IBM Education Assistance for z/OS V2R2

Item: Improve Subsystem Initialization and Management
Element/Component: BCP Scheduler

Material current as of May 2015

Agenda

- Trademarks
- Presentation Objectives
- Overview
- Usage & Invocation
- Installation
- Presentation Summary
- Appendix

Trademarks

Presentation Objectives

- This presentation will cover the following Subsystem Interface changes:
  - Updates to initialization routine (INITRTN) processing.
  - Logical subsystem deletion processing
  - Infrastructure related to logical subsystem deletion and services available to a subsystem.

Overview – Problem Statement

- Problem Statement / Need Addressed
  - Subsystems are permanently defined.
    - Subsystem names are reserved when the subsystem is defined, regardless of whether it could be successfully initialized. A subsystem cannot be “re-defined” in the case of an user or system error.
    - If a subsystem fails, even if it can be disabled or deactivated, the subsystem name is still reserved and cannot be reused.

Overview – Solution and Benefit

- Solution
  - There are multiple parts to the solution:
    - Initialization routine processing is changed to better handle user errors when defining a subsystem
    - New function is added to allow logical deletion of a subsystem.
    - Additional infrastructure is added to support logical deletion of a subsystem.
- Benefit / Value
  - Quicker and easier recovery from subsystem-related errors.

Usage & Invocation – Initialization Processing

- A common error when installing a subsystem is that the INITRTN is not accessible. Reasons:
  - The INITRTN that is specified is incorrect.
  - The INITRTN that is specified is not installed.
  - The INITRTN that is specified is installed, but is not APF-authorized.
- Traditionally, the subsystem is defined, but since the INITRTN is not executed, the subsystem is not active.
  - If the subsystem does not provide another way to accomplish initialization processing, the subsystem is left in limbo.
  - Typically requires zapping, IPL, or defining the subsystem with a different name to resolve.

Usage & Invocation – Initialization Processing (cont.)

- In z/OS V2R2, if the INITRTN cannot be found or is not APF-authorized, the subsystem is not defined.
  - This affects any INITRTN defined via IEFSSNxx, the SETSSI ADD command, and IEFSSI macro.
  - No manual “recovery” action is needed to remove the subsystem.
    - If the subsystem can be defined dynamically, either using SETSSI ADD or other method appropriate for the subsystem, you should be able to do it.
- This only affects the cases where the INITRTN cannot be found or is not APF-authorized.
  - ABENDs in the INITRTN, errors in the INITPARM, or subsystems that do not use an INITRTN are not affected.

Usage & Invocation – Initialization Processing Messages

- In the past, these INITRTN problems would have been accompanied by one of the following messages:

  ```
  IEE859I SUBSYSTEM xxxx NOT INITIALIZED - yyyyyyyy  NOT
  FOUND
  IEFJ004I SUBSYSTEM xxxx NOT INITIALIZED - yyyyyyyy
  NOT FOUND
  ```

- In z/OS V2R2, these messages have been replaced with:

  ```
  IEFJ027I SUBSYSTEM INITIALIZATION ROUTINE yyyyyyy NOT
  FOUND FOR SUBSYSTEM xxxx
  ```

Usage & Invocation – SSI DELETE processing

- Of course, INITRTN problems are not the only problems that can occur with a subsystem.
- In z/OS V2R2, there is a new SETSSI DELETE command to remove a subsystem.
- This command performs a logical deletion of the subsystem.
  - Subsystem routines that are processing at the time of the command are not terminated.
  - Any storage associated with the subsystem (load modules, control blocks, etc.) is not freed.

Usage & Invocation – SSI DELETE syntax

- Syntax:
  `SETSSI DELETE,S|SUB|SUBNAME=ssss,FORCE`
  - Note that FORCE is a required keyword.
  - Example:
    `SETSSI DELETE,S=SUB1,FORCE`
- The subsystem name may be enclosed in single quotes if it contains non-standard characters.
  - Example:
    `SETSSI DELETE,S='abcd',FORCE`
- A hexadecimal syntax is also accepted.
  - Example:
    `SETSSI DELETE,S='C1C2C3F1'X,FORCE`
  - This syntax requires all 4 bytes (8 characters, 0-9,A-F) to be specified. If needed, specify trailing blanks ('40'X).

Usage & Invocation – SSI DELETE restrictions

- Restrictions and notes:
  - You may not delete the MSTR subsystem or the primary subsystem (JES2/JES3). If you attempt to delete either of those, you will get the following message:
    ```
    IEFJ033I COMMAND REJECTED BECAUSE SUBSYSTEM
    subname IS NOT ELIGIBLE FOR DELETION
    ```
  - The subsystem is not required to be a dynamic subsystem to be deleted.
  - Individual subsystems may have their own requirements with regards to deletion. You may be required to take additional actions to delete a specific subsystem. Some subsystems may not support deletion at all, or may support deletion but cannot be dynamically added.
  - SETSSI DELETE is a “force” command. Removing an active subsystem can have dangerous side effects. Consider it to be a command of last resort. Use at your own risk.

Usage & Invocation – SSI DELETE detailed processing

- What happens when a subsystem is deleted?
  - The SSI attempts to deactivate the subsystem so that its function routines no longer receive control.
  - Various internal control blocks and queues are updated to reflect that the subsystem no longer exists.
  - If necessary, create an new dummy subsystem.
    - The subsystem name is !DMY, and it is an inactive subsystem.
  - A new SSCVT, which represents a subsystem, is created to replace the subsystem to be deleted.
    - The subsystem name is !DEL, and it is an inactive subsystem.
  - The !DEL SSCVT is swapped onto the SSCVT chain, replacing the SSCVT for the subsystem that is deleted.
  - If the subsystem has an EVENTRTN (new in z/OS V2R2), it is called to notify the subsystem that it has been deleted.
    - More information on EVENTRTNs to follow.

Usage & Invocation – SSI DELETE detailed processing (cont.)

- Because of serialization concerns and SSI functionality, the SETSSI command will create the !DEL subsystem and possibly the !DMY subsystem
  - A !DEL subsystem is created for every subsystem that is deleted.
    - It is necessary to have a “placeholder” subsystem to replace the subsystem that is deleted.
  - The !DMY subsystem is only created when you delete the last subsystem that was defined.
    - This is necessary to maintain serialization on SSI control blocks.
  - If you delete multiple subsystems, you may have multiple !DMY and !DEL subsystems.
  - Any DISPLAY SSI or IEFSSI REQUEST=QUERY will include the ! DMY or !DEL subsystems.

Usage & Invocation – SETSSI command security processing

- The SETSSI command is protected by security profile `MVS.SETSSI.request.ssnm` in the OPERCMDS class. Some of these changes may affect existing security profiles. Depending on how your installation defines security profiles, you may have actions to take.

- The request portion of the profile may be DELETE, in addition to the existing ADD, ACTIVATE, and DEACTIVATE.
  - Example profile names:

```
MVS.SETSSI.ADD.SUB1
MVS.SETSSI.DELETE.SUB2
```

Usage & Invocation – SETSSI command security processing (cont.)

- The ssnm portion of the profile may include characters that are not supported by your security product's command processors:
  - If your subsystem name includes lower case characters, use upper case characters when defining the resource name. For example, use `MVS.SETSSI.*.ABCD` for subsystem name “abcd”.
  - If your subsystem name includes “*”, “&”, or “%”, use the underscore character (“_”) in the resource name. For example, use `MVS.SETSSI.*.SUB_` for subsystem name “SUB%”.
  - If your subsystem name includes embedded blanks, use the underscore character (“_”) in the resource name. Trailing blanks do not use the underscore. For example, use `MVS.SETSSI.*.A_BC` for subsystem “A BC”, but use `MVS.SETSSI.*.A` for subsystem “A”.
  - For any other unsupported characters, define profiles with generic characters.

Usage & Invocation – Subsystem services

- A subsystem can optionally be notified when an “interesting” event occurs.
  - The SSI provides an EVENTRTN that can get control for subsystem events.
    - At this time, only the DELETE event is supported. However, the EVENTRTN is designed so that other events can be supported I the future.
  - An EVENTRTN is only available for dynamic subsystems (those subsystems defined using keyword-format IEFSSNxx parmlib members, the SETSSI command, or the IEFSSI macro.)
  - The EVENTRTN can only be defined through the IEFSSI macro:

```
IEFSSI SUBNAME=subname,
       REQUEST=OPTIONS,
       EVENTRTN=exitname
```

    - This would typically be done by the subsystem during initialization.

Usage & Invocation – Subsystem services (cont.)

- The environment for the EVENTRTN is similar to the INITRTN (key 0, supervisor state.) The EVENTRTN should be AMODE 31.
- The DELETE event occurs AFTER the subsystem has been deleted.
  - The subsystem no longer exists at this point.
- IEFSSI services may not be available from the EVENTRTN.
- The parameter list is mapped by macro IEFJSEPL.
- If you create an EVENTRTN, make sure you check the event type before doing any processing and ignore any event types that you don't recognize.

Usage & Invocation – DISPLAY SSI processing

- DISPLAY SSI was enhanced to show the subsystem name in hex:

```
IEFJ100I  15.25.44  SSI DISPLAY
SUBSYS=JES2 (PRIMARY) HEX=D1C5E2F2
   DYNAMIC=YES     STATUS=ACTIVE      COMMANDS=REJECT
SUBSYS=SDB1           HEX=E2C4C2F1
   DYNAMIC=YES     STATUS=ACTIVE      COMMANDS=REJECT
```

- DISPLAY SSI,ALL also includes the presence of an EVENTRTN:

```
IEFJ100I  15.25.44  SSI DISPLAY
 SUBSYS=SDB1           HEX=E2C4C2F1
    DYNAMIC=YES     STATUS=ACTIVE     COMMANDS=REJECT
  FUNC=236 237 239 240
  EVENTRTN=YES
```

Installation

- Although no actions need to be taken during installation, be aware:
  - The security profile for the SETSSI command has changed. You may need to review your security profiles to prevent DELETE commands from being issued or to handle subsystem names that include unusual characters.
  - INITRTN processing for subsystems has changed so that subsystems are not defined if the INITRTN cannot be found. Although this is probably a desirable change, it is different than prior releases.

Presentation Summary

- In review, z/OS V2R2 contains:
  - Updates to INITRTN processing to check if the INITRTN exists, and skip defining the subsystem if it does not.
  - A SETSSI DELETE command to logically delete a subsystem.
  - Other SSI infrastructure updates related to logical subsystem deletion, including updates to the DISPLAY SSI command and services available to dynamic subsystems.

Appendix

- Key publications:
  - z/OS MVS Using the Subsystem Interface (SA38-0679)
  - z/OS MVS System Commands (SA38-0666)
  - z/OS MVS System Messages (SA38-0675)
  - z/OS MVS Programming: Authorized Assembler Services Reference Vol. 2 (SA23-1373)