HP3000 Evolution

-Homestead

-Tune Up

-Migrate

Edited by Bob Green.
From articles written by Robelle, by *The 3000 Newswire*, and by experts in the HP 3000 field: Chris Edler, Paul Edwards, Marius Schild, Neil Armstrong, John Burke, Alan Wyman and Alan Heter, Stan Sieler and Gavin Scott, Ken Robertson, Eugene Volokh (VESoft), Alfredo Rego, Fred White, Steve Hammond, Wayne Boyer, Alan Yeo, Gunnar Fredlund, Terry O'Brien, Michael Marxmeier, Aaron Holmes, Dave Lo, and Glenn Cole.

robelle
solutions technology

Updated Friday, May 07, 2004

PRINTED IN CANADA

Robelle Solutions Technology Inc.

7360 137 Street, Suite 372

Surrey, B.C. Canada V3W 1A3

Phone:	604.501.2001
Fax:	604.501.2003
E-mail:	info@robelle.com
E-mail:	support@robelle.com
Web:	www.robelle.com

Contents

Introduction

Migrate, Homestead, or Evolve?

Edited and Published by Robelle

November 14th, 2001 Hewlett-Packard Company announced changes to the future of the HP 3000 platform that have impacted every HP 3000 customer, vendor and HP employee who worked in the CSY division. The 3000 platform has a lot of dedicated friends and supporters, Robelle included. HP gave us a 5-year notice of end of support for HP 3000, which is more than many other vendors would do.

About the Editor: Bob Green started work at Hewlett-Packard Company at age 19 while still an undergraduate at Stanford. When he joined the computer division, there were only 7 programmers in the group. Bob worked at HP for 7 years as programmer, technical writer, training instructor, course developer, and customer support engineer. He left HP to work for an HP customer site in Vancouver BC. Spent four years designing and implementing on-line order processing, invoicing, sales reports, inventory control, and accounts receivable on an HP 3000 server. It supported 37 user CRTs on a system with 128K bytes of main memory. In 1977 Bob founded Robelle to develop software tools for HP sites and to do consulting work. Bob wrote the firms premier products Qedit and Suprtool.

When Bob Green told Ron Seybold (editor of *The 3000 Newswire*) in the Spring of 2003 that he was putting together a new book on the 3000, Ron asked him some good questions:

Why do one of these now? After all, HP says the ecosystem is drying up...

"For the first time in years, owners of HP 3000 computer systems really have a strong need for information. They need to decide what to do about HP's dropping of the 3000. Do they devote all their resources to migrating, do they stay on the 3000 and homestead as long as possible, or do they, as many of our customers are doing, try a mixture of the two, migrating some applications while retaining the 3000 for others."

And why not just do it all on the Web?

"There is an amazing amount of information on the web about 3000 homesteading and migration, but the average person doesn't have time to sift through it all to find the useful bits. There is still a nice place for knowledgeable editing and printed books in today's world."

What's it going to cost?

"The cost will be $25 (US) per copy."

How can people get it?

"Information about on-line ordering is available on the web at `www.robelle.com/books` - simply supply a credit card number; the book can be shipped worldwide."

The Future

Even after HP waves goodbye in 2006, there will be 3rd party support for the 3000, both hardware and software. When HP cancelled support on the 10-year old 9x7 line, immediately a lively business sprung up to provide 3rd party maintenance for those systems. For software support, there are incredibly talented vendors offering quality service and products you know and trust: Allegro, Adager, Robelle, MB Foster, Beechglen and Bradmark, to name a few.

We don't want to minimize the disruption that HP's announcement will cause some users. The cost to change IT systems is high, especially when switching from a known reliable platform like MPE to a new platform of unknown quality. But these days you do not need everything in your shop on one platform. HP 3000 systems can easily share data with IBM, SUN, HP-UX and Windows, even Linux.

Updates to *HP 3000 Evolution*

Over time we will gather corrections and additions to this book and they will be published on a special web page:

```
http://www.robelle.com/books
```

This page will include a list of the web links from each chapter so that you don't have to type them in manually! If you find any that no longer work, notify the editor at bgreen@robelle.com - thanks.

The Title "HP 3000 Evolution" ?

While Bob was working on the idea for this book, the following short news item appeared in his email In basket from *The 3000 Newswire* and gave him the idea for the book title:

What's Migration Got To Do With It?

The term migration might have been the first used in the 3000 transition story, but watching HP move away from it shows the real picture of change in a community that never embraced it with much gusto. While HP's own communications are now pumping up evolution, one of the prospective vendors of a 3000 hardware emulator says the term migration didn't apply to many 3000 customers anyway.

"Migration was never the right answer for more than maybe 10-20 percent of customers," said Gavin Scott of Allegro Consultants, after his pronouncement that the 3000 community looks more stable than it has in several years. "But migration is what HP was preaching from Nov. 14th, 2001 onwards, and that affected a lot of people's thought processes last year. Homesteading has replaced Migration as the #1 topic of discussion."

"Since almost nobody is moving yet, it's hard to tell what direction they'll ultimately end up going in," Scott added. "But whereas last year people were at least facing in the migration direction, today they seem to be facing the "do nothing" (i.e., homestead) direction. Once they are forced to start actually taking steps, then they may yet head off in a different direction.

"Today homesteading may sound good because it's no work or cost at all for the most part -- but today you can still buy new 3000s from HP. Maybe people will start feeling more insecure next year, or in 2007."

If you don't panic, we believe you can continue to rely on your 3000s almost as long as you want.

Homestead <u>and</u> Migrate

At a time of your choice and in a manner of your choosing, you can migrate the applications off your HP 3000. Perhaps you will even migrate them one application at a time, step by step, in order to avoid the risk of a catastrophic IT failure. It isn't expensive to keep a 3000 running a few extra years. And if you move to HP-UX, you will find several of your favorite tools already living there, including our Qedit and Suprtool.

Each site is different, but it now appears that the most common response will be a mixture of migrating and homesteading, server by server, and application by application. Some applications are easy to migrate; others are quite difficult (what if you have lost some of the source code?).

This book is designed to help you survive without Hewlett-Packard, to keep your 3000 humming, **and** to migrate some IT logic and data from the 3000 to other platforms at a time of your choice. We mined the work of numerous HP 3000 experts for the most readable and timely articles, then wrote some of our own to fill in the gaps. Many of the authors kindly suggested articles by other authors, which allowed us to pull it all together in a short time. We could not include every good article on the topic – please accept our apology if yours is missing.

Thanks and Acknowledgments

Bob Green of Robelle selected and edited all of the chapters in **HP 3000 Evolution**, and wrote several of them. However, the book would not have been possible without the expertise and willing help of many people in the HP community:

Chris Edler (HP)

Paul Edwards (independent consultant)

Marius Schild (Robelle distributor in Holland)

Neil Armstrong (Robelle)

John Burke (independent consultant)

Alan Wyman and Alan Heter (US Foodservice)

Stan Sieler and Gavin Scott (Allegro Consultants)

Ken Robertson (formerly Robelle, now Adtech)

Eugene Volokh (VESoft)

Alfredo Rego (Adager)

Fred White (retired from Adager)

Steve Hammond (AAMC and *The 3000 Newswire*)

Wayne Boyer (Cal-Logic)

Alan Yeo (ScreenJet)

Gunnar Fredlund (CIO Technologies)

Michael Marxmeier (Eloquence)

Terry O'Brien (DISC)

Aaron Holmes (Robelle)

Dave Lo (formerly of Robelle)

Glenn Cole (consultant)

Also, thanks for proofreading and formatting to François Desrochers, Janine Edwards and Neil Armstrong of Robelle.

A Special Thanks

And a special thanks to **Ron Seybold**, the editor of ***The 3000 Newswire***. Ron is an amazing writer, editor and journalist. Ron actually makes a living by writing about our HP 3000 community and its concerns. His magazine is the most detailed, most accurate, most timely source of information that we have. It is a wealth of insights, technical tips, and news. Ron very generously agreed to let us reprint some articles from The NewsWire in this book. Without them, the breadth of the book would be much narrower.

We encourage everyone to become a subscriber to ***The 3000 Newswire*** (and an advertiser if you are in the business). Here is how the Newswire describes its mission and history:

> *Evolution requires explicit information: 3000 NewsWire maintains its mission in a transition era:*
>
> First published in 1995, The 3000 NewsWire remains the only independent publication dedicated exclusively to the HP 3000. Readers from some of the largest HP customer sites to single-person IT departments have discovered ways to achieve more with their 3000 resources through the NewsWire's pages. Now approaching our ninth year, the NewsWire continues its unique perspective of the 3000 marketplace, based on the dedication from its editor Ron Seybold, who wrote his first 3000 article in August, 1984.
>
> In this era of transition for HP 3000 customers, The 3000 NewsWire has added the mission of helping 3000 sites qualify viable alternatives. Whether 3000 users have decided to homestead on the 3000 platform indefinitely, or to migrate to something new or even for those sites which haven't decided anything yet, The 3000 NewsWire continues to deliver the best analysis and information to aid in the decision-making process. Making migration prove its mettle, or exploring the opportunity in OpenMPE, our mission remains the same: Delivering Independent Information to Maximize Your HP 3000.
>
> As they count on information that's always fresh, fair and undaunted, readers rely on the NewsWire to address management, administrative and development needs and concerns within its pages. If you haven't seen the NewsWire, or it's been a long time since you have, contact publisher Abby Lentz at AbbyL@io.com and she will get the current issue to you. If you have a question or concern about the current condition of the HP 3000, or

questions about its future, contact editor Ron Seybold at seybold@io.com and he will see if he can find a solution for you. With a dedication to the HP 3000 community The 3000 NewsWire continues to be always online to help.

The 3000 NewsWire can also be contacted at 3000newswire.com. To start a free trial subscription, just email the publisher Abby Lentz at `abbyl@io.com`

Now a Word From Our Sponsors

The 3000 users of the Suprtool and Qedit software products of Robelle made funding for this project possible. So it only seems equitable to let them have a few words:

Suprtool for MPE and HP-UX - www.suprtool.com

- High performance data manipulation tool
- Lightning fast extracts and sorts
- Exports data to other platforms
- Links data from multiple sources

Enhances TurboIMAGE, Oracle and Eloquence.

"I could not live without Suprtool. I absolutely love it. I use Suprtool mainly for database extractions since it is very quick. Some of our programs have embedded Suprtool calls for automated data extraction. I also use Suprtool to locate data on the fly - for example, when users request data that they cannot get using their regular reports/screens. In these instances, rather than using COBOL, I have created little Suprtool scripts which will search out data in any field in the database tables and produce a report or an output file. This makes the user very happy and also makes me a hero for just a moment!" - Michael J Adrian, Systems Programming and Operations Supervisor, Mt. Diablo Unified School District

Qedit for MPE and HP-UX – www.qedit.com

The full-screen editor for programmers, a classic on MPE and indispensible friend on HP-UX.

Avoid the "vi" Curse: The standard text editor provided with UNIX systems, including HP-UX, is called "vi". It runs in a character-based terminal window and assigns a special meaning to each keystroke (for example, "k" means cursor up). "Vi is free; why not use it?" If you have the time to become a vi expert, fine. If not, you can have the familiar, dependable Qedit.

"I look forward to going to work every day so that I can use Qedit. It makes me 4 to 5 times more productive." - Frank Kelly, Manager Information Resources, Reserve Officer Association

The Past

"I started on the HP 3000 before the first system was shipped from HP and I plan to be there long after the last 3000 is shipped. My firm Robelle is a 3000 vendor, and will remain a 3000 vendor. Robelle plans to support the 3000 with improved Qedit and Suprtool products and service for as long as we have customers, and many other vendors have the same plans. The 3000 and the people who know and support it will be around for a long time. "

"It seemed appropriate to start a book on the HP 3000, perhaps the **last** HP 3000 book, with some history of how this amazing computing platform developed. So turn to the first chapter and read on…" – Bob Green

Early History of the HP 3000:
The Rise, Fall and Rise of the HP 3000

By Christopher Edler

About the author: Chris Edler has a Master's of Science degree from California State University, Chico. He also holds a Master's of Business Administration degree from the University of Arizona and a Bachelor's degree in Economics from the University of Michigan. Chris wrote this history in 1995, before joining HP, twenty years after most of the events but soon enough that the participants could still remember the events. The history was published as Part Three of "HP's Early Computers, The Strongest Castle: The Rise, Fall and Rise of the HP 3000" in the Journal of the Computer History Association of California, ***The Analytical Engine****, Volume 3.1, November 1995. Reproduced here by permission of the author, Chris Edler. Email: chris.edler@hp.com*

"*The HP 3000 was God's gift to computing*" - Hank Cureton, HP engineer[1]

"*Listen, lad: I built this kingdom up from nuthin'. When I started here, all of this was swamp! Other kings said it was *daft* to build a castle in a swamp, but I built it all the same, just to show 'em! It sank into the swamp. So, I built a second one! That sank into the swamp. So I built a *third* one. That burned down, fell over, *then* sank into the swamp. But the fourth one stayed up. And that's what you're gonna get, lad: the *strongest* castle in these islands.*" - Monty Python and the Holy Grail[2]

The Alpha Project

The HP 3000 minicomputer was the first computer developed from the ground up by Hewlett-Packard.[3] It was conceived of in 1969, designed in 1970 and 1971, and first sold in November, 1972 -- only to be withdrawn from market several months later, and not re-released until 1973.[4,5] It has since become one of Hewlett-Packard's most successful products, and to this day contributes significant revenue to the company.

The history of the project is fascinating. This article will describe the early days of the HP 3000, what the machine was to be originally, what it actually became, and what changed (and why) in the meantime. It is a narrative of a machine that faced daunting failures in the beginning yet, after re-release, found relatively quick acceptance in the business data processing market -- a field then unfamiliar to Hewlett-Packard.[6]

The HP 3000 was, and still is, HP's premier business data processing machine. It runs a proprietary operating system, called MPE for MultiProgramming Environment, and supports both batch and terminal-based users. It can support these different users running different computer languages and applications. The 3000 has been very successful in industrial applications, competing against such industry giants as IBM in manufacturing shop floor applications, order entry, warehousing, and production control.[7]

FOOTNOTE: The task of deciding a name for the operating system was rather arduous, and was the cause of many a bellicose staff meeting. Finally, the name was chosen by an engineering manager as one that offended everyone equally. From an interview with Frank Hublou, op. cit.

The 3000 was a very important factor in Hewlett-Packard's rise to eminence among the world's computer firms. In terms of dollar volume, MPE and MPE-based systems have only very recently been surpassed in sales by the HP-UX line of Unix-compatible systems that conform to HP's current "open systems" concept.[8]

But to the historian, the HP 3000 is significant not only for its long and lucrative success, but for the advanced computing principles it embodied at the outset. The HP 3000 was:

- the first 16 bit machine to have a stack and virtual memory.
- the first minicomputer to implement COBOL.
- the first minicomputer to run a Database Management System.
- one of the first computers to be completely programmed in a high-level language.
- one of the first machines designed by hardware and software engineers.

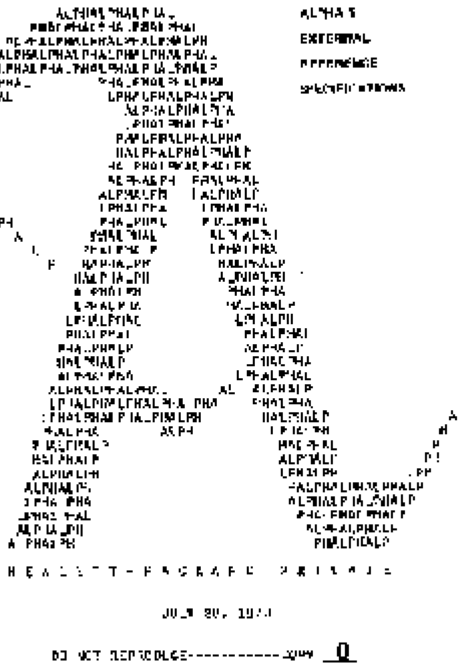
...all for under $100,000.[9,10,11]

The HP 3000 was, and is, undeniably a significant machine, but its achievements were hard-won. It was actually released twice, with an abortive introduction, quick recall, massive redesign, and reintroduction.

Alpha is Born

After the HP21XX series of real-time computers, released in the latter half of the Sixties, the next-generation Hewlett-Packard machine project comprised the 32-bit machine called Omega and a smaller, 16-bit machine named Alpha. Omega was a truly ambitious machine, essentially realizing the concepts of DEC's VAX while pre-dating it by almost 10 years. Unfortunately, it was a bit too ambitious for such a conservative company as Hewlett-Packard, and was cancelled in 1970, after nearly 2 years of work.

The Alpha project was supposed to be simultaneous with Omega, but there were actually a few engineers at most working on Alpha at any given time during the Omega development.[12] In fact, the Alpha 1's External Reference Specifications, included as Figure 2, show a release date of July 20, 1970, less than four months before the Omega was canceled.



ALPHA 1
EXTERNAL
REFERENCE
SPECIFICATIONS

H E W L E T T - P A C K A R D   P R I V A T E

JULY 20, 1970

DO NOT REPRODUCE-----------COPY 0

Figure 2 Alpha External Reference Specs[13]

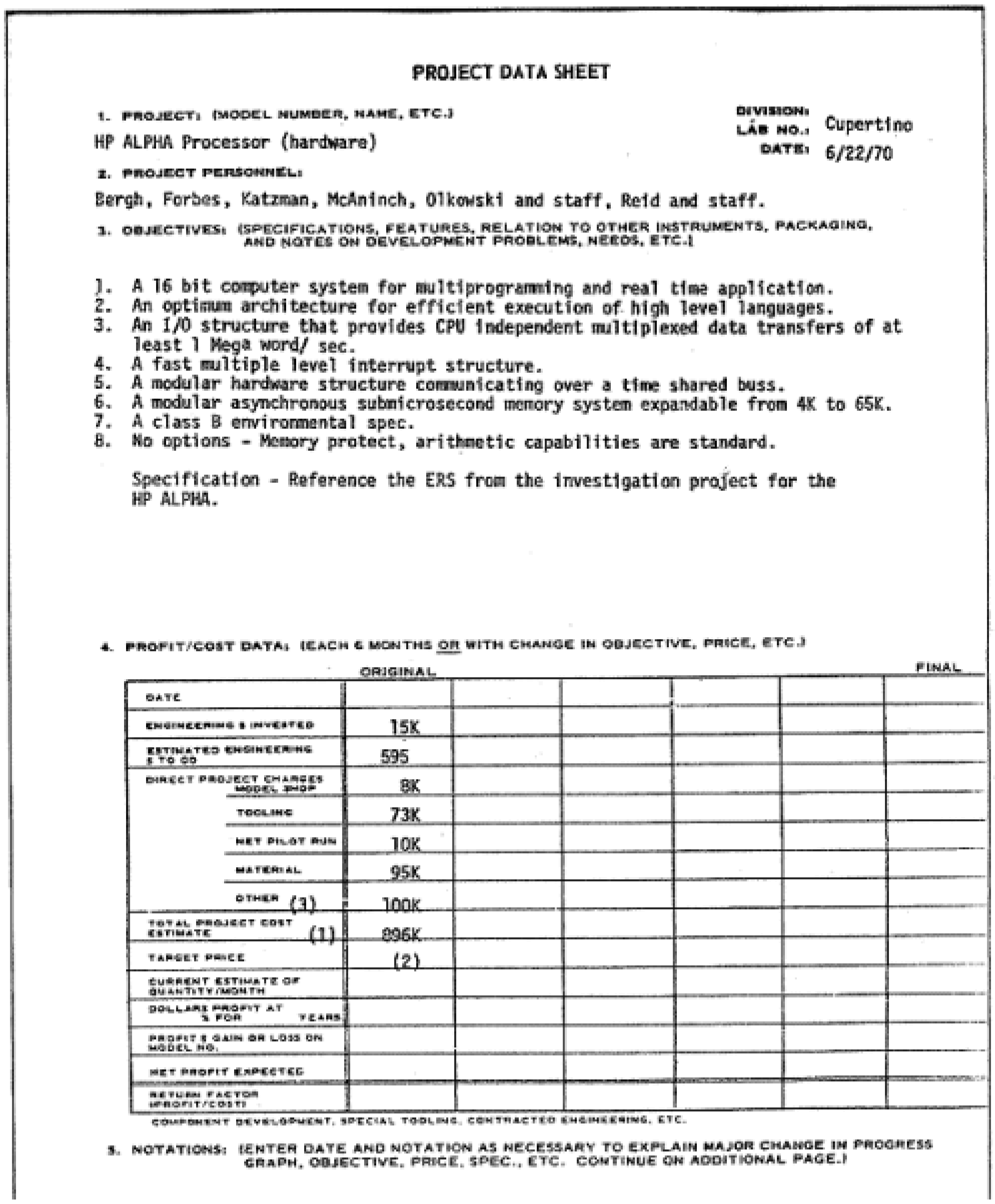
Tom Perkins, General Manager of HP's Data Products Division and star of the HP21XX line, was the man who cancelled the Omega project. He was promoted in the Fall of 1970 to head Corporate Development, after which George Newman was made "GM," as HP-ers called their divisional manager.[14]

Figure 3 shows the original Project Data Sheet for the Alpha hardware system.

Highlights from the "Objectives" section are:

- a 16-bit computer system for multiprogramming and real-time applications.
- an optimum architecture for efficient execution of high level languages
- an I/O structure that provides CPU independent multiplexed data transfers of at least 1 megaword/sec
- a fast multiple level interrupt structure
- a modular hardware structure communicating over a time-shared buss (sic)[15]

Figure 3 Project Data Sheet



PROJECT DATA SHEET

1. PROJECT: (MODEL NUMBER, NAME, ETC.)

HP ALPHA Processor (hardware)

DIVISION:
LAB NO.: Cupertino
DATE: 6/22/70

2. PROJECT PERSONNEL:

Bergh, Forbes, Katzman, McAninch, Olkowski and staff, Reid and staff.

3. OBJECTIVES: (SPECIFICATIONS, FEATURES, RELATION TO OTHER INSTRUMENTS, PACKAGING, AND NOTES ON DEVELOPMENT PROBLEMS, NEEDS, ETC.)

1. A 16 bit computer system for multiprogramming and real time application.
2. An optimum architecture for efficient execution of high level languages.
3. An I/O structure that provides CPU independent multiplexed data transfers of at least 1 Mega word/ sec.
4. A fast multiple level interrupt structure.
5. A modular hardware structure communicating over a time shared buss.
6. A modular asynchronous submicrosecond memory system expandable from 4K to 65K.
7. A class B environmental spec.
8. No options - Memory protect, arithmetic capabilities are standard.

Specification - Reference the ERS from the investigation project for the HP ALPHA.

4. PROFIT/COST DATA: (EACH 6 MONTHS OR WITH CHANGE IN OBJECTIVE, PRICE, ETC.)

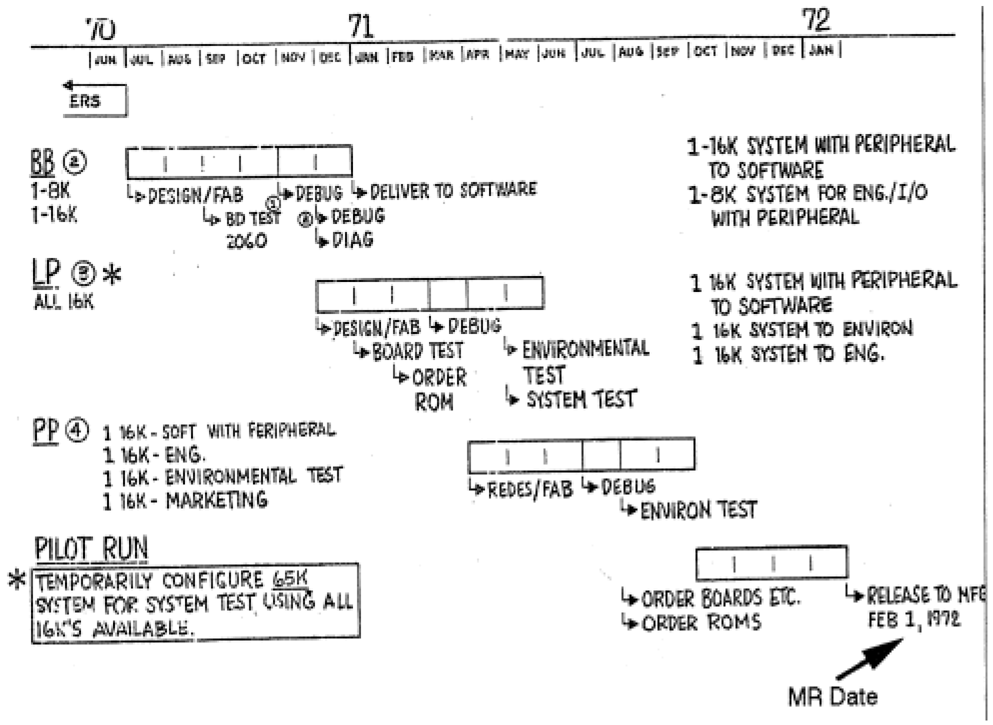
	ORIGINAL					FINAL
DATE						
ENGINEERING $ INVESTED	15K					
ESTIMATED ENGINEERING $ TO GO	595					
DIRECT PROJECT CHARGES MODEL SHOP	8K					
TOOLING	73K					
NET PILOT RUN	10K					
MATERIAL	95K					
OTHER (3)	100K					
TOTAL PROJECT COST ESTIMATE (1)	896K					
TARGET PRICE	(2)					
CURRENT ESTIMATE OF QUANTITY/MONTH						
DOLLARS PROFIT AT % FOR YEARS						
PROFIT $ GAIN OR LOSS ON MODEL NO.						
NET PROFIT EXPECTED						
RETURN FACTOR (PROFIT/COST)						

COMPONENT DEVELOPMENT, SPECIAL TOOLING, CONTRACTED ENGINEERING, ETC.

5. NOTATIONS: (ENTER DATE AND NOTATION AS NECESSARY TO EXPLAIN MAJOR CHANGE IN PROGRESS GRAPH, OBJECTIVE, PRICE, SPEC., ETC. CONTINUE ON ADDITIONAL PAGE.)

Note the estimated costs for the Alpha hardware of $896,000 in mid-1970. Figure 4 is a preliminary timeline that accompanied the Project Data Sheet, showing an MR (manufacturing release) date of February 1, 1972.

Figure 4 Original Timeline[16]



As mentioned, there was great dismay over the cancellation of the Omega machine, and the decision to go ahead with the Alpha was not received enthusiastically. Going back to "just another 16-bit machine"[17] had its discouraging aspects even beyond preoccupation with the word-size; the engineers at HP felt that they had been designing and working with a machine similar to Alpha (HP21XX) for a number of years. The prospect of going "back" to the Alpha might also have seemed pedestrian in contrast to eagerly anticipated work on the 32-bit Omega. For whatever reasons, Omega was sorely and widely missed.

But the Alpha, and subsequently the HP 3000, were to become Omega's memorial - intentionally or not - thanks to one significant decision. The developers, consciously or unconsciously, decided that the Alpha would be as much like the Omega as a 16-bit machine possibly could. As one engineer of that time put it, "We had an Omega mentality in an Alpha box;"[18] in practical terms this meant that almost everything about the Omega was "cut in half" but survived into the Alpha plan. Both were to be virtual memory, stack-based machines with three modes of operation - timeshare, batch and real-time. The major differences were in the "virtual-ness" of Alpha memory (only application code was "virtual", user data was limited to 64K), and in the input/output scheme - a particular disappointment for the I/O engineers, who had designed elaborate input/output hardware systems[19] for Omega's 32 bit data path, which could never be incorporated into the Alpha.[20]

Upper management saw the initial Alpha specifications and had some concerns over the functionality and feasibility of the machine. Arndt (Arnie) Bergh, one of the prime hardware designers for the project, recalls that, "...he [the general manager and one of the people who ordered the cancellation of the Omega] said 'You did it to me again. All I wanted was a simple, clean, inexpensive instrumentation computer that would do the job.'"[21]

In spite of reservations, the project was approved and specifications for the Alpha were completed. By the time the External Reference Specifications were released in July 1970, the vision for the Alpha was complete in broad outline. It would have a 16-bit data word with a single storage register. It has to be stack-based with a maximum memory of 128 Kbytes, or 64K words, of ferrite-core memory.[22] (A common confusion of the period devolved on the exact meaning of "XX K of memory". Most minicomputer memories were measured in 16-bit data words and generally described as nK (thousand)-data-word memories. IBM, however, introduced the concept of 8-bit words called "bytes", which became an alternate general usage that clouded the issue.) The architecture as initially defined allowed for multiple CPU's, although this feature was never realized in the finished product.[23]

The system was designed for modularity, with communication between modules over an asynchronous priority demand data bus. It utilized an "omnibus" concept in which the devices all shared the same bus. The maximum number of modules was seven.

The Operating System

A key distinction between the Alpha and other contemporary minis was to be its operating system - a feature then generally found only in mainframes. Minis were ordinarily bought with and for a single application, or were intended to be programmed in machine code by the user.

[Editor: this was a serious expansion of the scope of the project beyond a typical minicomputer. The "operating systems" on the HP 2100 systems of the time were little more than tape loader programs. They had no scheduler, no dispatcher, no memory management, and no spooler.]

HP envisioned Alpha as the first "general- purpose" or flexible-purpose minicomputer, and an operating system was essential to the concept. Alpha's was to be called MPE, for MultiProgramming Executive.

MPE was actually three operating systems in one. First and foremost, it was to be a timesharing operating system. This was to allow "multiprogramming multiaccess - in particular, time sharing with good term response".[24] The Alpha was intended primarily as a timesharing machine, and its architecture was chosen accordingly.[25]

Secondly, it was to be a real-time system. The designers described the operating environment as commercial real-time and process control systems with fast interrupt response time.[26] Alpha in this context was intended to replace 21XX-series computers in real-time operation.[27] This was a key issue for HP, which considered real-time operating systems to be a "core competency" for the company, and a significant source of revenue.[28]

The third capability of the operating system was a batch mode, in which jobs could be scheduled and run just as if from a user terminal. Proportioning among these three resources could be fine-tuned through MPE, in essence turning off modes not currently needed.

How was this to be accomplished? The External Reference Specifications sketched some of the ways:

- Program sharing of re-entrant code by two or more users simultaneously (e.g. compilers, intrinsics, etc.).
- Automatic resource allocation and overlay of infrequently used memory.
- Protection between users, and between users and the system.
- Modular I/O organization to maximize effective device usage.
- User programs are segmented such that total residence is not required to execute a program.
- Fast interrupt response time and environment switching, plus nesting of high priority interrupts."[29]

Language

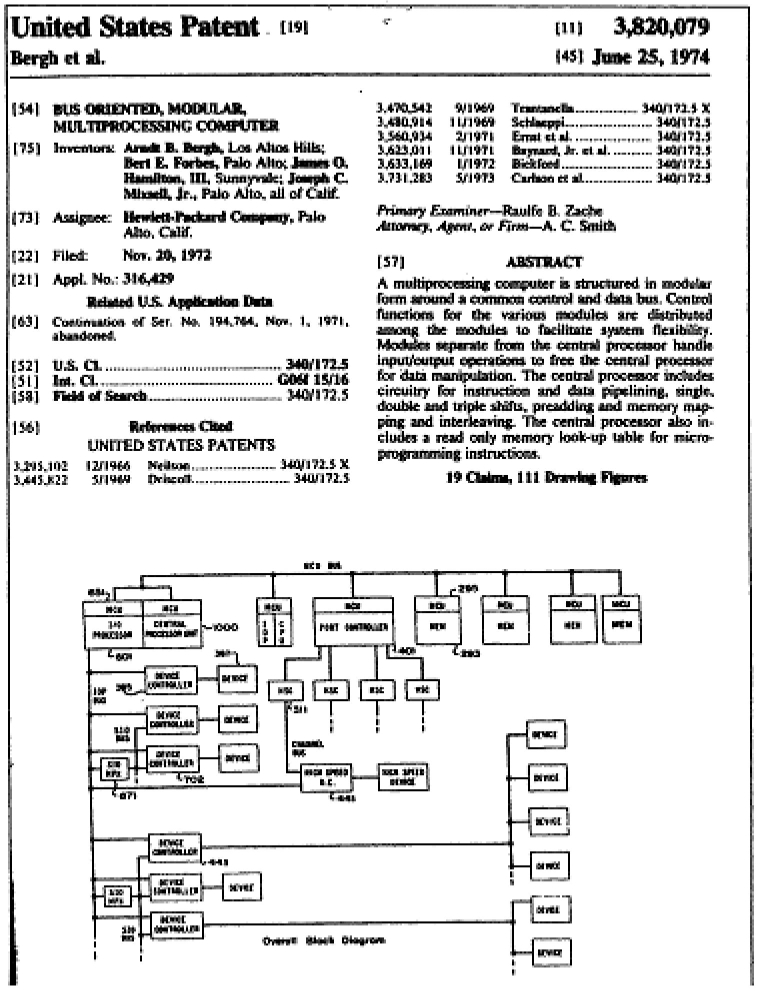
The primary programming language for the Alpha was to be called ASPL (later SPL), for Alpha Systems Programming Language. It was an ALGOL derivative very similar to that of the Burroughs B5500, and - as ALGOL was the first language to handle recursion[30] - was an excellent language for stack-based machines. Alpha was designed from the ground up to accept SPL; in fact, MPE itself was written in SPL, while most other manufacturers wrote their operating systems in assembler.

Filesystem and I/O

The Alpha file system featured named files and directories with controlled access. Similar to modern-day UNIX machines, the Alpha relied on device-independent file access, and made I/O devices accessible as labeled files.

Input and output processing could occur independently of the CPU through an I/O processor (IOP) scheme. A master device, the MCU, controlled assignment of time slices of the bus to modules. This concept was unique enough to earn a United States patent for the machine; the first page is shown in Figure 5.[31]

Figure 5 Original HP 3000 Patent



United States Patent [19]

Bergh et al.

[11] **3,820,079**

[45] **June 25, 1974**

[54] **BUS ORIENTED, MODULAR, MULTIPROCESSING COMPUTER**

[75] Inventors: **Arndt B. Bergh**, Los Altos Hills; **Bert E. Forbes**, Palo Alto; **James O. Hamilton, III**, Sunnyvale; **Joseph C. Mixsell, Jr.**, Palo Alto, all of Calif.

[73] Assignee: **Hewlett-Packard Company**, Palo Alto, Calif.

[22] Filed: **Nov. 20, 1972**

[21] Appl. No.: **316,429**

Related U.S. Application Data

[63] Continuation of Ser. No. 194,764, Nov. 1, 1971, abandoned.

[52] **U.S. Cl.** **340/172.5**
[51] **Int. Cl.** **G06f 15/16**
[58] **Field of Search** 340/172.5

[56] **References Cited**

UNITED STATES PATENTS

3,295,102	12/1966	Neilson	340/172.5 X
3,445,822	5/1969	Driscoll	340/172.5
3,470,542	9/1969	Trantanella	340/172.5 X
3,480,914	11/1969	Schlaeppi	340/172.5
3,560,934	2/1971	Ernst et al.	340/172.5
3,623,011	11/1971	Baynard, Jr. et al.	340/172.5
3,633,169	1/1972	Bickford	340/172.5
3,731,283	5/1973	Carlson et al.	340/172.5

Primary Examiner—Raulfe B. Zache
Attorney, Agent, or Firm—A. C. Smith

[57] **ABSTRACT**

A multiprocessing computer is structured in modular form around a common control and data bus. Control functions for the various modules are distributed among the modules to facilitate system flexibility. Modules separate from the central processor handle input/output operations to free the central processor for data manipulation. The central processor includes circuitry for instruction and data pipelining, single, double and triple shifts, preadding and memory mapping and interleaving. The central processor also includes a read only memory look-up table for microprogramming instructions.

19 Claims, 111 Drawing Figures

The Alpha had 170 instructions (opcodes). For comparison, the IBM System/360 had 142 opcodes. Each code was a 16-bit word, but stack instructions were packed two per word. [*Editor: the 3000 was implemented using microcode, with the underlying micro-instructions being 32 bits wide.]* Instructions were stored on a ROM (read only memory) chip 32 bits wide. [32,33,34]

The instructions were divided into functional areas as follows:

- Memory reference
- Stack & control
- Shifts, bit tests, conditional branches
- Immediate
- Linkage and control instructions

- Special opcodes
- Move opcodes
- Mini opcodes[35]

The machine was initially specified to give satisfactory response time for 16 users when equipped with 16k of memory, or 32 users with 32k of RAM. If the system was only running BASIC, the number of users could be increased by 50%.[36]

The schedule for completion, as quoted in the preliminary document of July 1970, was:

Approval 9/8/70

Preliminary ERS 11/1/70

Final ERS 4/1/71

Begin system integration 5/1/71

System integration 11/1/71

Release 12/31/71

IMS complete 3/1/72[37]

The projected price of the machine was $100,000. It was to be called the "HP System/3000".[38]

The System is Announced

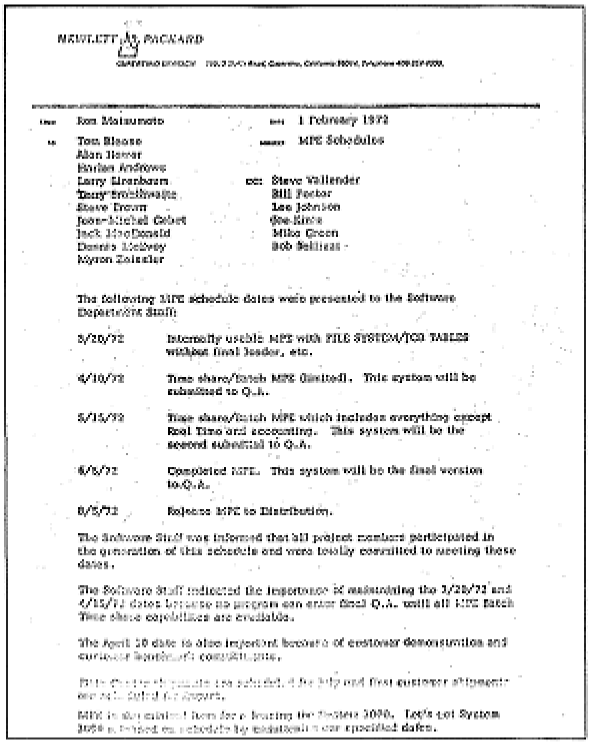
The premier showcases of the computer industry, where companies would announce and demonstrate their new technology, were the Joint Computer Conferences held in the spring and fall of each year. The industry and the press followed these conferences closely.

Hewlett-Packard announced its "System/3000" at the 1971 Fall Joint Computer Conference in Anaheim, CA. Although most Hewlett-Packard users and engineers remembered it vividly, the announcement only merited a small blurb on page 30 of the November 11, 1971 issue of *ComputerWorld*, with the headline "HP adds time-sharing system/3000".[39] The article stated that the 3000 had available RAM memory of 32 to 131K, a maximum cycle time of 950 nanoseconds (pretty good in those days), an instruction set of 170 commands, and could "be used as a front end to an IBM 360".[40] It was to have BASIC, FORTRAN, SPL, as well as COBOL "before the end of the year" (which didn't become available until 1973). The release was to be in August, 1972.[41]

Elsewhere, it was advertised to run up to 16 terminal users on a fully loaded system.[42]

A Cloud on the Horizon

In early 1972 things started to go wrong at the HP 3000 lab in Cupertino. The hardware was up and running in the form of three prototypes, PP1 (Production Prototype 1), PP2, and PR1 (Pilot Run 1).[43] The software effort was slowing down.



HEWLETT PACKARD

[illegible]

FROM Ron Matsumoto DATE 1 February 1972

TO Tom Blease SUBJECT MPE Schedules
Alan Hewer
Harlan Andrews
Larry Eisenbaum
Terry Braithwaite
Steve Brown
Jean-Michel Gobet
Jack MacDonald
Dennis McEvoy
Myron Zeissler

cc: Steve Vallender
Bill Foster
Lee Johnson
Joe Kirss
Mike Green
Bob Bellisas

The following MPE schedule dates were presented to the Software Department Staff:

3/20/72	Internally usable MPE with FILE SYSTEM/TCB TABLES without final loader, etc.
4/10/72	Time share/Batch MPE (limited). This system will be submitted to Q.A.
5/15/72	Time share/Batch MPE which includes everything except Real Time and accounting. This system will be the second submittal to Q.A.
6/5/72	Completed MPE. This system will be the final version to Q.A.
8/5/72	Release MPE to Distribution.

The Software Staff was informed that all project members participated in the generation of this schedule and were totally committed to meeting these dates.

The Software Staff indicated the importance of maintaining the 3/20/72 and 4/15/72 dates because no program can enter final Q.A. until all MPE Batch Time share capabilities are available.

The April 10 date is also important because of customer demonstration and [illegible] commitments.

[illegible]

MPE is the critical item for releasing the System 3000. Let's get System 3000 [illegible] on schedule by maintaining our specified dates.

Figure 6 Memo 2/1/72

The first evidence of this can be found in office documentation in early 1972, when the extent of the 3000's functionality was being reconsidered in an effort to protect the late 1972 deadline. The real-time aspect of the operating system, specifically, would later cease to be included (or even mentioned) in internal documentation.

Figure 6 shows a memo from Ron Matsumoto, a manager in the O/S section of the HP 3000 lab, dated February 1, 1972. Notice that on the 5/15/72 and 6/5/72 lines, real-time has been "stealthed out". This will be one of the last times that real-time is mentioned in a project scheduling memo. From now on, emphasis in Alpha development would be entirely on timeshare and batch functions.

Meanwhile, in a Corner of the Lab

Before we continue, Gentle Reader, allow me a brief aside concerning a project worked on in 1971 by four programmers in a secluded corner of the "Applications Lab", a section of the HP 3000 lab devoted to non-operating-system applications such as the system editor, and to computer-aided instruction software.[44]

File management software was being worked on in tandem with the rest of MPE, and the HP 3000 designers sought to include a file inquiry tool (called QUERY) to enable searching. Two of the programmers involved were Dick MacIntire and Lee Bollinger.[45] Eventually the designers felt that they could expand their utility into a data management application that could be used by not only the operating system, but by all software languages on the HP 3000. The key was a very new concept called database management systems, or DBMS - which may seem commonplace today, but were still in their infancy in the early 1970s.

Editor: Fred White and Jonathan Bale were the other two programmers working on the DBMS project, as opposed to the QUERY project. Read Fred White's very interesting history of the IMAGE project: "The Birth of IMAGE" at:

```
http://www.adager.com/TechnicalPapersHTML/BirthOfImage.html
```

Databases and DBMS's in the 60's and Early 70's

Early DBMS concepts can be traced back to the late 1950's, in academic papers that discussed "generalized routines", which could sort any file regardless of its data content.[46]

Early efforts at a DBMS included programs by GE, by SHARE, and by IBM itself - notably the first release of RPG for the Model 1401 in 1961 - and RCA's Retrieval Command-Oriented Language, developed in 1962[47]; but DBMS theory languished through the mid-60s for lack of a perceived need. One of the first commercially available database systems was written by IBM, initially for the Apollo program, and publicly released for the System/360 in 1969, under the name IMS.[48]

CODASYL, a group mandated to create standards for COBOL, meanwhile formed a List Processing Task Force (later called the Data Base Task Group) to construct COBOL database extensions. They released reports in 1969 and 1971 with recommendations for strategic "building blocks" of a database management system, such as a data description language (DDL)[49] and use of what was called the networked model of databases.[50]

In 1970, a third-party software firm called Cullinane released their database product, IDMS, for IBM 360 and 370, which was extremely successful.[51] Other key products of the period included DMS 1100 for the UNIVAC 1108, and DEC's DBMS-10, which ran on the PDP-10.[52]

IMAGE

At HP, the file management group and the inquiry group merged in September 1971 and decided to work on an integrated data management package. This package, named IMAGE most probably by Dick MacIntire, would allow access to data files by any process on the 3000 by the use of intrinsic commands.

The developers looked at a number of database systems then available. The main data center at HP corporate headquarters had a mainframe running the TOTAL database management system. The team also read and investigated the work of Leo J. Cohen, an early expert in database management concepts[53], as well as the CODASYL report released in 1971. Image followed CODASYL recommendations in being a network-based database, but the product itself was not in full compliance with CODASYL standards.

Scheduled for a second release of the system[54], Image was not a high priority item at the time of the first release. Developers on the project had very little clout when competing for valuable and rare computing time on the HP 3000 prototypes. Most development was done on either an HP2100 emulating a 3000 or (primarily during weekends) on the third HP 3000 prototype, the one frequently cannibalized to keep the other two running.

Development took place throughout 1972 and early 1973, with quality assurance testing beginning in mid-1973. The product could have been released in late 1973, but a late hire of a technical writer postponed the introduction until the 1974 rollout of the HP 3000 "CX" series.

...now, back to the HP 3000 lab in early 1972...

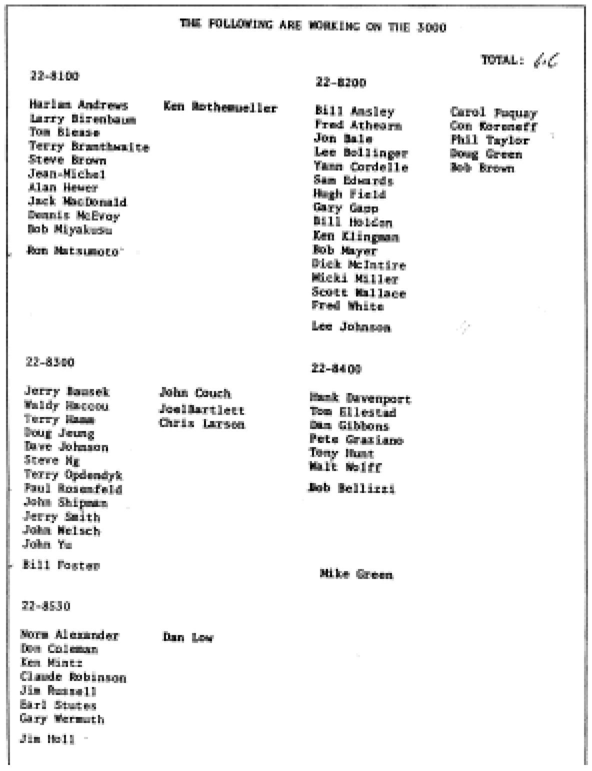
Sudden Infant Death of a Computer

"*...we fired it up, and it was just quite slow...*" Engineer Arnie Bergh on the first HP 3000[55]

Background – The HP 3000 Lab Environment

A tension always prevails between the people who develop a product and those who sell that product. This tension can be good, in that - in the words of Ed McCracken, then Marketing manager of the HP 3000 and now CEO of Silicon Graphics Corporation - "you generally have a marketing person and a manufacturing person who provide some balance against the dreams of an engineering manager."[56] But in less fortunate circumstances, the tension can be so strong as to prove catastrophic.

Figures 7 and 8 are organization charts of each section of the HP 3000 lab in 1972..



THE FOLLOWING ARE WORKING ON THE 3000

TOTAL: 66

22-8100

Harlan Andrews
Larry Birenbaum
Tom Blease
Terry Branthwaite
Steve Brown
Jean-Michel
Alan Hewer
Jack MacDonald
Dennis McEvoy
Bob Miyakusu

Ron Matsumoto

Ken Rothemueller

22-8200

Bill Ansley
Fred Athearn
Jon Bale
Lee Bollinger
Yann Cordelle
Sam Edwards
Hugh Field
Gary Gaşp
Bill Holden
Ken Klingman
Bob Mayer
Dick McIntire
Micki Miller
Scott Wallace
Fred White

Lee Johnson

Carol Fuquay
Con Koroneff
Phil Taylor
Doug Green
Bob Brown

22-8300

Jerry Bausek
Waldy Maccou
Terry Hamm
Doug Jeung
Dave Johnson
Steve Ng
Terry Opdendyk
Paul Rosenfeld
John Shipman
Jerry Smith
John Welsch
John Yu

Bill Foster

John Couch
JoelBartlett
Chris Larson

22-8400

Hank Davenport
Tom Ellestad
Dan Gibbons
Pete Graziano
Tony Hunt
Walt Wolff

Bob Bellizzi

Mike Green

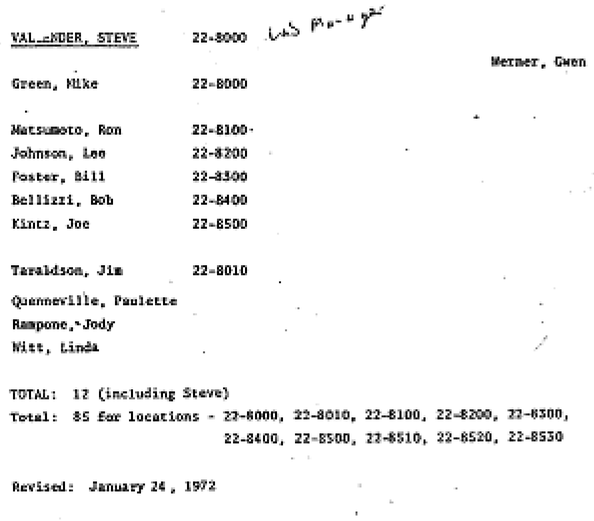
22-8530

Norm Alexander
Don Coleman
Ken Mintz
Claude Robinson
Jim Russell
Earl Stutes
Gary Wermuth

Jim Holl

Dan Low

Figure 8.



VAL_ENDER, STEVE	22-8000	[illegible]
		Werner, Gwen
Green, Mike	22-8000	
Matsumoto, Ron	22-8100	
Johnson, Leo	22-8200	
Foster, Bill	22-8300	
Bellizzi, Bob	22-8400	
Kintz, Joe	22-8500	
Taraldson, Jim	22-8010	
Quenneville, Paulette		
Rampone, Jody		
Witt, Linda		

TOTAL: 12 (including Steve)
Total: 85 for locations - 22-8000, 22-8010, 22-8100, 22-8200, 22-8300, 22-8400, 22-8500, 22-8510, 22-8520, 22-8530

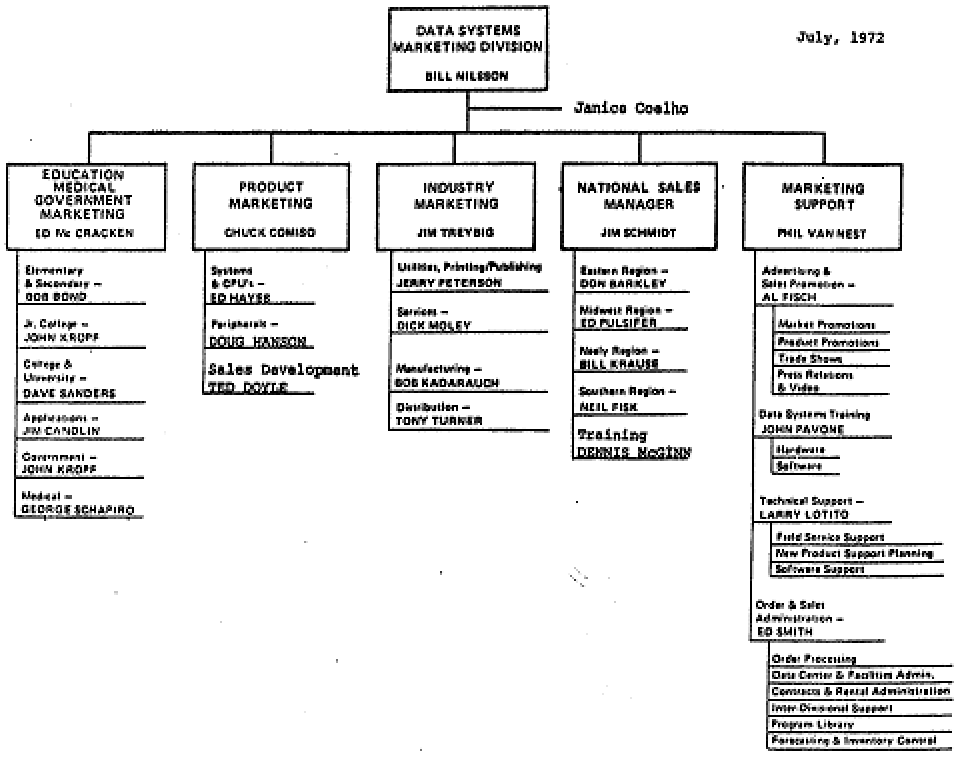
Revised: January 24 , 1972

There were between 60 and 85 people in the HP 3000 lab during the machine's development. This staff were divided into 6 "sections", each devoted to a part of the project, such as the operating system, user applications, or quality assurance. General management was the responsibility of Steve Vallender.

The HP 3000 lab was part of a much larger R & D organization devoted to the many products of the Data Systems Division, which included the HP21XX series of real-time computers.

Marketing was arranged somewhat differently. It was grouped by sector (Educational/Medical/Government, Industry), and comprised smaller product marketing and marketing support groups. Figure 9 shows an organizational chart of the Data Systems Division marketing organization in 1972.

Figure 9 Marketing Organization Chart



DATA SYSTEMS MARKETING DIVISION
BILL NILSSON
July, 1972
Janice Coelho
EDUCATION MEDICAL GOVERNMENT MARKETING
ED Mc CRACKEN
Elementary & Secondary – BOB BOND
Jr. College – JOHN KROPF
College & University – DAVE SANDERS
Applications – JIM CANDLIN
Government – JOHN KROPF
Medical – GEORGE SCHAPIRO
PRODUCT MARKETING
CHUCK COMISO
Systems & CPU's – ED HAYES
Peripherals – DOUG HANSON
Sales Development TED DOYLE
INDUSTRY MARKETING
JIM TREYBIG
Utilities, Printing/Publishing JERRY PETERSON
Services – DICK MOLEY
Manufacturing – BOB KADARAUCH
Distribution – TONY TURNER
NATIONAL SALES MANAGER
JIM SCHMIDT
Eastern Region – DON BARKLEY
Midwest Region – ED PULSIFER
Neely Region – BILL KRAUSE
Southern Region – NEIL FISK
Training DENNIS McGINN
MARKETING SUPPORT
PHIL VAN NEST
Advertising & Sales Promotion – AL FISCH
Market Promotions
Product Promotions
Trade Shows
Press Relations & Video
Data Systems Training JOHN PAVONE
Hardware
Software
Technical Support – LARRY LOTITO
Field Service Support
New Product Support Planning
Software Support
Order & Sales Administration – ED SMITH
Order Processing
Data Center & Facilities Admin.
Contracts & Rental Administration
Inter Divisional Support
Program Library
Forecasting & Inventory Control

There was little love lost between the HP 3000 lab and marketing; or, to put it a better way, there didn't seem to be a lot of mutual respect between the groups - especially with regard to the "owners" of the project, those who would determine the functionality of the product. One of the original engineers put it this way:

"They [marketing] wanted to drive the project from marketing...if they said it could do something, they expected the lab to make it happen..."[57]

This tension was so evident and strong that (in the words of Hank Cureton, an HP 3000 engineer at that time,) "People from marketing were banned from the lab".[58] The turf battle went even beyond that; engineers were not supposed to discuss the product or its progress with anyone in the marketing division. Lab manager Vallender "discouraged us from talking to marketing people", according to Jim Holl, Quality Assurance manager for the HP 3000 software.[59]

Denied access to development information, marketing was unaware of the progress of the product, and of its true specifications. They naively passed on whatever information they could glean to their sales representatives, who presented it to customers. Data sheets were created before the development was complete. Published performance data were, in general, more optimistic than what the lab could measure on running prototypes[60] - and this over-optimism was anything but naive. As Ed McCracken put it, "there was a lot of lying going on."[61] Someone in the lab was even overheard knowingly giving out "rosy" specifications to customers on the phone, hanging up, and saying, "Ah, I've just got to tell them something...".[62]

The tight lid on information about the 3000's progress provoked some speculation by members of the team. Some engineers suspected that the project was in danger of getting dropped. In their view, setbacks could have pushed the HP 3000 off the drawing board - and there had been significant setbacks already.[63] In light of the antagonism between lab and marketing, failures on the development end could have persuaded top management to give overall project direction to marketing. Vallender used a lot of energy defending against this outcome; he "knew how poorly we were doing and he didn't want pressure from marketing."[64]

Spring 1972: Systems Management Group Formed

Top management was indeed concerned. In May of 1972, Dick Hackborn (who would later become Vice President of HP) hired Dave Crockett to head up a marketing team called the Systems Management Group, which would be part of the lab, but at the same time empowered to "product market-ize" the HP 3000.[65]

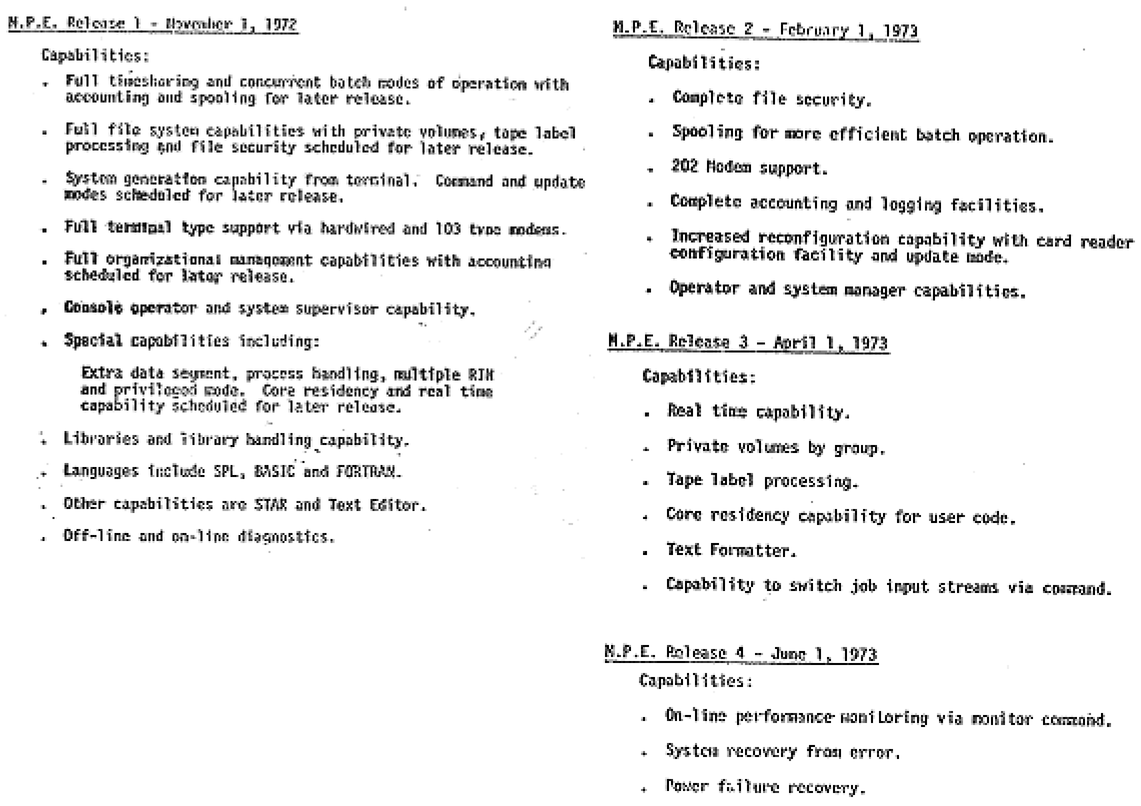
At this point the lab began a serious effort to determine what their new product could do. It is interesting to note that not until 1972, four years after the Omega/Alpha project began, did the lab hire a person - Jim Peachy of the Systems Management group - to analyze the performance of the new machine.

Peachy had done performance testing and tuning for one of the world's first timesharing systems at Dartmouth University in the mid-60s and had worked at Memorex and at General Electric. Three days after HP hired him, he announced "This isn't even a timesharing system."[66] After testing out the instructions, and timing out the loops, he stated categorically that there was "absolutely no way" for the HP 3000 as it existed to meet its timesharing targets. Marketing meanwhile was assuring HP's customers that, at minimum, 16 users could run on a small HP 3000, and 32 to 64 users could run on a 64K memory version of the machine.[67] The Systems Management Group stuck by its calculations and insisted that "this thing [the HP 3000] won't timeshare more than three users..."[68]

This was not an occasion for great joy. Some of the engineers had scant regard for these "polished marketing types from Memorex"[69]. Unfortunately, it was by then the late summer of 1972, and a full "fix" of the product by the release date was out of the question. Probably at this point, the August 1972 release date that had been announced at the 1971 Fall Joint Computer Conference was pushed to November of 1972.

September-November 1972: Delays

The release date, although pushed, was looming by September. Management attempted a full definition of the code set to be distributed as the first release of the MPE operating system. The new functionality contrasts materially with what was originally presented in the earlier memo. The new memo, dated September 25, 1972, is included as Figure 10 and 11.



M.P.E. Release 1 - November 1, 1972

Capabilities:

- Full timesharing and concurrent batch modes of operation with accounting and spooling for later release.
- Full file system capabilities with private volumes, tape label processing and file security scheduled for later release.
- System generation capability from terminal. Command and update modes scheduled for later release.
- Full terminal type support via hardwired and 103 type modems.
- Full organizational management capabilities with accounting scheduled for later release.
- Console operator and system supervisor capability.
- Special capabilities including:

  Extra data segment, process handling, multiple RIN and privileged mode. Core residency and real time capability scheduled for later release.
- Libraries and library handling capability.
- Languages include SPL, BASIC and FORTRAN.
- Other capabilities are STAR and Text Editor.
- Off-line and on-line diagnostics.

M.P.E. Release 2 - February 1, 1973

Capabilities:

- Complete file security.
- Spooling for more efficient batch operation.
- 202 Modem support.
- Complete accounting and logging facilities.
- Increased reconfiguration capability with card reader configuration facility and update mode.
- Operator and system manager capabilities.

M.P.E. Release 3 - April 1, 1973

Capabilities:

- Real time capability.
- Private volumes by group.
- Tape label processing.
- Core residency capability for user code.
- Text Formatter.
- Capability to switch job input streams via command.

M.P.E. Release 4 - June 1, 1973

Capabilities:

- On-line performance monitoring via monitor command.
- System recovery from error.
- Power failure recovery.

The memo on the left memo proposes four major versions of MPE, and details what will be included in each. It mentions that no real-time capability and no spooling (for batch) functionality will be included in the first release. Console operator capability was proposed here for MPE 1, but would later be deferred to MPE 2.

The schedule for the other MPE releases was as follows:

February 1, 1973 MPE 2: spooling

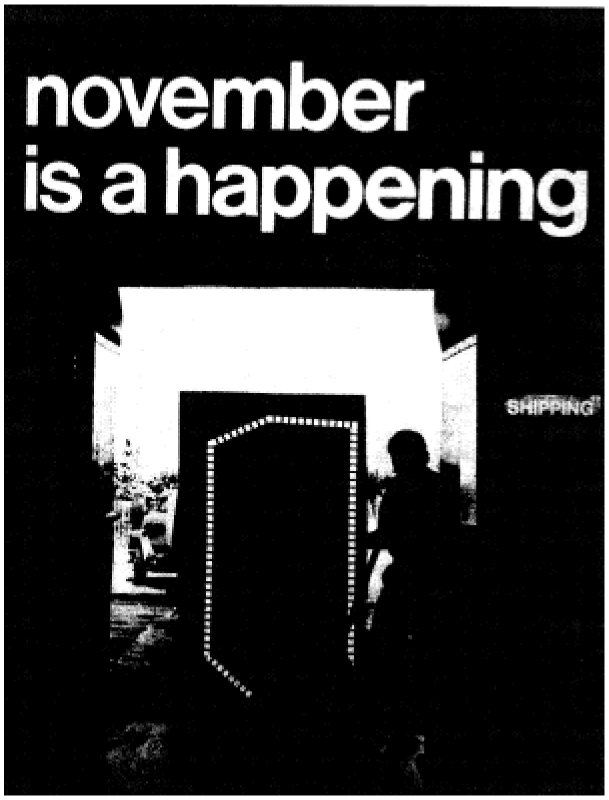
April 1, 1973 MPE 3: real-time

June 1, 1973 MPE 4: system recovery, performance monitoring

This information was to be released to the sales force, and lab members were to go to the regional sales offices and tell them personally.

To say that the salesmen were dismayed would be an understatement. Many of them had already collected commissions from HP 3000 purchases.[70] HP was forecasting that 120 units would be sold the first year, and a $100,000+ HP 3000 carried a nice sized commission.[71]

FOOTNOTE: HP eventually shipped 120 units for the year by the end of 1975. From "Hewlett-Packard takes on the computer giants", op. cit.



November 1972: "November is a Happening"

No one knows who first spoke this phrase, but in the weeks prior to the release of the first HP 3000, dozens of the posters shown in Figure 12 were placed all over the factory floor.

Figure 12 "November is a Happening" Poster

The picture shows an HP 3000 going out the shipping dock door, ostensibly to its first customer, the Lawrence Hall of Science at UC Berkeley. Unfortunately, as Frank Hublou put it, "they should have put it on the truck, drove it around the block, and brought the machine back. That's what should have happened."[72]

It didn't. HP installed the machine, turned it on, and discovered - along with the customer, of course - that the 3000 could only accommodate one or two users before falling to its knees. It was immediately returned.[73]

After the Lawrence Hall of Science debacle, many engineers quipped that they weren't sure if the "November is a Happening" poster showed a man sending the machine out or bringing it back in.[74] Feelings were prevalent throughout the lab that the machine was released much too early, although the engineers knew it could not meet specifications. The November release had been perceived as a firm deadline, which they had to honor regardless.[75]

Perhaps there was an unofficial "1 year after announcement law" in effect which had something to do with the November release. For example, the IBM System/360 was announced in 1963, and the first version was released (arguably in poor condition) in 1964; IBM's System/370 was announced in 1970, and released a year later.[76]

[Editor: this tradition has continued to the present. Just think of all the versions of Microsoft Windows that were delivered a year late.]

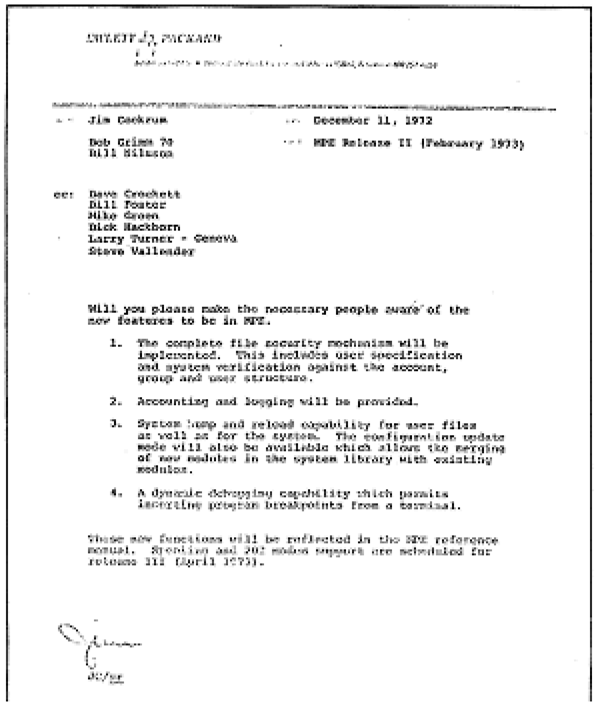
December 1972 – February 1973: Further Delays

Things didn't look that much better a month later, although a prototype used for training in December was able to run reasonably well with 4 users, rather than the earlier 2. During the two-week training session with customers, the prototype crashed about once every two hours, and the operating system "was patched more than 50 times."[77]

December 11 brought another schedule change, detailed in the memo included as Figure 13. Note that spooling, originally promised by February 1, was moved to MPE release 3 (April 1, 1973).

January 31 saw another memo, and another schedule push-back. It is included as Figure 14. In it release 2, scheduled for the following day (February 1), is moved to June 30, 1973. Release 3 is pushed to "late '73", and release 4, mentioned in the September 25, 1972 memo, is not even mentioned. Real-time is gone.

Figure 13 Memo 12/11/72



HEWLETT PACKARD

[illegible]

Jim Cockrum — December 11, 1972

Bob Grimm 70 — MPE Release II (February 1973)
Bill Wilson

cc: Dave Crockett
Bill Foster
Mike Green
Dick Hackborn
Larry Turner - Geneva
Steve Vallender

Will you please make the necessary people aware of the new features to be in MPE.

1. The complete file security mechanism will be implemented. This includes user specification and system verification against the account, group and user structure.

2. Accounting and logging will be provided.

3. System dump and reload capability for user files as well as for the system. The configuration update mode will also be available which allows the merging of new modules in the system library with existing modules.

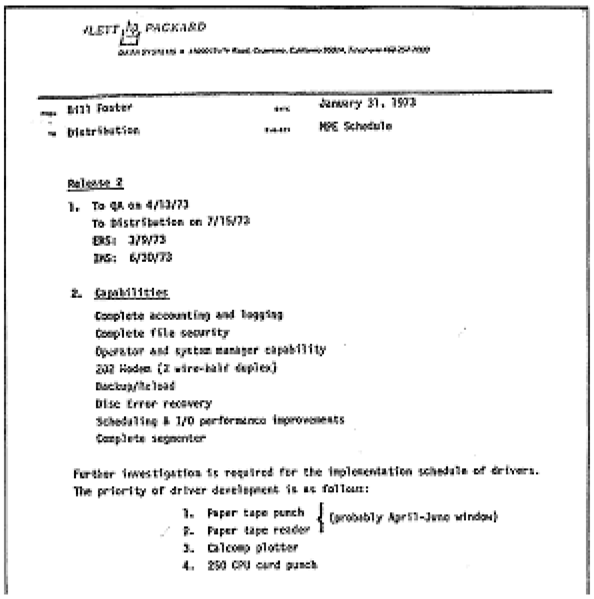
4. A dynamic debugging capability which permits inserting program breakpoints from a terminal.

These new functions will be reflected in the MPE reference manual. Spooling and 202 modem support are scheduled for release III (April 1973).

[illegible]

JC/gr

Figure 14 Memo 1/31/73



HEWLETT PACKARD

Bill Foster — January 31, 1973

Distribution — MPE Schedule

<u>Release 2</u>

1. To QA on 4/13/73
   To Distribution on 7/15/73
   ERS: 3/9/73
   IMS: 6/30/73

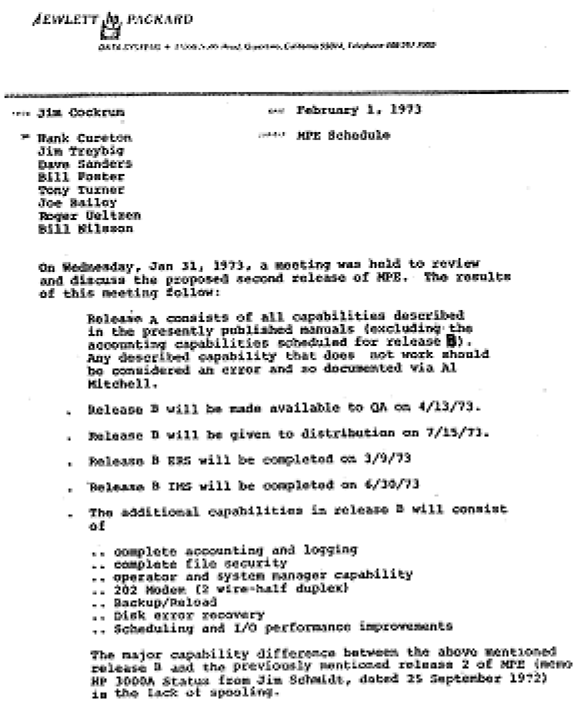
2. <u>Capabilities</u>
   Complete accounting and logging
   Complete file security
   Operator and system manager capability
   202 Modem (2 wire-half duplex)
   Backup/Reload
   Disc Error recovery
   Scheduling & I/O performance improvements
   Complete segmenter

Further investigation is required for the implementation schedule of drivers. The priority of driver development is as follows:

1. Paper tape punch } (probably April-June window)
2. Paper tape reader }
3. Calcomp plotter
4. 250 CPU card punch

Figure 15 shows a memo dated February 1, one day later. It is essentially the same as the January 31 memo, except it leaves out "complete segmentor" from the capabilities of release 2.

In the meantime, the January, 1973 issue of the *HP Journal* (a magazine devoted to the research and products of Hewlett-Packard), devoted a cover story and two other articles on the HP 3000. They were entitled "An economical full-scale multipurpose computer system", by Bert Forbes and Mike Green, "Central bus links modular HP 3000 hardware", by Jamshid Basiji



HEWLETT PACKARD

Jim Cockrum — February 1, 1973

Hank Cureton — MPE Schedule
Jim Treybig
Dave Sanders
Bill Foster
Tony Turner
Joe Bailey
Roger Ueltzen
Bill Nilsson

On Wednesday, Jan 31, 1973, a meeting was held to review and discuss the proposed second release of MPE. The results of this meeting follow:

Release A consists of all capabilities described in the presently published manuals (excluding the accounting capabilities scheduled for release B). Any described capability that does not work should be considered an error and so documented via Al Mitchell.

. Release B will be made available to QA on 4/13/73.

. Release B will be given to distribution on 7/15/73.

. Release B ERS will be completed on 3/9/73

. Release B IMS will be completed on 6/30/73

. The additional capabilities in release B will consist of

.. complete accounting and logging
.. complete file security
.. operator and system manager capability
.. 202 Modem (2 wire-half duplex)
.. Backup/Reload
.. Disk error recovery
.. Scheduling and I/O performance improvements

The major capability difference between the above mentioned release B and the previously mentioned release 2 of MPE (memo HP 3000A Status from Jim Schmidt, dated 25 September 1972) is the lack of spooling.

and Arnie Bergh, "Software for a multilingual computer", by Bill Foster, and "Single operating system serves all HP 3000 users", by Tom Blease and Alan Hewer.[78]

This was Hewlett-Packard's most formal announcement yet of the HP 3000. The first article stated that the primary purpose of the HP 3000 was "to provide, at low cost, a general purpose computer system capable of concurrent batch processing, on-line terminal processing, and real-time processing**, all in the same software**." (emphasis added)[79]

Also in early 1973, George Newman, the divisional manager since after the cancellation of the Omega project, moved to HP's calculator division. He was replaced by Bill Terry.

Spring 1973: There is a Problem . . .

By Spring 1973 the HP 3000 project had been worked on - either as Alpha or as Omega - for almost 5 years, and had cost over $20 million to develop. It was "by far the costliest [program] in the company's history".[80]

Some of the engineers felt that considering the HP 3000 as having been in development since 1968 was a little unfair. They felt that the "clock should have been reset" when the Omega was canceled and Alpha began. But management didn't see it that way.[81]

In late 1972, the sales forecasts for the HP 3000 had been downgraded to 80 for the year, instead of 120. But then, in April of 1973, Bill Terry appeared at an IEEE conference and said that it was now downgraded to 30. Further, at a meeting with security analysts, Mr. Terry said that the software wasn't running as well as hoped, and that it wouldn't support the original plan of 16 users.[82] (Note the "spin control" of stating that the original plan was 16 terminal users. The original plan had envisioned more than 16 users.)

Even though the software was not yet defect-free, the hardware was remarkably robust. One of the other early customers, Yale New Haven Hospital, stated that "the hardware has given very little trouble".[83] But other early customers were not having as easy a time with the HP 3000. An early system was loaned to RAIR Inc., a timesharing company in Mountain View, to test timesharing, but it was returned. National Bank of Detroit had trouble with putting more than 4 terminals on it.

It was becoming clear that the HP 3000 was having serious problems. These problems were noticed all the way up the company chain, to Bill Hewlett himself. Hewlett asked Barney Oliver (who would soon become head of all of HP's corporate R & D programs) to come in and save the product. He had stepped in earlier and pulled the HP 21XX line out of a mess. Oliver refused.[84]

Word was all over the company about the HP 3000. That summer, at the HP company picnic, there were conversations at how much money Hewlett-Packard was pouring into the 3000 "rathole" and when the project would be shut down.[85]

It was time to take action. Paul Ely, General Manager of the Microwave Division, was asked to take over Data Products Division and fix the HP 3000 problems. Ely had turned around the Microwave Division earlier and had a reputation as a "get the job done" kind of manager, a driver who could intimidate anyone easily.[86] His management style was not necessarily a typical HP one.

About this time there were a few changes in the Marketing area. Jim Treybig, one of the sector marketing managers, left the company and eventually helped found Tandem Corporation, and Ed McCracken became Marketing manager for the entire division.

Summer 1973: "Wow Ouch"

The new management team, once in place, acted quickly. Production of the HP 3000 was halted, and more importantly, the decision was made to recall the entire existing base of HP 3000s.

At this time, Dave Packard sent a memo to the HP 3000 team. It was only two lines long and said, essentially, that they would never again announce a product that did not then currently meet specifications. Its succinctness revealed how displeased Packard was with the outcome of the program.

Steve Vallender, the lab manager, made a copy for every member of the department. Before sending it out, he made a little comment on Packard's memo, scribbling the words "Wow" and "Ouch" in the margin. The memo was henceforth (and still is over 25 years later) referred to as the "Wow Ouch" memo.[87]

The next two weeks were, as Ed McCracken would later say, "one of the worst two weeks of my life".[88]

McCracken went back to the customers and "de-committed" each and every HP 3000 sold. In essence, he said that:

1. HP could not bring the software components of the system up to full specifications before Fall 1973.
2. HP was devoting "maximum resources" to correcting the problem.
3. The 3000 system would support no more than 4 to 6 simultaneous users.
4. Image, for those beta testing the product, would be delayed until January 1, 1974.[89]

Some customers literally broke down and cried. Many of them were extremely loyal to HP, and believed in the machines as strongly as any developer at the Cupertino lab.

The customers were offered HP 2000 timesharing systems instead. Some accepted them as a stop-gap, but still pushed for their HP 3000's as soon as possible.

One customer, Anderson College in Indiana, threatened the company with a breach-of-contract lawsuit. It was only averted by the decision of the plaintiffs to contact Bill Hewlett first before filing the papers. Hewlett issued an edict to do anything HP could to appease Anderson rather than going to court.[90]

The (New, Improved) HP 3000, Part II

"*… if you have a flop in a void, you may have time to resurrect it and be a success, still in that same void.*" HP 3000 Engineer Jim Holl[91]

Soon after the withdrawal of the HP 3000, lab manager Steve Vallender left the company. Bill Foster, who is now CEO of Stratus Computers (a competitor to Treybig's Tandem), took the reins of the lab from his earlier position as a section manager in the department.

MPE-B

During the [withdrawal of the 3000 from the market], it was the operating system designers who were in the "hot seat". The hardware was rather well-evolved; it had very few bugs upon release. Hewlett-Packard was in essence a hardware company that did not yet consider software a "core competency."[92]

After the 3000 withdrawal, the MPE lab staff diminished to about 8 programmers, all working on MPE-B. Mike Green, one of the brightest engineers at Cupertino, worked personally on the operating system. Bob Miyakusu, from the file system area, led the effort. The other applications were essentially finished.[93,94]

The MPE section attained a frenzy of development. The lab was still constrained by availability of machine time, so the staff worked 24 hours a day on the software. The operating system section split into two 12- hour shifts, to make sure that the computers were being used whenever possible. When one of the recalled machines came back, the lab grabbed it and put it to use.[95]

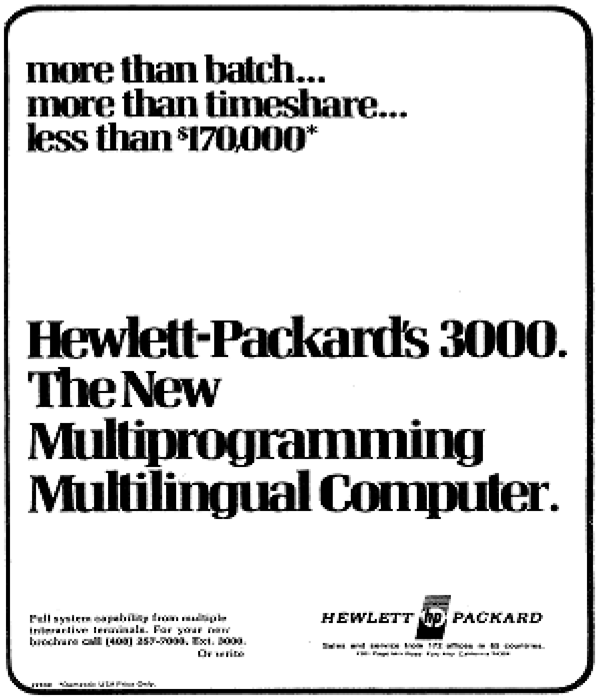
Functionality was also redefined at that time. The remaining real-time code in MPE, unofficially cancelled earlier, was deleted. On several occasions, naturally, a real-time module in MPE was removed, and other, supposedly non-related code promptly crashed. There is probably some real-time code buried deep in MPE to this day.[96]

After about six months spent ironing out the bugs in MPE, the HP 3000 was ready for re-release. It was now able to run 8 interactive users. This in itself was impressive; after the recall, outside experts had predicted that it would take up to "two man-years" to fix MPE.[97]

October 1973: Re-Release

The new HP 3000 was re-released in October, 1973. It now had MPE-B, and was thirty per cent faster thanks to tweaking by the hardware guys. It also cost twenty per cent less. [98,99] The press praised the new machine for having its "...sights and promises better aligned with performance...".[100]

Figure 16 shows an advertisement published after the re-release; it emphasizes the batch and timeshare capabilities of the machine. Notice that, near the bottom of the ad, the customer is urged to get "your new brochure" (emphasis added).



more than batch...
more than timeshare...
less than $170,000*

Hewlett-Packard's 3000.
The New
Multiprogramming
Multilingual Computer.

Full system capability from multiple interactive terminals. For your new brochure call (408) 257-7000, Ext. 3000. Or write

HEWLETT PACKARD

Sales and service from 172 offices in 65 countries.
[illegible]

[illegible] *Domestic USA Price Only.

November 1973: Exit to Tandem

Meanwhile, the HP 3000 project lost several key contributors. Jim Treybig, a marketing section manager who had left in early 1973 to join Tom Perkins (ex-General Manager of Data Products Division) in a venture capital company, started Tandem Computing. Tandem was to produce (and successfully has to this day) a "fault tolerant" minicomputer. Their machine would have redundant circuitry and other hardware to insure maximum up-time even in a catastrophic situation.

Hewlett-Packard was going through a slow period, and the company announced that it would shut down all of its divisions during the Thanksgiving holiday. During that week, Treybig interviewed many of his old comrades at HP to recruit them for his new company. More than a few accepted the offer to join Tandem at its inception. It should be noted that these people left with no ill-feeling from Hewlett-Packard, and there was very little animosity between HP and the newly-formed Tandem.[101] Nonetheless, the departure stripped HP of considerable talent: Mike Green (who wrote Timeshared BASIC), Tom Blease (who had worked on SPL and was an early proponent of the stacked architecture for the Omega/Alpha), and Bob Miyakusu (who led the MPE re-work during the recall period), were among the many who left.[102]

December 1974: Advent of the CX

In the twelve months after the re-release of the product, the lab was working on the next version. It was to be called the HP 3000 CX, to distinguish it from the earlier models (which were to be called, eventually "Series I").[103]

The CX embodied several key improvements in architecture, of which the most important was probably the main memory. The hardware designers returned to the original bus architecture and found a couple of free bits that had been set aside. Using these, they could increase the system's capacity to 128 Kbytes of main memory. The type of memory was also upgraded when Paul Ely issued an edict, "No more core memories", and they promptly went away[104]; HP used semiconductor memory in the new series and finally left the '60s behind.

Also gone was the wire-wrapped backplane, where the circuit boards were attached. The Series I backplane looked like a board splashed with spaghetti, and generated some interesting parasitic electrical currents that plagued the early 3000s. Mysteriously enough, these effects went away if one took some cardboard and stroked the backplane wires.[105]

A further improved operating system, MPE-C, was included, as well as RPG - to compete with IBM - and, probably, the first COBOL compiler on a minicomputer. The COBOL was included to comply with a 1960 Department of Defense specification stating that all machines purchased by them must be supplied with a COBOL compiler. [106,107] Bill Hewlett originally opposed installing COBOL on the 3000, for fear of suggesting that HP wanted to take on IBM; but the demands of the market eventually changed his view.[108]

And finally, HP's database management system IMAGE was sprung on an unsuspecting market. It was a $10,000 option at first, but a year later was offered free and bundled with every HP 3000 system.

Many people mark this as the time when the 3000 left infancy, and began its successful ascent. This is also where my story ends.

Editor: See also Bob Green's history of the Classic 3000 at

```
http://www.robelle.com/smugbook/classic.html
```

NOTES

[1] Hank Cureton, interview by author, tape recording, Cupertino, CA., 14 February 1995.

[2] Monty Python and the Holy Grail, "The Tale of Sir Lancelot", Scene 18.

[3] Carson Kan, Interview with the author, tape recording, 14 February 1995, Cupertino, CA.

[4] "A 10 Year History of the HP 3000," The HP News (1982):4.

[5] Bob Green, Correspondence with the author, December 5, 1994.

[6] George Cullison , Interview with the author, 13 February 1995, Cupertino, CA.

[7] James Cortada, Historical Dictionary of Data Processing (New York; Greenwood Press, 1987).

[8] Rick Justice, interview by author, tape recording, Cupertino, CA., 7 April 1995.

[9] Rich Edwards, "HP 3000 Systems Family History [September 1981]." Internal memorandum.

[10] Michael D Green, Bert E Forbes, "An economical full-scale multipurpose computer system," HP Journal (January 1973):2-8.

[11] Bob Green, op. cit.

[12] Rocky Graziano, interview by author, 21 March 1995.

[13] Alpha 1 External Reference Specifications (Palo Alto, Hewlett-Packard, 1970).

[14] Phyllis Loise Egan, "Hewlett-Packard, From Instrumentation to Computers: A Study of Strategy" (Masters Thesis, San Francisco State University, 1984).

[15] HP ALPHA Project Data Sheet (Palo Alto, Hewlett-Packard, 1970).

[16] Ibid..

[17] Graziano, op. cit.

[18] Kan, op. cit.

[19] Bill Foster, interview by author, 27 April 1995.

[20] Jim Holl, interview by author, tape recording, Cupertino, CA., 14 February 1995.

[21] Arnie Bergh, interview by author, tape recording, Los Altos Hills, CA., 13 February 1995.

[22] Bob Green, op. cit.

[23] Mark Matoza, interview by author, tape recording, Palo Alto, CA., 7 April 1995.

[24] Steve Vallender, "Alpha Multiprogramming System, [28 August 1970]." Internal memorandum.

[25] Bergh, op. cit.

[26] Alpha 1 External Reference Specifications, op. cit.

[27] Bob Miyakusu, interview by author, tape recording, Cupertino, CA., 14 February 1995.

[28] Kan, op. cit.

[29] Vallender, op. cit. pp. 1-2.

[30] Rene Moreau, The Computer Comes of Age, (Cambridge, MA;MIT Press, 1984).

[31] U S Patent Number 3820079 "Bus oriented, modular, multiprocessing computer." issued to A. Bergh, B. Forbes, J.O. Hamilton, J. O. Mixsell.

[32] Daniel Siewiorek, C Gordon Bell, and Allen Newell, Computer Structures: Principals and Examples, (New York; McGraw-Hill, 1982).

[33] System/3000 Software General Information (Palo Alto, Hewlett-Packard, 1971).

[34] Bill Foster (lecture presented to HP 3000 R & D staff, Cupertino, CA., c1971).

[35] Alpha 1 External Reference Specifications, op. cit.

[36] ALPHA Multiprogramming System, op. cit.

[37] ALPHA Multiprogramming System, op. cit.

[38] HP ALPHA Project Data Sheet, op. cit.

[39] "HP adds time-sharing system/3000," ComputerWorld (17 November 1971): 30.

[40] Ibid.

[41] Ibid.

[42] "System/3000 runs into time-share problems," Electronics News (30 April 1973): 1

[43] Bob Green, et. al., Image/3000 Handbook, (Seattle; Wordware, 1984).

[44] John Bale, interview by author, tape recording, Cupertino, CA., 13 February 1995.

[45] Bob Green, et. al., op. cit.

[46] W C McGee, "Generalization: Key to successful electronic data processing" Journal of the ACM 6 (January, 1959):1-23.

[47] James W. Cortada, Historical Dictionary of Data Processing: Technology, pp. 126-27 (Westport, CT; Greenwood Press, 1987)

[48] E H Sibley, "The development of data-base technology," ACM Computing Surveys 8 (1976): 2-7.

[49] Cortada, op. cit.

[50] J P Fry, E H Sibley, "The evolution of data-base management systems," ACM Computing Surveys 8 (1976): 7-42.

[51] R W Taylor, R L Frank, "CODASYL database management system," ACM Computing Surveys 8 (1976): 67-101.

[52] Cortada, op. cit.

[53] John Bale, op. cit.

[54] John Bale, op. cit.

[55] Bergh, op. cit.

[56] Ed McCracken, interview with the author, tape recording, 24April 1995, Mountain View, CA.

[57] Graziano, op. cit.

[58] Cureton, op. cit.

[59] Holl, op. cit.

[60] "HP 3000: A new way of thinking," Measure (Magazine for Hewlett-Packard Employees) (November, 1973).

[61] McCracken, op. cit.

[62] Frank Hublou, interview by author, tape recording, Cupertino, CA., 14 February 1995.

[63] McCracken, op. cit.

[64] Holl, op. cit.

[65] Jim Peachy, interview with the author, tape recording, 6 April 1995, Mountain View, CA.

[66] Ibid.

[67] Ibid.

[68] Ibid.

[69] Graziano, op. cit.

[70] Cureton, op. cit.

[71] "System/3000 runs into time-share problems," op. cit.

[72] Hublou, op. cit.

[73] "System/3000 runs into time-share problems," op. cit.

[74] Bob Heideman, interview by author, tape recording, Cupertino, CA., 13 February 1995.

[75] Graziano, op. cit.

[76] "As time goes by," Datamation 28 (September, 1982):65.

[77] Thomas Harbron Ph.D., correspondence with the author, Anderson University, Indiana, 5 February 1995.

[78] HP Journal (January 1973).

[79] Ibid.

[80] "System/3000 runs into time-share problems", op. cit.

[81] Graziano, op. cit.

[82] "System/3000 runs into time-share problems", op. cit..

[83] "System/3000 runs into time-share problems", op. cit.

[84] Foster, op. cit.

[85] Cullison, op. cit.

[86] Bergh, op. cit.

[87] Hublou, op. cit.

[88] McCracken, op. cit.

[89] Harbron, op. cit.

[90] Harbron, op. cit.

[91] Holl, op. cit.

[92] Bergh, op. cit.

[93] Miyakusu, op. cit.

[94] Graziano, op. cit.

[95] Holl, op. cit.

[96] Miyakusu, op. cit.

[97] "System/3000 runs into time-share problems," op. cit.

[98] Bergh, op. cit.

[99] Laton McCartney, Harvey Wilkinson, "The year mini users shed the security blanket," Infosystems (January 1974):24.

[100] Ibid.

[101] Miyakusu, op. cit.

[102] Kan, op. cit.

[103] Edwards, op. cit.

[104] Bergh, op. cit.

[105] Cullison, op. cit.

[106] Edwards, op. cit.

[107] Andrew L Freidman, Computer Systems Development: History, Organization, and Implementation (Chichester; Wiley, 1989).

[108] Foster, op. cit.

Deja Vue: The Spectrum Project

By Bob Green, Robelle

About the author: In 1974 Bob Green, acting as an HP employee, gave a talk at the first HP 3000 user meeting at Ricky's Hyatt House in Palo Alto, California. Later the products (Qedit and Suprtool) that he created for his firm Robelle were in widespread use at HP sites and successful enough that Bob and Robelle were invited inside HP's new "Spectrum" R&D effort, as you will see in this chapter.

After the initial development problems described in the previous chapter by Chris Edler, the HP 3000 grew and prospered without serious incident. From 1974 to 1984, HP continued to produce more powerful 3000 hardware running more capable software. Each new model was compatible with the previous version and a joy to install.

HP became a force in the commercial server business, with thousands of the 3000s installed worldwide. But the pressure was on to switch to a 32-bit architecture, as other manufacturers were doing.

Then HP announced a radical change: new 32-bit hardware for the 16-bit HP 3000 line. The project was called Spectrum. I will write the history of the Spectrum from an outsider's point of view. As a 3000 consumer and 3000 vendor, Robelle was excited and concerned about the prospect of a new hardware architecture. Certainly it would be wonderful to have more powerful processors, but what about the disruption to our steady incremental, risk-less progress.

The first notice we took of the Spectrum appeared in Robelle's December 1984 customer newsletter, with continuing news to follow for the next four years (my retrospective comments are interpolated in indented paragraphs):

December 12, 1984:

The first Spectrum machine will be an upgrade for the Series 68. Other size models will follow soon after, since HP is working on different Spectrum cpus in three divisions at once (in the past, all 3000 cpus came out of one division). This first Spectrum can be expected in the first half of 1986.

> Please make a note of that **1986** promised delivery date and remember that HP faced serious competition from DEC and others. Customers who loved the 3000, but had outgrown the power of the system, were demanding more capable models.

Spectrum is based on the RISC concept, modified by HP Labs. RISC stands for Reduced Instruction Set Computing. Such a computer has no micro code, only a small number of key instructions implemented in very fast logic. The original Berkeley RISC machine had only 16 instructions. Spectrum has more than 16, but not many more. HP selected the instructions for the fast base set by studying typical application mixes on the existing HP machines. Other functions will be done via subroutines or co-processors (e.g., a floating-point processor, an array processor, or a database processor).

The actual number of instructions in the Spectrum turned out to be about 130, not 16, but they were all simple enough to run in a single clock cycle. HP was the first computer company to go with the RISC philosophy and the only major one to risk the firm by converting all their computer models, both technical and commercial, to a single RISC design.

June 11, 1985:

HP's new Spectrum machine will have both Native-Mode software and 3000 software. The first Spectrum machine to be released will have 3-10 times more computing power than a 68, about 8-10 MIPS in Native Mode. Programs copied straight across will run about twice as fast as on a 68, and those that can be recompiled in Native Mode should run 6-8 times faster. Much of MPE, including the disk portion of the file system, has been recoded in Native Mode. Since most programs spend most of their time within MPE, even programs running in emulation mode should show good performance (unless they are compute-bound).

The expectations were building up in our minds: these machines would be much faster than our current models!

Spectrum will use much of the new operating system software that had been written for Vision, which saves a great deal of development time. Spectrum will use 32-bit data paths and will have a 64-bit address space. Forty Spectrum machines have been built and delivered for internal programming, but product announcement is not likely before 1986.

Vision was an alternative 32-bit computer project at HP, using traditional technology, which was cancelled to make way for the RISC design from HP Labs. Invoking Vision re-assured us that this project is possible, that progress is being made. It is now 6 months after the first announcement of the project.

August 16, 1985:

According to an HP Roundtable reported in the MARUG newsletter:

"Most of what is printed about Spectrum is not to be trusted. Spectrum will be introduced at the end of 1985 and delivered in Spring 1986. There are 40-50 prototypes running in the lab and the project team consists of 700-800 engineers. HP will co-introduce a commercial version and a technical version with the commercial version fine-tuned to handle many interactive users, transaction processing, IMAGE access, and the technical version will be structured for computational programs, engineering applications, and factory automation. HP will eventually offer a choice of

MPE and UNIX. Most software will be available on Spectrum at introduction time and over time all software will be available."

> HP tried to dispel rumors, but still predicted 1986 for delivery. HP will produce two Spectrum lines: the UNIX line for technical users and the MPE line for commercial users, using the exact same hardware.

"The following describes what will be required to convert – Least: restore files and IMAGE databases as they are and run. Next: recompile programs in native mode. Next: change over to new IMAGE database system. Next: change source code to take advantage of RISC." **Robelle Prediction: Spring 1986 for a Spectrum that will reliably run existing MPE applications is not an attainable release date.**

> The new relational HPIMAGE database mentioned here was cancelled much later in the project, after a brief encounter with end-users. I don't remember much about HPIMAGE, except that a lot of work went into it and it didn't succeed as hoped. TurboIMAGE ended up as the database of choice on the Spectrum.
>
> Without any inside information, but based just on past experience and common sense, Robelle tried to inject some caution about the 1986 release date.
>
> During the original traumatic HP 3000 project (see "Wow Ouch" in previous chapter), Dave Packard "sent a memo to the HP 3000 team. It was only two lines long and said, essentially, that they would never again announce a product that did not then currently meet specifications." [Chris Edler] The division listened for over 10 years, but eventually, people forget….

September 20, 1985:

From a Spring 85 UK conference: most existing peripherals will be supported and it will be possible to use networking software to link existing model HP 3000s to Spectrum, with the exception of Series II/III and 30/33. These would need a Series 37 or other current range machine to act as a gateway to Spectrum.

From an HP press release: "100 prototype models were already being used internally for system development as of April 1985."

HPE, the new operating system for the commercial Spectrum is a superset of MPE. It will have two modes of operation: Execute mode (HP 3000) and Native Mode. The switch between the two will be made on a procedure call, but there will be some programming work needed to translate parameters when switching.

> Execute mode was eventually called Compatibility Mode and switching between modes turned out to be major cpu bottleneck in the new system, albeit one that would be removed over time.

The Spectrum is rumored to provide 32 general-purpose registers to the user program and a virtual data space of 2 billion bytes.

December 30, 1985:

From Gerry Wade of HP: The name of the Spectrum machine, when it comes out, will not be Spectrum. Another company already has that name. Spectrum will use the IEEE standard for floating-point arithmetic and will also support the HP 3000 floating point. Each data file will have a flag attached to it that tells which type of floating-point data it contains (the formats are not the same).

> The file flag idea never happened, although the TurboIMAGE database did introduce a new data type to distinguish IEEE floating point. Information on implementation details is starting to flow, which helps us believe that the project is on schedule and likely to deliver the more powerful servers we desire..

June 16, 1986:

In reporting on Joel Birnbaum's Spectrum presentation, the *HP Chronicle* had these observations: "Comparisons with Amdahl and DEC mainframes in slides showed areas where the Spectrum computers topped the larger machines' benchmarks. 'Even with untuned operating systems software, it's significantly superior to the VAX 8600,' Birnbaum said."

> Joel was the HP Labs leader who was the spark plug of the RISC project, building on research that he had done previously at IBM. In retrospect, we can see that Joel was talking about the performance and delivery of the UNIX Spectrum, not the MPE version, but customers took this as a promise of vast performance improvements in the very near future.
>
> It is now past Spring 1986 and the promised new 3000 machines are nowhere in sight. In fact, HP has not yet announced the new models and pricing. This is the first slippage in the project, barely noticed at the time.

July 20, 1986:

Many people have been asking, "What is Robelle doing about Spectrum?" HP has invited us to join its Fast Start program for third parties and we have agreed. This program gives us pre-release access to Spectrum information and actual systems. We have visited Cupertino and run our software on the new machines. We are confident that all of our products will operate properly at the time that Spectrum is officially released."

> Since Suprtool and Qedit were essential to the large 3000 customers that HP was targeting, HP asked Robelle to start porting and testing our products on the new systems. But to do that, we had to sign a Non-Disclosure Agreement, the most

draconian one we had ever seen. We used careful wording in our announcement above. From this date on, until years later, we could not tell our customers anything useful about the new machines. HP was especially sensitive about their reliability and performance.

When we arrived in Cupertino to do our first testing, we found the prototype Spectrum systems crashing every few minutes and running slower than our tiny system 37. We were appalled. Nothing in HP's public statements had prepared us for the state of the project. I had personally gone through a similar situation with the original 3000 in 1972-74, and I wondered if upper management at HP knew how terrible things were. I thought about talking to them, but our NDA also prohibited us from talking to anyone at HP.

The UNIX versions of Spectrum, on the other hand, seemed to be humming along nicely, showing that it was not a hardware problem.

September 11, 1987 (over a year later):

First Spectrum Shipments: Rumor has it that HP shipped the first four 930 machines on Friday, August 21st, with more to follow every week thereafter. As of press time, we have been unable to find out whether ordinary mortals are allowed to touch these machines (as opposed to those who have signed non-disclosure agreements).

Due to the NDA, over a year passed with no Spectrum news in our newsletter. The project was now 18 months past the original promised delivery date, but was still far from completion. Many people wrote articles, about the Spectrum, mostly based on marketing hype from HP, but no one broke the embargo on real information. We were all terrified. The MPE group had dug themselves into a very deep hole, and no one wanted to be the one who caught the eventual backlash.

October 19, 1987:

The Spectrum Song

Orly Larson and his database singers performed again this year at HPWorld, including their new hit, "The Spectrum Song":

If it takes forever, we will wait for you!

For a thousand summers, we will wait for you!

'Til you're truly ready, 'til we're using you!

'Til we see you here, out in our shops!

From the HP Management Roundtable:

Schedule for Shipping Spectrums

"We are shipping equally around the world. Our first shipments went to both North America and Europe. We are acknowledging availability for 930s and 950s through December at this time … We expect by the end of November to be able to have acknowledged the entire backlog."

> HP continued to spin the "shipments" of Spectrums, without mentioning that these were not finished products. The models were 930 and 950 and the operating system was called MPE/XL, changed in later years to MPE/iX when POSIX was integrated into MPE. By this time, HP was worried about their stock price also and did not want any negative news in the financial press, no matter how accurate. As shown by the next Q&A at the roundtable…

Early 930/950 "Shipments"

Question:

"Are the 930s and 950s being shipped or not? In public you tell me they are shipped. In private, however, I hear from both users and HP that these machines are still covered by non-disclosure agreements and that access to these machines is very restricted, even when in customer sites. What is the story?"

Answer:

"MPE/XL architecture is very, very new. There's a million new lines [of code] that go into MPE/XL, and a lot of software sub-systems as well. And so we are being extremely cautious in how we proceed at this point. We're going through what we call a slow ramp-up through the remainder of this year and going into large shipments in 1988. The reason for that is that we want to fully test out the system capability in a large number of customer environments and we want to make sure that the information on what's going on in there and the people involved are protected from outside folks who either benevolently or not benevolently would like to find out what's going on. I'm sure we're going to run into some problems along the way that haven't been encountered in our earlier phases of testing. We haven't really hit these machines with full production pressure yet. We know from experience that when you do that, you uncover things that you could never uncover in testing, even though extremely rigorous. [Rumor has it that the customers receiving Spectrums now are not allowed to put them into production until 1988.] … A lot of people out there would like to see us have trouble and we want to make sure that, if we do have some problems, they're contained in our user community and they don't get out into the press and create problems that will come back to haunt many of you".

> Early Spectrum customers called us to ask which version of Suprtool and Qedit they needed for their new systems, and whether there were any problems that they should be aware of.

But legally, we could not even admit that we knew of the existence of the new servers. Anyone who broke the NDA could expect to be figuratively "shot". So we came up with the following wording: "If you had a new 3000, and we are not admitting that we know anything about a new 3000, you should be using Suprtool version 3.0 and Qedit version 3.6. On this hypothetical system, it might not be a good idea to hit Control-Y while copying a file from any other HP 3000. We can't tell you what will happen, but you won't like it."

February 12, 1988:

Spectrum Finally Leaves the Nest

Hewlett-Packard has officially released the 930 and 950 Spectrum computers, finally abandoning the protection of non-disclosure agreements. We have heard from several sources that the 930 and 950 attained Manufacturing Release during the month of January. This means that people who received "Control Shipment" Spectrums can now put them into production and let outsiders use them. You no longer need to sign any restrictive agreements to get a 930/950. It also means that we users can now compare notes on what the MPE XL systems are good for.

Interestingly, we didn't hear about the Manufacturing Release (MR) of the Spectrum from Hewlett-Packard itself. As far as we can determine, HP kept this event very quiet - no press conferences or splashes of publicity. Even some HP people in Cupertino were not aware that MR had occurred. Just because the 930 and 950 are released does not automatically guarantee that you can get one. Given the huge backlog of orders that HP has, it will make "controlled shipments" for a while, picking sites whose expectations match the state of the machines.

Users had been following Spectrum for almost 4 years now and you can see that we are eager for the release of the product. The MPE lab has grown to hundreds of engineers and technicians and hundreds of Spectrum servers. The amount of money being plowed into the project is awesome. Anyone with any kind of skills is being hired as a consultant, in an attempt to get the situation under control and begin shipping revenue-generating servers.

But we were premature in our February proclamation of Manufacturing Release, which is an HP corporate milestone that requires signed confirmation that the product passes the performance tests set for it in the design specifications.

March 31, 1988:

Spectrum Is Released but Not Released

In our last news memo, we reported that MPE XL users were now removed from non-disclosure restrictions and allowed to put their Spectrum machines into production. In the last month, that news has been confirmed by many sources.

We also concluded, and reported, that MPE XL had obtained MR (Manufacturing Release). **That is untrue.** MPE XL has apparently obtained SR (System Release), but not MR. "System Release" seems to be a new category of release, created just for MPE XL. We have heard from some new 950 customers who did not need to sign a non-disclosure agreement. However, one customer reported that before HP would allow him to order, he had to sign a document stating that he had no specific performance expectations (!?). On the other hand, we heard from a school that recently went live with 35 student sessions and had great response times ("the machine is coasting along at 10% cpu utilization").

> In order to stem the rising tide of bad expectations, HP released the MPE systems even though they could not pass the testing department. And the performance was still poor in many cases, less than the non-RISC 3000s being replaced, although excellent in a few other cases.

Non-disclosure restrictions are **not** lifted for everyone. Sites that are beta-testing subsystems which were not released with the initial MPE XL system are still restricted. Also, third-party FastStart companies such as ourselves are still restricted from passing on any performance or reliability information that we obtain from HP. We face no restrictions regarding performance information received from our customers, so please call with your experiences.

Non-disclosure continues – HP is picking their initial customers carefully and coaching them to only pass on the good news about their new systems. We are still frustrated to not be able to pass on our ideas about how users can improve the performance of the Spectrum.

October 12, 1988:

Childcraft / Series 950. Gary Porto at Childcraft reports that with MPE XL 1.1 the problem of a serial task in a batch job hogging the system is not so bad as it was with 1.0. This problem can occur with SUPRTOOL, QUERY, or any long serial task. The batch job still hogs the system, but at least people can get a minimum of work done. With 1.0, they couldn't even get a colon! Gary reports that he has 65 on-line users on his 64-megabyte Series 950 and that the performance is pretty good - as good as his Series 70.

> On the 4 year anniversary of the project, HP released version 1.1 of MPE/XL, which made the systems much more useful, but still not up to the original promised performance of 1984. However, the promise of the "Precision Architecture" (HPPA) was there, as certain tasks were amazingly fast.

By this time, HP salesmen were getting irritated with us for not giving our customers any kind of endorsement for the switch to the 930/950. But our NDA was not cancelled until Manufacturing Release. Finally, the sales force convinced HP Cupertino to send us a signed release from our NDA. I don't know when MR eventually happened.

From *HP World* magazine:

Early MPE XL Migration Results. London Business School is not a typical installation. Much of their software is written using double precision floating point Fortran which benefits considerably from the Precision Architecture. MIS Director Gordon Miller says "Our straight line performance is up considerably - one program runs 40 times faster - but the performance gains are very application dependent and cannot be accurately forecast beforehand."

Keith Howard of Collier-Jackson in Tampa, Florida participated in the Spectrum beta testing and upgraded from a Series 58 to a Series 950 - quite a leap. One application was found to be 6% slower due to constant switching between compatibility and native modes, but in most circumstances the machine was five to ten times faster than the Series 52 and one batch job ran 53 times faster!

Glaxo Export has temporarily deferred delivery on its second and third 950 systems due to implementation problems on the initial machine. John Walsh of Glaxo related at the Scarborough conference that the first processor was due to go live earlier this year, but when the company came to do a volume throughput test, the machine was unable to load the full operational workload of the Glaxo Series 70. ... Glaxo's problem was caused by the MPE XL 1.0 Loader table being too small, but Glaxo now has a copy of version 1.1 and is about to re-run the volume throughput test.

HP promises performance improvement for Precision Architecture over the next five years of 40-50% per year. Some of this will be achieved by further tuning of MPE XL -- version 1.1 is said to be at least 20% faster overall.

As with the original 3000 project, the birth of the Spectrum was traumatic, expensive and embarrassing. But it paid off. HP was able to roll out better servers for the 3000 line on a regular basis for the next 10 years.

Despite the numerous expansions and revisions to the HP 3000 hardware and software, upgrades have been painless. Even the conversion to the PA-RISC design was backward-compatible and reasonably painless (if you ignore the slipped schedules). Often the user just rolled the new system in on a Sunday, plugged it into the power, reloaded the files, and resumed production. The original 1974 MPE Pocket Reference Card is still useful; everything on it works under MPE/iX version 7.5, except the Cold Load instructions. I have programs that I wrote in 1972, for which the source code was lost years ago, and they still run in Compatibility Mode.

The keywords that describe the history and appeal of the HP 3000 are flexibility, functionality, reliability, and compatibility.

- Bob Green, July 2003

Alternative Resources for Homesteading

By "**Homesteading"** we mean "keeping some or all of your applications on the 3000 even as HP departs and keeping the applications fine-tuned for an indefinite future". (Later we will describe the reasons for migrating - sometimes compelling reasons - and the knowledge you need to migrate successfully.)

Alternative Resources are consultants, service firms, books, web sites, tutorials, and papers that will continue to support the 3000, even as Hewlett-Packard scales back and eventually disappears. The 3000 community is full of smart people who have dedicated their lives to the 3000. They are eager to hear from you and have a history of providing great service, often better service than you get on average from HP. The 3000 world has always had small, nimble firms that lead the way, pushing HP along and filling in when they faltered.

Here are two directories of HP 3000 consultants to show what we mean:

```
http://www.robelle.com/consultants.html
http://www.openmpe.org/mpeconsultant.htm
http://www.adager.com/HpResources8.html
```

Why Homestead?

There are two main arguments in favor of homesteading:

1. The 3000 is rock-solid reliable and should last a long time
2. The cost is low: you already own the system and the software.

But, there are many objections:

What if I need MPE help after HP is gone?

Firms are already competing to provide you with support. Read our profiles of three of them starting on page 48. You might decide to sign up with one of these small, responsive firms now instead of waiting until 2007.

What about hardware support, spare parts, upgrades?

Read our chapters on connecting non-HP peripherals to your 3000, on used equipment, and on independent hardware support (page 58).

What about change and growth in my applications?

If you need something that another system handles better than your 3000, use the other system. Your web server should probably be on a Linux or HP-UX server. Your 3000 can easily talk to the Linux box, and if your Java application crashes, your 3000 will keep processing transactions, unaffected. Read our chapters on using reliable protocols such as FTP and Telnet to link your 3000 to new technologies, on importing data from Excel, and on tapping the hidden powers in the software that you already own (starting at page 169).

What do I do now to get ready?

There are a number of simple steps that you can take now to reduce your risk over the coming years, starting with creating a system Bible binder that records your system configuration. See the "Homesteading Checklist" on page 72.

What about faster hardware in the future?

There is the possibility of a 3000 emulator to run on increasingly more powerful Intel chips. Read our interview with Mark Klein (page 76).

What about training and information for my staff?

See the next section "HP 3000 Tune Up" where we provide a wealth of chapters on operating and fine-tuning the 3000, with reviews and web links for many more. And staff training and assistance is available from numerous independent consultants.

For Example

THE 3000 NEWSWIRE

Take for example, this report from the June 2003 issue of ***The 3000 Newswire:***

Homesteading customers got the first major application provider's advice to stay on their HP 3000s last month, when SSA Global Technologies (SSA-GT) showed that some enterprise-grade vendors want to minimize costly transition changes. SSA-GT purchased the customer base and software assets of MANMAN during 2002.

The option to "support you until the nuts and bolts fall out of your systems" was the first one given to attendees of the Computer Aided Manufacturing Users Society (CAMUS) meeting in Dallas. "Those clients that are not sure of their long term direction can be assured that we will continue to support their use of MANMAN/HP past the 2006 deadline for HP support of the MPE environment," the statement read.

Sue Peyton of SSA-GT told customers at the meeting that the company is confident in the 3000 community's ability to develop its own resources independent of HP.

"ERP systems are like brain surgery — never to be repeated unless the patient is dying," CEO Mike Greenough said in a keynote speech at CAMUS. "Once you've got a system, there's no reason that system shouldn't be extended."

Customers at the conference couldn't see much point in making a commitment to move their application to another platform.

"I can't imagine that anybody would want to go to another platform, and say they want that same old green screen," said Carol Hoinski, IT Director for Teleflex Marine, Inc. Teleflex is homesteading on its HP 3000 and testing the extension products for SSA-GT.

From a business perspective, CAMUS president Warren Smith said the best value for HP sites who want to weather a change is to avoid the port to Unix, and "to move to a package that's already developed today, with additional features beyond the HP environment. But my lowest near-term risk of the three options is to stay right where I'm at; the lowest long-term risk is to stay where I'm at, probably through 2010."

Smith said that moving to a Unix MANMAN port — and about two dozen MANMAN sites have already put down deposits to participate in such a program — looks to him like it represents the "lowest net value, because you freeze the code. The highest ROI I could get would be to stay where I'm at, and pull the plug at sometime in the future."

Less Risky to Stay Than to Go

Dramatic changes to IT systems are very risky:

- Do you still have the people who programmed the system?
- Who knows how the existing system really works?
- Do you have the funds to completely re-engineer and re-invent the system (new servers, new software, new staff, new consultants, etc.)
- What about staff? You currently operate your HP 3000 with a secretary, but you worry that your new system and database may require adding one or more expensive staff members.

For some firms, the answers will obviously be yes and they will migrate. And a properly managed migration can add functionality to your IT systems as well. For this topic, see the last major section of this book, "Migrating a 3000 Application" on page 159.

- Bob Green

Software Support: Gilles Schipper

By Neil Armstrong

About the author: Neil Armstrong is the systems architect for Robelle's product Suprtool, also the primary programmer. Neil started with computers at age 13 and has worked with the HP 3000 since 1981. He also writes articles and papers, including this interview.

robelle

With HP announcing the end of their support for MPE, customers who decide to stay with the HP 3000 will be considering 3rd-party OS support. This is not unreasonable, since there are already at least 10 firms willing to offer this service. The OpenMPE web site has a list of support suppliers:

```
http://www.openmpe.org/mpephone.htm
```

However, when you start working on plans and solutions for independent 3rd-party support for your HP 3000's, it is important to make note of those people who have been in the business for many years. One of those people is **Gilles Schipper** of GSA Inc. (www.gsainc.com), whom I have known and worked with for nearly fifteen years ("gulp").

Gilles Schipper

I had the pleasure of working with Gilles, during my days in Toronto. I learned a great deal from Gilles, and not just technical things and system management, but also how to deal with customers and provide excellent customer service.

My favorite Gilles story is one from the early days. Gilles was recently out on his own having left HP and a customer called him in frustration, dealing with some performance problems.

After having recently bought some more memory, HP had come and installed the memory, but the promised performance gain had not happened. They asked Gilles to take a look. Gilles quickly found that the memory was installed, but the system had not been re-configured with the new memory size. The attention to detail that Gilles taught me when approaching a problem and in dealing with customer concerns in everyday business, has helped me to this very day.

I interviewed Gilles to bring myself up-to-date on what services he offers, and how HP's End-of-life announcement has impacted him.

Can you describe the services that you offer?

We offer software support and system administration services for HP3000 and HP9000 customers.

This includes 7x24x365 telephone support, on-site visits, o/s update and patching installations, performance consulting, network configuration, database optimization, 3rd-party software support, disaster recovery assistance, and virtually any systems administration and software support activity associated with the customers' HP systems.

All of this and more for one low monthly fee - with the assurance that the customer need never spend another nickel for any activity related to our system administration and software support while covered by our service.

You have been in the HP support business for 20 years; to what do you attribute your success?

We have been totally focused on customer support and service since we started business in 1983, after having worked at Hewlett-Packard. Our target market is very narrow, and we like to think we are big fish in a very little pond.

We are also fortunate enough to work with outstanding partners who also have many years of experience in the HP marketplace. These partners include an authorized HP reseller, as well as several 3rd-party software vendors whom we represent.

You are located in the Greater Toronto area; do you only work with clients in your area?

Our customers are located across North America. Specifically, they are in California, Pennsylvania, Colorado, and New York in the US, as well as Ontario, Quebec, New Brunswick, Nova Scotia, and Alberta in Canada.

We visit our customers regularly, and with modern communications capabilities, we are in direct communication at a moment's notice, even when we are absent physically.

Have many of your HP 3000 customers moved to HP-UX?

None as of today. And no plans to do so that we are aware of.

Clearly these plans will change as the future roadmap of the HP 3000 product line becomes more visible in the months ahead.

Of course, with our HP-UX knowledge and experience, we are well positioned to offer our HP 3000 customers appropriate guidance and direction should they choose to migrate to the HP 9000 platform.

Has the End-of-life announcement adversely affected you and your company?

Not at all.

In fact, we have gained added customers since the announcement.

We can only speculate as to reasons for that, but the bottom line is that our business has expanded since November 14, 2001.

We hope to see that trend continue as HP 3000 customers look to alternative support providers that are service-oriented and can meet the very specific and focused needs of HP 3000 users - and who can assure their customers that they will support them for many, many years to come, beyond HP's announced EOS date of 2006.

What do you recommend for customers who are looking for alternate support services?

I want them to know that when deciding upon which HP 3000 software support provider to choose, it is important to consider the history, stability and track record of that provider.

As you are aware, Neil, GSA Inc. has been offering software support services to HP 3000 users longer than any other 3rd-party support provider in existence anywhere - since July 1983.

We have many well-established companies, large and small, who have utilized this service and can testify to its quality and reliability.

Software Support: Allegro Consultants

By Neil Armstrong, Robelle

About the author: Neil P. Armstrong (not the astronaut) came to work for Robelle in 1992, starting in technical support, then switching to R&D and working his way up to be the person in charge of Suprtool. Neil has also worked closely with Allegro on several projects, which is why he wrote this profile.

Since fall 2001, Robelle has been training Allegro Consultants of San Jose, California (www.allegro.com) as a technical support resource for our customers. We are very pleased with the results. Now they are handling all the initial support contacts, while our in-house programming team is, as always, still available as backup for difficult and perplexing issues.

Allegro has the strongest team of technical expertise in the HP 3000 area, plus years of experience in HP-UX as well. This is the reason we selected them to assist us. With the prospect of declining volume in the future, it made sense to outsource the first line of support to a firm that already has an MPE/HP-UX staff in place, already knows our products from using them, and already provides technical support for other firms. By sharing these great people, we can ensure economical and knowledgeable support for our customers for years ahead.

Allegro currently provides support for a myriad of companies and products. Not only do they provide support for Robelle and all of our products, but they also support Express and Librarian from Tidal, SAFE/3000 from Monterey software, NFS/iX from Quest AND GCC, the Gnu Compiler Collection.

Allegro are the authors of, and provide backend support for, DeFrag/X and the various Toolboxes from Lund. They also sell, distrubute and maintain their own products, such as SPLash!, HourGlass and DiskPerf. Not to mention, they provide and maintain one of the largest collections of freeware outside of Interex, with dozens of helpful utilities on MPE, HP-UX, Linux and even AIX.

Allegro is currently working on their latest product, "Rosetta Store", which is being developed and marketed in conjunction with Orbit Software. Rosetta is a ":RESTORE" program that runs on non-MPE systems and allows the user to extract their files and databases from HP and 3rd-party MPE backup tapes. For example it can restore an MPE TurboImage database onto HP-UX or x86 Linux directly into an Eloquence database or into SQL, CSV, XML, text, binary, and RAW format files. Rosetta is designed to help people who are migrating data from MPE to new platforms and as a tool to get at archival data on old MPE backup tapes without the need for an HP 3000 system. Not only can it extract the files from tape, but it presents the extracted data in a form that is immediately useful on the new platform.

Allegro has the internal skills to support MPE and HP-UX as well. They are the Software Escalation Center for ICS in the US, as well as Eurodata in Canada. They also support a number of 3000 end-user sites directly for system software issues. They always have someone readily available and have the ability to read dump tapes.

One nice experience I have had with Allegro is that when they find third-party software running in privileged mode as the root cause for a system abort, they keep reading through the dump to help isolate the problem. Allegro has excellent relationships with many third-party software companies and have certainly helped many to diagnose and fix some of their more complex problems. In fact, Allegro often partners with these companies, taking over or assisting R&D efforts at companies like Adager, TidalSoft, HP and even Robelle.

Allegro is truly a customer-based and customer-focused company. In fact, the decision to take on NFS/iX from Quest and SAFE/3000 from Monterey Software was based on providing support for the customers after the original providers had announced the were no longer supporting the products.

A further example of their commitment to customers was that after the November 2001 announcement from HP, many of Allegro's customers voiced that they did not want to hear how to leave the 3000, but rather how they could stay on the 3000. With this information, Allegro committed to supporting their customers through to 2011. So if your intention is to stay on MPE, Allegro is a good firm to connect with.

If your choice is to leave the 3000, Allegro certainly has the expertise in migrating to Linux, HP-UX and AIX.

The strength of any software company lies in the people who work there. Allegro has one of the strongest technical teams assembled, as well as some excellent, and patient, front-line technical support people.

Start with Barry "Mr. Patient" Durand. He is the front-line support person for the Robelle products and a long time MPE and Robelle user. And, he is backed up by Barry Lake, providing decades of combined MPE experience as end users in real production environments.

This is not to mention the back-end developers and key gurus at Allegro, Gavin Scott, Steve Cooper and Stan Sieler, who frequently contribute to HP3000-L and to the 3000 community at large, and have worked with and on many key pieces of software in the 3000 community including TurboIMAGE.

The office is run by Cindy Scott, who answers the phone in her friendly manner and keeps the fridge stocked with various caffeinated products, ice cream bars and my personal favorite - Red Vines.

All in all, Allegro is one of the key HP 3000 software companies and this is why we have chosen them to help with our front-line support and why we continue to work with them on various R&D projects.

Software Support: Beechglen

"Tending a Steady Light for Support"

From *The 3000 NewsWire*

Mike Hornsby has a longer view of the 3000's future than HP does — partly because his company has a clear look at the computer's past. Hornsby heads up Beechglen, a company dedicated to support HP 3000 customers, regardless of the vintage of their computers (www.beechglen.com). Beechglen doesn't mind who calls for support, or what version of MPE the customer's got in place. Its "answer any question from anyone, anytime" approach even extends to software and hardware outside of the HP 3000 realm, a remarkable open door approach to support that Hornsby says his clients have respected, despite the potential for abuse.

It's also an approach that springs from Hornsby's roots as a Hewlett-Packard systems engineer. He served HP 3000 customers in an era when in-person support was commonplace, and HP SEs called to see how you were doing, not just to resolve problems. He specialized in HP 3000 networking and performance as a Senior Systems Engineer in HP's Cincinnati, Ohio office, then founded Beechglen with the wife Mary Jo in 1988. He said he left HP because the customer contact he enjoyed most was being transferred to HP's telephone Response Center. More than 13 years later Beechglen has a group of HP experts who team-solve problems when needed, sporting a reputation for being able to answer almost any 3000 question. The support keeps enterprise-level sites running on software and hardware which HP has long ago given up on.

It's that last feat which brings Hornsby to our interview space, after HP's announcement that it's leaving the 3000 business by the end of 2006. Customers who don't march in lockstep with HP's recommendations might be wondering who can serve their support needs beyond that date, or even now at a better value. Beechglen is just one of many companies making that pledge, but it's one of the largest with a deep 3000 background. We asked Hornsby how much of the 3000 customer base Beechglen's experts might be able to cover now that HP is conceding the territory, and how he views the prospects for continuing to use the 3000 and get it supported, too.

How did you get the idea to provide an alternative to HP's support for 3000 customers?

Beechglen evolved into support services after we partnered with Kelly Computer Systems. At the time they were the leading sellers of non-HP add-on memory for HP 3000 systems. We would provide performance tuning and analysis to those sites where the additional memory alone did not provide sufficient performance gains. We became the path of least resistance for support questions for these customers, and we worked out an ongoing relationship enhancing or replacing the phone-in support from HP. The support philosophy was always to "take care of the customer" by providing a "phone a friend" for any software related issues.

How many are you helping now?

We provide assistance for the entire HP 3000 community in a number of ways. We have a direct telephone support service, but also provide free assistance to anyone via the 3000-L mailing list and newsgroup, other miscellaneous newsgroups, the Interex AnswerLine, and even at HP's ITRC forum on the Web.

Are you willing to ramp up to support a lot more customers?

We have been in an active growth state for the past several years. Y2K conversions and testing created many support opportunities, and since then, many more HP 3000 and HP 9000 opportunities have developed.

What do you consider to be the special experience that Beechglen brings to this kind of work?

Our goal is to take care of the customer, as if we were them, knowing what we know. Our motto is 'anyone can call about anything anytime." From the beginning we have realized that it is much easier to answer a question at 2:00 AM than to cleanup after a guess at 8:00 AM. Our support is immediate, like having a 'phone a friend' who can answer the question or solve the problem more than 95 percent of the time on the first call.

For example: A customer calls asking how to stop his laser printer from switching paper trays when the first tray is empty, because the second tray is letterhead. Our consultative answer was, here is the escape sequence to label the trays so that the printer knows what is in each tray. But, it is also possible to print letterhead from the HP 3000 at no additional cost using environment files. The elimination of preprinted forms saved them the printing cost and the operational overhead of stocking and switching the forms.

Another example was a customer who was frustrated at having to dial in every night to check in on a credit card upload job. We showed her how to use a freeware e-mail client from Telemon to e-mail her pager if the job failed. The special experience is that we take care of our customers as if they were family.

People get concerned about the availability of hardware in the years to come. Has it been your experience that HP's parts supply is essential to repairing systems?

Certainly parts availability and staging is key to providing 4-hour onsite repair. Fortunately there are many excellent third-party service providers available. In our experience they are just as good and in many cases better than HP's service. In just about all cases this is their main business, and they are totally focused on parts and service. Many of our mission critical 'always on' customers also have hot backup systems. With HP's 11/14 announcement this will become more feasible for many more HP 3000 sites.

What about patches? HP probably won't be writing many within a few years. How could you provide support for a customer who needed some part of MPE patched to solve a problem, if HP's not patching them anymore?

First let me say that I have always been skeptical about patches. To paraphrase a famous coach, "three things can happen when you patch, and two of them are bad." Our approach has always been to be more consultative and look at the problem from a programming perspective. Most of the time people run into bugs because they are off the beaten path, and trying to do something the hard way. We steer them back and find a better alternative. This approach minimizes down time and risk of introducing newer more serious problems.

We have also created some of our own patches. We had several customers who required an extended security log in prompt, so we created LPRESET. Some good information on these types of things can be found in an article I wrote for Interact, "MPE Debug and Dump Analysis."

We keep hearing stories about how Beechglen is more responsive than HP's support. What kinds of service response levels do you offer, and can you compare them to HP's?

The only type of support service we provide is 24x7x365, so that "anyone can call about anything anytime." Our pricing is still significantly less expensive than HP's 8-5 coverage.

HP's phone-in support is limited to very specific, fairly closed-ended questions, related only to HP products, and was conceived and designed at a time when the average system manager was more of a specialist on the HP 3000. Today, the average system manager is more of a 'Jack or Jill of all trades,' part time system administrator, part time network administrator, and part time PC help center. They have many items in their 'to do today' list and most of them are numbered 1.

We help to offload some of those tasks, take the ball and run with it, reporting back on the results. This is especially effective for the kinds of tasks that are only done once in a great while, like adding a network printer or disc array. It might take a system administrator several hours to complete this task, whereas we can access the system remotely, and accomplish the same task in minutes.

Is there really any further reason for a 9x7 owner to be paying for HP's software support on their HP 3000 systems, now that 6.0 is being removed from HP's support in October?

No, for several reasons that may also apply to other HP 3000 models. First, software support is so expensive, so that for many models you could buy an entire matching system with a current license for less than one year of software support service.

Second, at this point it would be a mistake to take a system that is working and update it for the sake of HP's version support timetable. I would strongly recommend a strategy of postponing updates until a specific business need requires it. Finally, 6.5 and 7.0 have features which extend the high-end of the HP 3000 at the cost of degraded performance for the low-end systems.

HP wants to portray the 3000 customer base choosing to Homestead as less strategic about the way they use their 3000s. Is that your experience among your customer base — that the 3000s aren't the most important in those IT shops?

If I understand your question as: HP wants the HP 3000 base to move to other HP platforms ASAP, and those that don't share this view are somehow strategically defective? Then I have to disagree with that. We support both HP 3000 and HP 9000 systems and I personally made over 10,000 calls to these system administrators in the past year. I would estimate that less than 5 percent of the remaining HP 3000 installed base would even consider another HP platform.

Another way of saying this is: If you had a 10-year-old car that was paid for and was extremely reliable, would you trade it in just because that model was no longer going to be sold new? No, because the new car can't be more reliable — and for the expense you won't get any marginal benefits.

Reading strategically between the lines of HP's 11/14 announcement, one item seems very significant. Hewlett-Packard has basically admitted that they could not justify the effort to convert MPE/iX, IMAGE, and various subsystems to run on IA-64. This does not bode well for other complex operating systems, DBMS, and application packages, attempting the same "migration."

My recommendation is to ride out the wave of the coming 64-bit storm, and see where the chips eventually fall. Meanwhile, your current HP 3000 investment will continue to pay reliable dividends. Otherwise, customers could be faced with an expensive initial parallel investment, immediately followed by an equally expensive 64-bit upgrade. Another way of putting it would be that it would be a shame to jump out of the frying pan into the fire.

Would you say that HP's continued support for IMAGE is more essential than its 3000 support in general? Or would you recommend that customers start looking at moving their IMAGE data into other databases? How can third-parties step into support for the 3000's database?

The secret to the success of the HP 3000 has been that IMAGE and MPE were integrated together. This provided the ability to obtain the reliability and performance standards the HP 3000 attained. So in my view the two are inseparable.

I think you'll see the remaining installed base split into six categories:

1) Those with application packages that have financial resources to switch to a different package: These will switch gradually through 2006 and the switch will take a relatively short time.

2) Those that have complex custom code that have financial resources to switch to an application package: These will switch to application packages through

2006 and the switch may take 3-5 years. Many HP 3000 sites are already in this phase.

3) Those that have simple custom code and budget to convert to different platform: These will wait as long as possible for a stable alternative.

4) Those that have no budget: These will also wait as long as possible to choose one of the three alternatives above.

5) Those that have extremely complex applications that were evolved over 20 years in many different generations of programming languages: These will take years just to document what the existing system does, and at least another 10 years to accomplish a complete migration.

6) Those already in archival mode. The HP 3000 is for lookup and reference purposes due to statute or policy regulations. Overlooked is the fact that many HP 3000 applications will by law be running well past the end of 2006.

If you could only get one more significant IMAGE enhancement out the CSY labs, what would you hope they would do?

I would like to see many improvements to IMAGE, unfortunately there is not enough time till December of 2003 for any enhancements to make it into a release, then to be installed on enough systems, to get enough patches to make the enhancements recommendable for a production system

What's your outlook on the prospects of a company running home-grown applications on a 3000 to keep their apps on MPE, given the Nov. 14 announcements? Is there enough help out there in the 3000 community to let them stay successful with the 3000?

Definitely. I think you'll see a major percentage of the current HP 3000 base still in production usage well after 2012. We still have several customers running MPE/V on a wide variety of those models, some that have been off of HP support for more than 10 years.

Hardware Suppport, Upgrades and Spare Parts

By Bob Green, Robelle

About the author: Bob Green is not a hardware expert. In fact he prides himself on sticking to software. But he is a company manager, and he has been responsible for buying equipment. When Bob noticed that other sites were using 1 or 2 Gigabytes of main memory while his system still had 128MB, he encouraged his staff to explore buying extra 3000 memory on Ebay!

Hewlett-Packard is not the only place where you can get HP 3000 hardware, or extra memory, or spare parts, or hardware support. Independents have always existed and provided a reliable alternative.

Need Another 3000?

New HP 3000s may not be available, but many used 3000s will come onto the market over the next 10 years and the prices should be attractive. At Robelle, when we cancelled the lease on our 3000 server, it turned out to be cheaper to buy it and keep it as a spare than to pay the shipping back to HP. In fact, it was cheap enough that it was worth it just for the spare DDS/DAT drive!

The high-profile used 3000 broker is **Phoenix Systems**, a division of Client Systems in Denver, the 3000 distributor for North America.

```
http://www.phoenix3000.com/
```

Here is how they describe themselves: "Endorsed By HP Manufacturing, Only HP integrated/supported components used, Fully tested and guaranteed equipment, Customized configuration, HP certified technical and sales support staff, HP licensed MPE/iX operating software, Full range of HP support with no waiting period for certification, …" So they are the gold-plated source of used 3000s. Their web site does not show current inventory of systems or prices, but call them at 1-800-330-0568 or (303) 614-8221.

The OpenMPE web site has a list of brokers, with 15 names on it at the moment:

```
http://www.openmpe.org/mpebokers.htm
```

And Triolet lists 50+ vendors under "Hardware - Buy, Sell, Lease":

```
http://www.triolet.com/HPVend/hpvend.html
```

For example, Cypress Technology lists 9 refurbished HP 3000s for sale at prices ranging from $500 to $25,000 and gives the End-of-HP-support date for each.

```
http://www.cypress-tech.com/
```

Another interesting web site is Hp Trader Online, which has classified ads for 3000s and a reverse auction where you list your requirements and wait for brokers to bid for your business:

```
http://www.hptraderonline.com
```

More Memory?

When we wanted to improve the performance of our Admin HP 3000 at Robelle, we suspected that it would benefit from more memory. Paul Gobes (Manager, Tech Support, now at WestJet) researched our memory usage and found our memory upgrade. Paul started by measuring the memory situation using a free utility called **Ramusage**.

Ramusage is a small, free MPE program that reports your current memory and determines how much is used by the system and how much is available to users. It then calculates the effect of adding various amounts of memory.

For the production system at Robelle, which had 112 Mb of memory, Ramusage showed that adding only 64Mb of memory would double the amount of memory available to users. Here's the actual memory report:

```
RAMUSAGE [2.47] - LPS Toolbox [A.09b]
(c) 1995 Lund Performance Solutions

SERIES 928LX
MPE/iX 6.0
#CPUS: 1
Memory size: 112 MB (117,440,512 bytes; 28,672 4KB pages)

Memory usage by "type" of Object Class:

Class       #LogicalPages       #MB     % total

SYSTEM_CODE 3,406           13  11.9%
SYSTEM_DATA 10,443          40  36.4%
TURBO_DATA  2,923           11   10.2%
USER_CODE   4,234           16   14.8%
USER_DATA   2,882           11  10.1%
USER_STACK  1,548           6    5.4%
USER_FILE   3,235           12  11.3%
Totals:         28,671           111   100.0%

"User" pages are 51.7% of memory (58 Mb out of 112 Mb)

If you added:
32 Mb, you'd have 1.6 times as much "User" memory. ( 144 total Mb)
64 Mb, you'd have 2.1 times as much "User" memory. ( 176 total Mb)
96 Mb, you'd have 2.7 times as much "User" memory. ( 208 total Mb)
 . . . <some lines removed> . . .
```

Installing Ramusage is easy — it takes about 10-15 minutes. You can download it from Allegro's Web site at

```
http://www.allegro.com/software/hp3000/allegro.html#RAMUSAGE
```

There are three downloadable formats: :LZW, Store-to-disk (STD) and tar.Z with instructions for each, but you must set up the Allegro account first:

```
:hello manager.sys
:newacct allegro, mgr; cap = &
AM,AL,GL,DI,CV,UV,LG,PS,NA,NM,CS,ND,SF,BA,IA,PM,MR,DS,PH
:altacct allegro;access=(r,x,l:any;w,a:ac)
:altgroup pub.allegro; cap = BA,IA,PM,MR,DS,PH; &
access=(r,x,l:any;w,a,s:ac)
:newgroup data.allegro; access=(r,x,l:any;w,a,s:ac)
:newgroup doc.allegro; access=(r,x,l:any;w,a,s:ac)
:newgroup help.allegro; access=(r,x,l:any;w,a,s:ac)
```

We chose the STD method and followed the sample found on the same Web page.

After downloading to a Windows PC, we uploaded to our 3000 using FTP. If you have never enabled FTP on your 3000, it is easy. Just follow the instructions on this web page:

```
http://www.robelle.com/tips/ftp.html
```

Here are the FTP commands that will upload the "STD" copy of Ramusage from the PC:

```
C:\>ftp my3000
User (my3000.robelle.com:(none)): paul,manager.sys,pub
ftp> binary
ftp> put ramusage.std ramusage;rec=128,1,f,binary;code=2501
ftp> quit
```

Then on the 3000, you restore the files from the uploaded STD file:

```
:hello paul,manager.sys,pub
:file ramusage;dev=disc
:restore *ramusage; @.@.@; olddate; create
```

Finally, run the Ramusage program:

```
:run ramusage.pub.allegro
```

The Ramusage program, written by the legendary Stan Sieler, is a subset of the PAGES utility (part of the Lund Performance Solutions "System Manager's Toolbox"). When asked how to calculate the amount of memory to buy, Stan provided the following formula:

Money_available / price_per_MB = Amount_of_MB_to_buy

In other words, *buy as much as you can afford!*

Now where do you buy more memory?

Well, you can get it from HP if you know the part number... and you can find the right page on their web site.

Or you can try one of the third-party vendors who are listed under 'Memory' at the comprehensive vendor listing at

```
http://www.triolet.com/HPVend/hpvend.html
```

At that time, Kingston listed the 64Mb for our 928LX at $286, whereas HP was charging $1,090 (all prices are in US dollars). See the Kingston Web site at

```
http://shop.kingston.com/
```

Select 'Hewlett-Packard' from the pull-down list, then select your HP 3000 model number from the page that is displayed. At the time this book went to press in 2003, 512MB for a 9x9 was $665, and Kingston tells you the maximum 9x9 memory (3.2 Gb), how many sockets there are (32 in 16 banks of 2), and shows how the banks are arranged.

Another source might be third-party hardware maintainers like the ICS Group which had 128Mb for $600 (versus $572 for 128Mb at Kingston).

```
http://www.icsgroup.com
```

Finally, if you're really lucky (and brave), there's eBay.

```
http://www.ebay.com
```

The day that Paul looked for our memory, there were offers for 9x9 systems, so he contacted one of them (Sales@Hprecovery.com) and was quoted 128Mb of used memory for $95 (versus $572)! At that price it was worth the risk.

However you buy memory, it will probably take a small amount of research, because availability may be an issue. But it is also the cheapest and easiest way to boost your system performance.

Do It Yourself

When I asked Paul Gobes (pgobes@shaw.ca) what he would tell people about HP 3000 hardware without HP to hold his hand, here is what he said:

> "The main advice is that it's time to get out of the mindset that you always need to buy from HP and that if it comes from somewhere else it won't work. This is not the case. HP hardware is usually excellent and lasts way longer than most of our needs. The single exception here is probably their DDS/DAT drives."
>
> "Once you've tried buying 3rd party, experienced the huge savings, and had a few good dealings, you're more likely to look around."

"At Robelle we tried alternate sources on a variety of items. We bought HP memory chips, HP JetDirect boxes and even small routers right off Ebay. We found great deals on non-hp 72Gig SCSI disc drives from Pricewatch:

```
http://www.pricewatch.com/
```

In this case we did have problems with the hardware, but the vendor replaced it within a week. We also got help from other HP regulars like Allegro in tracking down sources when we needed to assemble components."

"So when your favorite big computer company stops supporting you, it's time to find out that others will support you."

Paul mentioned DDS/DAT drives and Disk drives as issues. Please read the next chapter in this book, "Connecting Modern SCSI Devices to the 3000" on page 64.

And since Paul also mentioned Allegro, I asked Stan Sieler of Allegro about used hardware and spare parts. Stan says:

"We have several HP 3000s and HP 9000s that were purchased either from eBay or surplus auctions. My comments fall into two categories: obtaining equipment and what to do once obtained.

1. Obtaining equipment
   - EBay (search for "hp 3000" and "hp3000" and "hp e3000")

     Pay attention to seller's feedback rating, location, and method of payment before bidding. (I no longer bid if the seller won't take PayPal.)

     Ask about condition, boards in the machine, etc.

     Ask about shipping cost.
   - Local surplus houses

     Inspect machine
2. Once obtained
   - Open system remove cards, vacuum, re-insert cards
   - Erase the old disk drives using WipeDisk:
     ```
     http://www.allegro.com/products/hp3000/wipedisk.html
     ```
   - Test memory, CPU, peripherals. (The online/offline diagnostics can be useful here, if you are able to run them.)
   - Remember: 9x7 computers take significantly longer to boot up as you increase the amount of memory in them."

Hardware Support

Now that you know you can obtain hardware for your HP 3000 shop for the foreseeable future, what about servicing that hardware? Of course, some people do first-line support themselves, keeping a spare system and swapping boards, but perhaps you need more insurance than that!

The OpenMPE web site list 14 firms offering hardware support for the 3000:

```
http://www.openmpe.org/mpehardware.htm
```

And Triolet lists 25 firms under "Hardware Maintenance":

```
http://www.triolet.com/HPVend/hpvend213.html#HRDMTC
```

For example, consider these three firms:

Atlantic Tech Services

```
http://atserve.com/
```

David Mason (dmason@atserve.com) points out that Atlantic Tech Services (ATS) is a hardware maintenance company and has been servicing the 3000 line since 1984 and will continue to do so.

ATS is located in NE Ohio, between Akron and Cleveland. It started as a regional maintenance company, but has grown into a multi-office company with coverage extending throughout the United States.

ICS Group

```
http://www.icsgroup.com/maintenance.htm
```

The ICS Group continues to provide independent maintenance for HP 3000 systems: "We will continue to provide hardware support for our beloved 3000 and partner with the gurus at Allegro to provide software support."

Terix Computer

```
http://www.terixcomputer.com/
```

Terix offers 24x7 hardware coverage for HP 3000 systems throughout the USA. They have partnered with IBM Global Services to provide on-site HP 3000 maintenance services.

Connecting Modern SCSI Devices to the 3000

By John Burke

THE 3000 NEWSWIRE

About the author: John Burke is one of the most distinguished contributors to the HP 3000 world. He writes the net.digest and Hidden Value columns monthly for ***The 3000 NewsWire*** *and this great article first appeared in the April 2003 issue of the magazine as "SCSI is SCSI: How to extend the life of your HP 3000 with modern, non-HP peripherals". It is Copyright 2003 by John Burke*

Here is an example of the type of question that prompted this article: "We have an old HP6000 SCSI SE disk rack. It has been running 4 2gig drives for a long time. One of the drives recently failed and we are looking at our options for not only replacing the lost storage but also increasing our available disk space. We are a small company so we are looking for relatively inexpensive options. Does anyone know if there is a larger drive that will fit into that specific rack? Failing that, is there any way we can get a single 9GB or 18GB HP drive to replace the whole rack?"

This article will address two issues raised by the above question, show why they are false, and examine some options that should help you run your HP 3000 for years to come. The first issue: you need to use only HP-branded storage peripherals. The second issue: because you have an old (say 9x7, 9x8 or even 9x9) system you are stuck using both old technology and just plain old peripherals. Both are urban legends and both are demonstrably false.

Special thanks go to Denys Beauchemin who contributed significant material to this article. Also contributing were Stan Sieler, Steve Dirickson, Wirt Atmar and Gavin Scott.

There is Nothing Magical about HP-Branded Peripherals

Back in the dark ages when many of us got our first exposure to MPE and the HP 3000, when HP actually made disk drives, there was a reason for purchasing an HP disk drive, "sector atomicity". 9x7s and earlier HP 3000s had a battery that maintained the state of memory for a limited time after loss of power. In my experience, this was usually between 30 minutes and an hour. These systems, however, also depended on special firmware in HP-made HP-IB and SCSI drives (sector atomicity) to ensure data integrity during a power loss. If power was restored within the life of the internal battery, the system started right back up where it left off, issuing a "Recover from Powerfail" message with no loss of data. It made for a great demo.

These earlier systems used so much power that the best you could hope for with a reasonable-sized battery was to keep the memory alive. In a happy set of coincidences for HP, just as it was contemplating getting out of the disk drive manufacturing business, CSY was developing the 9x8 line of systems that required less power. A small UPS could keep everything including the disk drives running in these newer systems. This seemed superior to just saving the state of memory. It also meant CSY could use industry standard disk drives, thus reducing the system cost and helping HP make the decision to exit the disk drive market.

Ah, but you say all your disk drives have an HP label on them? Don't be fooled by labels. Someone else, usually Seagate, made them. HP may in some cases add firmware to the drives so they work with certain HP diagnostics, but other than that, they are plain old industry standard drives. Which means that if you are willing to forego HP diagnostics, you can purchase and use plain old industry standard disk drives and other peripherals with your HP 3000 system. [For information about the naming scheme of Seagate drives go to

```
http://www.seagate.com/support/kb/disc/howto/interpret_model.html
```

So what, you say? Well, have you tried to buy a 4GB or 9GB 50-pin HP branded disk drive for your 9x7, 9x8 or 9x9 lately? HP does not sell them anymore. If you can even find one, a 4GB SE or FWD 50-pin HP branded disk drive will likely run you at least $500 and a 9GB SE or FWD 50-pin HP branded disk drive will likely run you at least $800. Oh, and by the way, no one is making new FWD drives anymore of any size.

What about tape drives? HP DDS2 and earlier drives, except for internal ones needed to boot the system, have long since gone off the support list.

SCSI is SCSI and You Can Connect Just About Anything to Anything

SCSI stands for Small Computer System Interface. It comes in a variety of flavors with a bewildering set of names attached such as SCSI-2, SCSI-3, SE-SCSI, FW-SCSI, HVD, LVD SCSI, Ultra SCSI, Ultra2 SCSI, Ultra3 SCSI, Ultra4 SCSI, Ultra-160, Ultra-320, etc. Pretty intimidating stuff. [For more than you ever wanted to know about the various SCSI definitions see http://www.paralan.com/glos.html or http://www.adaptec.com or, for a general index of sites with SCSI information, see http://www.scsi-planet.com/scsi-definitions/.] Don't despair though, for as Steve Dirickson, among others, have said "pretty much any kind of SCSI device can be connected to any other with the appropriate intermediary hardware." Various high quality adaptors and cables can be obtained from Paralan (www.paralan.com) or Granite Digital (www.granitedigital.com).

So, SCSI really is SCSI. It is a well-known, well-understood, evolving standard that makes it very easy to integrate and use all sorts of similar devices. MPE and the HP 3000 are rather behind the times, however, in supporting specific SCSI standards. Until recently, only SE-SCSI (SCSI-2) and FW-SCSI (HVD) were supported. Support for LVD SCSI was added with the A- and N-class systems and, with MPE/iX 7.5, these same systems now support Fibre Channel (FC). Also, MPE only supports a few types of SCSI devices such as disk, certain disk arrays, DDS, DLT and a printer.

Let's concentrate on the SE-SCSI and FW-SCSI interfaces, both seemingly older than dirt and disk and tape storage devices. But first, suppose you replace an old drive in your system, where should you put it? The 9x7s, 9x8s and 9x9s all have internal drive cages of varying sizes. It is tempting to fill up these bays with newer drives and, if space is at a critical premium, go ahead. However, if you can, heed the words of Gavin Scott:

```
"I'd recommend putting the new drive(s) in an external case rather than inside
the system, since that gives you much more flexibility and eliminates any
hassles associated with installing the drive inside the cabinet. It's the same
SCSI interface that you'd be plugging into, so apart from saving the money for
the case and cable, there's no functional difference. With the external case
you can control the power of the drive separately, watch the blinking lights,
move the drive from system to system (especially useful if you set it up as
its own volume set), etc."
```

At sites such as Granite Digital you can buy any number of rack mount, desktop and tower enclosures for disk systems. Here is another urban legend; LDEV 1 must be an internal drive. False. Or, the boot tape device has to be internal. False. You cannot tell by the path whether a drive is internal or external, and the path is the only thing MPE knows (or cares) about the physical location of the drive.

Once you come to terms with the fact that you can use almost any SCSI disk drive in your HP 3000, dealing with SE SCSI is a piece of cake and a whole world of possibilities opens up. With the right cable or adapter (see Paralan or Granite Digital) you are in business. A good Internet source for disk drives is www.dirtcheapdrives.com. But just because you can connect the latest LVD drive to your SE-SCSI adaptor, should you? Probably not, because you are still limited by the speed of the SE adaptor and so are just wasting your money. Now that you know you do not need the specific HP drives you once bought, you can pick up used or surplus drives ridiculously cheap (I once bought a 9GB SE SCSI drive in its own enclosure with new power supply and fan for $54 at www.compgeeks.com.). However, you are still dealing with old drives and old technology. Denys Beauchemin discovered that Seagate is now making some new technology drives with the old technology 50-pin SE-SCSI interface, the 18GB model ST318418N and the 36GB model ST336918N. A quick Internet search showed a price as low as $215 for the 36GB bare drive. Dirtcheapdrives.com has the 18GB model in a standalone enclosure for $239, a perfect solution to replace the 4x2 HP6000 in the question that started us off.

FW-SCSI is more problematic than SE-SCSI because no one even makes FW-SCSI (HVD) disk drives any more and you need more than just a simple cable or adapter to connect newer drives to an HVD adaptor. In fact, from the Paralan site, "HVD SCSI was rendered obsolete in the SPI-3 document of SCSI-3." So, what is one to do? Most systems with FW-SCSI adaptors need them for the increased throughput and capacity they provide over SE-SCSI. Paralan (www.paralan.com) and others make HVD-LVD converters. The Paralan MH17 is a standalone converter that allows you to connect a string of LVD disk drives to an HP FW-SCSI adaptor. Pretty cool.

But suppose your organization is moving to a Fibre Channel (FC) SAN environment and you would like to store your HP 3000 data on the SAN. Only the PCI-Bus A- and N-class systems (under MPE/iX 7.5) support native Fibre Channel. Denys Beauchemin discovered that Crossroads Systems (www.crossroads.com) now makes a storage router device (ServerAttach) for (ugh) legacy systems that allows you to connect from one to four FW-SCSI adaptors to up to two Fibre Channel (FC) resources (i.e. SAN). To the HP 3000, each LUN simply looks like a normal attached disk drive – the ServerAttach box, configurable via a web interface, handles all the really hard stuff. Way cool.

For tape storage systems, let me quote Denys. "As for tape drives, this is more problematic. First, get off DDS and get to DLT (4000/7000/8000) with DLTIV tapes. Whatever connectivity problems there are can be dealt with just like the disk drives. If you have a PCI machine, LTO or SuperDLT is the way of the future and they both require LVD connections. If you have a non-PCI machine, anything faster that a DLT8000 is wasted anyway because of the architecture." Note that while, as Denys says, LTO and SuperDLT are the "way of the future", HP has not committed to supporting either on the HP 3000. SuperDLT should probably work on A- and N-class systems using the existing DLT driver; however, I do not know anyone who has tried. LTO drives will require a new driver for the HP 3000 so we should probably not hold our breath on that one.

A quick word about configuring your new storage peripherals, do not get confused by the seemingly endless list of peripherals in IODFAULT.PUB.SYS. And, do not worry if your particular disk or tape drive is not listed in IODFAULT.PUB.SYS. Part of the SCSI standard allows for the interrogation of the device for such things as ID, size, etc. DSTAT ALL shows the disk ID returned by the drive, not what you entered in SYSGEN. When configuring in a new drive(s), just use an ID that is close. In fact, there is really no need for any more than two entries for disk drives in IODFAULT, one for SE drives and one for HVD drives so as to automatically configure in the correct driver. The same is true for tape drives.

Summary

If you plan to, or even think you might homestead or think there is even the slightest possibility you will still be using your HP 3000 for some years to come, you need to start thinking about your storage peripherals. Now. Disk drives and tape drives are the devices most likely to fail in your HP 3000 system. The good news is that you do not need to be stuck using old technology, nor are you limited to HP only peripherals. Here is a summary of what we recommend you do.

If you can, trade in your older machines for the A-class or N-class models. Yes, the A-class and some N-class systems suffer from CPU throttling. (HP's term. Some outside HP prefer CPU crippling.) However, even with the CPU throttling, most users will see significant improvement simply by moving to the A-class or N-class. [In the recent System Improvement Ballot (SIB), removing CPU throttling was far and away the number one item in the voting. However, HP has given no indication it will do anything about this so it would be unwise to purchase a low-end A- or N-class system in the hopes that you will get a "free" performance boost sometime in the future.] Both the A-class and N-class systems use the PCI bus. PCI cards are available for the A- and N-class for SE-SCSI, FW-SCSI and Ultra-3 SCSI (LVD). You can slap in any drive manufactured today; by anybody. Furthermore, with MPE/iX 7.5, PCI fiber channel adaptors are also supported, further expanding your choices. An A- or N-class system bought today should be going strong 10 years from now.

If for whatever reason you are going to homestead on the older systems or expect to use the older systems for a number of years to come, you have several options for storage solutions. For your SE-SCSI adaptors, you can use the new technology, old interface 18GB and 36GB Seagate drives. For your FW-SCSI (HVD) adaptors, since no one makes HVD drives anymore, you have to use a conversion solution. [You could of course replace your FW-SCSI adaptors with SE-SCSI adaptors, but this would reduce capacity and throughput.] One possibility is to use an LVD-HVD converter and hang a string of new LVD drives off each of your FW-SCSI adaptors. The other possibility is the Crossroads Systems router that allows you to connect from one to four FW-SCSI adaptors to up to two Fibre Channel (FC) resources (i.e. SAN). In all cases, get rid of those dusty old HP 6000 enclosures, disasters just waiting to happen.

As for tape drives, forget DDS and use DLT (4000/7000/8000) with DLTIV tapes. Whatever connectivity problems there are can be dealt with just like the disk drives. If you have a PCI machine, LTO or SuperDLT is the way of the future and they both use LVD connections, though whether HP will support either for the HP 3000 remains an open question.

The bottom line is you have numerous options to satisfy your HP 3000 storage needs, both now and into the future.

Disagreement From Hewlett-Packard

In reply to this chapter on SCSI peripherals, Jim Hawkins of HP wrote a column titled "Non-HP disks can prompt risks" in the July 2003 issue of *The 3000 Newswire.* His argument is that HP has found many SCSI disk drives to be unusable in the multi-user server environment of MPE, either because they don't fully comply with the SCSI standard or they don't comply with additional requirements that HP makes of suppliers:

"Before HP receives a device from a manufacturer, we provide to them a 'supplementary' HP Parallel SCSI Device Specificiton" (this document is only available to vendors supplying product to HP, so please don't request this document). This specification clearly states which of the myriad SCSI features HP server systems really, truly depend upon.

It seems that this manual is another one that should go on the wish list of OpenMPE for documentation that HP should distribute before 2006.

Jim Hawkins is Lead I/O Engineer in the HP MPE/iX R&D lab and has worked for HP for over 17 years in both the R&D lab and the support organization.

Homesteading By Tax Regulation

by Marius Schild, SAMCO

About the Author: Marius Schild,(www.samco.nl), the Robelle dealer in Holland, made us aware that in the EU, you must not only retain your business records for a tax audit, but your online computer data as well. This legal requirement may mean that many European companies will be "homesteading" with their HP 3000s more than they had planned.

"Your automated administration and the fiscal storage obligation"

After a business acquaintance pointed out the fiscal storage obligation to me, I received a brochure from the Dutch Inland Revenue with the title stated above.

As a result of the flood of new tax rules, the magnitude of brochures finding their way to my desk from several government institutions and my never-ending work activities, this very drastic alteration of the law hadn't caught my attention yet.

It is a rather significant change: all documents (which isn't a new fact) and all data storage backups that are related to the tax assessment will have to be kept available in principle for 7 years.

"Data storage backup" is the key concept here! You can no longer make do with printouts of the yearend closing. In fact, because of this law alteration, you must keep all data available online for the Inland Revenue for when they are examining a closed fiscal year and want to examine previous years. And since it must be online, you must keep the original computer systems as well. Again, they want to be able to go back to the previous 7 fiscal years.

This has some rather serious consequences for your automation!

Unfortunately the Inland Revenue hasn't made clear specifications of the data that must be available and stored for those previous years.

Naturally, it concerns the general financial data. And the Inland Revenue does make a distinction between the general data and the miscellaneous data.

General data are mainly:

- The general ledger
- The accounts receivables and accounts payables
- The purchasing and sales administration
- The stock control
- The salary administration
- Possibly some sub ledgers like the fixed assets sub ledger

Each organization may be required to keep some other general data. For example, a construction company may be required to retain project administration or an auditing service might be required to retain order administration.

For determining which miscellaneous data have to be available you can make an agreement with the Inland Revenue. However it is rather unclear what they precisely mean by determining this and what the miscellaneous data exactly incorporates. Because of this, the Inland Revenue demands that when in doubt you contact their offices.

Archiving and storage of data

A Rule of thumb is that, with a tax audit, the data has to be accessible and controllable within a reasonable time frame.

Data has to be kept in its original form. For example, after posting the invoice, purchase order or delivery order, these documents may not be destroyed. This isn't a new request, you already had to archive all original documents.

The difference resulting from this tax change is, all data you posted as a result of the original documents into your automated administration has to be stored and accessible and the Inland Revenue has to be able to request and print that data online. This results into a heavy burden on your computer systems; reorganizing the data after closing the year-end is no longer allowed by the law!

A solution could be a modification of your applications in such a way that the historical data is kept in separate databases and that the programming for handling the future requests and printouts can work with current and historical data.

Furthermore, whenever you purchase a new computer system, install a new operating system and/or "upgrade" the operating system, you have to check whether all data of the previous 7 years is available online.

If because of one of the above stated changes to your system setup the data is no longer accessible online, the Inland Revenue expects you to have the replaced system readily available, namely the old computer system or operating system with the appropriate programs and the previous 7 years of general data.

Printouts of the data

The consequence of this alteration of the law is that in principle it is against the law to print out the data and then delete the records from the system. Only if the administration is of "small size", which means the audit of the printouts can be done in a reasonably short time, is it allowable to delete the records on your system.

The question remaining is 'What does the Inland Revenue consider a "small size"'?

Technical specifications

The Inland Revenue has set up rules and requirements to which, technically, the data backups and even the file formats must comply. The rules often concern very specific storage media and/or file formats.

That could be very troublesome with custom-made financial systems which satisfied the demands of your company but no longer satisfy the Inland Revenue because of this alteration of the tax law.

Again, when in doubt you are required to contact the Inland Revenue. Hopefully the Inland Revenue realizes they have to work with employees who, besides knowledge of the tax laws, also have ample knowledge of our profession.

One could wonder whether the Inland Revenue can create clarity, as they themselves don't seem to know the specifics of the changes only the broad strokes of the change.

If you need more information, send an email to marius_schild@interex.nl

Homesteading Checklist

By Paul Edwards of Paul Edwards & Associates

About the Author: Paul Edwards is a highly experienced computer systems consultant, who has managed, designed, programmed, installed, and documented many major projects of software and hardware systems. His certifications include HP3000 Channel Partner Sales Professional, HP3000 Technical Consultant, MPE System Administrator, and HP9000 Technical Pre-Sales Professional. Paul Edwards & Associates www.peassoc.com

[Editor: A company who will not migrate to another platform until sometime in the far future, if ever, needs to take some simple planning steps in order to properly face the future. In this short article, Paul Edwards gives a checklist of those steps.]

Choices – select one (or more):

- ❑ Rehost - Convert the Applications
- ❑ Replace - Purchase New Applications
- ❑ Rewrite - Develop New Applications
- ❑ Abandon - Don't Use Obsolete Systems
- ❑ Stay - Homestead on the HP 3000
- ❑ Outsourcing - Applications and/or System
- ❑ Use a Combination of the Above

Systems Manager Notebook

Every site should have a System Manager Notebook that contains critical information. It is part of the Disaster Recovery Plan and is used to manually recreate your information if necessary. But it also acts as a homesteading memory for a site with an HP 3000 in a corner and whose only employee who knows MPE has left.

You can't have too much information. Use the Gold book. Keep it current!

Notebook Contents – check items off as you gather them

- ❑ Hardware Model and Serial Numbers
- ❑ License Agreements for All Software and Hardware
- ❑ Copy of Current Maintenance Agreements
- ❑ Equipment Warranty Information
- ❑ Complete Applications Documentation
- ❑ Operator Documentation

- ❑ SYSINFO.PRVXL.TELESUP Complete Listing
- ❑ SYSGEN Listing of Paths, Devices, Logging, and Misc.
- ❑ Listing of HPSWINFO.PUB.SYS
- ❑ Listing of DSTAT ALL
- ❑ Listing of NPCONFIG
- ❑ Listing of NMMGR Files
- ❑ Listings of NET.SYS Files
- ❑ Listing of SIU
- ❑ Listing of PSSWINVP.PRED.SYS
- ❑ Listing of REPORT @.@
- ❑ Listing of BULDJOB1 and 2
- ❑ Backup $STDLIST
- ❑ 3rd Party Software Installation Codes
- ❑ HP and 3rd Party Contact Phone Numbers
- ❑ Hardware and Software Historical Records
- ❑ Preventative Maintenance Schedules and Instructions
- ❑ Terminal and PC Configurations
- ❑ Copy of the MPE/iX Software Maintenance Manual and Checklist
- ❑ Patch/iX and Stage/iX Documentation

Prime Resources an HP 3000 Site

• HP	www.hp.com
• Manuals	docs.hp.com
• The 3000 Newswire	www.3000newswire.com
• HP3000-L	listserv@raven.utc.edu
• Jazz	jazz.external.hp.com
• INTEREX	www.interex.org
• OpenMPE	www.openmpe.org
• 3rd Party Sites such as	www.robelle.com, www.adager.com …

Software Support

- HP MPE/iX Support Ends 12/31/2006 or Earlier

- 3rd Parties Have Pledged Long Term Support, Needs to be Monitored
- Maintenance of Applications Systems
- ❑ Consultants and Independent Firms Ready to Assist – Try One Now
- ❑ Organize Your Applications Source Code
- ❑ Upgrade to a Supported Version of MPE/iX

Hardware Support

- HP Support Ends 12/31/2006 or Earlier
- HP Equipment is Very Reliable
- Disk Drives, Tape Drives, Printers are High Failure Items
- ❑ 3rd Party Support is Readily Available – Try It Now
- ❑ Stock Your Own Parts/Systems
- ❑ Upgrade to A-Class or N-Class Systems

Backup Strategy

- ❑ Backup Data and System (CSLT) Regularly
- ❑ Use Multiple Drives
- ❑ Verify Data Backups with VSTORE.PUB.SYS
- ❑ Verify CSLT with CHECKSLT.MPEXL.TELESUP
- ❑ Use Proper Storage and Age Tapes
- ❑ Run BULDACCT.PUB.SYS
- ❑ Consider DLT Technology

Training

- Keep Your Users and Technical Staff Trained
- Training Available from HP, Vendors, and 3rd Parties
- Utilize Training Opportunities at Conferences and Symposiums
- Cross Training is Important

MPE/iX Version Considerations

- 6.5 Supported through 12/31/2004
- 7.0 Supported through 12/31/2006
- 7.5 Supported through 12/31/2006

- PA-8700 Systems Require 7.5
- Upgrades and Patches Require HP Support Contract
- Newer Versions Have Additional Functionality

Making a Business Case

- Study the costs for each choice or combination (i.e., buy a new a/p package, but keep the custom applications on the 3000 indefinitely). Migration costs have to include:

  New Hardware and Software

  Training Users and Staff

  Data and Program Conversion and Testing

  Test System Hardware

  Additional Staffing

  JCL, UDC, and Command File Conversion

  New Written Procedures
- Plan For Company Business Growth
- New Functionality or Applications May be Needed
- Does Your HP 3000 Meet Your Needs
- How Can Your Company Be More Competitive
- Government Retention Requirements May Require Keeping the 3000
- HP or 3rd Party Support Availability
- Budget Money May be a Major Limiting Factor

Summary

- Homesteading is Often the Least Expensive Choice
- Risks Will Get Higher with Time If Resources/Suppliers Dwindle
- Proper Systems Management is Key
- Unless you can Make a Proper Business Case to Migrate, Stay on the Platform
- Planning is Critical to Success

A 3000 Emulator: Interview with Mark Klein

Waiting for the Key to MPE's New Factory

About this article: originally appeared in the March 2003 issue of ***The 3000 Newswire.*** *This interview with Mark Klein, President of DIS International, asks all the questions that an HP 3000 user might ask about the idea of emulating the 3000 on new hardware, and provides some reasonable answers.*

Mark Klein has masterful MPE experience, and he wants to put it to work on behalf of homesteading customers. The former R&D director for ORBiT Software returned to his independent development and consulting roots during the first year after HP's decision to exit the 3000 market. During most of that year Klein has served on the board of directors of OpenMPE, Inc., the organization dedicated to extending MPE's lifespan beyond HP's plans for the 3000 operating environment.

Few periods have been as quiet as the last three months of OpenMPE's existence. The organization has been conducting careful, intensive talks with HP to tackle the MPE licensing issue, a subject that is essential to the existence of any 3000 hardware emulators. Those emulators can make up the next generation of HP 3000 hardware, once Klein and the other board members wrest the details from HP on new MPE licensing.

New MPE licenses will open the door to OpenMPE's most creative work: marshalling the MPE technical expertise that has become widely available into some sort of virtual lab. Most homesteading advocates understand that the 3000 must continue to grow, and so OpenMPE must demonstrate to HP that it's possible to maintain MPE outside of HP lab.

Though he's not campaigning for the job, Klein is a good candidate to head up such a lab. Leveraging GCC work from the University of Utah on HP-UX, Klein did the bootstrap porting of the GNU Software Foundation's C++ compiler to MPE, work that set the stage for the HP 3000's compatibility with other systems. Without a C++ compiler, things like Web servers, Java, Samba file sharing, Internet domain name services and many of the platform's Posix tools wouldn't have arrived on the HP 3000 in time, if at all. Working nights and weekends for the equivalent of a year of full-time coding, Klein brought this new language to the HP 3000, one that opened the door for the HP 3000's interoperability.

Between his experience developing for HP — which honored him with the HP 3000 Contributor award — and years spent managing ORBiT Software's developers in an era when 20-plus years of MPE background was the norm for that group, Klein looks to us like a smart pick for an OpenMPE post that could emerge this year. We spoke with him about the prospects of making MPE thrive outside HP, first in December and then again in late January and February — the months when the air grew thick with anticipation over HP's MPE homestead licensing arrangement.

Can MPE be maintained by experts outside of HP?

I think that there is a very good possibility that it can. There are some very good people with an in depth knowledge of MPE internals, Networking internals, TurboIMAGE internals, etc. that could make this work. If anything, these people could be the "architects" for a future development effort, leveraging their skills with others that don't have that specialized knowledge.

Is there enough non-HP expertise available around the world to enhance MPE with new features in the years to come?

The same answer I just gave applies here as well. The main point is that I think there are enough of the architect quality people available to steer such an effort. Without that, it is my opinion that the chances of maintaining, let alone enhancing, MPE suffer significantly.

Significant parts of MPE have been from vendors other than HP. How do you feel about the prospects of replacing these sections with Open Source alternatives?

I don't see any reason that it couldn't be done with the exception of some of the lower level IPC code. Whether or not it is realistic to do this is another issue. I also suspect that there are other issues with the Open Source licenses that might make this a difficult course to follow.

One such example is the GNU requirement of "making the source available" to anyone that requests it. It is my understanding that HP desires not to make the MPE source openly available at this point in time, so incorporating GNU licensed code might not work. On the other hand, I can also see a scenario whereby the GNU code could be "plug replaceable" in the field by the ultimate user and thereby not cause MPE to fall under the GNU license. I'm not an attorney, so I could be all washed up here.

Has HP surprised you in any way regarding its willingness to help MPE survive beyond 2006?

Not really. I think they are trying to do the right thing for their customers as they've said all along, even if we don't agree with their conclusions. It is possible that the "business case" for MPE beyond 2006 was not well enough presented before last November for them to initially consider it. Since that time, it has become obvious that there is a case.

Why won't a traditional Open Source program work for continuing with MPE development?

This is a tough question, and a lot of people will disagree with me on this.

If you set the intellectual property issues aside, I don't think that the critical mass of developers is there to make it work. With a small pool of developers available, unless there is focus, there will be no movement to benefit the majority of MPE users. Would you rather have 20 different developers each releasing 20 different versions of MPE, or would you rather have 20 different developers working together on one release?

Let me use my port of GNU Compiler Collection [GCC] to MPE as an example. I'm one of a large number of contributors to the GCC effort. Because I felt a need, I did the port. But, while I've contributed the port back to the development effort, it has never been fully incorporated into the sources. Why? Because that port has a very small attraction to the masses of GCC users. The core development team wants to do things that benefit the masses.

Now, reduce that by an order of magnitude or two for MPE. You get the 20-release scenario. I think a focused development effort is the only way in our universe that we will see progress.

Can an independent effort to maintain MPE succeed if IMAGE is maintained by a separate company? Does one organization need all the code for the effort to be effective?

I think it can succeed. Heck, when I got my first consulting arrangement with HP in the early 1980's it was for the database lab. That lab was completely separate from the MPE lab at the time. While there might need to be close partnership or relationship between the companies to make it work, I don't see why it couldn't.

You've said that HP seems to be willing to license MPE source to individuals. What's the benefit of that kind of development plan, versus the Open Source arrangements that have powered Linux?

I hope I am not reading too much between the lines, but I have not heard them say "absolutely no" to the source licensing issue, and I would be interested in being such a licensee. Earlier I brushed aside the intellectual property concerns. I think those are major concerns. Private licensing can address those issues to HP's satisfaction, I would imagine.

Some companies want to simulate MPE on other platforms today. What's the reason a customer might want to do that, versus the harm they might do in choosing a simulated solution?

I cannot say that there would be harm in choosing a simulation method vs. an emulation method. The major difference between them is that you need to be able to recompile your software and possibly change some of your processes to work in the simulated environment. In the emulated environment, you should be able to pick up your existing applications and data, lock, stock and barrel, and move them to another platform. There are arguments that can be made on both side of this issue. I'll bet that if emulation becomes a reality, we'll see a presentation called "Simulation or Emulation" at some point in the future. I'll leave that to the marketing gurus of the interested parties.

The only compelling technical reason for emulation would be needing complete binary compatibility, and you can't find the source code for some application you absolutely require for your business. In that case, doing emulation is the only course, short of re-inventing the application.

Now that HP has agreed to license MPE to new users on a hardware emulator, what's the next biggest roadblock to making homesteading a viable Transition solution?

I think total cost of ownership and total cost of development. There are discussions about the hypothetical cost of a MPE license on the OpenMPE list right now. There have also been some discussions about what the emulator might cost. Sooner or later it will come down to dollars. The emulator companies will need enough return on their investment to justify that investment. The homesteaders will need to receive enough value to justify spending those dollars.

What do you see as the real deadline for a customer to leave their HP 3000? Support seems to be available at least to 2010, and the hardware availability might well be answered by an emulator. What does a customer have to consider when deciding their migration timeline?

I don't think there is a real deadline, as much as I hate to admit it. By that, I mean that each company will need to decide this themselves, based on their own business conditions. I know firsthand of one company that has decided that migration or replacement is just too expensive for them and that they will stay with the 3000 until they can no longer use it to run their business and then they will close up shop. I wonder how many others that we are not hearing about have come to this same conclusion?

So what's going to motivate change in the 3000 community if there is no real deadline?

Politically companies are going to be faced with a migration, even though there is no deadline, by virtue of the fact that HP has made the decision to end the platform. Look at what QSS is doing [by porting to Linux]. They're making the move, even though there's the possibility of staying on the 3000 for another five to 10 years. HP is saying do something. That's enough for a company doing business-critical applications on the 3000.

Is there a need for an in-person training initiative to foster 3000 and MPE skills? It seems like so much of the training now is for migration tasks. Does the market need more people that know how to manage the 3000?

That's a good question that I really am not sure I can answer since I have very little to do with the operational end of the 3000.

If you listen to some people on the MPE specific email lists, there is the thought that MPE can grow, post EOL. If that comes to pass, then surely more training will be needed. I guess this could become a chicken and egg thing, though.

How much of a lift does HP's decision to license MPE for emulators put into the OpenMPE movement?

Well, you're asking me, so you'll get my opinion. At the formation of the OpenMPE group, I think I was the lone voice talking about licensing issues. Most everyone else was concerned about Open Source. Slowly, I think it became obvious that the number one issue really was licensing. Without it, nothing else could happen.

With a replacement license scenario, the MPE universe would slowly get smaller until it completely died. With a licensing scenario whereby new MPE licenses could be created, the universe could grow.

I'm ecstatic about the decision to allow the creation of new MPE licenses. I can not express how big I think this really is and how favorable it can be to the people that don't want to see MPE die.

Your expertise is unique in the 3000 community. Is there enough hope on the homesteading table to keep you involved with HP 3000s and MPE?

I have always said that when I retire, it will be from the 3000. I'm too young to retire for at least another 15-20 years!

Is it important for the OpenMPE movement to have its own lab?

I don't know that it's necessarily important. A lot of the stuff could potentially be done by the emulator companies. However, there's a handful of us who think that MPE could be supported beyond end of life by others than HP, in case an emulator company doesn't come along and step up. When you start to look at people of the caliber of those leaving CSY now [through layoffs], there's a fear that the capability to make critical bug fixes won't exist in two years, or even after HP's end of life deadline.

Based on what Dave Wilde said at HP World, they're going to support a homesteading movement. I don't think they had their ducks all lined up in a row when they made that statement in terms of how they're going to support that movement, if they lose all of their MPE knowledge base.

One of the things that OpenMPE is attempting to do as an advocacy group is making certain that kind of knowledge exists outside of HP. I think that's the major goal right now.

Can you get to the point of being able to pick up some of the departing MPE talent through contracts? How do the expectations from the homesteading community align with the process that's in front of OpenMPE?

People tend to want everything, and they want it in backwards order. It's the chicken and egg thing: where do you get the funding for [those contracts] if there's no possibility of you doing anything with the people you'd get with that funding?

There's a logical progression you have to follow. Let's come up with intermediate tasks that will get you to the greater goal. The most immediate need is to come up with a licensing arrangement. If you can't license MPE, everything else is moot, in my own opinion.

A small group of customers is still wishing HP will change its mind about its 3000 business. Does it make any sense to you for that to happen now?

No. Whatever damage has been done to the market has already been done. As a die-hard MPE-er, I would love to see them reverse direction. In my opinion, I think there's too much momentum going downhill away from the platform for them to do that.

What about changing the end of support date?

That may be something worth considering. 2002 has been a planning year, and migrations haven't even begun yet. HP made the announcement so late in 2001 that the budgets for 2002 were already in place [for customers]. Very little activity could happen in 2002. As a consultant I sat on my butt for much of that year, because none of the migrations that were forecast for 2002 happened.

We're more than 14 months into this before the larger companies are starting to move. Can they move effectively and get off the platform before the end of support [in December of 2006]? No. So there's a strong argument for taking that end of support date and adapting it. It could be possible on a case by case basis.

HP 3000 Tune Up

By 2007, Hewlett-Packard is scheduled to set the remaining HP 3000 sites completely adrift. We will be on our own in keeping our 3000s humming – well on our own except for the independent vendors and the helpful 3000 users. For decades, much of the really useful training and tips for the HP 3000 has come from outside Hewlett-Packard. Do-It-Yourself has been the style, since the start. There is no reason to wait until 2007 – start doing it yourself now.

This section of the *HP 3000 Evolution* contains a selection from the hundreds of useful technical papers that have been written about this computer platform, supplemented by links to many other papers, often with excerpts and reviews.

The topics covered are somewhat arbitrary, but the aim is to solve a wide range of common problems, including:

- linking the 3000 to other computers,
- finding MPE resources and information
- dealing with system hangs and aborts
- flexing underused muscles (i.e., under-utilized software)
- programming with the command interpreter
- improving file and system security
- optimizing the IMAGE database (multiple chapters)

We conclude with an Interview of Vladimir Volokh (VESoft), probably the one person in the world who spends the most days per year working to tune up HP 3000s.

- Bob Green

HP 3000 Reference Information

By Bob Green, Robelle

About the author: Bob Green founded Robelle on the idea of technical excellence and sharing information. He has been a speaker at numerous international, national and local conferences related to the HP 3000, writing and presenting papers and tutorials which are acclaimed for their usefulness and interest. Two recent papers are "Building Better Software" and "Transforming TurboIMAGE Data to Eloquence, Oracle and More". Since December 2000 he has updated the Robelle home page daily with tips and news, searching out web resources that will benefit HP 3000 users.

As Hewlett-Packard scales back it's commitment to the HP 3000, it is important that end users take responsibility for their own systems. Toward that end, I have compiled a directory of useful reference material on the HP 3000 (excluding the topic of migration, which is covered in the last section of this book) .

OpenMPE

One site that has as its explicit goal to archive the useful information required to maintain and operate an HP 3000 is the "OpenMPE" site:

```
http://www.openmpe.org/
```

OpenMPE is non-profit organization whose goal is to extend the life and usefulness of the 3000; it is gathering 3000 information that will be needed after HP exits the business. For example, they already have list of companies that use the 3000, relative power ratings of the 3000 servers, 3000 software tier structure from HP, lists of hardware and software support organizations, freeware downloads, and links to MPE-related manuals.

3000 Manuals

As we go to press, HP has most of the MPE software manuals on-line in PDF and HTML format at

```
http://docs.hp.com
```

John Burke has links to many of the most useful on this web page

```
http://www.burke-consulting.com/indexhpdocs.htm
```

At HPWorld in September 2002, Hewlett-Packard agreed to make the 3000 hardware and software manuals available to independent web sites such as OpenMPE when they remove the manuals from the HP web site. I hope that someone at OpenMPE has already downloaded copies of the files, just in case.

Some other vendor's 3000-related manuals are on the web as well:

Robelle:

```
http://www.robelle.com/library/manuals/
```

Lund:

```
http://www.lund.com/support/
```

Adager:

```
http://www.adager.com/
```

Bradmark:

```
http://www.bradmark.com/site/downloads/downloads.html
```

However, many vendor web sites have no obvious sign of user documentation for their HP 3000 software products. For example, Cognos and DISC. So for these products, you should ensure that you carefully file copies of the documentation.

Interex

Another web site that should be useful to those with HP 3000 systems is the Interex users group web site:

```
http://www.interex.org/
```

However, it is difficult to find 3000 information on the site, many of the pages are dated, and the most useful information, which is the archive of papers from past HPWorld Conferences, requires that you login with your Interex user number and name. The papers are in PDF format and sorted by Author, Title and Track. The Track sorting is most useful, since we are presumably looking for papers under the MPE track. In case you are a member, know your membership number, and want to browse some technical papers, here are some links:

HPWorld 2002:

```
http://www.interex.org/conference/hpworld2002/proceedings/index.html
```

HPWorld 2001:

```
http://www.interex.org/conference/hpworld2001/proceedings/home.html
```

HPWorld 2000:

```
http://www.interex.org/conference/hpworld2000/proceedings/home.html
```

Although there were many useful papers published in years prior to 2000, I could not find them on the Interex web site.

The 3000 Newswire

A great source of information is "The 3000 Newswire" magazine. As a subscriber to the magazine, you can browse a great deal of on-line info at their web site.

```
http://www.3000newswire.com
```

3kWorld

Another site that was once quite useful was 3Kworld at http://www.3kworld.com - but it hasn't been updated in a while and may be winding down. Vendors continue to post press-releases on the site, but the site is showing signs of neglect.

Books on MPE:

Beyond RISC

This book is a guide to the PA-RISC version of the HP 3000 and the software of the MPE operating system. It is slightly dated, since MPE has had many releases since the conversion to RISC architecture, but it is still very useful. The book is out of print, but a few copies are still available at Allegro. Email to cindy@allegro.com if you would like to buy one.

MPE/iX System Administration Handbook

by Jon Diercks

```
http://diercks.net/jon/
```

This book was published in 2002 by Prentice Hall. It "covers the essential tools and skills required for successful MPE/iX system management. Diercks presents hands-on examples, solutions to common problems, and dozens of tips for streamlining operations and making the most of your HP e3000 system." Very useful.

```
http://diercks.net/mpe/
```

TurboIMAGE

TurboIMAGE (or IMAGE for short) is the database included with MPE. The user manuals are on-line at docs.hp.com and here are some other useful links:

The IMAGE FAQ answers the most common questions about this database and points to useful vendors, tools, etc.:

```
http://beechglen.com/pub/imagefaq.txt
```

Robelle's IMAGE Programming Tutorial teaches how to write programs that access IMAGE databases:

```
http://robelle.com/library/tutorials/pdfs/imgprog.pdf
```

Robelle's IMAGE Performance Analysis explains how to use HowMessy, DBLoadng, Adager and DBGeneral to monitor and improve the performance of your database:

```
http://robelle.com/library/tutorials/pdfs/imgperf.pdf
```

Adager has a wide-range of technical papers about IMAGE on their web site:

```
http://www.adager.com/TechnicalPapers.html
```

3000-L Mailing List

The 3000-L mailing list is a place where users can post and answer questions. It will probably continue to be a place where the confused system manager can go for help with his HP 3000.

UTC archives

```
http://raven.utc.edu/archives/hp3000-l.html
```

3kWorld archives:

```
http://www.3kworld.com/message.asp?appMode=threadList&mbo_PK=318
```

Other lists at UTC:

```
http://raven.utc.edu/archives/index.html
```

Other mailing lists

```
http://www.techgroupmd.com/TGU%20-%20CyberCafe.htm
```

3000 FAQs

FAQs are Frequently Asked Questions.

HP 3000 FAQ in HTML format:

```
http://www.3k.com/faq/hpfaqi.html
```

HP 3000 FAQ Search engine

```
http://qwebs.qss.com/faq3k.html
```

Robelle’s Encyclopedia of HP 3000 and related topics is not an FAQ, but it is an excellent resource of cross-linked articles:

```
http://www.robelle.com/library/smugbook/
```

Technical Papers on Jazz

Jazz is a web site at HP which developed as a resource for porting Internet and Posix utilities to MPE. It has a great deal of software for download, as well as numerous papers and reference documents. It is not known how long this server will stay in operation or where the content will eventually go:

```
http://jazz.external.hp.com/
```

Networking Trouble Shooting Tools

Presented by Eric Barkley at the Solutions Symposium and written by Jeff Hofmeister.

```
http://jazz.external.hp.com/papers/SolSymposium_03/NetworkingTools.htm
```

This is a comprehensive presentation of many tools and procedures to troubleshoot MPE/iX networking problems. It covers Nettool, Tracert, Xperf, Loopinit, Nmdump, Tcpip.netlook.sys, Linksumm, Netrcmd and Linktrcmd.

The authors also provide several tools that are mentioned in the talk, including two programs and three script files.

```
http://jazz.external.hp.com/papers/SolSymposium_03/erictools.tar
```

This tutorial provides a tremendous amount of useful information, but is virtually impossible to read due to the web presentation (graphics instead of text, with a small fuzzy font). It really needs a downloadable PDF or Word file with a reasonable-size font.

Hidden Value and Net.Digest

John Burke (www.burke-consulting.com) writes the "net.digest" and "Hidden Value" columns in "The 3000 Newswire". The net.digest columns and the Hidden Value columns are on-line at his web site (organized by date) as well as a "Best Of" compilation by topic at:

```
http://www.burke-consulting.com/best_of/best_of_TOC.htm
```

Note: The columns at John Burke's site are the author's editions, unedited and uncut, with some material that has never been published in the NewsWire. On the other hand, about a year's worth of these two NewsWire columns, from 1998-1999, can only be found in the archives of the 3000 NewsWire site. More than six years of MPE technical papers, indexed through a search engine, are available at

```
http://www.3000newswire.com/subscribers/TechHeads.html
```

More Technical Papers

Allegro has an excellent guide for new 3000 users as "Learning MPE":

```
http://allegro.com/papers/learnmpe.html
```

Robelle has 14 papers and 14 tutorials on-line covering topics from the System Debugger to programming the CI.

```
http://www.robelle.com/library/
```

Allegro has 13 papers on-line covering topics from "what to do when your system is hung" to "setting up a network printer".

```
http://allegro.com/papers/
```

Adager has most of the database papers written by their staff over the years, plus guest papers from Vesoft and others:

```
http://www.adager.com/TechnicalPapers.html
```

Interex has an excellent resource, "Managing HP 3000s for the Complete Novice" which is in 24 parts!

```
http://www.interex.org/enterprise/hp3000.html
```

News:

Of course *The 3000 Newswire* is the premier source of news.

```
http://www.3000newswire.com
```

However, Robelle also posts 3000-related news and links to its home page and archives them via monthly newsletters.

```
http://www.robelle.com/
```

John Dunlop maintains the www.hp3000links.com web site, and shows new links on his what's new page at

```
http://www.hp3000links.com/whatnew1.htm
```

3kWorld: some vendors post news and announcements on 3kworld, although the site shows signs of inattention. It is probably a good idea to save a copy of any articles here that you find useful.

```
http://www.3kworld.com
```

More 3000 Links:

Adager:

```
http://www.adager.com/OffCampusHP3000.html
```

Robelle:

```
http://robelle.com/support/hp3000.html
```

List of vendors:

```
http://www.triolet.com/HPVend/hpvend.html
```

Using Old Technology for New Problems

By Alan Heter and Alan Wyman, US Foodservice - Eagan

About the Authors: Alan Heter, after a brief stint with an IBM 360 (six months), has spent his entire career programming and performing system maintenance on HP 3000 equipment. He has been involved in education, manufacturing, and distribution. Alan Wyman has been working on the HP 3000 since the Series II in the middle '70s and has been involved in health care, manufacturing, publishing, and distribution.

If you are homesteading, at some point you will want to interface new technology to an existing HP 3000 application. Don't worry: most problems can be solved with the existing tools that you know and appreciate: an intranet client-server or application-server relationship can be written across multiple platforms with FTP, COBOL, and batch jobs. This chapter describes just such a real-life example, using the FTP Site Stream command, batch jobs, COBOL, and Byte Stream files. The example is an application server between an HP 9000 and an HP 3000, with the article covering mostly the HP 3000 portion of the application. Our division was required to exchange operational data from our HP 3000 with the corporate HP 9000 server.

How did we get to this point? Our corporate office required us to connect to their new dot com initiative. They had already completed their coding and were ready to accept data from us. Our location has a stand-alone HP 3000 and applications completely separate from the corporate systems. The technology of the corporate server was a given – our job was to come up with a solution that interconnected with that given.

Corporate had already made decisions to base the interface on HTTP Post and we would use Java to create these Posts. Post is a CGI action, a standard way of invoking a function on the server for the web.

We did not have Java expertise on-site. We are running the usual COBOL, TurboIMAGE, and VPLUS. We located a package called PerCOBOL that would allow us to write the programs in COBOL and compile them into Java.

At that point, the problems started brewing. Performance was not up to the necessary level using the HP 3000 implementation of Java. We were running on a 987/150, and even before this project, we needed a new machine. To complicate the problems, our company was being purchased and there was a freeze on all capital requests. This left us looking for alternatives.

What we did was go back to the basics. We knew that FTP existed on both platforms. One feature of FTP is the ability to start a process on the remote machine. This became the basis of the new strategy.

Since PerCOBOL was the development language, it was easy to discard the Java and recompile the source in HP COBOL. We had actually coded the original programs with sections of code that defined the minor differences between PerCOBOL and HP COBOL. We defined file formats to mimic the HTTP Post transactions.

We still had some details to work out. The file naming convention became very important. We had to have a way to determine the filename to process when the batch job was streamed by the remote system. One of the system variables in MPE/iX is HPSTREAMEDBY. This variable contains the complete logon of the job or session that initiated a job. Within the job this can be parsed to get the session name parameter. Corporate agreed to setup their programs to put a root part of the filename in the session name parameter of the logon. They would log on as R0078418,FTP.PRD. When the site stream command initiated a job, the job could check who streamed it and determine the file to look for. In this example, the inbound file from corporate would be named /PRD/PUB/dotcom/R0079418_in. Within the job the following commands would set a variable with the root filename.

```
:SETVAR fileroot LFT( '!HPSTREAMEDBY',POS(',','!HPSTREAMEDBY')-1)
:SHOWVAR fileroot
FILEROOT = R0079418
```

The root part of the filename must follow the MPE rules for a filename, even though the files were stored in HFS directories. In other words, start with a letter, no special characters, all uppercase, no more than 8 characters in length. This allows the root to be used as a session identifier. The corporate applications support chose to set a unique first letter for each transaction type and increment a counter for the remainder of the root filename.

Once the file was transferred to the HP 3000, the remote process initiates a batch process to handle the transaction. The batch process is started with FTP's Site command. This command allows the remote system to execute commands on the host system.

```
site stream aljftp01
```

This command starts a batch job on the HP 3000 to process the file. In the example, the job would be ALJFTP01. This job reads the inbound file and creates an outbound file to be picked up by the remote process.

The HP 3000 recognizes the site command. There are versions of FTP that require the use of the quote command to send the command to the HP 3000. In this case the command from the remote site looks like

```
quote site stream aljftp01
```

Here is an example using a PC in place of the corporate Unix server with an HP 3000 called SEVENOF9.

```
C:\WINDOWS>ftp
ftp>OPEN SEVENOF9
Connected to SEVENOF9.
220 HP ARPA FTP Server [A0010H08] (C) Hewlett-Packard Co. 2000 [PASV SUPPORT]
User (SEVENOF9:(none)): R0079418,DOTCOM/********.PRD
230 User logged on

ftp> put R0079418 /PRD/DOTCOM/alink/in/R0079418
200 PORT command ok.
150 File: /PRD/DOTCOM/alink/in/R0079418 opened; data connection will be opened
226 Transfer complete.
ftp: 91 bytes sent in 0.00 Seconds 91000.00 Kbytes/sec.

ftp> quote site stream aljftp01
200 STREAM command ok.
```

This is the first part of the job stream executed by the remote process. The HPSTREAMEDBY variable is parsed for the file root and file equations are created to reference the full filenames.

```
 :SETVAR fileroot LFT('!HPSTREAMEDBY',POS(',','!HPSTREAMEDBY')-1)
 :SETVAR filerootx '!fileroot'+'.tmp'
 :SHOWVAR fileroot
 FILEROOT = R0079418
 :FILE aldcin= /PRD/DOTCOM/alink/in/!fileroot
 :FILE aldcout= /PRD/DOTCOM/alink/out/!filerootx
 :IF FINFO('/PRD/DOTCOM/alink/out/!fileroot','EXISTS') THEN
  *** EXPRESSION FALSE: COMMANDS IGNORED UNTIL MATCHING ELSEIF/ELSE/ENDIF
 :    PURGE /PRD/DOTCOM/alink/out/!fileroot
 :ENDIF
  *** RESUME EXECUTION OF COMMANDS
 :RUN ALDC01OB.CODE.PRD
  ********************************************
  Begin ALSJ01 read WEB on 03/08/02 at 06:40:47
  Record size read = +000000063
```

The job processed the input file and created a new file to send back to the remote system. In the meantime, the remote system is polling the HP 3000 for the file. Because of this polling it was necessary to hide the output file until it was ready for transfer. We hid the output file by creating it with a .tmp extension, which was not expected by the remote system. After the processing was complete, we renamed the file to remove the .tmp extension and let the remote system "see" it.

The end of the jobstream handles the 'hide the file' routine. Note that we designed the directory structure to have an in and an out directory. This simplified the actual file names, yet kept the inbound and outbound separate.

```
End ALSJ01 for LOGIN-ID 2004     on 03/08/02 at 06:40:48
************************************************
END OF PROGRAM
RENAME /PRD/DOTCOM/alink/out/!filerootx, /PRD/DOTCOM/alink/out/!fileroot
:EOJ
```

The Unix server attempts FTP ‘get’ commands until it is successful. It pauses between Get operations to allow the job time to run on the HP 3000.

```
ftp> get /PRD/DOTCOM/alink/out/R0079418
200 PORT command ok.
150 File: /PRD/DOTCOM/alink/out/R0079418 opened; data connection will be
opened
226 Transfer complete.
ftp: 56475 bytes received in 0.27Seconds 209.17Kbytes/sec.

ftp> quit
221 Server is closing command connection
```

There is one warning. Only so many files can exist in a single group on the HP 3000. So we need to either purge the files periodically or purge them immediately with another site command. We chose to purge them periodically and this is done via a scheduled job. Failure to run that job for long periods of time can cause system failures.

The Unix server sends bytestream files to the HP 3000. These consist of a start of the file, the data, and an end to the file, with no record structure as there is on the HP 3000. Bytestream files must be opened using HPFOPEN rather than the standard COBOL OPEN statement. In our case, we set aside a large buffer for the inbound transaction because it could contain many ‘records’ as normally defined.

```
 01  SERVLET-RECORD                        VALUE SPACES.
     05  SERVLET-RECORD-A        PIC X(18824).
 01  WS-FILE-NAME-IN.
     05  FILLER                  PIC X     VALUE "'".
     05  WS-FILE-NAME-ACT        PIC X(06) VALUE "ALDCIN".
     05  FILLER                  PIC X     VALUE "'".
 01  WS-FILE-NAME-OUT.
     05  FILLER                  PIC X     VALUE "'".
     05  WS-FILE-NAME-ACT-OUT    PIC X(07) VALUE "ALDCOUT".
     05  FILLER                  PIC X     VALUE "'".
```

We define the HPFOPEN parameters in the working storage section. This includes the expected length and status return identifiers. Note that the record size matches the record defined above. Many of these same identifiers will also be used with the output file described later.

```
01  HWB-INTERNAL.
     05  HWB-INTERNAL-ITEMS.
         10  HWB-INT-CONST-0     PIC S9(09) COMP SYNC VALUE 0.
         10  HWB-INT-CONST-1     PIC S9(09) COMP SYNC VALUE 1.
         10  HWB-INT-CONST-2     PIC S9(09) COMP SYNC VALUE 2.
         10  HWB-INT-CONST-3     PIC S9(09) COMP SYNC VALUE 3.
         10  HWB-INT-CONST-4     PIC S9(09) COMP SYNC VALUE 4.
         10  HWB-INT-CONST-9     PIC S9(09) COMP SYNC VALUE 9.
         10  HWB-INT-CONST-10    PIC S9(09) COMP SYNC VALUE 10.
         10  HWB-INT-FILE-NAME   PIC X(256).
         10  HWB-INT-FNUM-D      PIC S9(09) COMP SYNC.
         10  FILLER              REDEFINES HWB-INT-FNUM-D.
             15  FILLER          PIC X(02).
             15  HWB-INT-FNUM    PIC S9(04) COMP.
         10  HWB-INT-PTR         PIC S9(09) COMP SYNC.
         10  HWB-INT-STATUS      PIC S9(09) COMP SYNC.
         10  FILLER              REDEFINES HWB-INT-STATUS.
             15  HWB-INT-STATUS-INFO    PIC S9(04) COMP.
             15  HWB-INT-STATUS-SUBSYS  PIC S9(04) COMP.
         10  HWB-INT-DISP        PIC S9(09) COMP SYNC VALUE 0.
     05  HWB-REC-SIZE            PIC S9(09) COMP VALUE -18824.
     05  HWB-ACT-SIZE            PIC S9(09) COMP VALUE 0.
     05  HWB-ERROR               PIC S9(04) COMP VALUE 0.
```

The actual call to HPFOPEN uses a number of parameters. In addition to the file number and status return identifiers, a list of other parameters help to define the file attributes to prepare to read the file. The parameters are each identified by a number as shown in Table 1.

Table 1 – Input HPFOPEN parameters

Parm #	Parameter Name	Description
2	Filename	This is the actual fully-qualified filename.
3	Domain	A value of 3 specifies an old permanent or temporary file.
6	Record Format	A 9 in this parameter denotes the bytestream file format.
11	Access	Read access is set with a zero.
77	Data Format	Access the file in native bytestream format by setting this to 2.

```
0100-OPEN-INPUT SECTION.
    CALL INTRINSIC "HPFOPEN" USING  HWB-INT-FNUM-D,
                                    HWB-INT-STATUS,
                                    2,  HWB-INT-FILE-NAME,
                                    3,  HWB-INT-CONST-3,
                                    6,  HWB-INT-CONST-9,
                                    11, HWB-INT-CONST-0,
                                    77, HWB-INT-CONST-2,
                                    0.
    IF HWB-INT-STATUS <> 0
        DISPLAY 'HPFOPEN FAILED' UPON SYSOUT
*       Mode 2 call will display error msg on $STDLIST
        CALL INTRINSIC "HPERRMSG" USING   2,
                                         \\,
                                         \\,
                                         HWB-INT-STATUS
        STOP RUN
     END-IF.
```

Once the file is open, we use a standard FREAD to read the data into the program. This has the advantage of letting the program know how much data is there to process. The FREAD has two parameters regarding the length of the file read - the expected length and the actual length. The expected length is the buffer size, whereas the actual length is what was actually read by the intrinsic. The program can use this information to parse the data into logical records.

```
0200-READ-INPUT SECTION.
   CALL INTRINSIC "FREAD" USING HWB-INT-FNUM,
                                SERVLET-RECORD,
                                HWB-REC-SIZE
                          GIVING HWB-ACT-SIZE.
   DISPLAY "Record size read = " HWB-ACT-SIZE.
   CALL INTRINSIC "FCHECK" USING HWB-INT-FNUM
                                 HWB-ERROR.
   IF HWB-ERROR = 0
       CONTINUE
   ELSE
       DISPLAY "File read error = " HWB-ERROR UPON SYSOUT
       STOP RUN
   END-IF.
```

Following good coding practices, the bytestream file is closed using the standard FCLOSE intrinsic. There is no difference in the close from that of a record format file.

```
0300-CLOSE-INPUT SECTION.
   CALL INTRINSIC "FCLOSE" USING HWB-INT-FNUM,
                                         HWB-INT-DISP,
                                         0.
```

For the output file, HPFOPEN requires additional parameters; these are described in Table 2.

Table 2 – Output HPFOPEN parameters

Parm #	Parameter Name	Description
2	Filename	Actual fully-qualified filename.
3	Domain	A value of 4 creates a new permanent file.
5	Designator	Other options specify the filename, the intrinsic is told this by passing a zero.
6	Record Format	A 9 in this parameter denotes the bytestream file format.
7	CCTL	Use a zero to create the file without carriage control.
10	File Type	A bytestream file is a standard file type.
11	Access	Write access is set with a 1.
13	Exclusive	A value of 1 will keep possession of this file in the calling process until it is closed.
50	File disposition	A value of 0, for no change, will keep the file on the system when it is closed.

```
0400-OPEN-OUTPUT SECTION.
    CALL INTRINSIC "HPFOPEN" USING  HWB-INT-FNUM-D,
                                    HWB-INT-STATUS,
                                    2,  HWB-INT-FILE-NAME,
                                    3,  HWB-INT-CONST-4,
                                    5,  HWB-INT-CONST-0,
                                    6,  HWB-INT-CONST-9,
                                    7,  HWB-INT-CONST-0,
                                    10, HWB-INT-CONST-0,
                                    11, HWB-INT-CONST-1,
                                    13, HWB-INT-CONST-1,
                                    50, HWB-INT-CONST-0,
                                    0.
    IF HWB-INT-STATUS <> 0
        CALL INTRINSIC "HPERRMSG" USING   2,
                                         \\,
                                         \\,
                                         HWB-INT-STATUS
        STOP RUN
    END-IF.
```

As with reading the input file, we do not use COBOL statements to write the output file. Instead, we use the FWRITE intrinsic to put the data into the file. This intrinsic requires the data buffer and the actual length of the data being sent. Once we write the data to the file, a standard FCLOSE releases the file, as with the input file.

```
0500-WRITE-OUTPUT SECTION.
   CALL INTRINSIC "FWRITE" USING HWB-INT-FNUM,
                                 SERVLET-RECORD,
                                 HWB-REC-SIZE,
                                 HWB-INT-STATUS.
    IF HWB-INT-STATUS <> 0
        CALL INTRINSIC "HPERRMSG" USING   2,
                                         \\,
                                         \\,
                                         HWB-INT-STATUS
        STOP RUN
     END-IF.

0600-CLOSE-OUTPUT SECTION.
   CALL INTRINSIC "FCLOSE" USING HWB-INT-FNUM,
                                 HWB-INT-DISP,
                                 0.
```

To minimize the data transfers made during the transaction, we needed a series of regularly scheduled data extracts. Corporate wanted to maintain a current customer and product listing, as well as sales history on the web servers. Originally, corporate was requiring real-time access to the data on the HP 3000, but a number of factors caused a change to a periodic update. The price lists were only updated weekly and the history updated daily. There were also performance issues on both the HP 3000 and the corporate machines. A database was setup to duplicate our data in the corporate format. This allowed a separation from the production databases.

This replication database allowed for some reformatting to the corporate data structures. TurboIMAGE data formats are not Oracle formats. The Oracle date format was also thought to be a problem. As it turned out, corporate had an intermediate database on their side, and they could translate to the Oracle format. They specified a format of ddMMMyyyy, for example 04APR2002. This was easy enough to do and to store in the TurboIMAGE database as a byte field.

Once the data was in the replication database, Suprtool extracted the data for transfer to the corporate servers. TurboIMAGE has an even-number of byte restriction for fields, but Oracle does not. So, we changed some of the fields to an odd-number of bytes at extract time by defining new fields.

```
!RUN SUPRTOOL.PUB.ROBELLE
BASE ALDC,5,CREATOR
GET CUST-MASTER
DEFINE CUST_NBR,CUST-NBR,8,DISPLAY
DEFINE CUST_NAME,CUST-NAME,25,BYTE
DEFINE OG_PRT_PRC_IND,PRINT-SW,1,BYTE
DEFINE STATE,ST-NAME,2,BYTE
DEFINE ZIP,ZIP-CODE,9,DISPLAY
DEFINE LAST_ORDD_DATE,ORDR-LST-DATE,10,BYTE
EXTRACT CUST_NBR
EXTRACT CUST_NAME
EXTRACT OG_PRT_PRC_IND
EXTRACT STATE
EXTRACT ZIP
EXTRACT LAST_ORDD_DATE
OUTPUT EXTRFILE,LINK
EXIT
```

Once FTP sent files to the corporate servers, we needed to load the data into the intermediate Oracle database on the HP-UX machine. Suprtool/UX was the answer. Before FTP'ing the files, we converted from the HP 3000 self-describing file format to the format needed for HP-UX. Robelle's SDUNIX utility, translated the single self-describing file into two files. One file contains the data; the other file holds the mini-dictionary that describes the data. The second file has the same name as the first, but with an .sd extension. These files must reside in the same directory.

FTP transferred these two files in binary mode. After the data transfer, it only takes four Suprtool commands to import the data into the Oracle database.

```
!SDUNIX;INFO="extrfile exsdfile NOLF"
!
!ECHO OPEN hp821npc                          > alfstcst
!ECHO USER !extruser !extrpass               >> alfstcst
!ECHO BINARY                                 >> alfstcst
!ECHO PUT extrfile  /var/opt/load/extrfile    >> alfstcst
!ECHO PUT exsdfile  /var/opt/load/extrfile.sd >> alfstcst
!ECHO EXIT                                   >> alfstcst
!FTP < alfstcst
!
!EOJ
```

Suprtool/UX commands needed to load the oracle database:

```
OPEN oracle_database
INPUT extrfile
ADD table_name
EXIT
```

Due to a related project that needs daily data from corporate, they are now sending us data using a Suprtool extract. They did not have Suprtool before this project, but they liked it so well that they are using it for that extract. Not a plug for Suprtool, just a satisfied customer.

Homesteading will mean that uncommon solutions will be needed for what will become more and more a common problem; trying to interface the HP 3000 to newer technologies. Sometimes it simply takes a combination of familiar solutions working together.

[*Editor: you will find numerous opportunities to link your 3000 to the other computers in your organization. For example, one of Robelle's Suprtool customers faced a common situation, where he extracted data from IMAGE for his end users and put it into Excel format, allowing them to review and modify it: the department budget, for example. But he had no easy way to put their updated budget back into his IMAGE database on the 3000. Here is how.*]

Importing Data from Excel into the 3000

```
http://www.robelle.com/tips/st-export-excel.html#import
```

For the export from Excel, we suggest you save the data as "formatted text - space delimited". The biggest problem you will then face is that your 3000 tool may expect the sign in a specific spot, or leading zeroes.

The new $Number function in Suprtool 4.7 allows you to easily "import" numbers into Suprtool, regardless of their format. With $Number, you can directly import numbers with signs, decimal places and even with or without leading zeroes.

Suprtool accepts these free-form "numbers" as display data types:

```
 1234.45-
 -12345
 -123.2134
 12343
 $123.45
```

Consider the following data:

```
Item-number New-Price
12345       +123.45
34563       + 27.5
21312       + 1.545
```

Let's say we want New-Price to be a double integer, and it currently occupies eight bytes starting in position six. Here is the Suprtool task:

```
 >in mynums
 >def item-number,1,5,byte
 >def new-price-ascii,6,8,display
 >def new-price,1,4,double
 >item new-price-ascii,dec,2
 >item new-price,dec,2
 >ext item-number
 >ext new-price=$number(new-price-ascii)
 >out somefile,link
 >xeq
```

So Your HP 3000 Is Hung. Now What?

by Allegro Consultants

About the authors: Allegro is a very technical group in California that specializes in MPE internals, HP-UX and AIX. They assist complex end-user sites, they provide technical support for other company's products, they write software products, and they stir the 3000 community about projects like the 3000 emulator. www.allegro.com

[*Editor: we start this chapter with a very helpful checklist from Allegro's web site which guides you through the steps to take when your system hangs.*]

Please follow this checklist to help the support people to better understand the system hang.

Does anything work?

Can you hit <return> at the console and get a prompt?

Yes: ____ No: ____

Can you run a program? Let's try `EDITOR`:

:editor

Did the banner display? ("`HP32201A.09.00 EDIT/3000`...")

Yes: ____ No: ____

Did you get the "`/`" prompt?

Yes: ____ No: ____

(You can use "`E`" to exit `EDITOR`)

Can anyone logon?

Yes: ____ No: ____

What does the "speedometer" say?

At the hardware console, press control-B. That should start a display, at the bottom left of the screen, of four-digit hex numbers. If the system is still somewhat alive, there should be a pair of hex numbers alternating between FFFF and F#FF (e.g., F7FF). We're interested in that F#FF, not the FFFF. (If one of the numbers is DEAD, then you have a System Abort, not a System Hang.)

Value: _______ (not "FFFF"!)

Is there disk activity? (E.g., disk activity lights flashing)

Yes: ____ No: ____ (If Yes, what ldevs?: ________________________)

Is there tape activity? (E.g., tape activity lights flashing)

Yes: ____ No: ____ (If Yes, what ldevs?: ________________________)

What are the last few console messages? (If possible cut/paste the console memory and email it to support.)

____

____

____

____

What were the last few things you remember the system doing? (e.g., jobs run, program run)

____

____

____

____

Take a dump:

(If this is the first hang you've had in a very long time, and if you have a large memory system, you might just decide to reboot instead):

1. press control-B, then TC, then press <return> ... do not enter RS!

   Note: do not wander away from the console ... you will need to be there when the computer asks questions:

```
Boot from primary path?     enter: Y
Interect with IPL?     enter: Y
```

2. Put a write-enabled tape into the tape drive.
3. At the ISL prompt, enter: DUMP

4. Enter a one line description of the dump, and press <return>
   The description should have your name & phone# in it, as well as a brief description of the hang.

   The dump should now start writing to the tape, and will print periods on the console while it is writing (one period per megabyte dumped). The dump will take between 5 minutes and 2 hours, depending upon how much memory your computer has, and how busy the computer was.

5. When the dump is done, eject the tape, write-protect it, and make sure it is labelled with the date and word "DUMP".

Restart the system as normal (e.g.: START)

The following is optional, but useful and speeds up dump analysis. If you skip it, then please send the dump tape you just created to the support group.

Load the dump onto your HP 3000:

1. Logon as `MGR.TELESUP,DUMPS` or `MANAGER.SYS,DUMPS`, or wherever you want to logon and put the dump.

   If you've never done this before, we suggest:

```
:HELLO MANAGER.SYS
:NEWGROUP DUMPS
:CHGROUP DUMPS
```

   (Note: you might want to `NEWGROUP/HOMEVS` the `DUMPS` groups to a user volume set that has a lot of available free space.)

2. Run the DAT (Dump Analysis Tool) program to load the dump:

```
:DAT.DAT.TELESUP
```

3. Tell DAT to load the dump into a file named something like HANG1.

   (There is a five character limit to the dump name. The first character has to be a letter, and may be followed by any combination of letters and digits.)

4. Enter the `getdump` command: (The following example creates a dump file called HANG1)

```
getdump HANG1
```

   DAT will request the tape (you may need to =REPLY to the tape request) and will allocate disk space to hold the dump. It then reads the tape onto disk, and displays a progress bar during loading.

5. Load the DAT macros: (HANG1 is the name from the prior step)

```
macstart "HANG1", "1"
```

   Note: if DAT says it found errors while loading the dump, and asks if it should continue, enter YES.

6. Pre-process the dump, saving time for support:

```
process_wait; ui_showjob
```

7. Exit DAT:

```
exit
```

8. STORE the dump to tape

```
:STORE HANG1@ ;; show; progress
```

(You can STORE the dump onto the original tape, if you wish ... the original dump tape is no longer needed, because the STORE tape version is more useful.)

9. Send the STORE tape with the dump to the support group.

Becoming a Dump Tape Expert

Editor: the preceding are instructions for the system operator or manager, for coping with a system hang. But if you want to be the kind of person who investigates such hangs (and system aborts), there is helpful information on the Internet.

For example, Stan Sieler of Allegro has another paper on their web site:

"Notes on System Failures, System Hangs and Memory Dumps for MPE/iX"

```
http://www.allegro.com/papers/dump.html
```

Stan writes:

Sometimes, the computer "dies"... so this note discusses system failures, system hangs, memory dumps, subsystem numbers, and interpreting a system abort number. Sometimes, the system is alive ... so the free speedometer is discussed.

There are two basic kinds of system failure that an MPE/iX (or MPE XL) user will encounter: a "System Failure" and a "system hang".

A System Failure reports the following information on the hardware console:

```
SYSTEM ABORT 504 FROM SUBSYSTEM 143
SECONDARY STATUS: INFO = -34, SUBSYS = 107
SYSTEM HALT 7, $01F8
```

Additionally, the hex display (enabled on the console by typing control-B) displays something like:

```
B007 0101 02F8 DEAD
```

Note that the "504" and "$1F8" above are the same value, shown in decimal and in hex. Further, the hex display shows "0101" and "02F8". These two numbers are reporting the following:

```
0101 02F8
```

The bold (and, depending on your Web browser, underlined) portions indicate packets 1 and 2 of the System Abort number (01F8) (i.e., the first two hex nibbles (01 and 02 above) of each 4-digit hex number are "packet numbers").

Note: if the System Abort number is in the range 0 to $FF (decimal 255), only one "Part" will be needed to represent it, and no "Part 2" will be shown.

"Basic System Problem Analysis"

Another source of expertise is Bill Cadier's paper at the 2003 Solutions Symposium entitled "Basic System Problem Analysis". The paper is available as an MS Word document, but the Word version is hard to read; the conversion process has inserted many pages with just the word 'Notes:' on them! Instead, use the HTML version at

```
http://jazz.external.hp.com/papers/SolSymposium_03/MPESysAnalysis.htm
```

Here is how Bill introduces his topic:

As the HP 3000 winds down it will be advantageous for owners of this system to be able to perform as much trouble shooting as possible. The amount of trouble shooting will be limited because source code for the OS is not available outside HP.

It is assumed that readers have good familiarity with the tools DEBUG, DAT and SAT. The documentation for these tools may be found online at:

```
http://docs.hp.com/mpeix/onlinedocs/32650-90901/32650-90901.html
```

You really need to be familiar with the internals of MPE to use this paper, but it is excellent for people who are planning to provide 3rd Party MPE support, analyzing dumps and chasing down system problems! Here is what the paper covers: Debug macros for accessing system, data structures, Process Management structures, Job/Session Management structures, File System structures, finding the GUFD of an opened or closed file, Virtual Space Management structures, Memory Management Structures, Dispatcher structures, Table Management, System Globals, the general registers and space registers, short vs long pointers, and the procedure calling convention. Finally, there is a detailed case study of analyzing an actual System Abort 663, plus a system hang.

Bill also provides a bit of good background information, such as:

The *PA-RISC Instruction Set Reference Manual* and the *Procedure Calling Convention* manual are pretty hard to come by. They are not at the docs.hp.com web site, so it is worth spending a little time going over some of the basics of the hardware.

DEBUG, DAT and SAT use aliases for certain of the registers, SP, the stack pointer will always be R30. DP, the data pointer (global variables in a program context) will always be R27. RP or the procedure return pointer is R2.

The procedure calling convention specifies that the first four argument values being passed in a procedure call be placed in registers R26 to R23. The first parameter going into R26 and onward to R23. The first parameter going into R26 and onward to R23. All additional parameters are placed into the stack frame that was created by the procedure making the call.

Parameters may require more than one register, a long pointer or LONGINT for example, will take two registers. If that occurs the registers must be aligned. This may result in one of the registers being skipped and left unused (more on this in a bit).

GR31 is called the "millicode RP" but it is also where the "BLE" instruction initially stores the current value of the PC register before making the branch. It moved to R2 immediately after that, in the "delay slot" of the branch.

R0 is a scratch register that contains the value 0. It is cannot be written to but it is legal to use R0 as a target register when a value is not required. For example, the "NO OP" instruction (one that does nothing) is 08000240 OR r0, r0, r0. Logically OR R0, through R0 giving R0... nothing.

Quick and Dirty Solutions with CI Programming

By Ken Robertson, formerly of Robelle, now with Adtech

About the author: Ken Robertson wrote this article while at Robelle and travelled the user group circuit presenting it. Ken is a programmer who loves techology, and loves clever solutions. Ken now lives in Montreal, where he is working on migration of HP 3000s.

Editor's Note:

If you are unfamiliar with the Command Interpreter on MPE, Jeff Vance's paper on "CI programming for MPE/iX 7.5" is a great introduction. This is a revised version of the CI Programming talk presented by Jeff several times in recent years. The speaker notes are an important part of this slideset. The first link is to browse the web version, the second is to download the PowerPoint presentation:

```
jazz.external.hp.com/papers/SolSymposium_03/CIProgramming_files/frame.htm
jazz.external.hp.com/papers/SolSymposium_03/CIProgramming.ppt
```

This tutorial covers all CI programming from the ground up and is very thorough. The topics covered are UDCs and scripts, parameters, variables, entry points, expressions and functions, i/o redirection and file I/O, error handling, script cleanup techniques, debugging and good practices, and lots of examples.

Since Jeff was intimately involved in the coding of the Command Interpreter, he is able to offer valuable insights. For example, he points out that there is overhead in opening and cataloging UDC files, so to make logons faster you can remove unneeded UDCs. In describing the ANYPARM option, Jeff explains where it is essential:

> The only way to capture user-entered delimiters, without requiring the user to quote everything. For example:

```
TELLT  user
ANYPARM msg = “”
# prepends timestamp and highlights msg text
   tell !user; at !hptimef: ![chr(27)]&dB !msg

:TELLT  op.sys Hi,, what’s up; system seems fast!
FROM S68 JEFF.UI/3:27 PM: HI,, what’s up; system seems…
```

> The only way to get an ANYPARM parameter value to default to “” (empty string) is as follows:

```
ANYPARM  p = ![“”] # correct
ANYPARM  p = “”    # wrong - value is literally the two quote marks
```

In 93 slides, Jeff takes us step by step through CI programming and provides numerous interesting and useful samples. For example:

```
alter priority of job just streamed -- great for online compiles  ;-)
PARM job=!HPLASTJOB; pri=CS       “Altp” script
altproc job=!job; pri=!pri
```

He also covers the final CI enhancements in MPE/IX 7.5 (here are some of them):

- new CI functions: anyparm, basename, dirname, fqualify, fsyntax, jobcnt, jinfo, pinfo, wordcnt, xword
- new CI variables: hpdatetime, hpdoy, hphhmmssmmm, hpleapyear, hpmaxpin, hpyyyymmdd
- new CI commands: abortproc, newci, newjobq, purgejobq, shutdown
- enhanced commands: INPUT from console, FOS store-to-disk, :showvar to see another job/sessions' variables, :copy to= a directory, :altjob HIPRI and jobq=, :limit +-N
- :HELP shows all CI variables, functions, ONLINEINFO, NEW

Now on to Ken's chapter:

An overworked, understaffed data processing department is all too common in today's ever belt-tightening, down-sizing and de-staffing companies.

An ad-hoc request may come to the harried data processing manager. She may throw her hands up in despair and say, "It can't be done. Not within the time frame that you need it in." Of course, every computer-literate person knows deep down in his heart that every programming request can be fulfilled, if the programmer has enough hours to code, debug, test, document and implement the new program. The informed DP manager knows that programming the Command Interpreter (CI) can sometimes reduce that time, changing the "impossible deadline" into something more achievable.

Getting Data Into and Out of Files

So you want to keep some data around for a while? Use a file! Well, you knew that already, I'll bet. What you probably didn't know is that you can get data into and out of files fairly easily, using I/O re-direction and the Print command. I/O re-direction allows input or output to be directed to a file instead of to your terminal. I/O re-direction uses the symbols ">", ">>" and "<". Use ">" to re-direct output to a temporary file. (You can make the file permanent if you use a file command.) Use ">>" to append output to the file. Finally, use "<" to re-direct input from a file.

```
echo Value 96 > myfile
echo This is the second line >> myfile
input my_var < myfile
setvar mynum_var str("!my_var",7,2)
setvar mynum_var_2 !mynum_var - (6 * 9 )
echo The answer to the meaning of life, the universe
echo and everything is !mynum_var_2.
```

After executing the above command file, the file Myfile will contain two lines, "Value 42" and "This is the second line". (Without quotes, of course.) The Input command uses I/O re-direction to read the first record of the file, and assigns the value to the variable my_var. The first Setvar extracts the number from the middle of the string, and proceeds to use the value in an important calculation in the next line.

How can you assign the data in the second and consequent lines of a file to variables? You use the Print command to select the record that you want from the file, sending the output to a new file.

```
print myfile;start=2;end=2 > myfile2
```

You can then use the Input command to extract the string from the second file.

[Editor: Jeff Vance points out that using a message file can be 4+ times faster. And for larger files, a method using alternate entry points yields to a performance boost of 31 times or better! These IO techniques are described in his CI programming paper referenced earlier.]

Rolling Your Own System Variables

It's easy enough to create a static file of Setvar commands that gets invoked at logon time, and it's not difficult to modify the file programmatically. For example, let's say that you would like to remember a particular variable from session to session, such as the name of your favorite printer. You can name the file that contains the Setvars, Mygvars. It will contain the line:

```
setvar my_printer "biglaser"
```

The value of this variable may change during your session, but you may want to keep it for the next time that you log on. To do this, you must replace your normal logoff procedure (the Bye or Exit command) with a command file that saves the variable in a file, and then logs you off.

```
byebye
purge mygvars > $null
file mygvars;save
echo setvar my_printer "!my_printer" > *mygvars
bye
```

Whenever you type byebye, the setvar command is written to Mygvars and you are then logged off. The default close disposition of an I/O re-direction file is TEMP, which is why you have to specify a file equation. Because you are never certain that this file exists beforehand, doing a Purge ensures that it does not.

Program Control - If/Else and While

One of the reasons we like programming the CI is its lack of goto's. Consider the following:

```
echo Powers of 2...
if "!hpjobname" = "KARNAK" then
    setvar loop_count 1
    setvar temp 1
    while !loop_count < 10 do
       setvar temp  !temp * 2
       echo 2^!loop_count = !temp
       setvar loop_count !loop_count + 1
    endwhile
else
   echo Sorry, you're not authorized to know.
endif
```

The above command file will display a "powers of two" table from one through nine for the user who is logged on as KARNAK. You can indent code inside If and While blocks for readability, because the CI doesn't care about leading spaces or blank lines. However, you must use the Comment command to insert comments.

There are times when you wish to exit out of a loop in an ungraceful manner. To do this, use the Return command. We often use this when we display help in a command file, and we don't want to clutter further code with a big if/endif block.

```
parm hidden_away="?"
if "!hidden_away" = "?" then
   echo This command file doesn't do much,
   echo but it does it well.
   return
endif
echo Local variable is !hidden_away.
```

Another way to terminate loops and nested command files is to use the escape command. This command will blow you right back to the CI. (Using the Return command only exits the current command file.) You can optionally set the CIERROR jcw by adding a value to the end of the Escape command.

```
escape 007
```

Simulating Arrays

It's true - arrays are not directly supported in the CI. However, because of some undocumented techniques (aka "tricks"), you can simulate arrays.

One question that may immediately pop into your head is "Why would I want to use arrays?" Arrays are useful for table driven events, such as returning days per month, sessions on ldevs, etc.

We won't keep you in suspense. Here's the core method:

```
setvar !variable_name!variable_index   value
```

By using the expression evaluation feature of the CI, you can have a changeable variable name in the Setvar command.

In much the same way, you can use command files to change sequences of commands, i.e., to create self-modifying code. For example,

```
weirdcmd
setvar variable_command "setvar apple ""orange"""
!variable_command
```

If you run the command file, and then display the contents of the variable, you will see:

```
weirdcmd                {execute command file, above}
showvar apple
APPLE = orange
```

To simulate arrays, you must assign a variable per each element. For example, you would assign months_12 for months(12). This variable can be either string or numeric, but keep in mind that string variables can be up to 1024 characters long (256 prior to MPE/iX 5.5).

Here are a few command files that allow you to manipulate arrays.

```
arraydef
parm array_name nbr_elements=0 initial_value=0
setvar array_index 0
while !array_index <= !nbr_elements do
   setvar !array_name!array_index !initial_value
   setvar array_index array_index + 1
endwhile
```

The command file Arraydef allocates variables for each element of the array that you need. The call sequence would be something like

```
arraydef months_ 12
```

Just as you put an index in parentheses, we like to put underbars at the end of array names so that they are more readable.

There is a limit on the storage (it was about 20K bytes in MPE/iX 5.0). The space used can be calculated by adding the number of characters in each variable name, plus 4 bytes per integer item, or the length of each string. For example, the integer variable "XYZZY" takes up 5 bytes for the name, and four more bytes for its integer value. When you run out of space, you get the following error from the CI:

```
Symbol table full: addition failed.  To continue, delete
some variables, or start a new session.  (CIERR 8122)
```

The following command file allows you to return the space used by your pseudo-array when you are finished using it.

```
arraydel
parm array_name nbr_elements=0
setvar array_index 0
while !array_index < !nbr_elements do
   deletevar !array_name!array_index
   setvar array_index array_index + 1
endwhile
```

To demonstrate how arrays can be set (and values returned), the following two command files, Arrayset and Arrayget, use the expression evaluation feature of the CI to convert the element number to the actual variable name. Setvar is called to set either the value, or the name of the variable passed to the command file.

```
arrayset
parm array_name element_nbr=0
anyparm set_value
setvar !array_name!element_nbr !setvalue
```

```
arrayget
parm destination_var array_name element_nbr=0
anyparm get_value
setvar !destination_var !array_name!element_nbr
```

Here's a quick command file to show how you can use arrays in a month table. It uses the Input command to prompt the user for the number of days for each month.

```
setvar months_nbr 0
while !months_nbr < 12 do
   setvar months_nbr months_nbr + 1
   input months_!months_nbr;                        &
   prompt="Enter the number of days in month
                                 !months_nbr:  "
endwhile
deletevar months_nbr
```

More About CI Programming

Ken Robertson's full paper on CI programming:

```
http://www.robelle.com/ftp/papers/progci.txt
```

Eugene Volokh's paper on Classic MPE CI programming:

```
http://www.adager.com/VeSoft/MpeProgramming.html
```

Eugene's paper on MPE/iX CI programming:

```
http://www.adager.com/VeSoft/ProgrammingXL.html
```

File Security: Protection in a High-Risk Environment

By Eugene Volokh, VESoft

About the author: Eugene Volokh is the co-founder of VESoft, the creator of MPEX, Security/3000 and VEAUDIT, the author of many famous HP 3000 papers such as "The Truth About Disc Files" and "Burn Before Reading". Eugene continues as Vice-President of R&D at Vesoft and, in his spare time, is a tenured professor at UCLA Law School, specializing in constitutional law. To read Eugene's thoughts on legal issues (and those of his colleagues), visit his weblog www.volokh.com.

MPE file security is an interesting sort of thing. It is powerful enough to cause a lot of confusion -- what's the difference between :ALTSEC, :RELEASE, and :SECURE? What do R, A, W, L, X, ANY, AC, AL, GU, GL, and CR mean? How do file, group, and account security work? Unfortunately, it's not powerful enough to do a lot of things that you'd very much like to do (such as grant a particular access to a particular user id). If it were only a little bit less fancy or a little bit more fancy, life would be a lot easier...

[Editor: with the addition of ACDs to MPE, it is now possible to specify very precise security by user. This feature was adapted from Unix and was needed to make MPE be Posix-compliant. That being said, few MPE shops use ACDs.]

In What Ways Can You Access a File?

First, a little general discussion. From the file system's point of view, there are five types of access you can have to a file:

- READ -- if you have READ access to a file, you can open the file and read data from it. Simple.
- APPEND -- if you have APPEND access to a file, you can open the file and add records to it. You can't read records (unless you also have READ access) or overwrite them or purge the file (unless you also have WRITE access).
- WRITE -- if you have WRITE access to a file, you can open the file, append records to it, overwriting existing records, or purge the file; you can't read records unless you also have READ access. Note that if you have WRITE access, you will always be allowed to :PURGE the file -- you can't grant a user write access and deny him purge access.
- LOCK -- if you have LOCK access to a file, you can open it EXCLUSIVELY and/or open it shared but then call the FLOCK intrinsic. Lock access doesn't explicitly imply read or write (though it would be pretty useless if you didn't also have read or write access to a file); note, however, that APPEND and WRITE access DO imply LOCK access -- if you have append/write access to a

file, you can always open it exclusively or FLOCK it even if you don't have lock access to it.

- EXECUTE -- if you have EXECUTE access to a program file, you can run it; if you have EXECUTE access to a job stream file, you can :STREAM it. You don't need any other access to do these things -- thus, you may deny a user READ access to a program or job stream and still let him :RUN it or :STREAM it.

You should remember these "access types" because they control what you may or may not do with file security. For instance, as I mentioned, the fact that there is no special "PURGE" access mode means that you can't allow writes but forbids :PURGEs; however, since there IS a special "APPEND" mode you can allow appending to a file (e.g. some sort of log file) but not allow overwriting.

Who Can Access a File?

Now that we've discussed HOW a file may be accessed, we must ask WHO it can be accessed by. Say that you've built a file and want to grant READ and APPEND access to certain people -- how do you do this?

Well, for every type of access (Read, Append, Write, Lock, and Execute, abbreviated R, A, W, L, and X), you may specify a set of "user class specifiers". You may say that the particular access involved is allowed to:

- ANY, meaning any user in the system;
- AC, meaning any user in the account in which the file resides;
- GU, meaning any user who signs on to the group in which the file resides;
- CR, meaning only the user who created the file;
- AL, meaning any user with AL ("Account Librarian") capability in the account in which the file resides;
- GL, meaning any user with GL ("Group Librarian") capability who signs on to the group in which the file resides;
- Any combination of the above;
- None of the above, in which case access is allowed only to a user with SM (System Manager) capability or a user with AM (Account Manager) capability in the file's account. An SM or AM user can ALWAYS do ANYTHING to a file.

Thus, when you say

```
:ALTSEC F;(R,A:ANY;W,L,X:CR)
```

you mean "anybody can Read or Append to the file F, but only the file's creator can Write to it, Lock it, or Execute it". Similarly, when you say:

```
:ALTSEC F;(X:AC;R,L:AL;W,A:GU)
```

you mean "anybody in the account can Execute the file, only users with AL capability can Read it or Lock it, and only users in the file's group can Write to it or Append to it".

File, Group, and Account Security -- The Triple Wall

So, thus far we've discussed HOW a file may be accessed and WHO you can allow to access it. Note that one thing conspicuously missing from our discussion is the ability to say "user JOE can read the file and user JOHN can write to it". This is because MPE provides no such ability. Unless you can arrange the right capabilities (SM, AM, AL, or GL) for the right users, you can't really control access by individual user id *(without ACDs)*.

One other important fact, however (which substantially changes the implications of what we've talked about), has to do with GROUP and ACCOUNT security. Just as you can restrict R, A, W, L, and X access on an individual file, you can also restrict it on a group (which means that the restriction applies to all the files in the group) or on an account (which means that the restriction applies to all the files in the account). In order to access a file, a user must satisfy the file restrictions AND the group restrictions AND the account restrictions. For instance:

```
:ALTACCT PROD;ACCESS=       (R,X:ANY; W,A,L:AC)
:ALTGROUP DATA.PROD;ACCESS= (R:ANY; X:AC; A:GU,AL; W:GU; L:CR)
:ALTSEC F.DATA.PROD;ACCESS= (R:CR; X:ANY; A:AC; W:CR,GU; L:AL)
```

What are the true access restrictions on F.DATA.PROD? We must COMBINE -- "AND" together -- all three sets of restrictions:

Access	Account	Group	File	Total
R	ANY	ANY	CR	CR
A	AC	GU or AL	AC	GU or AL
W	AC	GU	CR or GU	GU
L	AC	CR	AL	CR
X	ANY	AC	ANY	AC

As you see, the resultant access rights are as restrictive as the account, group, and file access rights put together. Even though account and group security allows anybody to read the file, file security allows only the CReator to read it -- therefore, only the creator may read it; even though account and file security let anybody execute the file, group security allows only ACcount users to execute it -- therefore, only users in the PROD account can execute it.

Now, we have the theory down; let's have a look at the practice. By default, all accounts (except the SYS) account have access restrictions of

```
(R, A, W, X, L: AC)
```

In other words, by default NOBODY outside an account can access a file in the account, EVEN IF FILE AND GROUP SECURITY ALLOW IT! Similarly, all groups (except PUB groups in any account) by default have access restrictions of

```
(R, A, W, X, L,S: GU)
```

Thus, NOBODY outside a group can access a file in the group, EVEN IF FILE AND ACCOUNT SECURITY ALLOW IT! On the other hand, default file security is

```
(R, A, W, X, L: ANY)
```

Access is allowed to anybody in the system who can pass the group and account restrictions, but as we just saw, with default restrictions this means that only a user in the file's group can access the file in any way!

You can see the serious problems that this causes. Say that I am JOHN.PROD and I want to let MARY.DEV read my file. First of all, I can't just allow MARY.DEV read it (since access rights can only be given to classes of users, such as ANY, not to individual user ids). However, say that I'm willing to let everybody read the file -- I still can't do this! For MARY.DEV to read my file, she must be able to pass file security (which is easy, since by default it is R:ANY), but also group and account security, which by default prohibit any access! To let MARY.DEV read my file, I have to:

- Go to my account manager and ask him to grant Read access to ANY on the group in which the file resides;
- Go to my system manager and ask him to grant Read access to ANY on the account in which the file resides;

Now, after I've gotten two other people involved in this, ALL files in the group in question are readable by ANYBODY (unless I explicitly :ALTSEC all those files to lower their file security, which [even with MPEX] , may be a cumbersome task). I've had to spend a lot of effort, and in the bargain have waived security on a lot more than what I wanted to waive it on.

Thus, we see several problems with file security:

- You can't grant access to an individual user id, only to an entire class of users.
- Default group and account security is such that a normal user must intervene with his Account Manager AND his System Manager to let people access one of his files.
- In the process of granting access to a file, the AM and SM must lower security on the account and the group, THUS WAIVING SECURITY ON ALL THE FILES IN THE GROUP.

- Finally (one thing we hadn't mentioned before), file-level security (implemented using :ALTSEC) is very hard to maintain since whenever you /KEEP the file in EDITOR, the file security is RESET to (R,A,W,L,X:ANY)! Thus, if you leave your group and account security open and rely on :ALTSEC to protect your files, you must make sure that EVERY TIME YOU DO AN EDITOR /KEEP (or re-build the file in any other way), YOU REPEAT THE :ALTSEC COMMAND. If you don't, you'll leave the file wide open.

:RELEASEing files – Why it's bad, Why people do it, When it's OK

Therefore, on the one hand we have far too much security -- group and account security gets in the way. On the other hand, we don't have enough security, since once we waive group and account security, we leave ourselves wide open. To avoid these problems, HP invented the :RELEASE and :SECURE commands, but these have very substantial problems of their own.

Simply put, :RELEASEing a file allows ANYBODY to do ANYTHING to the file. ANY USER ON THE SYSTEM can READ the file, WRITE to it, APPEND to it, LOCK it, EXECUTE it, PURGE it, do whatever he pleases to it. The advantage of this is you can let MARY.DEV read your file without having to change group/account security (since :RELEASE waives ALL security, file, group, and account); the disadvantage is that not only can MARY.DEV read it, but she can also modify it or purge it, AS CAN EVERYBODY ELSE IN THE SYSTEM. Again, you either have too much security (nobody can do anything) or far too much (everybody can do everything).

Unfortunately, the dangers of :RELEASE go way beyond the danger to the now-unsecured file. In particular, if you have A GROUP WITH PM CAPABILITY and you :RELEASE a PROGRAM FILE that resides in that group, then an ordinary, plain vanilla user can OBTAIN SM OR AM CAPABILITY. Naturally, I won't explain exactly how he'll do this, but trust me -- :RELEASEing program files in groups with PM capability (the program itself need not necessarily have PM) is A VERY BAD IDEA. We suggest to our MPEX , users that every night they execute the command

```
:RUN MPEX.PUB.VESOFT
%SECURE @.@.@ (CODE="PROG" and ISRELEASED and GPMCAP=ON)
```

which will find all the program files (CODE="PROG") that are :RELEASEd (SECURITY=OFF) and reside in privileged groups (GPMCAP=ON) and :SECURE them. If you don't do this -- if you let such files to stay on your system -- you can be extremely vulnerable to break-ins.

I personally think that virtually every :RELEASE is a bad idea, since it's very rare that you really want to let ANYBODY do ANYTHING to a file. What you ought to do instead is organize your file security in such a way that it's easy for people to selectively grant appropriate access to other people -- this isn't very simple to do, but I'll give a fairly straightforward recipe shortly. The only other thing that ought to be said about :RELEASE is that IT'S OK TO RELEASE DATABASES (using DBUTIL or MPEX). Since databases are protected from normal file system access by their PRIV file code, and since IMAGE security implements an additional layer of security on top of normal file system security, you can afford to release them. That way, file security will be entirely bypassed and the user will only have to specify the right IMAGE password to be able to access the database.

Structuring Your System Security to Avoid :Releases

We've established that :RELEASEing files is generally not a good thing to do. Unfortunately, with a default security set-up, :RELEASEs are often necessary (the only other alternative being to get BOTH your account manager AND your system manager involved, and then risk the security of other files by changing group and account security). The only way you can really avoid :RELEASEs is by making easier the legitimate things that people may want to do -- allow other people to read and/or write their files.

The first step is to :ALTACCT ALL YOUR ACCOUNTS TO HAVE (R,A,W,L,X:ANY). Is this a safe thing to do? Yes, because group security (by default (R,A,W,L,X,S:GU)) can protect your files perfectly well. The only thing you might be concerned about is that default security for PUB groups is (R,X:ANY;A,W,L,S:AL,GU); thus, the above :ALTACCT read and execute files in your PUB groups. This is probably not a problem (that's usually what PUB groups are for), but if it is, you can just :ALTGROUP your PUB group to deny R or X access (or BOTH) to non-AC users.

Now that your account security is waived, you can control file security by groups. I recommend that you build a group in each account called READANY:

```
:NEWGROUP READANY;(R,X:ANY;W,A,L,S:CR)
```

As you see, anybody can Read or Execute files in this group, but only the file's creator can modify them. Say that I want to let MARY.DEV read that file of mine -- all I need to do is say:

```
:RENAME MYFILE.EUGENE, MYFILE.READANY
```

Now Mary can read MYFILE but she can't modify it (since the file has W access allowed only to me, the file's creator). Alternatively, I might decide that anybody should be able to read all of my files -- I can then just go to my account manager and ask him to change my group security, without having to get the system manager involved. Then I could, if I want to, further control my file security by using :ALTSECs, but it's unlikely that I will, since file security gets so easily lost every time a file is re-built (e.g. in an EDITOR /KEEP or a re-:PREP of a program file).

Further, if you care about security within accounts, you might well have a group called READAC:

```
:NEWGROUP READAC;(R,X:AC;W,A,L,S:CR)
```

Any file that I want to "show" to some other user in my account can just be moved to the READAC group -- it will then be readable to all account users, but to no users (except those with SM) in other accounts.

[Editor: Here are some additional, detailed references on MPE security:]

Is Your e3000 Environment Secure?

by Mark Bixby.

"Keeping your 3000 safe from hackers until 2006 or beyond"

On-line web version:

```
http://jazz.external.hp.com/papers/SolSymposium_03/homestead-security-027-rev2.htm
```

Download Powerpoint version:

```
http://jazz.external.hp.com/papers/SolSymposium_03/homestead-security-027-rev2.ppt
```

This is a practical nuts and bolts tutorial by a real expert, not an abstract exhortation to better security. For example, it starts right out encouraging you to enable system logging to gather data on potential problems, then shows you the steps involved.

Another thing we like about this tutorial is that it highlights potential problems instead of glossing over them. For example, Mark in his slide on figuring out where a :HELLO attempt came from, he suggests enabling INETD connection logging to track all Telnet connections, but admits that there is currently no way to capture the IP address of a failed VT login attempt.

In many cases, Mark is able to give ingenious solutions to security issues in MPE, such as this script to list all the :RELEASEd files (a major security hole) on your system so that you can :SECURE them:

```
file temp;rec=,,b;disc=2147483647
listfile /,3 >*temp
xeq awk.hpbin.sys "'&
$1 == ""FILE:"" { file=$2 } &
/SECURITY IS OFF/ { print file}'" <*temp
purge *temp;temp
```

And Mark is willing to recommend third-party products when they offer the only solution: "VESoft's VEAUDIT/3000 product does a good job (of auditing who is using SM, OP and PM capability)."

Overall, a very thorough discussion of security issues on the 3000. Topics covered include Security-related manuals, system logging, INET logging (and other logging), listing all program files with PM capability, tracking account changes, packet sniffing, auditing across distributed systems, passwords, file security, networking security, and much more. Highly recommended.

MPE/iX Network Security

by Jeff Bandle.

```
http://jazz.external.hp.com/papers/SolSymposium_03/NWSecurity.htm
http://jazz.external.hp.com/papers/SolSymposium_03/NWSecurity.ppt
```

Jeff has written a general discussion of security followed by details to help ensure that your MPE systems connected to a network are secure. This tutorial is about 1/2 general networking security and 1/2 MPE-specific; it is introductory. When you get to the MPE section, you learn:

> MPE has an advantage because of it proprietary nature. A common type of attack usually will not work or if it did, it would only result in a process abort.
>
> The worst result would be a process abort with a loss of a networking service.

And Jeff shows an example of an INETDSEC.NET.SYS config line to limit Telnet access to your system:

```
telnet         allow   10.3-5 192.34.56.5 ahost anetwork

# The above entry allows the following hosts to attempt
# to access your system using telnet:
#              hosts in subnets 3 through 5 in network 10,
#              the host with Internet Address of 192.34.56.5,
#              the host by the name of "ahost",
#              all the hosts in the network "anetwork"
#
tftp      deny    192.23.4.3
```

Don't forget to read Jeff's notes as well as the slides. Read the Powerpoint version in the normal view; you can resize the slide window smaller and you can also minimize the contents window. In the HTM version, the notes are impossible to read on my monitor, and you cannot resize the slide window. People at HP must work on much larger monitors than I do....

Flexing Underused Muscles

By Bob Green, Robelle

*About the author: Bob Green created Robelle's second software product, Suprtool in 1978 for a client with a large customer database, and a billing program that took **48 hours** to process a small portion of those customers. As the size of HP 3000 applications has grown over the years, Bob has continued to look for ways to get more out of the 3000 system.*

The HP 3000 has an incredible amount of software from HP, from the third-party tools, and software contributed by users. Delving into this software can uncover functionality that you didn't even know you had, functionality that will extend the life and power of the 3000. These tools often lie installed, but unnoticed and unused. For the most part, we continue to use the few tools and techniques that we learned when we first met MPE and IMAGE.

Job Queues - Underused MPE Enhancement

HP adds enhancements to MPE, but many of us haven't bothered to learn them and use them to make our jobs easiser. For example, read this article by Neil Armstrong of Robelle about MPE's job queues:

```
http://www.robelle.com/tips/mpe.html#jobq
```

Best of "Hidden Value" and "net.digest"

John Burke writes two excellent columns in *The 3000 Newswire* that specialize in uncovering little-known functions of the HP 3000. Now his web site contains a "best of" section compiled from columns that appeared in *The 3000 NewsWire* between January 1997 and June 2000:

```
http://www.burke-consulting.com/best_of/best_of_TOC.htm
```

"Best of" has dozens of tips, including simple ones like this:

```
http://www.burke-consulting.com/best_of/6.2.htm
```

My HP e3000 has an internal CD-ROM drive. How can I read information off a CD in this drive?

"Get CDCOPY from jazz.external.hp.com: CDCOPY copies files from CD in ISO9660 format to your HFS file system on disc."

Our favorite "best of" tips is:

```
http://www.burke-consulting.com/best_of/6.29.htm
```

"Slick, low cost document storage and retrieval need not be an oxymoron"

The problem: "We are initiating a project for the storage and retrieval of raster images. Due to budgetary constraints and other factors, we would like to have as low a cost solution as possible. We will have scanned images of documents containing text and graphics in ordinary PCX, TIFF etc formats and want to be able to retrieve them on demand to a Windows95 desktop. Any ideas?"

Neil Harvey supplied a great response as someone who has "been there done that" which I quote in whole:

"My 0.2c worth of advice is to avoid storing the scanned images as BLOBs in some relational database. Instead, scan them and store them as serial numbers in a slashed sub directory structure. Store image number 0010132456.tif under a directory as follows:"

\\IMAGESERVER\Images\00\10\13\24\56.tif

"Each subdirectory will hold only 100 entries, and will be VERY swift to get to. The HP3000 can be used to maintain the serial number sequence. As for the index information, store this in the world's most robust and easy-to-use database called TurboImage."

Underused Functions in Robelle's Suprtool Product

Consider this reasonable user request:

> "Find all students from NY with 3.0 grade point, majoring in English, whose parents are alumni, and who have less than $5000 in student loans."

Once upon a time there was an HP 3000 site that had many user reports like the one above. Although their database was small, it was complex, with many links from the student master to the detail datasets about the student. Some reports like the one described above took two hours or more to run and interfered with interactive response.

Why were these reports taking so long and what could we do to help?

These were straightforward COBOL programs, such as you or I might write. They went serially through the student master, and for each student, they did a keyed DBFIND/DBGET on the each of the related details, looking for the information on that student (i.e., "state" is in the Address dataset, etc.).

Then I had a brain wave.

There must be a faster way to do this. Using Suprtool I knew that I could scan his or her entire database in just a few minutes, looking at every single bit of information about every single student!

Maybe using the indexed DBFINDs was not always the fastest way to retrieve complex related data.

Quickly I put together a Suprtool job that serially scanned each of the five datasets, extracting and sorting the relevant data by "studentid".

Then I spec'ed out a COBOL program to read the five files simultaneously, mashing together records with a matching "studentid", dropping those students who did not have an entry in all five files. The result was a report containing only the students who matched all the criteria.

And the execution time for the entire job was 20 minutes!

How Was Such an Improvement Possible?

How could the "indexed" approach be so slow and the brute-force "serial-sort" approach be so much faster?

Here is the explanation that I came up with. Visualize a hard disk containing the 5 datasets of this database, spread out across the disk space.

When you do indexed DBFINDs and DBGETs by "studentid", you are **causing the read head on the disk to jump all over the drive**, physically moving the mechanism for each call. This means that you are using the disk drive in the slowest way possible: random access.

When you do serial scans of a dataset, each entry retrieved is adjacent to the preceding entry, thus reducing the number of disk accesses needed, sometimes dramatically. For each new entry that you retrieve, you also retrieve the adjacent entries in the same disk page.

And with Suprtool, the contrast is even more vivid, since Suprtool reads about 50,000 bytes of adjacent entries with each disk access. Assume that each entry occupies 250 bytes, then each disk read would retrieve 200 entries!

Suprtool is using the disk hardware in the fastest way possible.

In addition, Suprtool saves the CPU overhead of the database calls, replacing them by a few file system calls and some highly optimized sort operations.

Suprlink Is Born

This experiment reduced run times by up to 6 and led to **one of those under-used third-party tools: Suprlink,** designed to speedily link related data.

For years, people had been telling us "We love Suprtool's speed, but couldn't we have multiple dataset extracts too?"

What they were really asking for was **fast multi-dataset extracts**, because they were accustomed to Suprtool's high speed. We resisted adding regular indexed lookups to Suprtool because we knew the result would often be slow performance.

With Suprlink we gave them multi-dataset extracts and a big performance improvement at the same time. We made Suprlink an integral part of our Suprtool product, and every site that has Suprtool also has Suprlink.

Suprlink is a unique program - it allows you to tradeoff temporary disk space (which is cheap) for clock time, which is always in short supply.

With Suprlink, you use Suprtool to serially scan the tables for your selected students, creating a sorted Self-Describing file from each table. Then Suprlink reads all the extract files simultaneously and merges the related fields together into a new, wider record with all the fields. It is amazingly fast.

Many of you already have Suprlink on your system, since you are already Suprtool users. But you may never have used it. Quite often, revising two of your slowest reports to use Suprlink can make it seem like you have upgraded the server - especially if the reports must run during the workday.

Taking better advantage of the tools available to you can **extend the life of your HP 3000 system.** P.S. Suprlink makes just as much sense on HP-UX as it does on MPE. So when you do eventually migrate one of your applications, you do not have to change the logic and coding of the extracts.

More Underused Muscles

HP 3000 web sites have dozens of "muscles" you could be exercising.

Email the Database Capacity Report to Yourself

```
http://www.robelle.com/tips/remote-dba.html
```

Neil Armstrong of Robelle starts this article on how to email a report to yourself from the HP 3000:

> While consulting at an Open Skies site recently, I heard consistently that people were forced to manually monitor database loads to ensure that datasets did not overflow or performance decline. My response was:
>
> "*What?, this is MPE, the HP 3000, we can automate anything.*"

The email program that I used is from the Telamon website and is available from their ftp download section under:

```
ftp://ftp.telamon.com/dist/
```

If you use your browser, go to www.telamon.com and look under "Download Support Files", then "Browse the Public FTP Directories", and "MPE_freeware". The file you want is either mail.nm or mail.nm.wrq -- download in the format that works best for you.

Jazz to the Rescue

The HP 3000 enthusiasts inside Hewlett-Packard set up a special server called "Jazz". Among other things, Jazz contains a comprehensive directory of open-source tools that have been ported to MPE:

```
http://jazz.external.hp.com/src/
```

For example, there are links to two versions of the GNU tool **wget**, which retrieves web pages. You need an internet connection, but do not need a browser, and wget runs nicely in batch jobs. This lead directly to an interesting technical note at Robelle on using wget to verify domain names for email addresses stored in an IMAGE database:

```
http://www.robelle.com/tips/url-wget.html
```

The dozens of utilities available through Jazz provide you with unlimited muscle power.

Do Migrating Secondaries Give You Migraines?

By F. Alfredo Rego, Adager

About the author: Alfredo Rego created Adager to alter the structure of IMAGE databases and has been the most dedicated advocate of this reliable and cost-effective database ever since. The Adager web site, www.adager.com, contains dozens of useful papers on IMAGE!

What are migrating secondaries?

Migrating secondaries are associated with the approach (called "hashing") that IMAGE uses to store (and retrieve) entries in master datasets.

What is hashing?

Hashing allows you to quickly access a specific master entry (potentially among billions of entries) according to the value of the entry's search field (or master "key"), regardless of the entry's location in the master dataset.

IMAGE's master datasets are optimized for hashed access. For online applications, which usually serve people who are impatiently waiting over the counter or over the telephone, this kind of quick access provides excellent performance, most of the time. For some caveats, please see Fred White's papers at

```
http://www.adager.com/TechnicalPapers.html
```

Like everything else in the universe, the advantages of hashing come tightly coupled with disadvantages. For instance, we have the very real possibility of synonyms (entries with different search-field values which, nevertheless, hash to the same location). Even though mathematicians have written countless dissertations on the desirability of "perfect" hashing algorithms that would not produce any synonyms at all, every hashing algorithm known today produces synonyms.

Are there other methods for fast storage and retrieval?

If you look on the Web for "indexing", you will be overwhelmed by the breadth and depth of the subject.

IMAGE uses a few specific approaches, listed here in order of sophistication:

- Serial access (DBGET modes 2 and 3). Compact datasets, without "holes" in the middle, perform best under serial access.
- Direct access (DBGET mode 4). This is the quickest way to get to any entry on any dataset (master or detail), provided that you know the exact address. This is the "fundamental" or "atomic" access method used — deep down inside — by all other methods.

- Chained access via doubly linked lists (DBGET modes 5 and 6). A chain consist of entries whose search-field values are equal (for details) or have the same hash value (for masters). Detail chains can be kept sorted via a sort field. You improve chained-access performance by repacking a dataset so that each chain has its entries in contiguous locations.
- Hashing (DBGET modes 7 and 8). Applicable only to masters.
- B-Tree indexing. Applicable only to masters.
- Third-Party Indexing (TPI). Applicable to masters and details. In addition to tree-based indexing, TPI supports keyword retrieval on any field ("native" IMAGE access methods apply only to search fields).
- "Hashing" and "indexing" are just two examples (among many) of techniques used by designers of database management systems to try to retrieve (and to store), as quickly as possible, the entries (or rows, or records) of interest for a given query.

Is hashing better than indexing (or vice versa)?

There Is No Such Thing As A Free Lunch (TINSTAAFL) or, if you prefer, *There Ain't No Such Thing As A Free Lunch* (TANSTAAFL).

Hashing comes with migrating secondaries and the need to rehash the entire master dataset when changing its capacity. Fortunately, the issue of migrating secondaries turns out to be a non-issue most of the time. For instance, secondaries don't get migrated to the moon — they usually get relocated within the same memory cache and the cost of doing so is negligible.

Fortunately, also, most synonym chains are short (in the single digits). So, even with synonym-chain chasing, you will probably stay within your memory cache if your "data locality" is good.

The tree-like structures associated with indexing come with node splitting, balancing, and reorganization. "Bushy" trees perform better than "stringy" trees. Trees with fewer levels perform better than "tall" trees. With tree-based indexing, you also duplicate the search-field data (you keep one copy in the dataset and one copy in the index). It's a jungle out there!

The minimum number of disc accesses for tree-based indexing is two (one for the tree index, one for the dataset). The minimum number of disc accesses for hashing is one (and, most of the time, this is all it takes).

A tree has several levels (more or less, depending on statistical considerations) that must be traversed via pointers.

So, is there a perfect way to store and retrieve database entries? Unfortunately, the answer is no.

What is a synonym chain?

A synonym chain is IMAGE's method of managing synonyms (members of the class of entries whose search-field values hash to the same location).

An entry which hashes to an empty location (or to a location temporarily occupied by a secondary) becomes a ***primary***. A primary "owns" its location and can't be evicted (i.e., it never migrates to a different location).

An entry which hashes to a location that is already occupied by a primary becomes, *ipso facto,* a ***secondary***. A secondary has to find a temporary location elsewhere and is always subject to "eviction" (if a new entry hashes to the secondary's temporary location) or "promotion" (if the primary that currently occupies the secondary's proper location gets deleted and this secondary becomes a primary by taking its place).

The primary has an important job: it serves as the synonym chain's head, which contains information regarding the total number of synonyms in the chain (if any) and keeps track of the first and last members in the chain. A secondary only has to worry about its predecessor (if any) and its successor (if any).

Why do some secondaries migrate?

Some secondaries migrate because primaries (new or old) have top priority.

Fortunately, not all secondaries migrate. The rules for migrating secondaries are quite clear:

Whenever a new legitimate primary hashes to some location which was previously "borrowed" by a secondary, IMAGE migrates the secondary elsewhere, to make room for the new primary.

Whenever IMAGE deletes a primary with secondaries, it migrates the first secondary and promotes it to the primary location, to take over the ChainHead functions.

Are all secondaries "bad"?

No. There are "good" secondaries and "bad" secondaries, according to their demands for a very valuable computer resource with direct impact on performance: disc access.

Good secondaries are those which we may access directly in memory buffers, without having to go to disc. Bad secondaries are those which force excessive disc activity.

IMAGE takes excellent advantage of the optimized disc-access methods provided by MPE/iX. It is a good idea to invest in as much main memory as possible, to minimize time-consuming trips to/from disc.

Messy synonym chains, with entries scattered all over, will probably contribute numerous bad synonyms. Cleanly-packed synonym chains, on the other hand, may contribute good synonyms which will be, for all practical purposes, equivalent to primary entries. Intra-memory operations are, after all, significantly faster than disc operations.

Under any circumstances, short and tidy synonym chains are much better than long and messy synonym chains. Use Robelle's HowMessy (available in the Robelle and Adager installation packages) to get a good view:

```
http://www.robelle.com/smugbook/howmessy.html
```

Why do secondaries (migrating or not) give you headaches?

There are three fundamental operations on entries:

- addition
- deletion
- finding

We want to do these operations as quickly as possible. Therefore, we want to avoid excessive disc accesses. Unfortunately, secondaries (migrating or not) tend to increase disc activity. As a consequence, the response time for online applications may deteriorate and the throughput of batch jobs may decline.

Quick review: Adding and deleting entries.

If a new entry's appointed location is vacant, we are home free. We just add the new entry on the spot and mark it as "occupied". The new primary entry will have tenure for life.

If a new entry's appointed location is already occupied, we must do a lot of work. There are two possibilities:

The current occupant *is* at its appointed place and, since it arrived before the new entry, it has "seniority" and tenure for life. The new entry, sadly, becomes a synonym and we must place it elsewhere as a secondary, linked to the primary by means of a synonym chain. Before we add the entry, though, we must make sure that it will not duplicate an existing entry's search-field value. We must, therefore, scan the whole synonym chain before we can add the new entry. If the synonym chain is long, this may take a while.

The current occupant did *not* hash to this location but was placed here as a secondary (subject to migration). Sorry: its time to migrate has come and must go elsewhere. After we evict the secondary, we can place the new entry in its appointed spot, as a primary.

Notice that the innocent-sounding expression "must go elsewhere" is easier said than done. Finding the "elsewhere" may take a long time if the dataset has a long, uninterrupted cluster of occupied entries. This usually happens if the dataset is quite full or if the distribution of search-field values pushes the limits of IMAGE's hashing algorithm.

Having found the "elsewhere" does not guarantee tenure there, since any secondary is subject to migration in the future. If we add a new entry which hashes to a spot occupied by a secondary, we must migrate the secondary elsewhere. If we delete a primary with secondaries, we must move the first secondary into the spot previously occupied by the deleted primary (since we *must* have a primary). IMAGE, under extreme circumstances, may spend a significant amount of time chasing its own tail while migrating secondaries all over the place.

Quick review: Finding existing entries.

Even in a static dataset (where we never add or delete entries), we may have performance problems when we simply want to find an existing entry.

If the entry happens to be at its appointed location, we are in good shape. If the entry is not at its appointed location, there are two possibilities:

The appointed spot is empty, in which case we know, immediately, that our entry does not exist (this is a valid "entry not found" result in a "find" task).

The appointed location contains some other synonym entry (which happens to hash to the same spot as the entry which interests us). Since this entry is the primary in a synonym chain, it keeps track of the number of synonyms. If there are no synonyms, we know that our entry does not exist. If there are synonyms, we must scan the synonym chain until we either find our entry or exhaust the chain. In either case, we may have to go to disc more than once (depending on the length of the chain, the messiness of the chain, the dataset's blocking factor, and so on).

What can you do about nasty secondaries?

If you are a theoretical type, you can spend endless pleasant hours dedicated to the fine art of figuring out the ideal mix of capacity, percent-full, search-field value distribution, and so on. This is a worthy endeavor, of course. But the sad thing is that your dataset, most likely, is *dynamic*. The minute you add or delete a few thousand entries, most of your glorious conclusions become invalid.

If you are a practical type, you might just as well accept reality, bite the bullet, and periodically repack your master datasets to balance the synonym load (just as you periodically repack your detail datasets, your disc drives, or the drawers in your desk). If your dataset is static, you are in luck: you just repack once.

As another alternative, you can enable your dataset for master dynamic capacity expansion (MDX). See Fred White's chapter on this topic later in the book.

If the widths of master ventilation shafts (free areas reserved for future secondaries) are narrow and suffocating, you may consider changes in the dataset capacity which, together with dataset repacking and MDX, may improve the situation.

An ideally-packed master dataset allows sufficient ventilation space for DBPUT's sake (so that it does not take forever to find an empty spot for a new secondary or for a migrating old secondary) without having so much empty space that a serial scan (or a backup) will take forever.

Furthermore, the ventilation space has to be intelligently distributed throughout the entire dataset.

The good news and the not-so-bad news.

Fortunately, it turns out that repacking a master dataset is a very efficient operation. Smart repacking of a master dataset takes only slightly longer than an optimized serial dataset scan.

Unfortunately, repacking does not seem to be an *a priori* activity. We can only repack master datasets *a posteriori.* We must repack *periodically*, and (unless we quit adding or deleting entries) we must keep repacking *ad infinitum.*

The same applies, of course, to tree-based indexing methods. The price of good performance is constant fine-tuning.

Buffing Your IMAGE

By Bob Green, Robelle

About the author: Bob Green's first software product was an add-on for HP's IMAGE database: "Malkin & Pinton Transaction Logging" that he wrote for an HP site. Once he had his own firm, Bob lost no time writing another IMAGE add-on, Suprtool, to extract and sort data from databases 10 to 20 times faster than previously possible. Then he wrote a paper to promote Suprtool called "Faster Batch Jobs" and presented it at a user group meeting. He has been writing about IMAGE ever since.

Paths Are For People

Imagine you have a customer order tracking system and you need a weekly report on sales by salesman. Many programmers automatically think of making the salesman code into a TurboIMAGE search key in order to optimize the report. Is this a good idea?

First a little background information, which may be review for many readers.

Path

A *path* is the structural relationship between a TurboIMAGE master dataset and a TurboIMAGE detail dataset linked through a common search item.

For example, the customer master dataset provides the index for unique customer names as well as the customer name and address. The line item detail dataset provides information on what a customer has ordered: customer number, item number, quantity, price, order number, salesman code, date ordered, date shipped, etc. There can of course be multiple line items per customer, and usually over time there are many. A **path** between the two datasets allows you to retrieve the line items for a specific known customer number.

Chain

A *chain* is a collection of detail entries that have the same search value; the head of the chain is in the master entry for that search value. The head of the chain points to the first detail entry with the search value, and the last one. Each detail entry has a forward and backward pointer, so retrieving the list of entries is relatively easy and fast (a maximum of one disk read per record). This is implemented in TurboIMAGE as a DBFIND followed by chained DBGETs (mode 5).

Every path you add to a TurboIMAGE dataset adds overhead and slows response time, because the pointer chains must be changed for each DBPUT and

DBDELETE into the detail dataset. The more paths, the more chains to update. And this overhead usually occurs at the worst possible time: when the end-user is entering or amending an order.

To decide how many inquiry paths you should have for a dataset, you need to know the volatility of the dataset: how often do people ask questions about the data (read access) versus how often they update the search keys in the data (write access)? The read/write ratio tells you whether the dataset is stable or volatile.

If the read/write ratio is **low**, you should not have many search paths, because the overhead of updating them during the many writes will overwhelm the benefit during the few reads. For example, a dataset that contains active customer orders is probably active.

However, if the read-write ratio is **high** then the dataset is more stable (i.e., it doesn't change that quickly). You can afford many more search keys in such a dataset. For example, a dataset that archives completed customer orders is probably very seldom (if ever) updated. Spending a second or more to write a record into this dataset makes sense, because it gives you many search keys per record and is only done once.

Suppose you only write records 500 times a day: the total CPU time will be less than 10 minutes each day. And even if the system is called upon to search for over a million records a day, the heavy use of keys will make the overall application work better. The only way to be confident that you have properly indexed your data is to measure, measure, measure.

The purpose of a path is to divide a dataset into small, manageable subsets of entries that can be retrieved, displayed, and updated easily for real people who are sitting at workstations!

A path should divide a dataset into many short chains, rather than a few long chains.

A report that is only run weekly may be marginally faster with a custom search key, but the extra search keys make the on-line usage slower during the entire week. Paths are never absolutely necessary. You can accomplish the same inquiry by a serial scan of the entire dataset. The only reason for a path is to make the retrieval faster than a serial scan.

Our weekly report may actually be slower with the search key, since it may need to read all or most of the dataset anyway, in which case a serial scan is almost always the fastest approach.

If the number of unique search values is less than the number of entries per disc page, a serial scan will be faster than a chained retrieval. That is because a chained read takes one disc read, while a serial read takes only 1/N disc reads (where N is the blocking factor per page).

It seldom makes sense to add an inquiry path strictly for a batch program. Paths should be reserved for on-line users.

Serial Dataset Scans: Good and Bad

A serial scan is an access method that reads an entire dataset in physical order to find what is wanted, rather than following a search key maintained for that dataset. MPE programmers are usually taught that TurboIMAGE search keys are efficient and serial scans are to be avoided, but that is an over-simplification.

There are situations where a serial scan is faster than a keyed search, and vice versa.

Good Serial Scans

Question:

What are two ways to perform the following data retrieval?

We have a large detail dataset called *Ord-Line* with 2,307,685 records. We have a list of 162,770 *Ord-Num* key values of interest. We want to extract the *Ord-Line* records for those keyvalues and we want them sorted by *Ord-Num*. The *Ord-Line* record size is 308 bytes.

Answer:

The traditional solution in TurboIMAGE is to call the **DBFIND** intrinsic for each of the 162,770 *Ord-Num* key values, then repeatedly call **DBGET** mode-5 (directed access) to retrieve the *Ord-Line* detail entries on that key-value chain (261,230 in all). If the list of *Ord-Num* values is sorted, the *Ord-Line* records will also be sorted.

An alternative solution is to call **DBGET** mode-2 (serial access) for each of the 2,308,685 detail entries (or use Suprtool's Get Command), select out the 261,230 records that match the *Ord-Num* table, then sort those records.

That sounds like a lot of work, but do we know that it will be slower than the traditional search-key method?

The traditional method takes about 162,770 **DBFIND** disc reads and 261,230 **DBGET** disc reads, for a total of 424,000 disc reads. The reason why it consumes about one disc read per record accessed is that the access is random and direct - when you retrieve one record the chances that the next record retrieved is adjacent is extremely tiny.

If you do the serial scan with **DBGET** it takes at most 177,514 disc reads. The reason is that each disc read transfers a page containing 13 records. Suprtool uses a 50,000-byte buffer and reads 159 records of 308 bytes per disc read. If you use Suprtool to do the task, it will take only 14,500 total disc reads*. **The Suprtool serial scan reduces disc reads by 96.6 percent.***

Suprtool can do serial scans much faster than other tools and application programs because it bypasses the overhead of TurboIMAGE by doing NOBUF/MR access to the underlying file of the dataset.

Bad Serial Scans

Question:

Will a serial scan be faster than keyed access if you have 1 million entries in an TurboIMAGE master dataset and you need to retrieve 1000 key values (which you have in a table). The entries are 400 bytes long and are packed 10 records per page in the master dataset.

Answer:

Keyed access takes about 1000 disc reads, since master entries are stored randomly by hashing the key value. Serial access takes about 100,000 disc reads, since they are packed 10 per page. Serial access with Suprtool takes about 6,500 disc reads, since Suprtool reads 12 pages per access. So even with Suprtool, the ***keyed access is much faster because we are reading a small portion of the dataset.***

Question:

You have deleted 90,000 old entries from a dataset with a capacity of 110,000. Why does the time to serially read the dataset remain the same?

Answer:

Because TurboIMAGE must scan the empty space in the dataset to find the remaining entries. There are no pointers around empty chunks in a dataset.

In a master dataset the entries are distributed randomly over the entire capacity and there is no way to find the next occupied entry after **N** except by looking at **N+1**, **N+2**, and so on, until you find it. *The time to read a master dataset is proportional to the capacity, not to the number of entries.*

In a detail dataset, the entries are stored chronologically adjacent to each other. As you add entries to a detail dataset, it moves up an EOF marker called the *highwater mark*. Serial scans must read all the space up to the highwater mark. When you delete an entry, it is put on a *delete-chain* for reuse on the next add operation, but the highwater mark is not necessarily reduced. *The time to read a detail dataset is proportional to the highwater mark, not to the number of entries.*

Moral:

Use a dataset packing tool such as Detpack from Adager after you clean out a detail dataset. This will reduce the highwater mark by repacking the entries. For a master dataset, reduce the capacity if you won't need the empty space in the near future.

Internal IMAGE Structures

IMAGE is the database of the HP 3000 system, and some knowledge of how it works is useful for anyone doing IT work on the system.

"Master Datasets" are one of the storage types of IMAGE and their job is to store records when there is one record expected per unique key field (also the search field). For example, one customer record per customer number.

IMAGE uses a mathematical formula called a hashing algorithm to transfer the key value into a record number within the master dataset. This "hash location" is the first place where IMAGE tries to store the record. It might be a good idea to read the previous chapter by Alfredo Rego at this point (See "Do Migrating Secondaries Give You Migraines?" on page 124).

Figure 1 below shows a hashing algorithm with a collision - same block.

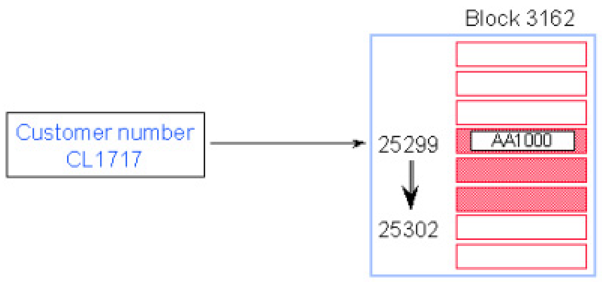


Figure 1. A Collision

- Customer number CL1717 hashes to the same record number as AA1000 location.
- IMAGE tries to find an empty location in the same block. If it finds one, no additional IO is required.
- CL1717 becomes a secondary entry. Primary and secondary entries are linked using pointers that form a chain.

Although hashing algorithms are designed to give unique results, sometimes IMAGE calculates the same record number for two different search item values. In this case a new entry may want to occupy the same space as an existing Primary entry. This is known as a collision, and the new entry is called a Secondary.

Most master datasets have secondaries. If you have a lot of secondaries, or if they are located in different blocks, you should be concerned about generating more IOs than needed.

In Figure 1, customer number CL1717 collides with AA1000, and CL1717 becomes a secondary. IMAGE then tries to find the closest free record location. In this case, the next empty record number is 25302, which is in the same block. No additional I/O is needed.

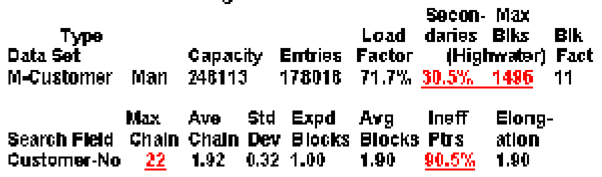
IMAGE uses pointers to link the primary entry with its secondary. Record 25299 for AA1000 has a pointer to record number 25302 and vice versa. This is known as a synonym chain.

Several programs can help identify inefficiencies in IMAGE databases. One, a program written by Robelle and distributed to Robelle and Adager customers, is called HowMessy. There is a very similar contributed program, Dbloadng, upon which HowMessy was patterned. Another option is the report feature of DBGeneral.

All these programs produce a report that shows the database's internal efficiency. Databases are inefficient by nature. Unless you never add new entries or delete existing entries (e.g., use your database for archival purposes), your database becomes more inefficient over time.

In the rest of this column we will examine the HowMessy/Dbloadng reports for two example master datasets.

Figure 2. M-Customer Dataset



Data Set	Type	Capacity	Entries	Load Factor	Secon-daries	Max Blks (Highwater)	Blk Fact
M-Customer	Man	248113	178018	71.7%	30.5%	1496	11

Search Field	Max Chain	Ave Chain	Std Dev	Expd Blocks	Avg Blocks	Ineff Ptrs	Elong-ation
Customer-No	22	1.92	0.32	1.00	1.90	90.5%	1.90

In a Manual Master Dataset (see Figure 2, above):

- The number of secondaries is not unusually high.
- However, there may be problems:
- Records are clustering (high Max Blks).
- Long synonym chain.
- High percentage of Inefficient Pointers.

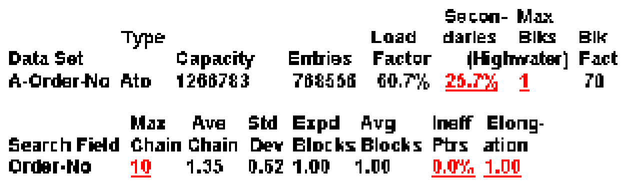
The values in Figure 2 are understood as follows:

- Secondaries: Percentage of entries that are secondaries
- Max Blks: Maximum number of contiguous blocks IMAGE has to read before finding a free location for a secondary. This is the worst-case scenario, not the actual number.
- Blk Fact: Blocking Factor column indicates the number of records in each block (not an MPE/iX page).
- Search Field: Key item name.
- Max Chain: Maximum number of entries in the longest synonym chain.

- Ave Chain: Average number of entries in the synonym chain.
- Std Dev: Standard deviation shows the deviation from the average that is required to encompass 90% of the cases. If the deviation is small, then most chains are close to the average.
- Expd Blocks: Number of blocks we expect to read in the best-case scenario (usually 1.00 for master datasets) to get a whole synonym chain.
- Avg Blocks: Number of blocks needed to hold the average chain.
- Ineff Ptrs: Percentage of secondary entries that span a block boundary.
- Elongation: Average number of blocks divided by the expected number of blocks.

Figure 2 does not show a tidy dataset. The number of secondaries is fairly high (30.5 percent, or around 54,000). The Maximum Chain length is 22. This is unusual for a master dataset. The high Max Blks value (1,496) suggests some clustering. All of this indicates that the hashing is not producing random distribution. If you add a new record whose primary address happens to be the first block of the contiguously-occupied 1,496 blocks, IMAGE will have to do many extra disk IOs just to find the first free entry for the new record (assuming that MDX has not been enabled on this dataset).

Figure 3. A-Order-No Dataset



Data Set	Type	Capacity	Entries	Load Factor	Secon-daries	Max Blks (Highwater)	Blk Fact
A-Order-No	Ato	1266783	768556	60.7%	25.7%	1	70

Search Field	Max Chain	Ave Chain	Std Dev	Expd Blocks	Avg Blocks	Ineff Ptrs	Elong-ation
Order-No	10	1.35	0.62	1.00	1.00	0.0%	1.00

Figure 3 above shows an Automatic Master Dataset.

- This is a very tidy dataset:
- Number of secondaries is acceptable.
- Max Blks, Ineff Ptrs and Elongation are at the minimum values, even if the Maximum Chain length is a bit high

Although the percentage of secondaries is similar to that of the M-Customer dataset of Figure 2 (30.5 percent versus 25.7 percent), the longest synonym chain has 10 records with a 1.35 average. The Inefficient Pointers and Elongation are at their lowest values. Max Blks of 1 means IMAGE can find a free record location within the same block every time it searches for one.

It means the hashing algorithm is working correctly, dispersing records evenly across the dataset.

Can you increase your capacity too much?

You have probably heard that increasing the capacity of a TurboIMAGE master dataset will improve the performance. So wouldn't increasing the capacity a lot be really good for performance?

First, let's review why increasing the capacity can improve the performance of Puts, Deletes and Keyed Retrievals. For this we need to understand how entries are stored in master datasets.

I like to think of master datasets as **magic dartboards** with the darts as your data entries. (Actually, I am only thinking of master datasets with ASCII-type keys, since they are the only ones that use hashing. If the key field is of a binary datatype, things happen completely differently.)

You start with an empty dartboard and you throw the first dart. This is like adding your first entry?

The dart hits the board at a "random" location (this is the "hashing) which becomes it's "primary" location, the place where it "belongs". The magic part of the dartboard is that if you throw the same dart again, it always goes to the same spot. This allows you to find it quickly. In TurboIMAGE terms, you perform repeatable mathematical calculations on the unique key value to produce a "hash location" in the master dataset. This is where that dart always lands.

Does the next dart land right next to the first one? Not likely.

The next dart lands at a new random location and the next and the next. If the hashing is working properly, the darts are spread randomly across the board. As the number of darts increases, the board gets filled and it becomes more likely that one of the darts will hit an existing dart!

When a dart wants to be in the same primary, this is called a collision and leads to placing the dart in a nearby "secondary" location. It takes more work to create, find and maintain secondary entries than it does primary. As the number of entries approaches the capacity of the master dataset, the percentage that are secondaries will increase (assuming that the distribution is actually random, which is sometimes isn't, but that is another story).

Note: TurboIMAGE tries to put colliding entries nearby to their primary location. For this reason, performance only degrades seriously when all the locations in that page of disk space are full and TurboIMAGE has to start reading and examining neighboring pages.

What happens when you increase the capacity of the master dataset?

You must remove all the darts and throw them again, because now each will have a new magic location. In TurboIMAGE terms, you need to start with a new empty master dataset and re-put all the entries. Because there is more space and the hash locations are random, the entries will spread out more, with fewer collisions!

This explains why increasing the capacity usually improves the performance of DBPUT, DBDELETE and DBGET mode 7 (keyed).

Now back to our original question: if increasing the capacity by 50% reduces collisions and improves performance, wouldn't increasing the capacity by 500% improve it even more?

The answer is "Yes and No"!

Yes, most functions will improve by a very small amount. However, once you have reduced the number of secondaries so that 99% of them reside in the same disk page as their primary, you can't improve performance anymore. The minimum time it takes to find an entry is one disk read – once you reach that point, adding unused capacity is pointless.

But there is a more important "No" answer.

If you increase the capacity of a master dataset to create a lot of empty space, at least one function, the **serial scan**, will be **much** slower.

A serial scan is a series of DBGET mode 2 calls to look at every entry in a dataset. It is used when you need to select entries by some criteria that is not the key field. For example, you want to select all the customers in the 215 area code, but the key field is customer-id.

Each DBGET mode 2 call must find and return the "next" physical entry in the master dataset. If the dataset is mostly empty, the next physical entry could be far away. But the DBGET cannot complete until it has read all the intervening disk space!

To read all the entries in the master dataset, DBGET must read every page in the dataset until it reaches the last entry. If you increase the size of a master dataset from 10,000 pages to a 100,000 pages, you have made serial scans take 10 times as long!

So increasing the capacity of a master dataset drastically above the expected number of entries is usually a very bad idea.

Note: TurboIMAGE has a new feature called Dynamic Dataset Expansion (MDX), which allows collisions to be put in an overflow area rather than in an adjacent free secondary location. This can be helpful in cases where the entry values are not hashing randomly or you need a quick fix but don't have time to shut the database down to do a capacity change. See the chapter on automatic dataset expansion on page 146.

Will adjacent detail entries have the same key value?

Earlier, we learned the mechanics of IMAGE master datasets and how performance is impacted by the percent emptiness or fullness of the dataset. Now we explore **detail datasets** and one of their performance factors.

What is the likelihood that entries with the same key are located at adjacent locations in a detail dataset? That would be very convenient and efficient.

When we retrieve the entries for a customer from a sales detail dataset, we do a DBFIND by the customer number, then a DBGET mode 5 to read each detail entry for that customer. Our natural tendency is to think that since these detail entries are logically "grouped" they are probably physically grouped as well. If they are, then retrieving them should be very quick. However, if they are likely to be physically far apart on the disk hardware, then performance will be much less.

The entries we are retrieving have something in common - they have the same key value. They may even be in the same disk page. If so, and there are 20 entries per page (4K bytes divided by 20 equals about 200 bytes per entry), then we would use 1/20th of a disk read per entry. Sounds fast!

But what are the odds that the entries with the same key value are adjacent?

To understand when they will be adjacent and when not, we must understand how entries are stored in detail datasets.

Internally, all IMAGE datasets are organized into logical blocks of entries, with the number of entries per block being the blockfactor. The block organization is a hangover from the original design of IMAGE on the Classic HP 3000 computers (MPE V). On MPE/iX, IMAGE actually performs page-level I/O. A page has 4,096 bytes. This means that one or more IMAGE blocks are read or written as part of each I/O operation.

Whether dealing with blocks or pages, the goal is to minimize the number of I/Os required to access the database.

Starting from An Empty Detail Dataset

When a detail dataset is empty, IMAGE adds records in the order that they are created, starting from record location one.

Records are stored in chronological sequence. That means records for the same customer are not necessarily stored consecutively, unless they happened to be added at the same time.

Assuming that customer id is a key, then records with the same customer id are linked together with pointers to form a chain. The first and last records in a chain are also linked from the corresponding master record for that customer id.

These chains by customer id are used when you do retrievals by customer id. IMAGE finds the matching entry in the master dataset, then follows the path chain to retrieve the detail entries.

Multiple Search Keys

Remember that only one search path per detail dataset can be optimized. If the customer id path into sales details is optimized, then the part number path CANNOT be optimized. The physical data can only be sorted one way.

It is tempting to add a search path for every kind of search that your application requires, but this is a bad strategy: "paths are for people." They exist so that on-line users can retrieve and examine the entries by that path. Paths are not necessary for reports that run only once a day - you can always use Suprtool and a serial scan instead, but that is another article.

"Entropy" of Detail Datasets

As IMAGE adds more records to the dataset, the chains for any given key value get longer and span more blocks. When records are deleted, their space is added to a "delete chain" and subsequent additions are fulfilled from this list, rather than being added to the end of the dataset.

Therefore, the more active the dataset is, in terms of additions and deletions, the more likely that adjacent records on a chain will be in different blocks. Over time, records for the same customer are scattered over multiple blocks

As a result, IMAGE has to perform a lot of extra disc I/O to read a whole chain for a given customer. In the worst case scenario (but a common one for active datasets), IMAGE does one disc I/O for each record in the chain.

Conclusion: it is possible for related entries to be adjacent, but only if the related entries were added at the same time and no entries have ever been deleted. For any real world dataset where entries are added and deleted dynamically over time, it is most likely that the related entries are not adjacent. A good rule of thumb is to assume an extra disk read for each entry that is retrieved.

Repacking a Detail Dataset

You can often improve performance by repacking a detail dataset. The repacking process usually groups all records along their "primary path" values. In other words, all the records for a customer are put into consecutive locations (if customer id is the primary path). Depending on the chain length, a single disk I/O may read all the appropriate records. The repacking default in Adager's Detpack and DBGeneral's dataset reorganization feature is along the primary path. For various reasons, you may not want or be able to change the primary path. If that is the case, both products give you the option to use any other path in the dataset.

Repacking removes the delete chain. It groups all the records at the beginning of the dataset without leaving empty locations in the middle of blocks.

Because most tools repack a dataset along the primary path, it is essential to pick the right one. A primary path should be the one with the longest average chain length (you can get this figure from Robelle's HowMessy report), that is also accessed the most frequently.

There is nothing to gain by selecting as your primary path a path with an average chain length of 1, since there is never another record with the same key value to retrieve.

Repacking only improves statistics for the specific path that is repacked. Other paths will usually not be improved in efficiency by a repacking.

Is a "Zero Secondaries" Report Good News?

If secondary entries slow down a master dataset, would a "zero secondaries" be ideal?

No! Just the opposite.

If your master dataset has no secondaries, it means that the key values are not hashing **randomly**. Instead, they are hashing into some regular pattern (usually into adjacent locations). Here is what a proper "random" dataset looks like, with P for primary and S for secondary:

```
---------------------------------------------------
|   P  PP   PS  P P        PSS  P   P    P     P  |
|PP  P P   P   PS    P  P    PS   PSS      PPP    |
| P   PS   PPP      P     P P P    PP PP PSS     P|
---------------------------------------------------
```

And here is what a dataset with no secondaries usually looks like:

```
---------------------------------------------------
|          PPPPPPPPPPPP                           |
|         PPPPPPPPPPPPP                           |
|         PPPPPPPPPPPP                            |
---------------------------------------------------
```

In this second example, all the entries are primaries and they are all clustered in one area, with no empty space between them. How could this happen?

The reason for this pattern is usually **binary keys** and transaction numbers.

For example, consider typical invoice numbers that start at 030001 and increase by one: 030002, 030003, 030004, etc. If these invoice numbers are stored in a "Binary" field (J2, I2, P8, etc.) instead of an "Ascii" field (X, Z), IMAGE disables random hashing and calculates the primary location as follows instead:

Divide the rightmost 32 bits of the binary value by the dataset capacity, and use the remainder as the primary location (dividing by the capacity ensures that you produce an entry number between 0 and the capacity).

If you try a few examples, you will see that monotonically increasing key values produce adjacent entry numbers. Assume that the capacity is 1000:

```
030001      =>  1
030002      =>  2
030003      =>  3
```

What is so bad about that? It seems perfect. Each DBFIND or keyed DBGET will take a maximum of one disk read to find each key value!! Very fast.

Unfortunately, the bad part happens when you change the sequence of invoice numbers to 040001, 040002, 040003,...

```
040001      =>  1
040002      =>  2
040003      =>  3
```

It is quite possible that the new invoice numbers will hash to the same location as the last batch. This causes a **collision**. The primary location is already occupied, and DBPUT must find a free space for a secondary entry. But the next physical entry is also occupied. In fact, there may be no free entries for quite a while.

The result of this wraparound collision is that DBPUT may have to search thousands of disk pages. And the next DBPUT, will look at the original location + 1, then do the long search completely over again! If this happens to you, it can freeze your database and your entire application.

What can you do?

1. Enable Master Dataset Expansion (MDX) on this dataset. I have never actually done this, but it should help quite a bit, because after searching a short time for an empty space, DBPUT will then go to the overflow area instead. However, this solution is not permanent. The overflow entries are on a single chain that will get longer and longer as you add entries, slowing down inquiries. Before that happens, you should implement one of the other solutions below.
2. Increase the capacity of the master dataset and hope that the clusters do not overlap with the new capacity.
3. Change the key field from a binary type to an ASCII type, such as X or Z. This of course means changing all the COBOL copylibs and recompiling all the programs, which may be quite time-consuming.
4. Convert the application to Eloquence on HP-UX. Since it does not have hashing, Eloquence does not have these problems.

Rene Woc of Adager adds this note:

I believe there is a 5th solution, one that guarantees no-secondaries with integer keys, if you know how the key values are assigned. This is the most common case. The solution is described in Fred White's article "Integer Keys: The Final Chapter" available from Adager's web site. It's the solution Adager customers with "integer keys" use. Any HP3000 user can always call Adager to obtain the proper capacity values for their particular case.

```
http://www.adager.com/TechnicalPapersHTML/IntegerKeysFinalChapter.html
```

"The Three Bears" from Fred White

In a classic paper title "The Three Bears of IMAGE", Fred White of Adager explains three features that can be disastrous if mis-used: Integer Keys (the topic we just covered), Sorted Paths, and too many paths.

```
http://www.adager.com/TechnicalPapersHTML/ThreeBears.html
```

Here is how Fred begins his discussion of Sorted Paths:

> **Mama Bear: the SORTED PATH pitfall**
>
> My first live encounter with a misuse of sorted paths arose in 1975.
>
> The facts surrounding this incident were told to me by Jonathan Bale who was still on the IMAGE project. Neither one of us remembers the exact numeric details so I have used poetic license by making up numbers which seem to be reasonably close to the actual ones involved in the incident.
>
> The user had created a database containing one automatic master dataset and one detail dataset related by a 2-character key and where the resulting path was sorted by some long-forgotten field(s).
>
> The user had written a program which read a record from an input file, added two blank characters to serve as the search field and then performed a DBPUT to the detail dataset. This was repeated for all records of the input file.
>
> At the time that Jon received a phone call, the tape had not moved for around 10 hours and the program had already been running(?) for at least 30 hours. . . . *[Read the entire paper.]*

What Detail Datasets Have the Bad Data?

You know there is an invalid key value in a IMAGE automatic master, but you don't know which of the 16 details contain the bad value. How to find them all?

You could create a custom Suprtool job that looks in each of the linked datasets. But perhaps you have not worked on this database in a year or so. It could take a long time to get the names of the related sets, and the names of the key field in each one, all spelled correctly! Nothing difficult, but time-consuming.

Or, you could use the **List;Related** command of Suprtool's Dbedit module. This extremely simple command shows you all the entries in the database that are linked to one specific entry!

For example, I know that there is an invalid key value "N45 0002" in the A-ACCOUNT-SITE automatic master, but I don't know what detail sets have the offending data. Remember, automatic entries are created, uh, automatically, when you insert data into a related detail. If you want to have the detail insertion to fail unless the key value has been previously inserted, you would use a manual master dataset.

```
:run suprtool.pub.robelle
>base cko       {open the database}
Password [;]?
>edit           {enter the Dbedit module}
#               {prompt changes to #}
#
#list a-account-site;related

List in File: A-ACCOUNT-SITE

    ACCOUNT-SITE       >N45 0002  {Enter the bad value}

ACCOUNT-SITE    = N45 0002

Related Records from the File : D-SITE-INFO
  Key to path: ACCOUNT-SITE
No records found for this path.

Related Records from the File : D-ADDRESS
  Key to path: ACCOUNT-SITE
No records found for this path.

Related Records from the File : D-CPUS
  Key to path: ACCOUNT-SITE
ACCOUNT-NO      = N45            ACCOUNT-SITE    = N45 0002
PRODUCT-CODE    = ST             PRODUCT-OPTION  = LIC
CPU-MODEL       = SERIES 997-800
TIER-PURCHASED  =                CPU-OS          = MPE/iX
CPU-NO          = 4              CPU-HANDLE      = BANDIT
CPU-ID-NO       = 818110156
SUPPORT-FLAG    = Y

ACCOUNT-NO      = N45            ACCOUNT-SITE    = N45 0002
PRODUCT-CODE    = ST             PRODUCT-OPTION  = LIC
CPU-MODEL       = SERIES 969-420
TIER-PURCHASED  =                CPU-OS          = MPE/iX
CPU-NO          = 5              CPU-HANDLE      = SMOKEY
CPU-ID-NO       = 301661304
SUPPORT-FLAG    = Y

ACCOUNT-NO      = N45            ACCOUNT-SITE    = N45 0002
PRODUCT-CODE    = LQ             PRODUCT-OPTION  = LIC
CPU-MODEL       = SERIES 969-420
TIER-PURCHASED  = 1              CPU-OS          = MPE/iX
CPU-NO          = 0              CPU-HANDLE      = SMOKEY
CPU-ID-NO       = 301661304
SUPPORT-FLAG    = Y
```

The List;Related command found 3 entries in the d-cpus dataset that contain the bad value, ACCOUNT-SITE = N45 0002.

We show our listing to the administration staff and they tell us that the proper ACCOUNT-SITE value is "N45 0001". But what is the fastest way to fix it? We could use a serial Get, an Extract and an Update, but that is a lot of syntax to get correctly.

Or, we could use the Dbedit **Change** command. With this command, Suprtool changes the value of a master entry, and then updates all the detail entries that were pointed to by the original value.

For example,

```
#change a-account-site

Change Key Value for File: A-ACCOUNT-SITE

Enter Existing Key Value to Find:
    ACCOUNT-SITE        >N45 0002
Enter New Key Value to Replace with:
    ACCOUNT-SITE        >N45 0001

ACCOUNT-SITE    = N45 0002

OK to change this entry [no]: y
Begin changes (be patient) .. .. end changes!!
```

That's it. Less than 5 minutes to solve a problem that could have taken an hour to debug and fix.

Notes: You can also do List;Related on a detail entry and Suprtool will show you all the master entries that point to it.

Dbedit is not yet supported in Suprtool/UX, but if there is enough interest Robelle will consider implementing it for Eloquence.

Dynamic Dataset Expansion

By Fred White, Adager (retired)

About the author: Fred White (Senior Research Scientist at Adager from 1981 until his retirement in 2001) and Jonathan Bale worked together at Hewlett-Packard as leaders of the original IMAGE/3000 development team. .

Introduction

A TurboIMAGE (or, for short, IMAGE) dataset is enabled for dynamic expansion by DBSCHEMA at RootFile creation time if the capacity specification for the dataset includes a maximum capacity, an initial capacity and an increment. For convenience, I will use DDX (detail dynamic expansion) and MDX (master dynamic expansion).

When the database is created by DBUTIL, the EOF (end-of-file) and disc space allocated to a dynamic dataset is determined by the initial capacity. The file's LIMIT is determined by the maximum capacity. The increment is unknown to the File System but IMAGE maintains it in the RootFile along with the initial, current and maximum capacities.

You may use Adager to enable (or disable) an existing dataset for dynamic expansion (i.e., to change the various IMAGE dataset capacity values and their corresponding file-system LIMIT and EOF settings).

The term dynamic refers to the fact that, whenever DBPUT determines that a dataset requires additional capacity, DBPUT can perform an online expansion of that dataset (bumping the file's EOF to a higher value and updating the database's RootFile and the dataset's user label to reflect this increase in current capacity).

DDX for details

When a DBPUT occurs on a dynamic detail that is "full" (relative to its current capacity), DBPUT checks whether the current capacity is less than the maximum capacity. If "yes," DBPUT asks the file system to allocate additional disc space to accommodate the specified increment of capacity. (If the current capacity is equal to the maximum capacity, DBPUT returns a dataset full error code.)

If the disc Space Manager is able to fulfill the request, the newly acquired space is zeroed out, the EOF is increased in keeping with the increased size of the space allocated to the dataset, DBPUT updates the RootFile and the dataset's user label to match the new current capacity and then utilizes the newly acquired space to complete the DBPUT. Otherwise, DBPUT returns a dataset full error code.

Benefits of DDX for details

You may create your detail datasets with smaller initial capacities with little fear of encountering a dataset full condition. This minimizes your initial disc space requirements while still permitting your detail datasets to grow (or not) on an as needed basis.

Over time, some details may expand significantly and others little, if at all. DDX may help you minimizes disc space usage and backup times.

You may never need to shut down your application simply to increase the capacity of a detail dataset.

Adding entries to masters

Before discussing master datasets that have been enabled for dynamic expansion, let's understand how IMAGE goes about adding an entry to an ordinary master (i.e., a non-MDX master).

DBPUT uses the master's capacity and the new entry's key value to calculate the primary address of the new entry. DBPUT accesses the dataset block containing the primary address. If the primary address location is empty, the new entry is placed there.

If the primary location is occupied by a secondary (of some other synonym chain), a search is made (see below) for an empty location and the secondary is moved (thus becoming a migrating secondary) to that location and the new entry is placed in its primary location.

If the primary location is occupied by a primary entry, DBPUT verifies that the new entry's key value doesn't match that of the primary entry and then traverses the synonym chain (if any) verifying that the new entry's key value doesn't match the key value of any of the synonyms (IMAGE does not allow duplicate key values for master entries). A search is then made (see below) for an empty location and the new entry is placed there and attached to the end of its synonym chain.

Searching for an empty location

DBPUT first checks the current block. If an empty location is not found in the current block, DBPUT cyclically searches successive blocks until an empty location is found.

The disc access and wall time to perform this search is usually quite small and significantly degrades performance only when the dataset is large and (a) through the mis-use of non-hashing keys (or the use of hashing key values whose primary addresses are not evenly distributed) one or more long clusters of occupied locations exist (even when the master is not nearly full) or (b) despite good distribution of entries the dataset becomes so full (>95%?) that sheer overcrowding results in long clusters which some searches are compelled to span.

Minimizing the performance degradation of long searches

If non-hashing keys have been mis-used, in some cases changing the capacity can result in all entries becoming primaries so that searching is never needed and the performance of DBPUT, DBFIND and directed DBGET is optimal.

If your case doesn't lend itself to that solution, your other option is to convert the keys to hashing keys with Adager, edit your applications software to allow for the new data types, and recompile.

If your search performance is bad with hashing keys, your only option for regaining performance is to change capacity (increasing it if the master is nearly full).

These approaches may (or may not) be expensive, depending on your circumstances, but the resulting performance gains may be worth the price. The only way to find out is to experiment.

The two storage areas of MDX masters

Dynamic masters have a primary storage area (which must contain all of the primaries and may contain secondaries) and a secondaries-only storage area (initially non-existent).

At dataset creation time, only the disc space for the primary area is allocated and its size is determined by the value of the initial capacity. Disc space is allocated for the secondaries-only area only when the master is dynamically expanded.

Additional expansions enlarge the secondaries-only area but the size of the primary area never changes.

Adding entries to MDX masters

DBPUT uses the initial capacity and the new entry's key value to calculate the primary address of the new entry. It then accesses the IMAGE block corresponding to the primary address. If the primary address location is empty, the new entry is placed there. If the primary location is occupied by a secondary (of some other synonym chain), a search is made for an empty location and the secondary is moved (thus becoming a migrating secondary) to that empty location and the new entry is placed in its primary location.

If the primary location is occupied by a primary entry, DBPUT verifies that the new entry's key value doesn't match that of the primary entry and then traverses the synonym chain (if any) verifying that the new key value doesn't match the key value of any of the synonyms (IMAGE does not allow duplicate key values for master entries). A search is made for an empty location and the new entry is placed there and attached to the end of its synonym chain.

Searching for an empty location in an MDX master

There are two possibilities, depending on the location of the current block. If the current block is within the primary area DBPUT checks the current block. If an empty location is not found in the current block, DBPUT cyclically searches successive blocks until an empty location is found or the cyclical searching becomes excessive, in which case DBPUT obtains an empty location from the secondaries-only area.

If the secondaries-only area is full or doesn't exist, DBPUT performs a dynamic expansion before obtaining the empty location.

DBPUT employs two thresholds to determine whether the search time is excessive. One is a count of the maximum number of IMAGE blocks to be examined. The other is a percent of the initial capacity.

A search time is considered to be excessive if either of these thresholds is reached without finding an empty location. If the current block is in the secondaries-only area DBPUT obtains an empty location directly from the secondaries-only area (whose space management is identical to that of detail datasets). If the delete chain head is non-zero, its value provides the address of the empty location. Otherwise, the secondaries-only area's HighWater Mark (HWM) is incremented by one to provide the address of the empty location.

Benefits of MDX for masters

You can eliminate performance degradation caused by long searches for empty locations. You may never need to shut down your application to increase the capacity of a master dataset.

Of course, if your master employs hashing keys, you can obtain better performance (except for serial DBGET) right from the start (and in the long run) by specifying an initial capacity that provides at least as much capacity as you can afford and about 10% more than you may ever need. The problem with this solution is that you may not know how much capacity you need.

By providing a large primary area, you minimize the likelihood of synonyms and (with the exception of serial reads and backups) maximize performance.

By providing a small primary area you maximize synonyms and minimize performance.

Your motivation for using MDX masters should not be to conserve disc space. If you wish to conserve on disc space usage, focus your efforts on detail datasets, not on master datasets. If you do a good job with your detail datasets, you can be more generous with disc space for your master datasets.

If you specify MDX for a master, stipulate a modest increment just in case you have underestimated your needs (and solely to protect against search-time performance degradation).

The primary address calculation for integer (i.e., non-hashing) keys is predictable and such keys should never be used unless you know exactly what you're doing. See my paper, The Use and Abuse of Non-hashing Keys, in Adager's web site:

```
http://www.adager.com/TechnicalPapers.html
```

However, if you have inherited a master with integer keys and your performance falls off a cliff, either (a) someone has mistakenly changed its capacity from the one carefully chosen by the designer to a capacity that destroyed the performance or (b) integer keys shouldn't have been used in the first place.

Shortcomings and Pitfalls

Jumbo datasets (i.e., exceeding 4 gigabytes in size) cannot be enabled for dynamic expansion.

Fortunately, IMAGE version C.10.05 introduced LargeFile datasets, which allow dynamic expansion and can be as large as 128 gigabytes.

Unfortunately, a database can have either jumbo datasets or LargeFile datasets, but not both. Fortunately, you can use Adager to convert a database that has jumbo datasets into a database that supports LargeFile datasets. "Use LargeFiles" is the Adager command that takes care of the mechanics, including the physical conversion of jumbo datasets (if any) into LargeFile datasets.

If your increments are too small and expansion occurs frequently, your disc space becomes fragmented over time. If your increments are unnecessarily large, at expansion time the disc Space Manager may not be able to allocate the requested disc space.

To avoid these last two problems, continue to monitor and anticipate your system-wide disc space needs. When you specify dynamic capacity expansion, don't request an unnecessarily large increment. A smaller increment has a better chance of being available at expansion time.

Furthermore, the expansion operation (typically a hiccup in your application's performance) can become an annoying pause when the increment is so large that it takes a long time for the disc Space Manager to allocate the space and for the space to be initialized.

Epilogue

The purpose of this paper was not to provide you with hard-and-fast rules on the use of DDX/MDX but rather to provide you with a basic understanding of how dynamic dataset expansion works and why you might want to use it so that you can make informed decisions when applying the concept to your databases.

Keeping HP 3000s on the Road

Interview with Vladimir Volokh, Founder of VESoft

About this interview: it was conducted by Ron Seybold, the editor of ***The 3000 Newswire*** *magazine, an excellent source of the information and news for people who work on the HP 3000 computer system. It gives us a glimpse of a part of the HP 3000 world that we sometimes forget, the part where only results matter, where the user doesn't care if the 3000 is not sexy anymore, and where accuracy and low cost are the measures of success.*

He arrived in a red two-door sedan that had been traveling Texas roads for weeks, but Vladimir Volokh was not lost. The founder of HP 3000 software utility provider VESoft was wrapping up a six-week tour of his customer sites in our home state. These days the 63-year-old is spending weeks out on the road, visiting HP 3000 owners to educate IT managers and consult at affordable fees. He’s probably in closer contact with more 3000 customers in the customers’ offices than any single software vendor. That title of vendor is one that Vladimir — his affable Russian manner makes it almost impossible to refer to him by anything other than his first name — carries with pride.

He’s got much to be proud of. He estimates his company has sold 15,000 licenses for its products in the more than 20 years Vesoft has been in business. VESoft exploited the richeness of MPE’s file system to create the MPEX , shell. He founded his company in Los Angeles with his then 12-year-old prodigy of a son Eugene, who’d already worked in HP’s labs in a summer job at age 14. The Volokhs immigrated from the Soviet Union in the 1970s, first founding one of the largest HP 3000 software companies, then building Movieline magazine on his wife Anne’s business acumen and literary skills.

Eugene remains with the company whose name bears his first initial alongside his father’s, despite a career in law that went from clerking for a Supreme Court Justice to a UCLA professorship, frequent appearances on TV news and writing articles in the likes of the Wall Street Journal. Vladimir has never wavered from two points of focus, the HP 3000 and software. He spends his days on the road visiting his customers to help them better manage what they already own, and perhaps add a VESoft product or two, but most of his enthusiasm is for what’s already installed. Of MPEX, , or Security/3000 or VEAudit, he says, “These are our babies. What parent does not want to talk about their children?”

Vladimir dropped into our hometown unexpected but welcome, and he took the pleasant diversion of a hearty meal at The County Line barbeque house while we all celebrated my 45th birthday together. (He’s more prone to order grocery store rotisserie chicken when he’s dining alone, or sardines.) At the rib house he cleaned his plate, leaving the bones stripped. He pushed the plate forward and smiled, saying

"That is a plate of someone who survived the Siege." In the worst days of World War II, Vladimir's family weathered the 900-plus days of hunger and death.

Knowing his history, it seemed that a state of business like the one HP 3000 customers are surviving might not seem so dire in comparison. Seeing he was flush with field research about this year's Transition-bound 3000 community, we asked him to relay the feelings and condition of your fellow customers. Vladimir has been through worse than having HP discontinue the system which nearly all of his customers are using. We found he's remaining optimistic while driving a modest rental car through Texas and other states, teaching customers to keep their HP 3000s tuned up — because who knows where their road may end?

You've been on the road showing software and training in person. Why do you feel this is better than something like, say, Internet-based instruction?

Somehow I developed this mode of operation, something nobody else seems to do. At least HP never sent their engineers around. I see the customers in their environments, their offices loaded with papers and magazines never opened. I am suspicious when once in awhile I see somebody with a clean desk.

You should talk, look at the computer, and pass knowledge that way. See what is applicable to their machine, and fix it while doing it. People should know their needs, but they don't.

Classroom education is good for algebra, but not for computers. Of a class of 10 people, in their offices they have 10 different configurations. You give them an average in the class that does not fit anybody. They carry the information home, and whatever they remember they are not sure of. People say they remember me from a seminar, but when I look at their systems, nothing from the seminar is implemented. It was a waste of their time. The best education should be onsite, on their machine, one to one. That's what I think is important, and it's what I'm doing.

Have the people you've seen appear to be dispirited, or ready to give up on their HP 3000s?

I would call them lost. It seems to me they don't know what to do. They cannot abandon it right now; their whole business is running on it. Small HP shops have one or two people, and they cannot do any conversion on their own. They have no experience or manpower. They are continuing with the 3000. I am 100 percent busy on the road, in Florida, Pennsylvania, Texas, in Chicago.

Most of them are using packaged software, so they wait on what their vendor will do. Vendors should take real thought on what to do. They use software from Summit, Amisys, MANMAN, Ecometry (Smith-Gardner).

Some customers have used customization options to the degree that they're not users of those vendors anymore. Either they will abandon customization and go back to a plain vanilla package, or who knows what? I would suspect a reasonable thing for them to do is find another package for another machine ready to use — of course it will not be on HP-UX — and go for it. After all, their loyalty should be to the company and themselves, not to HP.

Why not on HP-UX?

The [replacement] packages already running and already debugged are probably not running on UX. IBM has inertia working for them, and they never abandoned their proprietary systems. We still wonder why HP did that.

Have you seen that people believe they don't have to do much more with their HP 3000, since HP's announced its end of support for the product?

Yes. I tell them not to take this HP announcement as an excuse for not doing correct system management. I like the message on the NewsWire's back page from Alfredo Rego: "There are mighty forces at play in the HP 3000 world, but what we can concentrate on is the realm we're in, and make it right."

We don't know what will happen, or whether it will happen. In the meantime you are responsible for your system, yesterday, today and tomorrow.

I am surprised when a seasoned manager tells me, "We will probably convert the 3000 in two years." I say two years is a long time. What are you doing today? You have to live to see two years. If you run out of disk space, and the system builds log files every day, you don't know they are there. Many customers don't know these files exist. The system works independently of what you know, or what you don't know. On a recent visit a customer had one percent of system disk left, and they didn't even know. Five percent of other space left, and it's not contiguous space — it was split into the smallest spaces.

What can be worse than running on vapors? Not knowing you're running on vapors. Two or three MPEX , commands brought them up to 30 percent space.

Do customers you see immediately understand having so little space on their systems is bad?

Yes, when you tell them. Very often they don't know what utility to use to look at it.

Well, I've heard the former GM of the 3000 division say that opening up boot drive space on the new 7.5 release was really no big deal. He said, "Disk is cheap now."

This is a typical consideration of a person who sells things. You have to consider the whole disk issue. You have to hook it up. And even if a disk has a mean time between failures of 10 years, owning 10 disks makes the failure just one year. You back it up, and you incur more difficulties. It's not good to buy extra hardware. It's better to manage it. I see a lot of mis-management, not MIS management.

Why do you suppose some 3000 managers don't know about these things?

They are overworked. I do hundreds of visits a year. Whatever has wire in it, I've seen, is their responsibility. Telephone systems, Internet, HP 3000s. As a result they never have time to read any of the manuals. I suspect they have the minimum knowledge of Windows, NT, HP-UX. It's all just loaded on them.

In my mind, this overworking is an excuse to get outside help. Experts on Internet, knowing the LAN. As the old saying goes, education is not expensive. Ignorance is expensive.

Is there a customer base of HP 3000 owners which HP knows nothing about anymore?

I'm sure of it. And some vendors are so protective of their customers they don't let other vendors know they also have their customers. Many customers don't work with HP at all. What is more upsetting is that I see customers who pay HP for Predictive Support, and everyday HP logs in and only checks logfiles for hardware errors. Customers who have one percent left don't get told by HP. A computer is talking to a computer, and we don't see any live person involved.

Do customers seem to get the education on the 3000 that they need from HP?

HP isn't interested in teaching about using things like VOLUTIL, or SYSGEN. Education is very important. Eugene wrote that many years ago that in colleges they teach you the answers. They should teach you the questions. Every HP class gives answers, and they don't tell you what you should ask.

What's the background you find most common among 3000 managers?

Everybody, even myself and [Robelle founder] Bob Green, was hired to do an application. Nobody would hire somebody to do system management. Most people remain in applications, and only some of them go beyond the call of duty. Maybe it's an old man talking, but those youngsters don't go beyond the call of duty.

Of course, some questions people ask show they don't read the manuals or use HP's help system. HELP isn't really good enough. You type HELP and the command, but you have to know the command name. HP is inching in the right direction, so you can say HELP FUNCTIONS, for example.

In MPEX, , you can say HELP DISK, with either a C or K. We provide a contributed library with our products, VECSL, full of helpful hints, command files and an EBOOK group.

With people not trained as systems managers, and responsible for everything with wire in their shops, where can they find the benefit in making their 3000s more efficient? Do they complain about not having enough time?

They tell me this all the time. My answer is, "how can we help you?" My method is to get somebody to help, and that's how I make my own consulting, of four parts: show, tell, do and undo. I tell them they should involve other experts in other areas, like databases.

If you know MPE, you will know its limitations. Then you can look around and find a way to overcome them, with solutions from people like Bradmark, Adager, Robelle and us.

You're associated with a rather rare resource in the 3000 world: a person expert in MPE under age 35, your son Eugene. Is he still active at VESoft?

He is the E in VESoft. I am famous for being Eugene's father. He is forever our vice-president of research and development.

Why has he stayed for 20-plus years? He might be the only university law professor who's also a software company R&D vice president.

HP made it easy for this kind of person to stay, people like Alfredo Rego, Brad Tashenberg, Bob Green. [The 3000 market] is a size not too big or too small, very sophisticated and yet not perfect. People stay with it and they put in a lot of intellectual energy, and there's no way we can measure that. A lot of smart, able people stayed with HP for so long, and made it what it is today. If you think about it, every piece of HP's 3000 software was improved upon over the years, except for the compilers.

Do you think this intellectual energy resource was taken into account when HP made its 3000 business decision last fall?

No. If HP would have supported the vendors like MCBA and ASK, that would be creating an ecosystem. It was ruined, and now they complain about it. So it seems that big companies make big mistakes.

Do you think there's any chance of reversing the decline of that ecosystem?

In the deep of my heart I have this feeling: that maybe, just maybe, they might reverse it, seeing the loyalty of the customers. When HP was delaying the RISC architecture, people risked their resumes and careers to stay with HP. HP made the wrong decision to unbundle IMAGE, and again they didn't understand that without IMAGE, HP is not HP. Luckily, HP listened, and it's improved IMAGE, large IMAGE, not only TurboIMAGE, but as Fred White says, CargoIMAGE. It's big, fast and does everything. All this should be used, not abandoned.

In my business, I wouldn't even think of making new models of computers and in the same year abandon them. The lab continues to improve HP, while the marketing tells us we should not use it anymore.

Do you have an opinion about whether an emulator could successfully run MPE on non-HP hardware?

It depends on how it is written. Inherently it would create a lot of overhead, if it simulates everything. If the able people in Allegro would be doing it, that would be a plus. Of the licensing issues and the copyright issues, it seems that HP should not care.

There's free market beliefs at the heart of what you've done with your company. It seems like a big part of why you came to the US from Soviet Russia in the 1970s. How does it it look to you when HP hesitates to free up MPE?

Not too many people understand that is a freedom to succeed or to fail. I spent the big part of my life in the Soviet Union at the time when it was the Soviet Union. American corporations now somehow resemble the Soviet Union. Central planning, committee decision making, and thinking that they know best, and nobody else should say anything to that effect. HP and other American corporations are like that. They even use the Soviet phrase "five-year plan." To me it is very funny, because I grew up when the Soviet Union was using five-year plans. The results of that were disastrous, and that's why the Soviet Union is not around anymore.

Looking at this example, we shouldn't let it happen to certain parts of our economy. That's what we're talking about. HP seems unwilling to let people choose an alternative, and maybe to win where HP lost. They hide behind market forces and make one decision, and then they don't allow the market to take over.

Is there anything good you carry from your Soviet experience into the 3000 world?

I continue my teaching as I did 50 years ago in the Soviet school system. In that system, good students teach bad students. One advantage is that while explaining, good students will learn even better. You should at least scan the manual, and better to read it and learn the manual.

Have you discovered in your travels that 3000 managers have more security problems than five years ago?

You don't know if you're insecure. People feel performance, or they call you if it's bad. If security is bad, you never know until something happens. They do their best [to be secure], and they don't think their best might not be good.

It's especially amazing in the health industry. Managers are not diligent, and now there is legislation to protect health information privacy. Many managers will tell me "the [legislation] deadline comes in two years, so we're okay now." Seasoned managers act like schoolchildren, like learning the algebra only for the teacher. How about your responsibility to the customers buying insurance?

I told one of these managers in Texas "I wouldn't buy your health plan. My information would not be secure."

Do you think that new connectivity offered through the Internet makes HP 3000 shops more vulnerable?

I don't think so. Eighty percent of the violations happen in-house. The outside violation is more spectacular, newsworthy. It can happen in every shop. In personal life, people are very diligent. When it comes to security, I hear this every day — my users are so ignorant. My first thought is that this manager is ignorant, if they think their users are ignorant.

People think the application vendor is going to make their system secure because of HIPAA legislation. But the way to the system is through the hundred other accounts other than the application. Nobody ever counts how many user accounts, and it's always in the hundreds. You're responsible for the whole system.

I ask, how many SM users do you have? I have yet to see anyone who guesses right. Usually they are optimists — they say two, while it is really five.

You've built a very successful company by creating a product that improves on the 3000's operating system. Is HP catching up in the 22 years since MPEX, came out?

They are inching forward. After 25 years they have a PURGE command for filesets. However, it still does not purge the databases. And it still cannot select the files to purge. We vendors, we play this game better than HP. All this richness of the file system is there, and we make use of it.

To defend HP, they have this legendary compatibility to take care of, which is very important to preserve their market. It's more important to them than a new feature, which might cause a failure in New Zealand, for example. Not everybody realizes that if you have good features that are compatible, not so good features are also compatible. They probably chose correctly not to improve the little things, and keep it compatible. And after all, they sold you the box already, so there is no money in making a new feature that HP cannot sell.

What's remarkable is that with the changes in HP management over the years, somehow this line was always there to let the vendors improve upon. We should admit that one of the hands of HP has been doing it right, up until now.

People suggest that maintaining the compatibility has kept HP from being responsive to changes for the 3000. Do you think so?

This is the price that you pay. MPE is a very complex system, millions of lines of code written over the years by smart cookies. It's hard to maintain software written by smart cookies who hate documentation. And yet they do it, and that's why I think outside development of MPE is difficult to accomplish. That's why Open Source for MPE would be very difficult.

What's likely to happen to the average size of HP 3000-using companies — does the community become a group of smaller firms?

I am afraid so. Big companies have money to scrap everything, recreate or buy something. They take the loss and go on.

So how does ownership change, based on what you see on the road today?

Us Americans — I say us, because I have been here since 1975, and I am a citizen now — understand the language of cars. To make a point, I say that when I see one percent left, "you're running on empty." But then they say, "maybe we'll be converting, we don't need you to rebuild the engine."

I am just offering to add water and oil, until you reach the spot of converting. They argue, "but it works." I say, "so you run your car in second gear, and nobody told you to shift. So you change it more often than you should, and it's not as fast, and you buy more oil. Let's do little things — non-intrusive optimization. Let's balance your disks, watch your database capacity, not back up your garbage files. You'll survive to see what will happen in two years." Let's run our cars in the right gear, and add water and oil regularly, before they fail.

Migrating a 3000 Application

At Robelle, we migrated our Qedit and Suprtool products from MPE to HP-UX over ten years ago, and the struggle is still sharp in our memory. Now many users are considering reasons for migrating. In the first section of this book, we presented the reasons a particular site might refuse to migrate (i.e., homestead). In this section we present the counter-arguments.

If you are an ISV, such as Ecometry or Summit, you need to keep selling your application package and you need new computer servers for that purpose, so you either have to migrate your product or freeze the current product in favor of a new application product. If you are user of such an ISV, you may be forced to switch to the migrated package or a new one as the current package falls further and further behind your needs.

We start and end this section of the book with ISV application vendors (QSS for the education market and AMS for the vineyard market) because they are the ones who have started down the migration path fastest and we can learn from them.

You may need more computing power than a cluster of N-class 3000s can deliver (health records management for example). New hardware is much more powerful and cheaper and uses less space. In a few years the power bill for your 3000 may be enough to buy a reasonable server every month.

For example, consider these COBOL compile and execute times that Duane Percox of QSS has gathered for the HP 3000 and various COBOL compilers on alternative platforms:

```
 CPU Sec.     Server and Compiler
3.39  13.80  HP 3000 N4000; 1 440mhz cpu, 2.5gb ram, HP COBOL II.

1.26   7.83  Dell 500SC Tower; Pentium III 1ghz, 0.5gb ram,  Fujitsu NetCOBOL
```

The remaining tests are equally depressing for the 3000:

```
http://qwebs.qss.com/cobdata.html
```

You want to implement new capabilities, such as generating PDF or GIF files in your application. For new functions that cannot be done on the 3000, you can always connect the 3000 to a Windows or Linux box, but that assumes you have a reasonably competent 3000-Windows-Linux-Network person available to you. If you were on a pure Windows server instead, you would just install some off-the-shelf software.

If you are an end-user site who has written their own custom application, you may resist migrating for a long time, but people who know the 3000 are getting older – eventually they may all retire. (Of course, there will be 3000 expert consultants available for many years to come!)

The rest of this section explores the challenges and changes you will face if you migrate.

A New COBOL Compiler:

Even if you don’t use COBOL, the FORTRAN or C compiler that you use on your new platform will be different from the HP compiler that you use now. Read our two chapters on alternative compilers starting on page 179.

A New User Interface to Replace VPLUS

The 3000 comes with VPLUS, a powerful and flexible forms processing system. Other platforms come with GUI screens with radio buttons and pull-down lists. Read our three chapters on converting VPLUS, starting on page 188.

A New Database to Replace IMAGE

HP has never released TurboIMAGE on any platform other than the 3000, so it is very likely that you will be switching to another platform. Read our many chapters on Oracle, Eloquence, and Open-Source databases, starting on page 217.

A New Operating System to Replace MPE

Which one will you be using: HP-UX, Linux, IBM's iSeries, Windows, or some other vendor's variant of Unix. Whichever one you move to, there will be a lot of new skills to learn and basic assumptions to relearn. Focusing primarily on Unix, we provide the therapy you need starting on page 286.

–Bob Green

ISV journeys down migration road

QSS plans Linux pilot for its education applications

By John Burke

THE 3000 NEWS WIRE

About the author: John Burke is the founder of Burke Consulting and Technology Solutions (www.burke-consulting.com), which specializes in system management, consulting and outsourcing. John has over 25 years experience in systems, operations and development, is co-chair of SIGMPE, and has been writing regularly about HP 3000 issues for over 10 years. You can reach him at john@burke-consulting.com.

Quintessential School Systems (QSS), founded in 1990, is an HP 3000 ISV (www.qss.com) providing software and consulting services to K-12 school districts and community college systems. While developing, supporting and providing administrative and student records management computing solutions for these public school districts, QSS has also created a set of tools for HP 3000 developers. QSDK is a subroutine library toolkit for HP 3000 programmers who wish to develop network applications for their HP 3000 without the hassle of learning network programming. QWEBS is a Web server written by QSS that runs on the HP 3000.

When QSS talks about migrating its HP 3000 applications to Open Source, we all need to pay attention to what they are doing and how they are going about it.

Public school systems are understandably very cost conscious, so for competitive reasons QSS had already started investigating migrating its software to an Open Source solution before HP even announced on November 14, 2001 its intention to EOL the HP 3000. This put QSS ahead of most ISVs and non-ISVs in determining how to migrate traditional HP 3000 COBOL and IMAGE applications. At HPWorld 2002, Duane Percox, a principal in QSS, gave a talk titled "Migrating COBOL and IMAGE to Linux with Open Source", presenting QSS's pilot project experience. With his talk, Percox hoped to share some of QSS's experiences in order to help others choose a migration approach. Since HP World, I have had the opportunity to interview Duane about his presentation. This article combines information from his presentation with the questions and answers from my interview.

In this first of two parts we look at the project goals and objectives, the project plan and the initial decisions. Next month, we will look at how the test migration worked and what was discovered along the way.

It is important to keep in mind that as an ISV, QSS's focus is necessarily different from a non-ISV looking to migrate or replace homegrown applications. After all, for QSS, buying a new package is not an option. Also, QSS customers tend to be very cost sensitive, thus an Open Source approach has a lot of appeal for an ISV providing a complete packaged solution. Non-ISVs looking to migrate homegrown applications to other platforms might want to stay with commercial operating systems, databases and compilers for the vendor support.

Before starting the pilot project QSS had to choose a target OS, database and compiler. For the OS, QSS chose SuSe Linux. I asked Percox why Linux and why the SuSe distribution? "Our migration target is an Open Systems and/or POSIX-compliant OS," he said. "We chose Linux and HP-UX as the target platforms with Linux for the pilot project. With the cost of Linux development systems being so low, doing the pilot on Linux was a natural. We believe that Linux is a wonderful choice for ISV solutions. However, we have large customers who might feel more comfortable with an HP-supported OS. That is why we are targeting both.

"As for the SuSe distribution, we basically had seen good things about it on the Internet and so we chose it for our pilot project. QSS is currently working out business arrangements with SuSe and Red Hat. It will all come down to the business side of things. We are pleased with both distributions, and given that Red Hat owns 52 percent of the market, we are certainly not discounting them.

"Our goal is to be a TSP (total solution provider) and essentially build a custom sub-distribution from one of these two. We will then host a patch-site with approved patches from the source company. We don't think our customers will care which we choose because we are basically going to say that 'we own the installation' of the Linux box. We won't want anything other than QSS applications to be installed on the application server."

I asked if QSS had considered system management issues in choosing an OS. Percox replied, "We are building an application environment that will provide for the job scheduling, spooling, etc. The specific library and toolset layer we provide will insulate the application from the particulars of each OS. However, choosing to be POSIX-compliant is what will help us be very similar."

With the choice of an OS platform out of the way, QSS next turned to the database. Percox said, "One of our goals was to migrate to a SQL-92 compliant RDBMS. Within that goal, we wanted to evaluate whether any Open Source RDBMS was sufficiently capable to support a commercial grade application." QSS evaluated mySQL (www.mysql.com), PostgreSQL (www.postgresql.com), Interbase (info.borland.com/devsupport/interbase/opensource) and SAPdb (www.sapdb.org). The choice for the pilot project was PostgreSQL.

As Percox said in his presentation, "This is an ever-changing landscape, but one which is moving in a reasonably consistent manner. High performance data access (Web-based, read-only systems) favors mySQL. Bulletproof commercial quality with transaction support favors PostgreSQL and SAPdb. Interbase has not established a good open source community. PostgreSQL, Interbase and SAPdb have support for transactions and lock isolation. Version 4 (future) of mySQL is supposed to support transactions. A number of good books have been written about PostgreSQL, making it the easiest to learn. SAPdb is coming on strong and is worth considering down the road."

I asked whether QSS had considered Eloquence and if so, why it had chosen not to use it. Percox said the issue was cost.

"Our customers are public education and they are not just sitting around with spare money waiting to be spent on a database," he said, "even one as reasonably priced as Eloquence. Since we are doing the migration and spreading the cost over our installed base and future sales we can take the hit on converting the COBOL code from TurboIMAGE to SQL. To help keep the migration cost down for QSS we are developing the SQL abstraction layer that we believe will give us both the ability to drop in replacement calls and the ability to tune for performance when needed without having to re-write the entire COBOL code library."

The third and final decision was which COBOL compiler to use for the pilot project. QSS already used Whisper Technology's Programmer Studio so it did not need an IDE. As Percox said, "Because we use WT-PS, we can focus on command-line/script compiling. We don't need fancy IDEs and, in fact, having a common IDE regardless of language can be very helpful and improve productivity for those developers who code in multiple languages on the server."

QSS chose to use TinyCOBOL (www.tinycobol.org) for the pilot project. Percox explained, "The principle reason for choosing an Open Source COBOL was that at the time the project was planned, all the commercial COBOL compilers for Linux required run-time licensing on a per-user-execution basis. As an ISV that serves a cost-sensitive end-user vertical market, we must deploy our solutions with minimal (or no) run-time fees. gnu COBOL is moving along very slowly and is not yet ready. TinyCOBOL was the only Open Source COBOL we could find that generated IA-32 code for Linux and that supported most of the COBOL-85 constructs. Once we got going, two commercial COBOLS for Linux became available that did not require run-time licensing, Fujitsu NetCOBOL (http://www.netcobol.com) and theKompany Kobol (http://www.thekompany.com/products/kobol/)."

School systems are understandably cost-conscious — so for competitive reasons QSS had already started investigating migrating its software to Linux before HP even announced its intention in late 2001 to leave the HP 3000 market. This put QSS ahead of most ISVs and non-ISVs in determining how to migrate traditional HP 3000 COBOL and IMAGE applications.

At HP World 2002, QSS principal Duane Percox talked on "Migrating COBOL and IMAGE to Linux with Open Source." I have combined material from Percox's presentation with an interview to create this article. This month, we will look at how well the vendor's test migration worked and what QSS discovered along the way.

Moving to SQL

The company's key first step was to translate IMAGE calls to SQL.

Figure 1 shows the typical IMAGE Access Model. A COBOL program opens a database, does a find by some key-value, and then in a loop, gets records, selecting the desired subset of records by testing values. The database access is tightly coupled with the program and operating system.

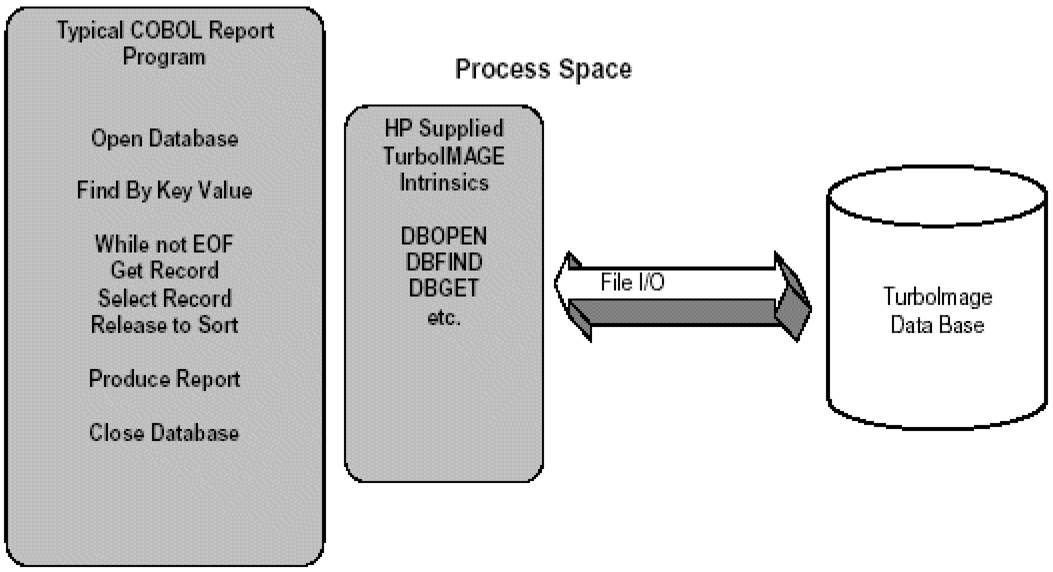
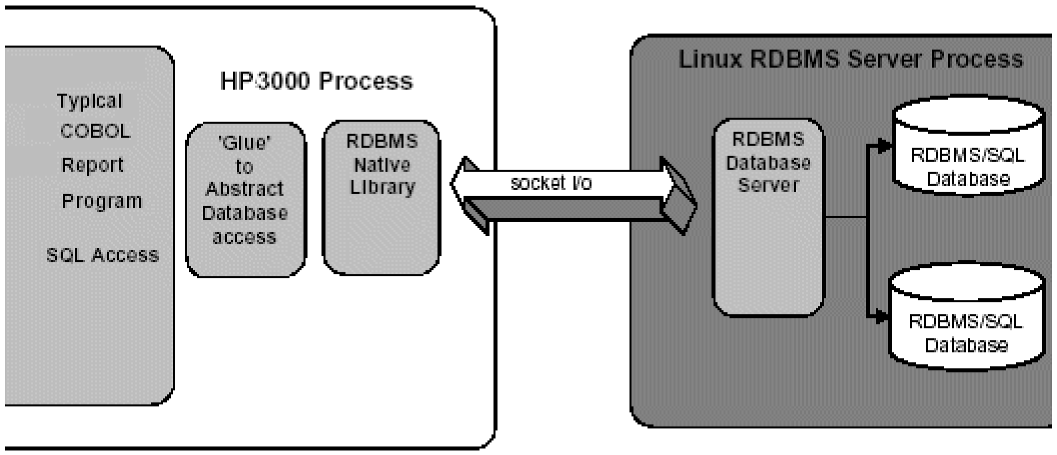


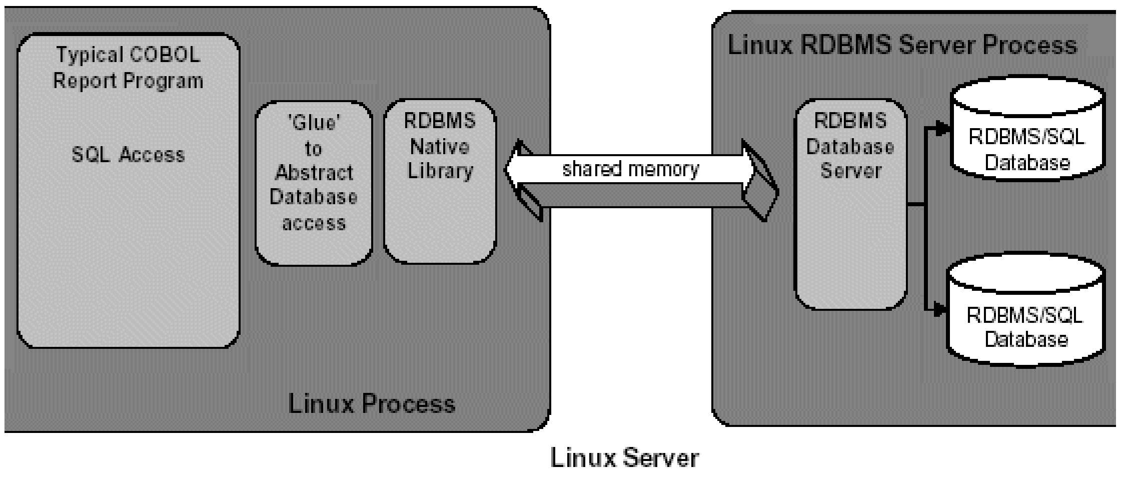
Figure 2 shows SQL access from a COBOL program on the HP 3000 to an relational database on a Linux server. QSS took this step as an intermediate proof of concept. All the programming remained on the HP 3000, but the relational database (PostgeSQL) resided on a Linux server.



QSS took out the IMAGE calls and replaced them with calls to its SQL interface library (QDBLIB – written in gnu 'c'). This was all possible because PostgreSQL has been ported to MPE and the database's native library routines run on MPE. This allowed QSS to run its code just as if it was running its IMAGE version. The only difference was that the programs were now accessing a PostgreSQL database on a Linux server.

The high number of connections to attach to networked database drove down performance. "Performance was reasonable for testing," Percox said, "but not stellar due primarily to the overhead of the SQL access versus IMAGE access and the TCP/IP network transport. The technical topology was HP COBOL II to QDBLIB to libpq to network to the PostgreSQL server process."

Figure 3 shows the final step that moved the application and its databases onto Linux. QSS ported QDBLIB to Linux, took sample COBOL code to the Linux box and used TinyCOBOL (www.tinycobol.org) to compile and run the same basic programs used in the previous example. Moving everything to one environment made the application much faster.



"Performance in this model was very good," Percox said, "especially considering the Linux server was a fairly basic IDE machine with little memory compared to what you would use for a production system. Also, when you access the relational database in this fashion, you are actually using shared memory and not TCP/IP transport. This is transparent to your application and handled completely by the libpq routines. This makes it trivial to move our database to another server, since no software modifications are required."

Porting the pilot

QSS took an existing database from its Fixed Assets system and ported it to PostgreSQL. The primary focus of the pilot project was the processing of a detail set with 70 fields and two paths, an organization number and an organization number concatenated with a 10-character asset number.

In order to simulate lots of database activity, QSS wrote a test program that would prompt for 10 organization number values. The test program then, in a loop for each entered organization number, would select the set of records that matched that organization number. It then wrote those records to a flat file.

One issue that is not immediately obvious when you consider migrating from MPE, COBOL, and IMAGE to Linux, COBOL, and a relational database are the differences in the way data is stored. Percox said typing makes up a lot of the differences.

"Relational databases are strongly typed, and IMAGE is not," he said. "This meant we ended up standardizing on these basic data types: CHAR[??], DATE, TIME, NUMERIC. When you get a record from IMAGE, you pass a memory address and IMAGE sends you a string of bytes starting at that memory location. These bytes typically overlay a record definition so the access is very fast. This is possible because of the weak data typing, and the fact that IMAGE just treats the information as bits and doesn't really care."

In contrast, he said, "Linux relational databases like PostgreSQL "will return a 2-dimensional table of 'c' terminated strings. The row of the table is equivalent to a record of the database, and the column is the same as a field. These strings are in human-readable format and cannot be used by your COBOL program unless they are converted to standard COBOL variables.

"For example, a number will be returned as '123.45'. You will want to convert this to your record buffer with the variable having a standard COBOL picture. This conversion of result data is what consumes the additional CPU cycles that cause slow database access. Consider a table of 10,000 rows and 70 columns. This would be 700,000 'c' strings that would have to be converted. No wonder IMAGE is faster."

The conversion overhead shows up while using a wrapper library, Percox explained. "This is why using a dumb IMAGE wrapper library can lead you into performance nightmare-land. By not adjusting your source code in any way you have no way to take advantage of the differences in the SQL engine. For example, instead of returning all 10,000 rows, it might be better to include selection criteria in the original SQL SELECT to reduce the number of rows returned, thereby reducing the number of conversions required. Also, do not return all columns, only the columns you need, thus significantly reducing the number of field conversions."

Open source pros and cons

QSS set out to use low-cost, vendor-independent Open Source solutions wherever possible. In the case of COBOL, there were only two options, gnu COBOL and TinyCOBOL. Percox said that gnu COBOL was clearly not ready and "is moving very slowly."

QSS chose to use TinyCOBOL for the pilot. However, Percox noted, "We discovered that TinyCOBOL does not have very good compiler error/warnings. We found it basically not usable for a production development shop. Copylib members have to be separated out into separate files in a directory. Compiler error messages report the line number, but the line number is not accurate when you have any copy members. Since our code is full of copy books this makes finding source code errors nearly impossible. In summary, we found TinyCOBOL was okay for test or demonstration projects, but not a long-term choice for our migration."

QSS started out using Whisper Technology's Programmer Studio and TinyCOBOL. Much of the QSS source had been developed with Robelle's Qedit and contained Qedit line tags that Percox said TinyCOBOL "mis-treated." I asked what QSS was going to use for its actual migration effort.

"We are currently debating this entire issue," he said, "and the debate centers around using CVS for source code management. If we switch to CVS then line-tags become a thing of the past and we use the built-in difference engine of CVS."

Percox also pointed out that "commercial COBOL compilers on open systems seem to have the ability for the programmer to define the source format, fixed column which would allow line tags or free format which gets confused by line-tags. We have decided to use Fujitsu NetCOBOL (www.netcobol.com) for the actual migration."

I asked whether it was fair to call the work QSS had done so far a feasibility study that turned into a migration plan. Percox replied, "I wouldn't put it like that. We established a migration plan framework with various components. One of the components of the plan was migrating to a SQL-92 compliant relational database. A sub-part to that plan was evaluating the ability of open source relational database to do work effectively. We did various investigative works prior to this pilot that validated the migration plan. The pilot project was used to establish additional validation for the plan with an actual QSS database."

New developments and lessons

Finally, I asked if there were any developments since Percox prepared his talk for HP World 2002 that QSS would like to share. Percox responded, "The primary development is the SAPdb Open Source relational database, which is looking very promising.

"With regard to Linux, as we continue to learn more and more about the community developing products for Linux, we are finding many great solutions for establishing high availability deployments with reasonably low cost. By low cost I am speaking of pricing in the single digit thousands for file systems and such that provide for high availability automatic fail-over and the sharing of resources in a clustered environment."

Percox said the savings in moving to Linux are substantial over using a vendor-supplied Unix. "What would cost tens of thousands of dollars, if not hundreds of thousands of dollars extra in the HP-UX world, is available in the Linux world for significantly less. A number of these vendors had booths at HP World and they were very knowledgeable about creating high availability solutions using Linux on clustered commodity priced IA-32 based servers."

I think the bottom line from the QSS experience is that the combination of an Open Source operating system (Linux) and an Open Source DBMS (PostgreSQL or SAPdb) is both a viable and attractive migration target if you are migrating. If your source code is COBOL, then you will probably have to purchase a commercial compiler since the Open Source COBOL compilers are not ready for prime time. Note that this experiment did not address interactive screens – that is the subject for another project.

My thanks go to QSS and Duane Percox for sharing the QSS migration experience and lessons learned. QSS's focus differs from 3000 customers looking to migrate or replace home-grown applications. For QSS, buying a new package is not an option. Since its customers tend to be very cost sensitive, an Open Source cost structure has a lot of appeal for an ISV like QSS providing a packaged solution. Customers looking to migrate homegrown applications to other platforms might want to stay with commercial operating systems, databases and compilers for the vendor support.

Note: Percox's 2002 slides are available to Interex members at

```
http://www.interex.org/conference/hpworld2002/proceedings
```

Migration Resources

By Robelle

If you do migrate, you will not be without information and resources, from web sites to technical books, to guides and tutorials, it is all available.

Migration Resources on the Web

NewsWire: HP-UX Admin for MPE Users (on 3kworld)

```
http://www.3kworld.com/newsroom.asp?sit_PK=67&appmode=itemDetail&news_pk=3856
```

"I think the biggest difference for most MPE administrators is just the concepts of admin work. For example, there aren't roles in UNIX like there are in MPE. (Example, you have manager.sys, operator.sys, etc..). In MPE you are used to having your operators be operators so that they can do backups, etc.. In UNIX there isn't a special class for this type of user, yet perhaps the operator needs root privileges to complete tasks. The administrator is left with the task of figuring out how to give the operator the permissions they need to do their job without giving permissions that could end up destroying the system."

SIU: Software Inventory Utility

HP's CSY division wrote a utility program to check your 3000 for migration issues. It scans your directories and produces an inventory report of what it finds. This is useful for system managers who are migrating, as well as those who are staying on the platform:

```
http://jazz.external.hp.com/src/scripts/siu/
```

"Performance of Migrated HP 3000 Applications" - by Kevin Cooper, HP

This presentation from the Spring 2003 Solutions Symposium helps answer the question: "what size system(s) will you need to run your migrated 3000 application on a new platform?" Here is a quote from the concluding slide:

> "Feedback from some early benchmarks suggests that an overall increase of 25-50% in processing power was about the right amount for those benchmarks. Each site needs to approximate what will be needed, based on the guidelines found here. Verify your approximations with performance tests before you go live!"

Online viewing:

```
http://jazz.external.hp.com/papers/SolSymposium_03/ssmigperf_kc.htm
```

Powerpoint file

```
http://jazz.external.hp.com/papers/SolSymposium_03/ssmigperf_kc.ppt
```

More Migration Information On the HP Web Site:

```
http://www.hp.com/products1/evolution/e3000/info_library/whitepapers.html
```

Among the useful papers found here in PDF format:

Compilers and User Interfaces (2146KB)

```
http://www.hp.com/products1/evolution/e3000/download/5981-3061EN.pdf
```

"The HP COBOL II language is a superset of the ANSI84 standard COBOL. The ANSI84 standard COBOL is available on many different platforms, (including Linux, HP-UX and Windows), in fact the ANSI COBOL standard has moved beyond the 84 standard. However, many of the COBOL enhancements and extensions that made HP COBOL II unique to the HP 3000 are not available outside of MPE/iX. This means that most COBOL programs that were written for the HP 3000 must be modified before they can be ported to another platform."

TurboIMAGE and databases (3657 KB)

```
http://www.hp.com/products1/evolution/e3000/download/5981-3062EN.pdf
```

"In this white paper, we'll explore three options for migrating TurboIMAGE applications, to take advantage of another more powerful DBMS:

1. Use a DBMS on the target platform that can be accessed using the TurboIMAGE intrinsic interface. There is one such DBMS; it is called **Eloquence**. Eloquence is supported on all of the recommended target platforms. Using Eloquence allows you to leave the TurboIMAGE intrinsic calls in your application after migrating it to HP-UX, Linux or Windows.
2. Redesign your application to use an RDBMS. Remove all the TurboIMAGE intrinsic calls from your application, and replace them with appropriate programmatic calls to access your chosen RDBMS on your target platform. These calls will most likely be take the form of **Embedded SQL** statements. This approach requires a good deal of work because of incompatibilities between TurboIMAGE and mainstream RDBMS products. These incompatibilities include differences in programming style, functionality differences such as locking, data type differences, dates, special characters and database optimization.
3. Use a mainstream relational DBMS such as Oracle or SQL Server, while leaving the TurboIMAGE intrinsic calls in your application on the target platform. This can be achieved using a software library that intercepts the TurboIMAGE intrinsic calls from your application, and translates them into

SQL calls for use with the relational DBMS. These libraries are sometimes referred to IMAGE Wrappers."

KSAM and MPE File System (2876 KB)

```
http://www.hp.com/products1/evolution/e3000/download/5981-3063EN.pdf
```

"There are a number of features of the MPE file system which are significantly different from the file systems of the recommended target platforms. These include:

- MPE's unique file naming convention
- MPE's three level directory structure
- MPE's support for specialized file types
- MPE's support for specialized file domains

MPE applications that make use of one of the KSAM access methods can be ported to any of the recommended target platforms. However, no KSAM access method is bundled with HP-UX, Linux or Windows. There are three replacements for KSAM…"

MPE Commands and networks (2774 KB)

```
http://www.hp.com/products1/evolution/e3000/download/5981-3064EN.pdf
```

"Batch processing on other platforms: Users of UNIX systems and Windows systems interact with the system somewhat differently from the way MPE users do. This reflects philosophical differences between the fundamental designs of the three operating systems. MPE was designed from the ground up to be used in commercial environments where end users are assumed to have little or no computer expertise.

- Control of the system is reserved for a system administrator or console operator.
- It is a server operating system only; there is no desktop version of MPE, and no built-in GUI.
- Most MPE applications are *not* based on a client/server model. That is, all processes (whether batch or online) run on the server.

UNIX was originally used in more technical environments, where the end users were more likely to have computer skills, and at least some familiarity with the command language (i.e. shell commands). This assumption has not carried over to commercial space, where users are more likely to be shielded from the underlying complexities of UNIX and the shell command language."

HP's Transition Series Webcasts

George Stachnik of HP has presented a series of webcasts on issues related to the end of the HP 3000. These webcasts discuss the transition from the HP 3000 to other HP platforms and the webcast archives are online.

Webcast #1: First Steps

```
http://www.hpclients.com/desktv/2698_01-22-02/e3000_transitions_e.htm
```

Webcast #2: Planning the project

```
http://hpmsweb1.com/content/250_257/04-23-02.htm
```

Webcast #3: Language Issues

```
http://hpmsweb1.com/content/310_365/05-21-02.htm
```

Webcast #4: User Interface Issues

```
http://www.hpbroadband.com/program.cfm?key=e3000pt4
```

Webcast #5: Database issues

```
http://www.hpbroadband.com/program.cfm?key=e30005
```

Webcast #6 Putting it All Together

```
http://www.hpbroadband.com/program.cfm?key=e3000mod6
```

Some Good Books on UNIX

UNIX-Haters Handbook

From its humble beginnings in 1969 as a quick way to get a game called "Space Travel" to run on a DEC PDP-7, UNIX has had its supporters and its detractors. Some of the latter group banded together on an Internet Mailing List called "UNIX-HATERS." That list has been cleverly turned into a systematic attack in book form. The *UNIX-Haters Handbook* was edited by Garfinkel, Weise and Strassmann, and was a famous book in print; it is now available in PDF format for free download:

```
http://research.microsoft.com/~daniel/unix-haters.html
```

You may find yourself laughing out loud on the bus while reading it and describe it to your spouse as "fun reading" as opposed to "work reading." It is often cruel and sarcastic but it is difficult not to empathize with the frustration that many of the users have endured, as reflected in some of their chapter headings:

UNIX - The world's first computer virus

Welcome New User! - Like Russian roulette with six bullets loaded

Documentation? - What documentation?

Snoozenet - I post, therefore I am

Terminal Insanity - Curses! foiled again!

The X-Windows Disaster - How to make a 50-MIPS workstation run like a PC

csh, pipes, and find - Power tools for power fools

Security - Oh, I'm sorry, sir, go ahead, I didn't realize you were root

The File System - Sure it corrupts your files, but look how fast it is!

You learn more about someone from their enemies than from their friends:

Two of the most famous products of Berkeley are LSD and UNIX. I don't think that this is a coincidence.-----Anonymous

Anyone else ever intend to type rm *.o and type this by accident: rm *>o ? Now you've got one new empty file called 'o', but plenty of room for it!" -----Dave Jones

Note: rm *>o removes all my files and redirects Stdout to the file o, while rm *.o removes only the files ending in .o. On most keyboards, * is shift-8 and > is shift-period, so this is a common typing error.

UNIX Power Tools

This is an excellent book for any UNIX user who has to write shell scripts. It was written by Jerry Peek, Tim O'Reilly, and Mike Loukides, published by O'Reilly and Associates/Bantam, and is available on the Net. A preview online at O'Reilly:

```
http://safari.oreilly.com/?XmlId=0-596-00330-7
```

This book is filled with examples, and is well-indexed and cross-referenced. At over 1000 pages, the book may look intimidating. However, this book is really a collection of independent articles, each just one to two pages long, that talk about a specific topic or solve a specific problem. The book is not meant to be read in order. Instead, look up something when you need to, or just flip to a random page and learn something new about UNIX.

For example, we learned the following two tips just from browsing:

Use **tail -f** to look at a file that is growing.

Use **set notify** to have background processes immediately tell you when they have changed states, instead of waiting for the next prompt.

Who is This Guy O'Reilly?

Wasn't he the quiet young Irish programmer who worked on UNIX?

No!

Actually O'Reilly is a publishing company that is the definitive source for books on UNIX. The covers of O'Reilly books are unique and easy to spot: a white background with an accent color, and a line drawing of a different animal -- except for *UNIX Power Tools* which has an electric drill.

If you plan to tackle any particular utility in UNIX, for example, uucp or awk, it's worth the time and money to purchase the appropriate O'Reilly book on the topic. You can order their books through Amazon or from their web page www.oreilly.com

Advanced Programming in the UNIX Environment

If you are writing software for UNIX you must get this book by Richard Stevens:

```
http://www.amazon.com/exec/obidos/ISBN=0201563177
```

It is jam-packed with readable and illuminating programming examples. Suppose you want your program to be aware of the type of terminal that is running it. If you look up "terminal" in the indexes of the three reference manuals for HP-UX, you'll find references to **ttytype**, **tty**, **ttyname** and **terminfo**. You read the brief and obscure man pages for these functions and you still don't have any idea how to use the information.

If you look in Stevens' book, you find an index reference for "terminal identification", which contains the C source code for **ttyname** and **isatty**, clearly explained and with output from sample runs. As well, Stevens spells out the historical background and the implications you should watch out for. Here's an example:

> POSIX.1 provides a run-time function that we can call to determine the name of the controlling terminal....Since most UNIX systems use **/dev/tty** as the name of the controlling terminal, this function is intended to aid portability to other operating systems....Two functions that are more interesting for a UNIX system are **isatty**, which returns true if a file descriptor refers to a terminal device, and **ttyname**, which returns the path name of the terminal device that is open on a file descriptor....The **isatty** function is trivial to implement...just try one of the terminal-specific functions (that doesn't change anything if it succeeds) and look at the return value.

Stevens' other books on UNIX and network topics are excellent also.

UNIX Resources on the Web

Robelle UNIX Tips

```
http://www.robelle.com/tips/unix.html
```

"Where Are My Temporary Files? When migrating from MPE to HP-UX, one common question is "where are my temporary files?". Unix doesn't have the "temp" and "perm" domains like MPE. Instead, Unix uses a random filename in the /var/tmp directory to function as a temporary file."

UNIX from the MPE Perspective, by Shawn Gordon

```
http://www.smga3000.com/papers/UnixvMPE.htm
```

"If your computer experience background is anything like mine, then you are already familiar with MPE and know DOS good enough to tell other people what to do, but you aren't sure if UNIX is an operating system or a court servitor. So I recently took it upon myself to learn UNIX and come up with enough meaningful equivalencies to the operating systems I was familiar with to make it easier. What I plan to do is share some of that experience with you and hopefully make your life a little easier if you have to deal with UNIX in the near future."

"One of the interesting things about UNIX is that unlike most operating systems, it wasn't originally developed to be sold or to sell hardware. UNIX was written in C, by one of the fathers of the C language, and the source code used to be given away with the operating system. Now most of these early installations were colleges and you can imagine what could happen to an operating system where the source code was readily available. So UNIX evolved to suit peoples needs as they arose, this is also why UNIX is rich with bizarre programs and commands, such as awk, grep, yacc and troff to name a few."

Gavin Scott: the Fun of Understanding Unix

```
http://www.3kworld.com/newsroom.asp?sit_PK=&appmode=itemDetail&news_pk=3979
```

"The pace at which Linux is improving and the rate at which high-quality "free" (both as in speech and as in beer) software is being produced is just amazing. Linux now has several "journaled" file systems similar to MPE's Transaction Manager, known as XM. And we're probably just about to the point where the Linux free database people (PostgreSQL, etc.) start talking to the file system people about doing explicit transactions between the database and file system. This is the point where you have the real equivalent of MPE and IMAGE's database-to-XM integration — the engineering that is responsible for much of the data integrity we so often use to tout MPE's superiority."

"Just the thought of being able to build real and robust business solutions using nothing but free software is quite intoxicating. Imagine never having to worry about the future viability of some small middleware vendor or even your DBMS supplier."

"For commercial developers it might mean that the money is going to be more in building specific, custom, end-user solutions rather than a lot of meta-solutions, middleware, tools and the like. That's because in the Linux model, anything that is general enough to be used by a lot of people is going to eventually be cost-free."

Gavin Scott: Virtual LINUX Under Windows

```
http://www.3kworld.com/newsroom.asp?sit_PK=&appmode=itemDetail&news_pk=3981
```

"For a beginner, I suggest getting RedHat 7.2 (you can try another distribution if you really want to for some reason), which you can download for free from RedHat or pay $60 if you want the pretty disks plus installation manual version. Do a complete install (check the "Everything" box), which will require just under 3Gb of disk space. Why live in a tent when you can have the Taj Mahal?"

"You may not even need a separate PC, if yours is running Windows NT/2000/XP, has 256Mb of memory or more and at least 4Gb of free disk space. You can use either VMware ($300, awesome, www.vmware.com) or Virtual PC for Windows ($200, haven't tried it, www.connectix.com) to run Linux inside a "virtual machine" under Windows. This means that you will have a complete Linux environment (X windows and all) running in a window (or full-screen so that you can't tell it's not the only OS running) at the same time you're running all your normal Windows stuff. Both products have downloadable, free 45-day demos available."

The UNIX Reference Desk

```
http://www.geek-girl.com/unix.html
```

Contains references to material culled from a number of different sources on computing in the UNIX environment. The resources in this document are divided into the following classes: General, Texinfo Pages, Applications, Programming, IBM AIX Systems, HP-UX Systems, Unix for PCs, Sun Systems, X Window System, Networking, Security, Humor. For example, in the General section, visit "UNIXhelp for Users", a beginner-level introduction to the UNIX operating system *Go here first!*

```
http://www.geek-girl.com/Unixhelp/
```

UNIX Guru Universe

```
http://www.ugu.com/sui/ugu/warp.ugu
```

This is an immense directory of useful web links.

HP-UX Resources on the Web

HP-UX Manuals On-line:

```
http://docs.hp.com/
```

Robelle's "About HP-UX" page:

```
http://www.robelle.com/abouthpux.html
```

Open Source Packages Ported to HP-UX

```
http://hpux.cs.utah.edu/
```

Software Porting And Archive Centre for HP-UX at Utah, USA. This site contains public domain packages which have been configured and ported to HP-UX.

comp.sys.hp.hpux mailing list FAQ

```
http://www.faqs.org/faqs/hp/hpux-faq/
```

hp-ux admin mailing list

```
http://www.dutchworks.nl/htbin/hpsysadmin
```

Tek-Tips HP-UX Forum

```
http://www.tek-tips.com/gthreadminder.cfm/lev2/3/lev3/20/pid/51
```

Exploit World: Vulnerabilities in HP-UX

```
http://www.insecure.org/sploits_hpux.html
```

HP: UX Software Depot

```
http://www.software.hp.com/
```

HP Learn HP-UX Resources

invent9k: Admin Functions Xref

```
http://invent9k.external.hp.com/~csy/sysadmin.cgi
```

This is a fun web tool that gives you the Unix translation of many common MPE functions. For example, if you select "Change User" from the pull-down list, here is what it says:

```
Changing a user's attributes
The MPE command ALTUSER is used to change a user's attributes, an example of
this is:   :ALTUSER BOB;HOME=SOURCE;UID=55

On HPUX the usermod (1M) command is used, an example of this is:
usermod -u 55 -g source bob

For further reference please see the man pages: usermod(1M)
```

invent9k: commands cross-reference

```
http://invent9k.external.hp.com/~csy/cmds.cgi
```

Similar to the previous web page, except that this one cross references MPE commands to their equivalent Unix commands. If you select ALLOW from the pull-down list (this command actually restricts who can execute certain commands), the cross-reference will say:

```
Sorry! No HP-UX equivalent exists. Please make another selection.
```

invent9k: programmer api cross-reference.

```
http://invent9k.external.hp.com/~csy/intr.cgi
```

Completes the trio of cross-references, this one for MPE intrinsics. If you ask about ARITRAP (which arms or disarms arithmetic traps for overflow, etc.), it shows the MAN page for fesettrapenable:

```
fesettrapenable(3M)

name
     fesettrapenable()  - set exception trap enable bits
      . . . <the rest of the man page> . . .
```

Software for Unix

Suprtool/UX: database utility for Eloquence and Oracle on HP-UX

```
http://www.suprtool.com/ux
```

Qedit: Easy-to-use full screen editor

```
http://www.robelle.com/products/qedit/moreinfo-ux.html
```

Eloquence (IMAGE-like database)

```
http://www.hp-eloquence.com
```

Screenjet: terminal emulator and Vplus replacement

```
http://www.screenjet.com
```

Ad Technologies: Tools For Migrating Code

```
http://www.adtech.com
```

Adtech's TransPort product migrates applications in COBOL and other languages to new platforms. Former Robelle employee Ken Robertson now works at Adtech and says their tools are quite "cool".

Ordina Denkart ViaNova

```
http://www.denkart.com/vianova/vn3000.htm
```

Transoft: Migration Tools:

```
http://www.transoft.com
```

A New COBOL Compiler

By Steve Hammond

About the author: Steve Hammond has worked in the HP 3000 environment for over 20 years. He has been everything from a junior programmer to a systems analyst to a system manager for employers using the 3000 platform. He was the chairman of Interex's SigPrint, the printer special interest group, for 10 years. He currently works for a professional association in Washington, DC, and is columnist for ***The 3000 Newswire****.*

Poor Grace Hopper. One fears the late, great admiral rolls over in her grave every time someone says COBOL on Unix or COBOL on Linux. Her language was created for big mainframes with lots of blinking lights and dozens of spinning tape drives. Not for servers that can fit under a desk, or worse, on a desktop.

But that's what is happening and many users of the HP 3000 have found a means to port the old workhorse of a language over to another platform as the end-of-life on the 3000 draws ever closer.

Two people who have gotten into the world of COBOL on another platform are Duane Percox of QSS in San Mateo, CA, and Ken Robertson of AD Technologies in Montreal. Their experiences have shown that the transition can be done, but there are some "bumps in the road" and planning helps make the transition smoother.

The COBOL products being used are Acucorp's AcuCOBOL, Microfocus COBOL and Fujitsu NetCOBOL. AcuCOBOL will run on both the 3000 and the 9000, so there is an advantage to following the HP migration path, but none of the trails are without effort.

The first difference anyone will notice between what the standard 3000 COBOL/iX user expects and what they will find in the new versions are syntactic. AcuCOBOL is syntactically the closest to COBOL/iX, having worked out most of the bugs. Meanwhile, a difference found in NetCOBOL is that the symbol <>, used to represent "not equal" is not recognized. It must be replaced with the actual words "not equal" or "not =". Other differences on various versions include that the syntax of "CALL...GIVING" must be replaced with "CALL...RETURNING". The new COBOLs do not accept variables in octal, so any of those values must be replaced with the hexadecimal representation; i.e. escape, represented as octal 33, must be represented as hex 1B. This is just the tip of the iceberg - there are many other differences, some because of bugs in the COBOL compiler, both on the HP 3000 platform and the target platforms.

On a different level, the issue of the 3000's file system versus any Unix-type of file system causes significant issues. "You don't have any file equations, " said Robertson. "You can't declare a file equation in the job to redirect it to a different file. You have to rethink it." Percox echoed the feeling, "You just can't do as much with the JCL." Beyond that, there are no message files, no RIO files, no circular files and no KSAM files. "We have just finished a project that used message files on the 3000," said Robertson. "We had to redesign how we did things."

Further issues include COPYLIBs - since COPYLIBs are KSAM files, they do not "travel" to a different platform. "We had to turn each section of a COPYLIB into a separate file and use them that way," added Percox. One thing that both Robertson and Percox noted is that Microfocus COBOL and NetCOBOL do not allow the use of macros. AcuCOBOL permits you to use these macros which all allow you to produce in-line code with variable substitutions. Of course, calling any intrinsics, such as FREAD or HPFOPEN, is not possible, again forcing some re-engineering.

Although it allows macros, Robertson noted that AcuCOBOL has some performance issues. AcuCOBOL does not create executables, so on a large project, with a large number of subprograms, performance may degrade.

Dealing with databases introduces a new set of issues. AcuCOBOL has an add-on giving it the Oracle access layer. This leads to a significant increase in cost, since Oracle requires a fee for every run-time user of the product. If you have hundreds of users, this could call for a sizeable financial commitment. Robertson had the projected cost of a conversion go over $700,000 when he factored in Oracle fees.

On the other side of the coin (figuratively and literally), for a few thousand dollars, Eloquence gives you a database which uses the same database calls as TurboImage, so COBOL changes are not as significant. Take note that Eloquence has not been tested yet in the field with more than 500 concurrent users.

Percox is in the process of migrating the QSS suite of programs to the Linux platform using PostgreSQL as the database. He has been able to write scripts on the Linux side to modify source code when it is moved there to deal with syntactical problems. He has also been able to use his HP 3000 to test some of the changes. "We do as much on the 3000 as possible to test some source code changes. Pascal and SPL subroutines are being recoded into C and changes are being made to the TurboImage databases to make them more SQL-like." For example, he has split up combined database fields to make them SQL-like. With help from his user community, he reports significant progress getting his software onto Linux.

Both these developers reflect the caveats of other system conversion professionals - the better the up-front planning and testing, the smoother the conversion. And as the MPE and the 3000 move to join CPM and the Apple II, more users are going to have to follow the path people like Ken and Duane have built.

More from Steve Hammond…

Interesting Aside: Shawn Gordon and KOBOL

HP 3000 veteran Shawn Gordon left the fold a few years ago. He saw computing taking a different direction and created a company called simply "The Kompany" in 1999. The Kompany's web site tells us that it is "a growing and rapidly developing company with a focus on multi-platform software for enterprise, developers and embedded devices."

The expatriot 3000 expert gravitated to the Linux operating system and found a niche there. "A lot of things that people like about the 3000, are there on Linux" said Gordon recently. "You can build a decent server for a thousand dollars and then run Linux on it. Linux is like MPE, it's a very stable OS. It just doesn't go down."

Realizing that over 80% of the business-oriented source code in the United States is still in COBOL, The Kompany developed Kobol, a fully ANSI compliant COBOL compiler for Linux, HP-UX and Windows. For 3000 users looking to move to a new platform, the compiler is very reasonably priced. Add on its MPE compatibility module, which includes support for macros and defined octal constants, and Allegro's Intrins/iX (currently available for the HP-UX platform and soon for Linux), it's still reasonable enough to purchase and not have your CFO blow an aneurism. A debugger is also available for obvious purposes.

Kobol uses the standard COBOL syntax, so COBOL developers need only learn the nuances of the platform instead of trying to determine how this version of the language functions. The compiler creates executable code, taking advantage of the associated performance benefits. Gordon reports that one of his customers converted, tested and implemented over 100,000 lines of COBOL code on Linux in a matter of just a few months with only two programmers. Kobol also currently works with Eloquence and the next version of the software, available this summer, will come with SQL preprocessor directives for a consistent interface to such database systems as MySQL, PostgreSQL, xBase, DB2, Oracle and ODBC.

Kobol does not support View/3000 screen-handling, but it does support the SCREEN-SECTION of COBOL for doing simple console-based screen reads.

Kobol is available from The Kompany (www.thekompany.com) - Kobol by download is $59.95, on CD is$74.95, Debugger $24.95 and the MPE plug-in $199.95. Intrins/iX is available from Allegro (www.allegro.com) for $5,000..

COBOL Migration: What You Can Do Now

"Even if you won't be moving for years, prepare your COBOL today"

By Wayne Boyer

THE 3000 NEWSWIRE

About the Author: Wayne Boyer (`callogic@aol.com`) has worked with HP 3000s for 24 years and is the president of Cal-Logic, a consulting organization specializing in business applications running on Hewlett Packard systems.

The announcement from Hewlett Packard that it plans to discontinue manufacturing its HP 3000 product line in November 2003 has prompted a large amount of discussion regarding ways of converting applications to run on alternative platforms. This chapter focuses on COBOL-based applications that you expect to move over to some new system in the future.

Handling HP's transition out of the 3000 community calls for solutions many organizations don't have today. However, if your organization utilizes COBOL as a primary programming language and you expect to re-use this source code on another platform in the future, this article can help you. You can get started now on some of the steps that you will need to take, regardless of what platform you end up on in the future.

For current HP COBOL sites, there are at least four possible non-HP COBOL compiler vendors offering products that will run on non-HP 3000 hardware. These vendors are Acucorp, Fujitsu, LegacyJ and Microfocus.

If you study each of these vendors' products, you will see differences between them — but fundamentally, they all are COBOL compilers that accept COBOL syntax. While Acucorp has gone to considerable effort to handle most of the non-standard HP-COBOL syntax, they have not yet managed to handle every such case. If you use one of the other three compilers, you will have slightly more work to do.

Most HP 3000-based COBOL source code contains some amount of code that is specific to either:

- The HP COBOL compiler syntax and operation
- The MPE/iX operating system
- Subroutine libraries

This chapter addresses the first category. Operating system and subroutine libraries of various types also need to be addressed. I hope to do so in a future article as more vendor solutions become available.

In addition, there can be variations in the level of standard COBOL being used. To ensure that we are thinking of the same thing, let's further describe each of these situations and the different COBOL standard levels.

HP COBOL compiler syntax and operation

This is any character string that exists in any source code file that effects how the HP COBOL compilers operate. Further, this refers to any such character string that is not part of the internationally accepted standard for the COBOL language. There are two types of character strings that exist in HP COBOL that are not part of the standard language. These are compiler directives (lines beginning with a "$") and HP-specific extensions to the regular COBOL language syntax.

Examples of these situations would be:

- $PAGE, an HP compiler directive that causes a form feed on the output listing;
- ACCEPT MY-DATA FREE ON INPUT ERROR …, an HP extension to the standard COBOL verb ACCEPT. Both 'FREE' and 'ON ERROR' are not standard COBOL;
- MOVE %33 TO MY-VALUE, an HP-specific octal constant which in this case corresponds to the ASCII Escape character (commonly used for printer control).

MPE/iX operating system

This is any character string that exists in any source code file that has no effect on the compiler but which does do something when the program is run. Examples would be:

- MOVE "!JOB J123,MANAGER.SYS " TO OUT-REC. This is part of a MPE jobstream embedded within COBOL source code.
- MOVE "FILE MYFILE;DEV=LP" TO COMMAND-BUF. This is MPE/iX command syntax that is probably intended for use with the COMMAND intrinsic.

Subroutine libraries

This is any set of subroutines referenced in a COBOL program via the CALL statement.

```
CALL "DBGET" USING DB-NAME, …, A typical Image subroutine call
CALL "VSHOWFORM" USING COM-AREA, … A typical VPLUS subroutine call
```

Language Versions

All COBOL source code on an MPE/iX system can be categorized into one of four possible levels of COBOL based upon the COBOL language standards that have evolved over the years. Listed below are these four categories.

COBOL-68: This is the original COBOL produced for the HP 3000. Such source code may only be used on a classic architecture HP 3000 or in Compatibility Mode ("CM") on a newer HP 3000 system. It is not difficult to upgrade old COBOL-68 source code to conform to a newer standard level.

COBOL-74: Many years ago, a COBOL-74 level compiler was released by Hewlett Packard. This was known as COBOL II. Currently, COBOL-74 source code can be either Compatibility Mode or Native Mode code. If you are still compiling source code as COBOL-74 code, look at what needs to be changed to treat all of your source code as COBOL-85 code.

COBOL-85: This is perhaps the most typical type of source code found on HP 3000 systems. COBOL-85 code can be compiled and linked into a Compatibility Mode PROG file or into a Native Mode NMPRG file.

COBOL-85 With 89 Addendum: Any source code compiled with the $CONTROL POST85 compiler directive must conform to the COBOL-85 standard and must also be compatible with the 1989 addendum to the COBOL language standard. All such source code must be compiled into Native Mode and is essentially the most up-to-date code from a language standards perspective. This is the situation that you will want all of your source code to be in before you start porting to a new platform.

What you can do?

So now that we have talked about the situation, what can be done about it? If you determine that you are going to move your code somewhere, you can work on making your HP 3000-based source code as standard and as up-to-date as possible. Here is a list of tasks that will make your source code much more portable to any future COBOL environment.

Add the $CONTROL STDWARN compiler directive to your COBCNTL file. This will direct the HP COBOL compiler to produce messages identifying non-standard COBOL syntax. Some but not all of these syntax issues are discussed further on in this article. Note that you will get warning messages related to various obsoleted clauses in the IDENTIFICATION DIVISION. These are easy to delete or convert into comment lines. Be prepared for a surprise when you add STDWARN. There are a lot of non-standard extensions in use. Many of these are common to other compilers though, so you do not need to remove them from your code.

Review your COBCNTL compiler control file contents and ensure that you are using as few versions of this file as possible. Optimally, you should only have one version of COBCNTL in use so that all of your source code is compiled with a consistent set of directives. As you make progress on improving your source code for portability, you can add the DIFF74 directive if you are using older code and eventually add the POST85 directive to ensure that your code is compiled according to the most recent COBOL language standard.

Note that with the DIFF74 directive, you will probably get warning messages related to various obsolete clauses in the IDENTIFICATION DIVISION. These are easy to delete or convert into comment lines. An easy way to do this is with a MPEX , global change on all of your source code. If you have never done this before, you can redirect the compiler to use a special COBCNTL file by issuing a :FILE command before the compile: FILE COBCNTL.PUB.SYS = yourcobcntlfilename

Develop a means of converting QEDIT type files or other filetypes into regular fixed ASCII EDTCT type files. When you eventually port your code over to another platform, QEDIT probably isn't going to be there and you will need to have a consistent, standard file type for your source code.

Compiler directives

Eliminate as many embedded compiler directives as possible. This may be the hardest part of porting your COBOL source code to a new platform. Some compiler directives follow which are easy to eliminate.

$COMMENT – This is a directive that is just a comment line. Convert all of these directives to regular COBOL comment lines (a "*" character in column 7).

$CONTROL QUOTE – This directive is something of a holdover from COBOL-68. Originally character strings could be delimited by single quote marks. This directive lets you perpetuate that method. You should change your single quote marks to double quote marks and remove this directive from your source code or COBCNTL if you have it there.

$CONTROL LIST and NOLIST – These directives control what source code is shown on a compile listing. Removing these directives will cause no harm other than probably an increased use of paper.

$PAGE and $TITLE – These directives effect only your compile listings and could easily be converted into normal standard COBOL comment lines.

$EDIT – This directive is probably not used in many organizations but it could be complex to eliminate. The logical alternatives would be COPY statements using the COPYLIB or $INCLUDE compiler directives. It is safe to assume that all alternative compilers will support something similar to $INCLUDE.

$DEFINE – This directive is used to define macros. In most cases removing macro definitions and their corresponding references in the source code is a complex task. I would recommend deferring this work until more is known about the future platform for your code.

Other steps

Isolate as many HP COBOL language extensions into the COPYLIB or into $INCLUDE files as possible. This will allow you to use one set of source code containing an HP language extension. Once you are ready to port to a different platform, you can change this one common shared piece of code instead of changing dozens of occurrences of the extension in multiple source code files.

Eliminate the usage of certain HP COBOL language extensions related to file locking: COBOLLOCK and COBOLUNLOCK, EXCLUSIVE and UN-EXCLUSIVE

While Acucorp's AcuCOBOL product supports quite a few HP language extensions, AcuCOBOL does not support the above extensions, and Acucorp is not intending to add support for these constructs in the future. Convert your source code to use the FLOCK and FUNLOCK intrinsics instead of these language extensions. You can also write little subroutines to hide these intrinsics so that eliminating these language extensions doesn't result in more usage of MPE/iX-specific subroutines.

Locate and eliminate all of these unusual HP COBOL language extensions. These extensions are essentially undocumented and are simply identified by their reserved word. Where possible, page references to HP's COBOL II/XL Reference Manual are given. If these extensions exist in source code processed by AcuCOBOL or any other compiler, compilation will fail.

Reserved Word	Page Reference
DIVIDE-BY-ZERO	No information available
BEGINNING	9-136
COMMON	5-4
DISABLE	E-10
ENABLE	E-10
ENDING	No information available
FILE-LIMITS	No information available
MORE-LABELS	9-138
PROCESSING	No information available

If you use RLs and/or XLs, prepare an automated method (perhaps a jobstream) of recreating each of your RL or XLs from only the source code. The library itself isn't going to get ported anywhere but your source code is. Having the ability to recreate each library ensures that you know exactly which source code files need to be compiled in order to create a library on your future target platform.

Eliminate any usage of subroutine switch stubs. These are used to handle switching between native mode and compatibility mode when a calling program and a called subroutine are not of the same mode.

Work toward having all code utilize 32-bit words instead of 16-bit words for binary data. Subroutine parameters need to align with the native word size This involves using the CALLALIGNED, LINKALIGNED and INDEX32 directives for 32-bit words. The 16-bit directives that you will want to phase out are CALLALIGNED16, LINKALIGNED16 and INDEX16.

Remember that in the future, you will probably be operating on a platform that is a 64 bit machine. The best that you can probably do now is to completely convert from 16 bit parameters to 32 bit. In general, this means changing the definition of some variables from 'PIC S9(04) COMP' to 'PIC S9(09) COMP'.

Avoid using the MPE/iX variable COBRUNTIME. This variable allows you to control how error or trap handling is managed. The chances of this existing on a different platform are small.

Re-name any variable or paragraph names that will conflict with the additional reserved words that are being added to COBOL with the pending 2002 COBOL standard. Also re-name any variable names or paragraph names that conflict with the vendor-specific reserved words in your future COBOL compiler.

More detailed information is in HP's reference manuals for COBOL, available at `www.docs.hp.com`. You can also order the COBOL Reference Manual Supplement and the old COBOL II/V manual.

Depending upon the state of your source code and your knowledge of your future environment, the information in this chapter can help you to eliminate a few months of delays in the process of porting your applications to a new environment. While the entire migration process will be difficult and time-consuming for many organizations, at least with a standard language like COBOL, migration is quite feasible.

A New User Interface to Replace VPLUS

By Alan Yeo, summarized by Bob Green

About the Author: Alan Yeo is CEO of ScreenJet Ltd, a UK-based company set up in 1999 to co-ordinate the design and development of the ScreenJet emulation and graphical user interface products for the HP 3000. Alan has also been the CEO of Affirm Ltd. since 1986. Affirm are an HP 3000 ISV specializing in manufacturing applications on the HP 3000. Alan has been designing, developing and implementing VPLUS-based applications since 1980. www.screenjet.com

Synopsis: The white paper written by Alan Yeo of ScreenJet provides a detailed explanation of the issues involved in migrating VPLUS applications. To summarize a lot of details, there are two main approaches: allow existing VPLUS calls to execute without change by porting the VPLUS API, or create similar functions but require the programmer to reorganize and upgrade the original source code.

Rather than reprint the entire white paper, which is quite detailed (it would be very helpful if you were creating a VPLUS migration tool of your own!), we are providing a summary, plus some key excerpts. You can download the paper at this web link:

```
http://207.21.244.161/download_mig.asp?SesEnc=&page=downloadlive
```

(You must give your email address to download the paper.)

Alan's firm ScreenJet has actually created tools for migrating VPLUS (see Alan's chapter, starting on page 188).

Key Points in the White Paper:

The problem is to migrate a VPLUS application to another platform. Some of the issues that make this non-trivial include the fact the VPLUS is designed for an HP terminal (which is not really dumb), that VPLUS is a complete form processing and management system, and that the VPLUS data buffer referenced by the application code must be kept in perfect sync with the data stream being passed between the actual terminal form.

"It is the mapping of this invisible data buffer that makes or breaks any VPLUS conversion, and either leaves it as an almost un-maintainable fossilized conversion of the VPLUS application as it was, or turns it into a format that allows future maintenance and enhancements to become as easy or easier than they were originally."

First Choice: Transformation or Fossilization

"Migrate the best of what you've got (obviously changing no more than you have to) and merge it with the best tools and processes you can find on your target platform. I'll call this Transformation. … Attempt to wrap virtually everything up in a shell on your target platform and pretend that you haven't migrated. What I'm calling the Fossilization option."

Since the creation of the white paper some 18 months ago, Alan now says that we have started to see some hybrid options emerging that provide the benefits of easy code migration, without all the downsides of some of the earlier Fossilization approaches.

Second Choice: Big Bang or Gentle Transition

At Robelle, with our history of promoting incremental development, we naturally would avoid any Big Bang migration strategy.

"Big Bang is good in that it gives you ultimate flexibility in what solution you choose, as it only has to work in the new environment. The downside is that you have an awful lot more to test at one time, and isolating where the problems are will be harder, as more components will have all changed at once…. Gentle transition is good in that your new user interface can be tested and live long before you migrate; you actually get some immediate pay back in the form of a better user interface. The downside is that if you want to do this, it will restrict some of the choices about what tools and services you can use."

Third Choice: VPLUS API or Application Changes

"This topic covers the issue of whether you wish to leave your application screen handling code untouched; i.e., leave all the existing VPLUS calls in the code, but have them resolved by a new VPLUS API that will in turn perform the processing necessary for your new User Interface. Or, do you go through the code, replacing the VPLUS calls with new code for your new User Interface?"

For a company that chooses to restructure their applications to be web-based and state-less, while retaining some of the VPLUS flavor and coding, see Gunnar Fredlund's chapter "From VPLUS to Web Application" next in this book.

Easy continuing development of the migrated application is a value; frozen IT applications seldom meet the needs of their users for long. So a migration path that recreates the VPLUS functions, but makes it extremely difficult to change, is less than ideal in most cases.

One problem that must be faced by all solutions that use a GUI: HP terminals and VPLUS have the ability to highlight several errors on the screen at the time, while the GUI has the concept of a single focus:

"One of the migration issues that will face everyone is this change in the user interface. For example, we are all used to applications that make extensive use of the 700/92 terminal highlighting facilities, such as blinking, inverse highlight, blinking inverse (or one of the thirteen other combinations, or in emulation mode a color mapping of these attributes) to convey meaning to the end user. These things have no direct correlation in a Windows or Browser interface."

"Likewise, we are used to an interface where both VPLUS and the application can highlight multiple fields simultaneously as being in error, and thus convey (via highlighting) to the user that they have multiple fields to correct. In a Windows interface, only one Field can have focus, thus having its contents highlighted. In a browser, we see something similar where we might see the Labels for Fields in error changed to red, to indicate multiple errors, but not the actual Fields changing their colors or attributes."

"What you are migrating is a Terminal-Based Server application. That application was written to control what Screen the user was on, to know what Screen they were on, and to control where a user went next. It can even prevent a user leaving a Screen until they have corrected something. It was not written with any code or logic to cope with the user just disappearing, or trying to reappear on a different Screen by hitting the Back button."

From VPLUS to Web Application

by Gunnar Fredlund, CIO Technologies

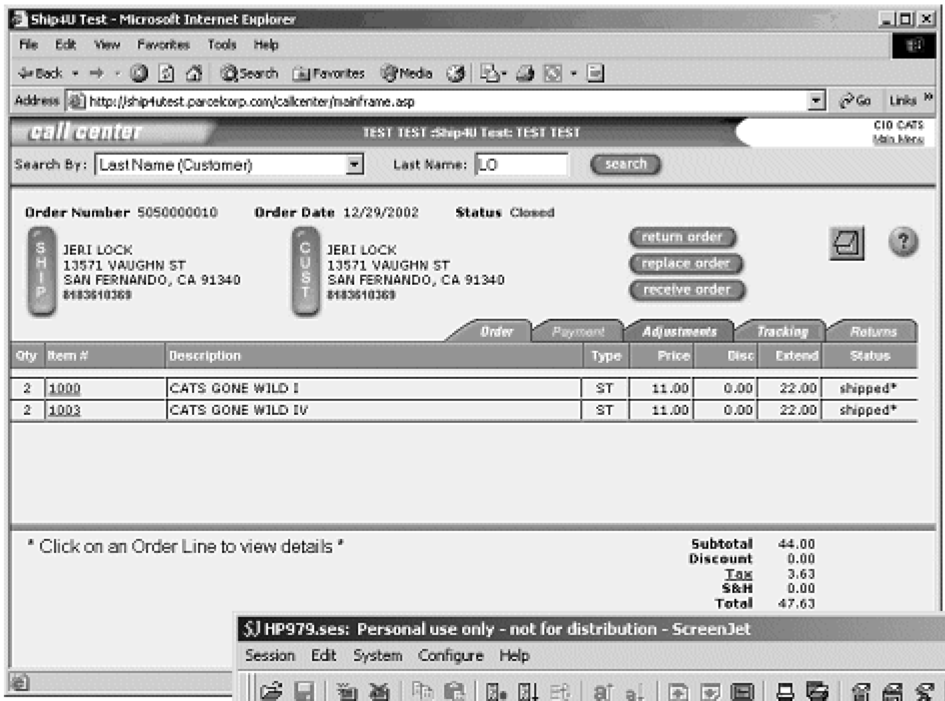
About the Author: Gunnar Fredlund was born and raised in Sweden, where he earned a BS in Computer Science and Mathematics at the University of Stockholm. He started using the HP 3000 as a primary development platform in the early days, then in 1985 moved to Santa Barbara, California for a short-term assignment. In 1987, he set up CIO Technologies, Inc. to focus on the design and implementation of mission critical systems. (www.ciotech.com) A central theme for Gunnar is a paradox - why so many companies refuse to let go of their legacy systems even though there has been such a dramatic change in hardware, databases and development tools. The analysis resulted in a new model for application development, blending core values from legacy systems with web technology.

When attempting to re-deploy a VPLUS application on another platform, there are numerous approaches that may be attractive, depending upon how many screens the application contains, how much staff re-training will cost, and how much application knowledge exists in the organization. Our firm, CIO Technologies, Inc. is a small software house that designs and implements custom mission-critical applications for the travel industry, retail back-office operations, credit card processing, fulfillment and warehouse management. Our effort to re-engineer our VPLUS interface began with a marketing wish, but switched to a business requirement when HP announced the end of the 3000 line.

VPLUS, Cobol and Image on the HP 3000 were our primary development tools. It was a very productive environment, when combined with the rock solid HP 3000 hardware. Over the years, we have used other technologies, such as C, Pascal, 4GL languages and Lisp. Many of these are more exciting than VPLUS, Cobol and Image, but we had yet to find a better toolset for business applications on the HP 3000.

Our requirements for a good application environment include:

- Rock solid hardware. You should be able to run mission-critical applications on a single system or use dual systems for 7 x 24 operations.
- A developer should be able to create databases, write programs and implement production systems without assistance from a large staff of experts.
- Little maintenance time should be required to stay compatible with the last operating system or tool release.
- The daily operation and management of the system and software should be minimal.
- The environment should be easy to interface with other environments.
- The system should be protected against misuse, hackers, viruses, etc.
- A good application environment should consist of proven technology.


Ship4U Test - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://ship4utest.parcelcorp.com/callcenter/mainframe.asp
Go Links
call center
TEST TEST :Ship4U Test: TEST TEST
CIO CATS
Main Menu
search
return order
replace order
receive order
Order Payment Adjustments Tracking Returns

Search By: Last Name (Customer) Last Name: LO

Order Number 5050000010 **Order Date** 12/29/2002 **Status** Closed

SHIP
JERI LOCK
13571 VAUGHN ST
SAN FERNANDO, CA 91340
8183610369

CUST
JERI LOCK
13571 VAUGHN ST
SAN FERNANDO, CA 91340
8183610369

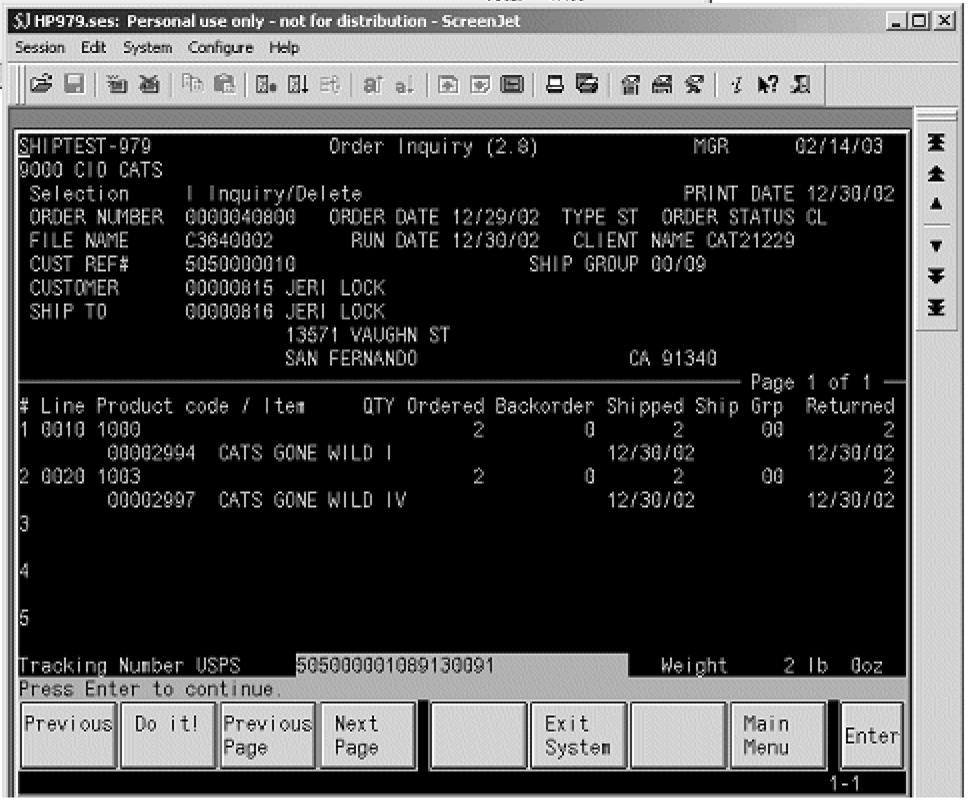
Qty	Item #	Description	Type	Price	Disc	Extend	Status
2	1000	CATS GONE WILD I	ST	11.00	0.00	22.00	shipped*
2	1003	CATS GONE WILD IV	ST	11.00	0.00	22.00	shipped*

* Click on an Order Line to view details *

Subtotal	44.00
Discount	0.00
Tax	3.63
S&H	0.00
Total	47.63

<–This is the new application look

And this is what our system used to look like –>


HP979.ses: Personal use only - not for distribution - ScreenJet
Session Edit System Configure Help
Previous
Do it!
Previous Page
Next Page
Exit System
Main Menu
Enter

```
SHIPTEST-979                Order Inquiry (2.0)            MGR      02/14/03
9000 CIO CATS
 Selection      I Inquiry/Delete                          PRINT DATE 12/30/02
 ORDER NUMBER  0000040800    ORDER DATE 12/29/02  TYPE ST  ORDER STATUS CL
 FILE NAME     C3640002        RUN DATE 12/30/02   CLIENT NAME CAT21229
 CUST REF#     5050000010                     SHIP GROUP 00/09
 CUSTOMER      00000815 JERI LOCK
 SHIP TO       00000816 JERI LOCK
                        13571 VAUGHN ST
                        SAN FERNANDO                   CA 91340
                                                          Page 1 of 1
# Line Product code / Item     QTY Ordered Backorder Shipped Ship Grp  Returned
1 0010 1000                              2         0       2      00          2
        00002994  CATS GONE WILD I                   12/30/02          12/30/02
2 0020 1003                              2         0       2      00          2
        00002997  CATS GONE WILD IV                  12/30/02          12/30/02
3

4

5

Tracking Number USPS     5050000010891300091      Weight     2 lb  0oz
Press Enter to continue.
                                                                        1-1
```

Our choice of HP 3000 technology made it possible for us to develop and maintain large applications, even though our company is very small.

The only critical issue we faced was the common IT perception that the HP 3000 was a dinosaur of a system, and the fact that our VPLUS terminal interface strengthened that impression.

For example, one customer installed a new financial system from a leading vendor; they spent over a year to customize it to handle a complicated internal billing system. Finally they gave up, and asked us to move the data to our system (we had them up running in less than a month). Since then, the customer has changed IT leadership twice, both times the new CIO started by evaluating whether to replace our old system with the modern system from the application vendor!

We listened to our customers: it was time **to find a replacement for VPLUS.** Happily, one customer helped fund development of a new user interface. This was the start of a series of projects that resulted in the new user interface, but also the creation of an application middleware, **Omnihost**, and a process to transform VPLUS applications into web applications.

The process has been very interesting; many of our long-term habits did not make sense anymore, and others turned out to have deeper intrinsic value than we ever thought. The result is a new model for application development, based on both the core values of the HP 3000 and the promises of the web-services technology.

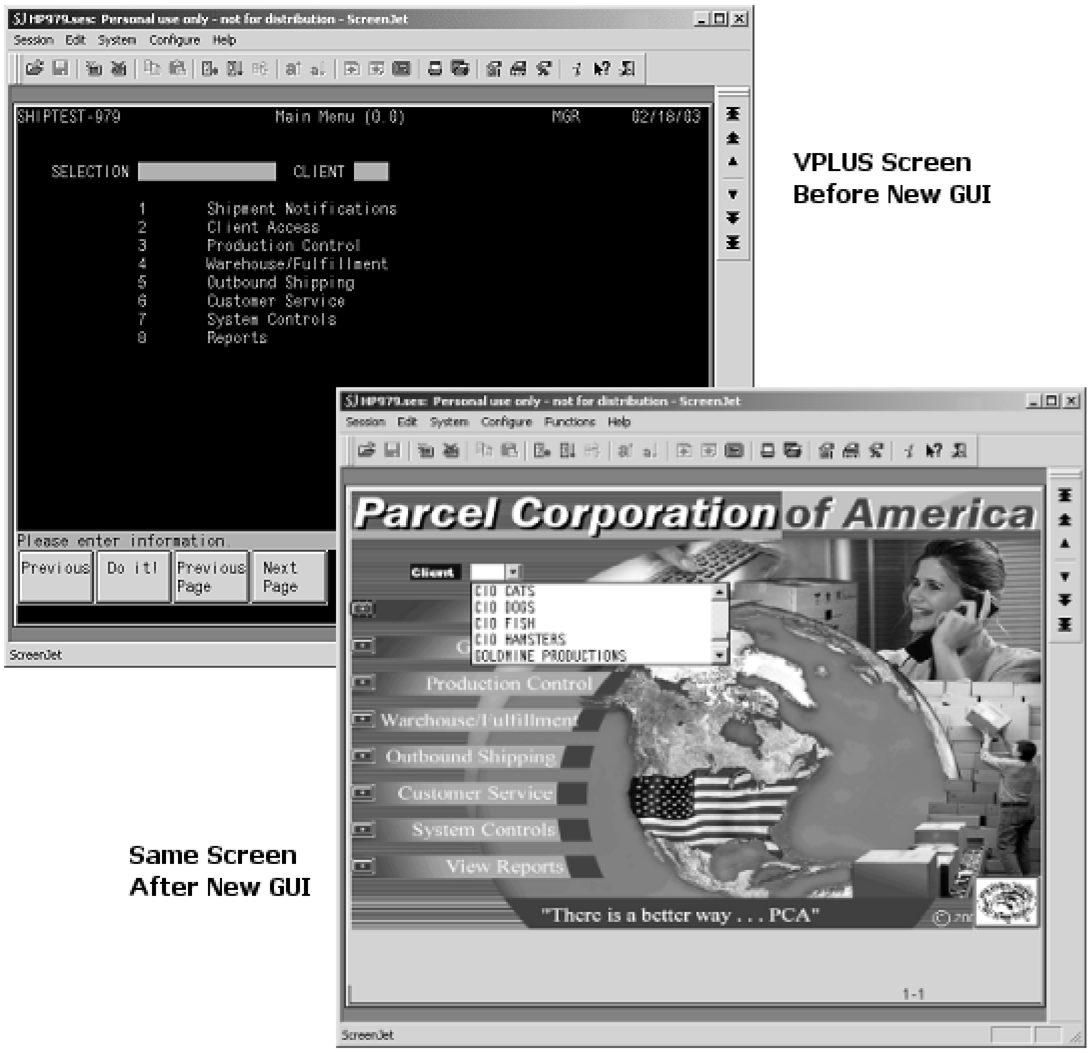
VPLUS to Windows Look-and-Feel

We were not sure what our new user interface should look like, should we go for a Windows or Web look and feel? We researched different tools and found a very interesting alternative, **ScreenJet 3000**, from ScreenJet Limited. The tool intercepts VPLUS calls and passes the control to a GUI screen defined by a Designer tool. ScreenJet gave us the opportunity to transfer the VPLUS screen to a Windows look and feel using 3D panels, combo boxes, radio buttons, check boxes and macros. It was a major improvement for the users; it took our application as far as we could without changing the 24 x 80 model.

- Cryptic codes were replaced by combo boxes or radio buttons
- Yes/No questions were turned into check boxes
- The function keys were replaced by buttons or icons
- Macros were used to link VPLUS data to the web (e.g. FedEx and UPS tracking).

All these changes could be done **without changing our COBOL code at all**. We used ScreenJet 3000 to transfer 225 VPLUS screens to GUI format.

This gave us a more modern look-and-feel, but the underlying logic was still totally VPLUS, as we will see.


HP979.ses: Personal use only - not for distribution - ScreenJet
Session Edit System Configure Help
VPLUS Screen
Before New GUI
HP979.ses: Personal use only - not for distribution - ScreenJet
Session Edit System Configure Functions Help
Parcel Corporation of America
Client
CIO CATS
CIO DOGS
CIO FISH
CIO HAMSTERS
GOLDMINE PRODUCTIONS
Production Control
Warehouse/Fulfillment
Outbound Shipping
Customer Service
System Controls
View Reports
"There is a better way . . . PCA"
1-1
ScreenJet
Same Screen
After New GUI

```
SHIPTEST-979              Main Menu (0.0)           MGR      02/18/03

    SELECTION                  CLIENT

              1        Shipment Notifications
              2        Client Access
              3        Production Control
              4        Warehouse/Fulfillment
              5        Outbound Shipping
              6        Customer Service
              7        System Controls
              8        Reports

Please enter information.
Previous  Do it!  Previous Page  Next Page
ScreenJet
```

The new user interface was popular with the users and we planned to keep if for a long time. Hewlett-Packard changed our plans overnight by discontinuing the HP3000 product line. We were now faced with the challenge of moving the entire application, including the new user interface.

It was natural to use our recent experience and start by creating a web application to replace the GUI interface, and then as a second step, move COBOL code and data base. We knew that our users would not mind running the HP 3000 a few more years as long as the user interface was state-of-the-art. At this time, we wanted to get away from the VPLUS limitations which could not easily be removed while using VPLUS to design screens:

- The **24 x 80 model** – all programs presented data in a screen twenty-four rows long by eighty characters wide. Often, similar functions were split into several programs due to screen space limitations, without thinking about user friendliness (e.g. programs for order inquiry, receive returns and order confirmation have almost the same look)

- A user looking at an order wants to see detail information, customer, ship-to, and product inventory in pop-up windows, without leaving the order screen – impossible with VPLUS calls
- A large VPLUS system requires a hierarchical menu system to navigate. It is hard for new users to learn this menu system, and it can be difficult to jump between tasks
- Web users expect to navigate using hyperlinks and buttons
- The web users expect a full scale of colors instead of Windows business gray or the green terminal screen
- The web model lets users see all the information that they can find; the VPLUS systems often restricted the users to certain menus
- Secure encrypted VPLUS sessions over Internet required us to set up VPN (Virtual Private Network) networks. Unfortunately, the VPN technology is not entirely standard so we needed different client software depending on the firewall / VPN installation at the server. We expected SSL (Secure Socket Layer) communication over Internet to be much easier to implement than VPN.

One ScreenJet limitation turned out to be very good for us: the feature set consisting of Windows objects was so small that we could handle the design ourselves. This is not the case with web design; we discovered quickly that we needed outside development and design help. To use colors to enhance the application sounded like a simple task. However, it was soon obvious that the color theme chosen by a programmer would not be acceptable! We needed a graphical designer.

We set as our goal to **keep the benefits of the HP 3000** environment (reliability, performance, simplicity, etc.), while at the same time **eliminate all HP 3000 traditions which made no sense** in the new environment.

Little did we know that our goal would force us to pursue a very narrow approach. There are numerous tools and frameworks for web applications, if you are ready to throw away your HP 3000 background, as well as several tools to migrate the HP 3000 model to new platforms.

The Challenge

- To develop a state-of-the-art web application
- To transfer the rock solid HP 3000 architecture to a state-less Internet environment
- To create a cost-efficient development environment

- To be able to migrate the application without changing the presentation
- To create a cost effective process to "webify" other VPLUS applications

To develop a state of the art web application

Our first step was to learn what a "state-of-the-art" web application should look like. To make things worse, almost all the leading Internet companies were falling apart at that time. Obviously the".com" upstarts did not have sound business plans, but they still dominated the web development discussion. This exercise was necessary because we thought exclusively in terms of the VPLUS look and feel. After years of HP 3000 development; we naturally thought in terms of 24 x 80 screens, instant response and hierarchical menu systems. We were leaving a very successful model for business applications in search of a new model.

We still thought of our applications as one entity, even though we had learned from client/server design that the system should be divided into tiers. So we started with the familiar three-tier model: presentation, application and data structure. However, analysis revealed that the web has made matters more complex. Not only do we have three tiers, but they are different in their nature.

In our example of warehouse management and fulfillment, the differences are evident:

- Over a long period of years, the data structure remains fairly constant– a customer order had payment terms, bill-to, ship-to and order lines with product codes, quantity, prices and discounts even before the first commercial computers were built.
- Typically, the application logic has a life cycle of seven years or more. A warehouse operation does not change drastically in a short time period, as it is very expensive to replace shelving, conveyer belts, scales, bar code scanners, etc.
- The presentation layer has a very short life span; many application packages comes out with a new version every year to stay competitive and compatible with current trends (client server, windows, web)

This knowledge convinced us that it is both possible and intelligent to **replace the presentation layer without rewriting the entire system.**

A balanced system should consist of three independent tiers, <u>data structure, application and presentation</u>. The three tiers have different life cycles and should be developed with different tools using standard interfaces between the different parts.

Independent – keep the data and replace the application logic, keep the logic and replace the presentation, or in case of our final phase in the migration plan – keep the presentation and application logic and move the data to a new data base.

Different tools – this is a consequence of the radically different life cycles. You cannot have a state of the art presentation if you don't use the current set of web tools. On the other hand, you cannot expect to use the code from a state-of-the-art tool unchanged five to ten years from now.

Standard Interfaces – using a standard to communicate between the tiers makes it easier to replace a tier and to test different tier options independently. For our SHIP4U system, we decided to use standard SQL as the interface between data structure and application, and XML between application and presentation.

For the new user interface, we removed the one-to-one relationship between programs and VPLUS forms (or forms families). In a web application it is not acceptable to display a customer order, and then require the user to traverse a menu system to return or replace a product from the same order. The user expects to have additional information available instantly, through pop-up screens or new windows.

This was the end of our old menu-based program structure. This meant that we had to isolate the **application logic** from the **VPLUS navigation and forms handling**. Four new programs - AddCustomer, ChgCustomer, GetCustomer and DelCustomer - replaced our one program for customer maintenance. We called the new programs "transactions."

The one to one relationship between application program and VPLUS form is replaced by a many to many relationship. One transaction can be used in many web screens and one screen can use many transactions.

This decision had a profound impact on the project. We now had a clear model to work with, and we reduced the size of the project to a fraction of what we initially anticipated.

First, we ignored all the logic for VPLUS screen handling, menu structure and access control (tied to menus). Next, we eliminated unnecessary redundancy in existing code. For example, order inquiry, confirmation, and return handling all start with an order search, followed by access to customer, ship-to and order information. In the new system, all that logic was implemented once in the order view, which is the starting point for all order related activities. Much of our old code was control statements handling the context-sensitive issues, e.g. which validations should be performed for an add and change, but not for the inquiry or delete. This context-sensitive code was not needed any more, as all transactions have a single task to perform. Finally, we could leave batch programs and reports unchanged, thanks to our decision to only replace the presentation layer. We estimate that the core application logic which we needed to find and reuse amounts to only 10–15% of our original COBOL code!

The typical VPLUS hierarchical menu system is thus flattened out. Everyone who has access to order inquiry can see the same data, and the former separate programs have been replaced with buttons, tabs and pop-up windows.

For example, the Web Order screen (see the first example screen in this article), has buttons, tabs and pop-up windows that link to 16 of the transactions: GetOrdHead, ReturnOrd, GetNotes, AddNotes, ReplaceOrd, ReciveReturn, GetShipTo, ChgShipTo, GetOrdAdj, GetItem, GetCust, ChgCust, GetPmtInfo, GetOrdLines, and GetTaxRate!

In a VPLUS-based system that would correspond to 10+ programs. The new system changes the way that access control is handled: the restriction on who can return an order cannot be linked to a menu; instead, in this case, it is linked to a pushbutton.

While working on the new application look-and-feel, we changed many long-standing habits:

- We had relied on training for our terminal users - in the web environment everyone expects to use the system right from the start.
- We had made extensive use of codes to save screen space; now we are replacing them with pull down lists or text fields.
- All VPLUS text fields had a fixed length, often very short (it takes a long time to realize that we don't have to save disk space any more).
- HP 3000 systems are often keyword based; the user had a report next to the terminal showing the product codes used when entering orders or adjusting inventory. The web user expects to be able to search for all information and needs new search functionality and pull-down lists.
- VPLUS offers few possibilities to stress important data due to the limited space, 24 x 80 and lack of color and graphics. E.g. we had a lead text "Discontinued" followed by a Y or N in the VPLUS screen; it is now replaced with a red text "Discontinued" in such a large bold font that it is hard to miss.
- The terminal screen has lots of fields with Yes or No values, e.g. "Tax Exempt:". Check boxes replace these fields for edit purposes, but a conditional display showing "Tax Exempt" for the few customers that are exempt is better than always displaying a lead text and a Yes or No.
- Data entry fields for start and stop dates have been replaced with buttons.
- Complex screens that cram information into a limited space are replaced by much simpler screens where additional information is available through pop-up screens and tabs.

The web design process can be summarized as a constant pursuit of simplicity with a goal of creating an intuitive user-friendly solution.

To transfer the rock solid HP3000 architecture to a state-less Internet environment

The old VPLUS system was designed for terminals connected directly to the HP 3000, which created a very robust environment. When we started using Windows and terminal emulators we encountered a new issue: the user session could instantly be interrupted by closing the window or by Windows itself.

The web adds many new dimensions, since the data is transmitted over public networks and must be encrypted, there can be delays caused by other activities (someone is using all Internet bandwidth by downloading a movie), and there are many layers of network equipments and servers you are dependent on without having any chance to control. On top of that, the typical web mode of operation is state-less while the old applications are context-sensitive (you log-on once, the customer maintenance program displays the old record before you can change it, etc.).

The only things guaranteed in an Internet connection are that you will lose information and that there is a potential for sabotage.

We realized that we could not make the Internet completely fail-safe, so we changed our ambition to – "how can we protect data integrity in our application using an unreliable connection?" The answer was again a paradigm shift; our transactions needed to be stateless.

The transactions were already restricted to one task; now we went a step further to make them stateless and binary – either the task is performed correctly or it is not performed at all. In addition, we needed a new mechanism for the user to see if an update took place or not.

Fast response is extremely important in a web environment, as web users have no patience for slow systems. If you start a search and nothing happens, you just repeat the search. VPLUS stopped phantom transactions by applying a keyboard lock while the HP 3000 was processing a previous input – that safety net does not exist in the web environment. A common web programming technique - screen refresh to keep the page current after a change - can also create a lot of phantom transactions (the user changed the ship-to address leading to a screen refresh, triggering not only GetShipTo but all other transactions which define the order page). The result is many more transactions than you would expect based on the HP 3000 system usage.

The solution is to implement a very scalable system. The design goal that the application should consist of binary, stateless transactions is a good start, since it allows us to duplicate the application on several machines. Techniques that wrap existing code in a web services layer can get you up fast, but it will be much harder to scale the system if you still have state dependencies in the code.

Two old HP 3000 habits are so valuable that we decided to keep the inheritance even though they don't represent the standard way of doing things:

- The VPLUS limitation to display a page at a time turned out to be a great feature. Long searches, e.g. last name = Smith, which normally would be completed and presented as a scrolling list with hundreds or thousands of names, are replaced by paging. The technique is not web standard but the response time is so much faster than the alternative.
- Windows and web users expect on-line reporting, which works fine for demos but often time-out in the real world when the program has to read millions of records. In the new web version of SHIP4U, we added the old HP 3000 way of doing things - a report request screen which starts the report in the background. As a service to the user, we convert the report to HTML by default, so the user can view the result in the browser.

The Internet is open for sabotage, so the application needs to be much more secure than our traditional VPLUS code. Several layers achieve this:

- the system is using SSL encryption to avoid someone "listening".
- all users have unique identities and secret passwords (which are updated every month) and the authentication takes place on the application server.
- the web-server has only presentation logic and no data. A hacker cannot steal information unless he hacks several machines.
- the Data resides on the application (or data base) server, and is not directly accessible from web. Data access methods such as ODBC are not allowed from the web server, as it would leave the data base vulnerable if a hacker found password information. A firewall or router between web and application server that only allows specific transactions between the servers can enhance the architecture security.
- all access control is on the transaction level, i.e. if a user uses our order screen to delete 500 orders and five different transactions are used to search, display and delete the order then the access control will be performed 2,500 times.

To create a cost-efficient development environment

The challenge is to be able to continue with a small project group; to allow the web designers to create the presentation layer without knowing the HP 3000, and to let the COBOL programmers concentrate on the application logic.

A successful application is a balance between technique, presentation and business logic. Keep it as simple as possible, but not simpler.

The transaction model is similar to the new web-services concept and our web application could have been implemented as several hundred web services. The reason we implemented our own method is simple: web-services was not a reality when we started the project. If we had started today the choice would have been much harder: dynamic XML or strict web services, maximum performance or standard compliance.

Perhaps the best alternatives is a compromise, implementing 80% - 90% of the code as transactions (to hide the complexity from the programmers and the functionality from the outside world), then add the extra XML headers and definitions for the transactions that should be accessible to the public as web-services.

The best systems are so intuitive to the users that they hardly notice the design behind it. But application development is full of paradoxes. It is easy to create complex systems that are hard to use, but very difficult to create simple systems without sacrificing functionality.

One very important aspect of our VPLUS, COBOL and Image environment was its technological simplicity; everyone could concentrate on the application instead of the technique. In the new environment, the application focus is constantly threatened by a changing technology.

The web environment offers too many design options. We learned by mistakes that we needed standards to restrict our web developers. A lot of features can create attention to your web site, but it is not cost effective to replace hundreds of VPLUS screens without standards and it would not give you an easy to use system.

At the same time our COBOL developers needed new standards to be able to switch to the transaction model.

Our conclusion was to isolate both our COBOL and web developers from the technology by hiding it in the middleware and in the standard-based interface between presentation and application.

To be able to migrate the application without changing the presentation

This issue worried us at the start as Hewlett-Packard's decision forces us to migrate the application to another platform. We do not want to change the presentation again, neither when migrating the application or the data base to another platform. However, our previous discussion leading to state-less transactions, middleware and strict interfaces solved that issue. The ASP code on the web server does not even know where the data comes from, so once the HP 3000 middleware (which is very compact) is moved to the new platform, the same transactions can exist on the new platform.

```
           12:44:03 2I WebMain -2  769 GetUser      N
: Buf len +000000318

:  [<?xml version="1.0" encoding="UTF-8"?>
               <SHIP4U><UDPBuf Ip-re]
:  [ply="" Socket-reply="" Request="GetUser" Request-date-time="]
:  [2003-04-15T12:45:32" Transaction-no="100308" Sequence-no="1"]
:  [ Client-id="" Call-center="PCA" User-id="GUNNARF" Max-lines=]
:  ["25"></UDPBuf><GetUser User-id="GUNNARF" Call-center="PCA"><]
:  [/GetUser></SHIP4U>]
```

```
          12:44:03 3I WebMain -2  769 GetUser      N    1 1
: Buf len +000000857

:  [<?xml version="1.0"?><SHIP4U><UDPBuf Request="GetUser" Reply]
:  [-date-time="2003-04-15T12:44:03" Transaction-no="100308" Seq]
:  [uence-no="1" Status="ok" Sequence-status="Unique" Client-id=]
:  ["" Call-center="PCA" User-id="GUNNARF" Program="WebMain -2" ]
:  [Sub-program="CS-GETUSER" /><GetUser User-id="GUNNARF" Call-c]
:  [enter="PCA" Name="Gunnar Fredlund" Title="CEO" Telephone="80]
:  [5-898-2444" Ext="" Home-telephone="805-682-5572" E-mail="gun]
:  [narf.ciotech.com" Start-date="1987-08-27" Finish-date="" Rea]
:  [son="" Printer="LP" Location="0001" Client-id="" Account-mgr]
:  [-sw="Y" Call-center-access="Y" Client-set-up-access="M" Syst]
:  [em-set-up-access="M" Reports-access="M" Cancel-ord-access="M]
:  [" Change-ord-access="M" Refund-access="M" Return-access="M" ]
:  [Exchange-access="M" Replacement-access="M" Create-ord-access]
:  [="M" Continuity-access="M" System-name="TEST4U" Serial-acces]
:  [s="M" /></SHIP4U>]
```

This example shows the XML communication between the HP 3000 and the web-server. The web server can receive the exact same XML format from a Linux or Win2000 server, it cannot tell the difference

The concept is so powerful that the transactions do not need to come from one single platform, but can come from mixed HP 3000, Linux and Win2000 platforms. We can migrate one module at a time, and we also got a new way to implement batch jobs: the batch job can create the XML and call the transaction as easily as the web browser.

The code sample below shows an extract from a batch program that uses the ReciveReturn transaction to return all customer orders listed in a batch file. The COBOL program calls our Omnihost middleware (via "cudpsend" and "sudpread") to request transactions and receive the results. The code shows how easy it is to create the XML format, to use cudpsend to send the XML-transaction, and to use sudpread to receive the reply from the application.

```
       MOVE "All-lines-sw"               TO XML-OUT-ATR-LABEL (3).
       MOVE "Y"                          TO XML-OUT-ATR-VALUE (3).

       MOVE "Line-no"                    TO XML-OUT-ATR-LABEL (4).
       MOVE " "                          TO XML-OUT-ATR-VALUE (4).

       MOVE "Return-qty"                 TO XML-OUT-ATR-LABEL (5).
       MOVE " "                          TO XML-OUT-ATR-VALUE (5).

       MOVE "Return-restock"             TO XML-OUT-ATR-LABEL (6).
       MOVE "N"                          TO XML-OUT-ATR-VALUE (6).

       MOVE "Return-goods"               TO XML-OUT-ATR-LABEL (7).
       MOVE "N"                          TO XML-OUT-ATR-VALUE (7).

       MOVE "Received-loc"               TO XML-OUT-ATR-LABEL (8).
       MOVE " "                          TO XML-OUT-ATR-VALUE (8).

       MOVE "Authorization"              TO XML-OUT-ATR-LABEL (9).
       MOVE "WEBMGR"                     TO XML-OUT-ATR-VALUE (9).

       MOVE "Mailing-date"               TO XML-OUT-ATR-LABEL (10).
       MOVE DA-CURRENT-DATE              TO WS-DATE.
       MOVE " "                          TO XML-OUT-ATR-VALUE (10).
       STRING WS-YEAR "-" WS-MONTH "-" WS-DAY DELIMITED BY SIZE
                                         INTO XML-OUT-ATR-VALUE (10).

       MOVE "RMA-no"                     TO XML-OUT-ATR-LABEL (11).
       MOVE " "                          TO XML-OUT-ATR-VALUE (11).

       MOVE 11                           TO XML-OUT-AT-NO.
       CALL "X-OUT-ADD-ELEMENT" USING GLOB XML-OUT XML-BUF.
```

```
*** FINISH XML DOCUMENT
       PERFORM CAB000-FINISH-XML.

*      CALL "X-PRINT-BUF" USING GLOB XML-BUF XML-OUT-LEN

*** SEND REQUEST ***
       CALL "cudpsend" USING UDP-SOCKET-OUT UDP-PORT-OUT
                             PARM-IP XML-BUF XML-OUT-LEN GIVING UDP-STAT
       IF UDP-STAT <> 0
           DISPLAY "cudpsend error " UDP-STAT
           STOP RUN
       END-IF.

*** RECEIVE REPLY ***
       MOVE UDP-MAX-LEN                  TO XML-OUT-LEN
       COMPUTE UDP-CLEN = FUNCTION LENGTH (UDP-CLIENT-ADDR)
       CALL "sudpread" USING UDP-SOCKET-IN UDP-CLIENT-ADDR
                             UDP-CLEN UDP-TIMEOUT XML-BUF XML-OUT-LEN
                        GIVING UDP-STAT.
       IF UDP-STAT <> 0
           DISPLAY "cudpread error " UDP-STAT
           CALL "FATAL-ERROR" USING GL-ERR-MSG
       END-IF.
```

To create a cost effective process to "webify" other VPLUS applications

This step is an ongoing process, as there are many legacy systems with sound application logic but an outdated presentation layer. The framework can also be used to add new features and modules to transformed systems.

Our first thought was to create an automatic translation between our existing COBOL code and the new web transactions. We more or less abandoned that approach when we realized how tiny a part of the code we needed to reuse to maintain the application logic. In our old COBOL programs we had used separate sections for data validation, which made it easy to isolate the application logic and copy it to the new transaction structure.

Instead, we streamlined our middleware, and our standards for COBOL and the Web, in order to implement features such as access control in middleware, sub programs for application logging, debugging and monitoring tools.

The critical factor so far has been web development; the COBOL coding has been fast and easy. To find a good standard for web applications was a major challenge and we just recently implemented it. With the new standard in place, the web development should be much more efficient and it will be time to look at automating the COBOL process.

Summary

It is possible to transfer a VPLUS system into a modern looking web-application in a cost-effective manner and at the same time minimize the risks. The process maintains the old application logic, even though the result is a partial rewrite. The process starts a profound paradigm shift - it would be impossible, to go back and create another VPLUS system or a hierarchical menu after this experience. However, even though everything looks new and different, it has a familiar feeling behind the surface – the core values from the HP 3000 era remain.

ScreenJet VPLUS to ACUCOBOL-GT and AcuBench

By Alan Yeo, ScreenJet

About the Author: Alan Yeo is CEO of ScreenJet Ltd, a UK-based company set up in 1999 to co-ordinate the design and development of the ScreenJet emulation and graphical user interface products for the HP 3000. Alan has also been the CEO of Affirm Ltd. since 1986. Affirm is an HP 3000 ISV specializing in manufacturing applications on the HP 3000. Alan has been designing, developing and implementing VPLUS-based applications since 1980. www.screenjet.com

Overview

The ScreenJet VPLUS conversion tool kit is aimed at HP 3000 sites with VPLUS applications written in COBOL. The migration target can be any of a wide range of platforms. The tool kit extracts screen information from VPLUS Formfiles and delivers it as ready-made GUI screens in the Acucorp, Inc. AcuBench IDE (Integrated Development Environment), as though the screens had been created initially in that IDE.

An optional replacement VPLUS API allows migrated HP 3000 COBOL applications to retain their existing VPLUS calls. At run-time the API resolves these calls and makes the appropriate calls to the new ACUCOBOL-GT GUI screens.

This combined ScreenJet solution means that there are no coding changes required to your existing COBOL code. Generated code and VPLUS API are all in ACUCOBOL-GT giving you a full native solution on your target platform.

For those familiar with developing on the HP 3000 in COBOL and VPLUS, no new language skills are required when porting to ACUCOBOL-GT. AcuBench also allows those familiar with Windows GUI development to integrate their skills with ongoing development of the applications. COBOL programmers not already familiar with Windows GUI development can quickly learn these techniques due to the AcuBench COBOL-friendly design.

Why ACUCOBOL and AcuBench?

ScreenJet had already spent a considerable amount of time and effort to develop a Thin Client GUI enhancement product for VPLUS on the HP 3000, when HP announced that they were going to discontinue the HP 3000. We had to look for a suitable environment to migrate COBOL and VPLUS applications, whilst still converting the restricted abilities of VPLUS into a true Graphical User Interface.

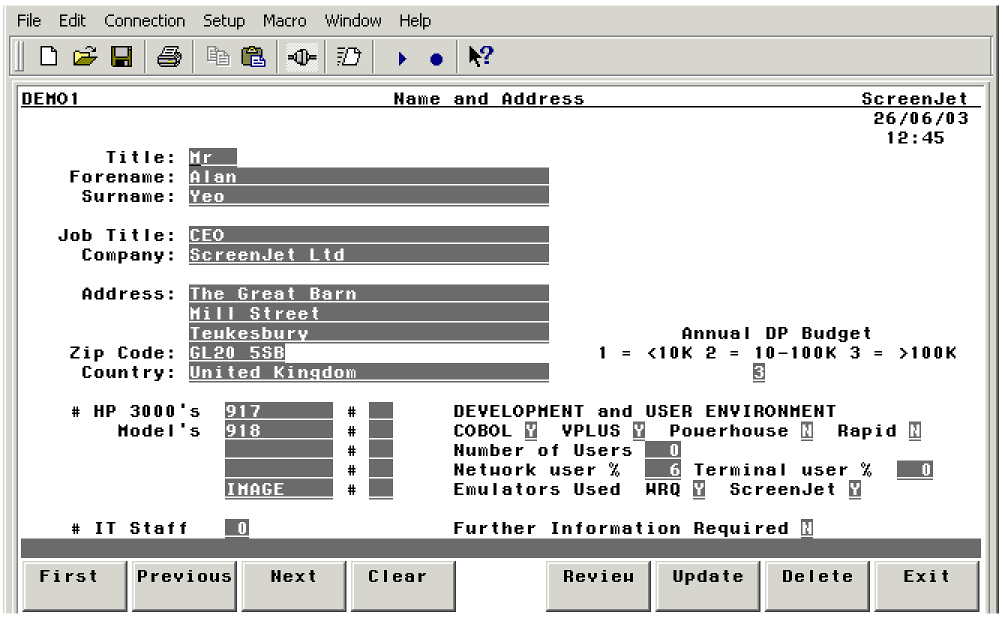
In ACUCOBOL-GT we discovered not only a level of compatibility with HP COBOL unsurpassed by other suppliers, but also a true Integrated Development Environment (IDE) with exactly the same model we were used to in VPLUS, where screen development is independent of the underlying application code

ACUCOBOL-GT is closely compatibile with HP COBOL, including the HP extensions, and is also the only other COBOL available on the HP 3000, thus allowing a vast amount of migration work to be tested on the HP 3000. In addition we were aware that other 3rd-parties were developing ACUCOBOL compatible tools for the 3000.

The AcuBench GUI screen structures support a direct conversion from VPLUS. AcuBench generates COBOL code, so that no new language skills are required before, during or after migration. Thin Client GUI deployment preserves the best of the Server Side development and deployment features found on the HP 3000. With ACUCOBOL we had a VPLUS/COBOL migration solution that was virtually independent of hardware platform and operating system, and thus enable migrated solutions to run on any of the 600+ supported ACCCOBOL platforms.

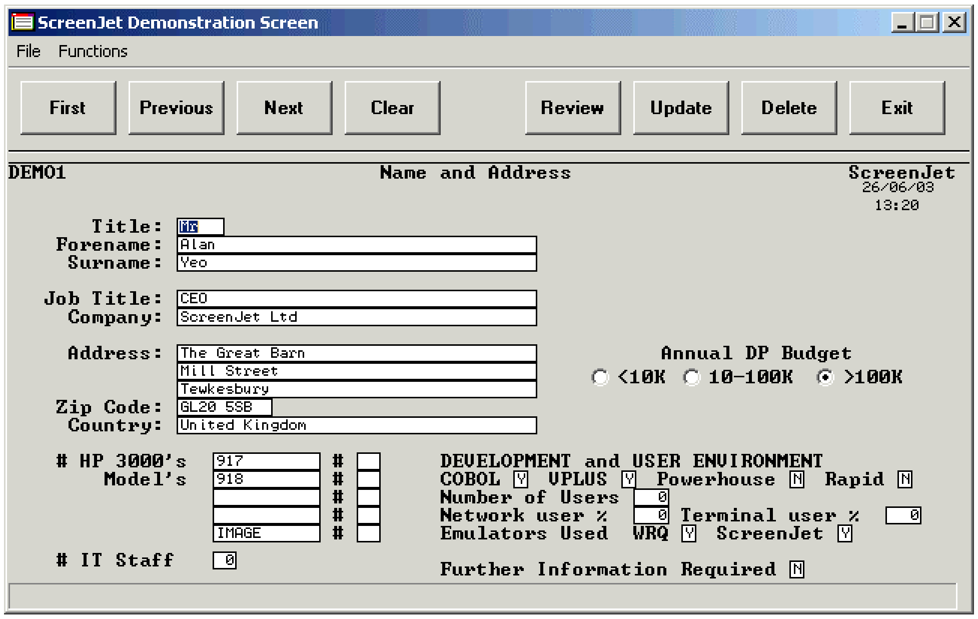
For rapid, risk-free migrations, a replacement user interface could be created that looks and functions almost identically to the original VPLUS interface, so that user re-training costs are reduced. The converted screens are in a format such that they can subsequently be enhanced in a step-by-step manner to incorporate additional Windows GUI functionality, using the AcuBench tool set.

Conversion Sample from this Original VPLUS Screen



```
DEMO1                         Name and Address                       ScreenJet
                                                                       26/06/03
                                                                          12:45
      Title: Mr
   Forename: Alan
    Surname: Yeo

  Job Title: CEO
    Company: ScreenJet Ltd

    Address: The Great Barn
             Mill Street
             Tewkesbury                        Annual DP Budget
   Zip Code: GL20 5SB                   1 = <10K 2 = 10-100K 3 = >100K
    Country: United Kingdom                              3

   # HP 3000's  917       #       DEVELOPMENT and USER ENVIRONMENT
       Model's  918       #       COBOL Y  VPLUS Y  Powerhouse N  Rapid N
                          #       Number of Users  0
                          #       Network user %   6 Terminal user %   0
                IMAGE     #       Emulators Used  WRQ Y  ScreenJet Y

   # IT Staff   0                 Further Information Required N
```

Generated GUI screen (with addition of Radio Button group):



```
DEMO1                         Name and Address                    ScreenJet
                                                                   26/06/03
                                                                     13:20
   Title: Mr
Forename: Alan
 Surname: Yeo

Job Title: CEO
  Company: ScreenJet Ltd

  Address: The Great Barn                        Annual DP Budget
           Mill Street                     ○ <10K  ○ 10-100K  ● >100K
           Tewkesbury
 Zip Code: GL20 5SB
  Country: United Kingdom

 # HP 3000's  917     #      DEVELOPMENT and USER ENVIRONMENT
    Model's   918     #      COBOL Y  VPLUS Y  Powerhouse N  Rapid N
                      #      Number of Users  0
                      #      Network user %   0 Terminal user %  0
              IMAGE   #      Emulators Used WRQ Y  ScreenJet Y

 # IT Staff  0                Further Information Required N
```

The Conversion Software

The ScreenJet conversion software has two components, both of which run on the HP 3000. Firstly, there is an extract program that reads the VPLUS Formfile to obtain all the information about the FormFile, the layout of the Forms, and the attributes of every individual label and field including any processing specification code. This information is stored in a database on the HP 3000 from which the subsequent selected ACUCOBOL formats are generated.

This database provides the flexibility that enables the co-existence of ongoing development work in VPLUS on the HP3000 whilst the conversion project is underway. Individual forms can be re-extracted from the Formfile as required, or new ones added into the database.

The second part of the conversion is a generation phase. You can select either an individual form or all the forms of a Formfile, and the desired conversion options. The generation process then creates the appropriate files and copy books for subsequent use by AcuBench. These files are created in formats so that they can either be accessed directly on the HP 3000 from AcuBench, if you are using software such as Samba, or they can be simply transferred to another server or the workstation where the AcuBench development is going to be undertaken.

Sample Conversion Dialog

```
:HELLO MGR.SJET3000,ACUDEMO
:EXTRACT
SNJ0710'02.10'A00  Copyright (C) ScreenJet Limited 2002

   Enter ScreenJet Application Id: DEMOF

            Enter Forms File Name: DEMOF
Enter Form Name (CR or @ for all): @


15:46:06 Processing DEMO1
15:46:10 Processing DEMO2
15:46:15 Processing DEMO3
15:46:19 Processing DEMO4


END OF PROGRAM
```

```
:GENERATE

SNJ0720'02.10'A00.06  Copyright (c) ScreenJet Limited 2002

Enter Application ID: DEMOF
     Enter Screen ID: @

16:04:29 Processing DEMOF
16:04:35 Processing DEMO1
16:04:59 Processing DEMO2
16:05:24 Processing DEMO3
16:05:47 Processing DEMO4


END OF PROGRAM
```

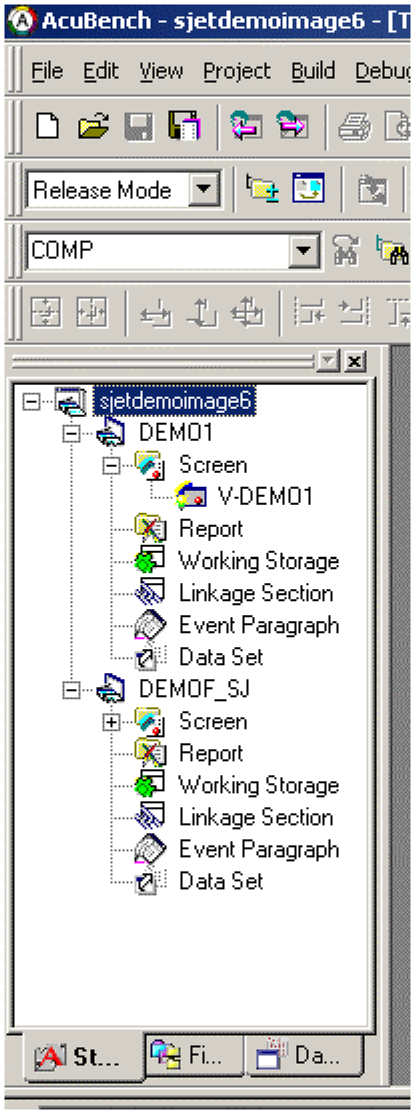
Generate will have created several files in the directories within ACUDEMO:

```
:LISTFILE ./

PATH= /SJET3000/ACUDEMO/CPY/

DEMO1.CPY     DEMO21.CPY     DEMO3.CPY     DEMO4.CPY     DEMOF_SJ.CPY

PATH= /SJET3000/ACUDEMO/PSF/

DEMO1.PSF     DEMO2.PSF     DEMO3.PSF     DEMO4.PSF     DEMOF_SJ.PSF
```

DEMO(n).CPY are the Copybook(s) generated for the form(s) DEMO1 etc., DEMOF_SJ.CPY is the Copybook generated by ScreenJet for the global DEMOF formfile information.

DEMO(n).PSF are the AcuBench Program Structure File(s) for the form(s) DEMO1 etc., DEMOF_SJ.PSF is the global PSF file for information in the DEMOF formfile.

Compiling and Maintaining the Screens in AcuBench



Whilst you may create any physical development environment you like, by default the AcuBench IDE creates a standard and straightforward directory structure to help manage all of the components of your applications. A single AcuBench project may contain all the Programs, Copy Books, and Screens used in an entire application, or you may choose to split the application into more logical project sections. Whilst individual screens can be split into separate AcuBench projects, it is likely that you will choose to keep all the generated screen Program Structure Files from a Formfile in a single AcuBench project.

Conceptually, you can think of this as your replacement Formfile, which allows you to easily continue development of the screens, and allows the generation by AcuBench of a compiled Library of screen programs for run-time execution (rather like a fast Formfile).

The addition of the screens into an AcuBench project is logical and simple. Firstly, from AcuBench you browse to the location where you have placed the generated Program Structure (.psf) and Copy Books (.cpy) files. This may be on the HP 3000, another server or locally on the workstation where you are running AcuBench. A simple mouse click adds them into the project. AcuBench then provides an option for you to generate the ACUCOBOL Source Code from the Program Structure Files.

In the Structural View taken from AcuBench shown here, DEMO1 is the .psf generated by ScreenJet for the DEMO1 form, and DEMOF_SJ is the global .psf generated for the DEMOF Formfile.

Under DEMO1 are the standard components generated by AcuBench:

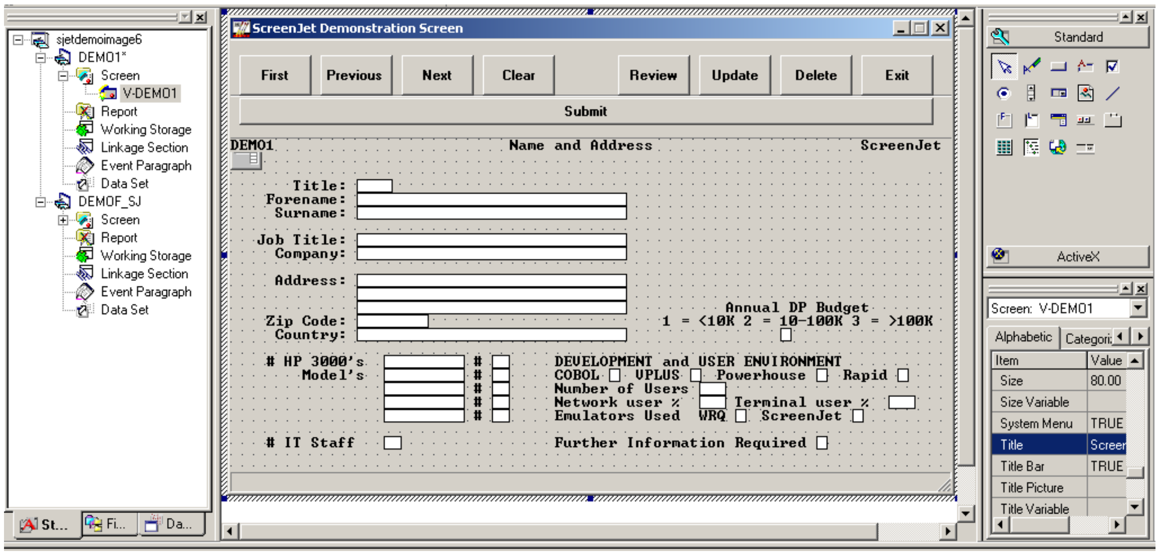
1. Screen, which contains the generated V-DEMO1 GUI screen.

2. Working Storage, which contains the definitions of all the objects (fields) on the screen.

3. Event Paragraph, which contains all the generated code to display the screen, and which also contains all the converted processing specification code.

Note that within AcuBench the physical screen has been prefixed with a “V-“ because screens are objects in ACUCOBOL, and objects cannot be named COBOL reserved words, unlike in VPLUS where Form Names could be. Your application code can continue to reference the screen as “DEMO1” however the code generated by ScreenJet maps DEMO1 to V-DEMO1 to ensure that no changes have to be made to your application.

Double clicking on the V-DEMO1 object will open the screen in the Windows GUI design window as shown below.



Screen: V-DEMO1

Item	Value
Size	80.00
Size Variable	
System Menu	TRUE
Title	Screen
Title Bar	TRUE
Title Picture	
Title Variable	

Once converted, you can immediately rearrange fields on the screen or start adding additional graphical components, or convert fields to objects such as Radio Buttons and Check Boxes. Much of this can be done transparently to your application code.

The Conversion Process

The conversion of HP 3000 VPLUS forms into the ACUCOBOL™-GT GUI screen formats has two main options:

1. The screens can be converted directly into Acubench Screen Template Files (.stf) files with accompanying copylib (.cpy) files, or
2. Program structure files and associated copylibs can be created.

Why Two options?

Whilst both options generate native ACUCOBOL GUI screens, the several ways that the application under conversion may have been written that will cause you to prefer one of the options over the other.

With the Program Structure File option, in essence ScreenJet creates an individual native ACUCOBOL program for each VPLUS Form that is converted. To complement these programs ScreenJet supplies a replacement VPLUS API Library that allows your existing application code to retain all its original VPLUS calls and procedures. The API intercepts these calls and calls the generated screen programs to perform all the processing that your application expected the VPLUS API to do on the HP 3000.

The Screen Template file option creates the Screen Section code used by ACUCOBOL programs to generate and display GUI screens. However, to use this option the VPLUS calls in your application will need to be modified to make the appropriate ACUCOBOL screen calls.

Which is the best option for me?

This really depends on how the application you wish to migrate was written, how much coding to wish to do during the migration, and where you want to end up.

With the Program Structure File conversion, virtually everything is done for you, apart from the testing. You will be given native AcuBench Program Structure Files upon which you can immediately start adding the wealth of GUI functionality available in ACUCOBOL using the AcuBench Integrated Development Environment (IDE). No changes are required to your application source code, all VPLUS calls are resolved by the ScreenJet VPLUS API.

The downside is that you will have to licence ScreenJet's VPLUS API. If you want to utilise the development power of AcuBench for rapid migrations, or your application is constructed with VPLUS calls scattered throughout the code, this is an ideal conversion option. The VPLUS API also provides a complete translation and execution for the VPLUS processing specification code provided in VPLUS. Therefore if an application has made extensive use of this feature, this option provides a directly replaceable function under ACUCOBOL.

With the Screen Template File option, all the Screen Code and necessary copy libraries are generated for you, but the logic supplied by the VPLUS API is not provided. The generated screen code will need to be incorporated with your application code, which will need to be modified to remove the VPLUS procedure code and replaced by ACUCOBOL screen handling code.

If you have an application where all the VPLUS logic is contained in a central library or subroutine, it is an ideal candidate for the .stf option. Your only recoding effort will be in one self-contained routine, which when working, will work for all of your application. No further licensing of the VPLUS API will be required. The downside is that as your application will no longer have an external procedure library to call as with VPLUS, any Processing Specification code will have to be now added to the application code.

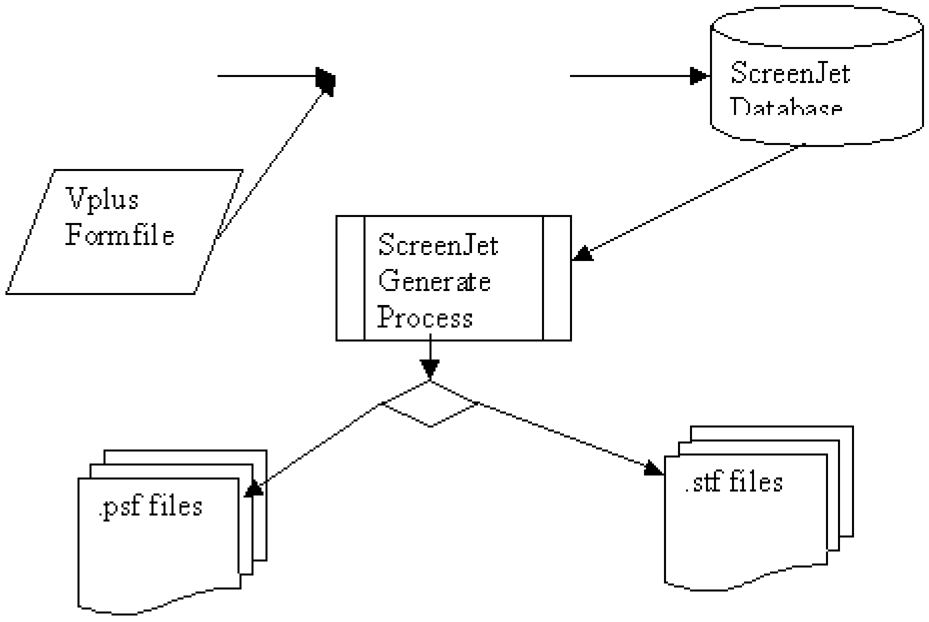
Is my initial choice final?

No.

If you choose the .psf route and to initially retain all your VPLUS calls, you can subsequently use AcuBench to generate the Screen Template Files and then either completely drop the need for the VPLUS API or slowly replace it as required, program by program, by replacing the VPLUS calls and transferring the generated Processing Specification code into the appropriate program sections.

If you choose the .stf option then your subsequent choices are slightly more restricted, as it is not possible to generate .psf files from the .stf format. However, if you decided that the .stf route was initially preferable, but for whatever subsequent reason decided that the .stf would have been the best choice, then the ScreenJet Conversion tool which extracted all the Formfile information before the conversion choice had to be made, can be re-run to generate the alternate format.

How does the Conversion Process Work?



What files are created?

The Screen Template File Option

With this option, an .stf file is created for each selected form; this is the ACUCOBOL code to create the GUI screen, together with a .cpy file copy book that defines the physical buffer layout of the fields on the screen. For example, if the .stf files are generated for Form "ORDERF" in Formfile "SALESF", then two files ORDERF.stf and ORDERF.cpy are generated.

In effect, with this option the conversion creates all the code and data structures for the GUI screens to replace the VPLUS Forms, but the "GLUE" code is required. To utilise these files you will need to copy them into the appropriate sections of the application code you have transferred into AcuBench. This may be your existing Screen Handling Program, a new Screen Handling Program or even the individual programs that previously used a VPLUS screen. The choices are yours, but all the work of creating the GUI screens that are true representations of the original VPLUS forms has been done for you.

The Program Structure File Option

With this option, a .psf file is created for each selected form, and a global one for the Formfile. These files are in the internal format used by AcuBench, as they would be if you had created a GUI screen program from scratch in AcuBench. In addition a .cpy file copy book that defines the physical buffer layout of the fields on the screen is created, plus a global one that contains shared information for all the forms in a Formfile. For example if the .stf files are generated for Form "ORDERF" in Formfile "SALESF" then four files ORDERF.stf , ORDERF.cpy and SALESF_SJ.stf, SALESF_SJ.cpy are generated.

There is a fundamental difference between the ScreenJet .psf conversion mode and the .stf mode and the other VPLUS conversion to ACUCOBOL options available. All other options generate COBOL code, so that from the point of conversion onwards all the maintenance of the screens is in COBOL. With the Program Structure File option from ScreenJet, a .psf file is created in the internal AcuBench Structure format.

This format allows you to use all the power of AcuBench immediately as the files are generated, so that full use can be made of the Screen design tools, windows objects, automatic code and copy book generation options. With this option your application development is suddenly exposed to the power of a modern IDE and all the benefits that brings regarding retaining staff and training new staff.

With this option, the conversion has created all the code and data structures necessary for the GUI screens needed to replace the VPLUS Forms, and has also created all the "GLUE" code required to utilise the screens.

What types of VPLUS Formfile can be converted?

The ScreenJet extract process can work on both VFORM and VFAST formfiles. However there are a couple of things that the current version cannot extract from a VFAST formfile. These are the screen description that we use for a default title bar for the generated GUI screen, and any processing specification code. So if you have lost the source for your VPLUS Formfile, the conversion tool can still do most of the work for you.

What doesn't the conversion handle?

There are some attributes of a Formfile that the ScreenJet tool doesn't convert or the VPLUS API does not handle. We have split these into two classes:.

Those that are redundant, inappropriate for a COBOL application, or that no device other than an old HP2645 terminal could handle. These we have no development plans to address, although as you will see from the later section on the VPLUS API, no VPLUS call should cause a failure.

We have also not implemented some of the very uncommon functions, or where design input would be required from users before we could decide the correct emulation. Unless specified below all other VPLUS data types, functions and calls are converted and supported.

1. Things not supported in current release, but which are being planned for future releases. (Please check web site www.screenjet.com for current status on these items).

a) Freeze/Append. This is a uniquely a terminal mode function where an area of the screen can be frozen on the display, and in effect discarded by the underlying program which then continues to append further forms under the frozen area of the display. This appending can have the appearance that the bottom area of the screen is scrolling under the frozen portion. The frozen portion of the display is not accessible to the user or the application but it is entirely up to the underlying application code when the frozen area of the display is removed. This type of processing is fairly alien to a Windows GUI or Browser Interface. Currently if Freeze/Append is used with GUI screens, the frozen screen is just removed and only the appended screen is displayed. The approach we are taking to this problem will provide a mechanism where the frozen screen is set inactive and uncloseable by the user, whilst append screens will be displayed in an additional window. Whilst this approach will work and will be better than the current process, it will appear strange to traditional Windows application users. Therefore unless Freeze/Append has been used extensively in an application, it may well be worthwhile considering revising the display logic at this time.

b) Application Ready Buffer. This was a fairly late development in VPLUS. The Application Ready Buffer allowed an additional formatting step after the data had been returned from the display and Field and Finish formatting had been performed. The intention being that this formatting could reduce even further the amount of work that the application code had to do to utilise the data. Whilst we are aware that some people may have used this feature, to date nobody has asked for it.

c) "VGETSAVEFIELD", "VPUTSAVEFIELD". Save fields are fully supported in the processing specification, however these two fairly recent additional programmatic calls to retrieve and set these values are not supported in the current release, but support is due in version 01.00.01 due Qtr 3 2003.

d) "VTURNOFF", "VTURNON". These calls that turn Block Mode on and off are partially supported by the VPLUS API by the setting of an Environment Variable

"SJ_MIXED_MODE" in the run-time parameters. This will cause the API to clear the screen. We hope in due course to be able to have the ACUCOBOL run-time do intelligent switching between GUI mode and Character Mode display windows.

e) "VPRINTSCREEN". With the transition to GUI screens there are many more ways to get full screen image shots that make the crude character representation performed by VPRINTSCREEN redundant. However, if you are running ScreenJet generated ACUCOBOL GUI screens on a Windows server, VPRINTSCREEN will direct a graphical screen print to your local printer, complete with date and time stamp.

2) Things for which support is unlikely, ever.

a) CONFIG Phase Statements. These statements are only actioned by VPLUS if it detects that the Terminal is an HP2645; otherwise it ignores them. As we suspect that the number of 2645 Terminals still in use for applications can probably be counted on one hand, we have chosen to not implement this functionality. It is also highly unlikely that any modern display device will be equipped with the lights, internal tape drives and paper tape readers that the Config statements used to control.

b) "VGETIEEELONG", "VPUTIEEELONG", "VGETIEEEREAL", "VPUTIEEEREAL", "VGETREAL", "VPUTREAL", "VGETLONG", "VPUTLONG". These number types are not supported by COBOL therefore it is highly unlikely that you will be attempting to port an application to ACUCOBOL-GT that uses these functions.

c) "VLOADFORM", "VUNLOADFORM". These commands caused VPLUS to load and unload forms into the display memory of particular types of Terminal when it detected their use. These functions have no modern equivalent, and as VPLUS ignored them unless specific terminal types were used, we have adopted the same approach with the ScreenJet VPLUS API.

d) "VARMSCP", "VGETSCPFIELD", "VGETSCPDATA". These cursor-positioning commands returned the field and the data from the current cursor position.

e) "VBLOCKREAD", "VBLOCKWRITE". These functions would read the whole screen including all the formatting escape sequences and, as such, have no GUI counterparts.

f) "VOPENBATCH", "VCLOSEBATCH", "VREADBATCH", "VWRITEBATCH", "VPOSTBATCH". These functions deal with files of screen data (from the same screen). Few applications use this approach at this time.

VPLUS API

The VPLUS API is written in ACUCOBOL so it is as portable as the applications you convert. It is supplied as an ACUCOBOL Library (.Lib) object.

What happens if an application makes one of the unsupported calls?

Whilst there may be no action possible for the above calls, they are supported in that the VPLUS API will recognise them and just determine that there is no possible action to be taken. Therefore any of these calls will just return to the calling program without any action having occurred or updates to the Comarea been made. To facilitate the debugging of migrated applications ScreenJet can optionally write details of any un-supported calls to a log file.

Concluding: ScreenJet Conversion

ScreenJet is an "easy-to-install and use" HP 3000 toolkit that provides the ability to convert or re-convert forms "as and when" required, which for most users will provide the most cost-effective and flexible option. Whilst the automatic conversion process could theoretically process all of the forms in your applications in a single day, in reality the conversion of your applications is unlikely to be approached as a big bang. However, if you only have a small number of forms to convert, or you wish to sub-contract the task, ScreenJet does provide options to have forms converted on a form-by-form basis.

A New Database to Replace IMAGE

HP does not provide TurboIMAGE on an other platforms besides the 3000, so you will be using a new database if you migrate!

You may wonder what the difference is between IMAGE and a relational database – to clear up some of the confusion, read Eugene Volokh's paper "Relational Databases vs. IMAGE: What the Fuss is All About"

```
http://www.adager.com/VeSoft/RelationalAndIMAGE.html
```

> "What are **relational databases** anyway? Are they more powerful than IMAGE? Less powerful? Faster? Slower? Slogans abound, but facts are hard to come by... Relational databases are nothing revolutionarily new. Their main advantage lies not in any radically different data representation structure, but rather in a lot of useful functionality/flexibility features. All those functionality/flexibility features are really **great**. If [the database] doesn't degrade performance too badly, you ought to like the relational system a lot better than you do IMAGE. The performance of the system depends primarily on how good a job [the database] does of optimizing the preprocessing of embedded queries. You'll love embedded queries, if they're fast. If they're slow, you'll hate them."

This section of our book covers migrating your TurboIMAGE database to Eloquence, Oracle, SQL Server, mySQL, PostgreSQL or SapDB. We have articles here from many good authors including Michael Marxmeier, Terry O'Brien, Aaron Holmes and your editor, Bob Green.

Another interesting database to consider is Pervasive. See the chapter "An ISV Ports to Windows" on page 298, which interviews Steve Smith on why he may adopt Pervasive as his replacement database.

While planning your migration, you may need these 3000 reference books:

TurboIMAGE Reference Manual:

```
http://docs.hp.com/mpeix/onlinedocs/30391-90012/30391-90012.html
```

HP COBOL Manual:

```
http://docs.hp.com/cgi-bin/doc3k/B3150090013.11820/1
```

Data Definitions in the HP COBOL Manual:

```
http://docs.hp.com/cgi-bin/doc3k/B3150090013.11820/52
```

Suprtool User Manuals:

```
http://robelle.com/library/manuals/
```

Introduction to Eloquence

by Bob Green, Robelle

About the author: Bob Green is the founder of Robelle. Bob is impressed by the simplicity and power of Eloquence, and the ease with which Robelle ported Suprtool from IMAGE to Eloquence, even though the internals of Eloquence are completely different from IMAGE.

robelle

Eloquence is a database that may be of interest to MPE users because it is very much like the TurboIMAGE database that they already use.

Eloquence does not run on the HP 3000, but it does run on the HP 9000 (HP-UX), one of the target platforms for HP 3000 users if they decide to migrate their applications. Of course, we realize that some users plan to stay on their reliable HP 3000s indefinitely, and we think that is great. Others may decide to adopt a totally new application on a relational database. Fine again, if you can afford it.

But many sites are happy with their existing HP 3000 applications that have years of corporate brainpower invested in them and would like to replicate them on a platform with a longer future. For them Eloquence is a real option.

Is Eloquence a clone of TurboIMAGE?

No. Eloquence is an integrated development environment, including a programming language, a database, utilities, and runtime. It was originally developed as a migration platform for the HP 250/260 customers. It is currently supported on HP-UX, LINUX and Windows, including the Itanium (IA-64) architecture on HP-UX.

Internally, Eloquence is nothing like TurboIMAGE; it is more like an SQL database. It doesn't use the hashing that TurboIMAGE uses to find key values; instead it uses B-trees (the same technology used for indexed sequential access, as in KSAM files; the tree structure of key values grows as more values are added). It uses a client/server architecture, so the application program does not even need to know where the database is located. It doesn't put each dataset in a separate file; instead it mixes all datasets in the same set of files.

Is Eloquence a new, untested product?

No. It has been in production use for over a decade and has 2500+ installations, worldwide. Eloquence has had over 10 years of constant development and improvement.

Why does HP say that Eloquence is recommended for small to mid-size applications?

The reason has nothing to do with any known inherent limits in Eloquence. It has to do with lack of experience. Eloquence developed as a database engine for applications aimed at smaller IT problems. As a result, the largest actual Eloquence database to date is smaller than the largest TurboIMAGE databases that users have experience with.

What size applications are we talking about?

Eloquence is currently deployed in applications with 250+ concurrent users and 10+GB of data. There is no hard limit in Eloquence that prescribes these sizes. They are just the largest applications that have been seen so far. It is expected that experience with larger applications will appear over the next year.

Will my programmers have to rewrite COBOL programs?

No. Eloquence supports the TurboIMAGE intrinsics (i.e., API, application program interface). It has the same parameters, same options, and returns the same results. Your programmers will be happy.

Will my Suprtool tasks need to be changed?

No, the logic and syntax is the same. There is one change, due to the extra flexibility of Eloquence. That is the Base command, which connects Suprtool to the database. Since Eloquence uses a client server architecture and allows you to connect to databases and servers on multiple systems, the syntax for naming the database is different. But that is a mechanical edit of the Suprtool jobs, where necessary.

There are a few Suprtool/MPE features which are currently not available on Suprtool for HP-UX: Edit command (Dbedit). Extracting a range of fields from an SQL database, Export command (STExport exists as a separate program), Hints are not available, Link command (Suprlink exists as a separate program), Table Command with the File option requires that the file being loaded is self-describing, Out= option of the Listredo command, Output Ask, Num,Key, and Num,Query, Output Temp (there are no temporary files in HP-UX), Output =Input, Totals to a file.

Does Eloquence have the bells and whistles of Oracle?

Some. Eloquence has integrated online backup. And because of the client/server model, remote database access becomes a given. Optional Eloquence products include an ODBC driver, a graphical user interface, and the SQL/R report writer which allows you to write reports against Eloquence databases in SQL syntax.

However, Eloquence does not have everything -- replication for example (an automatic copy of your data on another server for backup and performance reasons). Eloquence is working on a replication capability, but they do not have it today. Or constraints and triggers, which enforce business rules within the database - Eloquence does not have them.

Will my operations staff need to learn new commands?

Yes, the operation of Eloquence is quite different from TurboIMAGE, since it is a client/server architecture. However, the good news is that once Eloquence is set up, it runs virtually unattended.

Is Eloquence hard to administer?

Some databases require a full-time DBA to tune and monitor them. Eloquence is mostly self-maintaining. If you don't change the database structure and you don't run out of disk space, there is little administration to do. This is due to the original target market: dedicated applications in small to medium-size businesses where there is probably not even a dedicated computer operator!

How does the performance of Eloquence compare?

Experience so far indicates that Eloquence is fast and efficient as a TurboIMAGE replacement (something that big SQL databases were never intended to do).

Is Eloquence 100% compatible with TurboIMAGE?

Almost 100% compatible. The recently released Eloquence version B.07.00 supports all the TurboIMAGE intrinsics and limits. For example,

- Number of data items per database: Eloquence is 2048; TurboIMAGE is 1200
- Number of data sets per database: Eloquence is 500; TurboIMAGE is 240
- Paths per data set: Eloquence is 16/64; TurboIMAGE is 16/64

Do I Have to Change my Locking Calls?

No, but there are new options which you may want to use. From the 2002 paper by Tien-You Chen of Hewlett-Packard and Michael Marxmeier.

```
http://eloquence3000.com/downloads/hpworld2002.pdf
```

"Locking with the Eloquence database is fully compatible with TurboIMAGE but provides additional options.

"The most visible difference is that locking is optional and the application is free to implement the locking strategy which suits best to its requirements. The Eloquence database neither depends on explicit locking (DBLOCK) nor does it impose restrictions on locking (like requiring a set lock for specific operations).

"Instead of verifying if a write operation (such as DBPUT) is covered by a DBLOCK, Eloquence verifies if a conflicting DBLOCK has been granted which covers the same data. In this case a database status is returned. This behavior is fully compatible since in TurboIMAGE all write operations must be covered by a DBLOCK and consequently there can be no concurrent DBLOCK.

"Eloquence provides read locks [*which are not provided in TurboIMAGE,*] and allows for selective unlocking. DBLOCK modes 11 to 16 add read locking at the database, data set or item level, equivalent to the DBLOCK modes 1 to 6. DBUNLOCK has additional modes which allow to selectively release a previously acquired lock by passing the same lock qualifier as used with DBLOCK.

"Eloquence allows applications to use multiple DBLOCKs and is able to detect and act upon a deadlock. A single DBLOCK can never cause a deadlock, but in case an application program has a DBLOCK and blocks while waiting for another one to be granted (unconditional DBLOCK) a deadlock condition could arise. For example, application 1 has a lock on the data set CUSTOMERS and waits to get a lock on PARTS while application 2 has a lock on PARTS and waits to get a lock on CUSTOMERS. This is detected by Eloquence and a status is returned."

Can I just "restore" my databases onto HP-UX?

Almost. Eloquence supports the same data types as TurboIMAGE, the same record layouts, and the same indexing (plus new options). However, the file formats and internal structures of Eloquence are dramatically different from IMAGE. Only the programming interface is the same.

The transformation needed to convert IMAGE databases to Eloquence is simple and is described on this web page:

```
http://hp-eloquence.com/hp3k/migration.html
```

"Moving your database to HP-UX involves the following steps:

- Install the DBEXPORT utility on HP3000
- Export the database
- Create and import the database on Eloquence"

What about B-trees and Third-Party Indexing (TPI)?

Indexed sequential access is inherent in the internal design of Eloquence. Indexes are available to applications to implement indexed database access, such as partial key retrieval, using wildcard or ordered retrieval. With Eloquence B.07.00, both the TPI programming interface as well as using indexes with master data sets (i.e. TurboIMAGE B-tree feature) are supported. Advanced TPI functionality, such as keywords, relational access and independent indexing, currently is not supported and requires a third-party indexing product such as Omnidex. DISC has indicated some interest in adding Eloquence support to Omnidex, but we have not seen any product or beta announcement (a search of their web site shows no references to 'Eloquence'). According to Bradmark's web site, Superdex is not available on UNIX, so that is not an option.

What about automatic dataset expansion (MDX and DDX)?

The automatic expansion feature was added to TurboIMAGE after years of user requests. It allows the IMAGE database to grow automatically as more data is entered. Because Eloquence stores data in a different way and uses B-trees instead of hashing, it expands the space for datasets automatically. It has always had this "feature".

What about the things we have been waiting for in TurboIMAGE?

Eloquence has a number of things that users have asked for in TurboIMAGE. Some of them are a little technical, but let's see if we can explain them.

Read-only DBLOCK: TurboIMAGE has only one kind of DBLOCK, which locks out all other attempts to DBLOCK on that data, even if the other program does not need to update it. Eloquence has an option to lock data so that other programs cannot lock and change the data, but allows other programs to do their own Read-only locks.

Selective DBUNLOCK: TurboIMAGE releases all your locks on the database; Eloquence has an option to release specific dataset and data item locks, while retaining the others. This makes it easier to program independent modules that deal with specific subsets of the data, such that they can be invoked from several different functions. For example, adding a new customer to the database.

Nested Transactions: Using DBXBEGIN before a "transaction" of database changes allows you to dynamically abort and back out the changes with DBXUNDO (and ensures that the changes will be backed out in case of system/program abort). This is called a dynamic transaction. Both TurboIMAGE and Eloquence have this, for single or multiple databases. However, TurboIMAGE does not allow you to begin a new transaction while an existing transaction is active, while Eloquence does. This makes it easier to write modular programs. For example, you begin a transaction to create a customer order, in the midst of which you sometimes start a nested transaction to create a new customer.

Who owns Eloquence?

Eloquence was based on the design of software for the HP 250/260 systems and was owned by HP Germany. However, Marxmeier Software AG (www.marxmeier.com) has always done the R&D for Eloquence. Recently, HP and Marxmeier made an important announcement:

"Effective November 20, 2002, Hewlett-Packard transferred the HP Eloquence product responsibility to Marxmeier Software AG... Marxmeier Software AG has been responsible for the HP Eloquence product development and back level support under contract with Hewlett-Packard for the past 14 years. Marxmeier Software has, in addition, been responsible for worldwide Eloquence distribution since 1997."

We see this as a valuable development for Eloquence users, because the most focused, qualified and motivated people will be in charge of Eloquence and its future.

What about pricing? Is Eloquence going to break my budget?

No. Pricing is very reasonable.

The maximum you can spend is $7740 per server. And that is for unlimited users. If you only have 8 users, the license fee is $1282.

Where do I purchase Eloquence?

You can purchase a license either from Marxmeier directly, or from one of several authorized resellers - the license fee does not include any technical support or software updates, which are purchased separately.

```
http://www.hp-eloquence.com
```

What about software updates?

Marxmeier offers software updates to Eloquence at a fee which is about 10% of the license fee annually. Updates can be obtained directly or through a reseller. Software updates do not include any technical support, which is purchased separately.

What about technical support?

You can purchase technical support from one of several authorized resellers of Eloquence.

Eloquence: HP 3000 IMAGE Migration

By Michael Marxmeier, Marxmeier Software AG

Michael Marxmeier is the founder and President of Marxmeier Software AG (www.marxmeier.com), as well as a fine C programmer. He created Eloquence in 1989 as a migration solution to move HP250/HP260 applications to HP-UX. Michael sold the package to Hewlett-Packard, but has always been responsible for Eloquence development and support. The Eloquence product was transferred back to Marxmeier Software AG in 2002. This chapter is excerpted from a tutorial that Michael wrote for HPWorld 2003. Email: mike@marxmeier.com

Eloquence is a TurboIMAGE compatible database that runs on HP-UX, Linux and Windows.Eloquence supports all the TurboIMAGE intrinsics, almost all modes, and they behave identically. HP 3000 applications can usually be ported with no or only minor changes. Compatibility goes beyond intrinsic calls (and also includes a performance profile.) Applications are built on assumptions and take advantage of specific behavior. If those assumptions are no longer true, the application may still work, but no longer be useful.

For example, an IMAGE application can reasonably expect that a DBFIND or DBGET execute fast, independently of the chain length and that DBGET performance does not differ substantially between modes. If this is no longer true, the application may need to be rewritten, even though all intrinsic calls are available.

Applications may also depend on external utilities or third party tools. If your application relies on a specific tool (let's say, Robelle's SUPRTOOL), you want it available (Suprtool already works with Eloquence). If a tool is not available, significant changes to the application would be required. Eloquence goes a long way to make sure that not only the intrinsic calls are compatible, but also reasonable expectations are met. We are working with hp 3000 solution providers to make sure that your familiar environment is available.

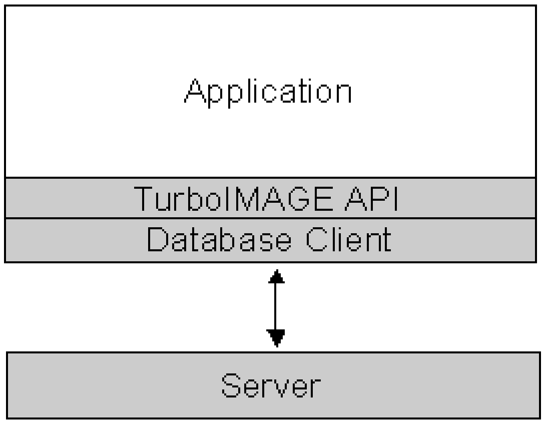
What is not supported with TurboIMAGE compatibility mode in Eloquence:

- DBCONTROL modes which are specific to TurboIMAGE implementation details
- Item level security
- Required changes: Eloquence requires the database name is terminated with a space, semicolon or NULL character

Eloguence's TurboIMAGE compatibility is implemented at different levels:

- The database server implements functionality at the backend

- The database client and utilities provide support for TurboIMAGE functionality
- The TurboIMAGE compatibility API implements source code compatibility



The TurboIMAGE compatibility API is implemented as a library on top of the native Eloquence database API. It does not impose a performance impact. The TurboIMAGE compatibility API provides source code compatibility with existing applications and translates TurboIMAGE intrinsic calls to the Eloquence API. This involves different alignment requirements, modes and status codes, emulating different behavior or converting arguments as required.

The Eloquence image3k library implements the TurboIMAGE intrinsics, and it is this library that the application (or language runtime) is linked against. If you are coding in C or C++, the image3k.h include file provides the function prototypes you will need.

Using Eloquence with AcuCOBOL

Link the Eloquence image3k library to the ACU COBOL runtime (runcbl). Then load the Eloquence image3k library dynamically (using CALL). Eloquence currently uses native byte order. On little endian platforms (Intel IA-32), COMP-5 type must be used instead of COMP. The –D5 compiler option maps all COMP to COMP-5.

Using Eloquence with Micro Focus Cobol

Link the Eloquence image3k library to the application. Eloquence currently uses native byte order. On little endian platforms (Intel IA-32), COMP-5 type must be used instead of COMP. A Micro Focus COBOL compiler directive may be used to map the COMP to the COMP-5 type.

```
MAKESYN "COMP-5" = "COMP"
```

Real World Migration Issues

Data set capacity

Since Eloquence data sets are dynamic and grow as required, data set capacity has a different meaning in Eloquence, because Eloquence has no concept of a data set specific capacity. When you ask for the capacity of a data set, Eloquence returns the highest record number allocated for a data set as the capacity value (DBINFO modes 202 and 205).

Problem: An application may check for "enough room" in a data set (which you cannot do on Eloquence, since there is no concept of a fixed capacity).

Solution: Remove or disable the capacity check.

Workaround: Return a "HUGE" value as the capacity (e.g., trap Eloquence DBINFO 202 and 205 modes and return an application-specific "capacity" value).

Don't lie to the schema

TurboIMAGE does not really care what you put in a character field, but Eloquence relies on the schema type information. Eloquence may need to convert strings to different encoding, or may need to do a byte order conversion, and uses indexes that require type specific ordering.

Solution: Use separate fields for different information, or use the correct item type.

Workaround: Use Eloquence on a single platform, or use Eloquence binary item type 'B'.

Character set encoding

On MPE, designers often use the HP-ROMAN8 character set encoding for text field data. HP-ROMAN8 encoding is typically not available on other platforms. Therefore, Eloquence defaults to the HP-ROMAN8 character set on HP-UX (and MPE) and to the ISO-8859-1 character set on other platforms. Eloquence performs conversion "on the fly".

Byte order

PA-RISC and Itanium (with HP-UX) use big endian byte order. Intel IA-32 and Itanium (Linux and Windows) use little endian byte order. Eloquence performs conversion "on the fly" if necessary.

Parameter alignment

TurboIMAGE requires most arguments to be 16 bit aligned. Eloquence relaxes most alignment restrictions. Eloquence does not require a specific alignment for string arguments

Record numbers

Eloquence uses a different algorithm to assign and reuse record numbers. TurboIMAGE uses a LIFO (last in first out) order to reuse deleted records (unless HWMPUT is active). Eloquence uses a FIFO (first in first out) order to use available record numbers. Eloquence does not support HWMPUT, application has no control over record number usage. HWMPUT is the setting that enables DBPUT to place entries at the high-water mark first, instead of at the delete chain head.

Problem: a DBDELETE / DBPUT sequence on Eloquence likely results in a different record number for the entry.

Solution: Fix the application so that it does not expect record numbers to remain inviolate over a DBDELETE/DBPUT.

Workaround: Use DBUPDATE mode 2 (same as DBUPDATE mode 1 and CIUPDATE) to do a critical item update. This does not result in a new record number.

Identical database names

Problem: TurboIMAGE supports the use of the same database name in different file groups. Eloquence requires a unique database name per server instance.

Solution: Use multiple server instances (eg. test / production environments), or add the group name to the database name (eg. DBNAME.GROUP).

Access to database files

TurboIMAGE databases reside in the file system. Applications could use file system operations to copy databases. But Eloquence databases reside in the volume files and are not accessible separately.

Solution: Copy the whole database environment or use dbstore to extract a single database and dbrestore to restore the database to another server instance. Or use dbexport / dbimport.

Moving Your Databases from TurboIMAGE to Eloquence

Schema files are compatible and no change is required. Eloquence includes MPE tools to export the database content to flat files. Then use FTP to transfer the schema file and the export files to the target system. On the target system run the schema processor, the dbcreate utility and the dbimport utility. Data migration from TurboIMAGE to Eloquence is a straightforward process.

Install the DBEXPORT utility on the HP3000

Eloquence comes with the dbexport and dbimport utilities which can be used to unload a database to one or multiple text files or respectively load a text file into the database. The export files are text files which follow a simple syntax.

An equivalent DBEXPORT utility is available for the HP3000 and can be used to unload your database. It can be downloaded from the Eloquence ftp server

DBEXPORT is used to export the database contents to one or multiple text files. It provides a convenient means to move your database contents to the Eloquence database. The current version should be used with Eloquence A.07.00 and supports all item types.

When running from the POSIX shell the arguments are separated by a space:

```
$ DBEXPORT -p SECRET -v TESTDB
```

When running from the MPE shell (CI) you need to enclose the arguments in quotes:

```
: DBEXPORT "-p SECRET -v TESTDB"
```

DBEXPORT creates a separate file for each dataset. The name is database.set.exp (for example: "SAD.03.exp"). An automatic data set or an empty data set is ignored.

Transferring the files

Transfer your schema file and the export files to the Eloquence system, into the directory where you wish to create the database.   When transferring by ftp, use text mode to transfer the schema file and use binary mode to transfer the export files.

Creating the database:

Run the Eloquence schema processor

```
$ dbschema schemafile
$ schema -T schemafile
```

Option -T selects TurboIMAGE compatibility mode.

```
$ dbcreate database
```

Import the data

Use dbimport to load the database

```
$ dbimport -v database
```

DBIMPORT finds all the database export files and imports them. For example, if the database name is SAD, it looks for SAD.03.exp, SAD.04.exp, etc.

The option -v displays the import progress. On the Windows and Linux platform you should specify the -z roman8 option to indicate the source data uses the HP-ROMAN8 encoding. This makes sure any national characters ("Umlaute") are converted.

That's it – your migration is complete.

Eloquence Product Components

The Eloquence product consists of multiple components. Besides the database, it comes with a complete software development and runtime environment and different options for user interfaces.

The Eloquence product includes the following major components:

- The Eloquence programming language, based on HP Business Basic.
- The Eloquence database management system, based on IMAGE, which we are focusing on in this chapter.
- Different user interface options (Text, Windows, Java and Web based user interfaces).
- Various development tools, including a graphical development environment on the Windows platform.

A quick product overview: There are about 2500+ installations of Eloquence worldwide. It is used by about 60+ VARs / ISVs. Installations cover a wide range of sizes from a single user to a few hundred concurrent users. In the past, Eloquence was typically used to implement vertical and customer specific solutions

Solutions based on Eloquence include ERP, Order Management, Material Management, Financial Accounting / Payroll, Civil Service, Banks and Financial Services. Typically, Eloquence customers are small to medium sized companies that use Eloquence-based solutions for business critical tasks, such as production planning, material management or order management.

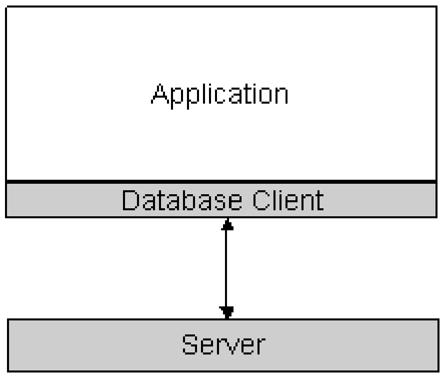
Eloquence Database Architecture

The Eloquence database is almost 100% compatible to TurboIMAGE at the application level. However, the underlying architecture is different. While the Eloquence database is almost 100% compatible with TurboIMAGE at the application level, the underlying concepts are different. This section explains some of the approaches taken by the Eloquence database and explains some of the architectural differences from TurboIMAGE.

Client/Server architecture

The Eloquence database uses a client/server-based architecture. The database server, on behalf of the application, performs all database access. The application is linked with a client library, which provides the database API (Application

Programming Interface) and submits any requests that cannot be processed locally to the server and returns the results and any status codes to the application.



The figure above shows an application with the database-programming interface, which is connected to the database server.

The connection between the client side (application) and the server is made through the network. In case the application is running on the same system as the server, shared memory is used for communication between client and server to reduce the communication overhead.

When a database is opened, the server uploads some structural information about the database to the client, so the client side is able to verify the call arguments and convert them as necessary. It also allows the client side to process most DBINFO calls locally.

Network transparent

Applications running on different machines and operating systems can access a common database. Requests and results are translated transparently.

The Eloquence database is network transparent; applications running on different machines and operating systems can access a common database. The database server and client API make sure that requests and results are translated as necessary to match the specific platform requirements.

For example: PA-RISC systems use a different byte order than Intel IA32 based systems. And MPE systems use a different character set encoding. The Eloquence database client will translate any values transparently as required.

Multiple platforms

Eloquence is available for multiple operating systems and hardware architectures:

- HP-UX
- Linux
- Windows NT/2000/XP
- Database client library on MPE (not yet released)

The current version supports HP-UX on PA-RISC and Linux and Windows NT/2000/XP on the Intel IA32 architecture. Future releases will add support for the IA64 (or IPF) architecture for the HPUX, Linux and Windows operating systems.

Indexing

The Eloquence database comes with integrated indexing capabilities and Eloquence uses indexes rather then hashing with master sets. Indexes are available to applications to implement indexed database access, such as partial key retrieval or ordered retrieval.

The TurboIMAGE compatibility extension allows an application to make use of indexes in a TurboIMAGE compatible manner, since Eloquence implements a commonly used subset of the TPI functionality

With Eloquence B.07.00 both the TPI programming interface as well as TurboIMAGE indexes (used with master sets, "super-chains") are supported.

Locking

Locking with the Eloquence database is fully compatible with TurboIMAGE but provides additional options.

The most visible difference is that locking is optional and the application is free to implement the locking strategy that suits best to its requirements. The Eloquence database neither depends on explicit locking (DBLOCK) nor does it impose restrictions on locking (like requiring a set lock for specific operations).

Instead of verifying if a write operation (such as DBPUT) is covered by a DBLOCK, Eloquence verifies if a conflicting DBLOCK has been granted which covers the same data. In this case a database status is returned. This behavior is fully compatible since in TurboIMAGE all write operations must be covered by a DBLOCK and consequently there can be no concurrent DBLOCK.

Transactions

The Eloquence database relies on transactions to ensure data consistency. Transactions in Eloquence are not specific to a database. All databases modified in a transaction are part of the transaction.

Eloquence does not expose incomplete transactions to concurrent processes. Changes in a transaction are only visible to the initiating process until the transaction is committed. This is called transaction isolation. If concurrent process accesses database content modified in an incomplete transaction the original content is returned.

Transactions are not limited in size: Incomplete transactions that overflow the cache are stored in the log volume and are only limited by the disk space available in the log volume. All database procedures (such as DBPUT) are internally considered a transaction (implicit transactions) which are committed automatically once the call is completed. For example, a DBPUT can consist of 50+ internal operations and may need to be aborted cleanly at any time. We use the transaction mechanism for this.

Nested transactions are supported: If an application makes use of transactions (explicit transactions) any internal transactions become sub-transactions and are merged on commit with the upper transaction level. Transactions can also be nested in applications and then behave the same way.

Database names

The database name is no longer restricted to 6 characters and can include dots and hyphen characters in addition to letters and digits.

Eloquence databases do not reside (logically) in the file system but are managed by a server process. A database name does not specify a file location, but addresses a specific server instance. Eloquence uses an extended format for database names. The database name can specify the network address of the database server as an option. To access a database on a remote system, simply add the server address to the database name. For example:

```
[[hostname][:service]/]database
```

`Hostname` specifies database server system.

`Service` specifies database server instance.

The hostname or IP address qualifies the system on which the database server resides. The default is the local system (localhost or IP address 127.0.0.1).

The service name or port number is used by the server process on that machine. The default is defined by the tcp service named eloqdb (which defaults to port 8102 during installation).

If the hostname or service is specified, it is separated with a slash from the database name.

The example below specifies the database SAMPLEDB on the local system, using the default database server instance in different forms:

```
localhost:eloqdb/SAMPLEDB
:eloqdb/SAMPLEDB
SAMPLEDB
```

Use the EQ_DBSERVER environment variable to specify the default server instance.   For example:

```
EQ_DBSERVER=invent9k.external.hp.com:8102
```

This specifies that the specified server instance manages the database. The default is used unless you provide more specific information.

Database security

The database server maintains a list of users. Database access privileges are assigned to groups, which are similar to IMAGE user classes. Then a user is made a member of one or more groups.

For each database, groups define the access rights. This is the equivalent of the TurboIMAGE user classes, which can be defined in the PASSWORDS section in the schema file. The schema processor creates two default groups "public" and "dba".

The group "dba" provides administrative access to the server and all databases, but does not allow read access to the data (it does have the right to erase a data set).

The group "public" public allows read/write access to the data, but is not authorized to perform administrative tasks (it has the access rights defined for user class 0).

Other groups are created and provide the access rights defined in the schema file. Users can be associated with up to eight groups and get the summary rights associated with the groups. Unless associated with a group, no access to the database is possible. By default, the schema processor associates the user "dba" with the group "dba" and the user "public" with the group "public" and all other groups defined in the schema file. This can be changed with the dbutil utility.

In order to connect to the Eloquence database server a user name and password is required. The new DBLOGON procedure may be used to specify user and password. A file can be specified as the user name or password. A default user is used if no specific user is specified.

The EQ_DBUSER and EQ_DBPASSWORD environment variables may be used to specify the default user or the password. For example:

```
EQ_DBUSER=file:/home/mike/dblogon
EQ_DBUSER=mike
EQ_DBPASSWORD=file:/home/mike/passwd
```

Database environment

A database environment consists of a configuration file, one or more data volumes, and a transaction log volume. Multiple database environments can coexist on the same machine, each managed by a separate server process.

Volume files

The Eloquence database does not reside in the file system but uses volume files, which reside in the file system as a container for the data and structural information. Volume files are a storage container managed by the database server. All volumes combined define the disk space accessible to the database server. The maximum size of a single volume file is 128 GB. As of Eloquence B.07.00 a single volume file is limited to 2 GB on the HP-UX and Linux platform (we currently don't use 64 bit file operations on HP-UX and Linux to maintain compatibility with older operating system versions). On the Windows platform we support 64 bit file operations so the 2 GB limit for a single volume does not apply.

A maximum of 255 volume files is supported by a single server instance, which defines an upper limit of about 500 GB for the current version and up to 32 TB once the 2 GB limit has been removed.

Volume files are specific to the system architecture (byte order).

Server catalog

Eloquence does not use a ROOT file. Database structural information is maintained in the database environment, called the server catalog. The catalog is an internal database maintained by the server process. The server catalog is created and initialized with the dbvolcreate utility (which is used to create the primary data volume) and maintained with database utilities, such as the schema processor or the dbutil utility. The dbdumpcat utility allows reading the server catalog.

Database limits – the Eloquence B.07.00 Image limits:

- 2048 data items
- 500 data sets
- 64 / 16 paths
- Entry length 5120 bytes

The Eloquence B.07.00 database either matches or exceeds the TurboIMAGE schema limits. Most TurboIMAGE limits do not apply to Eloquence because the underlying architecture is different, and Eloquence allocates most resources dynamically.

Although the Eloquence database may exceed the TurboIMAGE limits in some cases, you should be careful before making use of that fact. Applications may need to be reviewed, if they support the increased limits. As the IMAGE intrinsics do not specify buffer sizes (this is especially important with DBINFO as it returns a variable sized result) this may result in application crashes due to a buffer overflow. With the Eloquence TurboIMAGE compatibility API, DBINFO mode 406 returns information about database schema size.

Disk space within the volume files is allocated dynamically. A single data set can span multiple volumes and is therefore not limited to the size of a single volume file. Transactions can grow dynamically and are only limited by available disk space in the log volume(s).

Scalability

Database / data set size is limited by the disk space allocated to the database environment. – Current limit is ~500 GB. Hard limit is ~32 TB

Some Eloquence limits apply to a database environment rather then a specific database as all databases managed by a database environment share common resources. The size of a database or data set is only limited by the disk space available in the volume files.

The maximum disk space which can be allocated to a database environment is currently limited to about 500 GB. This is imposed by the current limitation of 2 GB per volume file.

The maximum amount of disk space, which can be managed by a database environment, is about 32 TB, which should be sufficient for a while. The number of concurrent users per database environment is limited to 1000. However we do not recommend using more than 500 concurrent users per database environment with the B.07.00 release.

Database Utilities

The database utilities can be categorized into three groups:

- Offline utilities which can only be used when the database server process is not active
- Administrative utilities which control server operation or request server status information
- Database utilities that access the database server

Offline utilities

- dbvolcreate / dbvolextend / dbvolchange / dblogreset - database volume management
- dbvoldump - display volume properties
- dbfsck - volume consistency check and simple repair tool
- dbcfix – database consistency check and repair tool
- dbrecover - forward recovery

Database utilities in the offline category operate on the volume files and can not be used while the database server is active

Administrative utilities

- dbctl - server management utility
- HTTP status monitor

Administrative utilities connect to the database server to control server operation or request server status information.

The Online Database utilities

- schema - Schema processor
- dbcreate / dberase / dbpurge - create / erase / purge database
- dbtables - database cross reference
- prschema - re-create schema from database
- dbdumpcat - catalog information utility
- dbexport / dbimport - export/import data base content to/from text file
- dbinfo - information on database tables
- dbutil - structural maintenance and database security management
- QUERY utility (different from QUERY/3000)

The online utilities make use of the database server to access a database.

Installation and Configuration of the Eloquence database

Evaluation license

By default the "Personal Edition" license key is installed. A temporary license key can be created during installation. A temporary license key can be requested from the Eloquence web site by filling out the temporary key request form

```
http://www.hp-eloquence.com/license/demo.html
```

Create eloqdb user/group

Create a user name and a group name e.g. eloqdb to be used as the owner/group of the database files. On Windows the system account is used by default. Create the user and group eloqdb. This user and group are used by the database server and own the database volume files. Administration of volume files should be performed either from root or this user. If this user is not used for administration it should be disabled.

Configure kernel parameters

Depending on the configuration, Eloquence has specific requirements for the kernel configuration. On Unix and Linux, Eloquence can use shared memory for communication, which requires additional kernel resources.

Some of the HP-UX kernel parameters that need to be configured are semaphore-related parameters, shared memory-related parameters, and process data size.

To configure kernel parameters, start SAM and select Kernel Configuration -> Configurable Parameters, change the kernel parameters as necessary and build a new kernel. Changing kernel parameters requires a system reboot.

Kernel: Semaphore configuration (EnableIPC enabled)

If you enable shared memory communication in the database server configuration file (EnableIPC), the kernel SYSV semaphore-related kernel parameters would likely need revision. For each local connection, a separate semaphore is needed. The semmni, semmap, semmns, semmnu and semume kernel parameters must be adapted accordingly.

If shared memory communication is not enabled, the semmns kernel parameter needs only be changed if a large number of I/O threads are used. The default number of 4 I/O threads is likely covered by the default kernel configuration.

In the following sections, the variable x specifies the number of concurrent connections (Threads configuration item) and y the number of i/o threads (IOThreads configuration item).

Semaphore configuration for EnableIPC=1

- Set the 'semmni' to at least x+20
- Set the 'semmap' to 'semmni' + 2
- Set the 'semmns' to at least x+y+20
- Set the 'semmnu' to at least x+20
- Set the 'semume' to at least x+20

If shared memory communication is enabled in the database server configuration file, the kernel SYSV shared memory-related kernel parameters might need revision. When EnableIPC=1 for each local connection, a separate shared memory segment is needed.

The shmmni and shmseg kernel parameters must be adapted accordingly. This is not required with EnableIPC=2. If shared memory communication is not enabled (or EnableIPC=2), the shared memory related kernel parameters do not need to be changed . In any case, the maxdsiz kernel parameter should be changed to allow at least 128 MB of process data size.

In the following sections the variable x specifies the number of concurrent connections (Threads configuration item).

Shared memory configuration

- Set the 'shmmni' to at least x+20
- Set the 'shmseg' to at least x+20

Data size

- Set the 'maxdsiz' to at least 0x08000000 (128MB)

Setup database environment

The database environment (which is a server instance) consists of

- Server configuration file (eloqdb.cfg)
- Primary data volume
- Transaction log volume(s)
- Additional data volume(s) as required

The Server configuration file

The default server configuration file is /etc/opt/eloquence6/eloqdb6.cfg

This file defines server properties, including configuration, scaling and tuning parameters, and volume files. For example, here is a simple server configuration:

```
[Server]
Service = eloqdb
ServiceHTTP = 8103
UID = eloqdb
GID = eloqdb
EnableIPC = 1
SyncMode = 0
[Config]
Threads = 100
IOThreads = 4
BufferCache = 64
CheckPtSize = 50
```

The Service configuration item specifies the service name (as defined in the services file) or the port number, which is used by the database server. The default value is eloqdb. The port number must be different for each server instance and may not already be used.

The ServiceHttp configuration item specifies the service name (as defined in the services file) or the port number used by the embedded web server. By default the HTTP status is disabled.

The Uid and Gid configuration items specify the user and group, which are used to run the server and own the volume files. This setting is ignored on Windows.

The EnableIPC configuration item activates the use of shared memory for communication. This significantly reduces system overhead but requires additional kernel configuration, as described earlier.

Shared memory: EnableIPC

- EnableIPC=0 (default) disables use of shared memory communication
- EnableIPC=1 enables use of shared memory on HP-UX and Linux
- EnableIPC=2 enables use of a single shared memory segment for HP-UX (recommended)

The SyncMode configuration item specifies if the server should write committed transactions to disk immediately. If disabled, a more efficient write strategy is used. SyncMode is enabled by default, which makes the database resistant in case of a system crash, however it causes additional disk load. A more efficient option is to enable forward logging.

Sync/Async mode

- SyncMode=1 (default) pushes all committed transactions to disk immediately and waits for completion
- SyncMode=0 (recommended) writes changes to disk asynchronously and does not wait for completion

The [Config] section defines the scaling and tuning parameter settings for the database server. This example configuration supports 100 concurrent users using a 64MB cache. You should at least have 128 MB memory on your system.

The Threads configuration item specifies the maximum number of connections. Each connection is associated with a thread running in the server context. Only a single connection is used by an application to access any number of databases managed by a single server (connection pooling). So, Threads defines the maximum number of concurrent connections for this server instance

The IOThreads configuration item defines the maximum number of concurrent I/O operations. The default is 4. This number may need to be increased depending on database size or number of concurrent users. The maximum usable setting also depends on the I/O capabilities of the server.

The BufferCache configuration item specifies the cache size in megabytes. The cache is used to reduce the number of disc accesses. The default (and min.) cache size is 5 MB.

The CheckPtSize configuration item specifies the amount of disk space used for the transaction journal. The default size is 10 MB. When this space is exhausted the server performs an internal checkpoint operation.

Create the Volume Files

```
dbvolcreate /var/opt/eloquence6/data01.vol
dbvolextend -t log /var/opt/eloquence6/log.vol
dbvolextend -t data /var/opt/eloquence6/data02.vol
```

The HP Eloquence database does not reside in the file system but uses volume files that reside in the file system as a container for the data and structural information. Please note that you need both a data volume and a log volume in order to start the data base server.

These commands should either be used by the system administrator (root) or executed by the eloqdb user.

The dbvolcreate utility is used to create a new server instance and creates the primary data volume, which contains the system catalog. The dbvolextend utility extends a database environment by another volume. This could either be a transaction log volume or another data volume. The volume type must be specified. The specified volume files are created and then added to the server configuration file. A single volume file is currently limited to 2 GB on HP-UX and Linux and additional data volumes may be required.

Configure Server Startup

Configure automatic startup of the Eloquence database. The startup configuration file defines which Eloquence services are started

- HP-UX: /etc/rc.config.d/eloquence6
- Linux: /etc/sysconfig/eloquence6

The Eloquence eloqsd service is often not needed and should not be started. Set the START_ELOQSD variable to 0 to disable the automatic start of the eloqsd service.

Starting the database server

On HP-UX:

```
/sbin/init.d/eloq6 start|stop|status|restart [instance …]
```

On Linux:

```
/etc/init.d/eloq6 start|stop|status|restart [instance …]
```

Other server operations:

- start – start server processes
- stop – stop server processes
- status – check status of server processes
- restart – restart server process

Troubleshooting

The Eloquence database writes diagnostic messages to the syslog:

- P-UX: /var/adm/syslog/syslog.log
- Linux: /var/log/messages
- Windows: application event log

Linux installation

Eloquence uses the RPM package manager. RedHat Linux 7.x to 9 and SuSE Linux 7.x to 8.x have been certified. Other Linux distributions may be used but additional manual configuration may be required.

For installation or update, execute the command below:

```
$ rpm -U Eloquence-B0700.rh8.i386.rpm
```

Note: the temporary license option is not available during installation on Linux.

Windows installation

Eloquence uses the Microsoft Installer engine that was introduced with Windows 2000. The MS installer engine is installed or updated before proceeding with the Eloquence installation (may require a reboot). The Microsoft Installer engine provides a common installation procedure across applications. The installer engine makes use of an .msi database, which specifies the product; the product files are contained in .cab files. Note: Different setup programs are used for Windows 2000/ XP/2003, Windows NT and Windows 9x.

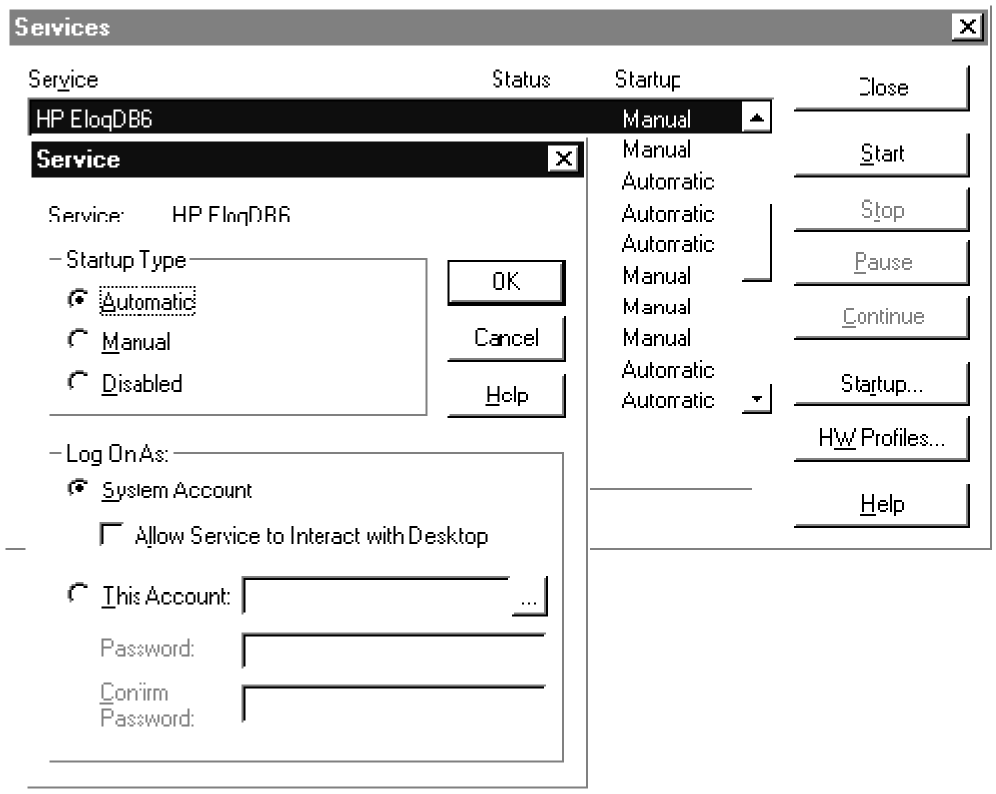
The setup program differs for the download and CD-ROM media version. The download version cannot be installed from a read-only media, as it requires write access to the local directory. The CD-ROM version is already unpacked and does not require a writeable directory.

Automatic Start

During installation, the Eloquence database is registered as a service. To configure automatic server startup, please open the service control panel (eloqdb6 service) and configure the service to start automatically.

After creating the database environment, start the service manually. In case the service startup fails, please check the windows application event log for any messages. Note: The eloqsd service is often not needed and should not be started.

See a sample Windows service configuration screen below.



Service: HP EloqDB6

Startup Type: [✓ Automatic] Manual Disabled

Log On As: [✓ System Account]

Allow Service to Interact with Desktop

This Account:

Password:

Confirm Password:

Database Backup

There are two supported backup strategies: off-line and on-line, with a related option: Forward logging.

Off-line backup has 3 steps:

1. Shutdown the eloqdb6 server process
2. Backup all volume files
3. Re-start the server process

And On-line backup has 4 steps:

1. Enable on-line backup mode
2. Backup the data volume file(s)
3. Backup of the log volume is optional
4. Disable on-line backup mode

On-line backup

In on-line backup mode, the Eloquence database makes sure the data volume(s) are consistent and not changed, even if the database is in use. All database modifications are saved temporarily in the transaction log volume. This makes sure any backup software can be used to create a consistent backup without interrupting ongoing work and allows for easy and straightforward integration into backup procedures. Since the database environment is backed up, the backup will cover all databases, which are managed in that environment.

When the backup mode is finished, Eloquence copies the pending changes from the transaction log volume to the data volume(s).

The dbctl utility is used to enable on-line backup mode. Here is an example backup script:

```
$ dbctl -u file:/root/credentials backup start
$ tar -cf /dev/rmt/0m /database
$ dbctl -u file:/root/credentials backup stop
```

The database is put in on-line backup mode by sending the backup start request with the dbctl utility. Then you use the tar utility (or your favorite backup tool) to create a backup of the database environment and finally, you finish the backup by sending the backup stop command.

Forward logging

Use forward logging to record all modifications since a previous backup. It is fast and involves only minimal processing.

The server provides the option to log all changes since a previous (on-line or off-line) backup, which can then be applied with the dbrecover utility. Forward recovery is integrated with the on-line backup and Eloquence is able to discover automatically, if and which part of the log file applies to a recovered backup. As an option, the database server can manage forward recovery logging automatically. Removal of old log files can easily be integrated into the backup procedure.

Forward logging is enabled in the server configuration

```
[ForwardLog]
FwLog = /path/to/fwlog-%N.log
```

More information

For more details on Eloquence, visit the Eloquence web site at

```
http://www.hp-eloquence.com
```

The TurboIMAGE DataTypes

By Bob Green

*About the author: Bob Green has worked with IMAGE since the start. He was the spark plug for famous **IMAGE/3000 Handbook**. That book is out of print, but if you find a copy, read the chapter entitled "Guided Tour to a DBPUT", which is Bob's favorite.*

A TurboIMAGE data item definition has three parts:

```
sub-item-count   datatype   sub-item-length
```

Sub-item-count is an integer from 1 to 255 that denotes the number of sub-items within an item. If omitted, the sub-item-count equals one by default. A data item whose sub-item count is 1 is a simple item. If the sub-item count is greater than 1, it is a compound item. We might also call it an array or a table of values.

Sub-item-length is an integer from 1 to 255. It is the number of halfwords, bytes, or nibbles (depending on the type designator) in a sub-item. If omitted, it is equal to 1 by default.

Datatype is a letter code signifying the format of data stored in this field.

The type designators E, I, J, K, P, R, U, X, and Z

Type	Description
E	ieee floating point. sub-item length is in halfwords
I	signed integer, sub-item length is in halfwords
J	signed integer, but conforms to COBOL standards (i.e. s9999 has max value 9999). sub-item length is in halfwords
K	unsigned integer, no negative value. 1 halfword = 0-65K, 2 halfwords= 0-2 Billion, sub-item length is in halfwords
P	packed decimal, sub-item length is in nibbles, 2 to 28, with one digit used for the sign (note: TurboIMAGE will let you create a P48 or even larger, but COBOL will not process it)
R	classic HP 3000 floating point, old, 2 halfwords or 4 halfwords, you should probably not be using this type!
U	uppercase ASCII chars, sub-item length is in bytes
X	any ASCII characters, sub-item length is in bytes
Z	zoned decimal number. sub-item length is in bytes

The size of the entire data item must be a multiple of halfwords (16 bits). Therefore, P types normally come in multiples of 4 and U/X/Z types come in multiples of 2. However, you actually multiply the sub-item-count by the sub-item-length to get the data item size, so 2X7 is valid. (Note: IMAGE does not enforce the upper-case attribute in U fields; that is up to the user programs.)

TurboIMAGE Compatibility with Languages

Although TurboIMAGE does not place any restrictions on the reasonableness of item datatypes (i.e., you can define J25 if you wish) and does not validate values before inserting them into the database, most TurboIMAGE databases use only the data types that can be processed by the programming languages on the HP 3000.

Here are the data types that are supported in COBOL and Fortran:

Type	Description
Xn	Character, n bytes, define as Character in FORTRAN, X(n) in COBOL
Un	Uppercase Character, n bytes, define as Character in Fortran, A(n) in COBOL
E2	Floating-Point, 4 bytes, define as Real in Fortran, not supported in HP COBOL
E4	Floating-point, 8 bytes, define as Double Precision in Fortran, not supported in HP COBOL
I1/J1	Integer, 2 bytes, define as Integer*2 in Fortran, S9 to S9(4) Comp in COBOL
I2/J2	Integer, 4 bytes, define as Integer*4 in Fortran, S9(5) to s9(9) Comp in COBOL. Suprtool/Powerhouse do I3/J3 also.
I4/J4	Integer, 8 bytes, define as S9(10) to S9(18) Comp in COBOL, not supported in Fortran.
K1	Logical, 2 bytes, define as Logical in Fortran, not supported in COBOL. Suprtool and Powerhouse support K2, K3, K4.
Zn	Zoned-Decimal, n bytes, s(n) Display in COBOL, overpunched
P4	Packed-Decimal, 2 bytes, s9(3) Comp-3 in COBOL, not supported in Fortran
P8	Packed-Decimal, 4 bytes, s9(7) Comp-3 in COBOL, not supported in Fortran
Pn	Packed-Decimal, n/2 bytes, s9(n-1) Comp-3 in COBOL, not supported in Fortran. Maximum N in HP COBOL is 19 (18 digits plus a sign)
Zn	Numeric Display, n bytes, s9(n) Display in COBOL, with sign "overpunched" in the units position (unless you specify SIGN IS SEPARATE, then there are only n-1 digits in the value)

Quick Start Oracle for the TurboIMAGE Developer

By Terry O'Brien, DISC

About the author: Terry O'Brien is a part-time software designer and part-time cattleman.Executive Vice President of Dynamic Information Systems, makers of Omnidex.President of Teamsoft, e3000 migration consultants to relational data bases. Terry O'Brien has enjoyed working with the HP 3000 and TurboIMAGE for almost 25 years and most recently has been working with Oracle, DB2 and SQLServer on Unix, Linux and Windows platforms.

For many developers and administrators that have used MPE/iX and TurboIMAGE, there was little joy in HP's announcement to end development and support of the OS and database package that had been in use for years. Many developers were content with their knowledge and capabilities of TurboIMAGE to develop decent, although limited applications. This is not to say that TurboIMAGE was not an excellent database system when it was originally conceived over twenty years ago.

However, even in its initial design, several key application development components such as a DATE data type and scaling of numeric values were not supported at the database level so countless amounts of application code has had to be developed again and again to compensate. And even more important, HP, either due to economic or political reasons, or simply because of a lack of internal capabilities, did not continue the evolution of TurboIMAGE. For all practical purposes, there has been little to no significant enhancements to TurboIMAGE during the past ten years during which there has been a proliferation of enhancements to the three main database contenders today: Oracle, Microsoft's Sql Server, and IBM's DB2.

And although homesteading and other migration options exist, this guide is for those shops that want to make a transition to the Oracle database with the end result that Oracle will enable the development of better and more advance applications than was ever possible with TurboIMAGE. This guide will provide an overview of the Oracle database using terms familiar to the TurboIMAGE developer with an objective of providing a roadmap of steps for learning the Oracle database.

Step 1 – Breaking the love affair with TurboIMAGE.

Before looking at some of the specific architectural and terminology differences between TurboIMAGE and Oracle, a comparison of the features between the two databases may help to create the proper mind-set of having to learn the whole new, and certainly more complex Oracle database system. It has been interesting to observe the TurboIMAGE experts expound on how much better TurboIMAGE is over the other database systems over the past several years. Unfortunately, although many of these people are indeed experts on TurboIMAGE, they had little to no knowledge of Oracle and for the most part, their assertions were simply wrong or based on earlier versions of Oracle.

TurboIMAGE today, has only a fraction of the capabilities, scalability, reliability, and maintainability of Oracle version 9. Certainly TurboIMAGE is simpler than Oracle but that simplicity comes at a price at the application development and maintenance level. Oracle is more complex because it does more for you at the database level to enable the applications to be simpler and more consistent. Further, TurboIMAGE, although very good at many OLTP applications, is simply horrible at OLAP and decision analysis applications due to the extremely limited indexing options. Although the third party tool, Omnidex alleviated some of TurboIMAGE's shortcomings in this regard, TurboIMAGE still falls short of handling large numbers of transactions seen in typical Data Warehouse and Data Mart applications.

Because of the number of items to explore in this section, a modified outline approach will be used to show the feature and a quick note on the use of the feature.

Similar Features between TurboIMAGE and Oracle:

TurboIMAGE and Oracle do share some similar capabilities and the following list highlights the major ones.

1. Transaction Management – Both TurboIMAGE and Oracle have the ability to control multiple data manipulation operations into a single transaction to facilitate transactional integrity within the database.
2. Referential integrity – In TurboIMAGE a Master record cannot be deleted if it has Detail records associated or Detail records cannot be added if there is not a Master record. Oracle has the same capability.
3. Callable API – TurboIMAGE and Oracle both support a callable API at the application level.
4. Backup module – TurboIMAGE has DBStore and Oracle has RMAN (Recovery Manager)
5. Logging module
6. Database structural access – TurboIMAGE has an API (DBINFO) interface to structural information where Oracle uses SQL Select statements against internal system tables.
7. User Security with restrictions to tables and columns although TurboIMAGE only supports user classes while Oracle supports users and classes of users called groups.
8. Generic Query/Update Tool (Query vs SQL*Plus)
9. Distributed Databases – Oracle can easily connect to remote databases, replicate databases across multiple servers, distribute transactions using a two phase commit (One phase to see if all servers are ready to commit and

the second phase to do the commit). TurboIMAGE supports remote databases but it requires Quest and other third parties facilitated hot backups and replicated databases.

Oracle features over TurboIMAGE

Because there are a number of Oracle features over TurboIMAGE, the features are broken down into General, Availability, Scalability, Performance, Security, Management, and Application Development sections.

General

1. Multiple Operating System Support (UNIX, Linux, Windows) – Oracle used to run on over 60 operating systems but as of Oracle 9, is down to the major flavors of Unix (HP UX & Compaq Tru64 Unix, IBM AIX, Sun Solaris, Linux, and Microsoft Windows NT/2000/XP/2003, OpenVMS, IBM OS/390, and IBM's Dynix/ptx). By using Oracle's internal file and OS tools, complete application portability can be obtained between UNIX, Linux and Windows. TurboIMAGE has always been constrained by only running on MPEiX and now with MPEiX end of support announcement, migration and porting costs will likely significantly change the total cost of ownership for TurboIMAGE. And although a TurboIMAGE clone, Eloquence exists on other platforms, it is not TurboIMAGE but a whole new database that looks like TurboIMAGE. Oracle looks and acts the same on all platforms.
2. PC and Laptop Development Environment (Windows and Linux) – TurboIMAGE development has always been more limited because it only ran on MPEiX and thus it has never been appropriate to give each developer their own and portable test system. With Oracle, it is very easy to allow each developer a copy of Oracle running on a Windows or Linux based PC. And with laptop capabilities the development of Oracle applications can be done most anywhere. Oracle provides low to no cost developer licenses of Oracle to facilitate development.
3. Integrated SQL interface – Oracle was designed from the ground up to support SQL. Although TurboIMAGE has an SQL interface via the Allbase TurboIMAGE connect, it is a limited implementation of SQL and not well integrated.
4. Triggers – Triggers are one of the most significant database concepts implemented during the last ten years and are heavily used within Oracle applications to facilitate development, handle special update, security or logging needs and for moving application down to the database and simplifying the applications. They are a great feature and it was unfortunate that TurboIMAGE was never able to add this feature.
5. Network Services for client/services – Oracle contains a built in network service for distributed databases, client/server development with ODBC and JDBC interfaces, and web applications. Although third party solutions are

available for TurboIMAGE, Oracle Net Services have continued to evolve and are functional and very high performance.

6. Built in Java Virtual Machine – Oracle is wholly endorsing the Java development platform and includes a Java Virtual Machine within the database for processing Java based trigger and store procedure code.

Availability

Oracle is used in many 24/7 environments and has developed additional modules over the years to facilitate extremely high availability rates..

1. Real Application Clusters – Oracles Real Application Clusters are the ability to have one or more processors accessing the same Oracle database. So any downtime on a given processor does not affect the other processors.
2. Advanced Replication – Although somewhat difficult to set up and administer, Oracle's Advanced Replication facilitates distributing transactions to multiple databases across separate servers.
3. Standby Database – In Oracle, it is easy to set up and manage a Standby database on a backup server that uses log file entries to maintain a standby backup copy of the database if the primary server goes down. Oracle calls the feature Data Guard.
4. Failsafe – Oracle Failsafe is a Windows only uptime and availability module that uses Microsoft's Cluster Server software to provide identical hardware and software setups so that if any one node fails, the other processing node can take over.
5. Block Media Recover – Should a particular Oracle block of data become corrupt, Oracles Recovery Manager allows the recovery of only the data blocks required while the rest of the table is available to other users.

Scalability

Oracle can continuously been improved with each major release to handle larger numbers of users, database sizes, tables, columns per row and other limiting features.

1. Large number of users – Oracle claims to be able to handle 10,000 simultaneous users. With Oracle's data replication and other network features, Oracle could push this limit across multiple servers to way beyond this number.
2. Capable of storing Terabytes (even Petrabytes) of storage – Although TurboIMAGE supports Jumbo data sets, Oracle can store and manage a much larger amount of data.
3. Built in data compression – Because so much data is character based, Oracle supports the VARCHAR2 data type that only stores the data. TurboIMAGE

always uses the defined amount of storage for a column even if the column contains fewer or even no characters.

4. Partitioned Tables and Indexes – Oracle has a sophisticated way to split a logical single table (set) into multiple underlying physical tables. For large amounts of data, this is an absolute requirement. Oracle also have numerous options to partition tables as well as split and merge tables.
5. Materialized Views – Oracle supports the concept of summarized views of data that can automatically be updated when the underlying data is changed. This facilitates OLAP and decision support based systems.
6. Data warehouse features – Oracle contains numerous data warehouse features such as Dimension Tables and Star Schema Optimization to support the development and maintenance of data warehouse applications.
7. Parallel processor support – Oracle has great support for multi-processor systems and can split queries and other processing across multiple processors. Further Real Application Clusters, allow scaling by allowing additional servers against the same database.
8. 64 bit support – Oracle supports the latest 64 bit operating systems.

Performance

Ten to fifteen years ago, the argument that TurboIMAGE was a higher performing database over the relational offerings was probably true. Today, however, only the simplest TurboIMAGE databases could outperform an Oracle database. This is due to Oracle's use of multiple processes, shared memory caches, and extensive indexing options. TurboIMAGE's indexing options have always been severely limited. DISC's Omnidex package helped by providing B-Tree, Concatenated Indexes, and Keyword indexes but lacked the level of true integration that Oracle can provide. TurboIMAGE did implement B-Tree indexes late in the product's life cycle, however the implementation was severely limited by not allowing multiple indexes on Master tables.

1. Cost-based Optimizer – Oracle has continued to advance its Cost-Based Optimizer over their original Rules-base optimizer since version 7. The optimizer as of Oracle version 9 handles a wide variety of SQL in an extremely efficient manner.
2. Concatenated Keys – Oracle supports indexes on one of more fields and is very intelligent in searching for parts of the index to facilitate retrieval performance.
3. Rich Index Module – Oracle has a much richer indexing environment over TurboIMAGE. There is extensive use of B-Tree indexes which can be installed on more than one column in any table (unlike a single key for a TurboIMAGE Master), Indexed Organized tables for when a table and the index are one in the same, Clustered Tables for keeping parent and child rows physically together on disk for high speed access, Bitmapped indexes

for indexing columns with few distinct types (low cardinality), and reverse indexes. Oracle also supports function based indexes that allow the specification of functions to manipulate the data before the data is indexed.

4. Keyword Indexes – Oracle supports a module for keyword indexing of structured data in its CATSYS module which is part of Oracle Text (formally known as Context and Intermedia) which can also keyword index external documents such as Word and PDF files. However the Oracle CATSYS module is not as fast on builds as Omnidex nor does it support drill-down counts. However, both Omnidex and a third party Oracle cartridge called Data Aware Indexes (DAX) can provide Omnidex style drill-downs within the Oracle database.

Security

Oracle implements a very rich security environment to handle the most demanding and sensitive security issues.

1. Rich Security model– Oracle contains a rich security model with users, groups, privileges that are easy to manage using SQL syntax or Oracle's GUI or third party GUI tools.
2. LDAP Security Module – Oracle can keep user security information in an LDAP directory so that a single security environment can be maintained across multiple databases.
3. Fine grained security – also known as Virtual Private Databases, Oracle provides a mechanism to manage additional security controls so that within the same table, one user could see certain rows while another user would see others. This moves security completely into the database instead of at the application level.
4. Encryption of data - Oracle allows the encryption of data on a column by column basis using a built in package (DBMS_OBFUSCATION)
5. Fine-Grained Auditing – Oracle can log specific SQL Select statements based on various criteria so that any queries against sensitive data can be monitored.

Management

Oracle was certainly more difficult to manage than TurboIMAGE in its previous versions. Most Oracle shops had to employ a full time DBA to handle routine database administration. In response to customer requests and the ease of database administration that Microsoft has developed around SQL Server, Oracle is now much easier to administer, including Oracle Managed Data files and UNDO logs which make Oracle more attractive to smaller shops. In addition to the features listed below, expect Oracle to continue to increase the ease of management and administration of an Oracle database.

1. GUI Administration Tool – although Oracle's entire administration can be handled via SQL statements to facilitate scripting, Oracle also provides a Java based GUI tool that can manage multiple Oracle databases on multiple servers from a single Windows based PC. Further, there are some excellent 3rd party tools such as Quest's TOAD (yes, it's a stupid name but it was originally shareware) to facilitate managing Oracle and developing applications.
2. Generic data load utility – The Oracle Loader utility can handle simple to complex data load functions.
3. Built in ETL functions – Oracle can link to external tables (non Oracle tables such as fixed length OS files) and then use SQL with all its available functions to read and manipulate the data. Oracle also has Multi-table insert statements and other ETL functions to facilitate data loading from other systems or databases.
4. Built in dynamic structural changes (new tables, columns, etc.) – Oracle can add tables, modify tables and column information dynamically. Although third party tools such as Adager gave TurboIMAGE some of this functionality, most of the structural changes require exclusive access to the database. Oracle can perform these functions while other operations are being performed.
5. Rich static and Dynamic Data Dictionary – Oracle contains an extensive collection of static structural information and dynamic run time information that can be queried using normal SQL techniques to diagnose and fix performance issues. If you need something, its most likely already there.

Application Development

Although several critical Oracle features have been explained that show how much richer Oracle is over TurboIMAGE, the real advantage is in the area of the development tools and features that facilitate application development. Here, there is simply no contest between Oracle and TurboIMAGE.

1. SQL access as the native interface – Oracle is built around SQL and it has a particularly rich implementation of SQL. However since Oracle implemented many of its features before the standards were implemented, Oracle is not always standards compliance with their SQL language. This is generally not a problem if there are no plans to create an application that works on multiple databases.
2. Rich data manipulation functions – Oracle's SQL language contains hundreds of functions for manipulating text, date, and time data which moves much of the application work to the database.
3. Built in 4th GL language – Oracle contains a full blown 4th generation language procedural language (PL/SQL) for building database stored procedures and trigger modules that fire during table updates, deletes, or

inserts. This package is as powerful as Cognos' QTP module and Speedware's Reactor module. Much of an application's logic can be coded in this language leaving the actual application to focus on the user interface.

4. Built in generalized report writer – Oracle has extended their implementation of SQL within their SQLPLus utility by adding a DEFINE command for declaring variables, a COMPUTE command to add totals and subtotals, a COLUMN command to support formatting and line breaks and column headings. The additional reporting capability within SQLPlus has all the reporting capability of the Cognos' QUIZ package.
5. Rich Data Type support – Oracle supports Date and Time data types and Numeric data types that have scaling and precision. TurboIMAGE supports only primitive binary and character data types so many applications developed their own approaches. Two of the most well known, the ASK MANMAN package (now owned by Computer Associates) and the Cognos Powerhouse software (Quiz, Quick, QTP) developed their own date implementations, which has caused untold additional application development. Note at least implementing a DATE data type was one of the worst oversights within TurboIMAGE. Oracle also supports Objects and Abstract data types for creating structures of columns such as an address containing street, city, region, and postal code.
6. Other data features – Oracle supports Sequences to support the automated generation of sequential keys, column defaults for columns which are not set during an INSERT statement. Oracle also supports Views for providing a subset of a table to an application as well as Synonyms for creating shorter or more meaningful names.
7. Views – Oracle has rich support for Views, including the ability to update Views sometimes using the INSTEAD OF Trigger. Views are great for providing limited amounts of columns and rows to different users.
8. Built in modules (DBMS packages) – Oracle contains several built in modules that support JOB processing, email processing and other key functions.
9. Rich interfaces – Oracle support ODBC, JDBC as well as Perl, PHP, Java, VB.NET and other .NET languages providing an extremely rich application development environment.
10. Built in XML and Web support – Oracle has added extensive XML support as well as support for hosting Web applications. Oracle has repacked the Apache Open Source web server to offer a complete Web deployment solution.
11. Large Object support and External File indexing – Oracle can store binary large objects (BLOBs), character large objects (CLOBs) within the database to and index external files such as Word and PDF files to create full document management systems.

12. Locking and Isolation – Oracle supports a sophisticated locking strategy that removes some of the complexity of application locking. Additionally Oracle provides several isolation options to allow concurrent reads while others are updating the database.
13. National Language and Unicode support – Oracle has extensive National Language support including the NCHAR data type for storing double byte data.
14. Few limitations in number database objects – Oracle has few limitations in terms of database design. Many of TurboIMAGE limitations have slowly been increased but the initial sever limitations such as the original 99 sets and a 2K media record caused all sorts of application work-arounds. Oracle has continuously increase their various limits so there is unlikely any limit in version 9 that would impact application development.

In additional to all the features listed, there are several other items such as Oracle's Spatial option for storing geographical data, a timezone option for applications across multiple time zones, the CATSYS system for creating keyworded indexes on structured text that also facilitate application development.

Step 2 – Understanding the Architectural and Terminology Differences

TurboIMAGE has its own set of terminology, as does Oracle. Additionally Oracle relies heavily on the standards based terminology of the SQL Access group. And architecturally, the database systems are much different..

In TurboIMAGE, a database is a collection of logically-related files containing both data and structural information. Each database is defined with a schema which is stored in an ASCII file and compiled by DBSCHEMA into a single physical file which is then used DBUTIL to create the physical operating system files used by the database. Each schema specifies Master and Detail Sets which will physically contain data entries. Each set is generally stored in a single physical operating system file (very large sets called jumbo sets being the exception) and there is one physical operating system file called the root file for each database that contains the database layout and control information. A TurboIMAGE application opens one or more databases via one or more calls to the DBOPEN library procedure.

In Oracle, a database is a collection of physical operating system files that contain one or more schemas along with system information. Each schema contains one or more tables along with other database objects such as indexes, views, and stored procedures. An Oracle application will connect (via an OOPEN api call or Connect using embedded SQL) to a database instance which has previously mounted and opened a database. In Oracle, an instance and a database are generally used interchangeably as in most shops there is a one to one relationship. However, a database can actually have multiple instances running against it and an instance can mount and open a single database at a time but the database that the instance mounts and opens can be different each time based on configuration files.

Operationally, the first time a TurboIMAGE database is opened; TurboIMAGE creates a global memory area called the Database Control Blocks. In Oracle, an Instance must be started before any application can connect to the database. The Instance, under Unix is a collection of distinct processes and areas of global memory called the System Global Area or SGA that are used to coordinate database functions. Under Windows, it is a single multi-threaded process. In TurboIMAGE, there are few configurable start up settings that are specified by the $CONTROL BLOCKMAX statement within the schema source file. In Oracle, there is an ASCII parameter file commonly known as the init.ora file which contains roughly 200 configurable parameters. Additionally, the Oracle Home and Oracle SID (Site Identifier) environment variables must be set before connecting to an Oracle database.

As mentioned previously, in TurboIMAGE, each set is physically stored in a single operating system file which is an MPEiX privileged file to restrict access to only privileged processes. The Oracle architecture for storing database information is much different. Oracle uses a concept called tablespaces which is simply a logical collection of physical operating system files. Additional operating system files can be added to a tablespace to increase its size. Tablespaces are then used to store the database objects such as tables, indexes, and rollback segments. When a table, view, store procedure, etc. is created, a tablespace is specified or the default tablespace for the schema is used.

Another fundamental difference between TurboIMAGE and Oracle is the treatment of sets and tables. In TurboIMAGE, there are three types of sets; Manual Master, Automatic Master and Details. Manual masters can be either stand alone or attached to one or more detail sets. The manual master serves also as an index to the detail sets. In Oracle, master and detail sets would simply be tables with no special distinction. The relationship between two tables is created using foreign key column constraints. Column constraints are generally a new term to the TurboIMAGE user even though TurboIMAGE had constraints but did not use the term as it came later as part of the SQL language. Column constraints limit or restrict what the database can do to a particular column.

Looking at the following example from the TurboIMAGE manual:

```
NAME:       SUP-MASTER, MANUAL (13/12,18);
ENTRY:      SUPPLIER (1),          <<X16 >>
                STREET-ADD,    <<X26>>
                CITY,              <<X12>>
                STATE,         <<X2>>
                ZIP;               <<X6>>
CAPACITY:   201;

NAME:       INVENTORY, DETAIL (12,14/13,18);
ENTRY:      STOCK#(PRODUCT),   <<U8>>
                ONHANDQTY,     <<J2>>
                SUPPLIER (!SUP-MASTER)<<X16>>
                UNIT-COST,         <<P8>>
                LASTSHIPDATE (DATE-MASTER), <<X6>>
                BINNUM;        <<Z2>>
CAPACITY:   450;
```

In this situation, TurboIMAGE essentially placed a primary key column constraint on the SUP-MASTER set that SUPPLIER is both a key and has to be a unique value. On the detail set, TurboIMAGE essentially placed a foreign key column constraint on the SUPPLIER column in the INVENTORY set which requires that SUPPLIER value must exist in the SUP-MASTER before it can exist in the INVENTORY set. Further, the foreign key constraint also restricts the data entry in the SUP-MASTER to be deleted if INVENTORY entries have used the SUPPLIER value.

In Oracle, we would specify the relationship between these two tables as follows:

```
CREATE TABLE SUP_MASTER
            (SUPPLIER      VARCHAR2(16) PRIMARY KEY,
             STREET_ADD VARCHAR2(26),
            CITY           VARCHAR2(12),
            STATE      CHAR(2),
            ZIP        CHAR(6));

CREATE TABLE INVENTORY
            (STOCK_NO      CHAR(8)
REFERENCES PRODUCT(STOCK_NO),
            ONHANDQTY  NUMERIC(7,0),
            SUPPLIER       VARCHAR(16)
REFERENCES SUP_MASTER(SUPPLIER),
            UNIT_COST      NUMERIC(9,2),
            LASTSHIPDATE   DATE,
            BINNUM     NUMERIC(2,0));

CREATE INDEX DATE_MASTER ON INVENTORY (LASTSHIPDATE);
```

There are several things to note in the TurboIMAGE example versus the Oracle example. Just like in TurboIMAGE, Oracle can specify relationships between tables that create a Master/Detail relationship. However, an Oracle table that is a detail of one table can also be the Master of another table allowing for a multi-tiered design. Although work-a-rounds are available in TurboIMAGE, this has always been a limiting design issue, again causing more application development to compensate.

Also, note that the Automatic Master DATE-MASTER was specified as an Index in the Oracle example as Automatic Masters in TurboIMAGE are indexes. They should have more properly been called an index which would have led to less confusion for new TurboIMAGE developers. Additionally, note that the dash in the item names was changed to an underscore since Oracle does not support the dash. This is a good thing since TurboIMAGE's use of the dash instead of the underscore has caused additional application development effort when parsing item names within an arithmetic context since the dash can be confused with the urinary minus operator.

Also the special character # sign had to be changed to a _NO. Another difference shown in the two schema specifications is that Oracle uses a single NUMERIC data type where as TurboIMAGE has several numeric types. Oracle approach makes it easier for the application developer. Also note that in Oracle, several columns were specified as VARCHAR2 with a maximum size that allows only the bytes required in the column to be stored. This can save large amounts of space on columns where the data rarely fills the entire column size such as a note field or an address field.

Another key area where TurboIMAGE and Oracle are vastly different is in the area of locking and concurrency. TurboIMAGE keeps locking information in a shared memory area where as Oracle uses a row level locking flag at each row being locked. Oracle's approach is more scalable in that write operations never block read operations. Before developing any Oracle applications, a thorough understanding of Oracle's model for locking and concurrency as well as transaction management should be developed.

This section highlighted some of the major architectural and terminology differences between TurboIMAGE and Oracle. The following is a list of some of the common terms discussed with their Oracle equivalents.

1. Set is an Oracle Table.
2. Data Entry is an Oracle row.
3. Field is an Oracle Column.
4. Record is an Oracle Row
5. Item has no Oracle equivalent as Oracle does not support global fields although they could be implemented using Objects.
6. Automatic Master is Oracle Index.
7. Manual Master is an Oracle Table with a single primary key. But in Oracle, a table can have more than one key or index associated with it.
8. Detail is an Oracle Table with foreign keys to the Manual Master
9. Path – no direct Oracle equivalent but is a Foreign Key with a constraint
10. Chain Count – no direct Oracle equivalent but could be implemented with a select count(*) against an indexed column.
11. Sorted Chains – no Oracle equivalent but since Oracle uses B-Tree indexes, information can be easily returned in sorted order.

12. User Class – Similar in Oracle but an Oracle user has its own password and set of privileges and access rights to data.
13. Library Procedures – called the Oracle Call Interface (OCI)
14. Intrinsic – called an API in Oracle.
15. Query functionality is available by Oracle's SQLPlus utility.

Step 3 – The Oracle Installation

Unlike TurboIMAGE which was essentially bundled with the operating system, Oracle generally requires a separate installation, although Oracle is working with the major hardware vendors such as Dell so that Oracle can be preinstalled on shipped hardware. If not, Oracle installation is definitely an area that trips up the new Oracle user and although not difficult in itself, because of the options available and tedious nature of the initial install, this is an area where an Oracle or outside consultant can be very worthwhile.

There are several good books that cover Oracle installation that are specific to the operating system. For instance, *Oracle 9i for Windows 2000 Tips and Techniques* by Scott Jesse, Mathew Hart and Micael Sale and published by Osborne is an excellent reference for installing and operating Oracle on Windows.

Another good book for installation because it documents all the configuration parameters is *Oracle in a Nutshell* by Rick Greenwald and David C. Kreines and published by O'Reilly.

Sorry, compared to TurboIMAGE, Oracle installation is tedious and time consuming. Fortunately it only has to be performed when generating new systems or upgrading to a newer version of Oracle.

Step 4 – Learning Oracle

The following are some recommendations on quickly getting up to speed with Oracle.

The first recommendation is to pick up a good SQL book and begin learning SQL both from the data manipulation part of the language (Select, Insert, Update, Delete) and the data definition part of the language (create table, alter table, create user, drop user, etc.)

Setting up Oracle on a laptop or PC running either Windows or Linux is one of the best ways to come up to speed quickly with Oracle. Oracle provides low to no cost developer editions of the product for this purpose. And although the Oracle documentation is available at the Oracle web site, downloading the PDF files to a local PC/Laptop means the documentation is always readily available even if no Internet connection is available. The Oracle documentation is a good starting point. However, the Oracle documentation can be a daunting in its size but is organized very well to find the particular topic of interest. The *Concepts* guide is a particularly good one for the new Oracle developer.

The next recommendation is to set up Oracle Enterprise Manager as the GUI interface will make the initial interactions with Oracle much easier. Oracle Enterprise Manager can also show the SQL being used to interact with the database so it is an excellent way to learn SQL and Oracle's specific syntax.

There are also a huge number of books published around Oracle. Two good ones covering installation were already mentioned. However, there is one book that is a virtual requirement for any new Oracle developer called *Expert One-on-one Oracle* by Thomas Kyte and published by Wrox. The information on locking and transaction management is extremely important for new developers and this book gives the best explanation available. Unfortunately Wrox was purchased by another publishing company so the book may be difficult to find. However, Tom Kyte maintains an online forum and should be able to help in locating copies.

```
http://asktom.oracle.com
```

Learning SQLPlus and PL/SQL should also be early on the list of items to master. PL/SQL can do so many things without having to write application code in COBOL or C that it is definitely worth the investment. A good place to start is *Learning Oracle PL/SQL* by Bill Pribyl with Steven Feuerstein published by O'Reilly.

And finally, there is Oracle University which is Oracle's education and training service. Classes are available in almost every major city and the classes are continuously offered throughout the year.

Step 5 – Setting up an Oracle schema

Once Oracle is properly installed, a user and schema must be created to create, store and manage database objects such as tables, views, and stored procedures.

In TurboIMAGE, the steps to set up a database are as follows:

1. Specify the database name, user classes, items, and sets into a schema using an ascii editor.
2. Compile the source schema using DBSCHEMA.
3. Create the physical file using DBUITL.
4. Store and retrieve data via one or more applications.

In Oracle, the steps to set up a new schema within an existing database are as follows:

1. Connect to an Oracle database instance using Oracle Enterprise Manager or SQLPlus as a user with database administrator capabilities.
2. Create a tableSpace to store the tables and other schema objects and specify one or more physical operating system data files for the tablespace.
3. Create the user who will own the schema and assign either default privileges or specify specific privileges.

4. Reconnect using Oracle Enterprise Manager or SQLPlus to issue Create table, create view, and create index commands to specify the tables and indexes.
5. Store and retrieve data via one or more applications.

Note that the steps to set up an Oracle schema and create tables and other data base objects are also straight forward. Oracle Enterprise Manager makes the process very simple. If using SQLPlus, then the syntax of the CREATE TABLESPACE and CREATE TABLE and other objects can be confusing initially as there are so many various options to consider. The GUI interface is a much easier way to get started for creating simple tables and other objects. For more complex tables and objects and for ongoing maintenance, the Oracle CREATE statements are usually stored in Ascii script files and then processed by SQLPlus.

Making the final leap

TurboIMAGE was a great part of the success of the HP 3000 computer system and was in its day an excellent database system even with the lack of Date data types and numeric scaling and precision and the design flaw of using a dash in item names. However, Hewlett Packard did not keep making the necessary investment in TurboIMAGE to keep it mainstream. And now with the end of support life looming for both MPEiX and TurboIMAGE, development strategies around TurboIMAGE have to be changed. Oracle is one viable option and with the proper mindset and an investment in time to learn Oracle, the TurboIMAGE developer can make the transition to the power and flexibility of Oracle.

Transform TurboIMAGE Data to Oracle

by Bob Green, Robelle

About the author: Bob Green has no problem writing about competing technologies. Once, in 1988, Bob sponsored a contest to promote the 3000 user group contributed library. He offered a $2500 first prize and $1000 second prize to the best new software contributions from users. When Bob selected the winners, they both were programs that competed directly with products of Bob's company Robelle. So Bob can appreciate the power of Oracle, while still loving IMAGE best in his heart.

Many customers contact our technical support team with questions and problems having to do with migrating TurboIMAGE data fields to Oracle. Since the two databases do not support the exact same datatypes, there are a number of cases where problems can arise.

The user's goal is to convert the data without loss and with the least possible changes to the programs that access and update that data. These programs are often written in COBOL, but Fortran and C are also common.

Internal and External Datatypes in Oracle

Now what happens when you prepare to transform this TurboIMAGE data for Oracle on another platform, such as HP-UX? Are there any traps that you can fall into?

Oracle has internal datatypes and external datatypes. The internal datatypes are the format that Oracle actually stores the data in. However, these internal datatypes, especially the Number format, would not be recognized by any of the standard programming languages. For example, the Number format is described as follows in the Oracle manual:

> "Oracle stores numeric data in variable-length format. Each value is stored in scientific notation, with one byte used to store the exponent and up to 20 bytes to store the mantissa. (However, there are only 38 digits of precision.) Oracle does not store leading and trailing zeros."

You cannot process such a number directly in Fortran, COBOL, or C.

Therefore, Oracle also has external datatypes, which are the formats that Oracle is willing to transform data into for calling code written in languages that include C, COBOL, Fortran and Java.

Here is how you would use the Oracle datatypes:

Oracle 7:

CHAR for X/U fields of up to 255 characters.

VARCHAR2 for X/U fields up to 2000 characters.

NUMBER for I/J/K/P/Z fields, up to 38 digits.

DATE for any field that contained a date value, such as SHIP-DATE X8, or DUE-DATE J2, or ORDER-DATE Z8. The Oracle date can also hold the time.

Oracle 8i and 9i:

CHAR can hold up to 2000 characters.

VARCHAR2 can hold up to 4000 characters.

So it appears that converting our datatypes to Oracle is straightforward.

Transforming Numeric Data

Summary: Export numbers as char, with decimal point. Define the Oracle field as NUMBER (x,y), where x is the total number of digits and y is the scale factor. You usually do not even need to change your COBOL copylibs, since Oracle can extract the NUMERIC fields in a external datatype that matches COBOL.

When exporting Image datasets to other sources, one of the common transfer file formats used is the "comma separated values" or CSV format. This is generally a flat file with one record per line, quotes around fields and commas between fields.

```
"cust-id","comment"
"12A","want web delivery"
```

Suprtool and other tools have options to generate CSV files. You will find Suprtool's Item command handy for defining the decimal place in your TurboIMAGE fields.

You will want the export file to have an explicit decimal place "." in numeric values where appropriate, since Oracle understands decimal places and remembers them. With a Display format, you only need to include an explicit decimal point as in 1234.89. STExport does that automatically when properly configured (Item command in Suprtool and Decimal Period in STExport).

SQL*Loader takes care of the alignment with the column definition e.g. NUMBER(12,2). If the file contains more decimals than the column definition, SQL*Loader rounds it up. For example, if you try to load 1234.5398 into NUMBER(12,2), the table will contain 1234.54. Negative values must have a leading sign (Sign Leading in STExport).

Migrate with CSV Files

If you use Suprtool to extract a dataset into a file, you can use Suprtool's STExport program to prepare the data in a format that the SQL*Loader will accept.

STExport lets you define the format of numeric data, including leading zeros and the position of the sign.

```
!COMMENT *** PREPARE DATA FOR ORACLE SQL-LOADER ***
!RUN STEXPORT.PUB.ROBELLE
IN DMRTABHM
ZERO LEADING
QUOTE NONE
COLUMNS FIXED
SIGN TRAILING
OUTPUT ABHMDATA
XEQ
EXIT
E
```

The resulting files are sent using the HP 3000's FTP client to the target computer with the Oracle database and are then imported with SQL*Loader.

```
!COMMENT *** FTP OUTPUT FILES TO DATAMART ***
!RUN FTP.ARPA.SYS
open 123.456.789.012
user <<login string and password>>
ascii
exitOnError
cd /isdmdata
cd macsdata_in
put ABHMDATA.pub.hsmacs ABHMDATA.txt
dir
quit
```

In SQL*Loader, use the Load Data command with the FIELDS TERMINATED BY "," clause to insert the data into your table. This is just one example, with fixed-length fields. STExport and SQL*Loader also have options to use variable-length fields.

Converting Your Program Data Declarations

However, what about converting our code that is written in COBOL and Fortran? For that, you must understand the concept of External Datatypes.

An External Datatype is the way an Oracle data value is presented to a user program. Since COBOL and Fortran cannot understand the internal format in which Oracle stores numbers, Oracle allows you to specify how you would like the data returned to your program.

But, Oracle only supports the External Datatypes that are common to most computer platforms. Unusual data types are not supported.

For character type fields, you normally select CHAR unless you want null-terminated STRING for C.

For numeric fields, here are the Oracle 7 choices that make sense for fields converted from TurboIMAGE:

Data Type	COBOL	Fortran
16-bit integer	PIC S9(4) COMP	Integer*2
32-bit integer	PIC S(9) COMP	Integer*4
32-bit float	PIC S9(n)V9(n) COMP-1	REAL*4
64-bit float	PIC S9(n)V9(n) COMP-1	REAL*8 or Double Precision
Packed-decimal	PIC S9(n)V9(n) COMP-3	Not supported
Display	PIC S9(n)V9(n) DISPLAY	Not supported

An Example

So, if you had a COBOL program with a buffer like this:

```
01 NEW-ORDER-LINE.
   05 ORDER-NUMBER PIC 9(8) COMP.
   05 CUSTOMER-NUMBER PIC 9(6) COMP.
   05 ITEM-NUMBER PIC 9(7) COMP-3.
   05 ORDER-QUANTITY 9(8) COMP.
   05 UNIT-PRICE S9(6)V9(2) COMP.
   05 EXT-PRICE S9(7)V9(2) COMP.
```

you could convert it to Oracle and use the exact same COBOL data area to process the Oracle data. Here is how:

- ORDER-NUMBER and CUSTOMER-NUMBER are K2 in TurboIMAGE, which converts to Number as the Oracle internal datatype, with 32-bit Integer as the external datatype.
- ITEM-NUMBER is P8 in TurboIMAGE, which converts to Number as the Oracle internal datatype, with Packed-decimal as the external datatype.
- UNIT-PRICE and EXT-PRICE are J2 in TurboIMAGE, which converts to Number in Oracle, with 32-bit integer as the external datatype.

Excellent. Easy data transformation and no changes to data definitions of our program.

Oracle Doesn't Support I4/J4

What if EXT-PRICE looked like this:

```
05 EXT-PRICE S9(10)V9(2) COMP.
```

In TurboIMAGE, this would be a J4, which converts to Number as the Oracle internal datatype.

But Oracle does not have an external datatype of 64-bit integer!

So you will have to use Packed-Decimal as the external datatype.

Which means changing the COBOL definition to

```
05 EXT-PRICE S9(10)V9(2) COMP-3.
```

But now your programs are different and any files you create with this data will probably need new data definitions. And the same conversion to packed will occur for any small integer that has a decimal place, even if the number of digits is less than 10.

This does not sound too bad, unless you want to keep the same source on MPE and UNIX. Or if you have hundreds of tasks that may process this data!

P.S. If you have this problem with the I4 fields in a big way, contact neil@robelle.com - Suprtool may have a solution for you.

P.P.S. Similar issues arise when migrating to other SQL databases, including mySQL.

P.P.P.S. If your internal number field has decimal positions, you will always need to convert it to an external packed datatype (COMP-3). If you convert it to an external integer datatype, you would lose the decimal places. This is true regardless of the size of the field.

Oracle References:

Amazon lists 1100 books on Oracle, so you will never want for reading material. Selecting just one or two is impossible, since one Oracle expert recommends 14 books on just performance tuning.

Oracle 7 documentation:

```
http://www.kbs.twi.tudelft.nl/Documentation/Database/Oracle7/DOC/dcommon/oin/
```

Oracle 8 documentation:

```
http://www-rohan.sdsu.edu/doc/oracle/
```

Oracle 9 documentation:

```
http://www.dbresources.com/oramanuals.php  (requires registration)]
```

The Underground Oracle FAQ Site

```
http://www.orafaq.com/
```

SQL Server And TurboIMAGE Data

by Bob Green, Robelle

About the author: Bob Green has made two major moves in his life (but still works on HP computers, as he did as a teenager). In 1974 he moved from Silicon Valley to Vancouver Canada for a one-year 3000 user project. 21 years later he moved from there to the small Caribbean island of Anguilla (www.news.ai), where he built a software center on the beach, with solar power and high-speed Internet. So he knows something about migrating.

In an earlier chapter, we explored the process of transforming TurboIMAGE data into formats that are acceptable to Oracle. If you haven't read that chapter, this would be a good time to read it, since it covers the basics of MPE/IMAGE datatypes, which are important regardless of why you are ransforming your data.

The datatypes in Microsoft SQL Server look a lot like Oracle datatypes.

Character Datatypes:

For character data, the most common datatype is VARCHAR(n) with a specified maximum number of characters. If all data values will be the same length, you can use CHAR(n) instead. If you want to use UNICODE values (i.e., characters beyond the 256 coded in ANSI such as appear in non-English languages), you will use NCHAR(n) and NVARCHAR(n).

Numeric Datatypes:

For integer values without a decimal place, you can use Integer datatypes:

Type	Description
TINYINT	8 bits, 1 byte, unsigned, values 0 to 255.
SMALLINT	16 bits, 2 bytes, signed, values –32,768 to +32,767. Same as I1 in TurboIMAGE.
INT or INTEGER	32 bits, 4 bytes, signed, values –2B to + 2B. Same as I2 in TurboIMAGE.
BIGINT	64 bits, 8 bytes, signed integer, aka QUAD; for very large values, up to 18 digits. This was introduced in SQL Server 2000 and is the same as I4 in TurboIMAGE. Before you select this datatype, ensure that COBOL supports it on your platform.

NUM or NUMBER or DEC	numbers with decimal places. You specify a **precision** and **scale** for the values. Precision is the maximum total digits in the values, with 38 the largest allowed by SQL Sever. Scale is the number of places to the right of the decimal. The maximum number of digits that can be placed to the left of the decimal is precision-scale. For example, DEC(7,2) means the same as S9(5)V9(2) in COBOL. NUMERIC or FLOAT is the datatype for any value with a decimal place. NUMERIC in SQL Server is much like NUMERIC in Oracle, although it does not have the odd "negative scale factors" of Oracle (scale factor-3 in Oracle actually multiplies the value by 1000!).
FLOAT(n)	approximate numeric values in floating point format. Supported in 4-byte and 8-byte formats. A floating point number has an exponent and a mantissa. FLOAT(n) specifies number of bits for the mantissa, which can be up to 53. 1 through 24 specify a single precision real (4 bytes) and 25 through 53 specify double precision (8 bytes). Same as e2 and e4 in TurboIMAGE.

Other SQL Server datatypes that you will find useful are MONEY and DATETIME.

SQL Tables and TurboIMAGE Datasets

Suppose you have this CUSTOMER table in your SQL Server database:

CUSTOMER Table Data

Column Name	**Datatype**	**Nullable**
CustomerID	INT	No
FirstName	VARCHAR(10)	No
LastName	VARCHAR(16)	No
Address1	VARCHAR(26)	No
Address2	VARCHAR(26)	No
City	VARCHAR(12)	No
State	VARCHAR(2)	No
ZipCode	VARCHAR(16)	No
CreditRating	DECIMAL(9,2)	No
CustomerStatus	VARCHAR(2)	No

And you have this M-CUSTOMER dataset in your TurboIMAGE database:

```
M-CUSTOMER     Master               Set# 1
           Entry:                       Offset
               CITY               X12     1
               CREDIT-RATING      J2      13      <<s9(7)V9(2)>>
               CUST-ACCOUNT       Z8      17      <<Search Field>>
               CUST-STATUS        X2      25
               NAME-FIRST         X10     27
               NAME-LAST          X16     37
               STATE-CODE         X2      53
               STREET-ADDRESS     2X25    55
               ZIP-CODE           X6      105
           Capacity: 211 (7)  Entries: 12  Bytes: 110
```

All of the X fields in TurboIMAGE have been converted to VARCHAR fields.

The CUST-ACCOUNT number field in TurboIMAGE was a Z8 field to enforce better hashing of the values, but Z indicates that it always contains numeric values. Therefore, INT is an appropriate SQL Server datatype for CustomerID.

The CREDIT-RATING field is a 32-bit integer on the HP 3000, defined with two decimal places in the COBOL programs. Since SQL Server has a NUMERIC datatype that is aware of decimal places, it is more appropriate to use that datatype than INT.

The ZIPCODE field has been expanded from X(6) to VARCHAR (16) to handle extended and foreign postal codes.

Repeated Items Not Supported in SQL Databases

The repeated item STREET-ADDRESS (2X25) is also known as a Compound Item. It consists of an array of 2 elements, each of which is 25 bytes long. SQL databases have no such data structure.

Therefore, STREET-ADDRESS will have be converted into two independent fields, ADDRESS1 and ADDRESS2. Here is how you would do the extract in Suprtool:

```
get m-customer
define address1,street-address(1)
define address2,street-address(2)
extract address1
extract address2
. . .
```

Migrating Date Fields to an SQL Database

Most SQL databases allow you to import dates in either Month/Day/Year format or Day/Month/Year, but there are still configuration commands that you must get correct.

The ISO standard format for dates is actually

```
YYYYMMDD  e.g. 20030428
```

If you export date fields in this format, you should have minimal problems loading into your SQL database. On the other hand, if you have an HP 3000 date field in the form MMDDYY (e.g. 042803) And export it in that format, you will need to configure your import carefully to ensure that your SQL database contains the correct date.

If you are using an odd binary date format, such as MPE's Calendar format or one of the PowerHouse date formats, you can use Suprtool to convert it into a standard YYYYMMDD format.

Moving TurboIMAGE Data to SQL Server

There are three ways to move your TurboIMAGE data to SQL Server:

1. Export your data to a file, then use the BULK INSERT statement in SQL.
2. Export your data to a file, then use the BCP command-line utility (BCP stands for Bulk Copy Program).
3. Export your TurboIMAGE data to a file or create an ODBC link to it, then use Data Transformation Services (DTS).

BULK INSERT and BCP are very similar in their parameters and options, but BULK INSERT is faster because it doesn't go through as many layers of network code. DTS is a much more sophisticated service, including a wizard to help define your import or export transaction.

The default SQL Server import file expects all field values to be in character format, then uses Tab as the field terminator and \newline as the row terminator.

Use Suprtool For SQL Server Import

Suprtool from Robelle has all the options needed to generate an import file that is compatible with SQL Server.

For example, to export the contents of the M-CUSTOMER dataset to the CUSTOMER table, you would use the following steps:

On the HP 3000, SUPRTOOL extracts the fields in the order they appear in the SQL table (or you could use a FORMAT file with SQL Server to do the reordering, but SUPRTOOL EXTRACT seems simpler to me):

```
:run suprtool
Base custdb.base
Get m-customer
Item credit-rating,decimal,2
Extract cust-account,name-first,name-list
Extract street-address(1),street-address(2)
Extract city, state-code,zip-code,credit-rating,cust-status
Output sdfile,link
Exit
```

This Suprtool task gets the column values out of the dataset and puts them in the proper order for the SQL table, but they are still in the native HP 3000 format. They still need to be converted to ASCII characters and have Tab terminators inserted. This is done using SUPRTOOL's STExport utility. Here are the commands to take the self-describing file (SDFILE) created by SUPRTOOL and convert it into the file that SQL Server expects:

```
:run stexport
Input sdfile    (created above by Suprtool)
Delimiter tab
Quote none
Date yyyymmdd
Output sqls01
Exit
```

By default, STExport creates a variable length record file for output. This is just what we need to copy to the Windows server. (Note: Although there are no date fields in this table, I included the suggested date format, YYYYMMDD, since this format is always recognized by SQL Server.)

Use FTP to transfer the STExport output file to your SQL Server system, but remember to do it as an ASCII transfer, not a BINARY transfer. This is so that the \newline characters are translated properly at the end of each row.

```
:run ftp.arpa.sys
open wins2.Robelle.com
user admin passwd
ascii
put sqls01 sqls01.txt
quit
```

On the Windows Server, you can use BCP or BULK INSERT to insert the rows into the CUSTOMER table.

In BCP you select the character option (-c):

```
bcp custdb..customer in sqls01.txt –c
               -S servername –U userid –P password
```

With BULK INSERT, you want to select 'char' as the file type:

```
BULK INSERT custdb..customer from "c:\sqls01.txt"
               WITH DATAFILETYPE='char'
```

If the file from the HP 3000 does not have the columns in the order of the SQL Table, or you need to skip some fields, then you need to create a **Format File.**

This is beyond the scope of this chapter, but is described well in the SQL Server user documentation.

Microsoft SQL Server Has No COBOL Precompiler

With Oracle, the package comes with pre-processor for Fortran and COBOL, but SQL Server seems to only come with a pre-processor for C++. What if you want to do SQL Server functions in your COBOL program? You must look to your compiler vendor.

For example, AcuSQL has an Embedded SQL (ESQL) precompiler that lets you embed standard SQL directly into ACUCOBOL program.

```
http://www.acucorp.com/Solutions/access3.html
http://www.acucorp.com/Solutions/acusql.html
```

MicroFocus COBOL says "COBOL precompilers for the different databases must be obtained from the appropriate database vendor." Fujistu COBOL claims Access to SQL Server via the ODBC interface.

I looked for a FORTRAN precompiler for SQL Server, but did not find one, so that problem is left to the reader. To wrap up, below are some resources on SQL Server to help you with your migration.

SQL Server Resources

"Teach Yourself Microsoft SQL Server 7 in 24 Hours" by Matthew Shepker. Book for beginners.

"Inside Microsoft SQL Server 2000", Kalen Delany. Advanced, internal structures.

SQL Server books online – Microsoft web site

```
http://www.microsoft.com/sql/techinfo/productdoc/2000/books.asp
```

SQL Team – news and forum site

```
http://www.sqlteam.com
```

SQL Server Resources

```
http://www.sqldts.com/dblinks.asp?nav=1,2,30,1
```

SQL City

```
http://www.mssqlcity.com/
```

DTS web site

```
http://www.sqldts.com/
```

DTS FAQ

```
http://www.sqldts.com/catlist.asp?nav=1,6,6,2
```

SQL Web sites

```
http://www.mssqlcity.com/Links.htm
```

Consider a Free Open-Source Database

by Aaron Holmes, Robelle

About the Author: Before he had even graduated from high school, Aaron Holmes began working for Robelle as a summer intern and then a junior programmer. One of his ongoing projects for us was evaluation of open-source databases, specifically mySQL, PostgreSQL, and SapDB.

From TurboIMAGE to mySQL

mySQL is an Open Source database that is commonly used as the backend database server for many Web applications for a huge number of Linux and Unix platforms, as well as Windows machines. The database is simple and fast, but has limited features, compared to Oracle or SQL Server.

The source and or binaries can be obtained from www.mysql.com or www.sourceforge.net and many other download mirrors around the globe.

As an experiment, we built a mySQL database that looked like an Image database, building a simple Master dataset and a single detail dataset. The byte type fields in Image were created as char fields. The I2 or J2 fields were created as int fields. The database resided on a small Windows 98 laptop system, but it could have been any Windows PC or Linux box.

Having done this, we extracted data from the TurboIMAGE database and used default STExport settings to output a file that was comma delimited, with each field enclosed in quotes.

We then attempted to import the comma-delimited file into mySQL.

In investigating how to import data into mySQL, we first tried the mySQLImport program, but it didn't seem robust and we could not figure out how to tell it what delimiters to use.

In looking at the documentation, we thought that the LOAD_FILE command might work, but further investigation showed that this command opens the file and returns the contents as a string. This feature is only used by Text and Blob columns of mySQL.

We finally had success with the **LOAD_DATA** statement after reading the documentation on the mySQL Website at:

```
http://www.mysql.com/doc/L/O/LOAD_DATA.html
```

We tried importing the data with the default STExport settings. However, when we looked at the data, some records were not imported correctly. The records seemed to have the data offset by one field. We found the problem to be records with either quotes or commas in the data.

Since quotes and commas were field data and also our delimiters and separators, we changed the Suprtool and STExport commands to the following:

```
get m-customer
out mcust,link
xeq
export in mcust
export quote none
export delim "?"
export out mcustexp
export exit
```

We got rid of the quotes surrounding each field by using the Quote None statement and changed the delimiter to a character that we knew did not exist in our data. In our case we chose the Question Mark.

Importing the data into our table then became a single simple command entered from within Mysql Monitor:

```
 load data infile 'm_customer_file.txt' into table m_customer
 fields terminated by '?';
```

mySQL Import Options

Here is the full syntax for LOAD_DATA:

```
LOAD_DATA 'filename.txt' INTO TABLE tablename
    FIELDS
      TERMINATED BY "\t"
      [OPTIONALLY] ENCLOSED BY "
      ESCAPED by "\\"
   LINES TERMINATED BY "\n"
```

If you don't specify a FIELDS clause, the command acts as follows: Look for line boundaries at newlines, break lines into fields at tabs, do not expect fields to be enclosed within any quoting characters, interpret occurrences of tab, newline or "\" preceded by "\" as literal characters that are part of the field values.

If you specify a FIELDS clause, you can change the delimiter from Tab to comma (or another character).

Having studied this syntax, which allows you to redefine everything from the delimiter to the end of line, we noticed the Escaped By clause. This allows you to insert the Delimiter character, whatever it is, into the actual field data by preceding it with a backslash (or whatever Escape character you select).

As a result of this experience, we added the Set Escape feature into Suprtool so that you automatically escape the delimiters with \.

Accessing the mySQL Database

Once we had the basics down for importing the datasets into the mySQL tables, we could then import the entire database into mySQL.

We were able to check the data using the mySQL admin tools, but for more flexible access we installed some PHP scripts for working with mySQL databases.

Overall, we found it relatively easy with STExport to duplicate our production IMAGE database structure and contents in mySQL.

Of course, we did do a little studying in two books:

"Sams Teach Yourself MySQL in 21 days" by Mark Maslakowski and Tony Butcher, ISBN 0-672-31914-4

"MySQL", by Paul DuBois ISBN 0-7357-0921-1

From TurboIMAGE to PostgreSQL

PostgreSQL is an Object-Relational database management system that supports almost all SQL constructs, including subselects, transactions, and user-defined types and functions. You can download, use, and modify the software for free, provided you include the copyright notice. Also, it is able to run under many OS types, include Unix variants and Windows machines under the Cygwin environment.

To obtain the source files and help documentation for PostgreSQL you can point your browser over www.postgresql.org and select the mirror site closest to you.

As an experiment, we downloaded and installed the software, then created a database and attempted to load it with existing data.

With the software compiled and running, the new database was created by the included c*reatedb* command, which accepts the syntax *createdb databasename.*

We needed some test data for our new database, so we used some old data from an existing address book program, which was stored in a comma delimited text file (Robelle's Suprtool creates such files easily from TurboIMAGE data on the HP 3000).

Within the empty database, we built a table using the *psql* command of PostgreSQL. To run *psql* you type **psql mydbname** from the command prompt. *Psql* is the interactive terminal program that allows you to interactively enter, edit, and execute SQL commands directly.

The actual code to create a table within *psql* is as follows:

```
CREATE TABLE addressbook (
           firstname            varchar(25),
           lastname             varchar(25),
           email                varchar(20),
           phone                int(11),
           age                  int(2),
           );
```

The new table mimics the existing data file that we had; included were a couple of int columns and several varchar columns of varying size. The design of the table was overall very simple.

The next step was to populate the table with the data from the text file. This was easy to do by using the *COPY* command, which loads large amounts of data (either character or Binary) from flat-text files. You tell *COPY* which file to load by specifying FROM `'/directory/file'`. By default, *COPY* uses a tab ("\t") character as a delimiter between fields; we changed this behavior by including `USING DELIMETERS ','`

The *COPY* command, however, has a few potential pitfalls. Such as if you don't have enough columns in the file, you will get an error, but if you have too many columns you will get a warning only and the extra columns are ignored. Also remember that *COPY* is executed as a transaction, meaning that a single error in the data causes an undo of the entire import operation.

As always, it is good practice to read over the intricacies of the *COPY* command in the PostgreSQL help docs, which are online at

```
http://www.postgresql.org/idocs
```

It is entirely possible to access, modify and remove the data from your database within the *psql* tool, but an alternative method is to use PHP access the database via the web. PHP, when compiled with the -with pgsql directive, comes with a vast library of functions for interacting with PostgreSQL (much the same as with mySQL).

Overall, the migration to PostgreSQL went smoothly without any unwarranted surprises. Of course this would not have been possible without the help of a few resources. Kudos to the PostgreSQL team for supplying the online documentation files on their website.

We also used two books to help us:

"PostgreSQL Developer's Handbook" by Evald Geschwinde and Hans-Jurgen Schonig. Published by Sams, 2002.

"Beginning Databases with PostgreSQL" by Richard Stones and Neil Matthew. Published by Wrox Press, 2001

PostgreSQL Data Types

The datatypes in PostgreSQL look a lot like Oracle datatypes.

Character Datatypes:

For character data, the most common datatype is VARCHAR(n) with a specified maximum number of characters. If all data values will be the same length, you can use CHAR(n) instead. For unlimited length text fields, use TEXT, which does not have a declared maximum size. Although the type text is not in the SQL standard, many other RDBMS packages have it as well. The longest possible character string that can be stored is about 1 GB.

There are no performance differences between these three types, apart from the increased storage size when using the blank-padded type (CHAR(n)).

Multi-byte Characters such as UNICODE are supported, but you may have to recompile the database (no problem, since the source code is freely available).

Numeric Datatypes:

Smallint, int2. An integer of 2 bytes. Use for TurboIMAGE I1 or J1 or K1 fields.

Integer, int4. An integer of 4 bytes. Use for I2, J2 or K2 fields. This is the default datatype for integer fields.

Bigint, int8. An integer of 8 bytes (equivalent to I4 "quad" integers on the HP 3000). Use for TurboIMAGE I4 or J4 fields. Oracle does not have this datatype, but SQL Server does.

The bigint type may not function correctly on all platforms, since it relies on compiler support for eight-byte integers. On a machine without such support, bigint acts the same as integer (but still takes up eight bytes of storage). However, the PostgreSQL team is not aware of any reasonable platform where this is actually the case.

You may want to use type Integer for indeces, rather than Smallint or Bigint.

Otherwise you must escape constants to avoid type casting errors. See the user manual for details.

Real. A 4-byte floating point field for approximate numeric values. Use for an E2 in TurboIMAGE.

Double Precision. An 8-byte floating point field for approximate numeric values. Use for an E4 in TurboIMAGE.

Numeric (p,s) - precise numbers with possible decimal places. Similar to the Numeric datatype in Oracle and SQL Server, except that there is no limit to the precision!

You specify a precision and scale for the values. Precision is the maximum total digits in the values. Scale is the number of places to the right of the decimal. The maximum number of digits that can be placed to the left of the decimal is precision minus scale. For example, DEC(7,2) means the same as S9(5)V9(2) in COBOL. NUMERIC in PostgreSQL is similar to NUMERIC in Oracle, although it does not have the odd "negative scale factors" of Oracle (scale factor-3 in Oracle actually multiplies the value by 1000!).

Numeric is suitable for storing monetary values and others where exactness is required. However, Numeric is much slower than Real or Double Precision.

There is currently a **Money** datatype in PostgreSQL, but it has been deprecatedbecause it is US-oriented. They suggest using Numeric instead.

Other Datatypes:

Bit. String of bits.

Bytea. For storing binary strings, i.e., values without a character set or collation sequence attached. Similar to the BINARY LARGE OBJECT or BLOB in SQL 99.

Boolean. Value can be either True or False.

Date. Accepts dates in almost any format and has commands to specify how to handle ambiguous cases. The default and preferred format is the ISO-8601 format, as in 1999-01-08.

There are also datatypes for storing geometric information. And, users may add new types to PostgreSQL using the CREATE TYPE command.

PostgreSQL Precompilers:

As with SQL Server and unlike Oracle, PostgreSQL does not come with a precompiler for COBOL or FORTRAN. However, you may be able to use an SQL compliant precompiler from your compiler supplier, and there is a full-featured implemented of ODBC.

PostgreSQL does provide precompilers/interfaces for C, Perl, PHP, Java, Python and TcL/TK.

SAPdb, Enterprise-Class Open-Source Database

SAPdb is designed for the business enterprise, with 24x7 uptime, scalability and high performance in mind. There are no limitations on database sizes or on the number of users. SAPdb is ACID compliant (fully supports ISO-SQL 92 Standards) and includes all RDBMS and enterprise features expected in an open DBMS such as Views, triggers, foreign keys, constraints of various kinds, stored procedures, versioning, hot backups, etc. This allows SQL applications written in other databases to be easily portable to SAPdb. SAPdb also includes a C/C++ precompiler and interfaces for Perl, Python, and PHP script languages. For COBOL and Fortran, you will need to consult your compiler source for an SQL pre-compiler. ODBC and JDBC drivers are supported for connectivity to Windows-based and Java-based applications respectively.

SAP also distributes a web-based interface for administering databases and performing SQL duties.

SapDB Installation and Configuration

SAPdb is an open-source database available for download from the SAPdb website (www.sapdb.org). It supports Linux, Solaris, WindowsNT/2000, HP-UX, and a few other platforms.

The manuals for SAPdb are sometimes hard to follow. We found the online docs easier to use than downloading the PDF's from sapdb.org because we found ourselves digging through the various manuals one by one to find the answer to questions.

We left everything as default, just changing the port setting to 9999 in WebAgent73.ini (which is found in the <install-path>/web/config directory). By leaving everything as default we had chosen to run the web tools software under the included SAPdb web server rather than our own web server (Apache, etc.). To run the web tools web server we executed wahttp (found in <install-path>/web/pgm). This starts the server and allows access to the database from the web (remember to do an 'export LD_LIBRARY_PATH=<install-path>/web/lib' command before running wahttp, otherwise wahttp will complain about missing library files).

Creating the SapDB Database and Tables

The easiest way to create a test database is to run the included script named create.demo_db.sh (which is in the <install-path>/depend/misc directory). It creates a new database named TST, plus new users. The first username is DBM, with a password of DBM. This is the database administration, who has privileges to use the SAPdb dbmcli tool. Remember, DBM does **not** perform SQL queries and functions on the database. The script creates a separate username TEST, with a password of TEST, for SQL purposes.

We connected to the new database via the web SQL interface by pointing my browser to http://localhost:9999/websql (this is the URL for the SQL tool - /webdbm gives you the database administration tool). Using the database name of TST, username TEST, and password TEST, we logged into the SQL front end for the database. The interface is fairly clean and easy to use. One complaint we have is that the web tools uses JavaScript and CSS, which can cause some issues in older browsers.

With the browser running the SQL interface, the 'show table' SQL statement shows the tables in the TST database, including the system and configuration tables. There's quite a few of them and if you look under the owner column you can see who they belong to. DBA and DOMAIN own the tables for database administration and configuration, and SYS owns the tables for system settings.

Since none of these tables belong to user TEST, we shouldn't be mucking around with them. Creating a new table will give us something to play with. Using the standard 'create table {…}' SQL syntax, we created a new table called alumni. I won't go into detail about each of the SQL commands that we used. If you do need help with them, you can refer to the "SQL Statements: Overview" chapter in the

SAPdb reference manual. Next, we put in a small amount of test data using 'INSERT'. This also went smoothly, so we decided to try loading bulk data from a comma-delimited file.

Here things became a little tricky… The difficulty was in finding the proper section of the documentation for importing data from external file sources. We finally found a tool named **repmcli**, which is included with SAPdb in the <install-path>/depend/bin folder.

We managed to load the test data into the table, although we had to put double quotes around each data field. Double quotes are a SAPdb default, and can be changed to a different character if necessary. If you are using Robelle's STExport tool, use the 'quote double' command.

The repmcli command includes a rich set of features; it can accomplish some fairly complicated things by executing a batch file. The batch file includes specific commands and SQL statements which tell SAPdb how to load the data. In our case we used a simple batch file, which I created in VI and named command.dat with the following code:

```
FASTLOAD TABLE alumni
               Id 1
               Birthdate 2
               Title 3
           INFILE 'testdata.txt'
```

In this batch file, alumni is the test table, and testdata.txt is our comma delimited data file with double quotes surrounding each field entry.

SAPdb Conclusion and Comparisons with Other Databases

SAPdb is the number one open source database in respect to features, and is definitely geared towards the enterprise server market. The result is a **very high learning curve** for SAPdb. I often found myself spending far too much time just looking for a certain command that I knew must exist, but could not easily find. The documentation is there, but is hard to navigate and I could not find any third-party books written about the database engine of SAP.

The natural comparison is between SAPdb and PostgreSQL, another open-source database. Both databases have many of the advanced features found in enterprise level database solutions, and focus on features/performance. PostgreSQL is nowhere near as hard to learn as SAPdb. It also comes with online documentation and many third-party publishers have books relating directly to PostgreSQL. For my needs I found that I could do everything I had to in PostgreSQL, and in half the time or less than it took me in SAPdb.

The real target audience for SAPdb is in the enterprise market, where behemoths like Oracle have dominated the market for years. SAPdb strives to offer all of the features available in Oracle with high performance, but for free. This sounds a bit too good to be true, and at this current moment I think SAPdb still has to make some progress. Your best bet by far is to try out SAPdb on a test server, with test data and see if it offers what you really need.

Converting Multi-Line Notes

By Dave Lo, formerly of Robelle

About the author: Dave Lo has a B.Sc. in Computer Science from University of British Columbia (UBC) and joined Robelle in 1989. Among many other skills, including patient hand-holding on the telephone and web design, Dave was our Perl programming expert. When we asked the Robelle staff, what to do about IMAGE's typical multi-record detail notes when migrating to SQL, Dave came up with this simple, but elegant solution. Dave is now a programmer at www.interactivetools.com, which creates website content management software. Email: dave@davelo.net

When exporting TurboImage datasets to other sources, such as Access or SQL, one of the common transfer file formats used is the "comma separated values" or CSV format. This is generally a flat file with one record per line, quotes around fields and commas between fields.

```
"cust-id","comment"
"12A","want web delivery"
```

Suprtool can easily produce this type of file with the OUTPUT,PRN command or with the STExport module. However, there is field type that cannot be handled this way directly: the Notes fields. These fields span multiple records even though they are logically one field. They usually look like the following:

```
cust-id    seq-num    comment
12A        001        want web delivery but
12A        002        limited by bandwidth,
12A        003        so use FTP.
88X        001        Send doc in "PDF" format.
99Z        001        Make sure all changes
99Z        002        are approved by John.
```

In the CSV transfer file, we want to merge the related records into a single record, and also convert any double quotes to single quotes (you need to do something about internal quotes if you want to use quotes around fields in the CSV file!). For example:

```
"12A","want web delivery but limited by bandwidth, so use FTP."
"88X","Send doc in 'PDF' format."
"99Z","Make sure all changes are approved by John."
```

How to Combine Multiple Notes into a Single Text Field?

Although Suprtool cannot produce this format directly, it can front-end the database extract portion, and let a straight-forward Perl script do the merging. To learn more about Perl, read our "Introduction to Perl" at

```
www.robelle.com/tips/perl.html
```

1. First use Suprtool to dump the Notes dataset, sorted by customer number and sequence number:

```
Get  Notes
Sort cust-id
Sort seq-num
Ext  cust-id, ^i, comment  {tab delimited}
Out  notefile
Xeq
```

2. Send the resulting notefile to your other OS/directory space where Perl is available.

3. Save the following Perl script as merge.pl :

```
# adjust these according to your in/out data format
#
#   Input:
#
$in_field_sep = "\t";

#
#   Output:
#
$quote = '"';
$alt_quote = "'";
$rec_sep = "\n";
$field_sep = ",";
$note_term = " ";

$keycount = 0;
$prev = "";
while ($line = <STDIN>) {
    $line =~ /([^$in_field_sep]+)$in_field_sep(.*)/;
    $key = $1;
    $text = $2;
    $text =~ s/$quote/$alt_quote/g;
    if ($key ne $prev) {
        if ($keycount>0) {  # close previous quote
            print $quote . $rec_sep;
        }
    print $quote . $key . $quote . $field_sep . $quote;
    $keycount++;
    }
    print $text . $note_term;
    $prev = $key;
}
print $quote . $rec_sep;
```

4. Run the Perl script against your notefile to produce the CSV file.

```
perl merge.pl <notefile >note.csv
```

How to Customize the Text Field

If your CSV file format requirements are slightly different, you can modify the Input/Output variables to suit your needs. Some common changes may include:

- Use two double-quotes to represent a single double-quote.

```
$alt_quote = '""';
```

For example,

```
"88X","Send doc in ""PDF"" format."
```

- Use a newline to terminate each portion of the note. This is often used when a single logical note is more than 64Kbytes long.

```
$note_term = "\n";
```

For example,

```
"12A","want web delivery but
limited by bandwidth,
so use FTP.
"
```

- Use a tab to separate fields.

```
$field_sep = "\t";
```

- There is no input field separator. For example, the key might be a 4-digit alphanumeric string, followed immediately by the comment.

```
A123Awant web delivery but
A123Alimited by bandwidth,
A123Aso use FTP.
B888XSend doc in "PDF" format.
Z999Make sure all changes
Z999are approved by John.
```

In this case, change the parsing of the "$line =~ ..." to

```
$line =~ /([A-Za-z0-9]{4})(.*)/;
```

As you can see, a little bit of Perl can indeed live up to its name of being a "Practical Extraction and Report Language".

Cleaning Your Data

By Neil Armstrong, Robelle

About the author: In 2001, Neil Armstrong, Suprtool Software Architect for Robelle, moved to the Caribbean island of Anguilla to join Bob Green. Since the move, Neil has done more enhancements to Suprtool than has ever been done to the product in such a short period!

It is very frustrating to have a data migration fail because your HP 3000 data fields contain characters, usually invisible, that make the database import fail on your target machine. Sometimes un-printable or extraneous characters that have no business in your database, get stored their anyway -- perhaps some tab characters in an address field or an embedded carriage return or line-feed. What you need is a way to "clean" your data before moving it to a new home.

Suprtool has just such a feature: the Clean Command (Suprtool's STExport module even has an option to clean **all** of the byte-type fields in a file). There are three things that Suprtool needs to know in order to "clean" a field: which characters to clean, what character to change the "bad" characters to, and also what fields to clean.

Defining a Clean Character

The Clean command tells Suprtool what characters to look for:

```
clean "^9","^10","."
```

This tells Suprtool to replace the tab character (Decimal 9), Line Feed (Decimal 10), and a period with the configured "Clean" character (often a Space).

The Clean command takes both, decimal notation and the character itself, however, it is probably most convenient to use the Decimal notation for the characters that you wish to clean. The Decimal notation is indicated by the "^" character and specifies the decimal values in ASCII; i.e., Decimal 0 thru Decimal 31 for non-printing characters. You can also specify a range or characters by using the following syntax:

```
clean "^0:^31","^240:^255"
```

Setting the Clean Character

By default, Suprtool replaces any of the characters specified in the Clean command with a Space. To specify an alternate "clean" character, use the following Set command:

```
set CleanChar "."
```

This causes Suprtool to replace any of the qualifying "to be cleaned" characters with a period.

Cleaning a Field

In the If and Extract commands, you call the Clean function the same way you normally use other functions. For example:

```
ext address1=$clean(address1)
```

This shows how to clean the field address1. The target field does not have to be the same as the source field.

```
def new-address,1,30
ext new-address=$clean(address1)
```

An Example of Cleaning Your Data

An example of how easy it is to clean your database of certain "bad" characters in byte-type fields is as follows:

```
>base mydb,1,;
>get customer
>clean "^9","^10","^0","^7"
>set cleanchar " "
>update
>ext address(1) = $clean(address(1))
>ext address(2) = $clean(address(2))
>ext address(3) = $clean(address(3))
>xeq
```

The above task looks at the three instances of address and replaces the tab, linefeed, null and bell characters with a space.

STExport

This same feature has been added to Suprtool's STExport module, except that STExport can automatically clean all the byte type fields for a given Self-Describing file. The commands are very similar, except STExport only needs to know what the replace character should be and what characters it needs to look for.

```
$ in mysdfile
$clean "^9","^10","^0","^7"
$set cleanchar " "
$out myexport
$xeq
```

Since the Cleanchar is by default set to space, the above task could simply be:

```
$in mysdfile
$clean "^9","^10","^0","^7"
$out myexport
$xeq
```

A New Operating System to Replace MPE

Unless the 3000 emulator project is successful, you will be learning a new server operating system: perhaps HP-UX, Linux, SunOS, or Windows 2000. Therefore, we show how to start learning UNIX on your MPE system, how to adapt to UNIX as an MPE user, how Robelle ported Qedit from MPE to HP-UX over ten years ago, and how one ISV is porting from MPE to Windows. -Bob Green

Learning Unix on MPE/iX

By Glenn Cole

About the author: Glenn Cole has been one of our most helpful Robelle customers for a long time. And this contribution continues his helpfulness.

```
> I am about to migrate from HP3000 to HP9000.  Will I have
> to learn totally new commands or is it the same?
```

In response to a cry for help from an MPE user on the HP 9000 mailing list, long-time 3000/9000 user Glenn Cole (glenn@non.hp.com) gave some good tips.

As others have written, it's a totally new world. As it happens, you can get a head start on the basic commands and concepts today, with your 3000.

While MPE has a single command interpreter (CI), UNIX has several from which to choose. Instead of being called a CI, though, it's called a "shell." On MPE, the UNIX-like interpreter is called the "POSIX shell." In the samples below, colon (:) is used as the MPE prompt, and dollar ($) is used as the POSIX shell prompt. It is likely that neither is used on your machine.

Launch the POSIX shell. (On HP-UX, you would just login.)

```
:xeq sh.hpbin.sys -L
```

Enter the POSIX (and UNIX) equivalent of :SHOWTIME

```
$date
```

Exit the POSIX shell. (On UNIX, this would log you off; on MPE, it just returns to the MPE prompt.)

```
$exit
```

The standard text editor in UNIX is vi. (emacs is also common, but it's not available on the 3000, or if it is, it's not installed by default.) The advantages of learning vi are that you'll find it on every UNIX or Linux box, and that it's powerful enough to use in daily development. The disadvantages are that it is nowhere near as powerful or easy to use as Qedit, and the learning curve is exponential. Other than that, it's fine.

HP has thoughtfully included a tutorial on vi, using text files that you view and edit with vi. To get started:

Launch the POSIX shell

```
:xeq sh.hpbin.sys -L
```

Copy the files to your directory

```
$cp /hpshell-examples/*.v .
```

cp is the UNIX 'copy' command

`/hpshell-examples/*.v` refers to all files in the /hpshell-examples directory ending in '.v' (the UNIX and POSIX shells use '*' instead of '@')

The dot (.) refers to the current directory

See what files were copied, with the UNIX version of :listf

```
$ls *.v
```

Open the intro file in vi

```
$vi browse.v
```

If you need to exit vi before learning how ;)

```
:q!
```

Hint: enter all three chars from inside of vi and you will be out.

Exit the POSIX shell

```
$exit
```

Of course, this is just the command level. For development languages, I use Java and Perl daily, both of which are freely available for the 3000 as well.

If you end up using Oracle on the 9000, you'll probably want something to help you load data into the database and extract data out; the tools Oracle provides are abysmal. Informix has decent tools, in the form of 'load' and 'unload' commands. For Oracle, I ended up writing Perl simulations of the Informix load and unload commands. While these work fine for my needs, I suspect Robelle's Suprtool/UX is far better, and that it would be a worthwhile investment for any Oracle/HP-UX shop.

Bottom line: Kiss the simplicity of MPE and Image good-bye, but think of the migration as an opportunity to learn things that can be applied to many other platforms, including Linux and even Mac OS X!

Unix Quickstart for MPE Users

by Neil Armstrong, Suprtool Software Architect

About the author: Neil Armstrong has been writing papers for Robelle and presenting them since he joined the company. A favorite title is "Living With 400 I/Os Per Second" about the high-end N-Class 3000 servers and performance tests he ran with Suprtool and large customer databases.

Robelle plans to support the 3000 with improved Qedit and Suprtool products and service as long as we have customers, but our products already run on the HP-UX platform. If you should decide to migrate some 3000 applications to Unix, here are some tips on UNIX to get your started.

Getting Used to Unix

This chapter relates the MPE commands and concepts that you are familiar with to similar commands and concepts on HP-UX. This will not be a debate on the merits of MPE/iX versus HP-UX. It is intended to assist the MPE user, programmer, and operator to navigate and use commands on the HP-UX operating system.

HELP

On MPE we had the Help command to find out the meaning of and syntax for various commands. On UNIX we have the man command. This is used to display on-line documents, often referred to as the "man pages".

Standard UNIX divides the man pages are divided into logical sections, but Sections 6 and 8 are not included on HP-UX:

1	User Commands
1M	System Maintenance
2	System Calls
3	Functions and Function Libraries
4	File Format
5	Miscellaneous
7	Device Files
9	Glossary

In the man command, you can specify the section to be searched, as in

```
man [-] [section[subsection]] entryname
```

If you can't remember that, just remember

```
man man
```

to get help on the man command.

If you don't specify a section, man searches all the sections in numeric order. If you don't know the name of the command, you can search for a keyword by doing

```
man -k sty
```

This gives you a summary of sections and entries where the keyword occurs.

This option only works if you have a man page index file, which can be found in /usr/share/lib/whatis. If your system does not have this file, you can create it easily by logging on as root (superuser) and then doing a catman command. This process takes a long time to run, so we suggest that you run it in the background by putting an "&" at the end of the command:

```
/etc/catman -w &
```

The man command searches the directories for the man page entries in the order described by the MANPATH variable. This is just like the HPPATH variable on MPE, but it specifically describes the order in which the man command should search for specific man pages. You can see the next page of a man page listing by pressing the spacebar. You can stop a man page listing by pressing "q".

Use the Space Bar Instead of Enter

When you try the man command, you will notice that the help display pauses when it fills the screen. To continue the display, you press the Space Bar, unlike MPE where you usually depress Enter or type "Y". The Space Bar works almost everywhere in UNIX to continue a display, while Q ends the display and / brings up a string search.

Command Interpreters = Shells

MPE has just two shells: the Command Interpreter and the POSIX shell. But on UNIX the shell is just another program. As a result, UNIX systems have many different command interpreters, commonly referred to as shells. HP-UX has at least 4 shells, and you can download more and install them yourself. The four most common are sh, ksh, csh and the Posix shell.

sh - the bourne shell

The Bourne shell, whose default prompt is "$", is the oldest of the shells and is on every UNIX system. It has no job control, is the default shell assigned to root, and is commonly used for writing command files or "scripts". Note: On HP-UX 10.X, sh runs the POSIX shell by default.

ksh - the korn shell

Our personal choice for shells is the Korn shell, because it is compatible with the Bourne shell and has many features of the "C" shell. Features of the Korn shell include command history editing, line editing, filename completion, command aliasing, and job control. This is the most MPE-like of the shells.

csh - the C shell

The C shell has a default prompt of "%". It was developed at Berkeley and has lots of tricky features.

One confusing part of UNIX systems is that some commands may be specific to a shell while other commands, such as rm and cp, are programs unto themselves. One way of identifying a built-in command is to check for a man page for that command. If you don't find one, then the command is just a shell command and is documented under the shell name.

The following are some simple commands that you will need:

Listf = ls

On MPE we commonly use the Listf or Listfile command to look at the attributes of files. On HP-UX we can look at the various attributes of a file with the ls command. To see all the attributes you use ls -l. For example, to see all the attributes of the files with names starting with proc, use:

```
$ls -l proc*
-rw-rw-r-- 1 neil users 968 Feb 1 14:46 proctime.c
-rw-rw-r-- 1 neil users 740 Feb 1 14:46 proctime.o
```

By default, the ls command does not list all of the files. To list the files that begin with a period (.), you must use the -a option.

```
$ls -a -l
```

This command lists all the files, including config files that start with a period and it shows all the attributes.

Fcopy, MPEX,, and cp

On MPE we could copy files using Fcopy, Copy or the MPEX Copy command. The MPEX Copy command allows the user to specify filesets when selecting file(s). The cp command in HP-UX also allows this by default.

```
cp source.file destination.file
cp /dev/src/*.c *.c
cp -R /users/me .
```

Purge and rm

Beginning with MPE/iX 5.0, we can purge files on MPE with wildcard characters. On HP-UX the command for purging files is rm, which stands for remove files, and it has always supported wildcards. **WARNING: Never use rm -r /* -- this will remove all the files from the file system, starting from the root directory.**

```
rm file1
rm file*
```

Rename and mv

On HP-UX the equivalent to MPE's Rename is **mv**, which stands for move.

```
mv file1 file2
```

Overwriting Files

One problem with the cp, mv, and rm commands is that, by default, they do not prompt the user when they are going to overwrite an existing file. To force a prompt, you must use the -i option with the commands. You can make the -i option the default for some commands by putting an alias command in your .profile file. For example, to make rm interactive by default:

```
alias rm='rm -i'
```

File Command

There is a file command on HP-UX, but it is completely different from the file command on MPE. On MPE the file command is used to define the attributes of a file when it is built, or on what device it is built, or to redefine it as being another file.

On HP-UX the file command attempts to determine the type of a file and print it. This is not as easy as it sounds, since UNIX has no file labels or file codes to indicate the file type. Instead, UNIX looks at the file extension (i.e., .c is C source code) and a magic number at the beginning of some files.

```
file proctime.*

proctime.c: {C program text}
proctime.o {s800 relocatable object}
```

On UNIX the closest thing to the MPE file command for redirecting files is actually a symbolic link.

This should be enough to get you started on HP-UX. And it will all apply to Linux as well. For many more UNIX tips for MPE users, read my full tutorial on this topic:

```
www.robelle.com/library/tutorials/pdfs/hpuxmpe.pdf
```

Porting Qedit from MPE to HP-UX

By Bob Green, Robelle

About the author: Bob has spent his entire professional career working on HP computers, from the 2116A to the latest N-class 3000. First thing he did when he founded Robelle was to write a text editor called Qedit, short for "quick editor", that had close to zero overhead on an HP 3000 server. Then he wrote and presented a technical paper that would promote Qedit by explaining the theory behind it: "Optimizing On-Line Response." In this long saga, which led to him being elected to the HP Users Hall of Fame, he has not missed out on HP-UX. In this chapter, Bob describes some of the more interesting technical problems that occurred in porting Qedit from MPE to HP-UX.

Our Qedit full-screen text editor was first written in 1977 and has been continually enhanced since then. In 1990 we ported Qedit (and then Suprtool) from MPE to HP-UX. We thought those of you who are now migrating your own code would be interested in the technical details of how we converted 100,000 lines of code that had been written over a period of 15 years.

From first inception to delivering a production product took about 20 months. The work effort is estimated as 19 person-months. We could never have achieved this short porting time if we had rewritten all of our source code into a new language as well as for a new operating system.

Most of our development, including Qedit, is done in SPL and SPLash! SPL was the original Systems Programming Language for the HP 3000. The SPL compiler only produces object-code for Classic HP 3000s (you can also run this code in compatibility-mode on MPE/iX machines). SPLash! is a compiler from Allegro Consultants Inc. (www.allegro.com) that translates SPL source code into HPPA RISC object-code. SPLash! is the compiler that we have used for Qedit and Suprtool since the conversion to RISC processors.

Subroutine libraries

Because SPL does not rely on a language-specific run-time library, we thought it might be possible to port the SPLash! object-code to HP-UX running on the HPPA RISC architecture. The only thing left would be to replace all of the subroutine libraries on which Qedit depends. Each set of libraries depends on other sets of libraries. The libraries look something like this: Qedit, Qedit Library, Calculator, Qhelp, and Other Common Libraries, Low-level Robelle Library, and finally the MPE and Compiler Libraries. Libraries at the upper levels can call libraries at any lower level. For example, Qedit makes many calls directly to the MPE library (e.g., readx, print, fopen, fread, ...). Similarly, the Qedit calculator makes calls to both the Robelle low-level library and the MPE library.

Object-code format

We had a theory that object-code files were the same on MPE/iX and HP-UX, but we needed to verify that. To show this was the case we compiled a small SPLash! program on MPE, copied the object-code to HP-UX, linked the object-code, and ran the resulting program. Our first test program just called the HP-UX "puts" routine to print a string.

We compiled the program using SPLash! and copied the resulting object-code to HP-UX with these commands:

```
:splash testsrc,testobj
:dscopy testobj to /users/bob/test.o:hpuxdev[user:pass]
```

The HP-UX command for linking is called ld. To link our example program, we used the command $ ld /lib/crt0.o test.o /lib/libc.a

This linked the three files /lib/crt0.o, test.o, and /lib/libc.a. The first file is /lib/crt zero dot oh (don't mix up the zero and the oh), which is the C run-time startup library. The test.o file is the SPLash! object-code generated on MPE. The /lib/libc.a file is one of the HP-UX run-time libraries. Like most HP-UX command names and commands, everything must be in lower case

By default, the ld command creates a program file called a.out. To run this program, we only had to type its name (like using implied run on MPE/iX). We tried running the resulting program and got:

```
$ a.out {run our sample program}
Hello, World! {result of running a.out}
```

This was exactly what we expected. The buffer that we initialized with "Hello, World!" was printed out on stdlist and the program stopped. We had now answered one of our most basic questions — we could compile programs on MPE, copy the object-code to HP-UX, link the object-code into a program and run it on HP-UX.

Floating-Point Numbers

In MPE, we use the compiler-library routines hpinext and hpextin to convert floating-point numbers from human-readable form to the internal IEEE format and vice versa. These routines have many options and handle different sizes of floating-point numbers. Rewriting these routines would be a lot of work.

Qedit does not use floating-point numbers very often, except in the calculator. In order to implement the calculator, we would need the functionality of the hpinext and hpextin routines. A search of the man pages revealed no documentation for either routine, although there were other routines to handle floating-point conversions. All of our code assumes that hpinext and hpextin are available. So we searched for these routines using the nm command:

```
$ cd /usr/lib
$ nm -r * | grep hpinext | more
```

The result of this search was as follows:

```
libcl.a:hpinext | 13836|extern|entry |$CODE$
libcl.sl:hpinext | 421568|extern|code |$CODE$
libcl.sl:hpinext | 421524|extern|entry |
libf.a:hpinext | 13836|extern|entry |$CODE$
libf.sl:hpinext | 421568|extern|code |$CODE$
libf.sl:hpinext | 421524|extern|entry |
```

Success! We had found the hpinext routine. We also found the hpextin routine in the same libraries (using the same technique). These routines were located in either the libcl.a or libf.a library.

This was one example of how familiar HP-UX is to MPE programmers. It is highly unlikely that any other version of Unix has the hpinext or the hpextin routines. Even if they did, they are unlikely to have the same parameters and work the same way as the HP version.

Implementing MPE Routines on HP-UX

To get ourselves started, we wrote HP-UX replacements for a number of MPE routines: binary, dbinary, ascii, dascii, print, readx, ccode, and hpsetccode. With this small set, we could start porting our most fundamental library source modules. Several modules started working on HP-UX. However, we would often try to port a new Robelle library only to find that we needed to implement additional MPE routines. So the process went back-and-forth between porting Robelle source modules and writing new MPE replacement routines. In total, we implemented about twenty MPE routines.

MPE File System Routines

Things went well until we had to port our first source module that needed to access files. Using sample code written by Stan Sieler of Allegro Consultants, Inc. we investigated how difficult it would be to implement the MPE fopen, fread, fwrite, and fclose routines.

The HP-UX file system is simple compared to the MPE file system. Files are collections of bytes. There is no implied structure to any file (although you may deduce the structure by examining the filename extension or by looking at some of the data in the file). In MPE, we have fixed-length, variable-length, ascii versus binary, record-length, blocking-factor, and many other file attributes.

Our software depends on a number of assertions that are provided by MPE's file system routines. For example, if you open a new file with a filecode of 111 and a record size of 256 words, write some records to it, close it, open the file again, then ask about its structure, you expect to get a filecode of 111 and a record size of 256 words. In HP-UX, there is no place to store this information (without using an auxiliary file).

It was at this point that we made a pivotal decision about how we would port our software. We would not emulate the MPE file system routines. Instead, we would reengineer our products to invoke an interface layer that would provide file system functionality. We would isolate the interface layer for each module that needed file system access into a separate source file (in object-oriented terminology this is called encapsulation).

Recognizing Qedit Files

On MPE, Qedit files are easily recognized as those with a filecode of 111 and a specific record length. But HP-UX doesn't have filecodes or record lengths. We designed a routine to examine the first 1,024 bytes of an HP-UX file. By looking at various relationships in this data, we hoped to determine if the file was a Qedit file or not.

To test effectiveness of our routine, we used dscopy to copy one Qedit file from MPE to HP-UX. We wrote a test program that opened every file on our HP-UX workstation and checked to see if it was a Qedit file. Our first implementations found many HP-UX files that the routine thought were Qedit files when they were not. But our final implementation found the one and only Qedit file that we had copied from MPE.

The 32-Bit Alignment Problem

The HP-UX C compiler prefers to have 32-bit variables aligned on 32-bit boundaries. In SPL and SPLash!, it is common to have 32-bit variables aligned on 16-bit boundaries. The interface layer for Qedit was much more complicated than any of the other interfaces that we had written. Because the interface was more sophisticated, large data structures had to be passed between SPLash! and C routines. We had to carefully implement these structures so that all 32-bit variables were aligned on 32-bit boundaries. We also had to insure that these structures themselves started on a 32-bit boundary.

You can force SPLash! to allocate arrays on 32-bit boundaries by making the arrays virtual. This often has unexpected side-effects (e.g., you cannot pass a virtual integer array to an option splash procedure that is only expecting an integer array). In Qedit, we ended up doing the following. We would allocate one more word of storage than was necessary and then would have code that looked like this:

```
if @ext'record mod 2 <> 0 then
@ext'record := @ext'record(1); ! assumes integer array
```

This code fragment for the local declaration of our ext'record data structure dynamically adjusts the address of the structure, but only if it does not already start on a 32-bit boundary!

“New” Files

The Qedit Redo module requires a “new” file which, on MPE, you cannot see with Listf or Listftemp. HP-UX does not have new files, but there is a solution. For the invisible scratch files needed by a program, use the following algorithm on HP-UX:

- get a temporary filename (use tmpnam or tempnam)
- open the temporary filename
- unlink the filename from the file system

If your program stops for any reason, the scratch file disappears and the space the file occupied returns to the HP-UX free space list.

Temporary Files

On MPE, Qedit uses temporary files for scratch files and for “hold” files. On HP-UX there really are no temporary files. HP-UX programs just create permanent files in a directory considered to be temporary. The different-looking filename is a result of the HP-UX routine tempnam which creates a file with a unique name in /usr/tmp (HP-UX 9.x or earlier). So we modified Qedit/UX to create the qeditscr, hold, and hold0 files in the /usr/tmp directory.

Full-Screen Mode

While migrating Qedit to HP-UX, we noticed a routine in the HP-UX manuals called **blmode**:

> "This terminal library interface allows support of block mode transfers with HP terminals."

Just what we needed and in fact, using blmode allowed us to quickly port the full-screen “visual-mode” portion of Qedit. The only major change required was that the blmode routine used a handshake based on the Classic 3000 and not on the new RISC systems. But we still had both versions in Qedit, so that was relatively painless.

This strategy worked well for 10 years. Then HP released HP-UX 11 and removed the functionality of the blmode routine (blmode was not standard Unix and we had worried about this eventuality for some time).

At first we thought we might give up on visual mode in HP-UX 11, but a large number of our users objected. They were now faced with migration to HP-UX are aghast at learning the obscure “vi” editor of Unix. In response, we wrote a low-level emulation of the block-mode screen update (i.e., we wrote our own version of blmode). Qedit/UX now supports visual mode on HP-UX 11, as well as on earlier versions.

One of our Qedit customers on HP-UX, Mike Connor of NTN-BCA, writes:

"Qedit for HP-UX has been indispensable in our migration from the HP 3000 to HP 9000. It is the only tool we have that is common to both platforms and has been very helpful in smoothing out the learning curve on HP-UX. Not only do we use Qedit/UX to edit scripts and program files, but we also use it for all kinds of data migration tasks. For example, we download data from an AS/400 machine, then use Qedit/UX to reformat the data for the HP 9000. Qedit greatly simplifies these kind of tasks."

An ISV Ports to Windows

Throwing Open A Window to Low-Cost Power

– interview of Steve Smith, founder of AMS Software, by Ron Seybold editor of *The 3000 Newswire.*

AMS Software is an application vendor that serves the small to midsize wineries who make up the vast majority of American vineyards. Steve Smith has cleared the path from HP 3000 to Windows as patiently as any winemaker, and he will see his vines grow ripe in a season soon. As AMS owner and development leader, Smith has been moving his company's application out of the MPE world for several years, and his company's work should bear fruit by this fall.

Smith has been looking for a cost-competitive future for his firm for some time. We first spoke with him in 1999, when Minisoft's Doug Greenup identified AMS as one of the smaller 3000 app providers that HP was overlooking. One reason that the 100 or so customers using Smith's AMS app didn't appear on the HP radar might have been because they did not buy 3000s from HP. Well over four years ago, AMS was dealing exclusively in used HP 3000s for its customers, and providing its own support. Even then HP was no longer a part of the AMS business plan, and his business was surviving.

Then, as Smith tells the story, HP announced it was dropping its 3000 business in 2006, and AMS competitors started carrying that news to his customers as well as prospects in the wine industry. These wineries have price pressures on their IT spending. New HP 3000s had been beyond of the wineries' budgets — and last year the affordable used systems became suspect as a result of HP's news. Smith had already seen this coming, and so had begun to research moving his application to Windows to use less costly Intel-based hardware.

We heard of Smith once again this spring, when database provider Pervasive Software offered him up as a success story of a conversion from IMAGE to the Wintel-based Pervasive database. While that deal hasn't been sealed yet, Pervasive's product looks good to Smith, a man not easily swayed by marketing enthusiasm.

Smith has been with the HP 3000 and MPE since the 1970s, so he's leading his customers and company away from MPE with some reluctance — but not a lot. We found a technically adept company leader who'd grown up with the 3000, one who's now gathered experience in the move to the Microsoft alternative, and we were hooked. As a significant part of the 3000 community looks over the platforms outside of MPE and IMAGE, we felt Smith's experience could shed light on where the pitfalls may lie on the path to the promised Redmond rewards. We spoke with him just after the latest Solutions Symposium by phone about what it's been like to adopt a Windows view after seeing the world from IMAGE-MPE eyes.

How long have you been working on moving your applications?

We've been working on it off and on for years, since 1998, but we picked up the pace about a year ago and have been working on it pretty steadily. Now I have pretty much the whole staff, six people, on it. We hope to have something to sell by the third quarter of 2003.

How do you feel about HP's decision to drop its 3000 business?

I suppose HP did us a favor. I don't want to speak heresy, but we're seeing so many capabilities out there that go beyond what we were offered on the HP 3000. It's always been difficult to sell HP 3000s, but the HP announcement made it even more difficult. It's pretty much impossible when the creator of the machine says "We don't believe in it anymore."

But we had already been working on a transition before the announcement. It pushed us down the road a little bit. We had a desire to offer a world-class product to people again. Ten or 15 years ago we had that.

Where did you decide to take your application and your customer base?

Microsoft's stuff has really matured a lot. It's still something I love to hate, but Windows XP and Windows 2000 are very capable operating systems. With the combination of HP saying goodbye to the 3000 and Microsoft technology becoming more dependable and robust, I really think it's possible.

The price of used HP 3000s is bound to drop over the next few years. Why not stay with a hardware choice that will get more cost effective?

We have a benchmark we love to quote, a sales report that runs in a little over an hour on a Series 927. It runs in 45 seconds on a Windows box we're moving this code to. Sure, it could run faster on the newer 3000s, but they cost $25,000. The box we're moving to has twin 2.4Ghz processors. I can buy that for three grand. Even if the price of used 3000s go to zero, we still have an image problem with processing power.

And why not move to HP-UX?

It's too expensive compared to what else is out there. It's cheaper than MPE, perhaps, but that's not the comparison I was making: Windows has become a stable operating system that's mainstream. We think it's reliable if you treat it right.

Since you've chosen Microsoft as the new platform, a more obvious choice of database would have been SQL Server. Why did you decide to use Pervasive's Wintel-based database instead?

SQL Server and Eloquence appear to be the front runners in the community doing migrations. But there are curious issues. I sat through the HP Webinar on the databases. What you hear about are nothing but problems with the concept of wrapper-ing SQL Server to emulate IMAGE. The truth is that it causes you to restructure quite a bit of code, or suffer performance problems, or both.

One of our rules in this migration is "don't change a line of code in the business logic if you can avoid it." You can't do a SQL port and follow that rule. You have to make logic changes to the programs and test then all over again.

What was the AMS application written in, and where did you move it?

It's 1.5 million lines of COBOL, and we had MicroFocus COBOL in mind. But MicroFocus has run-time fees, and that means they're getting a piece of my pie [as a software vendor], and I don't like that. It's more expensive than the one I chose, Fujitsu's NetCOBOL. MicroFocus also has as its underlying tenet that it should run on many different platforms, and while that was maybe a great idea for some, we have no intentions of running on more than one platform.

MicroFocus uses intermediate code, which you can compile and take to Unix or Windows. You realize there's some baggage around that concept if you don't need it. We know that Fujitsu's product is a dark horse, but I like to see what other people are doing and then make up my own mind. So far we are going just fine with it.

What's the unique value in using Pervasive, in your view?

The underlying technology is 10 or 15 years old and is stable. It started out as an ISAM system. They've layered a SQL interface over it. If you want to use the ISAM layer, you can. PC COBOL vendors have built a bridge to that ISAM layer, and the beauty of that for us is that all the KSAM stuff we'd built automatically ended up in this product. And we found a reasonable way to emulate IMAGE in this product. The few calls we had to KSAM intrinsics were converted to corresponding calls into the Pervasive product. When the day comes to SQL-enable the product, we can do so without an across-the-board performance problem, on a program-by-program basis.

What we're getting out of this is phenomenal performance, and I'm quite certain it's faster than a SQL wrapper. While you could probably wrapper KSAM with SQL Server, good luck. It wouldn't be with standard COBOL reads and writes, and it's not likely to be a simple process. Because we have developed our own COBOL pre-compiler, we can automatically make some changes to the COBOL code when targeting the Windows platform. We have used the pre-compiler to assist with the migration to Pervasive. With Pervasive, if you own your own precompiler you can handle almost any issue that comes up.

You mentioned some distinctions in price. What did the Pervasive choice total out at for you?

They have a workgroup engine which they cap at five users, and it's around $150. I have a few clients that can live within that constraint, but not very many.

Most would need 10-30 users. On the Web I have found a 10-user license selling for $975 and a 20-user license selling for around $1,900. I haven't finished negotiations with Pervasive, but they have hinted there are steep discounts for bundled purchases. I have to get a little further along in my conversion before I can find out when my clients want to migrate. At that time I will have quantities and so will be ready to negotiate price with Pervasive.

My guess is that these prices aren't wildly different from SQL Server. But I need the functionality Pervasive offers over SQL Server. I want my KSAM and IMAGE data in the same database. Eloquence is an excellent product to replace IMAGE. But it doesn't have a mature documentation set, not as complete as you'd find in a mainstream product. Eloquence won't support KSAM, and two-thirds of our product is in KSAM.

So you think there's a connection between IMAGE and ISAM?

Depending on how you have used IMAGE, you may find that IMAGE and ISAM are closer than IMAGE and SQL. Generally you do things in IMAGE a record at a time. You walk down datasets and chains a record at a time; you also do that in ISAM. In most cases an IMAGE chain can be reasonably emulated in an ISAM file. When accessing a SQL table, the data is retrieved from the database in chunks called record sets. You are actually working with a copy of the data. This concept is not bad if you are starting from scratch, but it can create a number of problems in the areas of locking and performance if you are trying to "wrap" SQL to make it look like IMAGE.

I should point out that we analyzed our specific use of IMAGE before making the choice to migrate to the Pervasive product and proved to ourselves that this would work for us. Others may have used IMAGE in a way that would make this migration much more difficult. One important factor for us was that all of our IMAGE database calls are in COBOL copy libraries. We have been able to use the conditional compilation feature of our COBOL pre-compiler to substitute appropriate calls to the Pervasive database in place of the calls to IMAGE.

What about operating environment stability? There's always been concerns about the Windows choice in that area.

I'm a Microsoft hater, so I'll start off with that. I certainly approached it with some skepticism. But what I realized about my complaints with Microsoft's operating system stability is that Windows is its own worst enemy. You can walk up to a Windows 2000 server, grab a mouse and just start clicking around in there. There's no audit trail of what you did and no way to put it back. The user-friendly nature of the operating system lends itself to tinkering and de-stabilization.

When an individual walks up to a 3000, they are not as likely to just idly explore its utilities and tweak things. It's just not that easy. I also believe that some of the mystique of the 3000's reliability has to do with discipline. I started working with computers in an era where the server was somewhat sacred, and you don't tinker with it without some trepidation. It's my experience with Windows 2000 servers that you can create a stable environment if you treat it with the respect that you would a 3000 box. I think it is also important to avoid using the raft of new features that Microsoft throws at you constantly. They sometimes do not work perfectly.

How do you keep the users of the systems from doing damage to the system inadvertently?

I think that's an issue. We are telling our customers that our software has to be warehoused on its own Windows server. We're going to ask that they not play with it, and of course I believe we're going to be allowed to password protect the machines, so a supervisor has the password and we have the password. Today on our 3000 servers at our installations a lot of people don't even know they have a system manager password. For the most part we have established a pattern of "let AMS take care of the server." There's no doubt that a Linux box or a Unix box would have many of the unfriendly attributes that they're reputed to have, and that would keep people away from it.

How will you protect these servers from the outside world?

They're always behind firewalls. One should acknowledge that one of the main things that protects the 3000 from attack is the ignorance of the person attacking. In some ways the 3000 is not that secure. You can hack at the 3000 all day to find the password and it won't lock you out, unless you buy add-on third-party software to protect it. The simple nature of the sign on process makes it an easier target for hacking.

Windows boxes have policies you can set to lock people out after the third try. To the extent that people insist on using a Windows server that they have, they are agreeing to supply the security for it.

Microsoft puts out a lot of patches for security, and you can read that either way: they have a huge problem, or they are very responsive. It should be noted that many of the patches are not for the core Windows operating system. If you don't load a lot of things on the server that you don't need, you can mitigate the risks somewhat.

Do any of the wineries need 24x7 availability?

Not now, though we're hoping to Web-enable the application which will increase the need for uptime. I don't see anything in Windows that would critically limit our uptime. I've had Windows servers we haven't rebooted for months at a time. Sure, I know of 3000s that have not been rebooted in years, but the distinction between years and months is not important in our environment.

What do you think you've expended in resources on moving your application so far?

Well there is what it cost, and what it should have cost. It's hard to estimate what the true costs should have been. There's been some wasted motion. We've changed our minds on some things several times. I'd say our approach has been pretty expensive. We have developed quite a few tools in-house.

For example, we've written our own database locking tool, because nothing locks like IMAGE, not SQL Server, not Pervasive either.

We have written our own COBOL pre-compiler, so that individual programs need not be changed by human fingers. The most expensive development was our Formspec/Reflection replacement, which was very expensive but key to our future. This tool automatically converts Formspec form files to Microsoft Visual studio .NET DLLs, supports form families, preserves Formspec field editing specifications and produces resizable screens like Reflection has. It also allows us to add calendar controls, calculator controls and lookup boxes to our application.

To answer your question, I would say I've spent between a third and a half-million dollars on the migration. If I could do it over with what I know right now, I could probably cut that in half easily. It's a lot of money, but I don't know that it is out of line when compared to the other options. I talked to companies who claimed to be able to convert the entire system automatically. Their price tags were not all that palatable and involved on-going run time fees, too. The approach we have taken gives us the most flexibility, perhaps at a cost.

Doing the work yourself, how many people have been involved?

Three full-time in-house COBOL programmers, a COBOL contractor, a full time in-house Visual Basic programmer and a full time VB contractor. The burn rate's pretty ugly.

What's the justification to spend that money?

This move was justified on the basis of extending the life of the product, in the hopes of picking up new customers. But we don't have to pick up new customers to justify the cost.

What about replacements for the third party software your customers use on their HP 3000s?

That's the miracle of being a cheapskate, I guess. Our product doesn't have any third party stuff in it. We use only what comes with the 3000's Fundamenatal Operating System, like STORE, which we've wrappered with a menu. Most of our clients have data that fits on a 2Gb disk drive.

How have you replaced MPE’s batch processing in Windows?

There’s absolutely nothing like a real job handler quite built into Windows. We have written a job file converter which produces “DOS” command files. We have written our own job queue manager in COBOL that runs as a service under Windows. One amazing thing to watch is how we test our jobs that produce reports. When testing these we are able to request a job on the 3000 the way we always have; the job is streamed on the 3000 and at the same time it is downloaded to the Windows server, converted on the fly to a “DOS” command file and “streamed” under our Windows system. The finished reports are both delivered to the tester’s desktop for review in our Windows-based spool-file viewer (developed in-house in VB and COBOL). The Windows server always finishes long before the 3000 – no contest – not even close. Of course our HP 3000 is only a 967.

You’ve made choices in design that require some sophisticated programming. Do you think this approach would be cost-justified for a small shop?

The way I look at it is that we’re building a new sandbox. I don’t want to touch the business logic. Our HP 3000 app is going to land in a new sandbox, and work the way it always did — but it will look much nicer, run much faster and have a new life. I'm not sure it's cost-justified for a small shop with custom code to go through this process.

If I only had a single installation of this application to migrate I believe I would come up with a different solution. I might be inclined to stay on the 3000 as long as possible if the application was meeting my needs. It is possible to extend the life of software on the 3000 and improve its look and feel. For a while we were planning to offer our clients our new Windows look on the good-old 3000 but in the end that did not seem like the best business decision for us.

Now that you’re almost ready to send code out to a customer, what would you change about your design choices?

It turns out that the newest version of Fujitsu COBOL is quite capable of building compelling Windows user interfaces. This was not the case when we started out on this project. If I were starting out again I would not introduce Visual Basic into the mix. The staff I’ve had for many years has learned the WIN32 API easily. My COBOL programmers have adapted very quickly to the new technology available to them, and they’re more productive than the VB programmers. If I had to do this all over again, I’d do the whole thing in COBOL, including the front end. I know this is heresy on the Windows platform.

We also made a failed attempt to use SQL Server which wasted several man-months of time; don’t go there. Another thing I might do differently is to use Adobe Acrobat reader as our report viewing tool. We have developed our own report viewing tool which works with both 3000 spool files and those we get from our new product. The cost of this tool was very high. Although it works better than Acrobat and integrates better into our product, Acrobat is also a capable tool. Of course you must convert spool files into PDFs, but that is much simpler than producing a full-blown report viewing tool.

How much of the work is in making choices?

We spent many months working with various technologies, including even a bit of Linux tinkering. I suppose we spent between six and 12 man-months probing and playing with the options available to us. While it is exciting to play with new things, it is also a sobering experience when you realize what is at stake.

It's very frightening to pick a database product and wonder if it's going to do the job, or a COBOL compiler and wonder if it's going to do the job. What if you pick the wrong tools? The consequences of making a mistake are enormous! On the 3000 you didn't have to think about that, because you didn't have as many choices. It's kind of a high-wire act to go through this process. Now we're far enough along that I know we're going to make it. But there were times when I wasn't so sure because the number of obstacles seemed nearly insurmountable.

It has been important during this project to be flexible and innovative while staying true to the goal, which for us is migrating a valuable asset that is both proven and reliable to a new environment. I will always respect what the 3000 gave us; a dependable cost-effective platform on which to offer solutions.

However, I believe there are bright exciting things ahead waiting to be discovered on other platforms. I can say without reservation and from personal experience that a well-built and properly maintained Windows 2000 server can be very reliable, certainly more reliable than the HP 3000 I started out on in 1976.

Index