The Client Event Debug Monitor

The client debug event monitor is used to troubleshoot problems with the System Coordinator or with Adabas products that work closely with it. Normally it is only used under guidance from Software AG Customer Support. However, as you will see by reading on, you might find it useful when troubleshooting your own systems too.

The default is that it is inactive, apart from the thread recovery feature. Thread recovery operates in client jobs, automatically releasing (recovering) internal System Coordinator threads that have been blocked. These thread blockages usually occur where failures (such as transaction abend) have occurred while a thread is still in use. This recovery can help to keep thread usage to a minimum. When operating in default mode there is no report (diagnostic dump) produced for the recovered thread.

The debug event monitor can write diagnostic information to a file (CORDUMP) based upon the settings that you make. The default is that no diagnostic reports are to be written. In order to enable diagnostic reports you must choose the type desired and you must also introduce the CORDUMP file to the job's execution control script (in some operating systems).

The CORDUMP file must be correctly in place for the reports to appear. In some operating systems it will automatically default to list output. Similarly, reports will only appear if an exact match for the event(s) you have chosen occurs.

▶ **To monitor an event you must do the following:**

1. Make the CORDUMP file available to your job. Details of the file are shown in the online help screens.

2. Set the details of the monitored event and specify the report contents you want to appear in CORDUMP. Press PF5 to save these details.

3. Activate the debug event monitor

- Setting Debug Event Monitor Controls

- Activation of the Debug Event Monitor

- Runtime Overheads for the Debug Event Monitor

Setting Debug Event Monitor Controls

The debug event monitor controls are set in runtime controls. In SYSCOR, modify your client controls; press PF10 then select option 2 (debug settings) which presents the following screen:

```
 13:52:10     ***** A D A B A S   SYSTEM COORDINATOR 8.1.2 *****     2006-12-12
                   -  Debug Event Monitor Controls  -              U1SCJBM1

 Thread Recovery (Y/N) ..........: Y     Maximum recovery reports ...: _____

 Debug all sessions (Y/N) .......: Y     Maximum debug reports ......: _____
      Response code: ___ Sub-code : _____ or mark for generic monitor : _
      Optionally for database ....: _____ and file number ............: _____
    Additional debug monitor (Y/N), use only as directed by Software AG:
      System Coordinator .........: N     Adabas Transaction Manager .: N
      Adabas Fastpath ............: N     Adabas Vista ...............: N

    Report content in order of output amount, mark one:
      None .......................: X     Client session only ..........: _
      All sessions for the client : _     All sessions for the job ......: _
      All memory for the job .....: _
    Additional report content (Y/N):
      CIB ...............: Y   CAB ..............: Y   ID table .........: Y
      Registers on entry : Y   TP areas ........: Y   Stack ...........: Y


 Command ==>
 Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit        Upd
```

Thread Recovery

As stated previously, this is active by default but thread recovery reports are only written to CORDUMP if a non-zero value is specified for `Maximum recovery reports`. You should only change these settings under guidance from Software AG.

Debug all sessions

Once activated, the default is to look for the event happening in all client sessions that run. This is because it is fairly random which of your commercial users will experience the problem being tracked. However, in some cases it is possible to reproduce the problem in controlled circumstances. In this case you can configure the monitor to allow you to control the exact client session(s) that are monitored at runtime. To do this you must be very confident that you can identify specific terminal sessions that will be used. Once you are in a position to do that you can change `Debug all sessions` to N.

Later, when event monitor is activated in your job (activation is described below) enter into a dialog with SYSCOR (see Display Session Information in the System Coordinator documentation). In that dialog mark the session(s) that you wish to be monitored with a T. This will cause those sessions to be monitored and reports to be created if the event occurs.

You must also specify the `Maximum debug reports`. This limits the number of times the event will cause a report to be written to the CORDUMP file. The default is zero so you must change it to cause reports to appear. This is deliberately cautious; just imagine that a common response code event is activated such as response code 9. Without stringent controls there might be thousands of reports sent to the CORDUMP file. This could severely impact system performance. Therefore you must set a sensible number of times the event is to be reported, usually 1.

Set the details of the event to be monitored

By far the most common event that needs to be monitored is an unexpected Adabas response code. On the screen above you can choose to monitor the following types of event:

- a specific response code without a subcode

- a specific response code with a specific subcode

- a generic response code monitor

  This control means all response codes are monitored except those which do not indicate an error of significance. For example, response codes 0, 3, 9 and 148, a full list can be found in the help screens.

When looking for the response codes desired you can also restrict the monitor to a specific database and file number.

Additional debug monitor

From time to time Software AG may supply a diagnostic fix in order to generate a report to the CORDUMP file. This is under your full control so that you are able to activate that report in the jobs that you want to, and not in others. Enter Y against the products that you wish to monitor, under guidance from Software AG.

Report content in order of output amount

You must be very careful to limit the amount of output that will be produced. There are good reasons for being very careful in this choice. Some reasons are obvious such as slowing the system down, creating too much output, filling the CORDUMP file or spool, etc. The `Report content` controls help you to implement reasonable limits and avoid excess output.

The default is None. You must choose another option to produce any reports.

The most commonly used `Report content` option is the `Client session only` option. This limits the output to the session context that experiences the event. This will generate by far the lowest amount of output. The other options generate significantly more output and should only be used when advised by Software AG. Mark one option with a non-blank character. Whichever option you select, you must also define a non-zero `Maximum debug reports`, otherwise no output will be produced.

Additional report content

All the `Additional report contents` (the sections of the dump output that will accompany the session context) are assumed to be required; they default to Y. There is usually no reason to change them unless requested by Software AG in order to diagnose a problem situation. If you decide to use the event monitor to trap your own response code situations, you should deselect these options in order to reduce the size of the output.

Activation of the Debug Event Monitor

The debug controls set in your client runtime controls will automatically be activated the next time the job starts. You can activate them dynamically from SYSCOR if you wish. You do this in the Session Monitoring zone of SYSCOR by displaying the jobs and marking the appropriate one with R to refresh the

controls.

Note:
An appropriate CORDUMP file must be in place as detailed earlier.

Runtime Overheads for the Debug Event Monitor

The CPU overhead required for monitoring events is minimal. However, as mentioned previously, the output options must be carefully set in order to avoid unnecessary generation of excessive output reports.