

# TPF Software News

Volume 2, Issue 1, Spring 2000

© 2000, TPF Software, Inc.

## New local variable window first rate

Now C and C++ programmers can see local variables as they were meant to be seen. Using the new TPF/GI local variable window, programmers can read char arrays in string format, view doubles in fixed point format, view shorts, ints, and longs in decimal format, and access each individual element of an array.

The new local variable window lets all local variables be modified as well as viewed. Struct fields, class fields, and array elements can also be modified.

The local variable window marks a new era in high-level language support by TPF Software, which in 1989 was the first toolmaker in the TPF world to support debugging at the source code level.

### *The local variable window marks a new era in high-level language support in TPF/GI.*

“We’re committed to fully supporting the high-level languages that are the future of TPF application development,” said Thiru Thirupuvanam, director of TPF Software. “This is in addition to our commitment to support TPF legacy code.”

TPF/GI’s local variable window is in the

same league as similar windows in Visual C++ and other mainstream tools.

“For most mainstream tool developers (Microsoft, Computer Associates, etc.), TPF has been like a stepchild,” Thiru said. “The community is so small, they figured, ‘Why bother?’ But TPF Software is investing in TPF’s future. We’re committed to bringing the latest technology to the TPF marketplace.

“And of course, our data and program isolation are unique, so we think TPF/GI can enhance everyone’s toolbox. It’s just a matter of having all the tools you need.”

To take a tour of the new TPF/GI local variable window, see page 7. ❖

### Table of Contents

- 1 New local variable window first rate
- 1 LOC support is extended
- 2 Step Count commands debut
- 2 See ALC carriage returns
- 3 Programs can be excluded from trace
- 4 New “audit-trail” logging in TPF/GI
- 5 Program Flow visualizes trace
- 5 Worldspan edits TPFDF using the TPF/GI API
- 6 Message Substitution in GI/TERM
- 7 TPF-to-TPF Function Server coming
- 7 Tour of local variable window
- 8 More TPF Software enhancements
- 8 “Network Trace” improvements

## LOC support is extended

TPF/GI’s local 3270 terminal emulation has been “extended”: the 3270 emulator now has improved support for extended attributes, including the Start Field Extended order (SFE), the Modify Field order (MF), and the Set Attribute order (SA).

The local 3270 is a terminal protocol used by some TPF operating system customers to provide an interface to their reservation systems. Extended attributes add features such as color and underlining that make the terminal display more friendly to users.

TPF/GI’s support of extended attributes means that TPF application programmers can debug their 3270 code more thoroughly using the powerful TPF/GI environment ❖



Figure 1: The 3270 emulator supports underlining and other extended attributes.

# Step Count commands debut

TPF/GI has added new commands that allow TPF programmers unprecedented control as they step through applications at the macro and instruction level.

In the past, programmers have had several ways to step through a low-level trace: they could step to the next macro or instruction they were tracing; they could “run slow,” which means stepping at timed intervals; or they could “run no trace,” which means running to program completion while ignoring any macro and instruction trace options that were set.

## The new step commands allow finer control over low-level trace.

Two new TPF/GI commands—*Step Count* and *Step Set Count*—give programmers much finer control over how far they step in macro- and instruction-level trace.

The new *Step Count* command allows programmers to step ahead a given number of instructions or macros. The program must be stopped in non-Source View trace, and either macro trace or instruction trace must be set with the RUN option. Once the Step Count command is selected, TPF/GI will run until a specific number of instructions or macros have been executed, or until the trace stops for another reason.

### A Step Count Example

For example, if a programmer is tracing all instructions with the RUN option and all macros with the NORUN option, selecting the *Step Count* command will step forward the given number of instructions or until the first macro executes. Trace data about the instructions and macros that are executed will appear in the machine instructions panel of the ECB window. And programmers can also use the Trace>Output command to view the additional trace information in TPF/GI’s

full-featured Trace Output Viewer.

The *Step Set Count* command is similar to

the *Step Count* command, but allows programmers to first set the number of instructions or macros to step forward. ❖

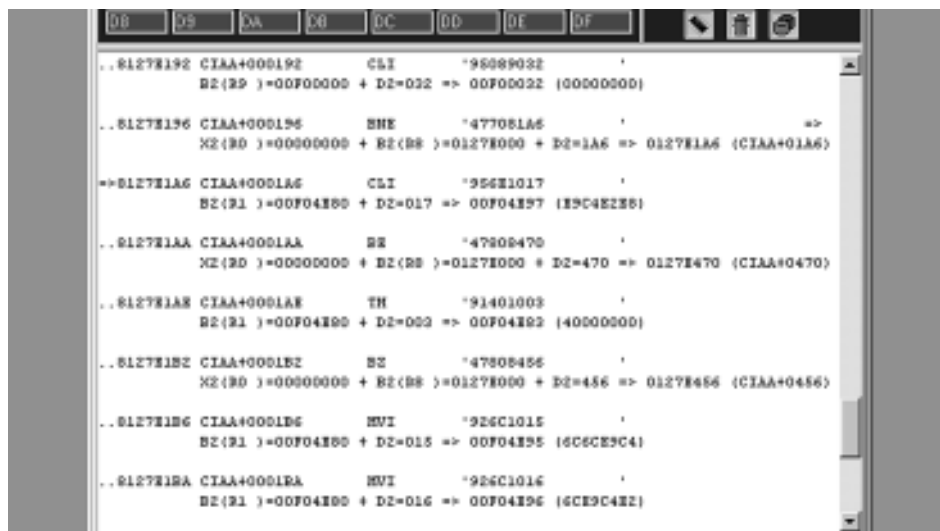


Figure 1: The ECB window now expands to show trace information for Step Count.

## See ALC carriage returns

The ALC terminal emulator in TPF/GI now has the ability to display visible carriage returns.

Why make carriage returns visible? Programmers can more readily spot errors in their ALC code if they can see where the carriage returns fall.

The visible carriage return option is turned off by default. Programmers who turn on the new feature can customize it by selecting a character, a foreground color, and a background color in which to display the carriage returns. ❖



Figure 1: TPF/GI’s ALC emulator now offers visible carriage returns.

# Programs can be excluded from trace

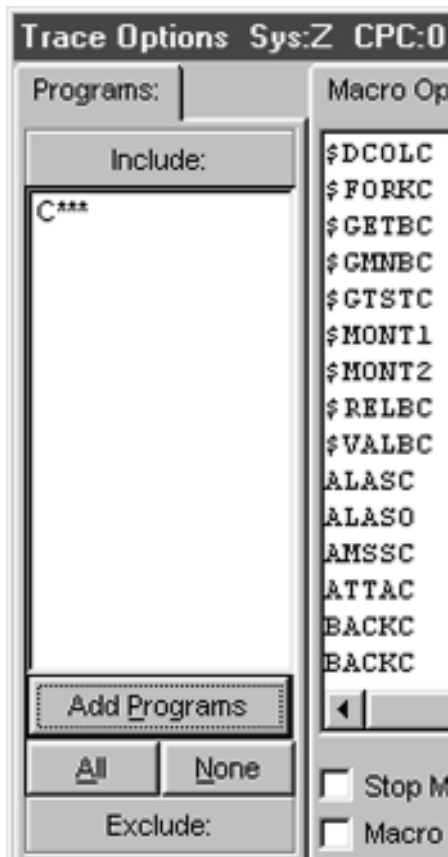


Figure 1: The Program Include tab..

programs. The new interface, designed by John Studt, fits both lists in the existing space by using “Outlook-style” tabs that look like buttons.

## Including Programs

To reach the Include list and its associated buttons, users select the tab labelled “Include.” Figure 1 shows the program area after the Include tab has been selected. In this figure, the range of all programs whose names begin with the letter “C” has been included. To include more programs, users can press the Add Programs button. To include all programs or no programs in the trace, users press the All or None buttons.

---

**Two lists were fit in the existing space by using “Outlook-style” button tabs.**

---

## Excluding Programs

To reach the Exclude list and its associated buttons, users select the tab labelled “Exclude.” Figure 2 shows the Program area after the Exclude tab has been selected. The particular program “CVAA” has been excluded; all programs beginning with “CVL” have been excluded as well. To exclude additional programs, users can select the Add Programs button. To clear the exclude list, users select the Clear List button.

Removing individual entries from either list is as easy as selecting the entry and pressing the Delete key on the keyboard.❖

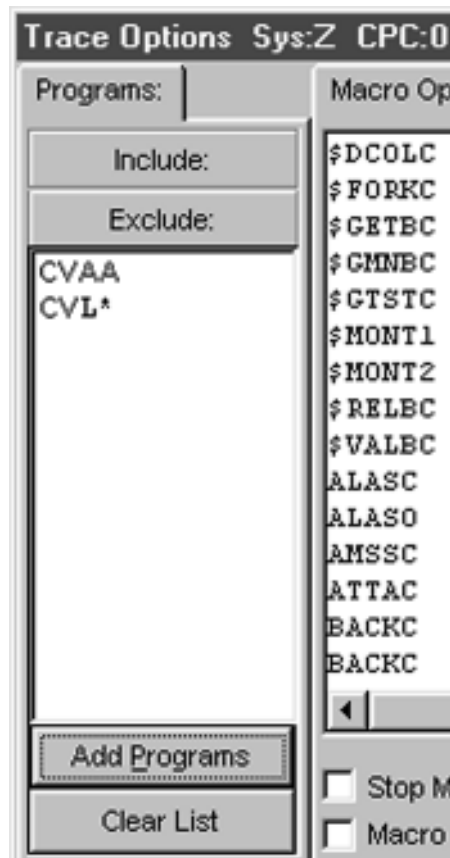


Figure 2: The Program Exclude tab..

Ninety percent of mission-critical TPF applications have been written in Assembler and are constantly being updated and modified. That’s why TPF Software is investing heavily to add new features to the assembler debugging in TPF/GI.

One recent enhancement to assembler debugging power is the ability to *exclude* programs from the macro and instruction trace.

## The Rationale Behind Excluding Programs

It has always been possible to *include* all programs in the trace, or to include subranges of programs, such as those whose names begin with a particular combination of letters. But it can sometimes be easier or more natural to *exclude* particular programs or ranges of programs. For example, a programmer may want to trace all programs except CVAA. Under the old system, he or she had no way of doing this short of individually adding all programs except CVAA to the trace.

---

**TPF/GI continues to make its macro and instruction trace more powerful.**

---

## User Interface Details

After the code was written to allow programs to be excluded from trace, the graphical user interface had to be changed to let programmers use the new feature.

Where before the Program area of the Trace Options Dialog was a simple list of programs to be traced, the new program exclude feature required that the same space accommodate two lists: one list for included programs and one for excluded

# New “audit trail” logging in TPF/GI: A picture is worth a thousand words

Some programmers who use TPF/GI need to provide an “audit trail” of their activities while debugging and testing. In order to make this record-keeping easier in a GUI environment, TPF/GI has introduced its new Visual Log.

## The Reasons for the Visual Log

In a visual environment, one of the best ways to record information is with pictures. After all, an environment such as TPF/GI uses colors and shapes to communicate information—information such as what bytes have changed, for example—and that information may be

lost if presented as text alone.

---

*In a GUI world,  
logs may need to  
be graphical too.*

---

What’s more, even when the visual information can be converted to plain text, eyeballing that information for correctness will still be easier if it is presented in the original, colorful format

of the graphical user interface.

Thinking along these lines leads to the realization that the best log in a GUI world may need to include images of the program’s windows.

## Point and Click

To make it easy for programmers to capture the images, TPF/GI has created a new tool button with a camera icon—the “camera button.”

After users have activated Visual Logging, the camera button appears. Then, when users want to capture the image of a window and place it in the Visual Log, they simply make sure the window has focus (by clicking it) and select the camera button.

The image of the window is captured and saved in a graphic file format in a folder on the user’s hard drive. (Each new log gets its own folder.) At the same time, the primary log file, in HTML format, is updated with a reference to the newly saved image.

## Reading the Visual Log

When the log file needs to be submitted, the entire folder containing the primary log file and all the image files is submitted. The Visual Log can then be read with a Web browser by double clicking the primary log file. An automatic zip option is available.

## Words Are Important, Too

The Visual Log file does not consist of pictures alone, however. For one thing, each image of a window is accompanied by the date and time that the “snapshot” was taken.

In addition, some events are placed in the log in text-only format. This includes events such as GI Console output. Users can customize Visual Logging to determine which text-only events are captured. ❖



Figure 1: TPF/GI’s new Visual Log is in HTML format and can be viewed in any web browser.

# Program Flow helps visualize trace

TPF/GI's Trace Output Viewer has added a new page—called the “Program Flow” page—that helps programmers visualize the flow of nested calls from one program to another within their applications.

## Mining an Avalanche of Useful Data

TPF/GI produces a great deal of trace information when users run a program while a macro or instruction trace is set. Each macro, instruction, store, and adstop that is being traced is reported upon.

---

*Icons and indentation illustrate the calls between programs.*

---

While users often create custom utilities to interpret, display, and evaluate this trace data, TPF/GI's Trace Output Viewer offers a lot of help. It contains a page for each kind of trace (macro, instruction, store, and adstop) as well as an “All Types” page. Clicking a page's tab causes the trace output file to be parsed and only the appropriate kind of trace information to be shown. For example, clicking the Macro tab displays only macro trace information.

In addition, the Trace Output Viewer contains a series of filters that let programmers further winnow what types of entries are displayed. For example, programmers can elect to see only trace information related to a certain program or a certain ECB ID. And there is a text search filter as well, so that programmers can elect to see only those entries that contain a certain text string.

## Showing the Flow

The new Program Flow page uses icons and indentation to display the nested calls from program to program within an application.

Like the other Trace Output Viewer

pages, the Program Flow page responds to the trace output filters and updates “live” as the user's test application runs in TPF/GI.

Thanks to Jerri Peterson for her persistence in requesting this feature. (We hope it meets her requirements.) ❖

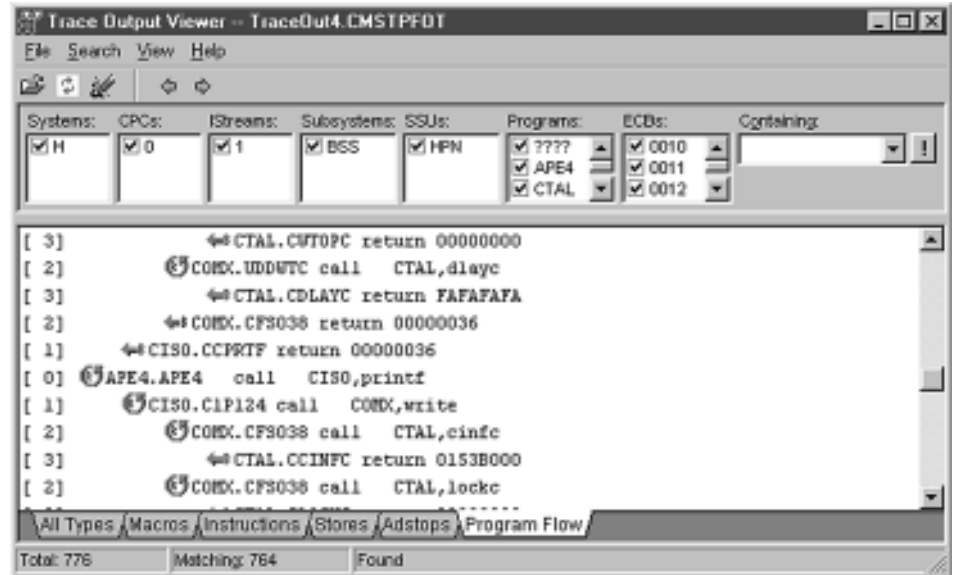


Figure 1: The Program Flow page displays the flow of calls between programs.

## Worldspan edits TPFDF using the TPF/GI API

Worldspan has harnessed the Plug-In Tools API to let programmers edit and view TPFDF records while they debug their TPF applications using TPF/GI.

TPFDF is the widely used, logical view of TPF database records pioneered by Swiss Air. The Plug-In Tools API is the Application Programming Interface that allows customers to enhance and customize TPF/GI by adding their own code and programmer tools.

In the past, customers have used a command level interface to list what DF files were open, to examine the SWOOSR record, and to find out the last action taken on the record. Worldspan's use of the API has allowed it to dynamically create DSECT panels for the DF records and to display these panels in the same block editor windows that TPF/GI uses.

The Worldspan windows are fully integrated with TPF/GI and refresh their contents automatically whenever an ECB stops or exits. The DLL that controls the DF windows, along with other supplemental windows required to display TPFDF information, were programmed by Jeff Longwell of Worldspan. ❖

# Message Substitution in GI/TERM

A versatile new message substitution feature has been added to GI/TERM. GI/TERM is the TPF Software application that provides ALC and 3270 terminal emulation for real TPF systems.

---

**GI/TERM provides ALC and 3270 terminal emulation for real TPF systems.**

---

## What is Message Substitution?

Message substitution is needed when programmers run “input files” to test their applications that work with ALC or 3270 terminals. Input files contain ALC or 3270 messages that are piped into the terminal as if a user has typed them.

Piping the messages into the terminal allows programmers to perform automated testing.

## More than Simple Messages

Input files can contain more than simple text messages, however; they can also contain “wildcard” strings.

---

**Input files can contain “wildcard” strings for which the programmer desires substitutions to be made.**

---

For example, the programmer may want to test a transaction involving a city pair, the date, and the time. Instead of hard-coding this information into her input file, she places wildcard strings which act like variables.

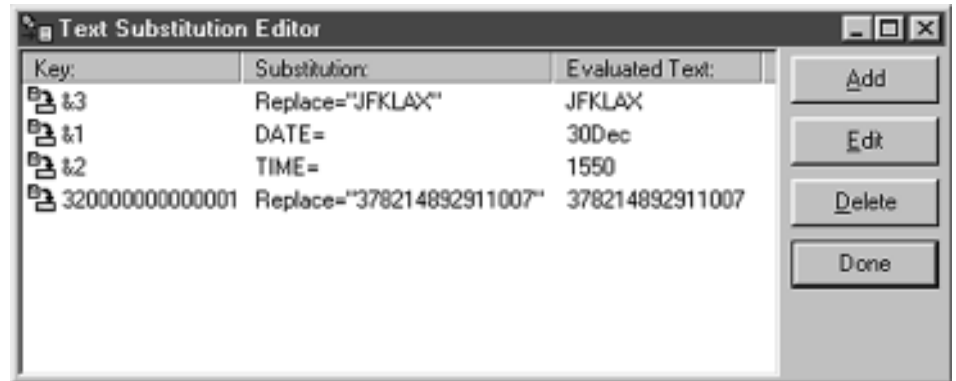


Figure 1: The Text Substitution Editor displays the wildcard substitutions that will be made in the input file.

In a first run of the input file, suppose the programmer wants to test a domestic flight. She brings up the message substitution editor and types in “JFKLAX” (Figure 2). Then, since different problems may arise with international flights, she does a second run of the input file with both a domestic and international city pair—“JFKLHR,” for instance. The dialogs in GI/TERM make it easy for her to type in the new text and rerun the test.

## Enhanced Substitution Capabilities

Currently, one TPF Software client achieves message substitution with an in-house product. The in-house product allows substitutions only for symbols consisting of an ampersand followed by a digit: &1, &2, &3, and so forth.

GI/TERM’s new Text Substitution Editor, however, allows users to substitute for symbolic strings of practically any size. Figure 1 offers the example of a credit card company that routinely places the dummy card number “3200000000000001” in its scripts and requires programmers to

replace it with a valid card number. Using the enhanced capabilities of GI/TERM’s message substitution editor, this can be automatic.

---

**Substitutions can be made for strings of any size.**

---

And consider this: Input files have to be maintained just like other code; using longer symbolic names, like using longer variable names in a C program, promotes clarity and makes maintenance easier.

## User Scripts

When GI/TERM’s message substitution feature finds a wildcard string, it can optionally call a user script to return the text. The user script can be written in Javascript, Visual Basic—or potentially even REXX with the correct third-party extension. ❖

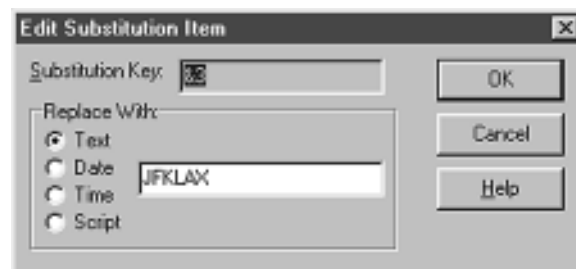


Figure 2: The Edit Substitution Item dialog allows users to change what is substituted.

# TPF-to-TPF Function Server coming

TTFS, the TPF to TPF Function Server, is currently under development and is expected to be generally available in June 2000.

TTFS is similar in concept to CTFS, which enables CMSTPF users to access resources (remote networks) that are not directly available to CMSTPF systems. TTFS similarly enables TPF users to access resources that are currently not physically available on the VPARS/TPF systems where their transactions are being tested.

## What TTFS Does

TTFS provides virtual connectivity and enables TPF programmers in test mode to access remote resources. Without TTFS, programmers must either schedule the physical resource (remote link or remote connection) or must test on the single VPARS which has network access.

**TTFS will let users access resources not physically available to their VPARS/TPF systems.**

“CTFS has been widely accepted by the CMSTPF community, and the customers have been requesting TTFS for their VPARS community,” said Luiz Maia, Director of TPF Software.

## A TTFS Example

Suppose a programmer needs to test a credit card transaction. In order to be completed, the credit card transaction

requires access to a remote system. But, currently, if the programmer’s VPARS has no physical connectivity to that remote system, the transaction can’t be completely tested without specially scheduling the resource or changing systems.

With TTFS, however, all requests for remote network access will be intercepted by TTFS, and the request will be sent through a TPF Server that has the actual remote links or remote connections. The response will be received by the TPF Server, and sent back to the originating TPF client (the VPARS).

## New Impetus

Although TTFS has been in development for some time, completing it became a top priority for TPF Software and various TPF customers thanks to the efforts of Steve Hayes at the 1999 Fall TPFUG. ❖

# Tour of Local Variable window

The new Local Variable window in TPF/GI allows C and C++ programmers to view and edit local variables.

Figure 2 (top right) shows several local variables being displayed. The variable KeepingCount is an instance of a class; the plus sign to the left of KeepingCount indicates that the variable can be expanded to show child fields. Figure 4 (bottom right) shows KeepingCount once it has been expanded by clicking the plus sign. Note that several child fields in KeepingCount also have plus signs, meaning that they are either structs, classes, or pointers. Programmers can continue expanding their view by clicking plus signs to any depth they wish.

Local variables can also be edited during debugging. Figure 3 (middle right) shows the value of c being set to 5.

When the value of a variable has been altered, by program or programmer, the row containing the variable turns red to highlight the change. ❖

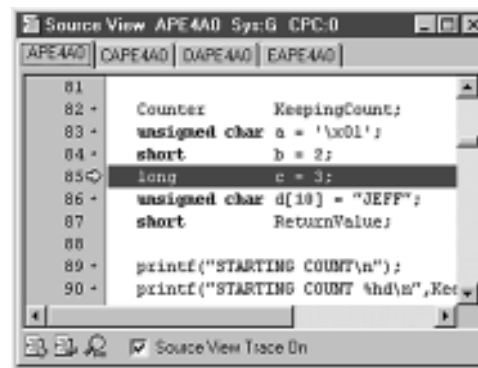


Figure 1 (above): The Source View of program APE4A0 has stopped just after the value 2 has been assigned to the local variable b.

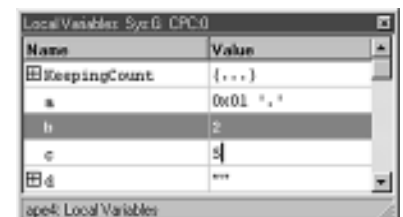
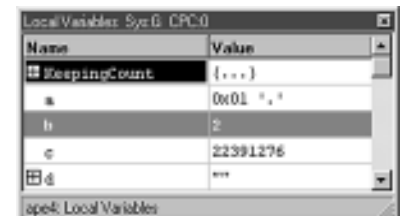


Figure 2 (top right): The entry in the Local Variables window for b has turned red to show that the value has changed.

Figure 3 (middle right): The programmer is changing the value of variable c to 5.

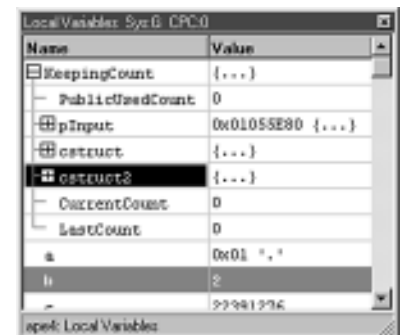


Figure 4 (bottom right): The programmer has expanded the view of the object variable named KeepingCount.

# More TPF Software enhancements

There are more enhancements to TPF Software products than can be described fully in 8 pages. Here are some summaries of additional enhancements.

## TPFAR Support

CTFS has been enhanced to include TPFAR support. Thanks to Diana and Val for all the effort. This feature is currently in Beta.

## Password Security Enhancements

To increase security, password support in TPF/GI is now handled by a centralized password object located in the APPMAN DLL. Thanks to Yakov for his continued support and work.

## Briefer ALC Terminal Log Format

The ALC terminal now logs I/O in a brief format. Previously, entire input and output screens were logged.

## MDBF Support

C++ support has been enhanced to support the Multiple Database Facility (MDBF). Thanks to Brian Williams for all his assistance in this area.

## PUT 11 Support

TPF/GI and CMSTPF now support Program Update Tape 11.

## Improved Installation Procedures

The installation procedures for TPF/GI

and GI/TERM have been improved.

Installation is simpler. The two-step administrator process of running a GIConfig program has been eliminated, and administrators can specify questions and answers for the setup dialog by changing a simple INI file.

Several changes were made to eliminate port conflicts with other PC applications. The installation now updates the services file to set up the TCP/IP connection. As a result, TPF/GI no longer requires hardcoded IP port addresses. In addition, the default TCP/IP port number was moved outside the PC dispensing range. ❖

# “Network Trace” improvements

TPF/GI and GI/TERM now offer improved capabilities for monitoring the pipeline between TPF Software’s PC applications and the host.

These improvements affect two areas: tracing network messages, and viewing network statistics.

## Tracing Network Messages

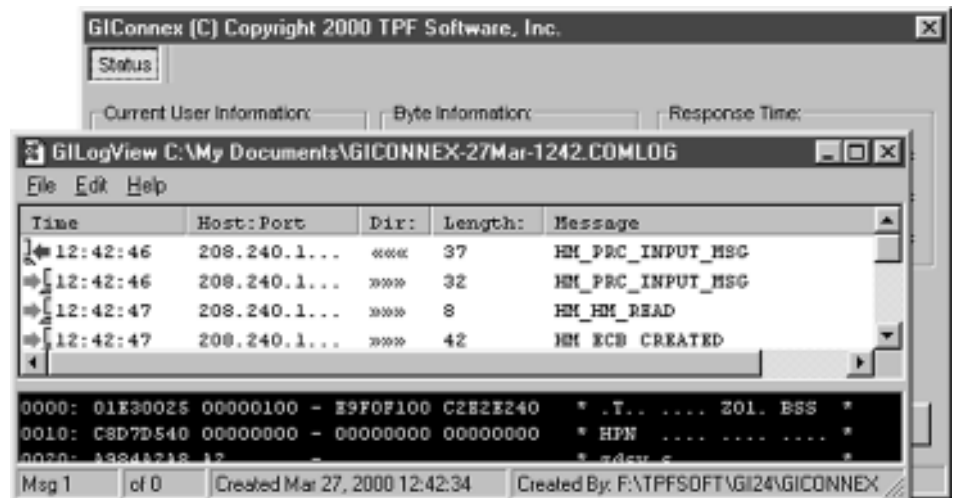
All application level TCP/IP network tracing has been consolidated for both TPF/GI and GI/TERM. Rather than implementing a proprietary, unique logging mechanism for each product, a common logging DLL has been created for all TPF Software products to allow logging of TCP/IP messages.

In addition to the logging DLL, a new executable named GILogViewer is being distributed to view these log files. GILogViewer allows users to save any given message to a PC file, and print any selected messages or all messages.

## Viewing Network Statistics

The new Network Trace also contains a new statistics dialog to let users view and report the status of their TCP/IP connection to the host.

The new network trace is available with GI/TERM and with TPF/GI 2.3.1. ❖





## Contact TPF Software

**Web Site** [www.tpf-software.com](http://www.tpf-software.com)

**Email** [info@tpf-software.com](mailto:info@tpf-software.com)

**Telephone** 919-676-5501

**Address** TPF Software, Inc.  
8729 Gleneagles Drive  
Raleigh, NC USA 27613-5419

**TPF Software News Editor:** Ed Jordan