Trap Diagnostic Facility Fact Sheet

The Trap Diagnostic Facility (TDF) from Arney Computer Systems is a revolutionary software diagnostic tool for use with the IBM z/OS operating system. TDF uses the most recent IBM hardware facilities to dynamically maintain control of the execution of the applications to be debugged. Using the hardware TRAP facility rather than using software recovery exits provides several advantages. Among them, user recovery routines can be debugged just as easily as other user-written code.

TDF does not require any code changes to the application to be debugged, eliminating the chance of errors being introduced into the code during the debugging process. The system was designed from the ground up to be used for debugging complex multi-tasking, server-oriented software environments. TDF's dual-mode debugging support provides for both interactive and non-interactive debugging.

THE TDF SERVER

The TDF Server runs in its own address space and provides support for any number of applications being debugged and any number of user interface sessions. The Server is a 64-bit z/OS application utilizing memory object technology to maintain the required data structures, trace data, user interface responses, and display data buffers above the 2 GB bar where they have no impact on the applications being debugged. On z/OS releases that support code execution above the bar, much of the TDF code is relocated there to reduce the amount of common storage used below the bar.

THE USER INTERFACE

An ISPF application provides the user interface to TDF. The ISPF session is used to perform interactive debugging tasks as well as the development of non-interactive traces. Menu-driven panels, complete with fast- path commands, can be used to perform debugging tasks or the low-level command interface can be used to directly perform many operations. Full screen operations such as type-over modifications for storage are provided as well as full screen MVS control block displays and editor panels used to list and modify various types of system information.

ROBUST PROGRAM INTERCEPT SUPPORT

TDF uses both SVC screening and PC screening methodologies to gain control at key points within a program's execution. In programs specified for intercept handling, System Breakpoints are inserted automatically at strategic points such as just prior to a program entry, before and after a new task is attached, at the definition and removal of recovery routines, and at the entry to a recovery routine. These System Breakpoints provide the opportunity for the user to perform tasks such as displaying storage, instruction tracing, or setting user breakpoints.

In addition, TDF provides the ability for the user to identify unique code requirements such as the use of shared code. When code (in either private or common storage) can be shared between multiple tasks, some being debugged and some not, TDF has the ability to create unique copies of the code for a debugged task. It can also provide "pass-through" support for tasks not being debugged so that the presence of traces or breakpoints inserted into the code do not affect the non-debugged tasks.

INTERACTIVE DEBUGGING

The ISPF user interface is used to perform interactive debugging operations on the user application. The interface is used to control the connected unit of work. Breakpoints can be inserted, storage areas and registers displayed and modified, and single instruction step tracing performed. These facilities allow detailed inspection of the program operation at the single instruction level of detail.

NON-INTERACTIVE DEBUGGING

One of the unique features of TDF is its support of a non-interactive, dynamic trace facility. This facility allows the execution of a user application at near native speed while still collecting debug information at strategic points of the execution. Up to 4,000 user-defined trace ids can be used along with trace points and map definitions to establish locations and data for data collection. The data to be collected at the trace point is defined by the associated trace id and one or more maps. The collected data along with the trace id information is asynchronously logged to an MVS data set for later viewing.

This high speed, Dynamic Trace facility can be performed on a Group of defined tasks. These tasks may reside in one or more address spaces. The set of tasks are treated as a single diagnostic unit for performing a trace. All trace points encountered by any of the tasks are logged to the same trace data set. This makes the Dynamic Trace facility very useful in resolving timing issues between related tasks, which are very difficult to find using traditional tracing methods.

DYNAMIC TRACE VS. INTERNAL TRACE

Having the ability to create dynamic traces on program code can be very useful even during the initial development of an application or software product. Today, every individual program or software product has its own internally designed and coded diagnostic facilities. Knowing the Dynamic Trace is there as an available tool, the developer can eliminate the time and effort required to design and implement internal diagnostics enabling products to be brought to market faster. In addition, the Dynamic Trace has the advantage of being able to be externally adapted to conditions that were not known at the time the original code was developed, making an existing internal diagnostic unsuitable for the newly discovered situation.

DISTRIBUTED TRACE SUPPORT

TDF provides a Trace Run-Time facility that can be distributed with a program or software product and installed along with the target code. Trace definitions can be created at the development site to gather debug data to assist in resolving a site-specific problem. The Trace definitions can be sent to the customer site and used along with the Run-Time facility to perform the Trace. When the resulting Trace data set is sent back to the development site, it can be viewed and analyzed to resolve the customer-specific problem.

OPERATING ENVIRONMENT

TDF requires z/OS version 1.9 or above operating system with an ISPF session running under TSO. No other software or specific hardware is required. The Server must be run as a started task or batch job.

TRY TDF IN YOUR INSTALLATION

TDF is available for local installation and testing at no charge. You are encouraged to use it before deciding if it meets your requirements. If it fails to completely please you, remove it from your computer system and pay nothing. The trial test system and documentation can be obtained from our Web site.

Arney Computer Systems
P.O. Box 382511
Duncanville, Texas 75138

Voice: 214-306-0754
Email: info@arneycomputer.com
Web Site: www.zosdebug.com

TDF
Trap Diagnostic Facility