ISPF Behind the Scenes

SHARE 115
Session 7471

Peter Van Dyke
IBM Australia
SHARE 115, Summer 2010
pvandyke@au1.ibm.com

Agenda

Understanding ISPF Dialogs

- ISPF Initialization
- ISPF task structure
- SELECT service
- LIBDEF
- Debugging Tools
ISPF Initialization

- Invoked from TSO as command processor
  - Expects a CPPL as input
  - Makes call to TSO routines

- Command format
  - ISPF or PDF
    - PANEL(ISR@PRIM) NEWAPPL(ISR)
  - ISPSTART
    - PANEL(ISP@MSTR) NEWAPPL(ISP)
  - ISPF CMD/PGM/PANEL
    - Gets APPLID of ISP if none specified

ISPF Initialization...

- PROFILES
  - ISSPROF
    - Read from ISPROF DD
      - If not found then read from ISPTLIB and write to ISPROF
  - xxxxPROF
    - Read from ISPROF DD
      - If not found then read from ISPTLIB
      - If still not found then read ISPROF member from ISPTLIB
    - Write back to ISPROF DD
  - If ZPROFAPP set then open read only extension
  - The enqueues done creating default profiles are on the first data set in the ISPTLIB concatenation
  - For batch jobs this can cause enqueue problems

```c
//ISPTLIB  DD DISP=(NEW,DELETE),
//     REC FM=FB,LRECL=80,SPACE=(TRK,(1,0,1))
// DD DSN=ISP.SISPTLIB,DISP=SHR
// ...
```
ISPF Initialization…

- Additional Tables
  - ISRPLIST (Personal reference list)
    - Read from ISPPROF DD
      - If not found created with TBCREATE
  - ISRLLIST (Personal library list)
    - Read from ISPPROF DD
      - If not found created with TBCREATE
- Command Tables
  - Read from ISPTLIB
    - ISPCMDS (ISPF command table)
      - Severe error if not found
    - xxxxCMDS (application command table)
      - User – if specified
      - Site – if specified

Screen initialization

- Each screen is started using parameters from product initialization
- START command can specify it's own CMD/PGM/PANEL
  - eg: ISPF invoked using command:
    - ISPF CMD(%mycmd myparms) or ISPF PANEL(mypanel)
  - START or SPLIT command will start the new screen and invoke command
    - CMD (%mycmd myparms) or PANEL(mypanel)

- REXX environment initialized on each screen start
  - Note: Must terminate screen to reload a modified REXX panel
    - exit
- ISPF error panel will restart a screen
KEYLISTS

- PF Key definitions associated with panels

- Coded with )PANEL statement

  \* )PANEL [KEYLIST(keylist name,[applid,[SHARED]])]

  - ISPKYLST is used if keylist not specified

- Normally in xxxxKEYS table in ISPTLIB

  * Use DTL to create

  \*!DOCTYPE DM SYSTEM>
  \*<KEYL NAME=MYKEYLST APPLID=ABC>
  \*  \*<KEYI KEY=F1 CMD=HELP FKA=YES>Help
  \*  \*<KEYI KEY=F2 CMD=SPLIT FKA=LONG>Split
  \*  \*<KEYI KEY=F3 CMD=EXIT FKA=YES>Exit
  \*  \*  \*<KEYI KEY=F24 CMD=CANCEL FKA=YES>Cancel
  \*</KEYL>

- Private copies

  * Created by Keylist Utility
  * Stored in xxxxPROF
  * Stored in ISPSPROF for applid ISP

KEYLISTS...

- KEYLIST Search

  * Keylists ignored if profile variable ZKLUSE=N
    (unless SHARED specified)

  - SHARED
    1. xxxxKEYS
    2. ISPKEYS

  - SHARED not specified
    1. xxxxPROF
    2. xxxxKEYS
    3. ISPSPROF
    4. ISPKEYS

  * ISPF uses the current APPLID if no applid is specified
**ISPF Services**

- All ISPF services are run from `ISPTASK TCB`
- Interface:
  - `ISPLINK`
    - Example: `CALL ISPLINK('SETMSG ', 'ISPZ001 ', ' ', 'ZCMD')`
  - `ISPEXEC`
    - Example: `ISPEXEC SETMSG MSG(ISPZ001) MSGLOC(ZCMD)`
- `ISPLINK` parameters
  - Positional
  - To omit a parameter and use default - code a blank
    - Note: An address starting with x'40' will be treated as an omitted parameter.
  - Last address in parameter list must have high order bit on
    - Use the `VL` keyword in Assembler call statements
  - Standard linkage conventions are observed
  - Keywords and names should be padded to the max length of 8
  - Numeric values are full word binary
    - Don't rely on coding constant. Compiler may not generate a full word value.

**ISPF Task Structure**

```
TSO
  TMP
  ATTACH

ISPSTART
  LINK

ISPMAIN
  ATTACH SZERO=NO

ISPTASK
```

---

Interactive Program Development Facility (ISPF)
ISPF Task Structure…

- DISPLAY of panel
  - Actual display is done by ISPMAIN task with SVC 93 (TPUT/TGET)
  - ISPMAIN will normally be in wait state waiting for user to press enter key
  - ISPTASK is in wait
  - ISPMAIN POST
  - ISPTASK WAIT
  - Display request
ISPF Task Structure...

**SELECT CMD**

- **ISPTASK**
  - Attach szero=0
- **CMD1**
  - Call
- **ISPLINK**
  - Post
  - Wait

**SELECT CMD invoking SELECT PGM**

- **PGM1**
- **ISPTASK**
  - Attach szero=0
- **CMD1**
  - Call
- **ISPLINK**
  - Select pgm
  - Post
  - Wait
ISPF Task Structure...

SELECT CMD with LIBDEF of ISPLLIB

ISPTASK

Wait

Attach szero=no
Tasklib=libdef dcb

CMD1

ISPLINK

Call

Post

ISPF Task Structure...

SELECT PGM with LIBDEF of ISPLLIB

ISPTASK

SVC 6
dcb=ISPLLIB

PGM1
ISPF SELECT Service

- **SELECT CMD**
  - For CLIST - Parsed and run by ISPF. ISPF is aware of TSO commands and will do the ATTACH
  - For REXX - ISPF attaches EXEC and REXX runs the exec. TSO commands are attached by REXX
    - SELECT PGM/CMD needs to be used to create new function pool unlike CLIST processing
    - SELECT is also needed for ISPTCM lookup and ISPF exits to be invoked
    - ISPF will pull from the data stack on end of the REXX exec unless the BARRIER keyword is used
  - Commands - Attached as command processors under ISPTASK
    - IKJTBLS called to do authorization check
    - IKJEFTSR is used to invoke authorized commands
  - NEST keyword - Allows nesting and output trapping
    - ISPF uses TSO macro: STACK BARRIER=*  
    - Default: STACK BARRIER=NEST

- **SELECT PGM**
  - LINK (SVC 06) macro used to invoke program
  - Authorization check done with call to IKJTBLS
  - LIBDEF only affects selected pgm
  - PARM - half word length followed by data

- **NEWAPPL**
  - Opens the following ISPF tables
    - xxxxPROF
    - xxxxCMDS
  - Edit will open xxxxEDIT
  - LIBDEF of ISPTLIB must be done prior to SELECT to affect xxxxCMDS
ISPF SELECT Authorized Commands

- **TSO**
- **TMP**
- **ISPSTART**
- **ISPMAIN**
- **ISPTASK**
- **IKJEFTSR**
- **AUTHCMD**

**ISPF LIBDEF - ISPLLIB**

- **ISPLLIB**
  - Used to pick up ISPF modules on product initialization
  - On invocation of ISPF it is used as TASKLIB to start an ISPF screen
- **LIBDEF of ISPLLIB**
  - **SELECT PGM**
    - DCB parm on LINK macro used to point to user load library
    - DCB parm only affect the module that LINK invokes
    - If EXCLDATA/EXCLLIBR used:
      - LINK with DCB=LIBDEFed dcb
    - Otherwise BLDL is done on libdef'd DCB
      - BLDL finds module:
        - LINK with DCB=LIBDEFed dcb
      - BLDL doesn't find module:
        - LINK with DCB=0
  - **SELECT CMD**
    - ATTACH of command is done with TASKLIB=LIBDEFed dcb
LIBDEF / ISPLLIB Example

000001 PROC 0
000002 ISPEXEC LIBDEF ISPLLIB DATASET ID(TEST.LOAD)
000003 ISPEXEC SELECT CMD(CMD1)
000004 EXIT CODE(0)
000005
000006 ----------------------------------------------------------------------
000007
000008 CMD1 CSECT
000009 SAVE (14, 12)
000010 ...
000011 CALL 1 SPLNIK (SELECT, SELLEN, SELBUFF), VL
000012 ...
000013 ATTACH EP=CMD2, ECB=

000014 ...
000015 SELECT DC 'SELECT'
000016 SELLEN DC F'17'
000017 SELBUFF DC C'PGM(PGM1) PASSLIB'
000018 ...
000019
000020 ----------------------------------------------------------------------
000021
000022 CMD2 CSECT
000023 ...
000024 ...
000025 CALL PGM2
000026 ...

ISPF LIBDEF - ISPLLIB …

SELECT CMD invoking SELECT PGM with LIBDEF of ISPLLIB
ISPF LIBDEF – ISPLLIB…

- MVS Search order
  - Link DCB=0
    1. JPA
    2. TASKLIB, STEPLIB, JOBLIB
    3. LPA
    4. Link List
  - Link DCB=libdef'd dcb
    1. JPA
    2. Specified dcb
    3. LPA
    4. Link List

ISPF SELECT - Variables

SELECT PGM calling REXX exec without using ISPF SELECT
ISPF SELECT – Variables…

SELECT PGM calling REXX exec using ISPF SELECT

- PGM1
  - variable flow
  - select cmd

- REXX EXEC
  - variable flow

- function pool
- shared pool
- profile pool

ISPF SELECT - Variables - VDEFINE

- SELECT
  - Creates function pool

- VDEFINE
  - creates function variable
  - Defines user storage to ISPF
  - Variable definition stays around until SELECT level ends or a VDELETE is done
  - 
  - If program storage goes away (freemain), at same SELECT level, no VDELETE, and reference variable ... Boom
**ISPF SELECT - Variables – VDEFINE…**

- **SELECT PGM1**
  - Creates function pool
  - **PGM1** calls **PGM2**
    - **PGM2** issues VDEFINE for variable
    - Variable control block created with pointer to **PGM2** storage
    - Return to **PGM1** - storage owned by **PGM2** may be freemained at this point
  - **DISPLAY** panel
    - Reference variable defined by **PGM2**
    - Results unpredictable

**ISPF Sub-Task Support**

*SELECT CMD invoking SELECT PGM that attaches user command program*

- **PGM1**
- **ISPTASK**
  - Attach szero=no
- **CMD2**
- **CMD1**
- **ISPLINK**
  - Select pgm
- **Post**
- **Wait**

- CMD2 should not issue ISPF services
- ISPTASK is not in a wait state (on SVC 6)
- Can't post ISPTASK to issue service request
**ISPF Debugging Tools**

- ISRDDN
- ISPVCALL
- Dialog Test
- Other

**ISRDDN**

- Scrollable list of allocated DD's and associated data set names
- Invoked using TSO ISRDDN or DDLIST commands
- Documented in the ISPF User's Guide Volume I
ISRDDN...

- **Line commands (actions)**
  - E - Edit data set
  - B - Browse data set
  - V - View data set
  - M - Display enhanced member list
  - F - Free the ddname
  - C - Compress a data set
  - Q - Show enqueue information
  - I - Show data set information

---

ISRDDN...

- **Special pseudo-ddnames**
  - APF
  - LPA
  - PARMLIB

- **Enqueues & enq contention**
  - ENQ
  - CON

- **Browsing storage & loaded modules**
  - LOAD modname
  - WHERE modname
  - BROWSE modname [+offset]
  - DISASM

- **Primary commands**
  - Data set commands
    - FIND string
    - RFIND
    - LOCATE ddstring
    - ONLY ddstring
    - EXCLUDE ddstring
    - RESET
    - SHORT or LONG
    - MEMBER name [ddstring]
    - SELECT modname
    - COUNT [ddstring]
    - CLIST [ddstring]
    - SAVE [ddstring]
    - DUPLICATES [ddstring]
    - MLIST
    - CUSTOM
ISPVCALL

- Produces trace with the following:
  - System and session information
  - Cached panels
  - Active command tables
  - ISPF configuration table values
  - Allocated DD's
  - LIBDEF status
  - Task structure
  - SVC table
  - ISPF command stack
  - A legend
  - Usage tips
  - Module trace information
    - ISPLINK calls
    - ISPEXEC calls
    - ENQ info
    - MSG changes
    - SVC99 list
- Trace output written to dynamically allocated variable blocked data set
- Trace started and stopped using TSO ISPVCALL command

Panel Trace

- Provides debugging capability for panel processing in ISPF applications
- Traces the panel service calls (DISPLAY, TBDISPL, and PQUERY)
- Traces ISPF processing of panel statements in )ABCINIT, ABCPROC, )INIT, REINIT, and )PROC sections of a panel
- Trace output written to dynamically allocated variable blocked data set
- Documented in Appendix C of the ISPF Dialog Developer's Guide
- Trace started and stopped using TSO ISPDPTRC command
Panel Trace

- Provides debugging capability for ISPF File Tailoring applications
- Traces the File Tailoring service calls (FTOPEN, FTINCL, FTCLOSE, and FTERASE)
- Traces ISPF processing of skeleton statements
- Trace output written to dynamically allocated variable blocked data set
- Documented in Appendix C of the ISPF Dialog Developer’s Guide
- Trace started and stopped using TSO ISPFTTRC command

Dialog Test

- Dialog Test - ISPF option 7
  - Invoke dialog functions (option 7.1)
  - Display panels and/or messages (option 7.2)
  - List variables (option 7.3)
  - View/modify ISPF tables (option 7.4)
  - Browse ISPF log (option 7.5)
  - Run ISPF services (option 7.6)
  - Trace Dialog service calls (option 7.7.1)
  - Trace variables (option 7.7.2)
  - Breakpoint services (option 7.8)
Other Debugging Tools

- ISPF command parameters
  - TEST / TESTX / TRACE / TRACEX
- LIST service
  - Dialog can use this to write out lines to the ISPF list data set
- LOG service
  - Write message to ISPF log data set
- ENVIRON command
  - TPUT/TGET trace
  - Read Partition Query buffer
  - Enable dump

ISPF Productivity Tool (IPT)

- Extends the productivity of ISPF
  - Seamless integration with ISPF
    - Enhanced functionality integrated into the standard ISPF member and data set list functions
  - Create and use Object Lists which can contain items such as
    - Data sets (PDS, sequential, VSAM, tape, migrated)
    - z/OS UNIX files
    - DB2 tables
    - APF, LPA, and Linklist libraries
  - Minimize panel navigation and improve productivity
    - Shortcuts
    - IPT Commands
    - Extensive “Find” capabilities across multiple files
      - DBCS and Hexadecimal searches
      - Case sensitive searches
      - Limited to specific columns
    - Global “change” function across multiple members
  - Reduce keystrokes