Error Handling

This chapter covers the following topics:

- General Error Handling

- Error Handling of the CALLDLI Interface

- Error Handling of the Consistency Interface

General Error Handling

ADL Response Codes

Whenever ADL detects an error situation, it issues an error or warning message. Each individual message is identified by a unique number in the range from 1 to 1500. The *Error Messages and Codes* documentation lists all messages in numerical sequence together with an explanation and recommended action.

Note that error messages with numbers in the range from 1 to 255 are reserved for and identical to Adabas response codes. A more detailed explanation of these can be found in the *Adabas Messages and Codes* documentation.

Abnormal Endings Caused by ADL

If an unrecoverable error is detected, ADL terminates the user program abnormally. In particular, batch programs are abended with the user completion code 700, or the `JDUMP` macro in z/OS and z/VSE respectively. IMS/TP applications are abended with the user completion code 700.

In some cases, ADL terminates batch programs abnormally with a system completion code `"0C1"` (operation exception). These errors can easily be identified by the `PSW` pointing to a character string like:

```
-> INVALID PCB NUMBER
```

or

```
-> ATTEMPT TO ACCESS NON CONVERTED PCB
```

These error situations, however, are rare and should not occur under normal circumstances.

CICS applications are terminated abnormally either with an abend code `DAZA`, accompanied by an ADL error message, or any other abend code in the form `DAZx` where `x` may be `"B"`, `"C"`, `"D"` or a number between 1 and 9. These abend codes are explained in the section *Error Messages and Codes*.

Adabas Response Codes

ADL treats Adabas response codes in two different ways. The first class of Adabas response codes is considered to be unrecoverable and thus leads to an abnormal termination of the user application. An example of an unrecoverable Adabas response code is the response code 148 (database not active).

The second class of Adabas response codes is considered to be recoverable. In particular, these are the response codes 3 (end-of-file or end-of-list), 9 (time limit has been exceeded), 145 (the hold queue is full) and 198 (duplicate value for unique descriptor).

System Request Failures

Failures during system requests from ADL (like `GETMAIN`, sequential I/O etc.) are considered to be unrecoverable and generally cause an abnormal termination of the application (see above in this section). You should try to determine the cause of the error. Most of these are related to improperly defined sequential files or insufficient region sizes.

Internal Errors

Software AG tries to minimize potential sources of internal errors like data exceptions, buffer overflows etc. Thus, most of the internal errors can be solved by adjusting the relevant parameters, for example buffer sizes. Should there be any internal error which requires some action by Software AG, this is clearly marked in the section *Error Messages and Codes* in the `"Action:"` clause of the particular error message.

Error Handling of the CALLDLI Interface

Adabas Response Codes

Due to the fundamental differences between the two data base systems, not all Adabas response codes can be translated into DL/I status codes. ADL transforms some of the recoverable Adabas response codes into DL/I status codes.

The Adabas response code 3 normally results in a DL/I status code `"GB"` (end of dataset) or `"GE"` (segment not found) being returned to the application. The response code 198 results in a DL/I status code "II" (segment to insert already exists in the data base).

The Adabas response code 9 normally occurs in online applications only. The typical reason for this is that the user at a particular terminal has been inactive for a period of time longer than the Adabas time limit. If ADL finds that no data has been backed out by Adabas, it simply re-issues the command without any effect on the application. Otherwise it abnormally terminates the transaction, causing the online system to back out its resources as well.

If the `RETRY` parameter is set to `"WAIT"`, ADL receives a response code 145 only when the Adabas hold queue is full. In this case, Adabas attempts to issue the command a second time. If the second attempt fails as well, the application terminates abnormally. Should you observe that ADL terminates applications frequently for this reason, you should increase the size of the Adabas hold queue. If the `RETRY` parameter is set to a numeric value, ADL terminates the application abnormally if a required record is in hold status. See the section ADL Parameter Module in the *ADL Installation* documentation for more details on the `RETRY` parameter.

DL/I Status Codes

The ADL `CALLDLI` Interface generates status codes which are identical to and have the same meaning as those of DL/I. This is in particular true for all status codes which are related to the contents of the data base (like `"GB"`, `"GE"`, `"GK"` etc.) or caused by invalid call or segment qualification formats (like `"AM"` or `"AJ"`).

There are, however, a couple of DL/I status codes which are strongly related to the physical design of DL/I. Examples for these are `"XD"` (error during data base buffer write) and `"XH"` (data base logging not active). Since these do not correspond to any Adabas response code, they will never be generated by ADL.

The meaning of the DL/I status codes can be viewed in the ADL Online Services with the "Messages and Codes" function.

Error Handling of the Consistency Interface

Consistency Response Codes

If a data base request originating from a Natural program or an Adabas direct call does not fulfill the rules and conditions required for the integrity of the database, the Consistency Interface returns the response code 216 in the response code field of the Adabas command block. In Natural programs, this corresponds to the `*ERROR-NR 3216`.

If this error occurs, the ADL Consistency Interface provides a more detailed error message on request. The corresponding error codes and message are explained in the section *Error Messages and Codes*. The section *How to Retrieve an Error Code and Message* below describes how an application program may obtain these comprehensive error messages.

How to Retrieve an Error Code and Message

Natural programs may retrieve the full ADL error message by a `"CALLNAT"` to the ADL-supplied Natural subprogram `ADLERROR`, supplying an 80 character string as output field.

Example:

```
IF *ERROR-NR = 3216
     CALLNAT 'ADLERROR' #ERRMES
END-IF
```

The layout of the error message is

```
ADL xxxx - error text
```

where `"xxxx"` is the actual error number. The meanings of the individual error numbers and messages are explained in the section *Error Messages and Codes* together with a recommended action for each error.

Programs using Adabas direct calls may retrieve an ADL Consistency Interface error message by issuing an Adabas `"S1"` call as shown below:

```
Command Code        : S1
Command ID          : blank or binary zero
File Number         : filled in by ADL
ISN Lower Limit     : zero
Command Option1     : blank
Command Option2     : blank
FB length           : 26 bytes minimum
RB length           : 80 bytes minimum
SB length           : 3 bytes minimum
VB length           : 2 bytes minimum
IB length           : not used
```

```
Format Buffer        : must contain CL8'ADLERROR'
Record Buffer        : on return contains the ADL error message
Search Buffer        : filled in by ADL
Value Buffer         : filled in by ADL (error number in binary format)
```

Any fields of the Adabas control block not mentioned above are not used.