Adabas System Coordinator

Adabas System Coordinator Operations and Programming Guide

Version 8.1.2

June 2008

Adabas System Coordinator

Table of Contents

1 Adabas System Coordinator Operations and Programming Guide 1
2 Starting the Adabas System Coordinator Daemon 3
3 Daemon Runtime Parameters 5
    CT – Command Timeout Limit 6
    FORCE – Overwrite ID Table Entry 7
    LOCAL - Define an Isolated Daemon 7
    LU – Length of Intermediate User Buffer 7
    MAXTTD – Limit Number of Thread Timeout Dumps 8
    MPMWTO – Display Information Messages 8
    NABS – Number of Attached Buffers 8
    NC - Number of Command Queue Elements 9
    NXC - Number of Sysplex Command Queue Elements 9
    PRODUCT - Identify the Services to be Made Available 9
4 Daemon Operator Commands 11
5 The Client Event Debug Monitor 13
    Setting Debug Event Monitor Controls 14
    Activation of the Debug Event Monitor 17
    Runtime Overheads for the Debug Event Monitor 17
6 User Queue Elements (UQEs) for Configuration File Access 19
7 API To Retrieve Runtime Control Site Information 21
    3GL API 22
    Natural API 22
    Return Codes 23

1 Adabas System Coordinator Operations and Programming Guide

This document provides information related to Adabas System Coordinator operations and programming.

The following topics are provided:

Operational Guidelines

- Starting the Adabas System Coordinator Daemon
- Daemon Runtime Parameters
- Daemon Operator Commands
- Using the Client Event Debug Monitor
- User Queue Elements (UQEs) for Configuration File Access

Programming Guidelines

- API To Retrieve Runtime Control Site Information

2 Starting the Adabas System Coordinator Daemon

Normally, there is one Adabas System Coordinator daemon per operating system image.

The daemon must be started before any TP monitors or batch jobs that use its services.

3 Daemon Runtime Parameters

- CT – Command Timeout Limit 6
- FORCE – Overwrite ID Table Entry 7
- LOCAL - Define an Isolated Daemon 7
- LU – Length of Intermediate User Buffer 7
- MAXTTD – Limit Number of Thread Timeout Dumps 8
- MPMWTO – Display Information Messages 8
- NABS – Number of Attached Buffers 8
- NC - Number of Command Queue Elements 9
- NXC - Number of Sysplex Command Queue Elements 9
- PRODUCT - Identify the Services to be Made Available 9

The following parameters can be entered using DDCARD input. The PRODUCT parameter is mandatory. All other parameters are optional.

Parameter	Usage
CT	Command timeout limit.
FORCE	Overwrite ID table entry.
LOCAL	Define an isolated daemon.
LU	Length of intermediate user buffer.
MAXTTD	Limit number of thread timeout dumps.
MPMWTO	Display information messages.
NABS	Number of attached buffers.
NC	Number of command queue elements.
NXC	Number of sysplex command queue elements
PRODUCT	Identify the services to be made available. Mandatory.

CT – Command Timeout Limit

Parameter	Use	Mimimum	Maximum	Default
CT	The maximum number of seconds (more precisely, units of 1.048576 seconds) that can elapse from the time a daemon request is completed until the results are retrieved by the sender through the interregion communication.	1	16,777,215	60

This parameter is used to prevent a request queue element (RQE) and attached buffer from being held indefinitely when a user with an outstanding request terminates abnormally.

Possible causes of a command timeout are

- address space is swapped out or cannot be dispatched;
- the task is cancelled or ABENDed;
- the task has low priority in a high-activity system.

FORCE – Overwrite ID Table Entry

Parameter	Use	Possible Values	Default
FORCE	Specify whether or not this daemon is to force an entry into the active node list.	YES | NO	NO

Possible values:

- `FORCE=YES`: Force an active entry, if one is available.

  `FORCE=YES` is usually not required. However, it may be needed if the previous daemon session ended abnormally, leaving the old entry in the active node list. Use this setting carefully.
- `FORCE=NO`: Causes an error if the node used by this daemon already appears in the active node list.

LOCAL - Define an Isolated Daemon

Parameter	Use	Possible Values	Default
LOCAL	Specify whether or not a daemon is to be isolated from other Entire Net-Work nodes.	YES | NO	NO

Possible values:

- `LOCAL=YES:` Isolates this daemon from other Entire Net-Work nodes.
- `LOCAL=NO:` The daemon can receive calls from other Entire Net-Work nodes.

LU – Length of Intermediate User Buffer

Parameter	Use	Mimimum	Maximum	Default
LU	Set the size of the intermediate user buffer area.	4000	65,535	65,535

The size specified must be large enough to accommodate all control information for commands passed to the node.

An error occurs if the `LU` parameter specifies a value greater than the byte count implied by the `NAB` parameter. If you change either parameter value, you may have to change them both.

MAXTTD – Limit Number of Thread Timeout Dumps

Parameter	Use	Possible Values	Default
MAXTTD	Limit the number of thread timeout dumps produced.	0-99999	No limit

When the SYSCO daemon loses contact with a database it issues message "CORD045E Thread Timeout", and produces a dump of internal control blocks for diagnostic purposes. Since a thread timeout is often a normal response, the dumps are usually not required, and can fill dump / listing files. `MAXTTD` can be used to limit the number of thread timeout dumps produced. If `MAXTTD=0` is specified, all thread timeout dumps will be eliminated.

MPMWTO – Display Information Messages

Parameter	Use	Possible Values	Default
MPMWTO	Specify whether or not to display information level (l-level) messages.	YES | NO	NO

By default, information level (I-level) messages are suppressed.

NABS – Number of Attached Buffers

Parameter	Use	Mimimum	Maximum	Default
NABS	Specify the number of attached buffers to be used.	0	500,000	16

An attached buffer is an internal buffer used for communication with the daemon.

For Adabas System Coordinator, this is an optional parameter that defines the number of attached buffers to be used for receiving requests from clients or from other daemon peers.

An attached buffer pool is allocated with a size equal to the value of the `NABS` parameter multiplied by 4096 bytes.

NC - Number of Command Queue Elements

Parameter	Use	Mimimum	Maximum	Default
NC	Set the maximum number of command queue elements.	20	32,767	100

The maximum number of command queue elements (CQEs) that can be processed simultaneously by this daemon.

NXC - Number of Sysplex Command Queue Elements

Parameter	Use	Mimimum	Maximum	Default
NXC	Set the maximum number of queue elements in the sysplex messaging command queue.	1	32,767	256

This parameter is used only for multi-system daemons using XCF messaging, or for full Parallel Systems daemons using dynamic transaction routing (DTR). The default value will be sufficient for most sites.

PRODUCT - Identify the Services to be Made Available

Parameter	Use	Possible Values	Default
PRODUCT	Specifies which product services are to be made available by Adabas System Coordinator: ■ AFP: Adabas Fastpath Asynchronous Buffer Manager ■ ATM: Adabas Transaction Manager ■ AVI: Adabas Vista daemon component ■ DTR: Dynamic Transaction Routing Service	AFP ATM AVI DTR	none

This parameter is used once for each service that is to be made available by the daemon.

Sites that use Adabas Fastpath will also require `PRODUCT=AFP`. Sites that use Adabas Vista or Adabas Transaction Manager and wish to support dynamic transaction routing in a clustered application will require `PRODUCT=AVI` and/or `PRODUCT=ATM`.

4 Daemon Operator Commands

The following operator commands are available through the z/OS `Modify (F)` command, VSE/ESA operator command, or z/VM and BS2000 commands.

Command	Description
STOP	Terminates the daemon in an orderly manner. The commands `SHUTDOWN` and `ADAEND` can also be used for the same purpose. The daemon should not be stopped while there are jobs active.
DPARM	Displays the runtime parameters for this execution of the daemon.
CAS DXCF	Displays message count information for cross-daemon XCF messages. The following counts are displayed in response messages CAS021I, CAS022I; ■ Msg-out: Total number of XCF messages sent by this daemon ■ Msg-in: Total number of XCF messages received by this daemon ■ Msg-rsp: Total number of message responses received ■ Msg-segs: Total number of message segments ■ Cq-Num: Number of queue elements in the Sysplex command queue. This will be equal to the command queue size specified, or defaulted, in the `NXC` parameter. ■ Cq-Hwm: Peak usage of the command queue. If this value is approaching the Cq-Num value it is advisable to increase the `NXC` value. ■ Cq-Full: Number of times a command queue full condition occurred. If greater than zero, increase the `NXC` value. ■ Cq-Post: Number of messages posted ■ RspPost: Number of responses posted This command is not required during normal daemon operations, but its use may be requested by Software AG personnel for diagnostic purposes.

Command	Description
	Note: XCF message counts will only be displayed if the daemon is running in "standard multi-system" or "parallel sysplex" mode.
CAS DXCF RESET	Displays message count information, as above, then resets all accumulated counters to zero.

5 The Client Event Debug Monitor

- Setting Debug Event Monitor Controls 14
- Activation of the Debug Event Monitor 17
- Runtime Overheads for the Debug Event Monitor 17

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

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.

6 User Queue Elements (UQEs) for Configuration File Access

Adabas System Coordinator uses internally-generated session communication IDs for configuration file access. It is normal for one or more active user queue elements to be seen in the Adabas nucleus that contains the configuration file. These UQEs are continually reused during the lifetime of a client job or service. If required, the Adabas `DELUF` command can be used to delete outstanding UQEs once all client activity is terminated. Alternatively that nucleus can be started with the `OPENRQ=YES` parameter. This will cause automatic deletion of Coordinator UQEs after an inactivity timeout period of one minute.

7 API To Retrieve Runtime Control Site Information

- 3GL API 22
- Natural API 22
- Return Codes 23

Adabas System Coordinator allows storing of site-specific data in runtime controls. The contents and format of the data are entirely under your control. A typical use for it might be to define your own runtime controls (for example dynamic Natural parameters). The data can be retrieved using the supplied APIs. Two APIs are provided; one for 3GL and Assembler programs and one for Natural programs.

3GL API

The 3GL API is contained in the supplied COR3GLI load module. There are also some supplied source members showing how to use the API:

- APIINF01: example of using the API in environments other than CICS
- APIINF02: example of using the API in CICS
- COR3GLIA: a parameter data area for calling COR3GLI

To use the 3GL API:

1. Allocate storage for the parameter data area (1024 bytes).
2. Initialize the storage to binary zeroes.
3. Set the interface version (field name INFVRS in COR3GLIA) and function (INFFNC).
4. Under CICS, set the name of the Adabas link module to be used (INFCICN). The link module must be capable of accepting parameter lists via the COMMAREA. If not under CICS, INFCICN must contain binary zeroes or spaces.
5. If using the reentrant ADALNKR, allocate a modified area and set its address (INFAMOD).
6. Link this program together with COR3GLI and, if not under CICS, your Adabas interface module, for example ADAUSER.
7. After calling COR3GLI, INFRC will contain 0000 and INFDATA will contain the site information for this session; or, INFRC will contain a non-zero return code and INFRT will contain an explanatory message.

Natural API

The Natural API is contained in library SYSCOR, subprogram CORNATI. There are also some supplied source members in SYSCOR, showing how to use the API:

- APIINF-P: example of calling CORNATI
- CORNATIA: parameter data area for calling CORNATI

To use the Natural API:

1. Ensure that subprogram CORNATI is available, by copying it to your Natural library or adding SYSCOR to your library's steplibs in Natural Security.
2. Set CORNATI-VERSION and CORNATI-FUNCTION.
3. Call CORNATI.
4. After calling CORNATI, CORNATI-RC will contain 0000 and CORNATI-SITEINFO will contain the site information for this session; or, CORNATI-RC will contain a non-zero return code and CORNATI-RT will contain an explanatory message.

Return Codes

These are the non-zero return codes which may be set:

Return Code	Description
0001	Invalid interface version (must be 01)
0002	Invalid function (must be GETINFO)
0003	System Coordinator not available
0004	System Coordinator internal error
0005	System Coordinator internal error
0006	Adabas interface not linked (3GL API only)
0008	No site information for this session