

# OS/390 New Users

Stop those Unnecessary IPLs:  
the Dynamic OS/390 Functions

**Session 2859 - SHARE 94 Technical Conference**

**Steve Schunk**

**Schunk and Associates, Inc. (SNK)**

**s.schunk@schunk-associates.com**

**www.schunk-associates.com**

**Copyright 2000 Schunk and Associates, Inc. (Permission granted to SHARE, Inc. to distribute.)**

  
**Disclaimer:**

This material is provided for your information, critical analysis, and discussion. You need to evaluate whether the methods and techniques presented are useful and appropriate for your particular installation and operation environment.

**Trademarks:**

Are the property of their respective owners.

# Agenda

- **Objectives:**

This is a Basic (NOVICE) Systems Management Session.

- **Summary.**

- **Questions.**

## Bibliography

### Systems Management:

OS/390 MVS JCL User's Guide	(GC28-1758)
OS/390 MVS JCL Reference	(GC28-1757)
OS/390 MVS Init and Tuning Guide	(SC28-1751)
OS/390 MVS Init and Tuning Reference	(SC28-1752)
OS/390 MVS Conversion Notebook	(GC28-1747)
OS/390 MVS System Commands	(GC28-1781)
OS/390 HCD User's Guide	(SC28-1848)
OS/390 HCD Planning	(GC28-1750)
TSO/E System Programming Command	(SC28-1972)

## Reasons for an IPL

Three major reasons for a **scheduled** IPL.

- Software Change
- System Parameter Change
- Hardware Change

But this usually means incurring an outage. Installations today are looking to achieve or, come as close as possible, to achieving 24 X 7 availability.

In this session we will discuss how you can control implementation of these types of changes, while avoiding IPLs.

## Software Change: Reasons

- Software Product Installation, or Upgrade, etc.
- Emergency Software Maintenance
- ISV Authorization Code change
- User Customization change

## Software Change: System Search Order

When a program requests a module, the system searches for the requested module in various system areas and libraries, in the following order:

1. Modules that were loaded under the current task (Load List Elements)
2. The job pack area (JPA)
3. Tasklib, steplib, joblib, or any libraries that were indicated by a DCB specified as an input parameter to the macro used to request the module (LINK, LINKX, LOAD, ATTACH, ATTACHX, XCTL or XCTLX)
4. Active Link Pack Area (LPA), which contains the FLPA and MLPA
5. Pageable link pack area (PLPA)
6. SYS1.LINKLIB and libraries concatenated to it through LNKLST

## Software Change: System Search Order

- When searching TASKLIBs, STEPLIBs, JOBLIBs, a specified DCB, or the LNKLST concatenation for a module, the system searches each data set directory for the first directory entry that matches the name of the module.
- The directory is located on DASD with the data set, and is updated whenever the module is changed.
- The directory entry contains information about the module and its location in storage.

## Software Change: System Search Order

- Fixed LPA, Modifiable LPA, and Pageable LPA comprise the Link Pack Area, an area of storage that contains re-enterable routines that are loaded at IPL time and can be used concurrently by all tasks in the system.
- LNKLST is a user-specified set of libraries that form part of the search order the systems uses to locate programs.
- LLA (LNKLST Lookaside) caches, in its address space, a copy of the directory entries for the libraries it manages. For modules that reside in LLA-managed libraries, the system can quickly search the directories in virtual storage instead of using I/O to search the directories on DASD.

**LPA, LNKLST and LLA are defined and initiated at IPL time.**

## Software Change: IPL Avoidance

- JOBLIB/STEPLIB DD Statement
- LLA: Modify LLA (LNKLST LookAside)
- LNKLST: Dynamic LNKLST
- LPA: Dynamic LPA
- EXIT: Dynamic EXIT

## Software Change: Joblib DD

Use a JOBLIB DD statement to define a private library that the system is to use for an entire job.

However, if you include a JOBLIB DD statement for the job and a STEPLIB DD statement for an individual job step, the system first searches the step library and then the system library for the program requested in the EXEC statement. The system ignores the JOBLIB library for that step.

```
//PAYROLL JOB JONES,CLASS=C  
//JOBLIB DD DSNAME=PRIVATE.LIB4,DISP=SHR  
//STEP1 EXEC PGM=SCAN  
//STEP2 EXEC PGM=UPDATE
```

The private library requested on the JOBLIB DD statement is cataloged. The system passes catalog information to subsequent job steps. The system searches for the programs SCAN and UPDATE first in PRIVATE.LIB4, then in SYS1.LINKLIB. DD statement DD1 refers to the private library requested in the JOBLIB DD statement.

## Software Change: Steplib DD

Use a STEPLIB DD statement to define a private library for one job step in a job.

```
//PAYROLL JOB BROWN,MSGLEVEL=1
//STEP1 EXEC PROC=LAB14
//STEP2 EXEC PGM=SPKCH
//STEPLIB DD DSN=PRIV.LIB5,DISP=SHR
//STEP3 EXEC PGM=TIL80
//STEPLIB DD DSN=PRIV.LIB12,DISP=SHR
```

The system searches PRIV.LIB5 for the program SPKCH and PRIV.LIB12 for TIL80. The system catalogs both private libraries. Subsequent job steps in the same job may refer to or receive a private library defined on a STEPLIB DD statement. Also, you can place a STEPLIB DD statement in an in-stream or cataloged procedure.

## Software Change: Modify LLA

- When an LLA-managed library is updated, the version of a module that is located by a directory entry saved in LLA will differ from the version located by the current directory entry on DASD for that module.
- If you update a load module in a library that LLA manages, affect the update to the system, by issuing the appropriate form of the MODIFY LLA command to refresh LLA's cache with the latest version of the directory information from DASD.
- If LLA is NOT refreshed, to re-cache the directory information, the system will continue to use the un-updated version of the load module.
- Use the MODIFY LLA command to cause the library lookaside (LLA) program to build a new copy of EITHER all or only part of the library directory indexes.

## Software Change: Modify LLA

The syntax for this command is:

```
F LLA,{REFRESH      }  
      {UPDATE=xx    }
```

- **MODIFY LLA,REFRESH**

- Rebuilds LLA's directory for the entire set of libraries managed by LLA.
- This action is often called a complete refresh of LLA.

- **MODIFY LLA,UPDATE=xx**

- Rebuilds LLA's directory only for specified libraries or modules.
- xx identifies the CSVLLAxx member that contains the names of the libraries for which directory information is to be refreshed.
- This action is often called a selective refresh of LLA.

## Software Change: Modify LLA

### F LLA,REFRESH

CSV210I LIBRARY LOOKASIDE REFRESHED

### F LLA,UPDATE=00

CSV210I LIBRARY LOOKASIDE UPDATED

## Software Change: Coding SYS1.PARMLIB(PROG##)

EXAMPLE : SYS1.PARMLIB(PROG00) To support Dynamic LNKST, APF, and EXITS.

### **APF FORMAT(DYNAMIC)**

<b>APF ADD DSNAME</b> (SYS1.LINKLIB)	VOLUME(&SYSR1)
APF ADD DSNAME(ISP.SISPLPA)	VOLUME(&SYSR1)
APF ADD DSNAME(SYS3.NCP.LOADLIB)	SMS
APF ADD DSNAME(SYS1.EMPTY.LOADLIB)	VOLUME(*MCAT*)
APF ADD DSNAME(SYS1.DSN410P.SDSNLINK)	VOLUME(&SYSR4)
APF ADD DSNAME(DSN410.RUNLIB.LOAD)	VOLUME(DB2TEC)

### **SYSLIB LINKLIB**(SYS2.NYPHLOAD)

SYSLIB LPALIB(SYS2.NYPHLPA)

### **LNKST DEFINE** NAME(PROD)

**LNKST ADD** NAME(PROD) DSNAME(SYS1.LINKLIB)

LNKST ADD NAME(PROD) DSNAME(SYS3.APPL.LINKLIB) VOLUME(SYS001)

LNKST DEFINE NAME(PROD.IPL) COPYFROM(PROD)

### **LNKST ACTIVATE** NAME(PROD)

**EXIT ADD** EXITNAME(SYS.IEFU83)

MODNAME(B88U83A) DSNAME(ISV1.BETA.APFLOAD)

EXIT ADD EXITNAME(SYSSTC.IEFU83)

MODNAME(B88U83A) DSNAME(ISV1.BETA.APFLOAD)

## Software Change: DYNAMIC LNKLST

Dynamic LNKLST supports modification of LNKLST, after an IPL, as follows:

- Define a LNKLST set of data sets for the LNKLST concatenation.
- Add data sets to or delete data sets from the LNKLST set.
- Remove the definition of a LNKLST set from the system.
- Test for the location of a specific module in the LNKLST concatenation.
- Activate a LNKLST set as the LNKLST concatenation for the system.
- Update an address space for jobs to use a LNKLST set.

### Dynamic LNKLST changes are not retained across IPLs.

To implement a change permanently, it is necessary to make the change in the **PROGxx** member of SYS1.PARMLIB.

## Software Change: DYNAMIC LNKLST

- The LNKLST set is defined at IPL time.
- Multiple LNKLST sets can be defined, however, only **ONE LNKLST** may be the CURRENT LNKLST set at any given time.
- The CURRENT LNKLST may not be modified...
  - It is necessary to create a new LNKLST
  - Modify the new LNKLST
  - Activate this new LNKLST to make it the new CURRENT LNKLST
- LNKLST set can be dynamically changed by using the SETPROG, LNKLST command.
- When a new LNKLST is activated, LLA is automatically refreshed. The new current LNKLST set will then be managed by LLA as the LNKLST.

## Software Change: DYNAMIC LNKLST

- Data sets from the previously current LNKLST continue to be managed by LLA, but only on an individual data set basis.
- The only way to remove them is to either Stop and Start LLA, or issue the MODIFY LLA,REFRESH or MODIFY LLA,UPDATE and specify the library to be removed.



## Software Change: DYNAMIC LNKLST

Replacing the version of a product in LKNLST :

This is an example PROGxx member used to create and update a new LNKLST set. After coding these statements in the PROGxx member, issue the SET PROG=xx command to invoke.

```
LNKLST DEFINE NAME(PROD01) COPYFROM(PROD)
LNKLST DELETE NAME(PROD01) DSNAME(ISV1.QUICKREF.V5R2M0.LINKLIB)
LNKLST ADD NAME(PROD01) DSNAME(ISV1.QUICKREF.V5R5M1.LINKLIB)
LNKLST ACTIVATE NAME(PROD01)
```

1. Create a new LNKLST set from the CURRENT LNKLST set.
2. Remove the old version product library from the new LNKLST set.
3. Add the new version product library to the new LNKLST set.
4. Activate this new LNKLST set.

## Software Change: DYNAMIC LNKLST

Replacing the version of a product in LKNLST:

This is an example of SETPROG commands, input from a system console, used to create and update a new LNKLST set . Each function is performed as each command is entered from the console.

```
SETPROG LNKLST,DEFINE, NAME=PROD01, COPYFROM=PROD  
SETPROG LNKLST,DELETE,NAME=PROD01,DSNAME=ISV1.QUICKREF.V5R2M0.LINKLIB  
SETPROG LNKLST,ADD,NAME=PROD01,DSNAME=ISV1.QUICKREF.V5R5M1.LINKLIB  
SETPROG LNKLST, ACTIVATE,NAME=PROD01
```

1. Create a new LNKLST set from the CURRENT LNKLST set.
2. Remove the old version product library from the new LNKLST set.
3. Add the new version product library to the new LNKLST set.
4. Activate this new LNKLST set.

## Software Change: Dynamic LNKLST

**SETPROG LNKLST DEFINE,NAME=TEST,COPYFROM=PROD**

CSV500I LNKLST SET TEST HAS BEEN DEFINED

**SETPROG LNKLST DELETE,NAME=TEST,DSNAME=ISV1.QUICKREF.V5R2M0.LINKLIB**

CSV501I DATA SET ISV1.QUICKREF.V5R2M0.LINKLIB

HAS BEEN DELETED FROM LNKLST SET TEST

**SETPROG LNKLST ADD,NAME=TEST,DSNAME=ISV1.QUICKREF.V5R5M1.LINKLIB**

CSV501I DATA SET ISV1.QUICKREF.V5R5M1.LINKLIB

HAS BEEN ADDED TO LNKLST SET TEST

**SETPROG LNKLST ACTIVATE,NAME=TEST**

CSV500I LNKLST SET TEST HAS BEEN ACTIVATED

## Software Change: DYNAMIC LPA

Dynamic LPA supports modification of LPA, after an IPL, as follows...

- Modules that are to be added to the LPA following IPL.
- Modules that are to be deleted from the LPA following IPL.
- Threshold values for minimum amounts of CSA storage that still must be available after an ADD operation.

Dynamic LPA changes are not retained across IPLs.

To implement a change permanently, it is necessary to make the change in the **LPALSTxx** member of SYS1.PARMLIB.

## Software Change: DYNAMIC LPA

- LPA is searched in this order:
  - Dynamic LPA modules, as specified in PROGxx members
  - Fixed LPA (FLPA) modules, as specified in IEAFIXxx members
  - Modified LPA (MLPA) modules, as specified in IEALPAXx members
  - Pageable LPA (PLPA) modules, loaded from libraries specified in LPALSTxx or PROGxx

**A module added to LPA dynamically will be found BEFORE one of the same name added to LPA during IPL.**

## Software Change: DYNAMIC LPA

- Modules that are added to the LPAALST using the dynamic LPA function are placed in CSA or ECSA. The CSAMIN parameter, can be used to ensure that a certain minimum amount of CSA and ECSA will be available after a module is added.
- Confirm the product supports dynamic LPA.
- Dynamic LPA can only delete modules that were added dynamically. It does not support deletion of modules from PLPA, FLPA or MLPA.
- Storage used by a dynamically ADDED module, will be freed when/if that module is dynamically DELETED.

## Software Change: DYNAMIC LPA

The syntax for this command is:

```
SETPROG LPA,{ADD,<MODNAME=(modname...,modname) | MASK=mask }  
    {      ,DSNAME=<dsname | LNKLST>                                }  
    {      <,<FIXED> <,<PAGEPROTPAGE>                                }  
{DELETE,MODNAME=(modname...,modname)                                }  
{      FORCE=YES <CURRENT | OLDEST>                                }  
{CSAMIN=(below,above)                                            }
```

## Software Change: DYNAMIC LPA

To add a modified version of a module to replace an existing LPA version:

```
SETPROG LPA,ADD,MODNAME=IGC0024G,DSNAME=SYS2.NYPHLPA
```

To delete this same modified version of the module, and revert to the LPA version:

```
SETPROG LPA,DELETE,MODNAME=IGC0024G
```

## Software Change: DYNAMIC EXIT

Use the SETPROG EXIT command to do the following:

- Add an exit routine to an exit
- Change the state of an exit routine
- Delete an exit routine from an exit
- Undefine an implicitly-defined exit
- Change the attributes of an exit.

You can use the SETPROG EXIT command to control exits that have been defined to the dynamic exits facility.

## Software Change: DYNAMIC EXIT

You can use the SETPROG EXIT command to control exits that have been defined to the dynamic exits facility.

Dynamic exits services are implemented by:

- The EXIT statement of the PROGxx parmlib member. The EXIT statement of PROGxx allows an installation to add exit routines to an exit, delete an exit routine for an exit, change the state of an exit routine, change the attributes of an exit, and undefine an implicitly defined exit.
- The PROGxx EXIT statement interacts with the PROG=xx parameter of IEASYSxx and the SET PROG=xx command. At IPL, you can use PROG=xx to specify the particular PROGxx parmlib member the system is to use. During normal processing, you can use the SET PROG=xx command to set a current PROGxx parmlib member.
- The SETPROG EXIT command.

## Software Change: DYNAMIC EXIT

The syntax for this command is:

```
SETPROG EXIT,{ADD,EXITNAME=exitname,MODNAME=modname      }
      { < ,STATE={ACTIVE|INACTIVE}>                        }
      { < ,DSNAME=dsname>                                   }
      { < ,JOBNAME={jobname|*}>                             }
      { < ,ABENDNUM=(n< ,CONSEC>)>                          }
      { < ,FIRST|LAST>                                     }
      {ATTRIB,EXITNAME=exitname,KEEPRC=(compare,kk)      }
      {DELETE,EXITNAME=exitname,MODNAME=modname          }
      { < ,FORCE={YES|NO}>                                  }
      {MODIFY,EXITNAME=exitname,MODNAME=modname           }
      { < ,STATE={ACTIVE|INACTIVE}>                        }
      { < ,JOBNAME={jobname|*}>                             }
      {UNDEFINE,EXITNAME=exitname                         }
```

## Software Change: DYNAMIC EXIT

- Associate exit routine MYMOD with the SMF exit known as SYS.IEFUJI, defined through the SYS statement in a SMFPRMxx parmlib member.
- The load module is in data set MY.DSN.

```
SETPROG EXIT,ADD,EXITNAME=SYS.IEFUJI,MODNAME=MYMOD,  
          DSNAME=MY.DSN,STATE=ACTIVE
```

## System Parameter Change: Reasons

- Software Product Installation, or Upgrade, etc.
- Emergency Software Maintenance
- ISV Authorization Code change
- User Customization change

## System Parameter Change: IPL Avoidance

- SETLOAD: Modify PARMLIB concatenation dynamically
- SETSSI: Add Subsystems dynamically
- RACF Database: Switch/Rename RACF Database
- PAGE Data Sets: ADD/REMOVE Page Data sets
- Catalogs: ALLOCATE(OPEN)/CLOSE ICF User Catalogs
- TSO PARMLIB: TSO/E

## System Parameter Change: SETLOAD

In OS/390 it is now possible to concatenate PARMLIB.

- At IPL you can concatenate up to 10 parmlib datasets in the LOADxx member using the PARMLIB keyword.
- The SETLOAD command allows you to switch dynamically from one parmlib concatenation (logical parmlib) to another without having to initiate an IPL.
- The SETLOAD command specifies the LOADxx member that contains the PARMLIB statements to use for the switch.
- The SETLOAD command specifies LOADxx member containing the new PARMLIB concatenation to be dynamically activated.
- Use the 'D PARMLIB' command to display the PARMLIB concatenation before and after using the SETLOAD.

## System Parameter Change: SETLOAD

There are two issues to be aware of when using concatenated PARMLIBs.

- The first issue is to keep track of possible like-named parmlib members that are going to be used from a different parmlibs in the concatenation, which have different contents, so as not to invoke the incorrect member.

In other words, be careful with same named members.

- The second issue is the implementation of PARMLIB support using the LOADxx member.
  - For most keywords, the keyword value on the last filter to match the environment is the value that will be used.
  - With the PARMLIB keyword, libraries specified on every PARMLIB keyword that apply to this system are concatenated in the **order they are processed**.
  - Therefore, it is recommended that the PARMLIB keywords be coded with the system (or LPAR) specific libraries first, followed by the SYSPLEX or machine specific libraries.
  - This will ensure that the appropriate PARMLIB members are invoked for that LPAR.

## System Parameter Change: SETLOAD

The syntax for this command is:

```
SETLOAD xx,PARMLIB      <,{DSNAME|DSN}=dsn>  
                        <,{VOLUME|VOL|VOLSER}=vol>
```

## System Parameter Change: SETLOAD

This command tells the system to process the PARMLIB statements in member LOAD02, which resides in a data set in the existing parmlib concatenation.

**SETLOAD 02,PARMLIB**

This command tells the system to process the PARMLIB statements in member LOAD03. Member LOAD03 resides in the data set 'sys4.parmlib'. Data set 'sys4.parmlib' is catalogued in the master catalog.

**SETLOAD 03,PARMLIB,DSN=sys4.parmlib**

This command tells the system to process the PARMLIB statements in member LOAD04. Member LOAD04 resides in the data set 'sys5.parmlib' which can be located on volume '123456'.

**SETLOAD 04,PARMLIB,DSN=sys5.parmlib,VOL=123456**

## System Parameter Change: SETSSI

A subsystem is a service provider that performs one or many functions, but does nothing until it is requested. Although the term "subsystem" is used in other ways, a subsystem must be the master subsystem (the master subsystem (MSTR) is a part of MVS and is not user-defined) or be defined to MVS in one of the following ways:

- Processing the IEFSSNxx parmlib member during IPL
- Issuing the SETSSI system command

Some examples of IBM-supplied subsystems that use the SSI:

- JES2, Netview, OPC

There are two types of subsystems:

- The primary subsystem, or the job entry subsystem that MVS uses to do work. It can be either JES2 or JES3.
- Secondary subsystems. Secondary subsystems provide functions as needed by IBM products, vendor products, or the installation. MVS communicates with subsystems through the SSI.

## System Parameter Change: SETSSI

Use the SETSSI to dynamically add, activate or deactivate a subsystem.

- You can issue the SETSSI command from one of the following:
  - A console that has master authority
  - A console to which an operator with sufficient RACF authority has logged on.
- If you issue a SETSSI ACTIVATE or DEACTIVATE command for a subsystem that does not allow SETSSI commands, the system ignores the command and issues an error message to the console.
- The SETSSI ADD command allows any subsystem except the primary subsystem to be dynamically defined.

## System Parameter Change: SETSSI

The syntax for this command is:

```
SETSSI {ADD,{SUBNAME|SUB|S}=subname
      < , {CONSNAME|C}=consname >
      < , {INITRTN|I}=initrtn< ,{INITPARM|P}=initparm > > }
      {DEACTIVATE|DEACT},{SUBNAME|SUB|S}=subname }
      {ACTIVATE|ACT},{SUBNAME|SUB|S}=subname }
```

## System Parameter Change: SETSSI

To define the 'CAW' subsystem to the system, call its initialization routine and pass the specified parameter to the initialization routine:

```
SETSSI ADD,SUBNAME=CAW,INITRTN=CAWINIT,INITPARM=HELLO
```

To temporarily stop new function requests to the subsystem to see if one of the function routines in the 'CAW' subsystem is causing abends:

```
SETSSI DEACTIVATE,SUBNAME=CAW
```

```
SETSSI DEACTIVATE,SUBNAME=FFST
```

```
IEFJ023I SETSSI DEACTIVATE COMMAND FOR SUBSYSTEM FFST COMPLETED  
WITH ERRORS
```

```
IEFJ036I SUBSYSTEM FFST IS NOT ENABLED FOR THE SETSSI COMMAND
```

## System Parameter Change: RACF Database Switch

USE THE RACF COMMAND to perform RACF Database Activate/Deactivate.

- Switch from using a specific primary database to using its corresponding backup database, perhaps because of a failure related to the primary database.
- Deactivate or reactivate primary or backup RACF databases.
  - Deactivating a specific primary database causes all RACF requests for access to that database to fail.
  - Deactivating a specific backup database causes RACF to stop duplicating information on that database.)
- When you deactivate a RACF database (using RVARY INACTIVE) or switch to a backup RACF database (using RVARY SWITCH), RACF automatically de-allocates the database.

## System Parameter Change: RACF Database Switch

- To reactivate a data set use the RVARY ACTIVE command; RVARY SWITCH will not activate an inactive data set. RACF automatically reallocates that database.
- This feature allows you to restore the database from a copy on tape or re-catalog the database on another volume without having to re-IPL your system.
- If you deactivate the primary RACF database and uncatalog it and replace it with an alternate database, the alternate database must be cataloged and have the same name as the original database before you can activate it. When you deactivate (and de-allocate) a RACF database, you can move the database from one direct access storage device to another.
- Before re-cataloging a database, you must first deactivate the database by issuing either the RVARY INACTIVE or the RVARY SWITCH command.

## System Parameter Change: RACF Database Switch

The syntax for this command is:

```
subsystem-prefix RVARV
    ACTIVE
    | INACTIVE NOCLASSACT(class-namelist| *)
    (NOTAPE)
    | DATASHARE | NODATASHARE
    | SWITCH
    DATASET(database-namelist | *)
    LIST | NOLIST
```

## System Parameter Change: RACF Database Switch

Operator wants to temporarily deactivate and de-allocate RACF to make repairs to a particular primary RACF database. The RACF subsystem prefix is #.

```
#RVARY SWITCH,DATASET(RACF.PRIM1)
```

```
ICH15013I RACF DATABASE STATUS:
```

ACTIVE	USE	NUMBER	VOLUME	DATASET
-----	-----	-----	-----	-----
NO	PRIM	1	*DEALLOC	RACF.PRIM1
NO	BACK	1	D94RB1	RACF.BACK1
YES	PRIM	2	D94RP2	RACF.PRIM2
NO	BACK	2	D94RB2	RACF.BACK2
YES	PRIM	3	D94RP3	RACF.PRIM3
NO	BACK	3	D94RB3	RACF.BACK3

Output following Deactivation and De-allocation of RACF.PRIM1

## System Parameter Change: RACF Database Switch

Operator wants to switch from using the primary database to using the backup database. The appropriate backup database is active, and the RACF subsystem prefix is #.

**#RVARY ACTIVE,DATASET(RACF.BACK1)**

**ICH15013I RACF DATABASE STATUS:**

<b>ACTIVE</b>	<b>USE</b>	<b>NUMBER</b>	<b>VOLUME</b>	<b>DATASET</b>
-----	-----	-----	-----	-----
NO		PRIM	1 *DEALLOC	RACF.PRIM1
<b>YES</b>		<b>BACK</b>	<b>1 D94RB1</b>	<b>RACF.BACK1</b>
YES		PRIM	2 D94RP2	RACF.PRIM2
NO		BACK	2 D94RB2	RACF.BACK2
YES		PRIM	3 D94RP3	RACF.PRIM3
NO		BACK	3 D94RB3	RACF.BACK3

**Output following the Activation of RACF.BACK1**

## System Parameter Change: PAGE Dataset ADD

PAGEADD adds auxiliary storage space (local page data sets) to the system. The page data sets added remain available to the system until you IPL with the CLPA (create link pack area) option, IPL with the CVIO (clear virtual I/O) option, or issue the PAGEDEL command. PAGEADD can also direct VIO pages away from a page data set that is being added.

You might need to add auxiliary storage space if any of the following conditions exist:

- The planned system load increases.
- The space provided during system initialization proves insufficient.
- Space is lost because of a hardware failure.

The number of page data sets that can be in use by the system is limited by the number specified on the PAGTOTL system parameter (see OS/390 MVS Initialization and Tuning Guide). If these limits are exceeded, the system issues a message, and you cannot add any more data sets of that type for the duration of this IPL.

## System Parameter Change: PAGE Dataset ADD

- The number of page data sets that can be in use by the system is limited by the number specified on the PAGTOTL system parameter (see OS/390 MVS Initialization and Tuning Guide). If these limits are exceeded, the system issues a message, and you cannot add any more page data sets during this IPL.
- The page data sets must be defined before you can issue the PAGEADD command. If the volume containing the data set is not mounted before you enter the command, the system issues a mount message.
- If the system detects a shortage of available auxiliary storage space, it issues the following message:

### **IRA200I AUXILIARY STORAGE SHORTAGE**

- The system rejects LOGONs and START commands until the shortage is relieved. If the shortage increases, the following message is issued:

### **IRA201I CRITICAL AUXILIARY STORAGE SHORTAGE**

- The system rejects LOGONs and START commands and might delay the starting of certain initiators until the shortage is relieved.

## System Parameter Change: PAGE Dataset ADD

The syntax for this command is:

```
PA {<PAGE=>}{dsname<,dsname>... }  
   {SWAP= }  
   {NONVIO= }
```

## System Parameter Change: PAGE Data set ADD

To add one page data set:

**PA PAGE=sys1.page3**

To add three page data sets:

**PA PAGE=sys1.page01,sys1.page02,sys1.page3**

To add SYS1.PAGE01 as a page data set, not to be used for VIO paging:

**PA NONVIO=sys1.page01**

**PA PAGE=PAGE.NPT2.LOCAL3**

**IEE783I PAGEADD COMMAND-**

**PAGE.NPT2.LOCAL3 PAGE DATA SET**

**NOW AVAILABLE FOR SYSTEM USE**

## System Parameter Change: PAGE Data set Delete

PAGEDEL may be used to delete, replace, or drain local page data sets. This command allows local page data sets removed or replaced without requiring an IPL.

You might need to delete, replace or drain local page data sets for any of the following reasons:

- The hardware is being reconfigured.
- The hardware is generating I/O errors.
- The page or swap configuration is being changed.
- System tuning requires the change.

**Misuse of this command can seriously impact system performance.**

## System Parameter Change: PAGE Data set Delete

- You cannot use PAGEDEL to delete, replace, or drain the PLPA, common, duplex, or the last local page data sets.
- When you enter a PAGEDEL command, the system issues a highlighted, non-rollable message to indicate that the command is accepted. The message remains on the console screen until the PAGEDEL command completes.
- When you enter a PAGEDEL command while a PAGEDEL command is already in progress, the system issues a message to indicate that the command is not accepted.
- The system rejects a PAGEDEL command that decreases the amount of auxiliary storage below a fixed percentage of the available auxiliary storage.
- To identify the page and swap data sets the system is currently using or the status of the PAGEDEL command, issue the DISPLAY ASM command.

## System Parameter Change: PAGE Data set Delete

The syntax of this command is:

```
PD{DELETE,{PAGE|SWAP}={dsname< ,dsname> ...}          }  
   {REPLACE,{PAGE|SWAP}={{(dsname,rdsname)< ,(dsname,rdsname)> ...}}  
   {DRAIN,{PAGE|SWAP}={dsname< ,dsname> ...}          }
```

## System Parameter Change: PAGE Dataset Delete

To delete a local page data set:

```
PD DELETE,PAGE=sys1.page3
```

To delete three local page data sets:

```
PD DELETE,PAGE=sys1.page01,sys1.page02,sys1.page3
```

To replace SYS1.PAGE01, a local page data set, and specify SYS1.PAGE04 to replace it:

```
PD REPLACE,PAGE=(sys1.page01,sys1.page04)
```

```
PD DELETE,PAGE=PAGE.NPT2.LOCAL3
```

```
IEE205I PAGEDEL COMMAND -
```

```
LOCAL PAGE DATA SET PAGE.NPT2.LOCAL3 DELETED
```

## System Parameter Change: CATALOG CLOSE/ALLOCATE

Use the MODIFY CATALOG command to communicate with the catalog address space to display information or request a specified service.

When an operator issues any MODIFY CATALOG command (except for MODIFY CATALOG,RESTART), messages return to that console exclusively.

For MODIFY CATALOG,RESTART, both the master console and the console issuing the command receive messages.

## System Parameter Change: CATALOG CLOSE/ALLOCATE

```
F CATALOG,{ABEND{(id)      }
           {  {(yyyyyyyy)}  }
           {  {(ALLOCATE)}  }
           {  {(ANALYSIS)}  }
           {  {(MODIFY) }   }
           {ALIASLEVEL(n)   }
           {ALLOCATE(nnnnnn...)<,NOISC> }
           {                |,NOVLF   }
           {ALLOCATED<(vvvvv)>      }
           {CATMAX(nn)              }
           {CLOSE(nnnnnn...)}
           {DUMPON                  }
           {DUMPOFF                 }
           {END(id)<,REDRIVE  >     }
           {  |,NOREDRIE          }
           {ENTRY<(cname)  >      }
           {  |(mmmmmmmm)         }
           {ISC(nnnnnn...)}
           . . .
```

(. . . Remainder of CATALOG command omitted for visual purposes.)

# System Parameter Change: CATALOG CLOSE/ALLOCATE

## F CATALOG,LIST

IEC351I CATALOG ADDRESS SPACE MODIFY COMMAND ACTIVE

IEC347I LIST ACTIVE CATALOG TASK(S) 478

\*CAS\*\*\*\*\*

\* FLAGS - TASK ADDRESS - JOBNAME / STEPNAME - ELAPSED TIME - ID \*

\*\*\*\*\*

\* ----- NOACTIVE --- NONE / --- 00.00.00 --- \*

\*\*\*\*\*

\* O-OLDEST, W-WAIT, A-ABEND, E-ENQ, R-RECALL, L-RLS \*

\*CAS\*\*\*\*\*

IEC352I CATALOG ADDRESS SPACE MODIFY COMMAND COMPLETED

## System Parameter Change: CATALOG CLOSE

- Closes an integrated catalog facility catalog dynamically, without affecting any existing allocations.
- With this parameter, you can dynamically change catalog attributes, such as share options and the number of strings. In the past, changing these attributes would have required an IPL (to affect the master catalog), or the termination and restart of a job or online system (to affect a user catalog).
- All of the catalog address-space private storage associated with the catalog is freed.
- The catalog is reopened with a new set of control blocks the next time a request is processed for that catalog.
- Rebuilding of the control blocks is transparent to the users of the catalog.

## System Parameter Change: CATALOG ALLOCATE (OPEN)

- OPEN - is obsolete and should no longer be used.
- Use the ALLOCATE sub-parameter instead.
- Allocates an integrated catalog facility catalog to the catalog address space.
- This action makes it unnecessary to dynamically allocate such catalogs in any user address space, which can enhance performance.
- For long-running jobs, it can also make it easier to take catalog devices offline.

## System Parameter Change: TSO PARMLIB

- Display the specifications in the active IKJTSOxx member of SYS1.PARMLIB
- Dynamically change the active member without a re-IPL
- Check the syntax of any IKJTSOxx member of SYS1.PARMLIB
  - one system
  - all systems in a parallel sysplex, or
  - a subset of systems in a parallel sysplex
- Note: You can also place IKJTSOxx members in data sets other than SYS1.PARMLIB.
- This extra flexibility allows you, for example, to separate your data from IBM supplied data. You can specify a list of PARMLIB data sets that will comprise a logical concatenation (logical PARMLIB) for the life of the MVS system (similar to LPALST and LNKLIST for LPA and linklist libraries).

## System Parameter Change: JES2 PROCLIB Concatenation

The active JES2 PROCxx concatenation can be changed by updating the JES2 Procedure JCL and doing a hot start of JES2.

- Hot Start: A hot start is a warm start of an abnormally terminated JES2 member without an intervening IPL.
- JES2 performs a hot start when a particular JES2 member has stopped but other systems have continued to function and have not experienced problems.
- When JES2 hot starts, all address spaces continue to execute as if JES2 had never terminated.
- Jobs that were processing on output devices are re-queued as if a \$I command had been issued.
- Jobs on input devices must be resubmitted and lines must be restarted.
- Hot starts have no affect on other members in a MAS configuration.

## Hardware Change: Reasons

- New or Upgraded Hardware
- Modify Hardware Access across LPARS
- Modify Eligible Device Table

## Hardware Change: IPL Avoidance

Hardware Change: IPL avoidance is possible using the following facility.

- Hardware Configuration Definition: Dynamic I/O reconfiguration  
Hardware and Software

## Hardware Change: IPL Avoidance

- HCD (Hardware Configuration Definition) offers dynamic I/O reconfiguration management.
- The HCD component of MVS consolidates the hardware and software I/O configuration processes under a single interactive, end-user interface.
- The validation checking that HCD does as you enter data into the IODF, helps to eliminate errors before you attempt to use the I/O configuration.
- This function allows you to change your peripheral hardware and software definitions dynamically - such as add devices or change devices, channel paths, and control units without performing a POR or an IPL.
- You may also perform software only changes, even if the hardware is not installed.

## Hardware Change: HCD - Dynamic Reconfiguration

- The IODF (Input/Output Definition File) is a VSAM linear data set that contains I/O definition information and is modifiable using HCD.
- This information in the IODF includes processor I/O definitions (formerly specified by IOCP input streams) and operating system I/O definitions (formerly specified by MVSCP input streams).
- A single IODF can contain several processor and several operating system I/O definitions.
- The IOCDS (Input/Output Configuration Data Set) is a configuration definition built by the I/O Configuration Program (IOCP) and stored on disk files associated with the processor controller.
- The Input/Output Configuration Program (IOCP) is a hardware utility that defines the hardware I/O configuration to the channel subsystem. For this definition IOCP retrieves information about the following: the channel paths in the processor complex, control units attached to the channel paths, and I/O devices assigned to the control unit.

## Hardware Change: HCD - Dynamic Reconfiguration

- The following is an outline of the steps required to perform both a Hardware/Software and a Software only Dynamic Reconfiguration.
- Build/Modify a work IODF, make the necessary updates to I/O configuration data in the work IODF.
- Build a production IODF, from the work IODF.
- Write Enable the IOCDS.
- Build the IOCDS. The IOCDS is created from a configuration definition in the production IODF.
- First, verify the I/O configuration using the ACTIVATE,TEST option.
- Upon successful test, Activate the configuration dynamically.

## Hardware Change: HCD - Dynamic Reconfiguration

- When you activate an IODF, HCD defines the I/O configuration to the channel subsystem and the operating system **IF BOTH a hardware/software reconfiguration is being performed.**
- **IF ONLY a software reconfiguration is being performed,** then HCD updates the I/O configuration definitions to the operating system only.
- Switch to this new IOCDs, for the next POR.
- Update LOADxx to this reflect the newly activated IODF for the next IPL.
- Create an IOCP data set, copy to diskette, for use with the Stand-alone IOCP at the HMC, if ever required.

**Upon successful activation of the new configuration, any device modifications having been made, are immediately reflected in the system.**

See the OS/390 HCD Manuals for more information on the relevance of the Dynamic Reconfiguration process.

## Changes Requiring an IPL

- Changing the PROCLIB concatenation in SYS1.PARMLIB(MSTJCLxx)
- Updates to SYS1.NUCLEUS
- Addition of a product which adds a new SVC  
(This information should be available from the ISV.)
- Addition of a product subsystem, if it does not support Dynamic Subsystems.  
(This information should be available from the ISV.)
- Addition of a product module to LPA, if does not support Dynamic LPA  
(This information should be available from the ISV.)

**The End**

Questions ?

Thank You !

**Please Complete Session Evaluations**