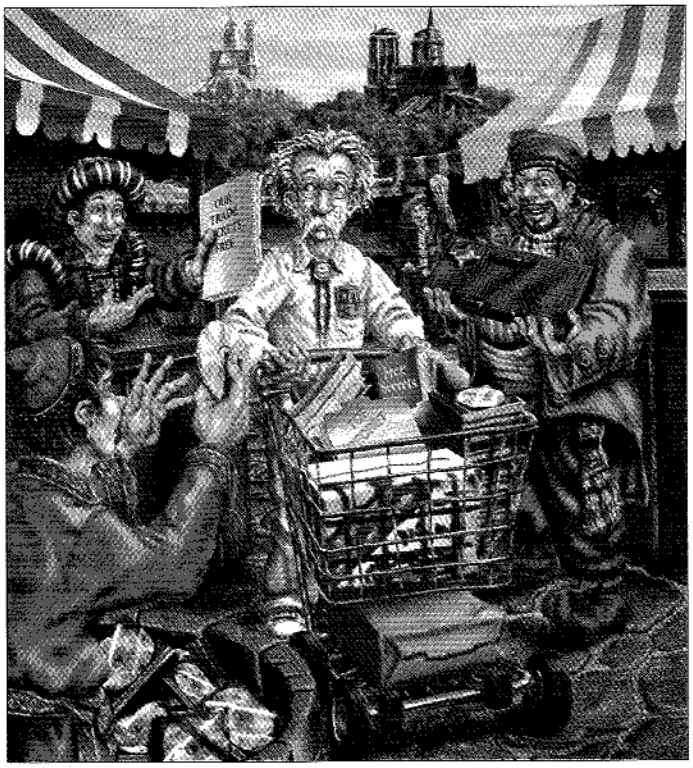
Ask Mr. Protocol

by Michael O'Brien



TOM BARRETT

"I believed that the most important software... needed to be built like cathedrals..." – Eric S. Raymond, *The Cathedral and the Bazaar*

"...the Linux community seemed to resemble a great babbling bazaar..." – Ibid.

"Given enough eyeballs, all bugs are shallow." – Ibid.

The Cathedral, the Bazaar and Mr. P.

Q: *Hey, didja hear? Netscape's giving away the store! Can you believe that, or what?*

A: I'd like to hear about the "or what" part before I commit, but as far as the facts go, it seems to be believable. I won't even begin to express this the way Mr. Protocol puts it, because it would be libelous if it weren't incomprehensible (as usual). I'll have to put my own spin on it (also as usual).

Netscape Communications Corp., having gone from 90% of the Web browser market to 60% in the usual hair-raisingly short space of time, has made a bold move. Rather than forging ahead with business as usual, and continuing to have its clock cleaned by Microsoft, it has decided to radically change the way it approaches the marketplace. It has not only decided to cease charging for its browser software and begin giving it away free to all comers, it has also decided to give away the source code to its browser suite.

Most news reports on this story have left me, at least, feeling confused. It's at times like this that I envy Mr. Protocol. He continues to believe that words on paper are so antique by the time he sees them that they couldn't possibly be of any use. Not only are the classics a closed book to him, books are a closed book to him. The sole exceptions to this are the various computer books that come out. He cackles over them the way you or I would find humor in a medical text of, say, 1887. In fact, that's a very good analogy: His mood takes on a sort of risible horror. I keep most books away from him, but I insist on my daily newspaper. As I say, he doesn't read it. He finds it useful only for lining the canary cage. This is a proceeding I can only view with wonder, because we here at Chez Protocol are blessed with neither canary nor cage. I have no idea how he makes the newspaper stay up there like that. After a while it goes away, I think by itself, because I never see Mr. Protocol touch it again. I avoid that end of the house most of the time. I'm only thankful that his worldview tends more to Magritte than to Bosch. In fact, he's got this picture–I think it's the only one he owns–which has a single vertical black bar on it, and underneath, in script, the words, "Ceci n'est pas une pipe." I don't want to think about it.

My confusion stems in part from the confusion of some of the reporters. These people find they have no idea why Netscape would do something that no company has ever done, at least, not since the early days of computing. Source code represents the keys to the kingdom for a software company. Few kingdoms make it a practice to throw open the doors of the treasury and yell, "Having a show! Come and get it! You want it, you haul it away!"

Others thought they had an idea, but on reading the stories, the idea seemed murky at best.

It turns out that Netscape's extraordinary action was inspired, at least in part, by a paper written by Eric S. Raymond, titled "The Cathedral and the Bazaar" (see `http://www.linuxresources.com/Er` ic/cathedral.html). This paper was given at several Linux conferences in 1997, and it attracted Netscape's attention because it illustrated a demonstrably successful software development model that Microsoft seems unlikely ever to embrace.

Mr. Raymond took notice some years ago of a trend in certain software efforts that has been remarked upon by Mr. Protocol in past columns. Project GNU is the most visible of these. Linux is the most famous such effort in the operating system world, but the model is strong enough to support more than one fish in the pond: FreeBSD, NetBSD and OpenBSD each fill a niche. FreeBSD, for example, is most robust at supporting high-end servers on Pentium hardware, as its TCP/IP stack is derivative of the extremely mature stack developed under DARPA's aegis at Berkeley, and it has been extensively optimized for Pentium hardware. Mr. Protocol prefers it for his own desktop network system, and it makes a good liniment and furniture polish as well.

All of these efforts share a common motto: "Many hands make light work, and many eyeballs make quick work of bugs."

The title of Mr. Raymond's paper comes from an analogy of two diametrically opposed models of software generation. The first, typified by the cathedral, represents the work of a single master designer or design team, working in solitude, with new software releases every six months to a year. No software is released before its time, and it is crafted according to the design of, and under the centralized direction of, a resident genius. Source code is highly proprietary, and competing products must be created in a "clean room" atmosphere where everyone working on the competition can certify that they have never been "infected" by sight of the source code of the first product.

In the bazaar, everybody works on everything, all the time, amid great din. The source code is released to the community. Work is coordinated by a core group, but anybody can do anything to the source code at any time. The central group's only "power" is the issuance of new releases bearing the group's imprimatur...which, for the classicists, amounts to a nihil obstat.

The most noticeable difference, to the end user, between the cathedral and the bazaar is that the bazaar can result in several new interim releases per day. Unless you're among the crowd that's in the thick of things, this can get very confusing very fast. As an example, in the FreeBSD world, it is not uncommon to set up a FreeBSD machine connected to the Net in such a way that it keeps its own source code mounted at all times, and synchronizes itself with the master FreeBSD software repository on a daily basis. Such machines often take up their night hours by performing a "make world," which recompiles and reinstalls all binaries for user commands and the operating system. Such systems literally reinvent themselves completely on a daily basis. If the "make world" fails, you've found a problem. Good! Lots of other people will have spotted the same problem and can contribute as necessary to a solution.

Eric Raymond points out that the main difference between the centralized, corporate software development model, which dominates the industry (exemplified primarily by Microsoft but also by Apple, Sun and Lotus) and the decentralized model followed by GNU, the free OS consortia, and various pieces of freeware (such as Raymond's own fetchmail) is one of involvement. The cathedral model, representing careful craftsmanship by a small, elite team, rarely, if ever, releases software to the general community in advance of the final release. Beta testers are carefully chosen, bound to nondisclosure, and even then only see software that its creators feel is already almost in its final form. No major changes and few minor changes are expected. Beta testing is only for bug-finding, in this model.

In the bazaar model, software spends its entire life in beta test. New versions are issued frequently, and in the case of a particularly hot development cycle, sometimes several times per day. A release is marked "stable" not just because it is stable, but to provide a point at which people who desire or require stability can cease tracking the blizzard of new releases. The user community is not only looked to for suggestions and bug reports, but for bug fixes as well as enhancements to the code itself. The advantage of the bazaar model is that it allows one person, or group of persons, to develop, maintain and extend the software code base using a development team far larger than could be afforded even by a large corporation.

So if you give the code away, how the heck do you make any money?

Mr. Protocol is glad you asked.

Show Me the Money

The bazaar model of software development only works if other people actually want the software, and want it a lot. Even if it doesn't work right to begin with, it has to show promise. But once it takes off, the entire development team is also a team of advertisers, press agents and word-of-mouth promoters. The product becomes not just a marketing item, but a cultural item. This is a status that most products only dream of attaining. If you pull this off, then you own the road. You may, in fact, hold title to the entire road throughout the area covered by the product, because the "generosity" makes any competing products created by the cathedral method look like pikers. The generosity involved, of course, is the exact same generosity shown by Tom Sawyer in passing out whitewash and paintbrushes next to his own fence. All these people are working for you for free. All you pay for is coordinating their efforts, and if you're politic enough, you can even throw out the "corrections" and "bug fixes" submitted by truly clueless wieners without alienating anybody. ("We're taking a coordinated and integrated approach to development in that area, which will be moving a little more slowly toward a more general solution. Thank you for playing, and come back again soon!")

Well, this sure works gangbusters at getting the product out there, we've seen it with Linux, emacs, vi, all sorts of mail systems and goodness knows what else. There remains that question about showing me the money?

Mr. Protocol says awright, already!

In Netscape's case, the product does not exist in isolation. Web browsers need Web servers, browser plug-ins, security packages, cache management software, everything but "bit detergent." Actually, you can sell that too, if the fraud statutes are lax in your locality. In fact, you can even deliver, if it's the tax statutes that are lax. But we digress.

Netscape expects to make its money at the big end, not the consumer end. It will act as coordinator and proprietor of its browser software. But the Web user community is now part and parcel of the development team–not even Microsoft can hire that many programmers–and their motivation is a lot higher, because they're working on something they themselves use.

Raymond points out that the cacophony and code fragmentation that would seem to be an obvious result of this software development model just don't arise. There are various reasons for this, which he goes into, but the point can be taken from the success of Linux et al that it does work…and lots of people are making money off of Linux, and FreeBSD too, by selling value-added products and services to a huge installed base.

Microsoft is giving away Internet Explorer, too…but not the source code. This means that Netscape's browser has the possibility of leapfrogging Explorer in capability and features, while having its own probably grotty code base cleaned up (consider how fast it was developed, after all, and by how few people), all at no cost to Netscape except for the cost of coordinating contributions and making up new (and frequent!) releases.

It's a model that could work. Where else might it be applied?

Newton's Lore

Mr. Protocol could suggest one, if he weren't busy spitting nails whenever it's mentioned.

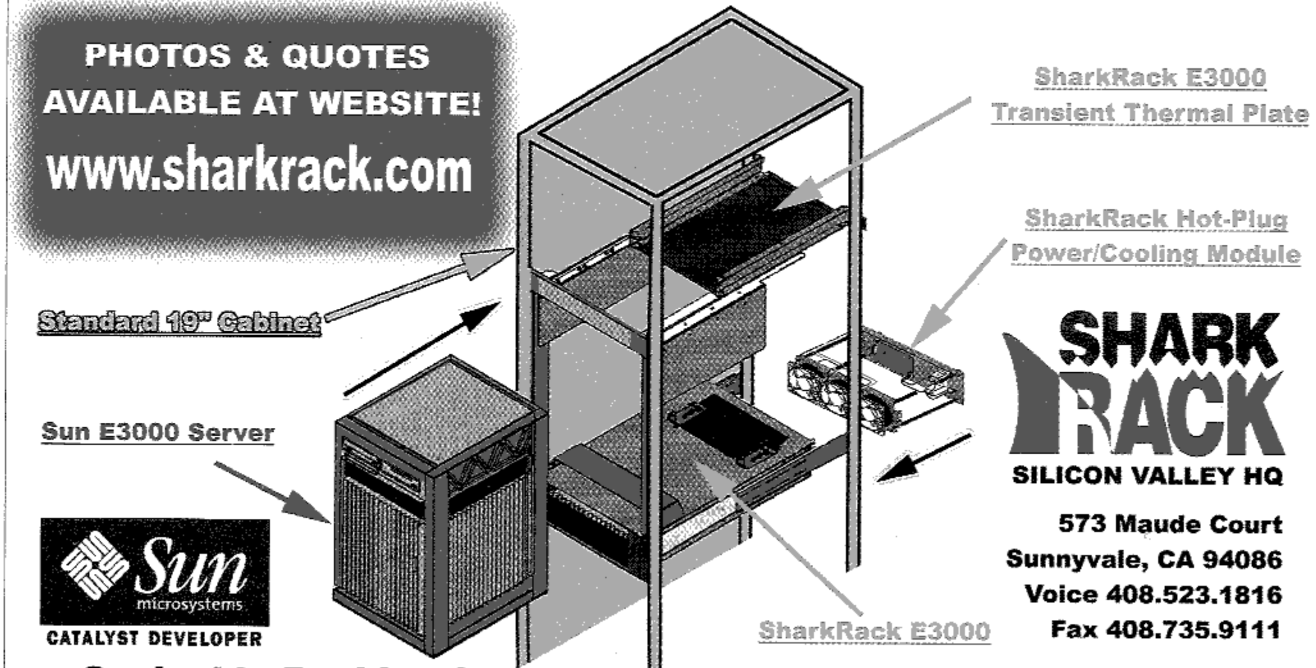
He has mentioned in times past that the Apple Newton MessagePad is a pretty neat piece of work. It's current incarnation, the MessagePad 2100, is nothing short of fabulous. It's as fast as a notebook, it fits in your hand, you can write on it in your own handwriting and it will get just about all of it right. It can browse the Net, be a personal organizer, send and receive email and has a huge base of available software. For just about every conceivable vertical and horizontal market niche, there are several competing software packages available. The "wow" factor is high, and although Apple's marketing has been lackluster, not to say nonexistent, one-on-one demonstrations by users have sold a lot of them. Overall sales of the 2100 and the eMate 300, which sports a built-in keyboard and looks like something built by space aliens, have been strong. It's easy to sell one: Just take it on any long-haul flight, use it for two to three hours, and then when every other notebook on the plane has given up the battery ghost, you can demonstrate your Newton to the suddenly computer-devoid notebook owners for about the next 20 hours.

On February 27, Apple announced that all development of the Newton OS was being halted. Existing stock of MessagePad 2100s and eMate 300s will be sold off, and no more will be manufactured. The stated reason was that Apple was forced to concentrate all of its efforts on a single OS, namely Mac OS. The actual reason, of course, was that after five years of producing some highly bogus products, Apple had finally gotten it right, which made it too boring to continue.

It is always easy to second-guess corporate planners, especially those working for a company in trouble, but this target is too tempting to resist. Apple may opt to get out of the handheld market altogether, and leave it to the 3Com Corp. PalmPilot, which is a beautiful piece of work but far less capable than any current Newton, and the Windows CE line of machines, whose sole advantage is the lack of an initial learning curve.

If Apple remains in the handheld market, it appears that the Newton technology is being eliminated in favor of a handheld tie-in to the upcoming NetPC product line of lightweight "network" computers. It is felt that memory and processor technology have reached the point where a stripped-down version of the Mac OS, or Rhapsody, or some sort of colloidal suspension of both, can be run on a handheld device. The result would have a Mac-like interface.

Now, there are noble antecedents here. The Mac interface was stolen wholesale from the Alto user interface, which in turn was developed as the



Ask Mr. Protocol

Smalltalk user interface (well, more or less; this analysis is on the order of "and then all the dinosaurs died and made oil," but bear with me), and Smalltalk was initially conceived of by Alan Kay as being the software basis for a device known as a Dynabook, which a Newton resembles more than a little. Closure is achieved by the fact that NewtonScript borrows most of its ideas from Unger's language Self, which is in turn a Smalltalk derivative. However, the GUI part of the Mac interface has been whanged on a heckuva lot since the Smalltalk days, and the Windows interface was stolen from the Mac interface; at least as of Windows 95 it was. And most of that whanging, both on Windows and the Mac, has been to tune the GUI to a fare-thee-well as the user interface for a desktop computer.

And the only, the *only,* advantage that WinCE has, is that it can be picked up by a Windows user with little or no learning curve. Mind you, it's still a desktop interface. Uses a keyboard with keys like pastilles. But there are a lot of Windows users out there, and not all of them have been permanently turned off by the disaster that was WinCE 1.0. So although they're slower than Newtons, and harder to use than Newtons, they're selling–because they look just like a Windows desktop, so you can use them right off, and because Microsoft is no dummy when it comes to marketing. Mr. P. doesn't see many commercials (and doesn't understand them when he does, which is good news for the joint credit card), but I do, and I've never seen a single commercial by Apple that really explains what a Newton is and what it can do. Not one.

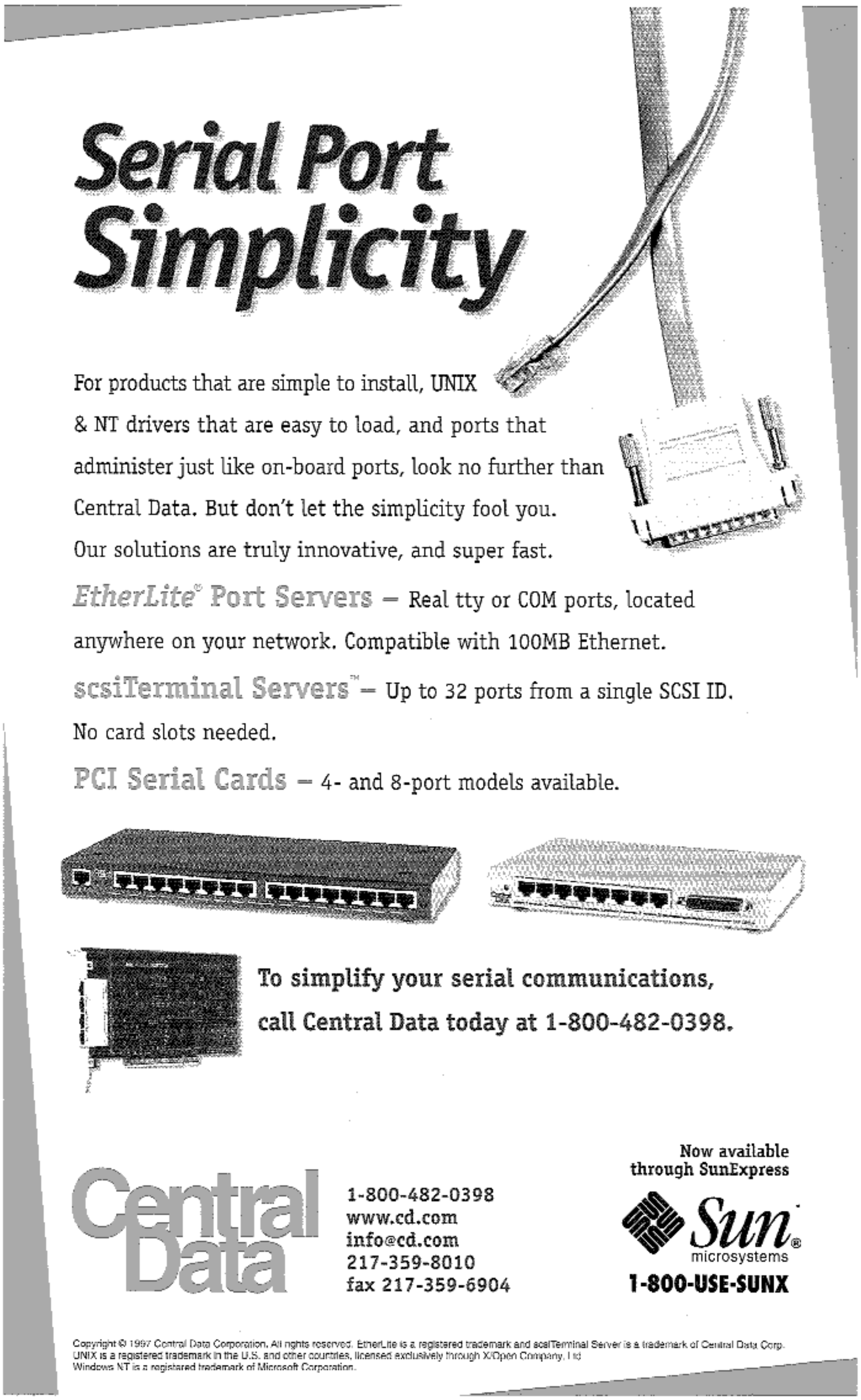
Now, with 10% or less of the desktop market, Apple is thinking of coming out with a handheld device that can be used with no learning curve at all...by that 10% of the market that already uses Macs. Gah. It must be something in the water. My aunt used to chalk it up to a green spray that the Russians were using.

So what does this have to do with cathedrals and bazaars?

Well, for one thing, Mr. P. has noticed that there is a great deal to be done with public space in between a cathedral and a bazaar. It is possible to combine the two methodologies. FreeBSD already does this, essentially, as it has a "core group," which is responsible not only for arbitrating what does and doesn't go into the "official" release, but also generates a good deal of it. Membership is open, on approval, to anyone who demonstrates those qualities that make the IETF work: rough consensus and working code. This "oligarchy of the masses" takes lots of contributions from outsiders and vets them, and checks new stuff into the release tree on a more-than-daily basis, but is itself responsible for most of the deep code in the system. Anyone not a member of the core group who persists in generating large amounts of useful code, or small amounts of critical code, is usually invited to join.

So the question is, can the bazaar method of development be applied to hardware? That is, if we have a situation where the most capable example of a given technology is assassinated by its proprietors, and if it looks like years could go by before anyone sane develops a replacement not crippled by inheriting an inappropriate design philosophy, can a popular movement step in to fill the gap?

The first answer is, "not a chance." Well, certainly the existing system cannot be taken over. In the case of the Newton, Steve Jobs reportedly turned down an outside offer for the Newton technology before deciding to kill it. Apple owns the road and will vigorously defend its cathedral against the infidel. So the new device would have to be a ground-up new hardware and software design.



Or would it? Actually, if it were possible to hijack an existing platform and convert it by means of new software into the desired result, there could be a chance. Prospective users go out and buy a WinWhizzo PalmPrint, then load up the new, free system from the Net. The only problem is it would probably require ROM replacements, and that's tricky enough that creating and installing new ROMs could sink the project right then and there. Not to mention the fact that the Newton hardware technology is far in advance of anything else that's even in sight on the horizon so far.

Don't count the bazaar out yet. Anyone with half a brain would have known that BSD UNIX was dead when DARPA cut off the money pipeline, and Solaris replaced SunOS. Doesn't look too dead so far. It survived by glomming onto a likely looking hardware platform and replacing all the software from the ground up. It took years to pull it off, though. And there was a much larger and more interested base of BSD UNIX users than there has ever been for the Newton.

Mr. Protocol, unfortunately, thinks it might be time to buy a MessagePad 2100 or two just to have on the shelf to get through the lean times. Some people are going to do some silly things, and it will be difficult to work around them. ⊷

__Mike O'Brien__ has been noodling around the UNIX world for far too long a time. He knows he started out with UNIX Research Version 5 (not System V, he hastens to point out), but forgets the year. He thinks it was around 1975 or so.

He founded and ran the first nationwide UNIX Users Group Software Distribution Center. He worked at Rand during the glory days of the Rand editor and the MH mail system, helped build CSNET (first at Rand and later at BBN Labs Inc.) and is now working at an aerospace research corporation.

__Mr. Protocol__ refuses to divulge his qualifications and may, in fact, have none whatsoever. His email address is amp@cpg.com.