



hp e3000
solution transition
advisor

**compilers and
user interfaces**



white paper

table of contents



executive summary	1
overview	1
compilers	2
porting software written in 3GL	3
selecting a compiler	4
fourth-generation languages	5
where to go for additional information	5
vplus: the hp e3000 user interface tool	6
using vplus migration tools	7
appendix a — vplus intrinsics	10
appendix b — compilers and related tools	11
appendix c — COBOL SCREEN SECTION: code example	13

executive summary

Because of HP's strategic decision to discontinue support of its HP e3000 platform, HP recognizes its customers are faced with making significant decisions on transitioning their HP e3000 solutions to other platforms. At this time, many questions arise on how to manage this process:

- What is the impact on affected business functions?
- What are the capital and human costs associated with transition efforts, both now and in the future?
- How can risk be mitigated?
- What is technically required for transitioning all or part of the HP e3000 solution?
- What are possible transition strategies?
- What are the appropriate Independent Software Vendors (ISV) migration tools or mix of tools to use in specific components of the HP e3000 software solution?

There are a total of six white papers in the "HP e3000 Solution Transition Advisor" series. Each one addresses the subject of e3000 transitions from a different angle. The subjects of the six white papers are:

- Business Planning Guide
- Technical Planning Guide
- Compilers and User Interfaces (this paper)
- TurboIMAGE and Databases
- KSAM and the MPE File System
- MPE Commands and Networks

overview

This is the third of a series of white papers from Hewlett-Packard, providing e3000 customers with a transition process roadmap to assess, analyze, select and manage the appropriate transition strategy for their HP e3000 solutions. It is not intended to handle all HP e3000 transition issues, but a guide to:

- Programming Languages: This paper will explore various ways of porting software written in a 3GL from MPE/iX to an alternative target platform.
- We'll provide some guidelines for selecting a compiler on your target platform.
- We'll explore the alternatives that exist for porting software written in a 4th Generation Language from MPE/iX to your target platform.
- User Interfaces: This white paper will examine the migration tools that are available for porting software that uses HP's VPLUS User Interface tool from MPE/iX to alternative target platforms.

compilers



The first migration component to consider is the compiler language that the application is written in. Over the years, many different languages have been used to write applications for the HP e3000. They can be divided into two groupings:

1. **Third-Generation Languages**, such as COBOL, FORTRAN or C. Most programs written using a 3GL must be compiled before they can be executed. Because the compilers are tightly coupled to their respective platforms, most programs written in a “3GL” will require some modification before they can be compiled successfully on other platforms.
2. **Fourth-Generation Languages** such as Speedware or Powerhouse. Programs written using a 4GL can usually be ported to other platforms with little or no modification, as long as the supplier of the 4GL supports a compiler on the target platform. If you have an application that was written in a 4GL, your first step should be to contact the ISV that makes the 4GL, and find out what their migration offerings are.

At the time when HP e3000 application development was at its peak, programmers tended to use third generation languages, particularly COBOL. Consequently, the majority of HP e3000 applications were written in COBOL.

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 e3000 are not available outside of MPE/iX. This means that most COBOL programs that were written for the HP e3000 must be modified before they can be ported to another platform.

COBOL was the most popular language for HP e3000 application development. However, a number of HP e3000 applications were written using other 3GLS, including Fortran, Business Basic, C and Pascal. In most cases, the situation for these languages is similar to the situation for COBOL. Programs written in any 3GL will generally require some modification before they can be successfully ported to another platform.

porting software written in 3GL



There are two approaches to 3GL migration:

1. **Use the same language** on the target platform that your application is written in today or
2. **Translate the current language** into another one available on the target platform.

As long as the appropriate compilers and tools are available for your target platform, the simplest approach is generally to continue using the language that your application is written in today. Most of the 3GLs that have been used most often on the HP e3000 are supported on other platforms, including HP-UX, Linux and Windows. Because of platform differences, some modification to the application software may be necessary. But in many instances, there are tools available that can make these changes automatically.

Most HP e3000 applications will be ported using the same language on the target platform. However, there are some circumstances that will warrant the **translation** of the application code to a different language. For example, translation will be necessary if the language that the application is written in is not available on the target platform. Another situation that warrants translation is when the target language allows for greater deployment possibilities. Still another argument for translation is when changing the language will allow for tapping into a larger pool of potential developers.

Translating an application is very difficult to do manually. Manually translating using basic editor tools can be done but it is very time consuming. It also may require that the source application code be frozen during translation because integration of changes is very complex. Furthermore, manual translation provides many opportunities to introduce errors into the application.

Fortunately, it is rarely necessary to translate applications from one language to another manually; there are a number of translation tools available that will do the job automatically. There are two types of translation tools:

1. straight translation
2. logic extraction

The straight translation approach takes source code and simply translates the syntax of one language to another. These tools are sometimes available as freeware, making them very low cost solutions. The problem with straight translation is that this approach often generates code that is difficult to support and not very well optimized. Typically, after a straight translation tool is used, some manual rework is still required. There are tools providers that have improved on this methodology by reworking their tools and providing consulting.

The logic extraction approach begins with the application source code and extrapolates the logic and algorithms that make up the program's business logic. This base information is then used to generate new code in the target language. This methodology requires more advanced software and usually requires consulting. The code that is generated is usually more supportable and the performance of the code is better than the straight translation.

Sometimes the available translation tools do not support the language your application is written in. In this case there are service providers that have specialized tools and programming resources that can provide the translation.

selecting a compiler

If you are planning to use the same language on the target machine as the source machine there are still additional decisions that must be made. For example there are at least four Cobol compilers available on HP-UX — so if you are porting COBOL code from MPE/iX to HP-UX, you must select one of these compilers. In Appendix B of this White Paper, you'll find a list of compilers available for the target platforms.

It would be very unusual for a commercial compiler to be “buggy”, technically out of date or a poor performer. A company offering such a compiler would not be able to stay in business long. So if you have a choice, selecting a compiler vendor will require that you look at other aspects of each vendor's product.

Some key questions to ask would include:

- How close does the syntax match that of the HP e3000 compiler?
Can the vendor document the specific differences between the HP version of the language and their version?
- Does the compiler support other features such as screen handling and database access that are useful?
- Is the compiler a part of an Integrated Development Environment (IDE)?
Is an IDE desired?
- What does the compiler cost? Is there a separate cost for a run time library or component?
- Is this compiler part of a larger integrated emulation environment?
- Is the sales and support infrastructure adequate?

Once the compiler has been selected the code that ran on the HP e3000 must be modified to remove references to MPE-specific features. The following are examples of the kinds of things that must be addressed:

- Three-Level File Names On the HP e3000, file names generally have a three level structure: (FILE.GROUP.ACCOUNT). Other platforms typically use more open ended file naming conventions. The HFS feature of MPE/iX may be a helpful tool for working around MPE's three level file names.
- Special File Types. MPE provides support for a number of different file types (for example, MSG files and KSAM files.)
- References to MPE Intrinsic and checking Condition Code. Intrinsic are parts of MPE/iX that can be “called” by user applications. Typically, MPE/iX applications call intrinsic in order to do things like access databases or desktop devices, or to do certain kinds of file operations. Some intrinsic reports their success or failure using a condition code. The intrinsic are unique to MPE/iX, and so the application logic that interacts with them must be changed.
- Use of MPE parameter passing. When you execute a program on an HP e3000, you have the opportunity to pass parameters to the program to control its operation. There are three kinds of parameters supported on MPE/iX: PARMs (which are numeric values), “INFO strings” (which are strings of ASCII data), and entry points (which are alternative starting points for program execution). Parameter passing is highly platform dependent, and so program logic that makes use of these tools will need to be changed.

There are tools that can help to address the above areas. In general, the tools fall into two groupings:

1. There are tools that automatically modify the source code to take the above differences into account.
2. There are tools that emulate MPE-specific features on various target platforms, thus eliminating the need to modify the source code.

The emulator tools can simulate the MPE environment so that no manual modification is necessary. Most of the emulator tools have pre-processors that modify the code so emulation of the MPE components is supported.

fourth-generation languages

Although most HP e3000 applications were written using 3rd generation languages such as COBOL, many of them have components that were developed using a fourth generation language, (or 4GL) such as Speedware or Powerhouse.

Programs written using a 4GL can usually be ported to other platforms with little or no modification, as long as the supplier of the 4GL supports a compiler or an interpreter on the target platform. If you have an application that was written in a 4GL, your first step should be to contact the ISV that makes the 4GL, and find out what their migration offerings are.

There are some 4GLs that are no longer supported by their original manufacturers on any of the target platforms. For example, Hewlett-Packard used to offer a series of products for the e3000 collectively called "the Rapid Products". These included a 4GL called TRANSACT. The best way to port software written in Transact to other platforms is to translate it to another 4GL that is currently supported. For example, Speedware (one of HP's Platinum Partners) can translate programs that were written using TRANSACT into Speedware's own 4GL.

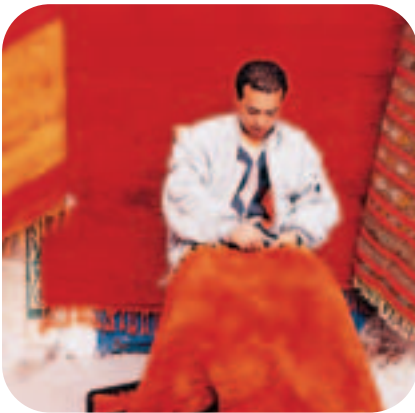


where to go for additional information

Whether you decide to port your code to the target platform in the language in which it's currently written, or whether you choose to translate it first from one language to another, you will need to select compilers and other migration tools to use in your project. For your convenience, Appendix B contains a list of the compilers and related migration tools. This list was complete as of the publication date of this white paper. However, recognizing that new tools are being developed all the time, HP has published a similar listing on the web.

The URL www.hp.com/go/e3000partners contains information about the HP e3000 transition partners program. There you'll find up-to-date lists of transition service providers and transition tools providers, as well as all the latest news on the HP e3000 transition program.

vplus: the hp e3000 user interface tool



In order to make it easy to create user interfaces (UI's) for HP e3000 applications, HP developed VPLUS for the e3000. VPLUS provides a variety of UI features that have contributed to the ease of use of the HP e3000 platform and its applications.

At the time VPLUS was designed, the term "desktop device" referred to a terminal. PCs were not yet in wide use. VPLUS required that the end user have a special kind of terminal on his or her desk. These were called **block mode** terminals. They were available from HP until the industry moved away from terminals and toward personal computers. Today, most HP e3000 users have a PC on their desktop, and they use a **block mode terminal emulator** such as WRQ's Reflection for Windows or Reflection for the Web to access their HP e3000 applications.

VPLUS applications include the following components:

- Each application includes a **formsfile**. Generated using a utility called FORMSPEC, this file contains the forms used by the application, including **processing specifications** which define editing rules that are carried out on the desktop.
- Each application program contains the application's business logic as well as CALL statements invoking VPLUS intrinsics.
- The VPLUS intrinsics are routines in MPE which manage the dialog with the block mode terminal. There is a complete list of the VPLUS intrinsics in Appendix B. A detailed explanation of each intrinsic is available in the **HP Data Entry and Forms Management System (VPLUS) Reference Manual HP 3000 MPE/iX Computer Systems Part Number: 32209-90024**. Available on www.docs.hp.com.

HP does not support VPLUS on any platform other than the HP e3000. However, there are a variety of tools that can be used to replace Vplus when migrating VPLUS applications to other platforms. In general, the tools can be divided up into the following categories:

1. Some are terminal (or terminal emulator) based. These include VPLUS emulation libraries which capture the VPLUS intrinsic calls from the application. Unlike HP's VPLUS product, most VPLUS emulation libraries do not require the use of a block mode terminal.
2. Some are client server based. These include VPLUS emulation libraries which interact with a client-based program running on the desktop (written in a language such as Visual Basic).
3. Some are web based. These include VPLUS emulation libraries which interact with a client-based web browser. Web based solutions have the advantage of using a standard, widely available client software package, (i.e. the browser) instead of a customized client-based program as in option 2 above.
4. There are screen technologies imbedded into some of the compiler offerings. For example, the AcuCOBOL language from Acucorp includes a robust user-interface facility. Migration tools are available that can automatically modify HP e3000 application source code to reference Acucorp's user interface facility instead of VPLUS.
5. There are screen solutions based on Visual Basic, Java, Cobol and XML. Some offer thin client, others offer robust fat clients and some are client-less.

using vplus migration tools



As you can see, there are a variety of tools that you can use to bring VPLUS applications over to HP-UX, Linux or Windows. Virtually all HP e3000 applications were originally written using a terminal-based application architecture. Fundamentally, there are three approaches that can be taken when porting HP e3000 application UI logic to your chosen target platform.

- You can continue to use the terminal based approach. Generally speaking, this is the simplest approach, requiring the least amount of re-programming.
- You can rewrite your HP e3000 software to use a client-server architecture. This would require that your application be split into a client program (running on the user's desktop), and a server program (running on the target server platform). This requires significant effort for re-programming.
- You can take advantage of one of the "shortcuts" to a client-server or web-based environment that are offered using one of the available HP e3000 migration tools.

No matter which of these approaches you select, it's important that the compiler and migration tools that you select support your chosen approach.

compiler based tools and application architecture

If you can afford the time and effort required, (and it could be significant), you should consider re-implementing the screens in a technology that is native to your target platform, and to the compiler that you're using there. Unlike the HP e3000's COBOL, many of the compilers which are available for the target platforms have some form of screen presentation capability built right into the compiler.

For example, Visual Basic, Visual C++, Java and some versions of COBOL have built in screen capabilities. Many versions of COBOL now support a SCREEN SECTION in the DATA DIVISION that supports PROCEDURE DIVISION screen level DISPLAYs and ACCEPTs. (see Appendix C for a COBOL SCREEN SECTION code example based on AcuCOBOL).

Using the screen capabilities of your targeted compiler may lock you into a particular UI migration approach. For example, if you choose to take advantage of Visual Basic's screen capabilities, then your software must be executed on a client system (i.e. on the user's desktop). This means splitting your application into a client component and a server component, effectively forcing you into a client/server architecture. Be sure to consider how your choice of tools will affect the architecture of your ported software.

Re-architecting HP e3000 software to use a client/server architecture as described above is a lot of work. Be sure to carefully weigh the additional benefits that will accrue from this effort against the expense. In some cases, it may make more sense to port your software while retaining its existing terminal-based architecture — at least for the time being.

The screen capabilities of many of the COBOL compilers that run on the target platforms can be effective replacements for VPLUS — keeping your application's existing terminal-based architecture. They require users to be logged on using a terminal or terminal emulator, but not necessarily a block mode terminal. If you are currently using block mode terminals, this can reduce the overall cost of your business solution significantly.

In many cases, the best approach to migrating HP e3000 UI logic is to a "shortcut" to a client/server or web-based environment. These shortcuts provide many of the benefits of these environments, without the necessity of rewriting a lot of application logic or other re-programming.

Some COBOL compiler suppliers have extended the screen capabilities of COBOL so that they can be used with a thin client program, requiring little if any additional programming on the client side. This is one way of getting to a client server architecture with very little additional coding necessary.

Another potential “shortcut” to client/server would be the use of a web-browser as a thin client program. For example, some VPLUS emulation libraries translate the VPLUS intrinsic calls into xml, which can be displayed by a web browser running on your client.

If your application currently uses VPLUS, there are a variety of tools available that can help you take “shortcuts” to client/server or to the web. These tools convert your VPLUS software to take advantage of technologies that are available to your target platform and compiler. For example, if you are using AcuCOBOL, your existing VPLUS forms (stored in FORMSFILES on the e3000) can be converted to forms that can be used by the screen capabilities of AcuCOBOL. By using the screen capabilities of AcuCOBOL, you have the advantages of using a native technology on your target platform.



Another component of VPlus replacement technology is screen maintenance. Once the VPlus formspec information is converted to the new technology, you will need tools to modify and update the screens. Each UI solution has a different way to perform this kind of maintenance. The best have “what you see is what you get” screen painters. Others require editing of code or flat files. This is another consideration when choosing a technology to display screen data; the maintenance aspect is important to the selection.

client server and security

Another thing to consider about client/server architectures is that the user never actually logs onto the server in the sense that one logs onto the HP e3000 from a terminal. The connection from client to server is made using IP protocol, which has its own security rules. Moving to a client/server architecture effectively means re-engineering your entire business solution around the IP security paradigm.

Terminal based interfaces require that a log-on session be established before the application program can be run. The session provides a dedicated output device (i.e. a terminal, or a PC running a terminal emulator) to which all user interaction I/O is sent (including VPlus). Most VPlus applications perform other user I/O besides VPlus, such as report listings and error messages, which are also routed to the user’s terminal. If the terminal is still available to the migrated program then the sections of code that don’t use VPlus I/O won’t have to change.

For web or client-server base implementations there is no terminal session. The application program is started by a server process. The only communication channel between the application running on the server, and the front-end client running on the desktop is through the VPlus intrinsics. This means that should your application contain code that expects to communicate with the desktop using other means (such as COBOL ACCEPT and DISPLAY statements), that code will need to be modified for the client-server environment.

A common “gotcha” in VPLUS migrations is the assumption that the application being migrated uses VPLUS intrinsics exclusively to communicate with the end user. Many VPLUS applications use a mixture of VPLUS and other APIs to communicate with the end user. The most common use of non-VPLUS terminal I/O in VPLUS applications is to handle errors or display reports. Here are some techniques to handle abort messages and reports:

- For program abort information a “core” dump file could be used to send error information.
- Another method to handle abort information is to use a log file where time stamped abort information is sent.
- HP used to sell terminals with integrated printers, which were typically used for short reports. In the migrated application, terminal based reports could be displayed in a separate window on the user’s PC, or they could be converted into HTML and displayed in a browser.

These additional changes require some technical knowledge and may require some consulting help.

table 1. shows a variety of terminal based and client/server based migration tools for VPLUS applications

product	comments/contacts	operating system
terminal based		
ScreenJet	Character based terminal emulator www.screenjet.com	HP-UX, Windows
WingSpan	Character based VPlus replacement. www.ordina-denkart.com	HP-UX, Windows, Linux
Transport	Part of their emulation environment. www.transport.bi-tech.com	HP-UX, Windows
AcuCOBOL	Provides terminal based interface thru their screen section. www.acucorp.com	HP-UX, Windows, Linux
ViewJ	Java based user interface that can use MPE Forms Files on the new platform. Also supports screen sections. www.legacyj.com	HP-UX, Windows, Linux
client/server		
edWin	Web and Client/Server based VPlus replacement. www.ordina-denkart.com	HP-UX, Windows
VB-View	Visual Basic based client. www.robustsystems.com	HP-UX, Windows
Intelligent Adapters	Provides Client/server functionality. www.transoft.com	HP-UX, Windows
AcuCOBOL	Provides client/server functionality using the screen section. www.acucorp.com	HP-UX, Windows, Linux
eXegete Client	Provides a full client environment to enhance the user interface. www.exegetesys.com	HP-UX, Windows
LOOKVP	VPlus to XML migration. www.cheops.fr	HP-UX, Windows
ScreenJet	VPlus to AcuCOBOL screens. www.screenjet.com	HP-UX, Windows, Linux

appendix a

vflux intrinsic



VCHANGEFIELD	Changes field attributes for specified fields at run-time.
VCLOSEBATCH	Closes batch file.
VCLOSEFORMF	Closes forms file.
VCLOSETERM	Closes terminal file.
VERRMSG	Returns message associated with error code.
VFIELDSETS	Performs field phase processing specifications.
VFINISHFORM	Performs final phase processing specified for form.
VGETBUFFER	Copies contents of data buffer into application.
VGETFIELD	Copies field contents from data buffer into application.
VGETFIELDINFO	Returns field information.
VGETFILEINFO	Returns forms file information.
VGETFORMINFO	Returns form information.
VGETKEYLABELS	Returns global or form function key labels.
VGETLANG	Returns the native language ID of the forms file being executed.
VGETNEXTFORM	Reads next form into form definition area of memory; window and data buffer are not affected.
VGETType	Copies field contents from data buffer to application, converting data to specified type.
VINITFORM	Sets data buffer to initial values for form.
VLOADFORMS	Loads the specified forms into terminal local form storage memory.
VOPENBATCH	Opens batch file for processing.
VOPENFORMF	Opens forms file for processing.
VOPENTERM	Opens terminal file for processing.
VPLACECURSOR	Positions the cursor at a specified field after a form is displayed.
VPOSTBATCH	Updates end of file mark in batch file after last record referenced.
VPRINTFORM	Prints current form and data buffer on off-line list device.
VPRINTSCREEN	Prints entire contents of screen on off-line list device.
VPUTBUFFER	Copies data from application to data buffer.
VPUTFIELD	Copies data from application to field in data buffer.
VPUJType	Copies data of specified type from application to data buffer, converting data to external format.
VPUTWINDOW	Copies message from application to window area in memory for later display
VREADBATCH	Reads record from batch file into data buffer.
VREADFIELDS	Reads input from terminal into data buffer.
VSETERROR	Sets error flag for data field in error and copies error message to window area
VSETKEYLABEL	Temporarily sets a new label for a function key.
VSETKEYLABELS	Temporarily sets new labels for function keys.
VSETLANG	Specifies the native language ID to be used with an international forms file.
VSHOWFORM	Updates terminal screen, merging the current form, any data in buffer, any key labels, and any message in window.
VUNLOADFORM	Unloads a specified form from terminal local form storage memory.
VWRITEBATCH	Writes data from data buffer to batch file.

appendix b

compilers and related tools



COBOL	product	comments/contacts	operating system
Compilers	AcuCOBOL	Supports many of the HP COBOLII/XL features. www.acucorp.com	HP-UX, Windows, Linux
	PERCobol	Supports many of the HP COBOLII/XL features. It is Java based. Runs on any platform the supports Java www.legacyj.com	HP-UX, Windows, Linux
	MFCOBOL	Used by most of the emulators www.microfocus.com	HP-UX, Windows
	NetCOBOL	Supports the .NET environment as well as others	HP-UX, Windows, Linux
Modifiers	AMXW	Has a pre-processor that's part of an emulation environment www.neartek.com	HP-UX, Windows
	Transport	Has a pre-processor that's part of an emulation environment www.transport.bi-tech.com	HP-UX, Windows
	Intelligent Adapters	Converts MPE specifics to native code www.transoft.com	HP-UX, Windows
	ViaNova 3000	Part of the Vianova conversion product which uses logic extraction www.ordina-denkart.com	HP-UX, Windows
	Trans/Port	Part of a conversion process www.adtech.com	HP-UX
Translators	ViaNova 3000	Part of the Vianova conversion product, which uses logic extraction. Multiple target languages available. www.ordina-denkart.com	HP-UX, Windows, Linux
	Cobol2C	Cobol to C converter www.orbit.com	HP-UX, Windows, Linux
Fortran	Fortran90	Fortran90 compiler which has some differences with the Fortran77 compiler on MPE/iX. www.hp.com	HP-UX
Compilers	GNU g77	Freeware Fortran 77 compiler. Part of the GNU Compiler Collection. www.gnu.org	HP-UX, Linux
Modifiers	AMXW	Part of an emulation environment. www.neartek.com	HP-UX
	ViaNova 3000	Part of the Vianova conversion product. www.ordina-denkart.com	HP-UX, Windows, Linux
Translators	ViaNova 3000	Part of the Vianova conversion product, which uses logic extraction. Multiple target languages available. www.ordina-denkart.com	HP-UX, Windows, Linux
Pascal	Pascal/UX	Will only be supported thru HP-UX 11i. It is a PA-RISC only compiler will not be ported to Itanium®. www.hp.com	HP-UX
Compilers	GNU Pascal	Freeware Pascal Compiler. www.gnu-pascal.de	HP-UX, Linux
Modifiers	GNU P2C	Freeware Pascal to C translator. www.gnu.org	HP-UX, Linux

SPL	product	comments/contacts	operating system
Compilers	SPLASH	SPL compiler that produces PA-RISC code. www.ordina-denkart.com www.allegro.com	HP-UX
Modifiers	ViaNova 3000	Part of the Vianova conversion product, which uses logic extraction . Multiple target languages available. www.ordina-denkart.com	HP-UX, Windows, Linux
	Trans/Port	SPL to C translator www.adtech.com	HP-UX, Windows, Linux
	SPLASH	SPL compiler that can generate C code. www.ordina-denkart.com www.allegro.com	HP-UX
RPG	UNIBOL 400	Have an AS/400 style RPG compiler that runs on Windows and HP-UX www.calsw.com	HP-UX, Windows
Compilers			
Modifiers	RTC	RPG to COBOL www.richter-software.com	HP-UX, Windows
C & c89	C & ac++	C & c++ compilers. www.hp.com	HP-UX
Compilers	Visual C++	Windows C++ www.microsoft.com	Windows
GCC	GCC	GNU Freeware compiler www.gnu.org	HP-UX, Linux
Compilers			
Transact	Speedware	Speedware provides migration services for Transact users. www.speedware.com	HP-UX, Windows
Translators			
BASIC	Eloquence	The Eloquence environment provides a Business Basic compiler www.marxmeier.com	HP-UX, Windows, Linux
Compilers	Visual Basic	Windows Visual Basic compiler www.microsoft.com	Windows
Cognos Powerhouse	Powerhouse	The Powerhouse 4GL is available on multiple platforms. The Axiant 4GL product can also be used to migrate. www.cognos.com	HP-UX, Windows
Speedware	Speedware	The Speedware 4GL is available on multiple platforms. www.speedware.com	HP-UX, Windows

appendix c

COBOL SCREEN SECTION

code example (courtesy of Acucorp)



```

:
:
:
data division.
file section.

working-storage section.

01 customer                                pic x(15).
01 sales-names.
    05 pic x(20) value "Marsha".
    05 pic x(20) value "Bill".
    05 pic x(20) value "George".
    05 pic x(20) value "Harry".
    05 pic x(20) value "Cindy".

01 sales-person-table redefines sales-names.
    03 sales-rep occurs num-reps times pic x(20).

01 current-rep                            pic 9 value 3.
01 rep-window                             pic x(10).
01 rep-idx                                 pic 9(2).

01 menu-selection                         pic 9(4).
   88 key-up                              value 52.
   88 key-down                            value 53.
   88 return-pressed                      value 13.
   88 exit-selected                       value 1.

screen section.

01 entry-form.
   03 sales-screen.
     04 customer-screen.
       05 "Customer name: ",line 5.
       05 using customer.
     04 sales-screen-cont.
       05 "Sales Rep.:", column 40.
       05 using sales-rep(current-rep), column 52
         before procedure set-window
         after procedure set-value.

procedure division.
level-1 section.
main-logic.

    display window erase.
    display entry-form.
    display "Press (F1) to Exit" line 24 column 1.
    perform until current-rep = 5 or exit-selected
        accept entry-form
        on exception
            accept menu-selection from escape
            end-accept

    end-perform.
    display "press to exit" line 24, column 1.
    accept omitted.
    stop run.
```

```

set-window.
  display window line (rep-line - current-rep + 1),
    column rep-column,
    size rep-win-size,
    lines num-reps,
    boxed,
    pop-up area is rep-window.

  perform varying rep-idx from 1 by 1 until rep-idx > num-reps
    display sales-rep(rep-idx) line rep-idx
  end-perform.
  perform process-rep with test after until return-pressed.
  close window rep-window.
  display sales-screen-cont.

```

```

process-rep.
  display omitted line current-rep size rep-size reverse
  accept omitted line current-rep , column 1 ,
    control key in menu-selection.
  if key-down
    display omitted line current-rep size rep-size
    add 1 to current-rep
    if current-rep > num-reps
      move num-reps to current-rep
    end-if
  end-if.

  if key-up
    display omitted line current-rep size rep-size
    subtract 1 from current-rep
    if current-rep < 1
      move 1 to current-rep
    end-if
  end-if.

```

```

set-value.
  display omitted.

```


All brand and product names are trademarks or registered trademarks of their respective companies.

notice

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Restricted Rights Legend

Use, duplication, or disclosure is subject to restrictions as set forth in contract subdivision (c)(1)(ii) of the Rights in Technical Data and Computer Software clause 52.227-FAR14.

Hewlett-Packard Company
3000 Hanover Street
Palo Alto, CA 94304, USA

© Copyright Hewlett-Packard Company 2003.
All rights reserved. Reproduction, adaptation or translation of this document is prohibited without prior written permission of Hewlett-Packard Company, except as allowed under the copyright laws.

Printed in USA 7/03
5981-3061EN rev. 1