free software report

Volume 1, Number 1

Free Software? Yes, Free Software

What is free software?

Free software is a strategy for the deployment of software that involves giving away source code and granting the right to freely redistribute and change it. While this strategy runs directly counter to traditional software business models, it has turned out to be remarkably successful in an increasing number of areas.

Nobel laureate John Maynard Keynes proposed the theory that free markets were more efficient than artificial markets. The balance sheet of what is left of the Soviet Union tends to bear this out. The inherent economics of software are that software is expensive to produce and maintain, but cheap to duplicate. This is not well represented by traditional software offerings, where development costs are rarely cited, and duplication is severely restricted.

By charging for production and maintenance and offering the freedom to duplicate, free software behaves like an ideal commodity in the free market. In contrast, proprietary software depends on a state-supported monopoly (copyright) to support prices, and thus behaves like a controlled commodity in an artificial market.

Why a free software report?

The purpose of the Free Software Report is to highlight free software's advantages to users, to uncover new opportunities for developers, and to relate the experiences of the marketplace in a way that helps readers determine the best way to take advantage of this new paradigm. This inaugural issue presents three different aspects of free software.

What proprietary software vendors call 'piracy', the free software world calls 'distribution'. This distribution is largely provided by customers, third parties and on-line computer networks. Among the several available "free software" models are freeware and shareware. Rich Morin, President of Prime Time Freeware, discusses their differences, benefits, drawbacks and their viable application in the current marketplace.

Many companies use free software in projects vital to their success. They welcome quality support for these projects, and know it's worth paying for. Wind River Systems, a real-time software solutions company, shares their success in using free software for profit. "...One of the first commercial software offerings in which free software was, and is, integral to the product's success."

"Software is a product...NOT!", written by Michael D. Shapiro, Ph.D., approaches the concept of software as a service. He proposes a change in the emphasis we place on building and maintaining software, to providing the service users need to do their jobs. This concept suggests that free software will be created by individuals and companies who have a use for it, thereby empowering organizations to create software and make their money in ways other than limiting its distribution.

We hope you find this premier issue of the Free Software Report useful, informative and enjoyable. Please let us know what you would like to see in future issues, what we are doing right and how we can improve the publication so you get the maximum benefit from free software. Submissions, criticisms, and comments can be sent to fsr@cygnus.com.

Winning With Free Software

For Wind River Systems, business and technical needs converge at a free software solution.

Wind River Systems™ (WRS) has been omnipresent in the real-time software systems market since 1981; its 1,000-company customer list reads like a “who’s who” of aerospace, telecommunications, process control and numerous other real-time sectors. Wind River Systems is led today by its two founders, Lawrence Berkeley Labs expatriates Jerry Fiddler and David Wilner. WRS employs 130 worldwide and has enjoyed a sales growth rate of 100 percent or greater since 1987.

As the real-time market evolved from an R&D outpost into a land of commercial opportunity, WRS shifted its focus from that of a consultancy to a marketing-oriented provider of real-time software solutions. VxWorks™, Wind River Systems’ real-time operating system and development environment, debuted in 1985 with a lengthy lists of “firsts”, including the first real-time operating system to support NFS™, and the first real-time operating system conceived as a cross-development environment. It was also one of the first commercial software offerings in which free software was, and is, integral to the product’s success.

“There were two choices: a commercial solution...or the GNU solution available over the net.”

The VxWorks package comprises the wind™ real-time OS kernel and a suite of development tools anchored by free software; the GNU C compiler and GNU debugger (a customized version called VxGDB™) are bundled with every system. A shell, linking loader, systems table and other programming assistants are also included.

“Yet another merit of working with free software: readily available source code.”

The impetus for choosing to bundle in GNU software was Sun Microsystems’ 1990 decision to discontinue its Sun 3-cross-68K tool set, leaving Wind River Systems and its customers “without a good, technical tool solution,” says Bob Wheaton, Wind River Systems’ Vice President of Marketing, Sales, and Services.

“We went out and looked at some alternatives to bundle in with our product. Wind River Systems wanted to offer our customers a C compiler and debugger,” says Wheaton. “There were two choices: a commercial solution, available from a couple of well-known compiler companies, or the GNU solution, available over the net.”

“Two discoveries were made. One was that, from a technical perspective, the GNU compiler seemed to be the best. It has a very good reputation in the marketplace. Our engineering department then took a good hard look at its technical merits compared to commercial compilers’, and came up with GNU as the leader,” Wheaton says. From a business perspective, GNU was not an immediate choice, at first: “The whole idea of free software is a little radical, to say the least.”. But GNU turned out to be an excellent business decision for Wind River Systems as well.

“The second thing we discovered was that commercial solutions would have turned out to be expensive not only for Wind River Systems, but for our customers, too. Wind River Systems maintenance contracts and royalty payments, along with the customer’s compiler license costs, would have probably doubled the price of our end product,” asserts Wheaton.

When WRS began implementing the free software tools for VxWorks, additional benefits became apparent. Catherine Jaeger, Wind River Systems’ marketing manager for VxWorks tools says, “Working with the GNU compiler made it easier for us to port VxWorks to new workstation platforms, because the GNU toolset could be easily ported.”

(continued on page 7)

Did You Know?
GNU tools were used to create VxWorks software from WRS. And, in the city of New York, over 12,000 traffic lights are operating under the control of Wind River Systems’ VxWorks.

Software is a product...NOT!

Michael D. Shapiro, Ph.D.

Much of our planning for software uses the hardware product model. While we're not perfect, we do a fairly good job of managing hardware engineering for required time, quality, and money. But software more often than not is late, doesn't do what we want to do, and costs too much. And that's a problem.

We can't see a solution to our problem — perhaps because we have made the solution invisible by thinking about it in the wrong terms.

I propose a vocabulary change. We will better understand the process if we consider software as a *service*, not a *product*. Let me expand on this statement.

I do not believe we must do any of the software-building activities differently. Instead, from the perspective of scheduling, budgeting, and delivering software, we should use the service paradigm rather than the product paradigm.

Hardware versus software.

We've all pondered the ways in which software differs from hardware:

◆ After initial development, most hardware cost is in manufacturing. Software has a small manufacturing cost.

◆ Hardware has physical parts that wear out, but can be replaced with identical or improved parts; the physical environment seldom changes. Software parts do not wear out, but often need replacement with greatly improved parts for changing environmental requirements.

◆ After initial development, hardware requires few engineering personnel for product maintenance. Software needs many highly skilled software engineers.

◆ Hardware is difficult and expensive to change in the field. Software can be upgraded easily and inexpensively in the field.

◆ Hardware engineering has matured into a craft. Software engineering still has a long way to go.

Hardware matches our notion of a product. But does software? Can we do better with a service paradigm?

Software as a service.

What is a service paradigm? Service is intangible. Software provides a delivery system for results. Users of our software do not care what program they have for a word processor, spreadsheet, or graphics editor. They want results: a proposal, a financial statement, or a presentation slide.

A service is successful when the customer wants to continue using it. Under my proposal, our emphasis would change from building and maintaining a product to providing the service users need to do their jobs.

We must plan and budget our efforts toward reaching this goal. And we should teach customers this same attitude for acquiring and using software.

Did Y ou Know?
Congruent Corporation ships GNU Utilities with Tool*Buster* , their Development Toolkit for Windows NT .

Upgrades as service.

When the software industry began, everything was free. Manufacturers gave away software to sell machines. Users freely shared software. Eventually, we realized we could make money from software by treating it as a product. Companies began charging for software, although they seldom sold it. Instead they leased or licensed it for a fee, often with a maintenance charge.

As a personal computers came along, software truly became a product. Now, we can go into a store and buy software off the shelf. We own it (no

(continued on page 4)

Software is a Product...Not!

(continued from page 3)

matter what the “license agreement” says) and can use it forever without ever paying any more.

But computer hardware capabilities keep increasing and we demand software improvements. Competitors appear, selling fancier versions of products. Companies enhance their products and sell “upgrades” through the mail.

With much fanfare, Microsoft started selling its 5.0 upgrade to MS-DOS in stores. Soon, most forms were selling upgrades — not only to their own products, but also as replacements for their own products, but also as replacements for their competitors’ products. Upgrades appeared on the best-selling software lists. Customers became accustomed to (but not necessarily happy with) paying for a product and, at intervals, its upgrades. The pricing scheme began to resemble that of a service like a book or record club.

Did Y ou Know?
Quaterdeck Office Systems, Inc. ships the GNU C/C++ compiler with the DESQview/X X1 1 and Motif Software Development Toolkit.

The next stage? Because companies complained about the need to continually purchase upgrades, Microsoft announced that it would begin an upgrade subscription service — much like a magazine subscription. So we’ve moved from software as a free service to software as a paid subscription service. In all these modes, we call it a product. But lurking in the background is that idea of software as a service, not a product. And if we plan and manage it as such, we might have a better hold on planning, scheduling, and performing its activities.

Avoiding maintenance.

An interesting thing happens when software is no longer a product. We no longer maintain it.

We’ve often said that software maintenance differs from hardware maintenance. Instead of keeping parts that wear out from failing, software maintenance now embodies fixing errors and enhancing software for new requirements or environments. But if software becomes a service, we emphasize adjustments for flaws as they appear and expansion to new and improved facilities.

What’s the difference? It’s a matter of attitude. With the attitude that software is a continuing service, we plan and budget for what was called maintenance as part of that service instead of tacking it on as a necessary, but often maligned, phase at the end of the product development process.

Will it work?

Can this change of vocabulary make a difference? *No*, if we merely apply a new word to old ideas. But *yes*, if we allow it to change our attitudes toward the way we think about, plan, and do software. ▼

Michael D. Shapiro
Computer Scientist
San Diego, California

UNIX Freeware is Different

Rich Morin
Prime Time Freeware

In the MS-DOS and Macintosh communities, freeware and shareware packages are typically distributed as executable binary images. This makes them easy to use, but impossible to repair, improve, or adapt. It also makes it very easy for miscreants to hide their work, as the code is essentially impossible to inspect.

“A great deal of support is available for most major freeware packages.”

The packages also tend to be aimed at end users, rather than systems administrators of or software developers. There are lots of games, desktop tools, and other end-user applications. Few languages or other programming utilities are available. This reflects the sharp division these communities have between “users” and “developers”.

The voluntary nature of shareware is also distorted by binary distribution. Binary packages can be “crippled”, forcing users to send money for the “full” versions. Timers can be put in, causing the packages to fail after a certain date. Source code distribution provides much less room for this sort of tactic.

In the UNIX community, the picture is considerably different. Packages tend to be tools for developing software, administering systems and/or networks, etc. Freeware (and some shareware) is distributed primarily as source code. Binary code, if present, is provided as a supplement to the source code.

Revision and adaptation are generally encouraged, and large discussion groups exist to carry debates on the merits of proposed changes. In the collegial atmosphere of the UNIX freeware community, the distinction between developers and users becomes very blurred. Developers use (and modify) each others' freeware as desired, then report back on interesting changes.

Bad Gnuse

UNIX freeware can be quite a challenge to install. If a package hasn't been tested on the target platform, it may fail to build or execute properly. Automated configuration software, if present, may be broken. GNU `grep-1.6` is convinced that SunOS 4.1.x has ANSI C header files. It doesn't, and some `Makefile` tweaking is required.

Alternatively, a package may uncover a vendor's bug. One of the tests for GNU `grep-1.6` fails under SunOS 4.1.1, due to a bug in Sun's `awk`. The bug is Sun's, but the test should probably avoid exercising it.

Did Y ou Know?
Motorola sells and supports the GNU C Compiler for their 88K, 56K, 96K and DSP-16 chip families.

"Because of the number of contributing freeware authors; long-standing bugs are very rare, and exceptional results are sometimes obtained"

Finally, freeware developers can become dependent on having an infrastructure of freeware around. You can't use `rftp` (recursive FTP) unless you already have `perl`. The GhostScript (GNU PostScript) suite is written in `C++`, and the author probably used `G++ (GNU C++)` to develop and test it. I haven't yet found a set of tools that has circular requirements, but it wouldn't surprise me all that much if I did.

Good Gnuse

Because of the number of contributing freeware authors, long-standing bugs are very rare, and exceptional results are sometimes obtained. The GNU `grep` utility, notwithstanding the installation nits mentioned above, is many times faster than the one distributed by most UNIX vendors.

Nonetheless, the README file points out areas where improvement is needed, and even suggests possible strategies. This is a healthy sign, indicating that the authors are not caught up in proprietary or ego-driven desires to keep the code as they wrote it. Try getting a commercial vendor to put its code out for inspection let alone to accept external code changes.

A great deal of support is available for most major freeware packages. Mailing lists, USENET news groups, and informal networks of friends and associates all help freeware users to get over their problems. If more formal support is needed, it is often available. There are consultants and companies with expertise in many freeware packages. The largest of these is Cygnus Support, the de facto R&D department for many of the GNU tools.

There is a cost to using UNIX-related freeware. The author isn't being paid to listen to your complaints, and you may have to make things work yourself. You certainly can't pound on the table and expect a fix. On the other hand, there is a great deal of freedom and knowledge to be gained from using and contributing to freeware.

Richard Morin produces Prime Time Freeware, a semi-annual CD-ROM collection of redistributable, UNIX-related source code and documentation.

Between releases, he consults, writes and teaches on UNIX topics. He may be reached at CFCL, P.O. Box 1488, Pacifica, CA 94044 or rdm@cfcl.com

The Free Software Community Puts A Free Compiler Back In Solaris 2

Sun Microsystems, Inc. decided to unbundle the C compiler from their latest operating system, Solaris 2. Sun users were extremely upset to lose what they saw as an essential component of the system software. Faced with dramatic increases in licensing fees, early Solaris 2 users turned to free software for a reasonable alternative.

Spearheading the effort to port the Free Software Foundation's GNU C compiler was Palo Alto based Cygnus Support, a company that specializes in providing commercial support for free software. To fund the development effort, Cygnus appealed to the early adopters of Solaris 2. They offered a year of technical support for up to 5 users, and a commitment that the compiler would ship with Solaris 2,in return for a prepaid fee of $2,000.

To insure wide distribution of thefree compiler and debugger, Cygnus negotiated with SunSoft, Inc. to make the GNU C development tools available on CDware. CDware is a free CD-ROM available from Sun and shipped at no cost to over 90,000 Sun users.

Over 50 organizations contributed to the initial effort, and the first GNU C development tools were released in the first week of July, 1992, one week before Sun shipped their own set of development tools.

The effort has not stopped there. The GNU C++ compiler and its libraries are being ported to Solaris 2. They will be available on November 1.

To receive more information on GNU C development tools, contact Cygnus Support. To receive a Catalyst CDware disc at no charge, call Catalyst CDware Subscriptions at 1-800-227-9227, ext 103.

We want to hear from you! In the next issue of The Free Software Report, we will dedicate an entire section to addressing any questions, concerns, or comments relating to free software. Several of these letters will published. Questions and concerns will be addressed by the highly qualified staff at Cygnus Support. Please send your inquiries to the editor:
fsr@cygnus.com
or send letters to:

Free Software Report
1937 Landings Drive
Mountain View, CA
94043

Did Y ou Know?
Network Computing Devices uses GCC exclusively for all of their software development.

More Reading On Free Software and Related Topics

"Programs To The People"
Technology Review
Feb/March 1991

"Free For All"
The Wall Street Journal
Technology Section
May 20, 1991

"Get Ready For GNU Software"
Computerworld
August 6, 1990

"Now What Do I Do?"
SunWorld
June 1992

"Gfree Gsoftware"
RS/Magazine
June 1992

"Free Software Distributions"
SunExpert Magazine
March 1992

"The Harm Of Patents"
The Economist
August 22, 1992

"A Look At Software Distribution"
SunExpert
November 1991

Winning With Free Software

(Continued from page 2)

"This also offers a degree of consistency to our customer base," she continues. "For those working with more than one workstation platform, the GNU software is a reliable toolset. This enables our customers to make the choice of which platforms they want to work with."

"So, from a product strategy standpoint, working with GNU has been very successful for us as well," Jaeger concludes.

Wheaton offers yet another merit of working with free software: readily available source code. "We have modified the GNU C compiler for the X Window™ client package available for VxWorks, called windX™. That job would have been a lot more difficult for us if we'd dealt with a commercial company, where the issue of ownership becomes muddled." WRS has also added remote debug capabilities to VxGDB, for cross development applications, and forwarded those contributions to the Free Software Foundation.

Wind River Systems looked to Cygnus Support of Palo Alto, CA, for assistance in fine-tuning the GNU software tools for all of VxWorks' supported environments. WRS chose Cygnus's Core Support development service as the means to seamlessly integrate the GNU tools into the various VxWorks versions.

Wheaton offers an enthusiastic summary of Wind River Systems' free software strategy: "We made a technical and business decision we're very happy with." ▼

Access Information

Congruent Corporation
110 Greene Street
New York, New York 10012
+1 212 431 5100 voice
+1 212 219 1532 fax

Cygnus Support
1937 Landings Drive
Mountain View, CA 94043
+1 415 903 1400 voice
+1 415 903 0122 fax
info@cygnus.com

Free Software Foundation
675 Massachusetts Avenue
Cambridge, MA 02139 USA
+1 617 876 3296
gnu@prep.ai.mit.edu

Free Software Report
1937 Landings Drive
Mountain View, CA 94043
+1 415 903 1400 voice
+1 415 903 0122 fax

IEEE
Institute of Electrical Electronics Engineering, Inc.
445 Hoes Lane
PO Box 1331
Piscataway, NJ 08855-1331
+1 908 981 0060 voice
+1 908 981 0027 fax

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
+1 206 882 8080

Motorola Computer Group
2900 South Diablo Way
Tempe, Arizona 85282
+1 602 438 3000 voice

Network Computing Devices, Inc.
PC-Xdivision
PO Box 4900
Beaverton, Oregon 97076-4900
1 800 800 9599 voice

Prime Time Freeware
CFCL
PO Box 1488
Pacifica, CA 94044
ptf@cfcl.com

Quarterdeck Office Systems, Inc.
150 Pico Boulevard
Santa Monica, CA 90405
1 800 354 3222

Sun Microsystems, Inc.
2550 Garcia Ave.
Mountain View, CA 94043
+1 415 960 1300

SunSoft, Inc.
2550 Garcia Ave.
Mountain View, CA 94043
+1 415 960 1300 voice
+1 415 969 9131 fax
1 800 227 9227 x103 CDware

Wind River Systems, Inc.
1010 Atlantic Ave.
Alameda, CA 94501
+1 510 748 4100 voice
+1 510 814 2010 fax

The Fine Print

The Free Software Report is published quarterly. A 4-issue subscription is available for $60.00 payable in U.S. Dollars through a U.S. bank. Send a check or money order to: **The Free Software Report 1937 Landings Drive, Mountain View, CA 94043, USA**

Please supply your name and title, company name, and address for receiving the Free Software Report. The most recent and subsequent issues will be mailed as soon as we receive your order.

Name____________________ Title____________________

Email____________________

Company or Organization____________________

Address____________________

City____________________ State__________ Postcode__________

Country____________________ Amount Included: $____________________

Comments ____________________

The Free Software Report
1937 Landings Drive
Mountain View, CA 94043

BULK RATE
U.S. POSTAGE
PAID
MOUNTAIN VIEW, CA
PERMIT NO. 521