Tim O'Reilly (author of this newsletter). In mid-1998, Olson raised another angel round of $6 million. Olson had started talking with VCs, but the response to his presentation to the Silicon Valley Band of Angels was so overwhelming that he decided to take the money and get on with the job of building Sendmail's business.

The reasons for Allman's decision to go commercial are instructive. He says, "Sendmail had gotten to the point where I was spending all my time maintaining it. I needed more resources to get it moving forward again, with new developments that were badly needed." Another factor was that the open SMTP e-mail standards were being undermined by proprietary mail formats. He felt that sendmail's adherence to interoperable standards was an important part of the Internet's success as an e-mail transport medium. Without additional resources to keep sendmail's dominant market position, he feared that not only would the program itself be replaced by competing products, but open e-mail standards might be threatened. Like many open-source businesses, Sendmail combines idealism and practicality. Sendmail's business plan calls for the creation of commercial value-added tools and services for ISPs and enterprises for whom email is mission

critical, while continuing to drive innovation and interoperability through open-source software development. Olson figures that if the company can convert only a small percentage of the ISPs and large companies that rely on sendmail, he can build a substantial business.

His pragmatic, hybrid business model is based on traditional product-line approaches. The commercial product lines deliver enhancements targeted to the needs of commercial users, such as cost-effective, pre-tested packages with easy to use GUIs and the full range of commercial services including marketing, sales, training, technical support and consulting. The open-source product line is for Internet developers and open-source software users that need to build their messaging infrastructure on proven standards-based technology.

Sendmail's implementation of this hybrid business model brings benefits to both of the company's customer segments. New mail server functionality and standards initiatives appear first as open source. This speeds innovation and assures the quality of enhancements through inspection and use by a worldwide network of technical experts, resulting in proven, best-of-breed solutions. The commercial lines fund additional development resources for open source and for those requirements that are unique to the commercial customer base. In this way everyone wins except the purely proprietary vendors who would take over the commercial segments of the Internet with closed approaches.

Sendmail, Inc., calls its model a hybrid. It targets both the commercial and open-source markets, only one of which generates direct revenue. Sendmail's first action as a corporate entity was the release of Sendmail 8.9 to the open-source community. The company wanted to send a strong message that creating a commercial entity to profit from Sendmail did not mean that it was abandoning the open-source community. The first proprietary release, Sendmail Pro, is expected in early 1999, and will be demonstrated in December at the Usenix LISA Conference in Boston.

Vancouver-based ActiveState Tool Corp. (www.activestate.com) is another company working with a hybrid business model. In 1995, founder Dick Hardt (then doing business as Hip Communications) developed the original port of Perl to Windows under contract to Microsoft (who included it in the Windows NT Resource Kit). Because open source was not a common concept in the Windows community, Hardt released the product as what he called "binary freeware." That is, the binaries were freely redistributable, but the source code was proprietary to Hip.

In 1997, Hardt sold Hip's Website development business and created ActiveState (with funding from publisher O'Reilly & Associates, whose CEO, Tim O'Reilly, is the author of this issue) as a new company to focus on Perl for Windows. Perl has been a dominant language on UNIX and the Web; it's becoming increasingly critical for Windows automation as well.

As with Sendmail, Inc., ActiveState's first move was to reach out to the open-source community. The original Perl for Win32 port had diverged

### A Brief Introduction to Open-Source Licensing

At its best, open-source licensing fosters cooperation, sharing and symbiosis between software creators and software consumers. There are many established and successful approaches to open-source licensing. Here are three important licenses, presented from oldest to newest.

**BSD-style licenses**

BSD-style licenses (so called because they were used for the Berkeley Standard Distribution of UNIX) are the oldest and least restrictive. They give licensees the option of creating private derived works (traditional commercial software with unpublished source code). Contribution of changes back to the public version is optional.

Some in the Open-Source community resent third parties taking from the public pool of software without contributing. (In economics, this is called the free-rider problem.) But despite the lack of a mandate, voluntary cooperation abounds. BSD-licensed software provides a great deal of the Internet's functionality through BIND, Apache and sendmail.

**GNU general public license (GPL)**

The GNU GPL, authored by Richard Stallman in 1983, is the GNU Project's implementation of the Copyleft concept. While copyright provides a monopoly on the right to create copies and derivative works, Copyleft grants unlimited permission to copy and modify. However, Copyleft obligates the user to distribute, without fee or additional license terms other than Copyleft, the source code to all derivative works.

The focus of the GNU Project is "free software, where free refers to freedom and not price." You can sell free software, but you must also give away the source code. The GPL is "viral" in that one cannot combine GPLed work with work governed by different licenses. If you enhance a GPLed work, your enhancements fall under the GPL terms. The GPL has a less-viral form, the LGPL, used for function libraries.

The GPL excels at preventing the proprietary fragmentation that has caused so much harm to the UNIX market. GPL success stories include the Linux kernel, the GNU C Compiler and the Samba file server.

**Mozilla public license (MozPL or MPL)**

The MPL, authored by Netscape Communications as part of its open-source release of Communicator 5, strikes a balance between the BSD license and the GPL. Private derivative works are permitted, while changes to MPL-covered source must be made freely available on the Internet. The MPL, however, is non-viral: additions to (as opposed to modifications of) the MPL-licensed source which form a "larger work" may be licensed differently and need not be published at all.

The economics of open source are still evolving, and so are the approaches to licensing.

significantly from the UNIX Perl standard. Along with Larry Wall, Gurusamy Sarathy (author of a competing port of Perl to Windows) and a number of other developers, Hardt spent much of the company's first year working on an effort that came to be called "OnePerl" (after the mailing list where much of the work was done). The Perl 5.005 release in the summer of 1998 incorporated the OnePerl work.

Only after the release of Perl 5.005, having seeded the market with a good product, did ActiveState turn to its proprietary business. It has developed a suite of value-added Perl products for the Windows market, including a debugger, facilities for generating COM objects with Perl, a Perl variant called PerlScript that works with Microsoft's Active Server Pages, and numerous other products that allow users to take advantage of Windows specific functionality. Packages range from less than $100 to $395.

Cygnus Solutions (www.cygnus.com) provides support and custom engineering services for open-source technologies. The GNU tools it develops are returned to the Net community via the GPL license while the Cygnus implementation of these tools is sold and supported by Cygnus directly. For example, microcontroller companies pay Cygnus for engineering services to make its tools work with their 32- and 64-bit microprocessors.

The company's sales have grown at a compound annual rate of over 65 percent since 1992 resulting in over $25 million in revenue for the last four quarters. It was co-founded in 1989 by Michael Tiemann and John Gilmore, an early Sun employee and a leader of the Free Software camp. In February 1997, it received more than $6 million from August Capital and Greylock Management. (Disclosure: Esther Dyson is an investor in and sits on the board of Cygnus.)

My own company, privately held O'Reilly & Associates (www.oreilly.com) publishes books on open-source topics such as Perl, Linux, Tcl/tk, Python, Sendmail, BIND and Samba. I became active in promoting open source and investing in open-source-related companies when I realized that over half the company's $40-plus million in annual revenues comes from the sale of open-source-related books. If there was this much money to be made selling books, I reasoned that there must be a software opportunity as well.

In addition to open-source book publishing, O'Reilly has also created a packaged Perl distribution called the Perl Resource Kit, which includes the software on CD-ROM plus a number of reference books. (This is much the same packaging strategy as Microsoft's NT Resource Kit, and sells for the same $149 price.) O'Reilly has also produced three Perl conferences to date (two in the US and one in Japan) and will be offering parallel conferences in August 1999 on several open-source technologies including Perl, Apache, Sendmail and Linux.

Like other companies involved in commercial activities based on open source, O'Reilly takes seriously its obligation to contribute back to the open-source community. In 1996, O'Reilly hired Perl creator Larry Wall to a full-time staff position. His mandate: to nourish the growth of Perl. O'Reilly organized the OnePerl effort to bring back together the Windows and UNIX versions of Perl, and also funded Wall's recent work to add Unicode and XML support to Perl. O'Reilly online affiliate Songline Studios also hosts the central Perl download site www.perl.com.

Other companies adding value to open source include Scriptics (www.scriptics.com), founded in 1997 by Tcl author John Ousterhout to sell development tools, technology extensions and commercial support services for Tcl while continuing to develop the open-source Tcl and Tk packages.

C2Net (www.c2.net), headed by Sameer Parekh, sells an SSL-encrypting, enhanced commercial version of the popular open-source Apache server.

BSDI (www.bsdi.com), which was established by Uunet founder Rick Adams and is headed by long-time UNIX guru Rob Kolstad, packages the BSD OS along with the Apache server and offers support services to its ISP customers.

Larry Augustin's VA Research sells Linux workstations providing hardware benchmarking, compatibility evaluations and system configurations.

By adding proprietary value to open source, these companies make money by selling tools, extensions, GUIs, hardware, support services, custom engineering services, conference programs and documentation. In effect, these companies fill in the gaps of the open-source model by providing products or services that will not otherwise emerge under the open-source development model. Also, some make hardware-specific versions of open-source standards for hardware vendors who use them to add value (or create minimal utility) for their platforms.

**Make your money on the side**

A newly emerging business model for open source is what we call the "make your money on the side" strategy. In this model, open source or value-add proprietary software is not the company's focus. Rather, these companies make money by being more accessible or helpful to customers already relying on open source. For example:

IBM announced in June 1998 that it will ship the Apache server with the IBM WebSphere Application Server. As part of its package, IBM will provide commercial, enterprise-level support for the Apache Server. Because of Apache's credibility as a market leader, IBM hopes to gain credibility (and additional sales) for its own added-value Web application servers. The addition of the IBM team is a big gain for the Apache Group, since IBM will be focussing its development resources on improving the NT port of Apache, which is weaker than the widely-used UNIX version.

Netscape released in April 1998 its Communicator source code. Netscape's strategy is to accelerate development and distribution of future high-quality versions of this product to business customers and individuals. By doing so, it seeds the commercial market for Netscape's enterprise solutions, consulting and Netcenter businesses, which will become the company's primary revenue sources.

Corel, Oracle and Informix have announced that they are porting (or have already ported) their products to Linux. Doing so should expand the number of potential customers for their products, and make them cheaper. Corel's new Netwinder product line uses Linux as the heart of a commodity-priced thin-client server product.

**Hidden Winners in the Open-Source Game**

While everyone focuses on the companies that have made an explicit commitment to open source, really big money has been made by companies that have simply used the world made possible by open source as a springboard for very different types of businesses. Here are a couple of the big winners:

UUNet. As noted earlier, Rick Adams was the developer of the open source B News package for Usenet News as well as the administrator of the world's largest Usenet hub. As the Usenet infrastructure groaned under the strain, Rick realized the need for commercial services. He went on to create the first ISP, and what later became a multi-billion dollar industry.

Cisco. While Cisco has been active in the IETF, no one would mistake it for an open-source company. Yet its explosive growth has been driven by the expansion of the Internet. Without the open protocols, innovative open-source applications, and low barriers to entry that created the Internet opportunity, Cisco would be a very much smaller company today.

Microsoft. Yes, this one may seem like a stretch. But consider: Internet integration is the compelling new feature in Windows 98. Without the Internet, Microsoft would have had to rely on such dubious innovations as Microsoft Bob to drive upgrade revenue.

As we explain in "Science of the New Renaissance" later in this issue (page 22), the relationship between open-source and commercial activity is more complex — and more fundamental to the progress of the industry — than most of us credit.

The success or failure of these business models will drive the growth of the open-source movement. For open source to grow beyond the developer community, it must become as acceptable for an IS manager to install open source as a Microsoft, Sun or HP solution. That will happen only when that IS manager sees reputable, dependable and reliable companies behind open source, and good tools.

These companies are still in their infancy. Some of them, like Red Hat Software, have been around for a number of years but are only now developing commercial, enterprise-oriented support capabilities. As venture firms and angel investors supply capital to these companies to build these capabilities, and as they discover how to scale them and make money along the way, the adoption of open source should increase dramatically.

Open-source companies may never achieve the same scale as Microsoft, but to succeed in the market all they need is for their revenues to exceed their costs.

**Opensourcing: Distributing Your Software Development**

Open source may be particularly important to corporate software developers, many of whose projects are designed not for sale, but simply to solve internal business problems. Sometimes the solutions are proprietary or of interest only to a single company, but often, a company just needs something that works. A large number of open-source projects start this way, and spread as the developers discover other people are trying to solve the same problems.

By increasing opportunities for collaboration among individuals in different organizations, the Internet makes a new kind of distributed software development possible. Developers can work together quite closely and share their work openly. Eric Raymond describes this as using “the entire world as [your] talent pool.”

The decision to extend development beyond a single organization may be the most significant aspect of the open-source movement. We call this new distributed development model “opensourcing,” which has connotations of outsourcing. Making opensourcing more explicit can be a big win for many companies.

Opensourcing is a way to make your small development team large. In its simplest form, opensourcing means finding other people who share the same problem as you and involve them in a common development effort. Opensourcing spreads development costs across multiple organizations and increases the opportunity for people with highly specialized expertise to contribute. The result is a community of contributors organized around a shared body of work.

How does an opensourcing effort get started? Raymond believes that building a community around code cannot happen without a “plausible promise,” some code which demonstrates, however poorly, the basic idea or direction others will build upon. “The best hacks start out as personal solutions to the author’s everyday problems, and spread because the problem turns out to be typical for a large class of users.”

Opensourcing makes sense because most companies need to develop software but do not want to be in the software business. Few have a proprietary stake in the software components of an application.

Software development is expensive, as is supporting existing software. Neither comes to an end until the software stops being useful. Opensourcing increases the number of stakeholders in a piece of software, which lowers development costs and increases the amount resources available. This increase in the number of developers and users can help eliminate bugs in software. Additionally, the process also fosters the growth of a community of people who can support the software.

## OPEN SOURCE AND THE WEB

### Building the next generation of infoware products

Everyone asks if Linux really has a chance to dethrone Windows. But that's the wrong question. Open source has already radically changed the rules of the computing game, with ripple effects that are tugging the center of gravity away from Microsoft.

To make this clear, let's start with a small story.

I was talking with some friends recently, friends who don't own a computer. They were thinking of getting one so they could use Amazon to buy books and CDs. Not to use "the Internet," not to use "the Web," but to use Amazon.

Now, that's the classic definition of a "killer application:" an application that makes someone go out and buy a new computer.

What's interesting is that the killer application is no longer a desktop productivity application or even a back-office enterprise software system, but an entirely new breed, something you might call an "information application," or perhaps even "infoware."

Information applications are used to computerize tasks that just couldn't be handled in the old computing model. A few years ago, if you wanted to search a database of a million books, you talked to a librarian, who knew the arcane search syntax of the available computerized search tools and might be able to find what you wanted. If you wanted to buy a book, you went to a bookstore, and looked through its relatively small selection. Now, tens of thousands of people find and buy books online from that multi-million record database every day.

The secret is that computers have come one step closer to the way that people communicate with each other. Web-based applications use plain English to build their interface: words and pictures, not specialized little controls that acquire meaning only as you learn the software.

Traditional software embeds small amounts of information in a lot of software; infoware embeds small amounts of software in a lot of information. The "actions" in an infoware product are generally fairly simple: make a choice, buy or sell, enter a small amount of data and get back a customized result. These actions are often accomplished by scripts attached to a hypertext link. These scripts may in turn access a full-fledged back-end software server, but the primary interaction with the user is managed largely via the Web page (which may well have been created by a writer, editor, or designer rather than by a programmer, with a relatively small amount of programming support).

Information interfaces are typically highly changeable. For example, Amazon's presentation of books is driven by sales rankings that are updated every hour. Customers can add comments and ratings on the fly, which then become a key part of the information-rich decision-support interface for purchasers. Behind the scenes, an army of administrators and programmers are continually rebuilding the product. Dynamic content isn't just auto-

matically generated, it is also often hand-tailored, typically using an array of quick and dirty scripting tools, most of them open source.

Information interfaces are not as efficient for tasks that you do over and over as pure software interfaces, but they are far better for tasks you do only rarely, or differently each time. In particular, they are good for interfaces in which you make choices based on information presented to you. Whether you're buying a book or CD at Amazon, or a stock at E*Trade, the actual purchase is a fairly trivial part of the interaction. It's the quality of the information provided to help you make a decision that forms the heart of the application you interact with.

**Perl at Yahoo! and Amazon.com**

When many people hear of Perl and the Web, they think first of common gateway interface (CGI) — external programs run by a web server to generate pages on the fly. But even though Perl is the language most commonly used to create CGI scripts, its importance to the next-generation infoware applications is far greater than that.

"We don't create content at Yahoo! We aggregate it," says Jeffrey Friedl, author of the book Mastering Regular Expressions and a full-time Perl programmer at Yahoo! "We have feeds from thousands of sources, each with their own format. We do massive amounts of 'feed processing' to clean this stuff up or to find out where to put it on Yahoo." For example, to link appropriate news stories to tickers at quotes.yahoo.com, Friedl needed to write a "name recognition" program to search for over 15,000 company names. Perl's ability to analyze text with powerful regular expressions made that possible.

Perl is also a central component in the system administration infrastructure used to keep the site live and current. Vast numbers of Perl scripts are continually crawling the Yahoo! servers and their links to external sites, and paging the staff whenever a URL doesn't return the expected result. The best-known of these crawlers is referred to as "the Grim Reaper." If an automated connection to a URL fails too many times, the page is delisted from the Yahoo! directory.

The Amazon.com authoring environment demonstrates Perl's unique power to tie together disparate computing tools; it's a "glue language" par excellence. A user creates a new document with a form that calls up a Perl program, which generates a partially completed SGML document, launches either Microsoft Word or GNU emacs (at the user's choice), but also integrates CVS (Concurrent Versioning System) and Amazon.com's homegrown SGML tools. The Amazon.com SGML classes are used to render different sections of the web site — for example, HTML with or without graphics — from the same source base. A Perl-based parser renders the SGML into HTML for approval before the author commits the changes.

The Web is transforming the whole computing paradigm. This was never clearer to me than back in 1994, before Microsoft had gotten the web religion, and I shared the stage (via satellite) with Microsoft VP Craig Mundie at an NTT event in Japan. Mundie was demonstrating the planned in-

terface for its “Tiger” server, that was supposed to enable video on demand. The interface emulated Windows, with cascading menus responding to a kind of virtual remote control channel clicker.

It was pretty obvious to those of us who were involved in the Web that the right interface for video on demand, when and if it comes, will be a Web-like information interface. It’s ironic that even then, Microsoft had the perfect interface for video-on-demand: its own CD-ROM-based movie encyclopedia, Cinemania. What better way to choose what movie to watch than to search by category, read a few reviews, watch a few film clips, and then, homework done, click on a hypertext link to start the movie? Cinemania has it all but the last step. It’s not until hypertext-based information products are connected to network servers that their real power becomes apparent. Suddenly, information is not an end in itself, but an interface that allows a user to control an application space far too complex for a traditional software application.

Information interfaces are particularly appropriate for decision-support applications. However, they also make sense for one-time tasks. In a sense, the use of “wizards” for software installation is an example of the same trend.

There are also information applications that use a simpler, more software-like interface for user interaction, but provide dynamic information output. My favorite example is something that would have been virtually unthinkable only a few years ago: getting maps and directions. For example, maps.yahoo.com lets you type in two addresses, and get back a map and a set of directions showing how to get to one from the other.

So what does all this have to do with open-source software?

The answer is rooted in the way paradigms typically shift in the computer industry. As a particular market segment matures, the existing players have an enormous vested interest in things continuing the way they are. This makes it difficult for them to embrace anything really new, and allows — almost requires — new players (“the barbarians,” to use Philippe Kahn’s phrase) to come in to create the new markets.

**As the rules change**

Indeed, the old players are often key enablers of their own demise, as they try to get into the new market but can’t fully embrace the new rules. Microsoft’s ascendancy over IBM as the ruling power of the computer industry is a classic example of how this happens. IBM gave away the market to Microsoft because it didn’t see that the shift of power was not only from the glass house to the desktop, but also from proprietary to commodity hardware and from hardware to software.

In the same way, despite its attempts to get into various information businesses, Microsoft still doesn’t seem to realize that software, as Microsoft has known it, is no longer the main driver of value creation in the computer business. (Even Microsoft’s own efforts in natural language are more “info” than “soft:” some rules of grammar, but lots of information about words and meanings.)

In the days of IBM's dominance, hardware was king, and the barriers to entry into the computer business were high. Most software was created by the hardware vendors, or by software vendors who were satellite to them.

The availability of the PC as a commodity platform (as well as the development of open systems platforms such as UNIX) changed the rules in a fundamental way. Suddenly, the barriers to entry were low, and entrepreneurs such as Mitchell Kapor of Lotus and Bill Gates took off.

If you look at the early history of the Web, you see a similar pattern. Microsoft's monopoly on desktop software had made the barriers to entry in the software business punishingly high. What's more, software applications had become increasingly complex, with Microsoft putting up deliberate barriers to entry against competitors. It was no longer possible for a single programmer in a garage (or a garret) to make an impact.

By contrast, the open-source paradigm lowers the barriers to entry. You can try a new product for free — and even beyond that, you can build your own custom version of it, also for free. Source code is available for massive independent peer review. If someone doesn't like a feature, they can add to it, subtract from it, or reimplement it. If they give their fix back to the community, it can be adopted widely very quickly.

Perhaps more importantly, because developers (at least initially) aren't trying to compete on the business end, but instead focusing simply on solving real problems, there is room for experimentation in a less punishing environment. As has often been said, open-source software "lets you scratch your own itch." (Cf. the archetypal HP engineer who designs for the next workbench.) Because of the distributed development paradigm, with new features being added by users, open-source programs "evolve" as much as they are designed.

Indeed, the evolutionary forces of the market are freer to operate as nature "intended" when unencumbered by marketing barriers or bundling deals, the equivalent of prosthetic devices that help the less-than-fit survive.

**Evolution breeds not a single winner, but diversity**

It is precisely the idiosyncratic nature of many of the open-source programs that is their greatest strength. Probably the clearest case of this is Perl, which has been referred to as "the duct tape of the Internet."

Larry Wall originally created Perl to automate some repetitive system administration tasks he was faced with. After releasing the software to the Net, he found more and more applications, and the language grew, often in unexpected directions.

Perl has been described as a "kitchen-sink language" because its features seem almost random to the designers of more "orthogonal" computer languages. Wall addressed this point head on in his keynote at O'Reilly's recent Perl Conference 2.0. He showed a picture that looked like a small child's scribble and then explained:

"This is a picture of many things. It's a picture of air molecules bounc-

ing around. It's a picture of the economy. It's a picture of all the relationships of the people in this room. It's a picture of what the typical human language looks like. It's a picture of your company's information systems. It's a picture of the World Wide Web. It's a picture of chaos, and of complexity....

"It's certainly a picture of how Perl is organized, since Perl is modeled on human languages. And the reason human languages are complex is because they have to deal with reality....

"This is important, and a little hard to understand. English is useful because it's a mess. Since English is a mess, it maps well onto the problem space, which is also a mess... Similarly, Perl was designed to be a mess (though in the nicest of possible ways)."

The open-source development paradigm is an incredibly efficient way of getting developers to work on features that matter. New software is developed in a tight feedback loop with customer demand, without distortions caused by marketing clout or top-down purchasing decisions. Bottom-up software development is ideal for solving bottom-up problems.

Using the open-source software at the heart of the Web, and its simpler development paradigm, entrepreneurs such as Jerry Yang and David Filo were able to do just that. It's no accident that Yahoo!, the world's largest and most successful Website, is built around freely available open-source software: the FreeBSD operating system, Apache, and Perl.

**Change the (rules of the) game**

Just as it was last time around, the key to the next stage of the computer industry is the commoditization of the previous stage. As Bob Young of Red Hat, the leading Linux distributor, has noted, his goal is not to dethrone Microsoft at the top of the operating systems heap, but rather, to shrink the dollar value of the operating systems market.

Open-source software is not trying to beat Microsoft at its own game. Instead it is changing the nature of the game.

To be sure, for all their astronomical market capitalization, information-application providers such as Amazon and Yahoo! are still tiny compared to Microsoft. But the writing is clear on the screen. The edges of human-computer interaction, the opportunities for computerizing tasks that haven't been computerized before, are in infoware, not in software.

As the new "killer applications" emerge, the role of software will increasingly be as an enabler for infoware. Note that in the shift from a hardware-centric to a software-centric computer industry, hardware didn't go away. IBM still flourishes as a company (though most of its peers have down-sized or capsized). But other hardware players emerged who were suited to the new rules: Dell, Compaq, and especially Intel.

Intel realized that the real opportunity for them was not in winning the computer systems wars, but in being an arms supplier to the combatants.

The real challenge for open-source software is not whether it will replace

Microsoft in dominating the desktop, but rather, whether it can craft a business model that will help it to become the “Intel Inside” of the next generation of computer applications.

Otherwise, the open-source pioneers will be shouldered aside just as Digital Research was in the PC operating system business by someone who understands precisely where the current opportunity lies.

However that turns out, open-source software has already created a fork in the road. Just as the early microcomputer pioneers (in both hardware and software) set the stage for today’s industry, open-source software has set the stage for the drama that is just now unfolding, and that will lead to a radical reshaping of the computer industry landscape over the next five to ten years.

## THE SCIENCE OF THE NEW RENAISSANCE

Linux creator Linus Torvalds reports that the name “Linus” was chosen for him because of his parents’ admiration for two-time Nobel laureate Linus Pauling. Pauling’s second Nobel prize was for the underlying work that led to Crick and Watson’s discovery of the structure of DNA.

The choice of Linus’ name is apposite. Science, after all, is ultimately an open-source enterprise. The scientific method rests on a process of discovery, and a process of justification. For scientific results to be justified, they must be replicable. Replication is not possible unless the source is shared: the hypothesis, the test conditions, and the results. The process of discovery can follow many paths, and at times scientific discoveries do occur in isolation. But ultimately the process of discovery must be served by sharing information: enabling other scientists to go forward where one cannot; pollinating the ideas of others so that something new may grow that otherwise would not have been born.

Where scientists talk of replication, open-source programmers talk of debugging. Where scientists talk of discovering, open-source programmers talk of creating. Ultimately, though the open-source movement is an extension of the scientific method, because at the heart of the computer industry lies computer science. Consider the words of Grace Hopper, inventor of the compiler:

“To me programming is more than an important practical art. It is also a gigantic undertaking in the foundations of knowledge.”

Though they may not have realized it at the time, Pauling, Crick and Watson stood at the threshold of a new era in biological science. At the time of the discovery of the double helix, science in biology and chemistry was primarily a craft, a practical art. It was practiced by a few men working in small groups, primarily under the auspices of academic research. The seeds of change had already been planted, however. With several medical breakthroughs, notably the polio vaccine and the discovery of penicillin, biological science was about to become an industry.

Today organic chemistry, molecular biology, and basic medical research are not practiced as a craft by a small body of practitioners, but pursued as

an industry. While research continues in academia, the vast majority of researchers, and the vast majority of research dollars, belong to the pharmaceutical industry. This alliance between science and industry is an uneasy one at best. While pharmaceutical companies can fund research at a level undreamed of in academic institutions, they also fund research with a vested interest. Consider: would a pharmaceutical company rather fund therapy-based or medication-based research?

**Open source as scientific method**

Computer science, too, must exist in an uneasy alliance with industry. Where once new work came primarily from academic computer scientists, now the computer industry drives new work forward. While the rank and file of open-source programmers are still the many computer science undergrads and graduate students around the world, more and more open-source programmers are working in industry rather than academic settings, and more and more companies are basing their business around the open-source model.

Computer science, though, differs fundamentally from all other sciences. Computer science has only one means of enabling peers to replicate results: share the source code. To demonstrate the validity of a program to someone, you must provide them the means to compile and run the program. The open-source development model, then, really is an extension of the scientific method.

Replication makes scientific results robust. One scientist cannot expect to account for all possible test conditions, nor necessarily have the test environment to fully test every aspect of a hypothesis. By sharing hypotheses and results with a community of peers, the scientist enables many eyes to see what one pair of eyes might miss. In the open-source development model, this same principle is expressed as “Given enough eyes, all bugs are shallow.” By sharing source code, open-source developers make software more robust. Programs get used and tested in a wider variety of contexts than one programmer could generate, and bugs get uncovered that otherwise would not be found.

The open sharing of scientific results also facilitates discovery. The scientific method minimizes duplication of effort because peers will know when they are working on similar projects. Progress does not stop simply because one scientist stops working on a project. If the results are worthy, other scientists will follow up. Similarly in the open-source development model, sharing source code facilitates creativity. Programmers working on complementary projects can leverage the results of the other or combine resources into a single project. One project may provide the inspiration for another project. And worthy projects need not be orphaned when a programmer moves on. With the source code available, others can step in and take over. The open-source community boasts several programs that have migrated project managers in this way, including BIND, Fetchmail and the GIMP (GNU Image Manipulation Program).

The open-source development model of today has its roots in the academic computer science of a decade or more ago. What makes open source dramatically more successful today, however, is rapid dissemination of information through the Internet. When Watson and Crick discovered the double

### The IETF as Open Source

One of the best exemplars of the open-source movement is that wonder of the technological age, the Internet Engineering Task Force, or IETF. Nowhere are the principles of Eric Raymond's "bazaar" more evident than in the history of the IETF.

The IETF defines the open standards that make the Internet possible. Much of the work is done via mailing lists, with face-to-face working group meetings held three times a year. Anyone can join the mailing lists or come to the meetings. Anyone who comes has a voice (but if you haven't done your homework, or have a partisan agenda, you're likely to be ignored or shouted down). Scott Bradner of Harvard, a long-time participant, notes: "There is no voting, just a attempt to understand the level of consensus. This can be by show of hands or humming or any other way the chair wants — but the agreement does have to be general. 51 to 49 will not do — it has to be 80/20 or 90/10, but it does not have to be unanimous."

There is a hierarchy of Area Directors and Working Group chairs, but in general, the IETF works on the model articulated by Dave Clark: "No kings, no priests, just a rough consensus and running code."

This last point is critical. The IETF is a bottom-up standards body. Sometimes, an Area Director identifies problems, and asks people to volunteer solutions. More often, Working Groups are simply formed by interested people getting together and asking to be recognized. Competing solutions are evaluated on their technical merits. The focus is on simplicity and interoperability.

To be sure, there is a lot of politics within the IETF, and it has taken great skill for the chairmen of the various working groups to make the process work. Moreover, as with any successful open-source project, there is a fundamental vision, mostly articulated by the Internet Architecture Board, which guides the effort. But these chairmen and architectural visionaries aren't appointed by some central authority; they emerge (and are subsequently chosen by their peers) through the rough-and-tumble search for optimal solutions.

It's no accident that Microsoft's already infamous "Halloween Document" (an internal Microsoft memo leaked to Eric Raymond that analyzed responses to the open-source movement, referred to in the memo as OSS) identified the IETF as much as Linux as the force to be neutralized:

"There is a large amount of IQ being expended in various IETF working groups which are quickly creating the architectural model for integration for these OSS projects...."

"OSS projects have been able to gain a foothold in many server applications because of the wide utility of highly commoditized, simple protocols. By extending these protocols and developing new protocols, we can deny OSS projects entry into the market."

helix, they could reasonably expect the information to travel from Cambridge to Cal Tech in a matter of days, weeks at most. Today the transmission of such information is effectively instantaneous. Open source has been born into a digital renaissance made possible by the Internet, just as modern science was made possible during the Renaissance by the invention of the printing press.

The Middle Ages lacked an affordable information infrastructure. Written works had to be copied by hand at great expense, and hence the information had to have an immediate value attached to it. Trade records, banking transactions, diplomatic correspondence, this information was concise enough and carried enough immediate value to be transmitted. The speculative writings of alchemists, priests, and philosophers — the men who would later be called scientists — took a much lower priority, and hence the information was disseminated much more slowly. The printing press changed all this by dramatically lowering the barriers to entry in the information infrastructure. Scholars who had previously worked in isolation could, for the first time, establish a sense of community with other scholars all over Europe. But this exercise in community building required an absolute commitment to the open sharing of information.

Born out of this community was the notion of academic freedom, and ultimately the process we now call the Scientific Method. None of this would have been possible without the need to form community, and the sharing of information has held the scientific community together for centuries.

Imagine for a moment, if Newton had withheld his laws of motion, and instead gone into business as a defense contractor to artillerists following the 30 Years War. “No, I won’t tell you how I know about parabolic trajectories, but I’ll calibrate your guns for a fee.” The very idea, of course, sounds absurd. Not only did science not evolve this way, but it could not have evolved this way. Such secrecy would have kept science from developing and evolving at all.

The Internet is the printing press of the digital age. Once again, the barriers to entry for the information infrastructure have been dramatically lowered. No longer does source code need to be distributed on paper tape as with the original version of Unix, or floppy disks, as in the early days of DOS, or even on CD-ROM. Any FTP or Web server can now serve as a cheap and instantaneous distribution point.

While this renaissance holds great promise, we must not forget the centuries-old scientific heritage on which the open-source development model is based. Computer science and the computer industry exist in an uneasy alliance today. There is pressure from industry giants to keep new developments proprietary for the sake of short-term financial gain. But as more and more of the development work in computer science has its origins in industry rather than academia, industry must take care to nourish computer science through the open sharing of ideas, namely the open-source development model. The computer industry must do this not out of any altruistic motives or to serve a greater cause, but for the most basic pragmatic reasons: enlightened self interest.

First of all, monetary reward is rarely the primary concern of open source’s best programmers. These people are involved in a reputation

game, and history has shown that scientific success outlives financial success. We remember a few of the great industrialists of the last hundred years: Carnegie, Rockefeller, Ford. We remember a great many more of the scientists and inventors from the last hundred years: Einstein, Edison... Pauling. When the history of this time is written a hundred years from now, people will perhaps remember the name of Bill Gates, but few other computer industrialists. They are much more likely to remember names like Dave Clark, Bill Joy... and Larry Wall or Linus Torvalds.

Second, and more important, industry needs the innovation science can provide. Open source can develop and debug new software with the speed and creativity of science. The computer industry needs the next generation of ideas that will come from open-source development.

To sustain the digital renaissance we need open-source development. Open-source development drives progress not just in computer science, but in the computer industry as well.

---

**COMING SOON**

- Home-based local area networks.
- Wireless synchronization.
- Search and meaning.
- And much more... (If you know of any good examples of the categories listed above, please let us know.)

## RESOURCES & PHONE NUMBERS

*Because many of the players important in the open-source movement are not necessarily associated with individual companies, contact information is a bit more complicated than usual. Here are the places to go for more information about the technologies mentioned in this issue:*

The open-source movement in general: www.opensource.org (maintained by Eric Raymond) and opensource.oreilly.com (maintained by O'Reilly). People: Eric Raymond, (610) 296-5718; esr@snark.thyrsus.com, Tim O'Reilly, 707-829-0515, tim@oreilly.com.

The Halloween Document: www.opensource.org/halloween1.html (annotated by Eric Raymond).

Perl: www.perl.com and www.perl.org. People: Larry Wall, (650) 691-9063; larry@wall.org.

Apache: www.apache.org; for market share figures, see www.netcraft.co.uk; for the IBM connection, talk to James Barry, jmbarry@ibm.com.

Linux: www.linuxresources.com, a site maintained by The Linux Journal (see below) is a good starting point. People: Linus Torvalds, Transmeta Corporation, (408) 327-9830; torvalds@transmeta.com.

FreeBSD: www.freebsd.org; People: Jordan Hubbard, 925-682-7859; jkh@cdrom.com.

Samba: samba.anu.edu.au/samba (NOT www.samba.org); People: Andrew Tridgell, tridge@samba.edu.au.

The GNU Project and the Free Software Foundation: www.fsf.org; People: Richard Stallman, (617) 253-8830; rms@gnu.org.

Tcl: www.tclconsortium.org, www.scriptics.com; People: John Ousterhout, Scriptics, Inc., (650) 843-6902; ouster@scriptics.com.

Sendmail: www.sendmail.com; People: Eric Allman, eric@sendmail.com.

Bind: The Internet Software Consortium, www.isc.org; People: Paul Vixie, vixie@isc.org.

Python: www.python.org; People: Guido van Rossum, Corporation for National Research Initiatives, (703) 620-8990; guido@cnri.reston.va.us.

**Companies mentioned in the newsletter:**

ActiveState Tool Corp, www.activestate.com, 604-606-4606; CEO Dick Hardt, dick_hardt@activestate.com.

Cygnus Solutions, www.cygnus.com, (408) 542-9667, CEO Alex Daly; founders Michael Tiemann, tiemann@cygnus.com, John Gilmore, gnu@toad.com.

O'Reilly & Associates, Inc., www.oreilly.com, 707-829-0515; CEO Tim O'Reilly, tim@oreilly.com. For O'Reilly Research, Mark Jacobsen, markj@oreilly.com.

Scriptics, Inc., www.scriptics.com, (650) 843-6902; CEO John Ousterhout, ouster@scriptics.com.

Sendmail, Inc., www.sendmail.com, (510) 594-3150; CEO Greg Olson, greg@sendmail.com.

Suse, www.suse.com, (510) 835-7873; Scott Mcneil, President, mcneil@suse.com.

RedHat Software, www.redhat.com; (919) 547-0012; Chairman Bob Young, bob@redhat.com

Walnut Creek CD ROM, www.cdrom.com; CEO Bob Bruce, 650-800-786-9907; rab@cdrom.com.

**Books:**

Stig HackVän's forthcoming book on open-source licensing and associated business models from O'Reilly & Associates, appearing early in 1999.

Conferences:

Usenix LISA: December 6-12 at the Marriott Copley Place in Boston; see www.usenix.org/events/lisa98/ for more information.

Linux Expo: May 18 - 22, 1999, Raleigh Convention and Conference Center Complex, Raleigh, NC.

O'Reilly Open Source Conferences, August 21-24, 1999, Monterey Convention Center, Monterey, CA. (conferences.oreilly.com) Includes separate conferences on Perl, Apache, Linux, FreeBSD, Sendmail and business models.

# Release 1.0 Calendar

1998

| | |
|---|---|
| Nov 26-28 | **Doors of Perception 5: Play** - Amsterdam. The Netherlands Design Institute's conference on multimedia, the Internet, design and culture. To register call 31 (20) 420-1711; fax, 31 (20) 626-5845; registration@doorsofperception.com; www.doorsofperception.com. |
| Dec 7-10 | **CNET Builder.com Live** - New Orleans, LA. Web designers talk shop. Call (888) 599-2638; fax, (415) 956-3439; www.builder.com/live. |
| Dec 7-10 | **Java Business Expo and Comdex/Enterprise** - New York, NY. With Scott McNealy, Ed Zander and Jeff Papows. To register call (888) 528-2397; www.eventinfo.zdevents.com. |
| Dec 8-9 | **Jupiter Digital News Forum** - Atlanta, GA. Technologies and strategies for success in delivering news to the online consumer. For more information call (212) 780-6060; www.jup.com/events/forums/news/. |
| Dec 8-11 | **CAUSE98: The Networked Academy** - Seattle, WA. Information technology in higher education. Sponsored by Educause. To register call (303) 449-4430; fax, (303) 440-0461; e-mail conf@educause.edu; www.educause.edu. |

1999

| | |
|---|---|
| January 17-21 | **RSA '99** - San Jose, CA. 8th annual data security conference and exposition. To register call (888) 806-1545; fax, (513) 733-1302; www.rsa.com/conf99/home.html. |
| February 7-10 | **Demo 99** - Indian Wells, CA. Chris Shipley picks the hot startups. Call Alexa Hanes (650) 286-2730; e-mail alexa@demo.com; www.demo.com. |
| February 8-10 | **Wireless 1999** - New Orleans, LA. Sponsored by the Cellular Telecommunications Industry Association. To register call (415) 979-2289; fax, (415) 979-2250; www.ctiashow.com. |
| February 9-12 | **Milia '99** - Cannes, France. The international content market for interactive media. Contact Barney Bernhard, (212) 689-4220; fax, (212) 689-4348; e-mail infomilia-us@compuserve.com; www.milia.com. |
| February 17-20 | **TED9** - Monterey, CA. Richard Saul Wurman's annual multi-disciplinary gathering. To register call (401) 848-2299; www.ted.com. |
| March 21-24 | ****PC Forum** - Scottsdale, Arizona. Sponsored by EDventure Holdings. You read the newsletter; now meet the players. Call Daphne Kis, (212) 924-8800; fax, (212) 924-0240; daphne@edventure.com; www.edventure.com. |
| April 13-16 | **Spring Voice on the Net** - Las Vegas, NV. Internet telephony and related technologies. Produced by Pulver.com. For information call (516) 753-2640; fax, (516) 293-3996; pulver.com/von99. |

| | |
|---|---|
| June 22-24 | PC Expo - New York, NY. Over 100,000 corporate technology buyers in search of new toys. Sponsored by Miller Freemand. For information call (800) 829-3976; www.pcexpo.com. |
| July 14-17 | **Genetic and Evolutionary Computation Conference** - Orlando, FL. A joint meeting of the Eighth International Conference on Genetic Algorithms and the Fourth Annual Genetic Programming Conference. More information at www-illigal.ge.uiuc.edu/gecco/. |

** Events Esther plans to attend.*
*# Events Kevin plans to attend.*

*Lack of a symbol is no indication of lack of merit.*
*The full, current calendar is available on our Website (www.edventure.com). Please let us know about other events we should include. - Mari Katsunuma*

---

*Release 1.0 is published monthly except for a combined July/August issue by EDventure Holdings Inc., 104 Fifth Avenue, New York, NY 10011-6901; (212) 924-8800; fax (212) 924-0240; http://www.edventure.com. It covers software, the Internet, electronic commerce, convergence, online services, groupware, text management, connectivity, messaging, wireless communications, intellectual property law and other unpredictable topics. Editor: Esther Dyson (edyson@edventure.com); publisher: Daphne Kis (daphne@edventure.com); managing editor: Kevin Werbach (kevin@edventure.com); office manager: Helen Martin (helen@edventure.com); circulation manager: Scott Giering (scott@edventure.com); marketing manager: Mari Katsunuma (mari@edventure.com); assistant: Trista Schroeder (trista@edventure.com); receptionist: Philena Taylor (philena@edventure.com).*  *Subscriptions cost $695 per year, $750 overseas.*

**PC Forum 1999**
Back to Reality: Where's the Net ROI?

We are pleased to announce the twenty-second annual PC Forum, to be held this year in Scottsdale, Arizona from March 21 to 24.

Much of the fluff that drove the market for Internet stocks is gone, but some Internet companies are now making real money. Investors and managers, advertisers and merchants are making the distinction between great ideas and solid businesses. Leading multinational corporations are betting their futures on the Net.

This year we want to help ask the tough questions: Which business models work? Who is making them work? Where is pricing — of content, of eyeballs, or placement deals — heading? What is worth paying for? We will explore all these themes and more.

Confirmed speakers include:

**Charles Brewer**, president & ceo, MindSpring
**JoMei Chang**, ceo, Vitria Technology
**Barry Diller**, chairman & ceo, USA Networks
**John Doerr**, general partner, Kleiner Perkins Caufield & Byers
**Katharine Graham**, chairman, Washington Post Companies
**Christina Jones**, president, pcorder.com
**Bill Joy**, vice president, research & founder, Sun Microsystems
**Mitchell Kertzman**, president & ceo, Network Computer
**Tim Koogle**, president & ceo, Yahoo!
**Geraldine Laybourne**, co-chairman, Oxygen Media
**Shelly Lazarus**, chairman & ceo, Ogilvy & Mather
**Bob Lessin**, chairman & ceo, Wit Capital
**Peter Neupert**, ceo, Drugstore.com
**Martin Sorrell**, ceo, WPP Group
**Les Vadasz**, senior vice president, Intel

Company presentations and demos will include:

Blue Martini
Centraal
Lexeme
Neuromedia
Oblix

and several others to be announced!

**For more information, please call EDventure Holdings at (212) 924-8800.**

# RELEASE 1.0

## SUBSCRIPTION FORM

Please enter my subscription to Release 1.0 at the rate of $795 per year in the U.S. and Canada. Overseas subscriptions are $850, airmail postage included. Payment must be enclosed. Satisfaction guaranteed or your money back.

Name _____

Title _____

Company _____

Address _____

City _____ State _____ Zip _____

Country _____

Telephone _____ Fax _____

E-mail _____ URL _____

☐ Check enclosed

☐ Charge my

☐ American Express ☐ Master Card ☐ Visa

Card Number _____ Expiration Date _____

Name and Billing Address _____

Signature _____

☐ Please send me information on your multiple copy rate.

Please fill in the information above and send to:

EDventure Holdings Inc.
104 Fifth Avenue, 20th Floor
New York, NY 10011

If you have any questions, please contact us at 1 (212) 924-8800;
fax 1 (212) 924-0240; e-mail us@edventure.com; www.edventure.com.

Daphne Kis
Publisher

11-98