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____________________

Company or Organization______________________________

Address____

City__________________ State__________ Zip __________

Country__________________ Amount Included: $____________

Comments ____

____

e-mail____

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

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

gdb: A Key T ool...

Continued from page 6

and g++ (GNU C/C++ compilers), gas (assembler), gld (linker), emacs (editor), and GNU implementations of Unix utilities. The sources to the latest public versions of all GNU tools are available at various ftp sites throughout the Internet.

If you are already a user of emacs, the GNU editor, or if you are interested in emacs, you will be especially pleased with the integration between emacs and gdb, which includes automatic location of the source lines of the function being debugged, and gdb-mode keybindings. X windows interfaces to gdb (including emacs), are also available.

I hope that this article has provoked an interest in seriously investigating gdb as one of your development tools. You can get the sources from the Internet easily enough. You're likely to be using a platform that gdb already supports, so building and installing gdb for your system will likely be a matter of two or three commands. Then you and your colleagues will be able to start using gdb for your everyday work. ✌

Jose A. Fernandez, is a senior computer scientist at Inference Corp., where he does product development on a variety of Unix and non-Unix platforms. He is highly dependant on GNU tools: emacs, GNU make, gcc, and gdb.

More Reading On Free Software and Related Topics

"Prime Time Freeware for UNIX"
Rich Morin, Prime Time Freeware

"What's GNU?"
SunWorld, July 1993

"Ada is worth another look, and it's now free"
InfoWorld, July 1993

"C/C++ Resource Guide"
Dec Professional, August 1993

"Cygnus Support—Tiemann's One and Only Big Win"
Software Entrepreneur's Forum Newsletter, August 1993

"Revolution and Opportunity"
Microtimes, June 28, 1993

"Are software vendors really ready for open systems?"
Personal Engineering & Instrumentation News, October 1993

"Minding your problems"
SunWorld, August 1993

The Fine Print

Access Information

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

Digital Equipment Corp.
MK02-2/A6 Digital Drive
Merrimac, NH 03054
603 894 2201 voice - Glen Thompson

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

Hewlett Packard
19091 Pruneridge Ave.
Cupertino, CA 95014
408 447 5334 voice - Kelly Emo

Hitachi America, Ltd.
Semiconductor & I.C. Division
Hitachi Plaza
2000 Sierra Point Parkway
Brisbane, CA 94005-1835
415 589 9000 voice

Intel Corporation
2200 Mission College Blvd.
PO Box 58119
Santa Clara, CA 95052-8119
408 765 8080 voice

Novell, Inc.
122 East 1700 South
Provo, Utah 84606-6194
801 429 5933 voice
801 429 3098 fax
800 NETWARE voice

Pegasus Mail
Contact David Harris via the Internet at
david@pmail.gen.nz

Sun Microsystems Computer Corporation
2550 Garcia Avenue
Mountain View, CA 94043
415 904 7000 x239 voice - Mark Richardson

We want to hear from you!
Please send your inquiries to the editor:
fsr@cygnus.com
or send letters to:

Free Software Report
1937 Landings Drive
Mountain View, CA 94043

gdb: A Key T ool...

Continued from page 5

chain, etc.) Here's a practical example. Suppose you want to stop at the function `malloc()`. You would type: break malloc

Breakpoints can be conditionalized: when a breakpoint is reached, execution will only stop if some specified exception is true. Breakpoints can be ignored for some number of executions. Breakpoints can have attached commands: when a breakpoint is reached, a list of gdb commands can be executed (such as printing a variable value and continuing from the breakpoint).

The `watch` command is used to set watchpoints in your program. You use a watchpoint to stop execution whenever the value of an expression changes, without having to predict a particular place where this may happen. Quoting from the on-line documentation: "Watchpoints currently execute two orders of magnitude more slowly than other breakpoints, but this can be well worth it to catch errors where you have no clue what part of your program is the culprit."

Here's a practical example. Suppose your program is inadvertently scribbling over an area of memory, like corrupting the head of a linked list. You could set a watchpoint using:

```
watch LinkedListRoot->Next->Previous != LinkedListRoot
```

If you've managed to isolate your program's problem, say, after a certain function is called, you could use a combination of breakpoints and watchpoints:

```
tbreak MeaningfulFunction commands watch MeaningfulMemoryExpression end
```

In the above, after `MeaningfulFunction` is called, gdb will start watching `MeaningfulMemoryExpression`.

gdb understands the C, C++, CHILL, and Modula-2 languages. It can deduce the language of the code being debugged (through the suffix of the source file). You can also manually set the language if necessary.

As you can see, gdb provides a considerable advantage in multiplatform development. Swift debugging is largely a matter of learning to use this debugger. Developing code on different platforms can be difficult; while the programming language may not change, the debugger probably will, and some debuggers are far less useful than others. gdb's wide availability gives its users an advantage when they move between different platforms; the way they use the debugger remains unchanged!

gdb is officially maintained by Cygnus Support. Cygnus periodically releases new versions of gdb containing bug fixes and new features to their customers and to the Internet community. gdb users who need special binary releases, quick response to any problems they have, special modifications, new hardware support, or new software environment support, can contract with Cygnus Support for highly expert assistance. Cygnus Support offers its services for all GNU tools, which include gcc

Continued on page 7

gdb provides the following special support for C++:

- When you want a breakpoint in a function whose name is overloaded, gdb breakpoint menus help you specify which function definition you want.
- Setting breakpoints using regular expressions is helpful for setting breakpoints on overloaded functions that are not members of any special classes.
- You can print inheritance relationships as well as other information for a given type.
- You can control whether C++ symbols display in their source form, both when displaying code as C++ source and when displaying disassemblies.
- You can choose whether to print derived (actual) or declared types of objects.
- You can control the format for printing virtual function tables.
- Specify a particular definition of an overloaded symbol using the same notation that is used to declare such symbols in C++ `-> symbol (types..)`. You can also use the gdb command-line word completion facilities to list the available choices, or to finish the type list for you.

PMail

Continued from page 4

continues to cover its costs and provide me with a basic living, I see no need to increase costs - and in fact, I actively try to keep costs down as much as I can.

In an ideal world I'd have some kind of long term research funding and wouldn't have to charge anything at all, but I suppose I have to consider life in the real world on at least an occasional basis... :-)

TT: What future do you see for PMail?

DH: That's a hard one... I really don't know. E-mail is changing so fast and so often that it's getting very hard to know where it will go next. I live PMail pretty much from day to day with no real expectations for it; as long as it's useful to people I'll continue to support it and help it to grow.

TT: Do you see the world of computing more accepting of the concept of free software as compared to the past?

DH: Yes and no. The area of greatest acceptance has clearly been the academic world, with the business world being much slower to catch on. There's still a quite clearly-defined stigma, associated with words like "shareware" and "freeware", which makes most MIS people cautious of using free software; the myth that "the only way to get quality and support is by paying for it" is still depressingly alive and well. On the other hand, there have been some definite ground-breakers in recent years so the tide is perhaps beginning to turn a little; the GNU Project, Packet drivers, NCSA Telnet, Vernon Buerg's LIST and a number of other free or very inexpensive programs (maybe even including PMail) have begun breaking the ice.

TT: What are your thoughts on the future of free software in a competitive, computing marketplace?

DH: The biggest impediment to the acceptance of high-quality free software has always been a reluctance on the part of MIS people to "risk" it. This attitude is changing, albeit slowly. More and more MIS sections are willing to consider free software as alternative solutions and as this trend spreads there is going to be a greater and greater need for high quality free software in the marketplace. For those people willing to invest the time and effort in providing a free service to the broader community of users, the opportunities are only going to get larger and larger.

Did You Know?

GNU tools for NetWare will enable NLM development capabilities on all 3 PIN platforms (SMCC SPARC, DEC Alpha, and HP PA-RISC) using a single toolchain.
A simple re-compile is all that is necessary to take full advantage of the features offered on the three PIN platforms.

`gdb`: A Key Tool...

Continued from page 2

this overview, I have hinted that `gdb` has many more features than those listed. Your interest will, I am sure, be piqued...

Here's a brief examination of `gdb`'s capabilities. The "`print`" command is used to examine data. To examine a variable, use:

```
print VariableName
```

To examine the value of a member of a struct, use:

```
print VariableName->MemberName
```

The `print` command can evaluate very complicated expressions.

Quoting from its on-line documentation, "`print` and many other `gdb` commands accept an expression and compute its value. Any kind of constant, variable or operator defined by the programming language you are using is valid in an expression in `gdb`. This includes conditional expressions, function calls, casts and string constants. It unfortunately does not include symbols defined by preprocessor '`#define`' commands."

Here's a practical example. Suppose you know address `0xBEEFF00D` is the start of a `malloc()`'d instance of `struct FOO`. Suppose you want to examine that area of memory as a `struct FOO`. You would type: print

```
*((struct FOO *) 0xBEEFF00D)
```

The `break` command is used to set breakpoints in your program. You use a breakpoint to stop execution whenever your program gets to a certain point in your program. Once your program is stopped, you can examine all of the program state (variables, registers, the call

Continued on page 6

PMail

Continued from page 3

electronic mail to everyone, and keeping PMail free is one way of encouraging that.

TT: How does PMail differ from its proprietary competition?

DH: To tell the truth, I don't really know. I go to great lengths not to look at commercial mail packages because I would much sooner be driven by the features my users want than by competing with other developers. From what I've read and been told, though, the biggest difference seems to be that PMail is based on real standards and is thus easier to integrate into heterogeneous environments.

TT: Why not make PMail proprietary; implement license managers, limit releases to one a year or whenever you see fit, limit duplication?

DH: That's rather like suggesting that a religious person give up his or her religion - it's not even worth contemplating. All the things you describe hit directly at the users - they inconvenience or cost the people who have the greatest need to benefit from the system and I won't have any part of that.

In part, I've always believed that e-mail is a basic utility - something as essential to a network as `dir` is to a DOS system. I've always found it rather shocking that the most important network product on the market provides no e-mail services, and PMail is partially an attempt to address that shortcoming. People shouldn't have to pay for e-mail. If they have a network then I believe they have a right to expect e-mail to be there.

Did You Know?

Advanced Micro Devices ships the complete GNU development toolkit from Cygnus Support with each Am29205 evaluation kit?

TT: How are improvements and upgrades made to PMail?

DH: I work continuously on PMail and periodically release new versions with varying levels of improvement; in general I'm trying to work towards a schedule of one or at most two major upgrades each year to each PMail platform, with an arbitrary number of sub-releases for maintenance. Because PMail is free, it is easy to distribute: it is widely available on the world-wide Internet (principally at the site `risc.ua.edu`, in `/pub/network/pegasus`), on CompuServe in the NOVUSER area, and on numerous bulletin boards.

TT: Why do you think PMail has seen such success to date? Is it because it is technically superior to its proprietary competition or because it is free?

DH: I wouldn't be so presumptuous as to say that PMail was technically superior to any other product - I prefer to leave that decision to the users. I think PMail is extremely well-tailored to the areas it seeks to serve - most specifically the Internet and NetWare MHS environments: this may account for much of its acceptance. In my experience, many sites try it because it's free then find they like it, so I guess you can say that its free status has had a great deal to do with its acceptance as well. Having said this, I would hate to think that sites use PMail ONLY because it's free; I like to think of PMail as a well integrated, technically competent package that people enjoy using too.

TT: You will be releasing version 3.0 of PMail very soon. It is rumored that you will charge the same amount of money you you are currently charging for the existing version. Why is that?

DH: This isn't quite accurate - there will be a very small price increase, principally to cover courier delivery (which is now standard for manual orders) and changes in the exchange rate between the US and NZ dollar.

Making money is not a major aim of mine with PMail, as long as it

Continued on page 5

A Few of PMail's Supporters and Users

Department of Defense/Navy
Charleston, South Carolina

Conservation Committee of
California Oil and Gas Producers

Alberta Power Ltd.
Edmonton, Alberta, Canada

State government of Minnesota

Cambridge Systematics Inc.
Cambridge, Mass.

MediSense Inc.
Waltham, Massachusetts

—Pegasus Mail—

A Free E-mail System For Novell's Netware

An interview by Terri Thatcher with PMail author, David Harris. The support of PMail is an ongoing effort driven by Harris and contributors via the Internet and the PMail Users Group. PMail is free software and was first introduced on the Internet in 1990 and has enjoyed world-wide attention, recognition and acceptance.

TT: What is Pmail?

DH: Pegasus Mail (PMail) is an electronic mail system principally intended for use on Novell NetWare LANs. It supports both RFC822 and NetWare MHS mail formats - RFC822 mail can be sent and received using a variety of transports including two native dedicated SMTP servers, POP3 clients and UUCP gateways. It's extremely feature-rich, easy to use, and totally free. There are currently versions of PMail for DOS, Macintosh and Windows.

TT: How long has it been around?

DH: The earliest version of PMail was released in 1990; since then it has been through over thirty revisions and is growing stronger by the day.

While it's impossible to tell the actual number of users for PMail, it's safe to assume that it is in excess of a million world-wide at over 20,000 sites.

TT: Why spend your time supporting a free software product?

DH: I can't give any psychologically sound answer to this question, nor to its frequent partner - "why is it free?". I've always regarded PMail as a service to the Internet community - as my contribution towards furthering the growth of the most important communications system in the world. Ever since the first day I released PMail it has been free for anyone to use and it's still an overriding concern of mine that it stays that way.

"Why not make PMail proprietary...?" "That's like asking a religious person to give up his/her religion. It's not even worth contemplating."

I started selling optional printed manual sets a couple of years ago, principally to cover development costs, but I always try to stress to people that manual purchases really ARE optional - there isn't even a moral obligation to buy. I'm not doing any of this for money, but I can't really tell you why I AM doing it... I guess it just feels right.

TT: What advantage do you feel free versions of PMail provide that are not available with proprietary versions of e-mail?

DH: I've always thought the greatest advantage is that PMail is standards-based; it uses standard formats for mail messages and has no proprietary or closed transports or formats. This makes it easy to integrate with other systems whereas many of the commercial offerings require expensive or complex gateways which often transform the message in unpleasant ways.

PMail is very easy to install and maintain - it takes about 30 seconds and there is no ongoing maintenance at all, because it is fully-integrated with the NetWare user database. This changes somewhat in NetWare 4.0, which is a nightmare for an e-mail developer to support, but I'm working on that.

Finally, because PMail is free, it's possible to make it available to absolutely everyone in the organization. Many large sites find the cost constraints of expensive commercial mailers limiting in that they can only afford to provide e-mail to certain groups of users, which seriously limits its effectiveness. My prime goal is to give

Continued on page 4

Did Y ou Know?

Hitachi America Ltd., offers the GNU development toolchain as a standard product for testing the Hitachi H8/300 processor.

Free Software and NetWare
Continued from page 1

down to PC prices, and now everybody wants high-performance, networked computers. The standards are in place: NetWare on PCs, AppleTalk on Macintoshes, and TCP/IP on Unix machines. Most vendors have written the protocol translators so that their systems can interoperate with the installed base. The task that remains is to deliver top application performance to users, and this is where free software comes in (again).

"By using free software to create a common development environment, Novell will offer the first unified environment that targets both Intel and high-performance RISC platforms..."

Compatibility and performance are the two key issues for network applications. Application compatibility is achieved through the use of consistent interfaces called APIs—Application Program Interfaces on ABI's—Application Binary Interfaces. The API layer is typically specified in ANSI C, the *lingua franca* of portable programming. An Application Binary Interface (ABI) permits the highest performance connection by describing how two applications must communicate without intermediate translation.

One of the key features of Novell's NetWare is that it is extensible. This means that users can add services to their network operating system as they choose. The mechanism that supports this is the NetWare Loadable Module, or NLM. Users write NLM's to add features such as directory services, network administration, security, and database services to the NetWare kernel. Because they are linked directly into the kernel, NLM's provide performance that cannot be achieved using operating systems like Unix.

To provide the basis for portable, binary compliant applications, as well as to achieve the greatest application performance, Novell has tapped Cygnus Support to port the GNU C and C++ development tools to support the latest version of NetWare. With these tools, thousands of developers who currently develop NetWare services and applications will be able to use a single toolchain for NetWare on a variety of microprocessors or, Process Independant NetWare (PIN) platforms.

The benefit to users and developers alike is enormous. The single greatest obstacle to the success of Unix has been the fact that it could not be treated as a common market. In the Unix world, each vendor supplies their own, unique set of tools, forcing application developers to choose how they will deploy their limited resources. By using free software to create a common development environment, Novell will offer the first unified environment that targets both Intel and high-performance RISC platforms such as Sun's SPARC, Digital's Alpha, and HP's PA-RISC architectures.

The first release of the GNU tools for PIN will be on Novell's UnixWare and Sun's Solaris operating systems. Because these tools are freely redistributable, support for other environments and processor architectures are solely a function of market demand. Cygnus and Novell share the belief that this use of free software will deliver the win-win solution that developers and users have been demanding. When delivered in March of 1994, free software will set a new standard for performance, function, and value for users.

`gdb`: A Key Tool in the Development of Open Systems

By Jose Fernandez

Software projects live and die by the tools supporting them. Good tools improve the productivity of the programming team and the quality of the created code. The expanding population of software tool vendors and the significant growth of some of those vendors is evidence that people are looking for and finding tools to help them work effectively.

`gdb`, the GNU debugger, is a particularly valuable tool. It provides a rich set of debugging capabilities, it is available on a number of platforms, it is well maintained and constantly being improved, and its sources are available to the public under the GNU Public License.

The following is a very brief tour of `gdb`'s capabilities. I've tried to touch on the standard features provided by all debuggers. In

Continued on page 5

free software report

Volume 2, Number 3

Free Software: An Agent For Open Systems

Free Software and the Evolution of NetW are

By Michael T iemann

Ten years ago, a small start-up in Provo, Utah began developing software that would enable users to link together personal computers and peripherals on a network. Today Novell, Inc. is one of the largest software companies, and NetWare, its flagship product, runs on an estimated 70% of all computer networks. Ten years ago, it was pretty much inconceivable that one would want to network together personal computers, mainframes, and Unix machines. But, today all three classes of machines are vitally important in virtually every company employing more than mom and pop. Ten years ago it would have been absurd to think that something as simple as NetWare could develop into a billion dollar software company.

Is there a chance that free software will accomplish something like this in the near future?

From the beginning, standards have been a key requirement for networking. It's just not possible for machines to interoperate if they use incompatible protocols. Novell's task for NetWare remains the same today as it was ten years ago: to make every network application "NetWare aware", and to keep its primary product current with ever-changing computer technology and application requirements. Novell's success is a strong testimony to the fact that it has done both of these consistently.

Over the course of the past ten years, the industry has learned that there's more to a computer network than being able to initiate a print job on any printer from any console. The duality of computing (managed by operating systems) and communications (managed by network services) has become quite blurry as operating systems increase their support for networking and networking services act more and more like operating systems. The interesting question is not *which will win?*, but, *how can the user most benefit?*.

Unix, as most of us know it, has always had strong networking support. This is because the Computer Science Research Group at the University of California, Berkeley, integrated the TCP/IP protocol into their Unix kernel. In no time Sun Microsystems communicated their vision "the network *is* the computer", other Unix workstation vendors built upon the free Berkeley code, and networked computing was born.

It only took ten years, but PCs have reached workstation performance. Workstations have come

Continued on page 2

In this issue:

Free Software and the Evolution of NetWare pg. 1
Michael Tiemann

gdb: A Key Tool in the Development of Open Systems pg. 2
Jose Fernandez

Pegasus Mail - A Free E-mail System for Novell's NetWare pg. 3
Terri Thatcher/David Harris

More Reading on Free Software pg. 7

Access Information pg. 7