MVS / z/OS Sysprog Concepts: Course Notes

Oct 8, 2026 · @bhushan

Course overview

This course takes you from "what is an address space" to reading dumps, applying PTFs and tuning a z/OS system, in 13 modules. It is written for new and aspiring systems programmers who know basic JCL and TSO/ISPF.

What you will be able to do by the end

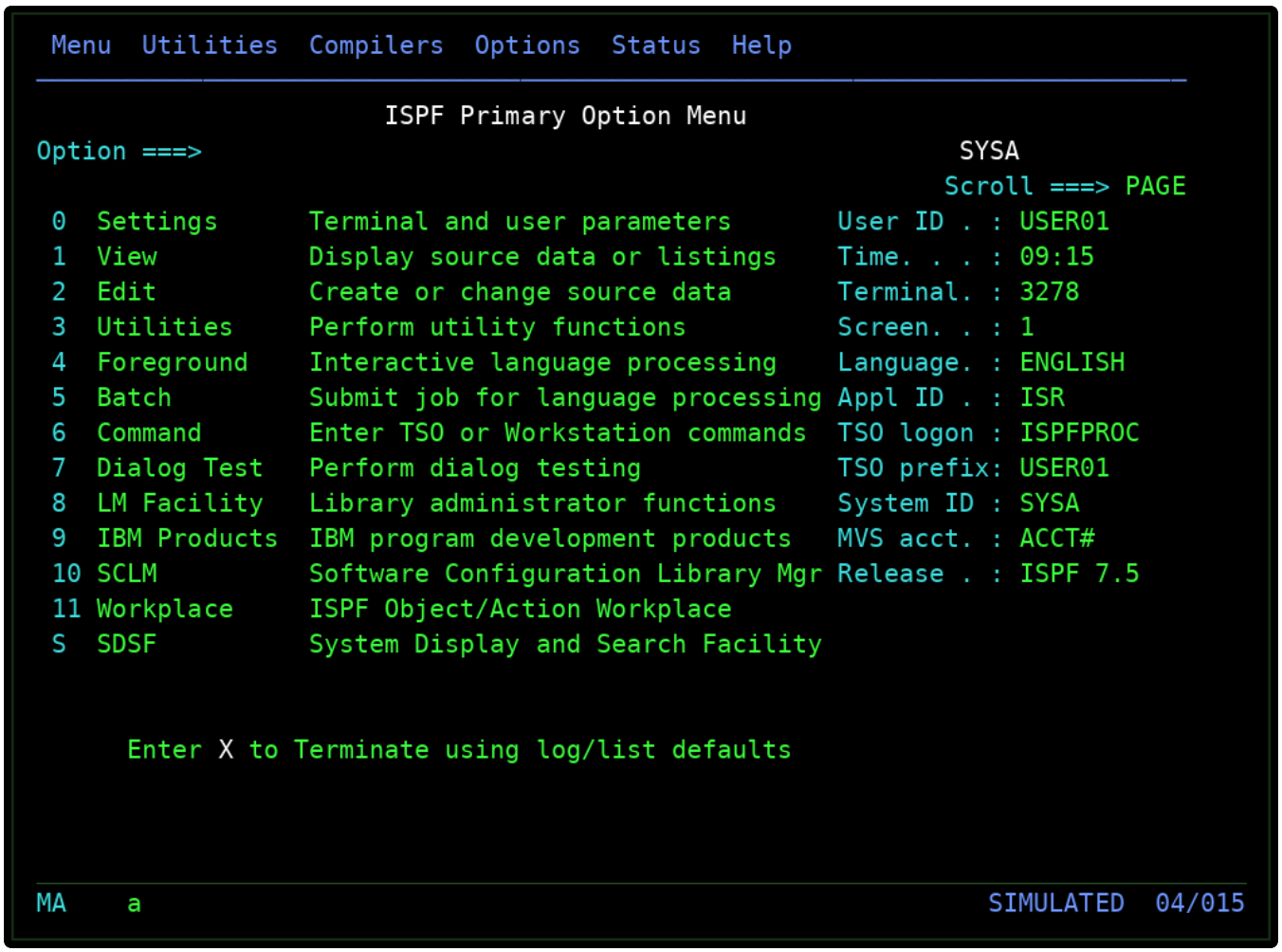
- Explain how z/OS (the descendant of MVS) organizes storage, address spaces and work
- Follow an IPL and know which PARMLIB members drive it
- Issue and read operator commands with confidence
- Manage datasets, catalogs, JES2 and SDSF
- Install maintenance with SMP/E and secure resources with RACF
- Diagnose common ABENDs from a dump and the system log

How to use these notes

1. Work through modules in order; each ends with a short recap.
2. Type every command on a test system (see the lab section at the end). Reading alone will not stick.
3. Use the cheat sheet in the last section as a revision page.

About the screens

The terminal "screenshots" in this course are **simulated 3270-style screens** that I typed to match typical ISPF, SDSF and console layouts. They are not captures from a real LPAR, so system names, volsers, job numbers and timestamps are made up and your own system will differ in detail. Message IDs and command syntax follow standard IBM z/OS behavior; always check the current IBM documentation for your z/OS release.



Simulated screen: ISPF Primary Option Menu, the starting point for most work in this course.

Module	Topic	Core idea
1	Architecture	Address spaces and virtual storage
2	IPL	How the system starts
3	Console commands	Controlling a running system
4	Datasets and catalogs	Where data lives
5	JCL and utilities	The sysprog toolbox
6	PARMLIB, LPA, APF	System configuration
7	JES2 and SDSF	Job flow and monitoring
8	SMP/E	Installing and maintaining software

Module	Topic	Core idea
9	RACF	Security
10	Problem determination	ABENDs and dumps
11	WLM and performance	Workload goals

Modules 12 and 13 go deeper on JES2 (spool, checkpoint, initiators) and SMP/E (relationships, HOLDDATA, maintenance cycle).

Module 1: MVS and z/OS architecture

z/OS gives every job its own private virtual address space, while a shared set of common areas holds the operating system itself.

Key concepts

- **Address space**: the range of virtual storage assigned to a job, started task, TSO user or system component. Each gets a unique ASID.
- **TCB (Task Control Block)**: represents a unit of work running under the control of a program in an address space.
- **SRB (Service Request Block)**: a lighter unit of work, scheduled to run in a specific address space, used mostly by system components.
- **Dispatching**: the dispatcher picks the highest-priority ready TCB or SRB for each CP.
- **Cross-memory services**: let programs in one address space reach data or run code in another without copying through common storage.

Virtual storage layout (31-bit and 64-bit view)

Addressing grew in three steps: 24-bit (16 MB), 31-bit (2 GB), and 64-bit. The boundaries are named the *line* (16 MB) and the *bar* (2 GB).

Area	Location	Purpose
Nucleus (resident)	Bottom of common area	Core system code, always in storage
PLPA / LPA	Common, shared by all	Pageable link pack area: shared reentrant modules

Area	Location	Purpose
CSA	Common, shared by all	Common service area: data shared between address spaces
SQA	Common, fixed	System queue area: system control blocks, fixed in real storage
Private area	Unique per address space	User programs, LSQA, SWA, subpools
Extended areas	Above the 16 MB line	Same layout, 31-bit versions (ECSA, ESQA, EPLPA)

Simulated screen: viewing active address spaces

Operators and sysprogs often start with `D A,L` to list active jobs and started tasks.

```
08.12.44 SYSA     d a,l
08.12.44 SYSA     IEE114I 08.12.44 2026.281 ACTIVITY 211
  JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
  00004    00027    00002     00045    00015    00002/00020         00021
 LLA      LLA      LLA      NSW  S    VLF      VLF      VLF      NSW  S
 JES2     IEFPROC  JES2     NSW  S    RACF     RACF     RACF     NSW  S
 VTAM     VTAM     VTAM     NSW  S    TCPIP    TCPIP    TCPIP    NSW  S
 SDSF     SDSF     SDSF     NSW  S    BOB      BOB      OWT      NSW  S
```

```
 SYSA     SYSA     26281 09:12:44
 D A,L
 IEE114I 09.12.44 2026.281 ACTIVITY 211
  JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
  00004    00027    00002     00045    00015    00002/00020         00021
  LLA      LLA      LLA      NSW  S    VLF      VLF      VLF      NSW  S
  JES2     IEFPROC  JES2     NSW  S    RACF     RACF     RACF     NSW  S
  VTAM     VTAM     VTAM     NSW  S    TCPIP    TCPIP    TCPIP    NSW  S
  SDSF     SDSF     SDSF     NSW  S    BOB      BOB      OWT      NSW  S
  USER01   OMVS     OMVS     OWT  AO   MYJOB1   STEP1    CALC     NSW  SO
 ===>
 MA    a                                                  SIMULATED  04/015
```

Read it like this: the first row summarizes counts (jobs, started tasks, TSO users, system address spaces, initiators). Each row below shows jobname, stepname, procstep and a status flag. `NSW` means non-swappable.

Recap

- One virtual address space per unit of work; common areas are shared by all.
- TCBs run user and application work; SRBs run quick system work.
- The 16 MB line and 2 GB bar are the dividing lines for storage and addressing modes (AMODE 24, 31, 64).

Module 2: IPL and system initialization

An IPL (Initial Program Load) loads the nucleus from the sysres volume and then reads configuration from PARMLIB until the master scheduler is up.

The IPL sequence in six steps

1. The operator picks the **IPL device** (sysres) and the **LOAD parameter** on the HMC or console. The load parm is typically `ddddxxMN` (IODF device, LOADxx suffix, nucleus ID).
2. The IPL text and the **IPL (nucleus) loader** read the nucleus into storage.
3. **LOADxx** (in SYS1.PARMLIB or SYSn.IPLPARM) names the IODF, the master catalog, the IEASYM and IEASYS members, and the NUCLST.
4. **NIP (Nucleus Initialization Program)** builds the system storage map, reads **IEASYSxx** and **IEASYMxx**, and sets up LPA, CSA, SQA, link list and APF list.
5. The **master scheduler** starts and processes **COMMNDxx** and **IEFSSNxx** (subsystem definitions).
6. **JES2** (or JES3) starts, then the rest of the started tasks and applications you automate.

Key members and what they control

Member	What it defines
LOADxx	IODF, master catalog, parmlib concatenation, IEASYM suffix
IEASYSxx	System parameters: CSA/SQA sizes, MAXUSER, PAGE datasets, link list suffix
IEASYMxx	System symbols such as &SYSNAME and &SYSCLONE
IEFSSNxx	Subsystems (JES2, RACF, DB2 interface and others)
COMMNDxx	Commands issued automatically at IPL
PROGxx	APF list, link list, LPA additions, exits
PARMLIB concatenation	Ordered list of datasets searched for each member

Simulated screen: console messages during an IPL

This shows the shape of what an operator sees. Exact messages vary by release and configuration.

```
IEA371I SYS1.IPLPARM ON DEVICE 0A82 SELECTED FOR IPL PARAMETERS
IEA246I LOAD ID 00 SELECTED
IEA246I NUCLST ID 00 SELECTED
IEA007I STATIC SYSTEM SYMBOL VALUES
IEA008I SYSTEM PARAMETERS FROM IEASYS00
IEA008I SYSTEM PARAMETERS FROM IEASYS01
IEE252I MEMBER IEASYM00 FOUND IN SYS1.PARMLIB
IEE252I MEMBER IEASYS00 FOUND IN SYS1.PARMLIB
IEA101A SPECIFY SYSTEM PARAMETERS FOR RELEASE z/OS 03.01.00
R 00,CLPA
IEE600I REPLY TO 00 IS;CLPA
IEE252I MEMBER PROG01 FOUND IN SYS1.PARMLIB
IXC418I SYSTEM SYSA IS NOW ACTIVE IN SYSPLEX PLEX1
$HASP426 SPECIFY OPTIONS - HASP-II 3.1
R 01,NOREQ
$HASP400 ENTER REQUESTS -
```

```
 IEA371I SYS1.IPLPARM ON DEVICE 0A82 SELECTED FOR IPL PARAMETERS
 IEA246I LOAD ID 02 SELECTED
 IEA246I NUCLST ID 00 SELECTED
 IEA007I STATIC SYSTEM SYMBOL VALUES
   &SYSNAME.  = "SYSA"
   &SYSCLONE. = "SA"
 IEE252I MEMBER IEASYM00 FOUND IN SYS1.PARMLIB
 IEE252I MEMBER IEASYS00 FOUND IN SYS1.PARMLIB
 IEE252I MEMBER IEASYS01 FOUND IN SYS1.PARMLIB
 IEA008I SYSTEM PARAMETERS FROM IEASYS00
 IEA008I SYSTEM PARAMETERS FROM IEASYS01
*00 IEA101A SPECIFY SYSTEM PARAMETERS FOR RELEASE z/OS 03.01.00
R 00,CLPA
 IEE600I REPLY TO 00 IS;CLPA
 IEE252I MEMBER PROG01 FOUND IN SYS1.PARMLIB
 IXC418I SYSTEM SYSA IS NOW ACTIVE IN SYSPLEX PLEX1
*01 $HASP426 SPECIFY OPTIONS - HASP-II 3.1
R 01,NOREQ
 IEE600I REPLY TO 01 IS;NOREQ
 $HASP400 ENTER REQUESTS -
 $HASP373 JES2     STARTED - COLD START
 IEF403I JES2 - STARTED - TIME=07.59.44

===>
MA    a                                                SIMULATED  04/015
```

What the operator reply options mean

- `CLPA` creates a fresh pageable link pack area (use after PLPA-level changes).
- `CLPA` is not needed on a routine re-IPL, so omit it to save time.
- `NOREQ` or `WARM`, `COLD`, `FORMAT` at the $HASP426 prompt control how JES2 recovers its spool and job queue.

Recap

- LOAD parm points to LOADxx, which points to everything else.
- NIP reads IEASYSxx, then the master scheduler starts subsystems and JES2.
- A bad PARMLIB member is the most common cause of a failed IPL, so always keep a known-good fallback.

Module 3: Operator console commands

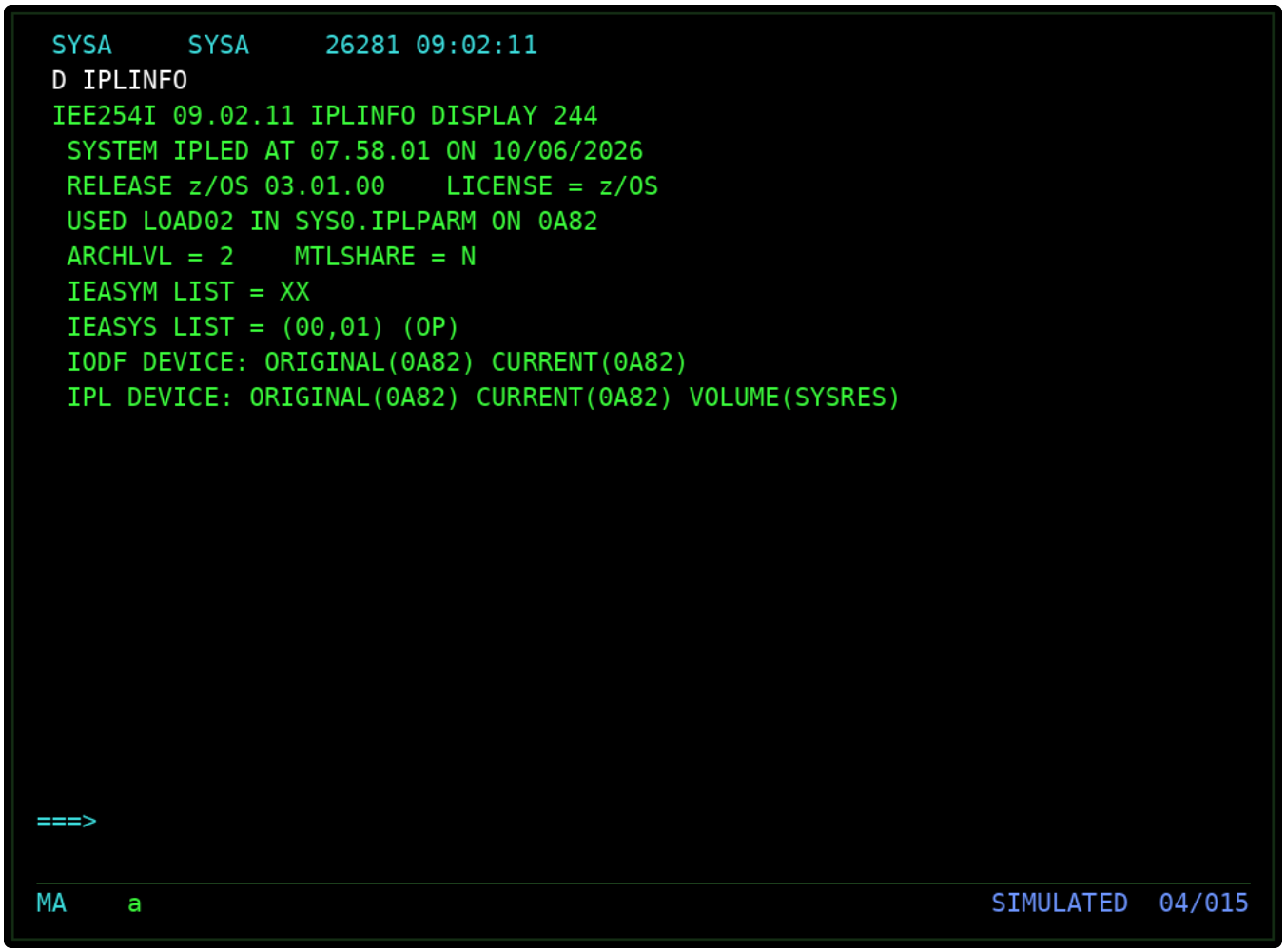
A small set of commands (D, V, SET, SETPROG, F, S, P, C, K) covers most day-to-day control of z/OS; learn their syntax until it is automatic.

The command families

Verb	Meaning	Example
`D`	Display information	`D IPLINFO`, `D PARMLIB`, `D M=CPU`
`V`	Vary a device, path or CPU online/offline	`V 0A82,ONLINE`
`SET`	Activate a PARMLIB member	`SET PROG=01`, `SET SMF=00`
`SETPROG`	Change APF, LNKLST, LPA and exits dynamically	`SETPROG APF,ADD,DSNAME=USER.LOAD,SMS`
`S`	Start a started task	`S JES2`, `S MYPROC,PARM='X'`
`P`	Stop a started task or job	`P MYPROC`
`C`	Cancel a job, started task or TSO user	`C MYJOB`, `C U=USERID`
`F`	Modify (send a command to) a running task	`F LLA,REFRESH`
`K`	Control console display and routing	`K A,100`
`R`	Reply to an outstanding WTOR	`R 05,CLPA`

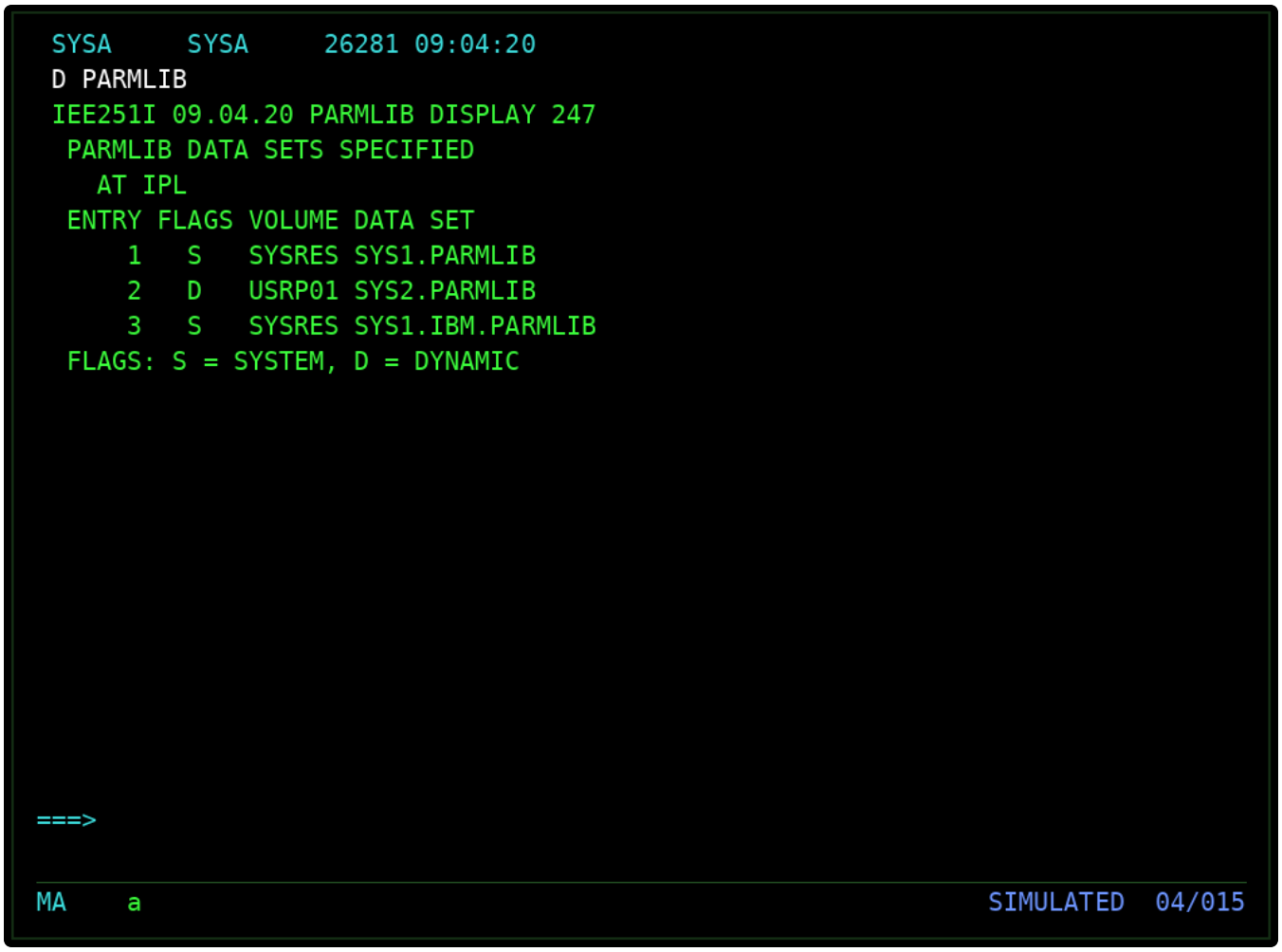
Simulated screen: D IPLINFO

```
D IPLINFO
IEE254I 09.02.11 IPLINFO DISPLAY 244
SYSTEM IPLED AT 07.58.01 ON 10/06/2026
RELEASE z/OS 03.01.00    LICENSE = z/OS
USED LOAD02 IN SYS0.IPLPARM ON 0A82
ARCHLVL = 2    MTLSHARE = N
IEASYM LIST = XX
IEASYS LIST = (00,01) (OP)
IODF DEVICE: ORIGINAL(0A82) CURRENT(0A82)
IPL DEVICE: ORIGINAL(0A82) CURRENT(0A82) VOLUME(SYSRES)
```



Simulated screen: D PARMLIB

```
D PARMLIB
IEE251I 09.04.20 PARMLIB DISPLAY 247
PARMLIB DATA SETS SPECIFIED AT IPL
ENTRY  FLAGS  VOLUME  DATA SET
   1    S      SYSRES  SYS1.PARMLIB
   2    D      USRP01  SYS2.PARMLIB
   3    S      SYSRES  SYS1.IBM.PARMLIB
```



Useful display and control patterns

- `D A,L` lists active address spaces; `D A,jobname` shows one job.
- `D U,,ALLOC,0A82` shows what is allocated to a device.
- `D GRS,C` shows GRS contention (who holds the resource, who waits).
- `D M=CPU` shows processor status; `D M=STOR` shows central storage.
- `D PROG,APF` shows the APF list; `D PROG,LNKLST` shows the link list.
- `F LLA,REFRESH` makes LLA pick up new module versions in LNKLST datasets.
- `D R,L` lists outstanding operator replies (WTORs).

Command safety tips

1. Spell out the target. `C MYJOB` can cancel the wrong copy if two are running, so add `,A=asid` when in doubt.
2. Check with `D A,jobname` before you `P`, and prefer `P` over `C` for production tasks so they can shut down cleanly.
3. Run `D PARMLIB` and `D PROG,*` before and after a configuration change so you know what changed.

Recap

- D shows, V varies, SET and SETPROG change configuration, S/P/C/F manage tasks.
- Learn the five "first look" commands: D IPLINFO, D A,L, D PARMLIB, D GRS,C, D R,L.

Module 4: Datasets, DASD, VTOC and catalogs

Every dataset lives on a volume described by that volume's VTOC and is found by name through a catalog.

Dataset attributes you must know

Attribute	Meaning	Common values
DSORG	Organization	PS (sequential), PO (partitioned), DA, VSAM
RECFM	Record format	F, FB, V, VB, U (add A/M for control chars)
LRECL	Logical record length	80 for JCL/source, 133 for print, up to 32760 for VB
BLKSIZE	Block size	Best set to half-track or 0 for system-determined
DSNTYPE	PDS or PDSE	LIBRARY for PDSE, blank for PDS

PDS vs PDSE

- A **PDS** has a directory at the front and members after it; deleted members leave gaps until you compress.
- A **PDSE** reuses space automatically, has no fixed directory size and supports more members, but needs SMS and cannot be used for some early-IPL datasets.

Volumes and the VTOC

A DASD volume has a label (volser), a **VTOC** (list of datasets and their extents) and optionally a **VTOC index** and **VVDS** (VSAM volume dataset). The VTOC is what you see with `IEHLIST LISTVTOC` or ISPF 3.4.

Catalogs (ICF)

- The **master catalog** is named in LOADxx and holds system datasets and aliases.
- **User catalogs** hold the rest; an **alias** in the master catalog points to a user catalog by high-level qualifier.
- A catalog is a VSAM dataset, so you manage it with IDCAMS (`LISTCAT`, `DEFINE`, `EXPORT`, `IMPORT`).
- With SMS, almost all datasets are cataloged, and the catalog finds the volume for you.

Simulated screen: ISPF 3.4 dataset list

```
 DSLIST - Data Sets Matching SYS1.PARM*                  Row 1 of 3
 Command ===>                                          Scroll ===> PAGE

 Command - Enter "/" to select action                   Message  Volume
 ------------------------------------------------------------------------------
---
          SYS1.PARMLIB                                                  SYSRES
          SYS1.PARMLIB.BACKUP                                           SYSRES
          SYS1.PARMLIB.TEST                                             USRP01
 ****************************** End of Data Set list
****************************
```

```
DSLIST - Data Sets Matching SYS1.PARM*                        Row 1 of 3
Command ===>                                          Scroll ===> PAGE

Command - Enter "/" to select action                  Message          Volume
-------------------------------------------------------------------------------
         SYS1.PARMLIB                                                 SYSRES
         SYS1.PARMLIB.BACKUP                                          SYSRES
         SYS1.PARMLIB.TEST                                            USRP01
****************************** End of Data Set list ****************************
```

```
MA    a                                                  SIMULATED  04/015
```

Simulated screen: IDCAMS LISTCAT output

```
LISTCAT ENTRIES('SYS1.PARMLIB') ALL
NONVSAM ------- SYS1.PARMLIB
     IN-CAT --- CATALOG.Z31.MASTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------2026.012
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---SCSYSTEM    MANAGEMENTCLASS-(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------SYSRES     DEVTYPE------X'3010200F'
     ASSOCIATIONS
       (NULL)
```

```
LISTCAT ENTRIES('SYS1.PARMLIB') ALL

NONVSAM ------- SYS1.PARMLIB
     IN-CAT --- CATALOG.Z31.MASTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------2026.012
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---SCSYSTEM    MANAGEMENTCLASS-(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------SYSRES     DEVTYPE------X'3010200F'
     ASSOCIATIONS
       (NULL)

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0




===>
MA    a                                                SIMULATED  04/015
```

Practical rules for sysprogs

1. Keep the master catalog small. Move user data to user catalogs and use aliases.
2. Back up catalogs (`EXPORT`) before and after structural changes.
3. Use `LISTCAT LEVEL(hlq)` to see everything under an alias, and `LISTCAT ENTRIES(name) ALL` for one dataset.
4. Use `SPACE` in cylinders or `AVGREC` to avoid fragmentation, and let SMS choose volumes where possible.

Recap

- VTOC lists what is on a volume; the catalog says where a name lives.
- Know RECFM, LRECL, BLKSIZE and DSORG by heart; they decide whether a program can read the data.
- Master catalog holds the system; user catalogs hold the rest, linked by aliases.

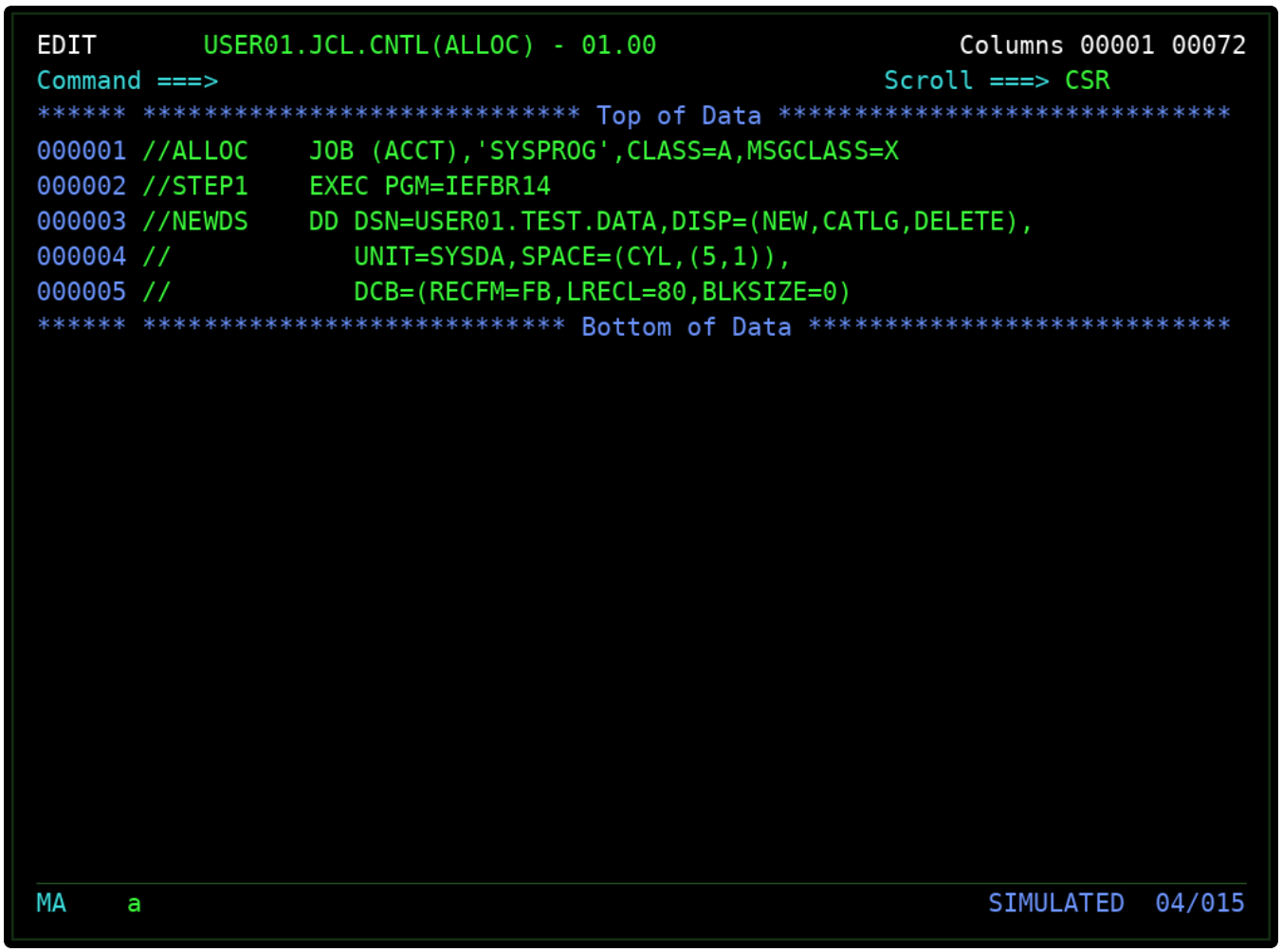
Module 5: JCL and utilities for sysprogs

Six IBM utilities (IEFBR14, IEBGENER, IEBCOPY, IDCAMS, ADRDSSU, IEHLIST) cover most allocation, copy, catalog and backup tasks.

Utility	Used for
IEFBR14	Do nothing; allocate or delete datasets via DD statements
IEBGENER	Copy sequential data or a member to another dataset
IEBCOPY	Copy, compress, unload and reload PDS and PDSE libraries
IDCAMS	VSAM and catalog work: DEFINE, DELETE, LISTCAT, REPRO, EXPORT
ADRDSSU (DFSMSdss)	Dump, restore, copy and move datasets or whole volumes
IEHLIST	List a volume's VTOC or a PDS directory

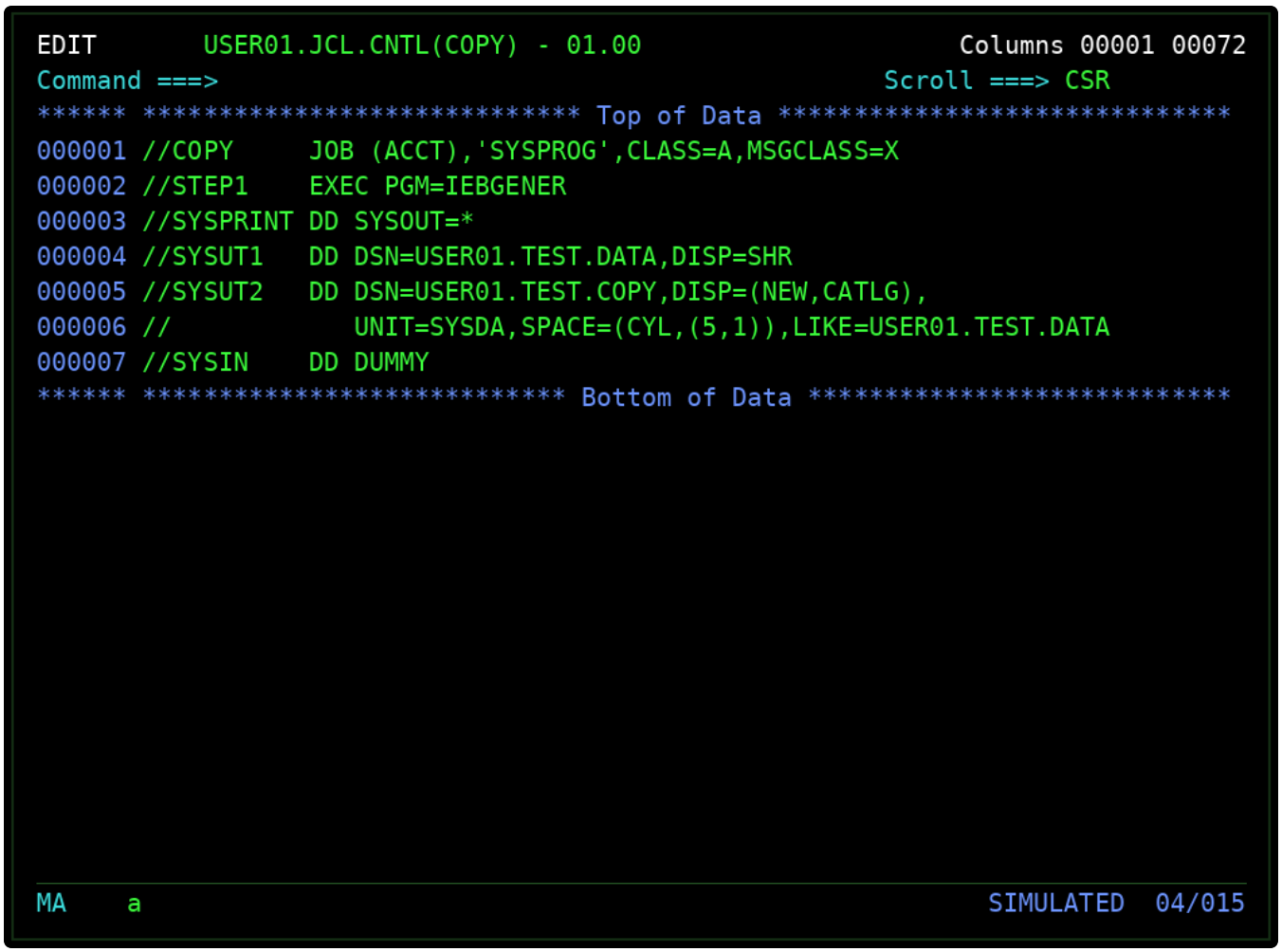
Allocate a dataset with IEFBR14

```
//ALLOC    JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
//STEP1    EXEC PGM=IEFBR14
//NEWDS    DD DSN=USER01.TEST.DATA,DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSDA,SPACE=(CYL,(5,1)),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0)
```



Copy a sequential file with IEBGENER

```
//COPY     JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
//STEP1    EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DSN=USER01.TEST.DATA,DISP=SHR
//SYSUT2   DD DSN=USER01.TEST.COPY,DISP=(NEW,CATLG),
//            UNIT=SYSDA,SPACE=(CYL,(5,1)),LIKE=USER01.TEST.DATA
//SYSIN    DD DUMMY
```



Compress a PDS with IEBCOPY

```
//COMPRESS JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
//STEP1    EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//PDS      DD DSN=USER01.SOURCE.LIB,DISP=OLD
//SYSIN    DD *
  COPY INDD=PDS,OUTDD=PDS
/*
```

```
EDIT       USER01.JCL.CNTL(COMPRESS) - 01.00               Columns 00001 00072
Command ===>                                             Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 //COMPRESS JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
000002 //STEP1    EXEC PGM=IEBCOPY
000003 //SYSPRINT DD SYSOUT=*
000004 //PDS      DD DSN=USER01.SOURCE.LIB,DISP=OLD
000005 //SYSIN    DD *
000006   COPY INDD=PDS,OUTDD=PDS
000007 /*
****** **************************** Bottom of Data ****************************
MA    a                                                  SIMULATED  04/015
```

Define a VSAM KSDS with IDCAMS

```
//DEFVSAM  JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEFINE CLUSTER (NAME(USER01.CUST.KSDS)       -
         INDEXED KEYS(8 0) RECORDSIZE(80 200)  -
         CYLINDERS(5 1) VOLUMES(USRP01))       -
     DATA (NAME(USER01.CUST.KSDS.DATA))        -
     INDEX (NAME(USER01.CUST.KSDS.INDEX))
/*
```

```
EDIT       USER01.JCL.CNTL(DEFVSAM) - 01.00               Columns 00001 00072
Command ===>                                            Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 //DEFVSAM  JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
000002 //STEP1    EXEC PGM=IDCAMS
000003 //SYSPRINT DD SYSOUT=*
000004 //SYSIN    DD *
000005   DEFINE CLUSTER (NAME(USER01.CUST.KSDS)       -
000006          INDEXED KEYS(8 0) RECORDSIZE(80 200)  -
000007          CYLINDERS(5 1) VOLUMES(USRP01))       -
000008      DATA (NAME(USER01.CUST.KSDS.DATA))        -
000009      INDEX (NAME(USER01.CUST.KSDS.INDEX))
000010 /*
****** **************************** Bottom of Data ****************************

MA    a                                                SIMULATED  04/015
```

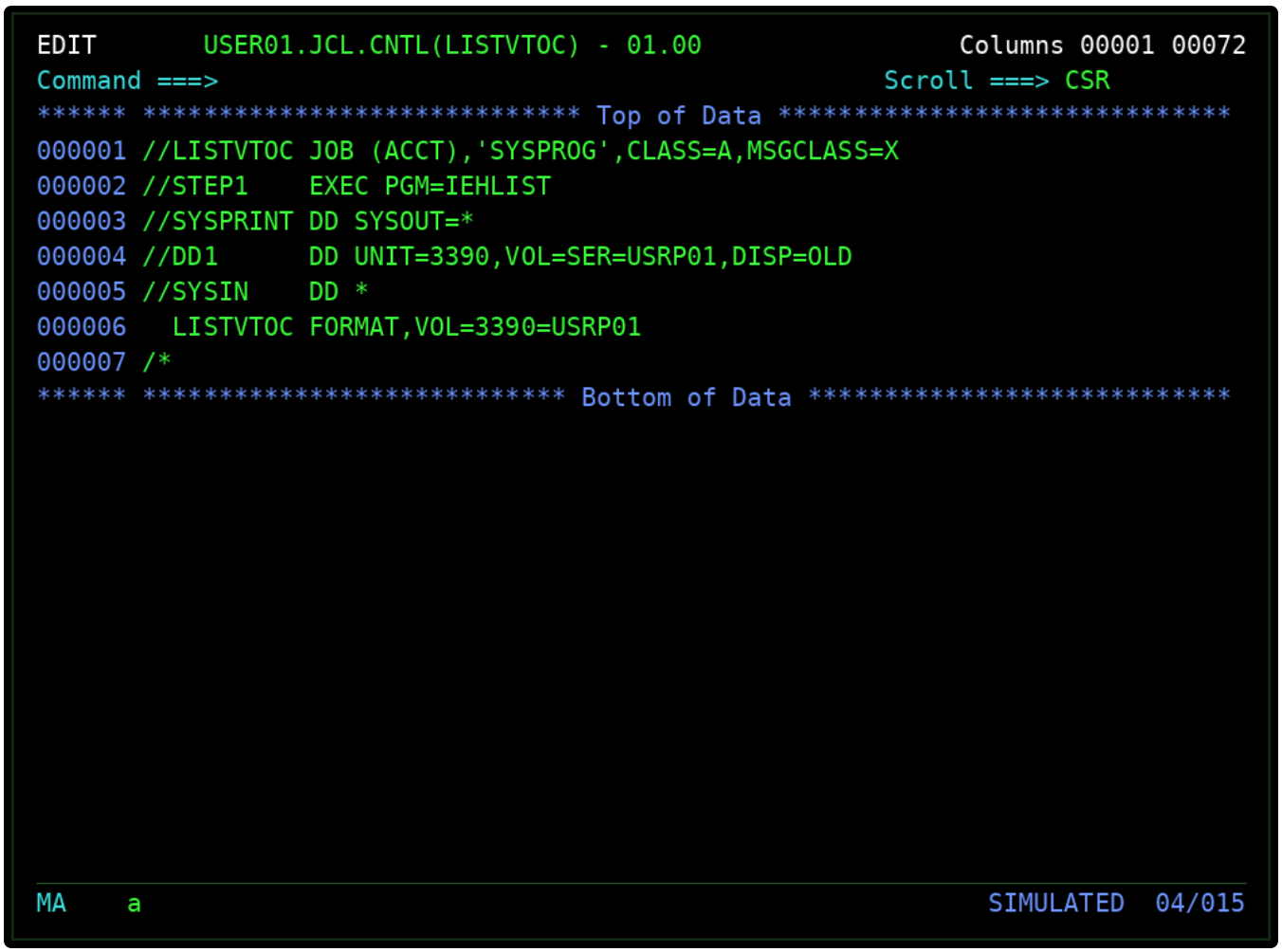
Back up a dataset with ADRDSSU

```
//DUMP     JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=*
//OUT      DD DSN=USER01.BACKUP.DUMP,DISP=(NEW,CATLG),
//            UNIT=SYSDA,SPACE=(CYL,(50,10))
//SYSIN    DD *
  DUMP DATASET(INCLUDE(USER01.TEST.**)) OUTDDNAME(OUT)
/*
```

```
EDIT       USER01.JCL.CNTL(DUMP) - 01.00                 Columns 00001 00072
Command ===>                                           Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 //DUMP     JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
000002 //STEP1    EXEC PGM=ADRDSSU
000003 //SYSPRINT DD SYSOUT=*
000004 //OUT      DD DSN=USER01.BACKUP.DUMP,DISP=(NEW,CATLG),
000005 //            UNIT=SYSDA,SPACE=(CYL,(50,10))
000006 //SYSIN    DD *
000007   DUMP DATASET(INCLUDE(USER01.TEST.**)) OUTDDNAME(OUT)
000008 /*
****** **************************** Bottom of Data ****************************
MA    a                                                SIMULATED  04/015
```

List a volume's VTOC with IEHLIST

```
//LISTVTOC JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
//STEP1    EXEC PGM=IEHLIST
//SYSPRINT DD SYSOUT=*
//DD1      DD UNIT=3390,VOL=SER=USRP01,DISP=OLD
//SYSIN    DD *
  LISTVTOC FORMAT,VOL=3390=USRP01
/*
```



The output lists every dataset on the volume with its creation date, expiration and extent count, then reports free space in empty cylinders and blank DSCBs.

```
SDSF OUTPUT DISPLAY LISTVTOC JOB00201  DSID     2 LINE 0       COLUMNS 02- 81
COMMAND INPUT ===>                                  SCROLL ===> CSR
******************************** TOP OF DATA *********************************
 IEHLIST SYSTEM UTILITY              DATE 2026.281   TIME 09.50.12

 CONTROL STATEMENTS
   LISTVTOC FORMAT,VOL=3390=USRP01

 DUMP OF VTOC ENTRIES FOR DATA SETS ON VOLUME USRP01

 DATA SET NAME                    SER NO  CREATED   EXPIRES  SEC EXT
 SYS1.VTOCIX.USRP01              USRP01  2026.012  0000.000  000 001
 SYS1.VVDS.VUSRP01               USRP01  2026.012  0000.000  000 001
 USER01.CUST.KSDS.DATA           USRP01  2026.281  0000.000  000 001
 USER01.CUST.KSDS.INDEX          USRP01  2026.281  0000.000  000 001
 USER01.TEST.COPY                USRP01  2026.281  0000.000  000 001
 USER01.TEST.DATA                USRP01  2026.281  0000.000  000 001

 THERE ARE 00012345 EMPTY CYLINDERS PLUS 00000 EMPTY TRACKS ON THIS VOLUME
 THERE ARE 000042 BLANK DSCBS IN THE VTOC ON THIS VOLUME
******************************** BOTTOM OF DATA ******************************


MA    a                                                  SIMULATED  04/015
```

Simulated screen: SDSF output of a job

```
JOB ALLOC (JOB00123)  STEPNAME  PROCSTEP  CC
------------------------------------------------
  ALLOC    STEP1                          0000
IEF142I ALLOC STEP1 - STEP WAS EXECUTED - COND CODE 0000
IEF285I   USER01.TEST.DATA   CATALOGED
IEF285I   VOL SER NOS= USRP01
IEF373I STEP/STEP1   /START 2026281.0912
IEF374I STEP/STEP1   /STOP  2026281.0912 CPU 0MIN 00.01SEC
```

```
SDSF OUTPUT DISPLAY ALLOC    JOB00123  DSID     3 LINE 0       COLUMNS 02- 81
COMMAND INPUT ===>                                  SCROLL ===> CSR
******************************** TOP OF DATA *********************************
 ICH70001I USER01   LAST ACCESS AT 09:10:44 ON THURSDAY, OCTOBER 8, 2026
 IEF142I ALLOC STEP1 - STEP WAS EXECUTED - COND CODE 0000
 IEF285I   USER01.TEST.DATA                          CATALOGED
 IEF285I   VOL SER NOS= USRP01.
 IEF373I STEP/STEP1   /START 2026281.0912
 IEF374I STEP/STEP1   /STOP  2026281.0912 CPU    0MIN 00.01SEC SRB    0MIN 00.00
 IEF375I  JOB/ALLOC   /START 2026281.0912
 IEF376I  JOB/ALLOC   /STOP  2026281.0912 CPU    0MIN 00.01SEC SRB    0MIN 00.00
******************************** BOTTOM OF DATA ******************************
MA    a                                                    SIMULATED  04/015
```

Recap

- IEFBR14 allocates, IEBGENER copies, IEBCOPY handles libraries, IDCAMS handles VSAM and catalogs, ADRDSSU moves and backs up, IEHLIST lists a VTOC.
- Always read the return code and the IEF285I lines to confirm what was cataloged or deleted.

Module 6: PARMLIB, LPA, link list and APF

These four mechanisms decide which programs the system can find quickly and which are trusted to run authorized.

The program search order

When a program is requested, MVS looks in this order (simplified): job pack area, task libraries (`STEPLIB` / `JOBLIB`), the **LPA** (resident, then pageable, then dynamic), and finally the **link list** (LNKLST datasets). Putting a module in LPA or the link list makes it available system-wide without a STEPLIB.

Concept	What it is	Where defined
PARMLIB	Concatenated libraries of system parameter members	LOADxx
LPA	Link Pack Area: shared, reentrant modules in common storage	LPALSTxx, PROGxx
Link list	Datasets searched after LPA for any program	PROGxx (LNKLST), LNKLSTxx
APF	Authorized Program Facility: libraries whose programs can run authorized	PROGxx, IEAAPFxx

APF authorization

A program runs authorized only if its module is link-edited with the AC(1) attribute **and** it is loaded from an APF-authorized library. Both conditions are required, and every library in the STEPLIB concatenation must be authorized or the whole concatenation loses authorization.

Simulated PROGxx member

```
APF FORMAT(DYNAMIC)
APF ADD DSNAME(SYS1.LINKLIB)      SMS
APF ADD DSNAME(SYS1.SVCLIB)       SMS
APF ADD DSNAME(USER.TEST.AUTHLIB) VOLUME(USRP01)
LNKLST DEFINE NAME(LNKLST01) COPYFROM(CURRENT)
LNKLST ADD NAME(LNKLST01) DSNAME(USER.UTIL.LOAD)
LNKLST ACTIVATE NAME(LNKLST01)
```

```
EDIT       SYS1.PARMLIB(PROG01) - 01.00                  Columns 00001 00072
Command ===>                                           Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 APF FORMAT(DYNAMIC)
000002 APF ADD DSNAME(SYS1.LINKLIB)      SMS
000003 APF ADD DSNAME(SYS1.SVCLIB)       SMS
000004 APF ADD DSNAME(USER.TEST.AUTHLIB) VOLUME(USRP01)
000005 LNKLST DEFINE NAME(LNKLST01) COPYFROM(CURRENT)
000006 LNKLST ADD NAME(LNKLST01) DSNAME(USER.UTIL.LOAD)
000007 LNKLST ACTIVATE NAME(LNKLST01)
****** **************************** Bottom of Data ****************************

MA    a                                                SIMULATED  04/015
```

Dynamic changes with SETPROG

Since APF FORMAT(DYNAMIC) is on, you can change the list without an IPL.

```
SETPROG APF,ADD,DSNAME=USER.TEST.AUTHLIB,VOLUME=USRP01
CSV410I DATA SET USER.TEST.AUTHLIB ADDED TO APF LIST

D PROG,APF,DSNAME=USER.TEST.AUTHLIB
CSV450I 10.41.12 PROG,APF DISPLAY
  ENTRY  VOLUME  DSNAME
     44  USRP01  USER.TEST.AUTHLIB
```

Image — Simulated console screen: SETPROG APF and D PROG,APF

Refreshing LLA after changing the link list

```
SETPROG LNKLST,DEFINE,NAME=LNKLST02,COPYFROM=CURRENT
SETPROG LNKLST,ADD,NAME=LNKLST02,DSNAME=USER.UTIL.LOAD
SETPROG LNKLST,ACTIVATE,NAME=LNKLST02
F LLA,REFRESH
```

Image — Simulated console screen: SETPROG LNKLST, F LLA,REFRESH and D PROG,LNKLST

Rules of thumb

1. Never authorize a library that ordinary users can write to; an APF library with weak RACF protection is a security hole.
2. Prefer the link list for utilities used everywhere and LPA only for modules that need to be shared and resident.
3. After any PROGxx edit, `SET PROG=xx` activates it, and `D PROG,*` confirms the result.

Recap

- Search order: job pack, STEPLIB, LPA, link list.
- APF needs both AC(1) and an authorized library.
- Use PROGxx for permanent settings and SETPROG for dynamic ones.

Module 7: JES2 and SDSF

JES2 receives, queues, schedules and prints jobs; SDSF is the ISPF tool you use to watch and control all of it.

The life of a job in JES2

1. **Input**: JES2 reads the JCL from the internal reader (submit), a card reader, or a remote source and assigns a job number.
2. **Conversion**: the JCL is checked and turned into internal text; syntax errors stop the job here.
3. **Queued for execution**: the job waits in its class queue until a matching initiator is free.
4. **Execution**: an initiator runs each step; output goes to spool as SYSOUT.
5. **Output**: JES2 queues the output by class and destination for printing or viewing.
6. **Purge**: when output is released and printed or deleted, JES2 frees the spool space.

Key JES2 objects

Object	Role
Spool	Datasets (SPOOL volumes) holding job text and SYSOUT
Checkpoint	Data that lets JES2 restart and recover the queues; one or two datasets
Initiator	Address space that runs jobs of assigned classes
Class	Job class A, B, and so on; routes jobs to initiators
Destination	Where output goes: a printer, a remote node, a user

Common JES2 commands (issued with a leading $)

Command	Purpose
`$DJ1234`	Display a job
`$HJ1234` / `$AJ1234`	Hold or release a job
`$CJ1234`	Cancel a job
`$DI`	Display initiators
`$TI1,CLASS=AB`	Change classes for initiator 1
`$D SPOOL`	Display spool utilization
`$P JES2`	Stop JES2 (use with care)

Simulated screen: SDSF DA (Display Active)

```
SDSF DA SYSA   (ALL)       PAG  0  CPU/L    8/  6  LINE 1-6 (6)
COMMAND INPUT ===>                                  SCROLL ===> CSR
NP   JOBNAME  StepName ProcStep JobID    Owner    C Pos DP Real Paging  SIO
     MYJOB1   STEP1    CALC     JOB01234 USER01   A     NS 1200   0.00 0.00
     TSO1     TSO1     TSO1     TSU01001 USER01   TS    NS 4500   0.00 1.20
     JES2     IEFPROC  JES2     STC00001 STCUSER  STC   NS 6800   0.00 0.00
     VTAM     VTAM     VTAM     STC00002 STCUSER  STC   NS 3100   0.00 0.00
     TCPIP    TCPIP    TCPIP    STC00003 STCUSER  STC   NS 5200   0.00 0.00
     SDSF     SDSF     SDSF     STC00004 STCUSER  STC   NS 2300   0.00 0.00
```

Image — Simulated SDSF DA panel

Simulated screen: SDSF ST (Status) with job actions

```
SDSF STATUS DISPLAY ALL CLASSES                        LINE 1-4 (4)
COMMAND INPUT ===>                                    SCROLL ===> CSR
NP   JOBNAME  JobID    Owner    Prty Queue    C  Pos  SAff  ASys
     MYJOB1   JOB01234 USER01      1 EXECUTION A        SYSA  SYSA
 S   MYJOB2   JOB01235 USER01      1 PRINT     A    12
 P   MYJOB3   JOB01236 USER01      1 EXECUTION B        SYSA  SYSA
     MYJOB4   JOB01237 USER01      1 AWAITING  A     3
```

Image — Simulated SDSF ST panel with line commands

Prefix letters in the NP column act on that row: `S` shows output, `H` holds, `A` releases, `C` cancels, `P` purges.

Other SDSF panels worth knowing

- `LOG` shows the system log (SYSLOG and OPERLOG), and `LOG` with `F` finds a message ID.
- `INIT` shows initiators and lets you start or drain them.
- `PR` shows printers; `O` shows output by class; `H` shows held output.
- `SYS` shows system-wide address space and CPU use; `ENC` shows enclaves.

Recap

- Job flow is input, conversion, queued, execution, output, purge.
- Spool and checkpoint are the two datasets whose loss or exhaustion stops JES2.
- Learn SDSF line commands (S, H, A, C, P) and the DA, ST, LOG, INIT panels.

Module 8: SMP/E

SMP/E is the tool that installs, tracks and removes IBM and vendor software, so every change to a z/OS product goes through it.

Core terms

Term	Meaning
SYSMOD	A unit of change: FUNCTION (new product), PTF (fix), APAR (temporary fix), USERMOD (your own change)
CSI	Consolidated Software Inventory: a VSAM dataset that holds the state of every element and SYSMOD
Zone	A section of the CSI: GLOBAL (control), TARGET (running libraries), DLIB (distribution libraries)
Target libraries	The datasets the system actually runs from
Distribution libraries	A clean copy used to rebuild or back out target libraries
HOLDDATA	Instructions flagging a SYSMOD as needing a manual action (ERROR, SYSTEM, USER)

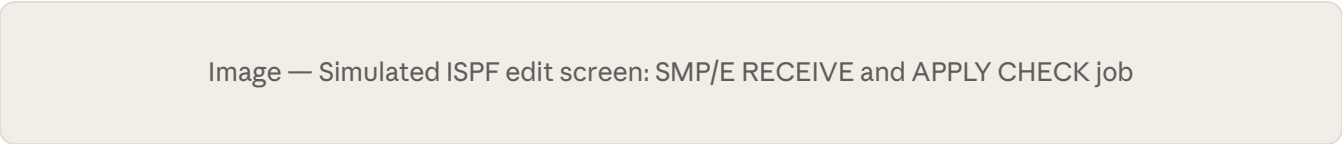
The standard SMP/E flow: RECEIVE, APPLY, ACCEPT

1. **RECEIVE**: bring SYSMODs and HOLDDATA into the GLOBAL zone.
2. **APPLY CHECK**: test an APPLY without changing anything. Always do this first.
3. **APPLY**: install into the target zone and target libraries. You can still restore (undo) at this point.
4. **ACCEPT**: copy into the distribution zone. After this the change is permanent.

Only ACCEPT after the system has run well with the APPLY, usually after a trial period.

Receive and apply a PTF

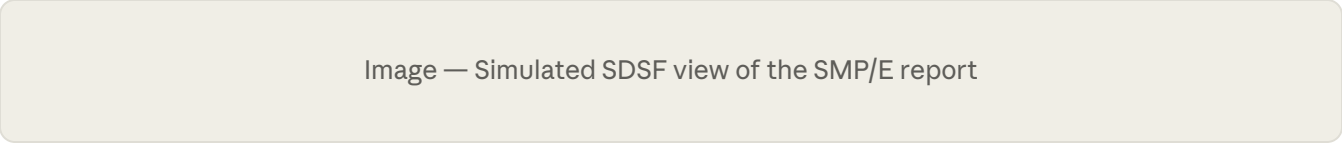
```
//SMPEJOB  JOB (ACCT),'SYSPROG',CLASS=A,MSGCLASS=X
//STEP1    EXEC PGM=GIMSMP,REGION=0M,PARM='DATE=U'
//SMPCSI   DD DSN=SMPE.GLOBAL.CSI,DISP=SHR
//SMPPTFIN DD DSN=SMPE.PTFIN,DISP=SHR
//SMPHOLD  DD DUMMY
//SMPOUT   DD SYSOUT=*
//SMPRPT   DD SYSOUT=*
//SMPLIST  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SMPCNTL  DD *
  SET BOUNDARY(GLOBAL).
  RECEIVE SELECT(UK12345).
  SET BOUNDARY(MVST100).
  APPLY SELECT(UK12345) CHECK.
/*
```



If CHECK ends with return code 0, change `APPLY ... CHECK` to `APPLY SELECT(UK12345).` and run again. After testing, use `ACCEPT SELECT(UK12345) CHECK` and then `ACCEPT SELECT(UK12345)`.

Simulated screen: SMP/E report snippet

```
GIM20502I RECEIVE PROCESSING IS COMPLETE.  THE HIGHEST RETURN CODE WAS 0000.
GIM43401I APPLY PROCESSING FOR UK12345 WAS SUCCESSFUL.
GIM20501I APPLY PROCESSING IS COMPLETE.  THE HIGHEST RETURN CODE WAS 0000.
GIM25601I SYSMOD UK12345 WAS APPLIED TO ELEMENT HBB77C0 IN ZONE MVST100.
```



Handy commands

- `LIST SYSMODS` and `LIST PTFS` show what is installed.
- `REPORT ERRSYSMODS` finds PTFs that fix known errors in your installed software.
- `RESTORE SELECT(ptf)` backs out an applied PTF that has not been accepted.
- `APPLY ... BYPASS(HOLDSYSTEM)` bypasses a system hold; use only after you have done the action the hold describes.

Recap

- SYSMODs are tracked in the CSI across GLOBAL, TARGET and DLIB zones.
- RECEIVE, then APPLY CHECK and APPLY, then ACCEPT.
- Read HOLDDATA before you apply.

Module 9: Security with RACF

RACF (Resource Access Control Facility) decides, for every access request, whether a user may touch a dataset, a command or a resource, and records the decision.

Building blocks

Object	What it is
User ID	An identity with a password or passphrase and attributes (SPECIAL, OPERATIONS, AUDITOR)
Group	A set of users that share access; groups form a tree
Profile	A rule protecting a resource (a dataset profile, a general resource profile)
Class	A category of resources: DATASET, FACILITY, TSOAUTH, OPERCMDS, SURROGAT and so on
Access level	NONE, EXECUTE, READ, UPDATE, CONTROL, ALTER

User attributes

- **SPECIAL**: can create and change profiles for most resources (the security administrator).
- **OPERATIONS**: broad access to datasets and resources unless a profile denies it.
- **AUDITOR**: can set audit options and read audit data, but not change access rules.

Common commands (issued from TSO or a batch IKJEFT01 job)

```
ADDUSER  USER01  NAME('JANE SYSPROG') DFLTGRP(SYSPROG) PASSWORD(TEMP01)
ALTUSER  USER01  SPECIAL
CONNECT  USER01  GROUP(DEVGRP)
ADDSD    'USER01.TEST.**' UACC(NONE)
PERMIT   'USER01.TEST.**' ID(DEVGRP) ACCESS(READ)
RDEFINE  FACILITY BPX.SUPERUSER UACC(NONE)
PERMIT   BPX.SUPERUSER CLASS(FACILITY) ID(USER01) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

The last line matters: classes held in memory with RACLIST must be refreshed before a profile change takes effect.

Simulated screen: LISTUSER

```
LISTUSER USER01
USER=USER01   NAME=JANE SYSPROG   OWNER=SYSADM    CREATED=26.100
 DEFAULT-GROUP=SYSPROG  PASSDATE=26.281 PASS-INTERVAL=  90 PHRASEDATE=N/A
 ATTRIBUTES=SPECIAL
 REVOKE DATE=NONE            RESUME DATE=NONE
 LAST-ACCESS=26.281/09:45:12
 CLASS AUTHORIZATIONS=NONE
 GROUP=SYSPROG  AUTH=USE      CONNECT-OWNER=SYSADM
 GROUP=DEVGRP   AUTH=USE      CONNECT-OWNER=SYSADM
```

Simulated screen: LISTDSD

```
LISTDSD DATASET('USER01.TEST.**') ALL
INFORMATION FOR DATASET USER01.TEST.** (G)
LEVEL OWNER      UNIVERSAL ACCESS   WARNING  ERASE
----- ----------  ----------------   -------  -----
   00 USER01          NONE             NO       NO
AUDITING
 FAILURES(READ)
ID         ACCESS   ALTER COUNT   CONTROL COUNT   UPDATE COUNT   READ COUNT
------     ------   -----------   -------------   ------------   ----------
DEVGRP     READ     000000        000000          000000         000000
```

Image — Simulated RACF LISTUSER and LISTDSD output

Violation messages you will see

- `ICH408I USER(USER02) GROUP(DEVGRP) NAME(...) USER01.TEST.DATA CL(DATASET) ... INSUFFICIENT ACCESS AUTHORITY`: the user has a profile match but too low an access level.
- `ICH408I ... RESOURCE NOT PROTECTED`: no profile exists; behavior then depends on PROTECT-ALL.
- An S913 ABEND often means the same thing seen from the job's side.

Rules of good practice

1. Give least access: UACC(NONE) on sensitive profiles and explicit PERMITs for groups, not individuals.
2. Protect every APF library, PARMLIB and the system catalog with UPDATE restricted to sysprogs.
3. Turn on SETROPTS PROTECTALL(FAIL) so unprotected datasets are denied by default, after testing.
4. Review SMF type 80 records and RACF reports regularly.

Recap

- Users, groups, profiles and classes make up RACF; access levels run NONE to ALTER.
- Use SETROPTS RACLIST ... REFRESH after changing a RACLISTed class.
- ICH408I tells you exactly who tried what and why it failed.

Module 10: Problem determination

Most z/OS problems can be narrowed down by reading three things in order: the job's messages, the ABEND code and reason, and the system log around the failure time.

ABEND codes you will meet constantly

Code	Meaning	Typical cause and first check
S0C1	Operation exception	Branching into data or garbage; bad entry point or corrupt load module
S0C4	Protection exception	Program touched storage it does not own; check the PSW address and failing instruction in the dump
S0C7	Data exception	Invalid packed or zoned decimal data in a numeric field; find the field and its contents

Code	Meaning	Typical cause and first check
S222	Job cancelled by operator or TSO user	Someone issued CANCEL; check the log for who and why
S322	CPU time limit exceeded	TIME parameter too small or program looped; look at the step's TIME on EXEC or JOB
S806	Module not found	Program name wrong or the load library is missing from STEPLIB or the link list
S878	Not enough virtual storage	REGION too small; increase REGION or check for storage leaks
S913	RACF security violation	User lacks access to a dataset or resource; read the ICH408I message
S0CB	Decimal divide exception	Division by zero or overflow in a decimal divide; check the divisor field

The format is `Sxxx` for system ABENDs and `Unnnn` for user ABENDs set by a program; the hex value after `RC=` or `REASON CODE=` often tells you the exact reason.

Types of dumps

Dump type	Used for
SYSUDUMP	Formatted dump of the failing program's private area; easiest for application problems
SYSABEND	Larger formatted dump that also includes system areas
SYSMDUMP	Unformatted dump of the failing address space; read with IPCS
SVC dump (SYS1.DUMPnn)	Captures system and multiple address spaces; requested with the DUMP command or taken by a component on failure
Stand-alone dump	Taken when the whole system is hung or failing; the last resort

Simulated screen: ABEND message in the job log

```
IEA995I SYMPTOM DUMP OUTPUT 541
  SYSTEM COMPLETION CODE=0C7  REASON CODE=00000007
  TIME=09.58.14  SEQ=00231  CPU=0000  ASID=0034
  PSW AT TIME OF ERROR  078D1000   80AF3C2A  ILC 4  INTC 07
  NO ACTIVE MODULE FOUND
  NAME=UNKNOWN
  DATA AT PSW  00AF3C24 - 0DE0F5D0  F000D010  4780D01E
  GPR  0-3  00000000  00AF3D40  00000008  00000010
IEF450I MYJOB STEP1 - ABEND=S0C7 U0000 REASON=00000007
```

Image — Simulated job log showing an S0C7 ABEND

Here INTC 07 confirms a data exception; the PSW address `00AF3C2A` is the instruction after the failure and the offset into your module helps you find the source statement from the compiler listing.

Simulated screen: searching the system log in SDSF

```
 SDSF SYSLOG    SYSA   10/08/2026     LINE 4812
 COMMAND INPUT ===> F 'MYJOB1' PREV
 N 4000000 SYSA     26281 09:58:14.12 JOB01234 IEF450I MYJOB1 STEP1 -
ABEND=S0C7 U0000 REASON=00000007
 N 4000000 SYSA     26281 09:58:14.14 JOB01234 IEF404I MYJOB1 - ENDED -
TIME=09.58.14
```

Image — Simulated SDSF SYSLOG search

IPCS basics (reading an SVC or SYSMDUMP)

1. Allocate the dump to IPCS (option 0, DEFAULTS, then option 2, ANALYSIS).
2. Run `STATUS` to see the system status, then `STATUS FAILDATA` to see the failing component.
3. Use `LIST` to look at a storage address, and `SUMMARY FORMAT` for address space summaries.
4. Use `VERBEXIT LOGDATA` to see the LOGREC record that preceded the dump.

A simple troubleshooting routine

1. Find the first error message in the job log; ignore later cascades.
2. Note the ABEND code and reason code; look them up in the IBM "MVS System Codes" manual.
3. Check SYSLOG and LOGREC (EREP) around the same time for hardware or system issues.
4. Open the dump, find the PSW and the failing module, then compare with the program listing.
5. Reproduce with a small input, fix, retest, and record what you learned.

Recap

- Start from the first error message, then the ABEND code and reason code.
- S0C4 and S0C7 are program faults; S806, S913 and S878 are environment or setup problems.
- Use SYSUDUMP for app problems, SVC dumps and IPCS for system problems.

Module 11: Performance and workload management

Workload Manager (WLM) lets you state business goals for each kind of work, and z/OS then adjusts dispatching priority and resources to meet them.

WLM concepts

Concept	Meaning
Service definition	The full set of WLM rules, kept in a couple of datasets and installed from the WLM ISPF application
Service policy	A named set of goals you can switch between (for example DAY and NIGHT)
Workload	A label that groups service classes for reporting
Service class	A group of work with the same goal and importance
Classification rules	Match jobs and transactions (by subsystem, job name, userid) to service classes
Report class	A way to report on work without giving it a goal

Goal types

- **Response time**: a percent of transactions should finish within a time (for example 90% in 0.5 sec).
- **Velocity**: how much of the time work runs rather than waits; used for started tasks and batch.
- **Discretionary**: gets what is left after other goals are met.

Importance and performance index

Each goal has an **importance** from 1 (highest) to 5. WLM computes a **performance index (PI)** for every service class: a PI of 1.0 means the goal is met exactly, below 1.0 means it is beating the goal, above 1.0 means it is missing the goal. WLM helps the work with the worst PI and highest importance first.

The SRM and resource basics

The System Resources Manager still handles real storage, paging and swapping decisions; WLM supplies the goals and SRM and the dispatcher carry them out.

Simulated screen: RMF Monitor III style summary

```
RMF V3R1  Sysplex Summary - PLEX1                  Line 1 of 6
Command ===>                                      Scroll ===> HALF

Service Class        Period Imp  Goal       Actual   PI   Users
BATCHHI                  1   3  Vel 60%     64%    0.94      4
CICSHI                   1   1  RT 90%/0.5  91%    0.99     38
STCHIGH                  1   1  Vel 70%     68%    1.03      9
TSOPRD                   1   2  RT 85%/1.0  88%    0.97     12
BATCHLO                  1   5  Disc         --     --      7
SYSSTC                  --   --  Sys          --     --     45
```

Image — Simulated RMF Sysplex Summary

In this made-up example STCHIGH has a PI of 1.03, so it is slightly missing its goal; WLM will give it more resources, taking them from work that is beating its goal.

Typical performance indicators to watch

- CPU busy (MVS and LPAR) and the share used by the highest-importance work
- Paging rate and available frames (central storage shortages)

- I/O response time and device queueing (IOSQ, pending, connect and disconnect time)
- Enqueue delays (`D GRS,C`) and spool or catalog contention
- Delay reasons in RMF for each service class: CPU, storage, I/O, enqueue, other

Tuning approach

1. Find the service class with the highest PI and importance.
2. Check its delay reasons to see what it is waiting for.
3. Fix the resource or change the goal, not the symptom alone.
4. Re-measure over a full business cycle before you decide.

Recap

- WLM manages by goals: service classes, importance, goal type and PI.
- Use RMF to see which service classes miss their goals and why.
- Change one thing at a time and measure again.

Module 12: JES2 in depth

JES2 stays healthy only if three things stay healthy: the spool, the checkpoint and the initiators. Most JES2 incidents come down to one of them.

How JES2 is built and started

- JES2 runs as a started task (the JES2 PROC) and talks to other address spaces through the subsystem interface.
- Its initialization statements come from the JES2 parameter member named in the PROC (often called JES2PARM or similar). The statements you will meet most are `SPOOLDEF`, `CKPTDEF`, `MASDEF`, `JOBCLASS(x)`, `OUTCLASS(x)`, `INITnnn` and `PRTnnn`.
- **Cold start** rebuilds everything and loses queued jobs and output. **Warm start** keeps the job queue and spool. Always use warm unless IBM or your site procedure says otherwise.

The checkpoint

The checkpoint holds the job queue and output queue control information, so JES2 can restart and, in a multi-access spool (MAS) configuration, so all members see the same queues.

Item	What to know
CKPT1 / CKPT2	Primary and secondary checkpoint, on DASD or in a coupling facility structure
Mode	DUPLEX keeps both copies current; DUAL is the older dual-dataset mode
MASDEF	Controls how members share and hold the checkpoint (hold time, dormancy)
Reconfiguration	`$T CKPTDEF,RECONFIG=YES` starts the dialog to move or replace a checkpoint dataset

If JES2 loses a checkpoint it issues operator prompts, so know your site's recovery steps before you need them.

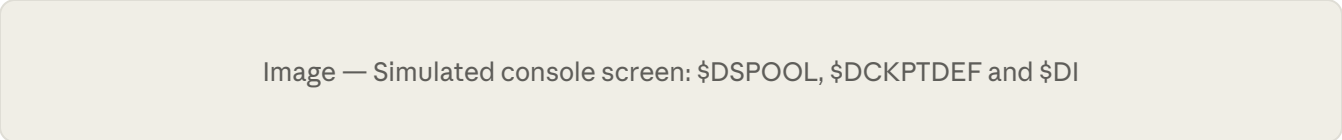
Spool management

- The spool is a set of datasets on spool volumes, divided into track groups. Each job's JCL, messages and SYSOUT live there.
- `$DSPOOL` shows each volume's percent full. Watch for volumes above about 70 to 80 percent, because JES2 starts raising resource-shortage messages as space runs low.
- Add capacity with a new spool volume and `$S SPL(volser)`. Drain a volume you want to remove with `$P SPL(volser)`.
- When the spool fills, jobs and TSO users stall. Purge old held output, then address the cause (a looping job writing endless SYSOUT is typical).

Initiators, classes and affinity

- A **job class** (`JOBCLASS`) sets limits and defaults for jobs; a class can be run by JES2-managed initiators or by WLM-managed initiators.
- Each JES2-managed **initiator** is told which classes it serves. A job in class A waits forever if no active initiator serves class A.
- `SYSAFF` on a job (or system affinity) restricts which MAS member can run it.

Simulated screen: spool, checkpoint and initiators



Everyday job control commands

Command	Effect
`$DJ1234`	Display job 1234 (job number without the JOB prefix)
`$HJ1234` / `$AJ1234`	Hold or release the job
`$TJ1234,C=B`	Change the job's class to B
`$EJ1234`	Restart the job (only if its restart conditions are met)
`$CJ1234`	Cancel the job
`$PJ1234`	Purge the job and its output
`$SI3` / `$PI3`	Start or drain initiator 3
`$TI3,CLASS=AB`	Change the classes initiator 3 serves

Image — Simulated console screen: job control commands

Troubleshooting checklist

1. **Job will not start**: run `$DJ` and check the queue, class, hold status and system affinity, then `$DI` to confirm an active initiator serves that class.
2. **Job ran but no output**: check `OUTCLASS`, the SYSOUT class on the job, and whether output is held.
3. **Everything slows or stalls**: check `$DSPOOL` for a full volume and the log for resource-shortage messages.
4. **Checkpoint messages at IPL or during run**: stop and follow your site's recovery procedure; do not guess with `$P JES2` or a cold start.
5. **JES2 shutdown**: stop applications, drain initiators, then `$PJES2`. Use `$PJES2,ABEND` only in an emergency.

Recap

- Spool, checkpoint and initiators are JES2's three failure points.
- Learn `$DSPOOL`, `$DCKPTDEF`, `$DI` and `$DJ` as your first four diagnostics.

- Never cold start JES2 without a plan; warm start keeps your queues.

Module 13: SMP/E in depth

The skill in SMP/E is not the three commands, it is reading prerequisites, HOLDDATA and reports so you know what a maintenance run will do before you let it.

Elements, DDDEFs and zones

- An **element** is a piece of a product that SMP/E tracks: a module (MOD), macro (MAC), source (SRC) or a load module (LMOD) built from modules.
- A **DDDEF** entry in a zone tells SMP/E which dataset a DD name stands for (for example, which dataset is SYS1.LINKLIB for that zone). A missing or wrong DDDEF is a very common cause of APPLY failures.
- **OPTIONS** entries set defaults for processing, and a **zone set** lets you work with a matching target and distribution zone pair together.

SYSMOD relationships

Relationship	Meaning
PRE	Another SYSMOD must be installed first (hard prerequisite)
REQ	Another SYSMOD must be installed at the same time or before
IFREQ	Required only if a specific product or SYSMOD is also present
SUP	This SYSMOD replaces (supersedes) an older one
SUPING	This SYSMOD is superseded by another

You rarely read these by hand; `GROUPEXTEND` on APPLY or ACCEPT tells SMP/E to pull in required SYSMODs that are already received.

HOLDDATA in practice

- **ERROR holds** flag a PTF as being in error (often shown as PE). The hold is cleared when the fixing PTF is applied, so look for the resolving PTF in the same RECEIVE.

- **SYSTEM holds** carry a reason ID such as ACTION, DOC, IPL, EXIT or DDDEF. They require you to do something (read a note, change a setting, schedule an IPL) before or after the install.
- **USER holds** are your own site's holds on a SYSMOD.
- Bypass a hold only after you have done what it says. Example form: `BYPASS(HOLDSYSTEM(ACTION,DOC))`. Check the exact syntax in the SMP/E Commands manual for your release.

Commands you will use beyond RECEIVE, APPLY and ACCEPT

Command	Use
`LIST`	Show SYSMODs, elements, zones or DDDEFs in a zone
`REPORT ERRSYSMODS`	List error holds that affect SYSMODs you have installed
`REPORT MISSINGFIX`	Find fix categories (FIXCAT) that are not installed
`RESTORE`	Back out an applied SYSMOD that has not been accepted
`REJECT`	Remove a received SYSMOD you have not applied
`UCLIN`	Update CSI entries directly (such as a DDDEF); use with great care
`LINK`	Relink modules into a load module

A maintenance cycle you can follow

1. **RECEIVE** the PTFs and the latest HOLDDATA (for example from an ORDER or a service package).
2. Run `REPORT ERRSYSMODS` and `REPORT MISSINGFIX` to see what is in error or missing.
3. **APPLY CHECK** with `GROUPEXTEND`; read the report for missing prerequisites and holds.
4. Do the actions the SYSTEM holds describe, then **APPLY** for real.
5. Test, and IPL if the holds require it.
6. After a trial period, **ACCEPT CHECK**, then **ACCEPT**.
7. Keep the job outputs; they are your audit trail.

Simulated screens: a maintenance job and its listing

Image — Simulated ISPF edit screen: maintenance job with APPLY CHECK and REPORT ERRSYSM…

Image — Simulated SDSF view of an SMP/E LIST output with a PTF in error

The second screen is a simplified view. Real SMP/E reports are formatted differently and use many more columns and message IDs.

Reading return codes

- **0000**: success.
- **0004**: warnings; read them, since many are harmless but some mean a SYSMOD or part was skipped.
- **0008**: at least one SYSMOD was not processed; look for a missing prerequisite, a hold or an error in a DDDEF.
- **0012 and above**: severe errors such as a damaged CSI or a dataset that cannot be allocated.

Recap

- Know the relationships (PRE, REQ, SUP) and use GROUPEXTEND to pull in what is already received.
- Read HOLDDATA before APPLY; ERROR holds mean a known problem, SYSTEM holds mean work for you.
- RESTORE before ACCEPT is your safety net; ACCEPT is permanent.

Quick reference, self-test and labs

Command cheat sheet

Task	Command
IPL information	`D IPLINFO`
Active jobs and started tasks	`D A,L`
PARMLIB concatenation	`D PARMLIB`
APF list / link list	`D PROG,APF` / `D PROG,LNKLST`

Task	Command
Device status	`D U,,,0A82,1`
GRS contention	`D GRS,C`
Outstanding replies	`D R,L`
Activate PROGxx	`SET PROG=01`
Add APF dataset dynamically	`SETPROG APF,ADD,DSNAME=dsn,SMS`
Refresh LLA	`F LLA,REFRESH`
Start / stop / cancel task	`S name` / `P name` / `C name`
JES2 job display	`$DJ1234`
Spool usage	`$D SPOOL`

Key datasets

Dataset	Purpose
SYS1.PARMLIB	System parameter members
SYS1.PROCLIB	Standard cataloged procedures
SYS1.LINKLIB	Main link list library
SYS1.LPALIB	Modules for the link pack area
SYS1.MACLIB	System macros
SYS1.LOGREC	Hardware and software error records
SYS1.DUMPnn	SVC dump datasets
SYS1.SVCLIB	SVC routines

Self-test (15 questions with answers)

1. What are the 16 MB line and the 2 GB bar? *The 24-bit and 31-bit addressing limits. Storage below the line is 24-bit addressable; between the line and the bar is 31-bit.*
2. What is an ASID? *The address space identifier: a number that uniquely identifies an address space.*

3. Which PARMLIB member is read first at IPL to locate other members? *LOADxx.*
4. What does the CLPA reply do? *Creates a new pageable link pack area from LPALST libraries at IPL.*
5. What is the difference between `P` and `C`? *`P` asks a task to stop cleanly; `C` forces cancellation.*
6. Which utility copies and compresses a PDS? *IEBCOPY.*
7. What is a VTOC? *The volume table of contents that lists datasets and their extents on a DASD volume.*
8. How do you find which catalog holds a dataset? *`LISTCAT ENTRIES('dsn') ALL`, or ISPF 3.4 with the volume shown and the catalog info.*
9. What two conditions make a program run APF-authorized? *Link-edited with AC(1) and loaded from an APF-authorized library.*
10. What are the three SMP/E zone types? *GLOBAL, TARGET and DLIB (distribution).*
11. Why run APPLY with CHECK first? *It verifies the install would succeed without changing the system.*
12. What does ICH408I tell you? *A RACF authorization failure: who, which resource, and the access they lacked.*
13. What does S0C7 usually mean? *A data exception: invalid numeric data in a decimal or numeric field.*
14. What does a PI above 1.0 mean in WLM? *The service class is missing its goal.*
15. After changing a RACLISTed class profile, what must you do? *Issue `SETROPTS RACLIST(class) REFRESH`.*

Hands-on labs

- [ ] **Lab 1 (30 min)**: Log on to a z/OS test system (for example the IBM Z Xplore learning environment or a hobbyist emulator). Run `D IPLINFO`, `D A,L`, `D PARMLIB` and `D PROG,APF` and write down what each tells you.
- [ ] **Lab 2 (45 min)**: Write and submit the IEFBR14, IEBGENER and IEBCOPY jobs from Module 5. In SDSF, read the output and confirm the IEF285I messages and return codes.
- [ ] **Lab 3 (60 min)**: Create a test library, add it to your STEPLIB, run a program from it, and then trigger an S806 on purpose by misspelling the program name. Find the message in SYSLOG and note the fix.

Where to go next

- IBM documentation: *z/OS MVS Initialization and Tuning Reference*, *z/OS MVS System Commands*, *z/OS MVS System Codes*, *SMP/E Commands* and *z/OS Security Server RACF Command Language Reference*.
- Practice: IBM Z Xplore challenges and the free IBM Z trial offerings (check the current availability and terms on IBM's site).
- Habit: keep a personal notes file of every command you type and what it did, because systems programming is learned by repetition.