



This is the tutorial for IBM Debug Tool for z/OS[®], one of the IBM zSeries[®] problem determination tools.

Scenarios for starting the debugger for LE batch programs

- Trigger the debugger with an LE TEST option in JCL, and
 1. Display the debugger on a graphical user interface
 2. Display the debugger on a TIM (Debug Tool terminal interface manager) terminal through a session manager
 3. Display the debugger on a dedicated TIM terminal
 4. Display the debugger on a dedicated non-TIM terminal
- Trigger the debugger with the LE 'user exit data set' facility, and
 5. Display the debugger on a graphical user interface
 6. Display the debugger on a TIM terminal through a session manager
 7. Display the debugger on a dedicated TIM terminal
 8. Display the debugger on a dedicated non-TIM terminal

Running and debugging an LE batch program under TSO

- Use the 'Debug Tool setup file' online panels to run the program and display the debugger on the TSO terminal

In this section, you will see a scenario for starting Debug Tool for a batch application. In this scenario the application runs as a batch job, and a TEST option is coded in the JCL that will trigger the debugger when the job runs. The debugger displays on a Debug Tool terminal interface manager terminal session that you connect to through a session manager. You can skip this section if you do not plan to use Debug Tool this way on your system.

Debug in batch using the 'user exit data set' facility and a dedicated Debug Tool TIM terminal



- **Description**
 - A session manager is used to get a Debug Tool TIM (terminal interface manager) terminal session
 - The LE options data set is updated from the TIM session to set a trigger for the debugger
 - A special library may be required in STEPLIB or JOBLIB to enable the Debug Tool LE user exit
 - The debugger displays on the TIM terminal session
- **This method can be used:**
 - To debug LE programs running in batch jobs
 - including programs that access IMS™, DB2®, or other types of databases
 - and non-LE programs that run in the call chain under an LE program
 - If the Debug Tool TIM feature is installed on your system
- **When not to use this method:**
 - If the TIM is not installed, use a GUI or dedicated non-TIM terminal instead

3

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

This is a commonly used, simple method used to start the debugger. You use your session manager to get a Debug Tool TIM (terminal interface manager) session. From that session, you set a trigger that will start the debugger when the application runs, by updating a special file called the 'LE options data set'. The program is submitted to batch, and the debugger is displayed on the TIM session.

This method can be used to debug Language Environment (LE) conforming programs running in a batch job, including programs that access DB2, IMS, and other types of databases. Even non-LE subroutines can be debugged using this method, if there is at least one LE program higher in the call chain.

If Debug Tool terminal interface manager sessions have not been installed on your system's network, consider using a non-TIM terminal or a graphical user interface instead, which are described in other sections.

Select Debug Tool from the session manager menu



```
Session Manager                Software University Main Menu                17/12/2010 10:28:16
LU  TCP00055                    DNET074
Sel  Description                APPLID  Status  ACB      LOGMODE  NODE
-----
- PF1  TSO session 1            TSOA    ACTIVE  ISZ004   D4A32792
- PF2  TSO session 2            TSOB    AVAIL
- PF3  CICS AOR1 - CICS A      CICSACB1 AVAIL
- PF4  CICS AOR2 - CICS B      CICSACB2 AVAIL
- PF5  CICS AOR3 - CICS C      CICSACB3 AVAIL
- PF6  CICS AOR4 - CICS D      CICSACB4 AVAIL
- PF7  CICS AOR5 - CICS E      CICSACB5 AVAIL
- PF8  DEBUG TOOL              EQASESSM AVAIL
- PF9  IMS A                    IMSACB   UNAVAIL
- PF10 IMS B                    IMSACB2  AVAIL
- PF23 Disc from Session Manager  CMD
- PF24 Logoff from Session Manager  CMD
- 810 System Management Menu      AVAIL

HARDCOPY = /H
====> |

ESC: Menu @@  Bwd <<  Fwd >>  Cut *c  Paste *p  Push />  Enter
```

Your session manager will look different than this one, and will have different menu options

Note: If Debug Tool is not already defined on your menu, you may be able to add a session definition that connects to application EQASESSM

4 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

In this scenario, terminal session manager software is used to switch between TSO and a Debug Tool session. A terminal session manager may be installed on your system to give users an easy way to access different applications or systems from a single terminal. There are different terminal session manager products available. This example uses one called IBM session manager. There may be a different one on your system, so your session manager will probably look different than the one shown here, and the menu options will be different. Even so, most session managers provide a menu of systems that can be selected, or a special "hotkey" you can use to switch between sessions.

In this example the user is already logged on to TSO. If your session manager shows the debugger as one of the options, select it. If your menu does not have an option for the debugger, depending on your authorizations, you may be able to define a new session for it. If you define your own session, try connecting to an APPL with a name of EQASESSM, which is the default network name for the Debug Tool application. If your session manager does not give you a way to connect to Debug Tool, consider contacting your help desk or the Debug Tool installer so that the session manager can be configured correctly.

Log on to the Debug Tool TIM session



```
DEBUG TOOL TERMINAL INTERFACE MANAGER
Terminal TRMDT007

EQAY001I Session manager escape ==>
EQAY001I Enter your userid: TSS16
EQAY001I Password: █
```

Log on to Debug Tool with your User ID and password

Enter


PF3=EXIT

5 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

That opened a terminal session for the debugger. Notice the title "Debug Tool terminal interface manager" on the Debug Tool terminal, which indicates that it is a TIM terminal. If your Debug Tool terminal does not have this title, then you either have not correctly connected to it, or you have a non-TIM terminal session. If the TIM feature is not installed on your system, take one of the tutorial sections that describes how to debug using a non-TIM terminal, instead of this section.

Next, log on to with your ID and password. Typically, this is the same ID and password you use to log onto TSO. Do not be confused by thinking that you are logging onto TSO twice. You are not logging onto TSO here, you are logging on to Debug Tool. The ID and password are typed in, and enter is pressed.

The Debug Tool TIM terminal is ready
Edit your LE options data set




DEBUG TOOL TERMINAL INTERFACE MANAGER

EQAY001I Session manager escape ==> █

EQAY001I Terminal TRMDT007 connected for user TSS16

EQAY001I Ready for Debug Tool



PF3=EXIT PF10=Edit LE options data set PF12=LOGOFF

6	IBM Debug Tool for z/OS tutorial	© 2012 IBM Corporation
----------	----------------------------------	------------------------

At this point your Debug Tool terminal is ready to receive a debugging session. The next step is to set a trigger that will start the debugger when the application runs. Press the F10 key to edit the LE options data set.

Specify the name of the Debug Tool user exit data set



DEBUG TOOL TERMINAL INTERFACE MANAGER
* Specify TEST Run-time Option Data Set *

Specify the name of a TEST run-time option data set that you want to create or edit.

Data Set Name . . . 'TSS16.DBGTOOL.EQUAUOPTS'
Volume Serial . . . █ (If not cataloged)

Specify the correct name of your user option data set. There is a specific naming convention that must be used on each system.

Use the default file name unless instructed otherwise, or if a different user ID is needed

Enter

PF1=Help ENTER=Process PF12=Cancel

7 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

You will store your debugging profile in a file. The file name must follow a specific convention. By default, it is yourID.DBGTOOL.EQUAUOPTS.

If you have not used this panel before, it will automatically fill in a file name in the 'data set name' field. Use the default name unless you have been instructed otherwise. The naming convention may have been customized for your system, and the convention must be followed exactly for this feature to work. Later, when you run your batch job, a Debug Tool system exit will look for a file according to the convention.

Press Enter. If the file does not already exist, it will be created automatically.

If the run-time option data set does not already exist, this allocation panel is displayed



```
DEBUG TOOL TERMINAL INTERFACE MANAGER
* Allocate TEST Run-time Option Data Set *
Data set name: 'TSS16.DBGT00L.EQAUOPTS'

New Setup File Dataset Allocation Parameters
Management class . . . . .      (Blank for default management class)
Storage class . . . . .          (Blank for default storage class)
Volume serial . . . . .         (Blank for system default volume) **
Device type . . . . .           (Generic unit or device address) **
Data class . . . . .            (Blank for default data class)
Space units . . . . . BLKS    (BLKS, TRKS, CYLS, KB, MB, BYTES
                                or RECORDS)
Average record unit . . . . .   (M, K, or U)
Primary quantity . . . 20     (In above units)
Secondary quantity . . . 20   (In above units)
Directory blocks . . . 0      (Zero for sequential data set)
Record format . . . . . FB    (FB or VB)
Record length . . . . . 80
Block size . . . . . 32720
Data set name type . . . . .

( ** Only one of these fields may be specified)

PF1=Help  ENTER=Create file  PF12=Cancel
```



8

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

In this example, the file did not exist, so the file allocation panel is displayed where you can specify attributes for the new file. On many systems, you do not need to change anything and you can take the defaults. Press Enter to continue.

A panel is displayed where you specify your debugging profile



```
DEBUG TOOL TERMINAL INTERFACE MANAGER
* Edit TEST Run-time Option Data Set *
Enter test program names:

Program name(s) ==>   ==>   ==>   ==>  
                ==>   ==>   ==>   ==>  

Select Debug Tool options

Test Option      ==> TEST      Test/Notest
Test Level       ==> ALL       All/Error/None
Commands File    ==>  
Prompt Level     ==> PROMPT
Preferences File ==>  

EQAOPTS File     ==>  

Other run-time options:
==>  

Current debug display information:
VTAM%TSS16

PF1=Help PF4=Save PF9=Change Display Info PF12=CANCEL
```

9

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

Now the run-time option data set has been created, and the panel titled: 'Edit test run-time option data set' is displayed, where you can specify your debugging profile.

Specify the name of one or more main programs that will trigger the debugger when they run



```
DEBUG TOOL TERMINAL INTERFACE MANAGER
* Edit TEST Run-time Option Data Set *
Enter test program names:
Program name(s) ==> SAM1 ==> ABC* ==>
Select Debug Tool options
Test Option      ==> TEST      Test/Notest
Test Level       ==> ALL       All/Error/None
Commands File    ==>
Prompt Level     ==> PROMPT
Preferences File ==>
EQAOPTS File     ==>
Other run-time options:
==>
Current debug display information:
VTAM%TSS16
PF1=Help PF4=Save PF9=Change Display Info PF12=CANCEL
```

Specify the names of **main** programs

* = wildcard
Note: * can be used by itself

F9

10

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

Specify the name or names of the main programs that should trigger the debugger when they run. Only the highest level LE program, typically the main program, will trigger the debugger. There is no reason to specify names of lower level programs or subprograms that you want to debug. In fact, entering subprogram names on this panel will not have any effect at all. After the debugger starts, you will be able to enter commands to tell the debugger to stop at any subprogram you want to see.

For any individual batch job step, there is only one main program. However, you can use the same profile to trigger different jobs by specifying more than one main program name. You can specify an asterisk as a wildcard as the last character in the program name. If you specify an asterisk as the full name by itself, any time your debugging profile is active, the highest level LE program will trigger the debugger, regardless of its name.

Change the 'test option' field to NOTEST to turn the debugging profile off, or leave it as TEST if you want the debugger to trigger. You can specify the name of a file that contains a debugging script in either the 'commands file' or 'preferences file' fields. A script is a file that contains a series of debugger commands that you want to run automatically when the debugger starts.

Notice the line near the bottom of the panel that reads 'current debug display information'. In this example, under that line is the setting 'VTAM%TSS16', which indicates that the debugger will display on a TIM terminal identified by user ID TSS16. If you want to change where the debugger will display, press F9.

You can control where the debugger is displayed



DEBUG TOOL TERMINAL INTERFACE MANAGER
* Change session type *

Select (/) a session type and provide parameters:

Full-screen mode using the Debug Tool Terminal Interface Manager

User ID ==> TSS16 User ID

Remote debug mode

Address ==> _____

Port ==> _____

PF1=Help PF4=Save PF12=Cancel

11 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

The 'change session type' panel gives you options to control where the debugger will be displayed when it triggers. You can send it to your TIM terminal by leaving the defaults: the 'Full-screen mode' option is selected and your ID is in the 'user ID' field. You can work with a colleague who has logged on to their own Debug Tool terminal. Enter their ID, and the debugger will display on their TIM terminal when it starts.

You can optionally specify that the debugger should display on a graphical user interface by selecting 'remote debug mode', and the IP address and port number of your workstation. This option assumes that you have software installed on your workstation that provides the graphical user interface, such as the Debug Tool plug-in for CICS Explorer®.

Press F4 to save your changes and return.

Save your profile settings



```
DEBUG TOOL TERMINAL INTERFACE MANAGER
* Edit TEST Run-time Option Data Set *
Enter test program names:

Program name(s) ==> SAM1 ==> ABC* ==>      ==>
                ==>      ==>      ==>      ==>

Select Debug Tool options

Test Option      ==> TEST Test/Notest
Test Level       ==> ALL  All/Error/None
Commands File    ==>
Prompt Level     ==> PROMPT
Preferences File ==>

EQAOPTS File     ==>

Other run-time options:
==>

Current debug display information:
VTAM%TSS16

PF1=Help PF4=Save PF9=Change Display Info PF12=CANCEL
```



12

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

That returns to the 'edit test run-time option data set' panel. Press F4 again to save your profile.

The debugging profile was saved in your run-time options data set



```
DEBUG TOOL TERMINAL INTERFACE MANAGER

EQAY001I Session manager escape ==> █

EQAY001I Terminal TRMDT007 connected for user TSS16
EQAY001I Ready for Debug Tool

PF3=EXIT PF10=Edit LE options data set PF12=LOGOFF
```

Hot key to your session manager


At this point your profile has been saved, and your Debug Tool terminal is ready to receive a debugging session. Next the TSO terminal session will be selected. Typically, there is a special "hot key" you can press to jump to the session manager menu, such as a specific PA key or a key combination. Press whatever hot key or key combination you use to re-display your session manager menu.

Select the TSO session



```
Session Manager          Software University Main Menu          17/12/2010 09:03:19
LU   TCP00055          DNET074
Sel  Description          APPLID  Status  ACB      LOGMODE  NODE
-----
S PF1 TSO session 1          TSOA    ACTIVE  ISZ004   D4A32792
- PF2 TSO session 2          TSOB    AVAIL
- PF3 CICS AOR1 - CICS A    CICSACB1 AVAIL
- PF4 CICS AOR2 - CICS B    CICSACB2 AVAIL
- PF5 CICS AOR3 - CICS C    CICSACB3 AVAIL
- PF6 CICS AOR4 - CICS D    CICSACB4 AVAIL
- PF7 CICS AOR5 - CICS E    CICSACB5 AVAIL
- PF8 DEBUG TOOL          EQASESSM ACTIVE
- PF9 IMS A              IMSACB  UNAVAIL
- PF10 IMS B             IMSACB2 AVAIL
- PF23 Disc from Session Manager  CMD
- PF24 Logoff from Session Manager  CMD
- 810 System Management Menu      AVAIL

HARDCOPY = /H

====> _____ 
ESC: Menu @@   Bwd <<   Fwd >>   Cut *c   Paste *p   Push />
```

14

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

Select the TSO session.

Open the JCL that will be used to run the batch program

```
Menu Utilities Compilers Options Status Help
OS/390 Primary Option Menu
Option ==> 2
0 Settings Terminal and user parameters User ID . . : TSS16
1 View Display source data or listings Time. . . : 09:53
2 Edit Create or change source data Terminal. : 3278A
3 Utilities Perform utility functions Screen. . : 1
4 Foreground Interactive language processing Language. : ENGLISH
5 Batch Submit job for language processing Appl ID . : ISR
6 Command Enter TSO or Workstation commands TSO logon : TPROC02
7 Dialog Test Perform dialog testing TSO prefix: TSS16
8 LM Facility Library administrator functions System ID : STLABF6
9 IBM Products IBM program development products MVS acct. : *
10 SCLM SW Configuration Library Manager Release . : ISPF 6.1
11 Workplace ISPF Object/Action Workplace
12 OS/390 System OS/390 system programmer applications
13 OS/390 User OS/390 user applications
DB DataBase Interactive Database Functions
G SWS/STL Display SWS/STL Selection Panel
Enter X to Terminate using log/list defaults
```



15


IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

In TSO, open the JCL that runs the application you want to debug. In this example, the ISPF editor is selected.

Open the JCL that will be used to run the batch program 

```
Menu RefList RefMode Utilities Workstation Help
Edit Entry Panel
Command ==> _____
ISPF Library:
Project . . . TSS16
Group . . . ADLAB
Type . . . JCL
Member . . . XSAM (Blank or pattern for member selection list)
Other Partitioned, Sequential or VSAM Data Set, or z/OS UNIX file:
Name . . . _____ +
Volume Serial _____ (If not cataloged)
Workstation File:
File Name . . . _____
Options
Initial Macro . . . _____ - Confirm Cancel/Move/Replace
Profile Name . . . _____ - Mixed Mode
Format Name . . . _____ - Edit on Workstation
Data Set Password . . . _____ - Preserve VB record length
Record Length . . . _____
```



16

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

Open the JCL in the editor.

Submit the JCL that will run the program in batch



The screenshot shows a terminal window with the following content:

```
File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT TSS16 ADLAB.JCL (XSAM) - 01.17 Columns 00001 00072
Command ==> SUBMIT Scroll ==> CSR
***** Top of Data *****
000001 //TSS16D JOB (ACCTG), 'IBM TOOLS WORKSHOP', REGION=4M, CLASS=A,
000002 // MSGCLASS=H, NOTIFY=&SYSUID, MSGLEVEL=(1,1)
000003 //*
000004 //PRINT1 EXEC PGM=IDCAMS
000005 //SYSPRINT DD SYSOUT=*
000006 //FILE DD DSN=&SYSUID..ADLAB.FILES(CUST
000007 //SYSIN DD *
000008 PRINT INFILE(FILE) COUNT(1)
000009 //*
000010 //RUNSAM1 EXEC PGM=SAM1, REGION=4M
000011 //STEPLIB DD DISP=SHR, DSN=&SYSUID..ADLAB.LOAD
000012 // DD DISP=SHR, DSN=DEBUG.TOOL.BATCH.EXITLIB
000013 //***** OPTIONAL DD'S FOR DEBUG TOOL *****
000014 /** //CEE0PTS DD *
000015 /** TEST(,,VTAM%TSS16:
000016 /** //INSPLOG DD SYSO
000017 /** //EQADEBUG DD DSN=
000018 /** // DD DSN=
000019 /** //INSPREF DD DSN=
```

Annotations in the screenshot include:

- A red circle around the `SUBMIT` command.
- A yellow box labeled `Enter` with an arrow pointing to the `DD DISP=SHR, DSN=DEBUG.TOOL.BATCH.EXITLIB` line.
- A callout box: "If needed, add a special library to STEPLIB or JOBLIB".
- A callout box: "Depending on how your program was linked, you may need to add a special Debug Tool library to STEPLIB or JOBLIB. The name of the library will be provided by your system programmer."

Footer information: 17 | IBM Debug Tool for z/OS tutorial | © 2012 IBM Corporation

Here is the JCL that will run the program. On some systems, you may need to add a special Debug Tool library to the STEPLIB or JOBLIB DD. This library enables the system exit that reads the 'User exit data set' to determine whether to start the debugger. On other systems, this library is not needed. If it is needed, get the name from your system programmer or help desk.

Submit the job.

The job was submitted



```
IKJ56250I JOB TSS16D (JOB09303) SUBMITTED  
*** █
```

Hot key to your session manager

18 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

When a job step runs where the debugging profile is active and the highest level LE program matches one of the program names in the profile, the debugger will trigger. In this example, it will display on the Debug Tool TIM terminal. Press whatever hot key or key combination is needed to display the session manager.

Select the Debug Tool session



```
Session Manager          Software University Main Menu          17/12/2010 12:01:15
LU  TCP00055                    DNET074
Sel  Description                APPLID  Status  ACB      LOGMODE  NODE
-----
- PF1  TSO session 1            TSOA    ACTIVE  ISZ004   D4A32792
- PF2  TSO session 2            TSOB    AVAIL
- PF3  CICS AOR1 - CICS A       CICSACB1 AVAIL
- PF4  CICS AOR2 - CICS B       CICSACB2 AVAIL
- PF5  CICS AOR3 - CICS C       CICSACB3 AVAIL
- PF6  CICS AOR4 - CICS D       CICSACB4 AVAIL
- PF7  CICS AOR5 - CICS E       CICSACB5 AVAIL
- S PF8  DEBUG TOOL            EQASESSM ACTIVE
- PF9  IMS A                    IMSACB  UNAVAIL
- PF10 IMS B                    IMSACB2 AVAIL
- PF23 Disc from Session Manager  CMD
- PF24 Logoff from Session Manager  CMD
- 810 System Management Menu      AVAIL

HARDCOPY = /H
====> |
ESC: Menu @@  Bwd <<  Fwd >>  Cut *c  Paste *p  Push />
```



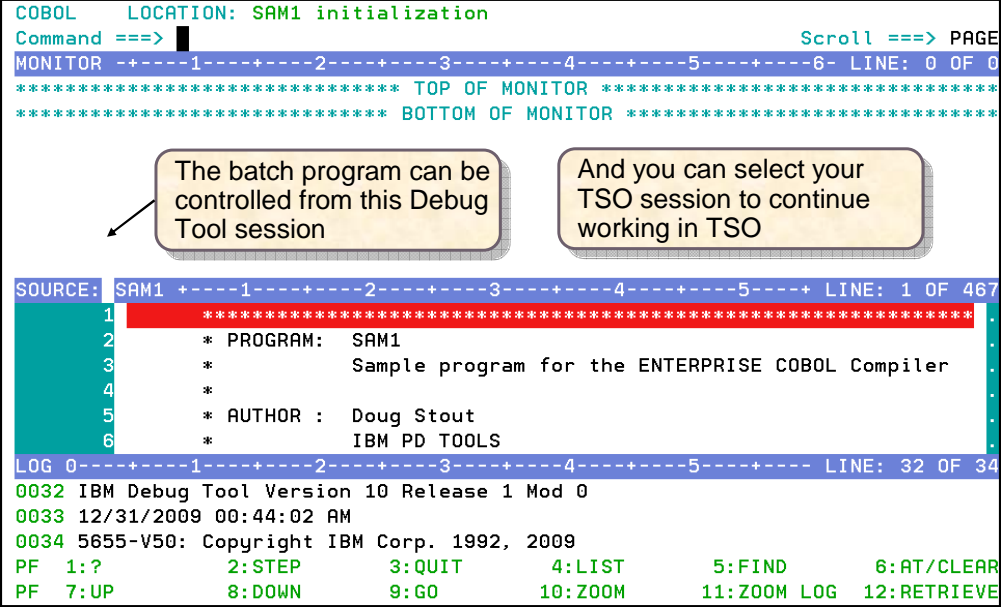
19

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

In the session manager, select the Debug Tool session.

The debugger automatically displays on the TIM terminal



The screenshot displays the IBM Debug Tool interface. At the top, it shows 'COBOL LOCATION: SAM1 initialization' and a 'Command ==>' prompt. Below this is a 'MONITOR' window with a header 'MONITOR --+---1---+---2---+---3---+---4---+---5---+---6- LINE: 0 OF 0' and content '***** TOP OF MONITOR *****' and '***** BOTTOM OF MONITOR *****'. A callout box points to the 'MONITOR' header with the text 'The batch program can be controlled from this Debug Tool session'. To the right, another callout box contains the text 'And you can select your TSO session to continue working in TSO'. Below the monitor is a 'SOURCE:' window showing the source code for 'SAM1' with line numbers 1 through 6. The code includes comments: '* PROGRAM: SAM1', '* Sample program for the ENTERPRISE COBOL Compiler', '* AUTHOR : Doug Stout', and '* IBM PD TOOLS'. Below the source code is a 'LOG' window with a header 'LOG 0--++--1--++--2--++--3--++--4--++--5--++-- LINE: 32 OF 34' and log messages: '0032 IBM Debug Tool Version 10 Release 1 Mod 0', '0033 12/31/2009 00:44:02 AM', and '0034 5655-V50: Copyright IBM Corp. 1992, 2009'. At the bottom of the log window, there is a list of function keys: 'PF 1:?', '2:STEP', '3:QUIT', '4:LIST', '5:FIND', '6:AT/CLEAR', '7:UP', '8:DOWN', '9:GO', '10:ZOOM', '11:ZOOM LOG', and '12:RETRIEVE'. The footer of the screenshot contains the page number '20', the text 'IBM Debug Tool for z/OS tutorial', and the copyright notice '© 2012 IBM Corporation'.

The batch job is running on the host z/OS system, and Debug Tool is communicating with the Debug Tool terminal. You can control the program from the Debug Tool terminal, and you can also select your TSO session to continue working in TSO at the same time. Be aware that the TSO session is no longer needed. You could log off from TSO and continue to debug, because Debug Tool is controlling the application that is running in its own batch address space.

How the Debug Tool user exit data set facility works



How it works:

- The user exit data set utility saves your debugging profile in a 'user exit data set'
 - By default, the naming convention is: user-ID.DBGTOOL.EQUAUMPTS
 - The system programmer can customize the naming convention on your system. If that is the case:
 - You need to learn the naming convention
 - Ensure that you name your user exit data set correctly
- Debug Tool provides special LE exits. If an exit is enabled when the program runs:
 - It uses the user ID to determine the name of your data set
 - The exit reads your user option data set
 - If the main program name matches any of the programs in the profile, the exit passes a TEST option to LE to start the debugger

That was an example of how to start Debug Tool for a batch application using the Debug Tool 'User Exit data set' utility and a TIM terminal. At this point, you may be comfortable with the process, and have all of the information you need to start the debugger. The rest of this section goes into technical details about how it works. You can opt to skip the rest of this section, if you are not interested in the details.

Here is how it works. The 'User exit data set' utility saves your Debug Tool profile in a file. The test option file has to be named using a specific naming convention. By default, user ID.DBGTOOL.EQUAUMPTS. However, your systems programmer can customize the naming convention. Use the default naming convention unless you have been instructed otherwise by your system programmer.

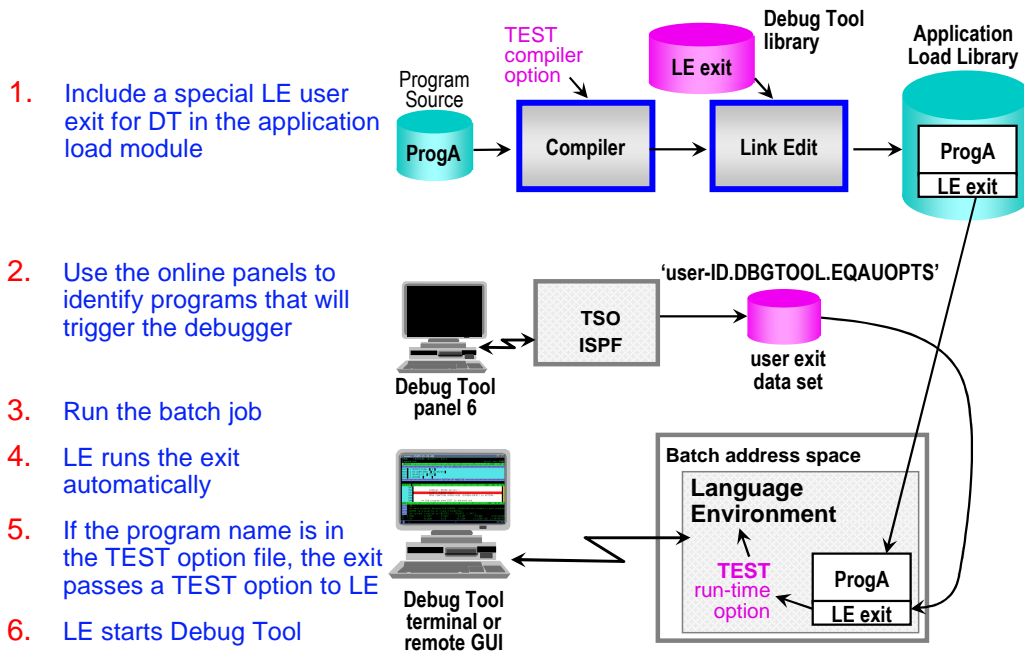
Debug tool provides special exits for Language Environment. If one of these exits is enabled when your program runs, it uses the user ID of the job to determine the user ID portion of the file name. The exit program then reads your test option data set. If it finds a match on the program name in your profile, the exit passes a TEST option to Language Environment, which causes the debugger to start.

- There are two ways to enable a Debug Tool LE exit program
 - Option 1: A supplied Debug Tool exit is link-edited into the application load module
 - Option 2: A supplied Debug Tool exit is made available in STEPLIB or JOBLIB in run-time JCL
 - The system programmer must set up a special library that Debug Tool users can add to their STEPLIB or JOBLIB concatenation

There are two ways to enable a Debug Tool LE exit program. The first way is to link-edit one of the supplied Debug Tool exits into an application load module. Then it will be enabled automatically whenever the program runs. The second way is to make the exit available by adding a special library to STEPLIB or JOBLIB of the run-time JCL. The system programmer must set up this special library so that Debug Tool users can use it.

Enabling the user exit data set utility - how it works

Option 1: The exit is linked into application load module



1. Include a special LE user exit for DT in the application load module
2. Use the online panels to identify programs that will trigger the debugger
3. Run the batch job
4. LE runs the exit automatically
5. If the program name is in the TEST option file, the exit passes a TEST option to LE
6. LE starts Debug Tool

23

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

This chart shows how the supplied LE exit works if your organization has chosen to link edit the exit into your application load modules. In the first step, the LE exit is included into your application load module when the program is compiled and linked. This could be made part of the standard compile process for all of your batch programs.

In the second step, the Debug Tool 'User exit data set' panels are used to identify the name of the programs that will trigger the debugger, and the terminal or workstation where the debugger will be displayed. These settings are stored in the file. In the third step, the job is submitted.

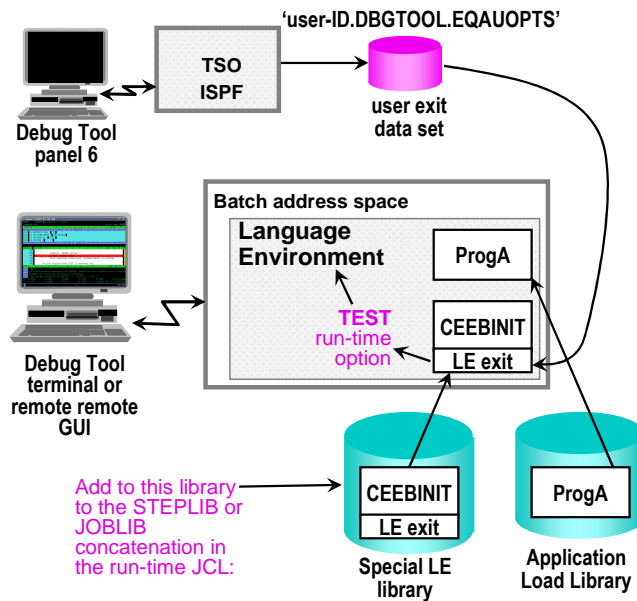
When the program runs, Language Environment automatically runs the exit program because it was included in the load module. This capability is a normal part of Language Environment initialization. The exit uses the user ID of the job to determine the name of the user exit data set. The exit opens the file and reads the options stored there. If there is match on the program name, then the exit passes a TEST option to Language Environment, and Language Environment starts Debug Tool.

Enabling the user exit data set utility - how it works

Option 2: STEPLIB or JOBLIB to the exit



1. The system programmer prepares a special library with the exit
2. Use the utility to identify programs that will trigger the debugger
3. Add the special library to STEPLIB in run-time JCL
4. Run the batch job
5. LE runs the exit automatically
6. If the program name is in the TEST option file, the exit passes a TEST option to LE
7. LE starts Debug Tool



24

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

This chart shows how the supplied LE exit works if your organization has chosen NOT to link edit the exit directly into your application load modules, but instead has chosen to make the exit available by adding it to your STEPLIB concatenation. In the first step, the system programmer prepares a special library that contains the exit module.

In the second step, the Debug Tool 'User exit data set' panels are used to identify the name of the programs that will trigger the debugger, and the terminal or workstation where the debugger will be displayed. These settings are stored in the file.

In the third step, the JCL to run the job is edited. The special load library is added to the STEPLIB or JOBLIB concatenation. In the fourth step, the modified JCL is submitted. When the program runs, Language Environment automatically runs the exit program that was made available in STEPLIB or JOBLIB. The exit uses the user ID of the job to determine the name of the user exit data set. The exit opens the file and reads the options stored there. If there is match on the program name, then the exit passes a TEST option to Language Environment, and Language Environment starts Debug Tool.

Enabling the user exit data set utility - how it works

Option 2: STEPLIB to the exit



Example:

- Note: Get the name of the library from your system programmer

```
//STEP5 EXEC PGM=MYPROG
//STEPLIB DD DSN=MY.PROGRAM.LIB,DISP=SHR
//          DD DSN=DBGTOOL.BATCH.LOADLIB,DISP=SHR
```

Tip: You can override STEPLIB in a step that is in a PROC without changing the PROC.

- This example adds or overrides the third library in the STEPLIB concatenation of the step named ASTEP in PROC99

```
//RUN99 EXEC PROC99
//ASTEP.STEPLIB DD
//          DD
//          DD DSN=DBGTOOL.BATCH.LOADLIB,DISP=SHR
```

Here are examples of coding your JCL if you are using the method where you add a special library to STEPLIB or JOBLIB. In the first example, the library is added to the existing STEPLIB concatenation.

That is fine, if your JCL is self-contained. However, your JCL may run a JCL PROC, and the STEPLIB that you need to modify may be hard coded in the PROC. If that is the case, you can code JCL as shown in the second example. That example assumes that STEPLIB is already coded in a step named ASTEP in a proc named PROC99. In this example the first two DD statements in the overridden STEPLIB concatenation remain unchanged, and a third library is either added or overridden.

Comparing the two ways to enable the user exit data set utility



- In option 1, a special Debug Tool module is link-edited into the application load module
 - Advantage:
 - No JCL changes are needed at run-time to start Debug Tool !
 - Considerations:
 - The exit module always runs once each time the main program starts, but incurs negligible overhead when not debugging
 - It may be preferred, but not technically necessary, to remove the exit module before promoting to production
- In option 2, the exit is made available at run-time in STEPLIB or JOBLIB
 - Advantages:
 - No change to the compile process (to link the exit into the application module)
 - You can control when the exit runs with JCL
 - Consideration:
 - Run-time JCL must be modified to start Debug Tool

Two options have been described for enabling the Language Environment exit that Debug Tool supplies. If it is up to you to choose which option to use, here is a comparison.

In option one, the supplied Language Environment exit is link edited directly into the application load module. The advantage to this method is that no JCL changes are needed at runtime to start the debugger. You just submit your JCL without any modifications whatsoever. You control whether to start the debugger strictly by specifying program names in the 'user exit data set' panel. A consideration for this method is that the exit will always run once each time a main program is executed, although this incurs negligible overhead. Even so, it may be preferred to remove the exit module before promoting the application module to production, to eliminate any possible risk of inadvertently triggering the debugger during a production run.

In option 2, the exit is made available at run time by adding a special library to STEPLIB or JOBLIB in your JCL. The advantages of this option are that no changes are needed to your compilation processes, and you can control when the exit runs with changes to your JCL. The consideration is that your JCL requires this simple modification before you can trigger the debugger.

Enabling the user exit data set utility A deep-dive into the details



- Debug Tool provides four versions of LE exit programs:
 - EQAD3CXT: a single exit that can be used for all LE batch, IMS/TM, and DB2 SPs
 - EQADBCXT: can be used for batch LE applications only (including IMS and DB2 batch)
 - EQADICXT: can be use for online IMS (IMS/TM) LE applications only
 - EQADDCXT: can be used for DB2 stored procedures only
 - Installation and use are described in the Debug Tool User's Guide
 - Source for the exits (in library SEQASAMP) can be customized to:
 - Change the naming convention of the TEST option data set and produce diagnostic messages
- To enable them:
 - Option 1: Link the appropriate version of the exit into each application load module, or
 - Option 2: The system programmer can create 3 new load libraries, which can be added to STEPLIB or JOBLIB to enable the exits
 - A loadlib for batch applications containing a copy of LE module CEEBINIT linked with exit module EQAD3CXT or EQADBCXT
 - A loadlib for IMS/TM applications containing a copy of LE module CEEBINIT linked with exit module EQAD3CXT or EQADICXT
 - A loadlib for DB2 stored procedures containing a copy of LE module CEEPIPI linked with exit module EQAD3CXT or EQADDCXT

27

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

If you are curious about exactly how to implement the supplied Language Environment exits, here is a deep-dive into the details. Debug Tool provides three exit programs that work with the 'user exit data set' utility. A module named EQADBCXT is used to start Debug Tool for **batch** Language Environment applications. A module named EQADICXT can be used to start Debug Tool for **online IMS/TM applications**, and a module named EQADDCXT is used to start Debug Tool for **DB2 stored procedures**. These modules are shipped in Debug Tool library SEQAMOD. Source code for the exits is also shipped with Debug Tool in the SEQASAMP sample library. The source code can be modified and assembled to change the naming convention for user exit data sets, and to request that the exit produce messages as an aid to the installer.

To enable the exits, one option is to link edit the appropriate version of the exit into each application load module. The exit is enabled because it is present in the module. The second option to enable the exits is to create new load libraries, which the Debug Tool user can add to their STEPLIB or JOBLIB. First, create a load library for batch applications. It will contain module CEEBINIT, from the Language Environment SCEERUN library, with the supplied exit module EQADBCXT link-edited into it. Also, create a load library for IMS/TM applications. It will contain module CEEBINIT, from the Language Environment SCEERUN library, with the supplied exit module EQADICXT link-edited into it. Finally, create a load library for DB2 stored procedure programs. It will contain module CEEPIPI, from the Language Environment SCEERUN library, with the supplied exit module EQADDCXT link-edited into it.

That is the end of this section, which described starting the debugger for a program running in batch, triggering the debugger using the Debug Tool 'user exit data set' utility, and displaying the debugger on a Debug Tool TIM terminal.

Feedback



Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_DTv12s07bDebuggingBatchSMTimExit.ppt

This module is also available in PDF format at: [../DTv12s07bDebuggingBatchSMTimExit.pdf](..../DTv12s07bDebuggingBatchSMTimExit.pdf)

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, CICS, CICS Explorer, DB2, IMS, z/OS, and zSeries are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2012. All rights reserved.