z/OS V1R13

Language environment:
LE C-RTL I/O ABEND recovery part 1

Session objectives

- Describe new language environment C/C++ run-time library support for I/O abend recovery.

Overview

- Problem statement / need addressed
  - The language environment C/C++ run-time Library (C-RTL) always attempts to ignore abend conditions that occur during low-level OS I/O processing. Sometimes situations are encountered where a condition can not be ignored. DFSMS will issue an abend and LE condition handling takes over. The C-RTL does not recover (return gracefully to the application) in these instances.
  - Application developers write condition handlers or SIGABND handlers to attempt to recover when the C-RTL can not ignore these types of abend conditions.
    - Restrictions apply to what can be done in the abend handler
  - Having the C-RTL recover in some situations (when the abend can be ignored) and not recover in others is not user friendly behavior.
- Solution
  - Provide a mechanism to enable the C-RTL to recover gracefully from an abend condition during output or CLOSE processing, when the abend can not be ignored.
  - The C-RTL function that triggered the abend condition will gracefully return to the application instead of bringing down the enclave (if condition handling is not in effect).
- Benefit / value
  - Improved reliability. Knowing that an abend during a low-level OS I/O operation will not terminate the language environment.

Usage and invocation

- Two different ways to invoke abend recovery behavior:
  - New fopen()/freopen() keyword
  - New environment variable
- Either method can be used to control how the C-RTL treats abend conditions that can not be ignored.
- When either method is set up to recover from the abend, the C-RTL will instruct the function to return a failing value to the application and set errno to 92.
- Diagnostic information will also be set in the __amrc structure.
- Common abend conditions where an ignore might not be allowed:
  - B37-04 – insufficient space on volume to allocate another extent
  - D37-04 – primary used, no secondary allocated
  - E37-04 – primary used, secondary used
  - B14-0C – no space left in the directory
- fopen()/freopen() keyword usage
  - abend=*abend* | *recover*
  - The value **abend** instructs the runtime library to ignore abend conditions that can be ignored. No attempt is made to recover from abend conditions that cannot be ignored.
  - The value **recover** instructs the runtime library to attempt to recover from an abend issued during certain low-level I/O operations (WRITE / CHECK sequence and CLOSE).
  - This method specifies the behavior for only the stream being opened.
  - This method overrides the setting of the _EDC_IO_ABEND environment variable

  - `fopen("//'myfile.data'","wb,type=record,abend=recover");`

Environment variable usage
  - Environment variable name is _EDC_IO_ABEND
  - **ABEND** and **RECOVER** are the possible values
    - The values invoke the same behavior as their relative fopen() keyword values.
    - When unset or set to something other than **RECOVER**, the default behavior is **ABEND**.
    - The setting of the environment variable defines the behavior for the life of an open stream
    - The environment variable can be overridden by the fopen() keyword.
      - `setenv("_EDC_IO_ABEND", "RECOVER", 1);`

CEECAA Updates (with offsets)

CEECAASHAB_RECOVER_IN_ESTAE_MODE (+30C)

[bit in the CEECAAFLAG1 field] when ON, the language environment ESTAE resumes to the abend shunt in the mode and key in which the language environment ESTAE was established.

CEECAASHAB_KEY (+30D)

[character] IPK result when CEECAASHAB is set.

These fields are added to safe guard against key switching and doing a retry in a different key than which the recovery routine was established.

Interactions and dependencies

- Software dependencies
  - None
- Hardware dependencies
  - None
- Exploiters
  - SMP/E (deployment manager) which is planned for z/OS 1.13 is expecting the recovery for the SD37 r=4 to occur in the C-RTL with a graceful return to the c function that triggered the abend

Migration and coexistence considerations

- None

Installation

- None

Session summary

- C-RTL can now recover gracefully from abend conditions that can not be ignored during output or CLOSE processing.
- Behavior is controlled via two mechansism
  - New fopen() keyword, abend=
  - New environment variable, _EDC_IO_ABEND

Appendix - References

- XL C/C++ run-time library Reference (SA22-7821)
- XL C/C++ programming Guide (SC09-4765)
- Language environment vendor Interfaces (SA22-7568)