Sun™ Mainframe Transaction Processing Software Supplement for Debugging Online Programs

Sun Microsystems, Inc.
www.sun.com

Part No. 817-5679-10
January 2004, Revision A

Submit comments about this document at: http://www.sun.com/hwdocs/feedback

Contents

Preface

This document describes how to debug online transactions in the Sun™ Mainframe Transaction Processing software environment. It replaces Chapter 14, "Debugging Online Programs," in the *Sun Mainframe Transaction Processing Software Developer's Guide*. "Error Messages" on page 18 contains the new and changed error messages associated with the enhancements made to the Debug Facility.

Shell Prompts

Shell	Prompt
C shell	*machine-name*%
C shell superuser	*machine-name*#
Bourne shell and Korn shell	$
Bourne shell and Korn shell superuser	#

Typographic Conventions

Typeface*	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`
`AaBbCc123`	What you type, when contrasted with on-screen computer output	`% su` `Password:`
AaBbCc123	Book titles, new words or terms, words to be emphasized. Replace command-line variables with real names or values.	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* be superuser to do this. To delete a file, type `rm` *filename*.

* The settings on your browser might differ from these settings.

Accessing Sun Documentation

You can view, print, or purchase a broad selection of Sun documentation, including localized versions, at:

http://www.sun.com/documentation

Contacting Sun Technical Support

If you have technical questions about this product that are not answered in this document, go to:

http://www.sun.com/service/contacting

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by going to:

http://www.sun.com/hwdocs/feedback

Please include the title and part number of your document with your feedback:

Sun Mainframe Transaction Processing Software Supplement for Debugging Online Programs, part number 817-5679-10

Debugging Online Programs

Sun MTP has its own Debug Facility for command-level debugging. It also supports COBOL, C language, and Liant Open PL/I source-level debuggers. The topics in this chapter are:

Refer to the *Sun Mainframe Transaction Processing Software Developer's Guide* for information about debugging batch programs.

Using the Debug Facility

The Sun MTP Debug Facility monitors `EXEC CICS` statements in COBOL, C, and PL/I programs. You can use the Debug Facility to debug transactions on a local system or on a remote system from which a transaction is routed.

▼ To Invoke the Debug Facility

1. **Start a region.**
2. **Type the `CEDF` transaction on a blank transaction screen.**
   - You can use `CEDF` with options to debug COBOL programs. See "`CEDF` Options" on page 2.
   - You can use the `CEDF` transaction alone.

3. **When the Set Breakpoint screen shown in FIGURE 1 is displayed, set the command-level debug options, or you can select a source-level debugger.**

4. **Press Enter.**

   After you set the options, they are in effect for any transactions started at the terminal.

5. **Type a transaction to start debugging.**

   See "To Set Breakpoints During a Debug Session" on page 7.

CEDF Options

Note – In the C or PL/I environment, do not use CEDF with options. Enter the CEDF transaction alone.

The format of the CEDF transaction is:

CEDF [*trmid*|*sysid*][,ON|,OFF|,ANIM|,RANIM]

CEDF[,ON|,OFF|,ANIM|,RANIM]

Option	Description
trmid	Four-character terminal identifier for any terminal, local or remote, logged in to the region. This is the EIBTERMID. A transaction submitted from this terminal can be debugged in the window where the Debug Facility is invoked. The Animator, C, and CodeWatch debuggers do not operate with this option over intersystem communications connections (TCP/IP and SNA).
sysid	Four-character identifier that matches the SysId field in the TCT– System Entries screen. When *sysid* is specified, any LU6.2 attach request received from this remote system is debugged.
,ON	Initializes command-level debugging. If you only enter CEDF,ON, debugging defaults to the terminal that invoked the CEDF transaction.
,OFF	,OFF: Terminates all debugging options.
,ANIM	,ANIM: Initializes the COBOL Animator. See "Using Animator" on page 10.
,RANIM	,RANIM: Initializes the Remote COBOL Animator. See "Using Remote Animator" on page 11.

Note – If you are using *trmid* or *sysid* in the CEDF command, type a space between CEDF and the *trmid* or *sysid*. Otherwise, do not type a space between CEDF and the comma.

CEDF enables debugging for single or multiple terminal sessions.

You can use CEDF *trmid* to debug an inbound transaction-routed transaction, if a terminal definition for the remote terminal exists in the region. Refer to the *Sun Mainframe Transaction Processing Software Administrator's Guide* for more information about ISC shippable terminals.

For all other inbound ISC requests, use CEDF *sysid*. If another attach is received on the connection after CEDF is invoked on a terminal, it is not queued for debug. However, if multiple debug transactions occur for the same terminal, they are queued.

When debugging transactions that execute a START request, the invoking transaction is processed first, then the started transaction is processed.

Note – The transaction CEDF CPRT is not valid.

▼ To Select a Debugging Mode

1. **Type CEDF with no options on a blank transaction screen.**
2. **When the Set Breakpoint screen is displayed (FIGURE 1), select a debugging mode:**
   - Command-level debugging (top five options on the Set Breakpoint screen).

     Sun MTP debugs EXEC CICS commands based on your settings. Your COBOL, C, or, PL/I source code is not debugged.
   - Source-level debugging:
     - Invoke the COBOL Animator.
     - Invoke the COBOL Remote Animator.
     - Invoke the C source debugger.
     - Invoke CodeWatch, the PL/I source debugger.
3. **Provide a value in the DISPLAY field, under the following conditions:**
   - If you are using Animator from a TN3270 or IBM 3270 terminal
   - If you are using Animator from a local terminal client and want to perform debugging in a separate window
   - If you are using the C source debugger
   - If you are using the CodeWatch PL/I debugger

Note – The DISPLAY field is automatically populated from the `DISPLAY` variable of the shell where you started the local terminal client. If `$DISPLAY` is not set in your shell, the DISPLAY field is blank.

See TABLE 1 for a description of the DISPLAY field.

4. **Press Enter.**

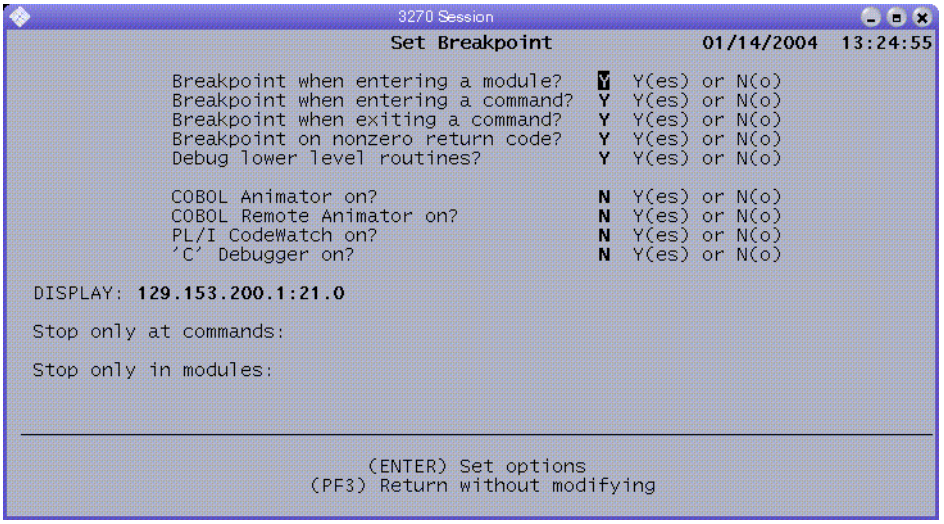

FIGURE 1 Debug Facility—Set Breakpoint Screen

The options on the Set Breakpoint screen are described in the following table.

TABLE 1 Set Breakpoint Screen Options

Option	Description	Valid Values
Breakpoint when entering a module	Stops program execution when a new object module is loaded	Y: Enable breakpoint N: Disable breakpoint
Breakpoint when entering a command	Stops execution and returns to the Debug Facility main menu just before any CICS command is executed.	Y: Enable breakpoint N: Disable breakpoint
Breakpoint when exiting a command	Stops execution and returns to the Debug Facility main menu when any CICS command is exited.	Y: Enable breakpoint N: Disable breakpoint
Breakpoint on nonzero return code	Halts execution of the current program when the return code from the CICS command is nonzero, indicating an error condition.	Y: Enable breakpoint N: Disable breakpoint

TABLE 1 Set Breakpoint Screen Options *(Continued)*

Option	Description	Valid Values
Debug lower level routines	Specifies whether to debug CICS programs that are entered by an `EXEC CICS LINK` command	`Y`: Enter the Breakpoint Processor whenever one of the breakpoint conditions is true for the program entered by the `LINK` command. `N`: Breakpoint Processor ignores any breakpoint conditions in the program entered by the `LINK` command.
COBOL Animator on	Starts Animator when you execute COBOL transactions. For an explanation of the options available in Animator, refer to the *Micro Focus Animator Operating Guide*. See "Using Animator" on page 10. This option is set or reset only at the start of a transaction. To turn Animator off during a transaction, you must use the `Zoom` Animator option or type `Esc+Y`. When Animator is set to `Y`, turn off the other debug options.	`Y`: Start Animator `N`: Stop Animator
COBOL Remote Animator on	Starts Remote Animator when a COBOL transaction is started. For an explanation of the options available in Remote Animator, refer to the Micro Focus Remote Animator documentation. To debug using Remote Animator, see "Using Remote Animator" on page 11. When Remote Animator is set to `Y`, turn off the other debug options.	`Y`: Start Remote Animator `N`: Stop Remote Animator
PL/I CodeWatch on	Starts the PL/I CodeWatch debugger when you execute Liant Open PL/I transactions. See "Debugging PL/I Programs" on page 15 and refer to the Liant PL/I *CodeWatch Reference Manual*. When PL/I CodeWatch is set to `Y`, turn off the other debug options.	`Y`: Enable CodeWatch `N`: Disable CodeWatch
C Debugger on	Starts the C source debugger when you execute C transactions. See "Debugging C Language Programs" on page 13. When C Debugger is set to `Y`, turn off the other debug options.	`Y`: Enable the C debugger `N`: Disable the C debugger

TABLE 1 Set Breakpoint Screen Options *(Continued)*

Option	Description	Valid Values

DISPLAY	Indicates where the debugger's screen will be displayed.
	• When using Animator, if this field is blank (`$DISPLAY` is not set), Animator is displayed in the local client window. To display it in a different window, type a value in this field.
	• Animator running on a TN3270 client or an IBM 3270 device requires a value in this field.
	• CodeWatch and C debuggers running on any terminal require a value in this field.
		Valid values are similar to the following:
		`myhost:32.0`
		`123.45.678.900:32.0`

Stop only at commands	Specify up to three CICS commands, such as `SEND MAP`, for the system to control breakpoints. Applies only if you enabled one or more of the options on the upper half of the screen. If no commands are entered, program execution stops at all commands based on the five options set on the upper half of the screen. If one or more commands are set, only these commands trigger the Breakpoint Processor.	CICS commands
Stop only in modules	Specify up to five programs for the system to control breakpoints. Applies only if you enabled one or more of the options on the upper half of the screen. If all five program names are blank, program execution stops at commands in all programs based on the five options set in the upper half of the screen. If one or more program names are entered, only commands in these programs trigger the Breakpoint Processor. If you are using Animator, Remote Animator, CodeWatch, or the C debugger, this option is ignored.	Program names

▼ To Set Breakpoints During a Debug Session

1. **Type `CEDF` before executing a transaction or press PF10 when the transaction is stopped by a previous breakpoint request.**

   The Set Breakpoint screen is displayed.

2. **Type in the values for the breakpoints and press Enter.**

   The system either returns a blank screen so that you can type a transaction identifier, or the Debug Facility main screen is displayed and stops whenever one of the specified conditions occurs.

Debug Processor Main Screen

The Debug Processor main screen is displayed after you set the debug options and type a program name or transaction. The Debug Processor main screen has three areas:

- The Execution Interface Block (EIB) display area. Refer to the *Sun Mainframe Transaction Processing Software Developer's Guide* for a description of each of the fields in the EIB area.
- The CICS command and main storage address area.
- The PF key interpretation area.

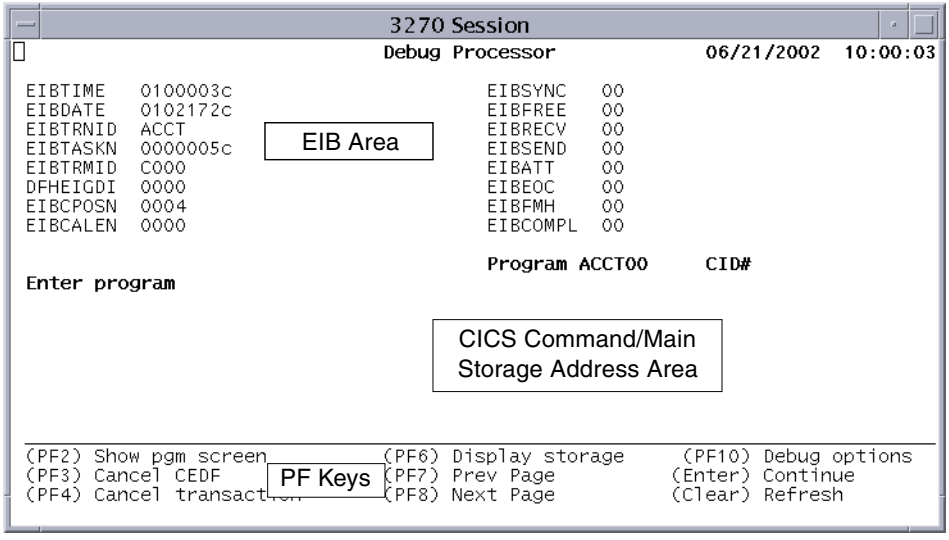

FIGURE 2 Debug Processor—Main Screen

The following function keys are active on the Debug Processor main screen:

Function Key	Action
PF2	Displays the terminal screen as shown to the user at this point in the program. Press any Attention Key, such as the PF2 key, to return to the Debug Processor screen.
PF3	Cancels `CEDF` and allows the program to finish executing. No confirmation prompt is displayed.
PF4	Displays the Cancel Transaction screen, which prompts for verification to cancel the program. Press Enter to cancel. Press PF3 to return to the Debug Processor screen.
PF6	Displays the Main Storage screen. See "To Display Main Storage" on page 8.
PF7	Displays the previous page of the EIB display area.
PF8	Displays the next page of the EIB display area.
PF10	Displays the Set Breakpoint screen. See TABLE 1 for descriptions of the breakpoints.
Clear	Redisplays the Debug Processor screen.
Enter	Allows program execution to continue until the next breakpoint is encountered. The debugger continues program execution until a breakpoint is reached or until the end of the current transaction.

▼ To Display Main Storage

- **When the debugger stops on an `EXEC CICS` command, press PF6 on the Debug Processor screen.**

  The system displays hexadecimal and character representations of 256-byte pages. The storage display initially defaults to the starting address of the current program.

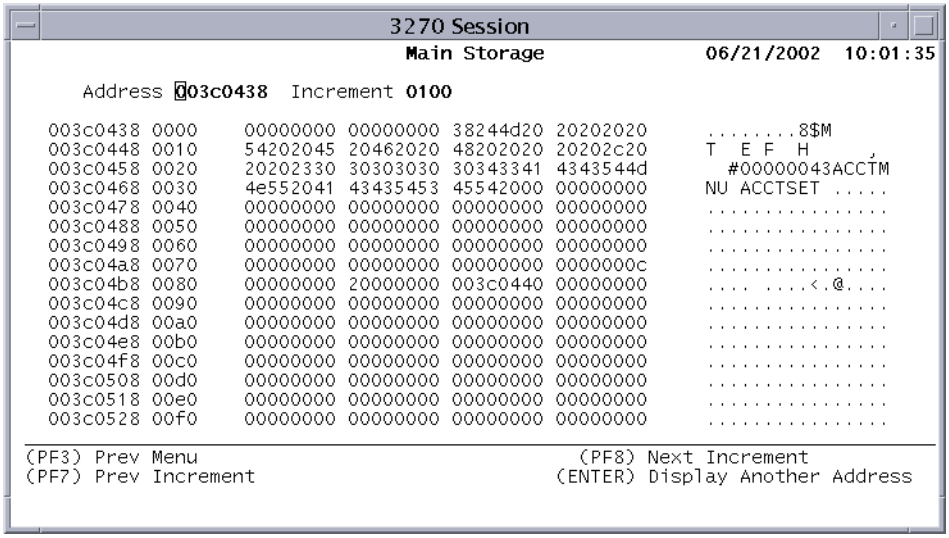

FIGURE 3 Debug Processor—Main Storage Screen

You can modify the following fields:

Address

Address of the first byte in the display area. This field is automatically updated when the area of memory is changed by the actions of any of the active function keys (PF7, PF8, or Enter).

Increment

Number of bytes, in hexadecimal format, by which the display changes when a page forward or page back occurs. If the number for the increment is set greater than 100 (hexadecimal), the successive displays omit some intervening bytes.

The following function keys are active on this screen:

Function Key	Action
PF3	Returns to the Debug Processor main screen.
PF7	Pages backward by the increment specified in the Increment field.
PF8	Pages forward by the increment specified in the Increment field.
Enter	Applies any changes that were made to the Address and Increment fields. If the Address is changed, the area of the memory buffer is changed to display the desired address.

▼ To Turn Off Debugging

- **Use one of the following methods:**
  - Type the `CEDF,OFF` transaction on a blank transaction screen.
  - Press PF3 on the Debug Processor main screen (FIGURE 2).
- **Or**

1. **Type `CEDF` on a blank transaction screen.**
2. **When the Set Breakpoint screen is displayed, disable each option by typing an `N`.**
3. **Press Enter to accept the changes and return to a blank transaction screen.**

Debugging COBOL Programs

The Micro Focus Animator debugger is used to debug COBOL programs in the Sun MTP environment.

Before using Animator with Sun MTP, you must be familiar with the basics of COBOL animation. Refer to the Micro Focus Animator documentation for this information.

Note – Remote Animator enables you to debug programs from 3270 terminals. See "Using Remote Animator" on page 11.

If you attempt to invoke Animator on a region whose transaction server was built without COBOL support, the following message is displayed:

```
Animator cannot be invoked on a non-COBOL system
```

Using Animator

Animator works on X terminals and ASCII character devices. Animator enables debugging of multiple terminal sessions.

If you do not provide a value in the DISPLAY field on the Set Breakpoint screen, Animator is displayed in the same terminal window where the transactions you are debugging execute. Because both Animator and CICS online screens are displayed in the same window, the Animator screen can be overlaid by CICS input or output

commands (for example, `SEND` and `SEND MAP`), or the CICS online screen can be overlaid by the Animator screen. You can toggle between the Animator screen and the CICS screen using the F2 (PF2) key.

▼ To Debug Programs Using Animator

1. **Start a region.**
2. **Connect to the region using a terminal.**
3. **Start Animator using one of the following methods:**
   - `CEDF,ANIM`. See "`CEDF` Options" on page 2.
   - `CEDF` with no options. On the Set Breakpoint screen, set the option COBOL Animator On to `Y`. If you want Animator to be displayed in its own window, also type a value in the DISPLAY field.
   - Manually, just before or at a breakpoint in program execution.
4. **Press Enter.**

   Your terminal is now in debug mode.
5. **Type a transaction to start debugging.**
6. **Debug the programs.**
7. **When you are finished, turn off Animator.**

   See "To Turn Off Debugging" on page 10.

Using Remote Animator

The Remote Animator feature of Sun MTP can improve developer productivity by integrating with the Net Express IDE (integrated development environment). It enables you to debug Sun MTP COBOL applications from the Net Express environment through transactions submitted on any terminal in the network. It removes the restriction that required you to use the local client (`unikixl`) terminal to debug COBOL applications.

Net Express runs on personal computers running Microsoft Windows operating systems. Refer to the Micro Focus documentation for more information about the Net Express IDE and remote animation.

▼ To Debug Programs Using Remote Animator

1. **Set the following environment variables in the region's setup file or at the shell prompt:**

   a. **Set** `KIXREMANIMPORT` **to the desired port number.**

      Use a value from 1025 through 65,535.

   b. **Set** `KIXREMANIMTOUT` **to the timeout value in seconds.**

      This value is the time the region will wait for a cross-session Animator server before returning an error. The default is 5 seconds.

2. **Review the region setup file to ensure that:**

   a. **The** `COBDIR` **environment variable is set to the directory where Server Express is installed.**

   b. `$COBDIR/lib` **is included in the** `LD_LIBRARY_PATH` **environment variable.**

   c. `$COBDIR/bin` **is included in the** `PATH` **environment variable.**

3. **Make sure that the COBOL source files are the same on the host system and the personal computer.**

   a. **Build the project in Net Express, which will create** `.int` **and** `.idy` **files.**

   b. **Make a note of the path name where the Net Express project is located.**

      You need this information in Step 9.

4. **In a new window on the host system, start the debug server process by typing the command:**

   ```
   $ animserv32 AUTO PORT=nnnn
   ```

   where *nnnn* is the value specified in the `KIXREMANIMPORT` environment variable.

   Note – This process must run in the foreground.

5. **Start a region.**

6. **Connect to the region from the personal computer using TN3270 emulator software, such as Sun MTP J3270.**

7. **In the emulator window, start Remote Animator on the host system using one of the following methods:**

   - `CEDF,RANIM`

- `CEDF` *termid*`,RANIM`
- `CEDF` with no options, then set the COBOL Remote Animator On option to `Y` on the Set Breakpoint screen

8. **When the Remote Animation activation message is displayed, make a note of the termid, host name, and port number.**
9. **Start Remote Animator on the personal computer.**
   - If you did not connect to the region using the Sun MTP J3270 terminal emulator, you must use the following command in a DOS window on the personal computer. The command must all be on one line.

```
C:\> MFNETX /DEBUGSERVER /DEBUGPROJ:Net-Express-project-path /DEBUGID:termid
/DEBUGMACHINE:machine-name /DEBUGPORT:port-number
```

   - If you used the Sun MTP J3270 terminal emulator to connect to the region, the command is executed automatically.
10. **On a blank transaction screen in the emulator window, type a transaction to start debugging.**
11. **When the Remote Animator window is displayed, you can debug using the available options.**
12. **When you finish debugging, turn off Remote Animator.**

    See "To Turn Off Debugging" on page 10.

Debugging C Language Programs

Sun MTP supports the debugging of C language programs by providing an interface to the DBX symbolic debugger. Refer to the debugger documentation for information about using the debugger.

▼ To Debug Programs

1. **Compile the programs with the `-g` option on the compile command.**
2. **Build the shared objects.**

   Refer to the *Sun Mainframe Transaction Processing Software Developer's Guide* for information about building C shared objects.

3. **Define the C transactions in the PCT and PPT.**

4. **Create the debugger initialization file and save it in your home directory.**

   Refer to your debugger documentation for information about the initialization file. The following example shows the required code in the `.dbxrc` initialization file for the Solaris™ DBX debugger.

   ```
   use /.../.../src/CICS_structures
   ```

5. **Start the region.**

6. **Connect to the region using a terminal client.**

7. **On a blank transaction screen, type the `CEDF` transaction.**

8. **On the Set Breakpoint screen, set the C Debugger On option to `Y`, and set all other options to `N`.**

9. **In the DISPLAY field, type a value for the terminal client.**

10. **Press Enter.**

    Your terminal is now is debug mode.

11. **Type the C transaction you want to debug.**

    The debugger is automatically started and attaches itself to the transaction server.

12. **When the debugger window is displayed, type the debugger command that displays the source code where the debugger attached to the transaction server.**

    The code should be similar to:

    ```
    ...
    while(1) {} i = i;
    return;
    ```

    The actual stop/attach point is the `while(1)` instruction.

13. **You can now type commands in the debug window to set breakpoints in the C application code and continue debugging.**

    Refer to the debugger documentation for the correct commands.

    When an `EXEC CICS RETURN TRANSID` or an `EXEC CICS RETURN` (or its logical equivalent) is encountered, the current debug session is de-instantiated automatically and a new debugger session in instantiated when the transaction server continues the transaction process. This de-instantiate/re-instantiate procedure is necessary because the debugger cannot follow the application code across process ID (PID) boundaries from transaction server to transaction server.

14. **When you are finished, turn off debugging.**

    See "To Turn Off Debugging" on page 10.

Debugging PL/I Programs

CodeWatch is the Liant Open PL/I source debugger. It enables you to debug PL/I applications in a multiuser, multitransaction server environment.

In a multiuser environment, some users might be using CodeWatch while others are not. The number of simultaneous CodeWatch users is limited only by the number of active copies of the CodeWatch process that can be present on the system. Simultaneous users who are not running CodeWatch can run transactions with no interference.

When running in CodeWatch mode, the `STARTCW` program (`startcw.so` shared-object) becomes the first module of every PL/I application program that processes the initiation of any transaction. It triggers the CodeWatch process and waits for attachment. When the CodeWatch window is displayed, you only need to enter commands or breakpoints or type `C` to continue with the debug process.

Note – To use the CodeWatch debugger, make sure that the directory containing the `unikixtran` executable in use is placed before `$UNIKIX/bin` in the `PATH` variable. For example, if `$UNIKIX/local/bin` contains the `unikixtran` executable, it must be the first entry in the search path.

CodeWatch provides a graphical user interface (GUI) for debugging. If you want to use the GUI, you must set the variable `CWSTARTD_COMMAND` to a valid command before starting the region. Refer to the *CodeWatch Reference Manual* documentation for information about setting this variable.

▼ To Set Up CodeWatch

1. **Make your program code compliant so that you can debug it in the CodeWatch environment.**

   Refer to the Liant Open PL/I *Language Reference Guide*, *User's Guide*, and the *CodeWatch Reference Guide* for instructions.

2. **Set the** `CODEWATCH_STBPATH` **and** `CODEWATCH_SRCPATH` **environment variables and ensure that they include** `$UNIKIX/lib` **in the search path.**

   Refer to the *CodeWatch Reference Manual* for a complete description of these environment variables.

3. **If you want to use the CodeWatch GUI, set the** `CWSTARTD_COMMAND` **variable.**

4. **Create the CodeWatch setup file.**

   The commands in this file are used to put the region in CodeWatch debugging mode. Your CodeWatch setup file must contain the commands shown in the following `startcw.cwf` file:

```
shlib /aaaa/bbbb/..../lib/startcw.so
env STARTCW
b innerloop
c
goto loopexit
```

Note – The first line of this example indicates that you must provide the full path name of the `startcw.so` object.

   The `startcw.so` shared library is located in the `$UNIKIX/lib` directory, and an entry for it exists in the PPT. These two items are automatically set when you build the Sun MTP software.

5. **Set the** `CODEWATCH_INIT` **environment variable to a file that contains the CodeWatch commands to put the region into CodeWatch mode.**

   For example:

```
CODEWATCH_INIT=$KIXSYS/startcw.cwf
```

▼ To Debug Programs

1. **Start the region.**

2. **Connect to the region with a terminal.**

3. **On a blank transaction screen, type the** `CEDF` **transaction.**

4. **On the Set Breakpoint screen set the PL/I CodeWatch On option to** `Y`**, and set all other options to** `N`**.**

5. **In the DISPLAY field, type a value for the terminal.**

6. **Press Enter.**

   Your terminal is now in debug mode.

7. **Type the transaction to be debugged.**

   Just before the PL/I application code begins processing, a CodeWatch process is automatically invoked and attaches to the current transaction server, as explained in "To Set Up CodeWatch" on page 15.

8. **To debug your programs after the CodeWatch environment is invoked, use either method:**

   - Type the CodeWatch debug commands. Refer to the *Liant CodeWatch Reference Manual*.
   - Create a command file that can be run from the debug environment.

     For example, to debug the `ACCT` Primer transaction program in the CodeWatch environment, type the following command at the CodeWatch prompt:

```
read acct00.cwf
```

     This command tells CodeWatch to read the debug commands from the `acct00.cwf` file, which contains the following:

```
shlib /pkgs/mtp/MTP8.0.0/primer/pl1/prog/acct00.so
env ACCT00
b ACCT00%ENTRY
c
p12
```

Note – In this example, the full path name of the `acct00.so` shared object is the path that is defined in the PPT.

9. **When the transaction ends, CodeWatch automatically terminates and the CodeWatch window is dismissed.**

10. **When the same or a different transaction server picks up the next transaction, a new CodeWatch process is started.**

    This sequence continues until you are finished debugging.

11. **When you are finished, turn off debugging mode.**

    See "To Turn Off Debugging" on page 10.

Error Messages

New error messages:

`KIX0440E COBOL Animator is not supported on a TR terminal`

Description: You cannot debug a COBOL program with Animator on a remote system. The `CEDF` transaction is running on a transaction routed terminal.

Solution: You must animate your COBOL program on the local region.

`KIX0441E PL/I CodeWatch is not supported on a TR terminal`

Description: You cannot debug a Liant PL/I program with CodeWatch on a remote system. The `CEDF` transaction is running on a transaction routed terminal.

Solution: You must debug your PL/I program on the local region.

`KIX0442E C debugger is not supported on a TR terminal`

Description: You cannot debug a C program with a C debugger on a remote system. The `CEDF` transaction is running on a transaction routed terminal.

Solution: You must debug your C program on the local region.

`KIX0444E Enter a DISPLAY value to Animate on this 3270 type terminal`

Description: To invoke Animator from a TN3270 terminal client, you must specify a value in the DISPLAY field on the Set Breakpoint screen of the Debug Facility. This is the value of the X windows `DISPLAY` environment variable where you want the Animator window to display.

Solution: Enter the value of the `DISPLAY` environment variable on the Set Breakpoint screen of the Debug Facility; for example, `host1:32.0`.

`KIX0445E Please enter a DISPLAY value`

Description: To invoke either the Liant PL/I CodeWatch debugger, or the default C debugger, you must specify a value in the DISPLAY field on the Set Breakpoint screen of the Debug Facility. This is the value of the X windows `DISPLAY` environment variable where you want the debugger window to display.

Solution: Enter the value of the `DISPLAY` environment variable on the Set Breakpoint screen of the Debug Facility; for example, `host1:32.0`.

Changed error messages:

`KIX0455E CEDF ,ANIM is not supported on a 3270 type terminal`

Description: You cannot invoke the COBOL Animator with the `CEDF ,ANIM` on a 3720 EBCDIC-type terminal.

Solution: Run the `CEDF` transaction without any parameters to display the Debug Facility's Set Breakpoint screen. Type `Y` in the COBOL Animator On field, and make sure there is a valid value in the DISPLAY field.

`KIX0470E CEDF ,ANIM is not supported on a TR terminal`

Description: You cannot debug a COBOL program with Animator on a remote system. The `CEDF` transaction was executed on a transaction routed terminal.

Solution: You must animate your COBOL program on the local region.