Enterprise Computing Insights

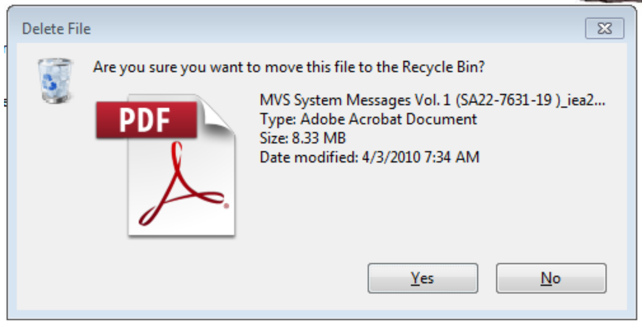
What are Messages and Codes?

#9 in a Series

When working with an enterprise server[1] and z/OS, you will encounter messages from the operating system and related products on an on-going basis. These messages are the way that the operating system or the software you are using communicates information to you or the need for you to take action. Sometimes the meaning of the message and what needs to be done will be obvious just by reading the message text. More often than not, especially for someone new to z/OS, that will not be the case. You will need to obtain further information and decide what needs to be done. This article will acquaint you with z/OS messages, codes, and the tools that can be used to help you understand them.

Messages

Think of the interaction between an end user (human) and a computing system as a conversation. The end user makes a request of the computing system – perhaps by entering a TSO[2] command to display some information or requesting an application program to be run. The response to the end user request may include a message – a way for the operating system or application program to provide information about the status of the request. For example, the requested action may have completed successfully or the system may have encountered an error. If you have used a personal computer, you have probably encountered various messages. Figure 1 shows an example of a Windows message that requires a decision on the part of the end user.

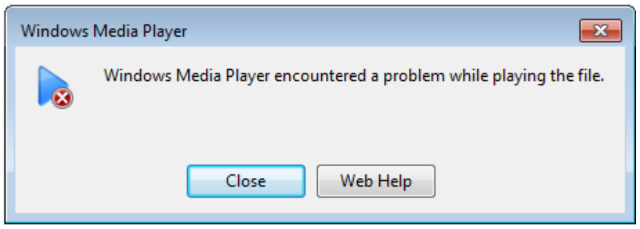

Delete File

Are you sure you want to move this file to the Recycle Bin?

MVS System Messages Vol. 1 (SA22-7631-19)_iea2...
Type: Adobe Acrobat Document
Size: 8.33 MB
Date modified: 4/3/2010 7:34 AM

Figure 1 An example of a message that requires a decision

Figure 2 shows an example of a Windows informational message that requires further investigation to determine what needs to be done.


Windows Media Player

Windows Media Player encountered a problem while playing the file.

Figure 2 An example of an information message

Managing Messages

When using an enterprise server, you will find that there are several sources of messages; the z/OS operating system itself, "add on" software products such as a database or transaction manager, and application programs.

The destination for messages is also varied. One primary destination for messages is an operations console. Here a human operator is responsible for monitoring and responding to messages. A long time ago, the z/OS operator's main job was to manage messages – read them, understand them, respond to them, take action, etc., much like you do on a PC today. As we have seen, a message may be informational in nature or a message may ask that you make a decision. In today's world, with very large and very fast systems processing literally

[1] See ECI No. 5
[2] See ECI No. 10

thousands of transactions per second and hundreds of batch JOBs[3], the flow of messages is just too fast for a human to read and respond in a timely manner. Consequently, while messages are still extremely important, many of them have been automated or suppressed. If they have been “automated”, it means that an automation tool or product has been set up to respond to an action or decision message with a “standard” response so that a human operator does not have to. And many informational messages are simply suppressed – not displayed at all for an operator, usually because it is not really very informative and nothing is going to be done any way (e.g. message IEF403I *jobname* STARTED that indicates that a particular JOB has begun and message IEF404I *jobname* ENDED that indicates that a particular JOB has ended is not very informative in a system that runs hundreds or thousands of JOBs). Even though a message is suppressed at the operator’s console, it can still be entered into a log so as not to be “lost” if needed. So, the operator at an operations console is left with the really important messages – the ones that can’t be automated or suppressed and need some human thought before a response is provided.

In addition to an operator console, messages can be found in various logs that record messages and other events. Examples include OPERLOG (which records all of the console messages from all the systems in a Parallel Sysplex configuration as well as all of the commands or responses entered by the operators), SYSLOG (which is similar to OPERLOG, but for a single system) and a Job LOG (where JOB related messages are sent as defined by MSGCLASS on the JOB statement for the JOB). Messages associated with a JOB may also be sent to a SYSOUT data set. If MSGCLASS and SYSOUT are directed to the same data set, then all messages for a JOB will be found in the same place. When examining a message sent to any of these destinations, it is helpful to understand the general structure of a message.

Message Format

The general format of a z/OS message is:

CCCCnnnnns *message text*

where:
CCCC is a three or four character string that identifies the z/OS component, subsystem, or product that issued the message. If a fourth character is present, it identifies a subcomponent of the component (identified by the first three characters) that issued the message. The character string is pretty cryptic, but after you have seen enough z/OS messages, you will begin to recognize what part of the system a particular character string identifies. For example, messages that begin with “IEA” are issued by the supervisor component of z/OS and messages that begin with “HAS “are from the Job Entry Subsystem (JES). These identifiers are documented, but there are so many of them that it is not worth trying to memorize them. Exposure to and working with messages is the best way to learn the identifiers and their corresponding relationship to the system. (Note: the z/OS three character message prefix is often the same three characters used to name the component module identifier for the software module that issues the message). You can find the three character message identifier for z/OS components in the MVS Diagnosis Reference manual (GA22-7588).

nnnnn is a three to five character number that uniquely identifies the message

“s” is a suffix that identifies the message “type” – i.e. the system expectations as to what needs to be done. The characters you will find in the “s” position of a message are:

- A – an action message; the system is expecting an action to be taken
- D – a decision message; the system is expecting a choice to be made
- E – an error/eventual action message
- I – an informational message; something the system thinks you should know about
- S – a severe error
- W – a wait or attention message; processing stops until an action is taken

[3] See ECI No. 4

Finally, the “message text” provides the actual message text of the message. Some simple messages can be understood simply by reading the message text, but usually the message text needs some additional explanation.

Understanding Messages

One way to get additional information regarding a message is to look it up in the books that document the details of the messages. There are several volumes of message documentation (SA22-7631 through 7640) so consulting a book may not always be convenient. Instead, you can use an IBM program called LookAt that is designed to provide a detailed explanation for a particular message when you provide just the message identifier. LookAt can be used on the web (at: http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/) or installed on a PC or on mobile device and used locally. It can also be used directly from TSO. Another option used to understand messages is to install a z/OS vendor product like MVS QuickRef by Chicago-Soft. QuickRef is similar in concept to LookAt in that it provides explanatory information for z/OS messages and codes. But it also provides other z/OS reference information such as JCL syntax, usage for various z/OS utilities and other useful information.

While z/OS messages bear some resemblance to messages on a PC, there are some significant differences as well. First, they look very different in that they are not displayed in a pop up window and the z/OS message identifier has structure as described above. Second, the volume of messages produced by an enterprise server with hundreds or thousands of end users is considerably more than that produced by a PC with a single user. We have seen how suppressing and automating messages can be used to significantly reduce the amount of message traffic that a human operator needs to deal with. Third, while the basic message format described above is always used, the basic message itself may be surrounded by other information depending on where the message is recorded. For example, a message displayed on an operator console will be preceded by a unique numeric console message identifier. This is so that the operator can reply selectively to any particular message that needs a response (messages usually do not have to be replied to in strict chronological order). So, if there are three outstanding messages that need a response and their console message identifiers are: 26, 47, and 53, an operator can type the following at the console:

```
R47,end
```

Doing so indicates that the operator is responding to the message with the console identifier of “47” and the response for the message in this case is “end”. Messages with console message identifiers of 26 and 53 in this case are still outstanding and need replies at some point.

Codes

The term “code” is short for "completion code". A completion code is issued by the system or an application program in the event something terminates abnormally (an “abnormal end” is usually referred to using the contraction "abend"). The completion code provides the reason for the abend. A system completion code consists of an “S” followed by a unique three digit hexadecimal number (e.g. S0C1) that is used to look up additional information in the MVS Codes manual (SA22-7626).

If a user code is issued (e.g. by an application program) it consists of a “U” followed by a four digit decimal number (e.g. U0230). User completion codes are documented in the product documentation for the product that issues the code so there is no central repository of all the user completion codes as there is for z/OS system completion codes.

System and User codes are occasionally found in the message text that accompanies an abend. Since completion codes are usually the result of an error condition, they are also typically encountered in dumps and error logs (e.g. LOGREC) where they will be of use to whomever is diagnosing the problem.

Some errors are so serious that the operating system cannot continue processing. In these cases the system issues a wait state completion code and processing stops. Many of these wait state completion codes have to do with errors (abends) encountered during z/OS initial program load (IPL).

Summary

Messages and codes are a way of communicating information to or requesting an action or decision from a user of an enterprise computing configuration.

We will explore other aspects of Enterprise Computing in subsequent articles.