Enterprise COBOL Dump Debugging

Duration

3 days.

Participants

Application programmers. COBOL programming experience and knowledge of MVS JCL are required.

Objectives

After successfully completing this course, you will be able to:

- Identify the program or module that was processing at the time of abend and locate the source code instruction last executed and related data values.
- Use the Task Management control blocks and Save Area Trace to determine levels of control within the task and to locate the register contents of all application programs.
- Use the Data Management control blocks to identify all currently open datasets, the logical record being processed in each, buffer contents, DASD information, and possible causes of Input/ Output errors.
- Resolve typical abends such as SOC7, SOC1, SOC4, SO13, SB37, SD37, SE37, S001, 5222, S322, U4038 and file status errors.

Format

Lecture and classroom workshops. You will work with many different Enterprise COBOL dumps solving common abends and implementing debugging strategies. To provide dumps that mirror your computer environment, we supply abending COBOL programs for compilation and execution on your computer to generate and print dumps for classroom use.

Related Courses

This course is a pure dumps reading course. When developing or modifying programs, programmers often use tools to help with the debugging process. Consider taking these courses:

- Xpediter/ TSO
- Xpediter/ CICS
- Xpediter/ DB2
- IBM z/ OS Debug Tool
- RDz

Topic Outline

Debugging Tools

Compiler listings and control blocks
Linkage editor maps
CEEDMPs
AbendAid dumps (optional)
TEST compiler option
USE FOR DEBUGGING DECLARATIVES
IBM publications to use in solving dumps

Enterprise COBOL Dump Debugging *(continued)*

Topic Outline

Analyzing the Problem

- Using the abend code and abend message to find cause of error
- System vs User abends
- Using compile-time and run-time options
- Determining the domain of the problem
- Narrowing the scope of the problem
- Finding the failing statement
- Examining related data
- Tracing through control blocks

Debugging on a TSO Screen

- Tips for navigating through and debugging a hexadecimal dump in TSO ISPF 3.8 or SDSF

Debugging Procedures and Approaches for Solving

- Data exceptions
- Addressability errors
- I/ O problems
- Logic and looping problems
- Static (linkage editor) and dynamic call problems
- Miscellaneous data problems
- Miscellaneous JCL-related problems

Technical Details

- The Basics
  - MVS z/ OS system components
  - Data formats
  - JCL essentials
  - CEEDMP layout
  - Reading a CEEDMP
- Analyzing JES2/ JES3 LOG and JCL message output
  - Allocation
  - Deallocation
- Analyzing compiler output
  - Procedure division
    - LIST and OFFSET
  - Data division
    - MAP
  - Compiler options in effect (PARMs)
  - Memory map
  - TGT

Enterprise COBOL Dump Debugging *(continued)*

Topic Outline

Technical Details ***(continued)***

- Analyzing Binder (linkage editor) output
  - MAP output
  - XREF output
  - Linkage editor options (PARMs)
- Analyzing CEEDMP and AbendAid output
  - PSW
  - Register save areas (forward and backward chaining)
  - General purpose registers at time of abend
- Control blocks
  - TCB, FCB/ RUNCOM, THDCOM, CLLE, CBOVEC, ASCB, PRB, SVRB, CDE, XTLST, TIOT, LLE, DEB, DCB, IOB, ACB and other VSAM control blocks
- Analyzing TEST(SYM) or TEST
- Relating compiler output, linkage editor output, and dumps
  - BLF, BLW, BLL, HEXLOC
  - Interpreting the hexadecimal
- Multiple entry points
  - Save area trace
  - RSA
- Locating the failing instruction and the module containing it
- Locating data
  - Physical sequential
  - VSAM
  - Passed data
- Locating buffers and dataset label information
- Dynamic and static calls
  - Effect on dump
  - Tracing flow of control and return points
- Determining the cause of the abend