DFSMS Basics: How to Write ACS Routines Hands-on Lab (Section 1)

Neal Bohling and Tom Reed, IBM

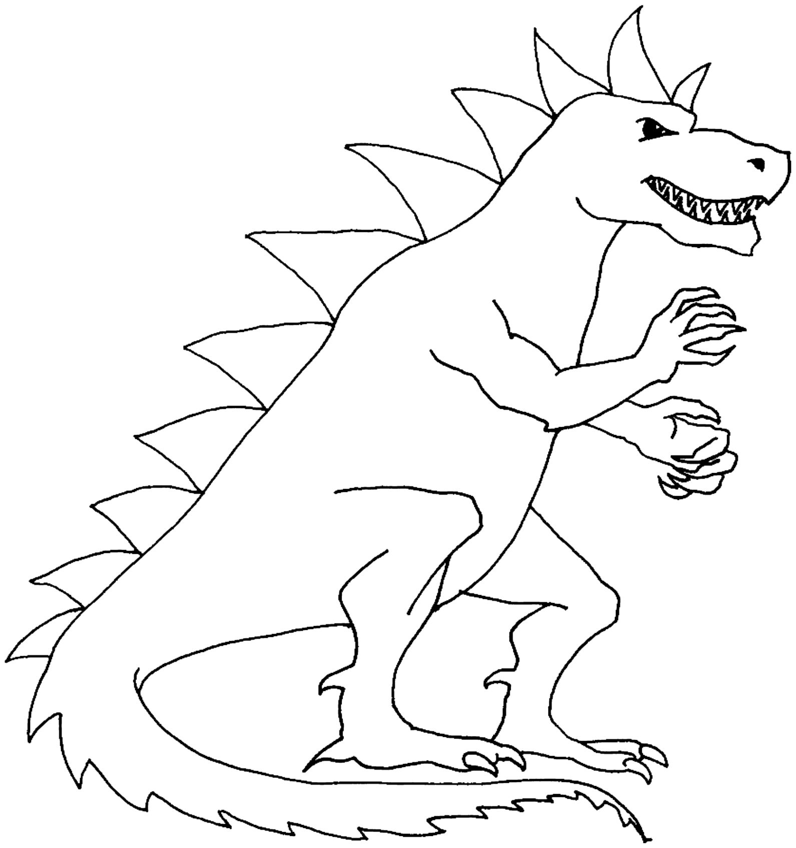
August 8, 2014
Session Number 16115

#SHAREorg

SHARE in Pittsburgh 2014

Agenda

- Short intro to SMS
- Configuration Overview
- ACS Overview
- **LAB**



Introduction to SMS

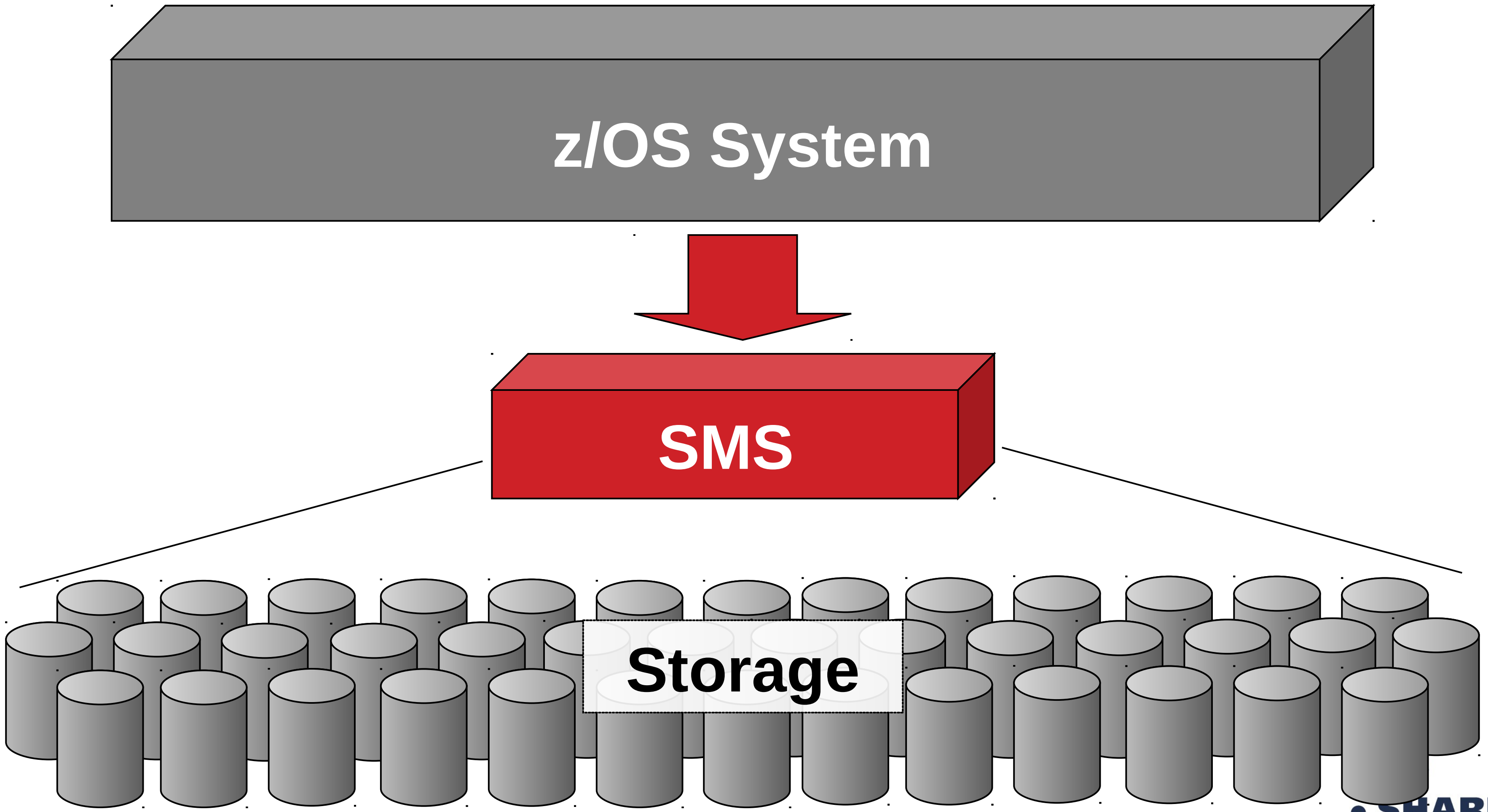
S STORAGE

M MANAGEMENT

S SUBSYSTEM

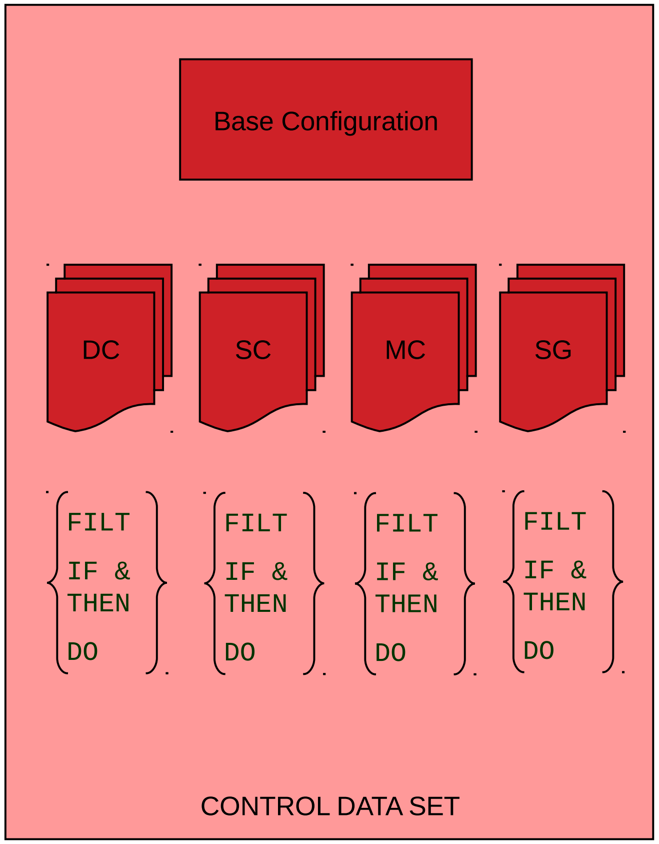
- DFSMS facility designed for automating and centralizing storage management.
- Allows you to define
  - Data allocation characteristics
  - Performance and availability goals,
  - Backup and retention requirements
  - Storage requirements
- Benefits:
  - Improves storage space use
  - Allows central control
  - Enables you to manage storage growth more efficiently

Introduction to SMS Environment



SMS Configuration Includes

- **Base Configuration**
  - Installation defaults (device geometry)
  - Systems included in the *SMS complex*
- **Constructs**
  - Data Classes – basic allocation defaults
  - Storage Classes – access attributes
  - Management Classes – migration information
  - Storage Groups – collection of volumes
- **Automating Class Selection (ACS)**
  - User-defined script
  - One per construct
  - Selects construct based on various criteria
- **Stored in the Control Data Sets (CDS)**
  - Active CDS – ACDS
  - Source CDS – SCDS
  - Communication CDS - COMMDS



Introduction to ACS Environment

- **What is an ACS Routine?**
  - User written code
  - Stored in a flat file or PDS
  - Selects which SMS classes and groups to assign
  - One per type of construct (DC / SC / MC / SG)
  - They run at ALLOCATION time (with some exceptions)
  - Minimal configuration should have a Storage Class and Storage Group ACS routine

ACS Standard Flow / Example

```
PROC DATACLAS
/* MY FILTLISTS */
FILTLIST ORIGSTG +
  INCLUDE('LARRY','CURLY',MO*) +
  EXCLUDE('SHEMP')

/* LOGIC */
IF( &HLQ EQ &ORIGSTG ) THEN DO
   SET &DATACLAS = 'STGDC'
   END
ELSE SET &DATACLAS = ''

WRITE 'DATACLAS = ' &DATACLAS
END
```

1. PROC

2. FILTLIST

3. LOGIC

4. SET

5. END

Introduction to ACS Environment (cont)

- **ACS Language Statements**
  - `PROC` - beginning of routine
  - `FILTLIST` – defines filter criteria
  - `DO` – start of statement group
  - `SELECT` – defines a set of conditional statements
  - `IF` – conditional statement
  - `SET` – assigns a read/write variable
  - `WRITE` – sends message to end user
  - `EXIT` – immediately terminates ACS routine
  - `END` – end of statement group
  - `/* COMMENT */` - comments a line

Variables

- **Always start with an &**
- **Two types: READ ONLY, READ/WRITE**

- **READ ONLY**
  - 47 different variables
  - Contain data set and system information
  - Reflect what is known at the time of the request
  - Can only be used for comparison
  - Examples: &DSORG, &DSNTYPE, &SIZE, &HLQ

- **READ/WRITE Variables**
  - Used to assign values
  - Only 4 variables
    - &DATACLAS
    - &STORCLAS
    - &MGMTCLAS
    - &STORGRP

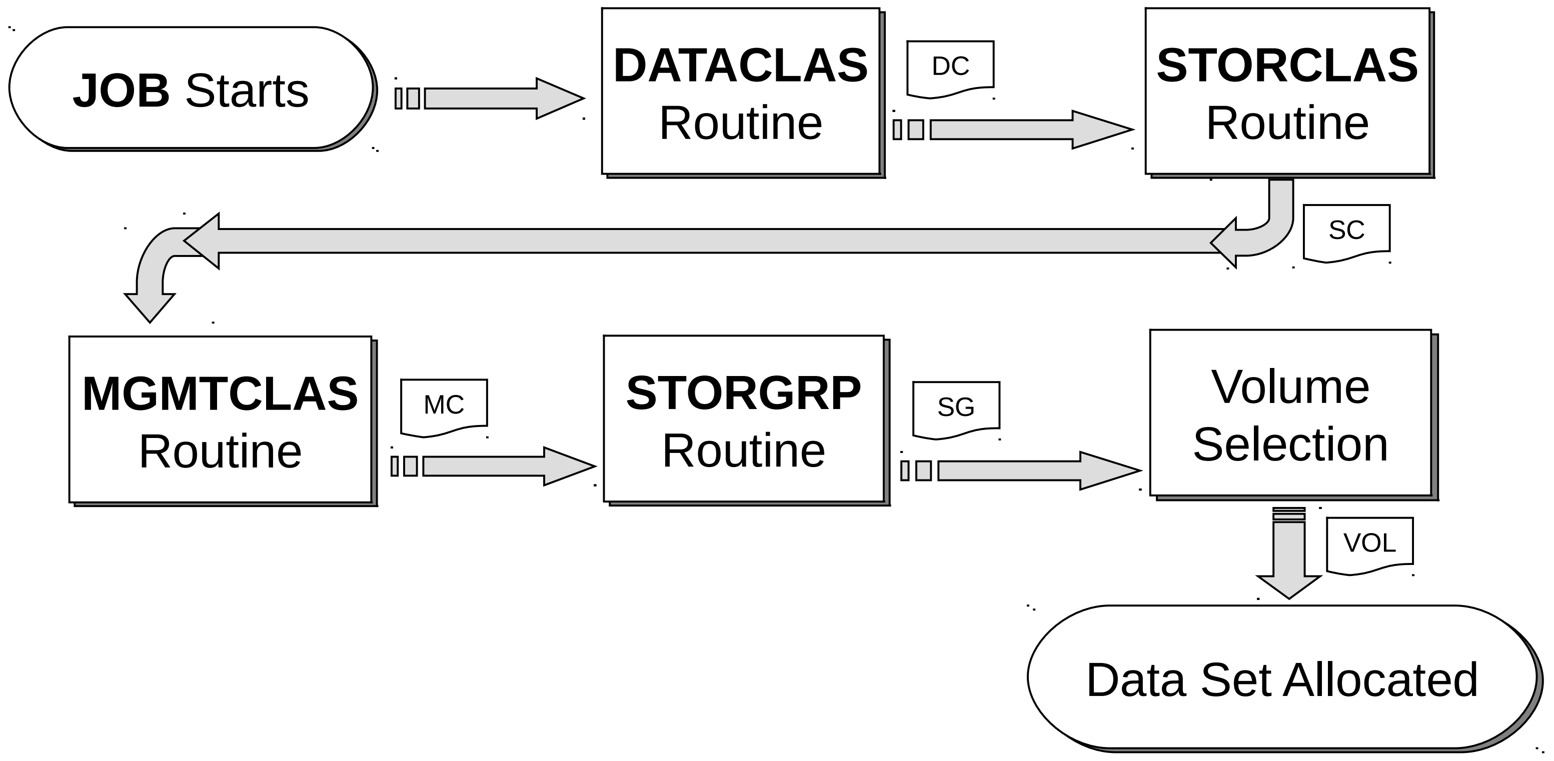
ACS Standard Flow / Example of Variables

```
PROC DATACLAS
/* MY FILTLISTS */
FILTLIST ORIGSTG +
  INCLUDE('LARRY','CURLY',MO*) +
  EXCLUDE('SHEMP')

/* LOGIC */
IF( &HLQ EQ &ORIGSTG ) THEN DO
   SET &DATACLAS = 'STGDC'
   END
ELSE SET &DATACLAS = ''

WRITE 'DATACLAS = ' &DATACLAS
END
```

&HLQ is READ ONLY

&DATACLAS is READ/WRITE

ACS Routine Process Flow



Introduction to ACS Environment (cont)

- **ACS General Rules**
  - ***Know your logic before you code***
  - **Keep them simple and straightforward**
    - Minimize exceptions
    - Maximize FILTLIST usage
  - **Keep them easy to maintain and understand**
    - Use SELECT instead of IF when possible
    - EXIT the routine as soon as possible
    - Use OTHERWISE whenever possible
    - Group selections by SET
    - Comments, comments, comments

A Few “Gotchas”

- **Numeric constants are easy: just numbers**
  - *&NQUAL = 5*
- **Suffixes : sizes require KB or MB suffix**
  - *&MAXSIZE = 100MB*
- **String literals are in single quotes**
  - *&HLQ = ‘TEST’*
- **Masks are in NOT in quotes**
  - &DSN = SYS1.*LIB
- **&& is AND, | is OR**
- **Watch for fall-through logic in your IF and SELECT**

From Idea to Active – Enabling ACS

- **Write the ACS Routines**
  - Saved in a text format

- **Translate ACS Routines**
  - Converts to byte code and inserts into the SCDS

- **Validate the SMS Configuration**
  - Verifies your construct allocation (do they all exist?)

- **Activate the SMS Configuration**

- ***Note: translate / validate from the highest z/OS level in your PLEX***

Lab Time

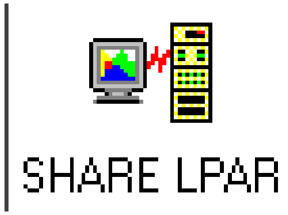
See your handout and start the lab!

Lab Agenda

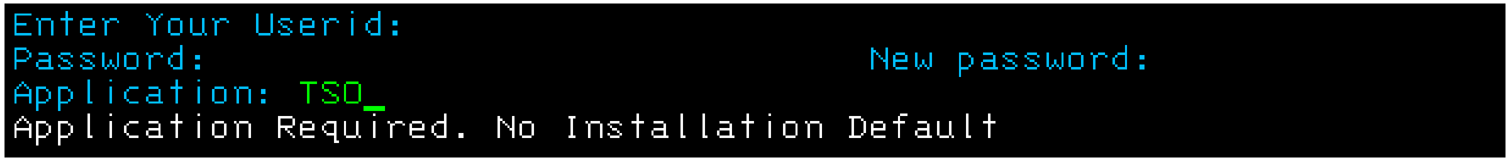
- Get Logged In
- Resources Overview
- Set up ISMF Profile
- 10 Labs
  - We'll do the first 5 together
  - Last 5 are for you to play!

Logging In

- **Find SHARE LPAR icon on the Desktop**



- **Enter TSO** next to Application:



```
Enter Your Userid:
Password:                                    New password:
Application: TSO_
Application Required. No Installation Default
```

- **Log in with your workstation's USERID**
  - SHARA01 thru SHARA20

- **Note:** uses default key mappings:
  - ENTER: New line
  - CTRL: Execute / enter
  - F7/F8: Page up / page down

Lab Pre-stuff

- **Set yourself up as a Storage Administrator**
  - Enter **ISMF** (command ISMF from main menu)
  - **0** - Profile Options
  - **0** - User Mode Selections
  - **2** - Storage Administrator
  - End/Exit 3 times

Resources

- **Lab TSO USERIDs**
  - SHARA01 thru SHARA20
- **Lab Data Sets**
  - SHARAxx.S16115.ACS
  - SHARAxx.S16115.ACS.EXAMPLES
  - SHARAxx.S16115.SMS.SCDS
- **Publications**
  - DFSMS Storage Administration Reference (handout)
    - Link: https://ibm.biz/BdRmgJ
  - DFSMS Implementing System-Managed Storage
  - DFSMS Using the Interactive Storage Management Facility
  - Links are in SHARAxx.S16115.DOCLINKS

Lab Data Sets

- **SHARAxx.S16115.ACS – edit / use these for the lab**
  - DCLAB – data class
  - SCLAB – storage class
  - MCLAB – management class
  - SGLAB – storage group

- **SHARAxx.S16115.SMS.SCDS**
  - SMS Configuration SCDS

- **SHARAxx.S16115.ACS.EXAMPLES:**
  - Minimal configuration examples:
    - SCMIN and SGMIN
  - Example routines:
    - DCEXAMPL, SCEXAMPL, MCEXAMPL and SGEXAMPL
  - Example solutions
    - DCANSWER, SCANSWER , MCANSWER and SGANSWER

Lab Configuration

- **Available SCDS:**
  - **SHARAxx.S16115.SMS.SCDS**

- **Available DATACLAS(s)**
  - Default, Extended, HFS, PDSE, VEAEXTND
- **Available STORCLAS(s)**
  - Default, Extended, GSPACE
- **Available MGMTCLAS(s)**
  - RLSEIMM
- **Available STORGRP(s)**
  - Default, Extended

Lab 1

- **Create the Framework for Each ACS Routine**
- Put the PROC and END statements for each routine
  - Open DCLAB of data set **SHARAxx.S16115.ACS**
  - Enter statements:

```
PROC &DATACLAS
END
```

  - Repeat for
    - SCLAB (PROC &STORCLAS)
    - MCLAB (PROC &MGMTCLAS)
    - SGLAB (PROC &STORGRP)

Lab 1b

- **Translate Your Routines**
- ISMF Option **7** - “Automatic Class Selection”
- Option **2** - “Translate”
  - For CDS, use your **SHARAxx.S16115.SCDS**
- On the translate page, enter the data set information:

– SCDS Name:	**'SHARAxx.S16115.SMS.SCDS'**
– ACS Source Data Set:	**'SHARAxx.S16115.ACS'**
– ACS Source Member:	**DCLAB**
• Repeat for SCLAB, MCLAB, SGLAB	
– Listing Data Set:	**LISTING**

Lab 2

- **Create some Filter Lists**
  - Use the **STORAGE CLASS** ACS routine created in Lab 1 (SCLAB)
  - Modify the routine such that it contains 2 FILTLISTs
    - Create a filter of SYSTEM for HLQ of SYS1 and SYS2
    - Create a filter named SPF for wild card SPF*
- **Solution:**

```
PROC &STORCLAS
/* FILTLISTS */
FILTLIST SYSTEM INCLUDE('SYS1','SYS2')
FILTLIST SPF INCLUDE(SPF*)
END
```

- Translate the ACS routine (ISMF 7 / 2)

Lab 3

- **Add some If/Then Logic**
  - Again, use the SCLAB ACS routine created in the previous lab
  - Modify the routine such that it contains two **IF** statements
    - Compare the HLQ to the SYSTEM filter and set a null (‘’) SC
    - Compare the second-level qualifier to the SPF filter and if it matches, set a Storage Class of Default
- **Solution:**

```
PROC &STORCLAS
/* FILTLISTS */
FILTLIST SYSTEM INCLUDE('SYS1','SYS2')
FILTLIST SPF INCLUDE(SPF*)
IF &HLQ EQ &SYSTEM THEN SET &STORCLAS=''
IF &DSN(2)=&SPF THEN SET &STORCLAS='DEFAULT'
END
```

- **Translate to ensure no errors (ISMF 7/2)**

Lab 4

- **SELECT Logic**
  - Use the SC ACS routine created in the previous lab
  - Modify the routine such that it contains a SELECT statement
    - SELECT on Read/Write variable &DATACLAS
    - When incoming DC is VEAEXTND set the SC to EXTENDED
- **Solution** (new lines only):

```
SELECT (&DATACLAS)
  WHEN( 'VEAEXTND' ) SET &STORCLAS = 'EXTENDED'
  END
```

- **Remember to Translate!**

Lab 4 - Solution

SHARAxx.S16115.ACS(SCLAB):

```
PROC &STORCLAS
/* FILTLISTS */
FILTLIST SYSTEM INCLUDE('SYS1','SYS2')
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &HLQ EQ &SYSTEM THEN SET &STORCLAS=''
IF &DSN(2)=&SPF THEN SET &STORCLAS='DEFAULT'
SELECT (&DATACLAS)
  WHEN ('VEAEXTENT') SET &STORCLAS='EXTENDED'
  END
END
```

Lab 5

- **WRITE Statement**
  - Use the SC ACS routine created in the previous lab
  - Modify the routine such that it contains a WRITE statement(s) to indicate which storage class is assigned
    - Syntax: WRITE ‘message’
- **Solution:**

```
WRITE 'STORCLAS SET TO ' &STORCLAS
```

- **Remember to Translate!**

Lab 5 - Solution

SHARAxx.S16115.ACS(SCLAB):

```
PROC &STORCLAS
/* FILTLISTS */
FILTLIST SYSTEM INCLUDE('SYS1','SYS2')
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &HLQ EQ &SYSTEM THEN SET &STORCLAS=''
IF &DSN(2)=&SPF THEN SET &STORCLAS='DEFAULT'
SELECT (&DATACLAS)
  WHEN ('VEAEXTENT') SET &STORCLAS='EXTENDED'
  END
WRITE 'STORCLAS SET TO ' &STORCLAS
END
```

Lab 6

- **Add some logic to the Storage Group Routine (SGLAB)**
  - Create FILTLIST for wildcard SPF*
  - Use IF or SELECT to set the Storage Group to 'Default'
    If the second-level qualifier matches the FILTLIST

- **Translate the ACS routine**

- Solution available on Slide 31

Lab 7

- **Add similar logic to the Data Class (DCLAB) and Management Class (MCLAB) Routines**
  - Create FILTLIST for SPF*
  - If the 2nd level qualifier starts with SPF*
    - Set Data Class to Default
    - Set Management Class to RLSEIMM.

- **Translate the ACS routines**
- Solution available on Slide 32

Lab 8

- **Use SELECT Logic**
  - When the DSNTYPE is LIBRARY:
    - Set a Data Class of PDSE
    - Set a Storage Class of Default, and
    - Set a Storage Group of Default
  - When the DSNTYPE is PDS:
    - Set a Data Class of Default
    - Set a Storage Class of Default
    - Set a Storage Group of Default

- **Translate the ACS routines**
- Solutions available on Slide 33

Lab 9

- **Using whatever logic you prefer, create a rule where:**
  If MAXSIZE > 100MB and DSORG = VS, set the following values:
  - Data Class of ‘VEAEXTND’,
  - Storage Class of ‘Extended’
  - Storage Group of ‘Extended’

  - *HINT: The SC aspect of this assignment is already complete***.**
- **Translate the ACS routines**
- Solutions available on Slide 36

Lab 9 Assignment (cont)

- **Validate** the configuration
  - ISMF Option 7
  - Choose option 3 - “Validate”
  - Enter values:
    - SCDS: Use your configuration (SCDS)
    - ACS Routine Type: *
    - Listing Data Set: LISTING   (optional)

- **If all goes well, you'll see:**
  -
```
VALIDATION RESULT:  VALIDATION SUCCESSFUL
SCDS NAME:          SHARA01.S16115.SMS.SCDS
ACS ROUTINE TYPE:   *
DATE OF VALIDATION: 2014/07/22
TIME OF VALIDATION: 21:04
```

Lab 10 – Best Practices

- You may have noticed, I didn't follow my own rules:
  - No EXIT after a SET
  - No WRITE before every EXIT
  - Didn't Logically group selections by SET

- **Re-write the ACS routines to conform to the best practices.**

- **No wrong answer, use your judgment!**
- One possible solution on page 39

Lab 11 – (Optional) Test Your Routines

- Use ISMF 7.4 to test your routines
  - Option **1** for DEFINE new test

- Build a test case with the following rules:
  - Expected result: **NULL**
  - Description: **Test1**
  - DSN: **SPFA1.ANYTHING.ANYTHING**

- Store it in **SHARAxx.S16115.ACS(TEST1)**

Lab 11 – Test Your Routines

- Build another test case, **TEST2**, with rules:
  - Expected result: **EXTENDED**
  - Description: **Extended Test**
  - DSN: **MY.EXT.DATASET**
  - DSORG: **VS**
  - Size: **50000** (in KB)
  - Space_Type: **K**
  - Second_Qty: **50000**

- Store it in **SHARAxx.S16115.ACS(TEST2)**

Lab 11 – Test Your Routines

- Run both tests (option 3)
  - CDS: **'SHARAxx.S16115.SCDS'**
  - ACS Test Library: **'SHARAxx.S16115.ACS'**
  - Select all routines

- What results do you get from each test?

- Do they match expectations?

Summary

- **Upon completion of this session, you should…**
  - Have a better understanding of the ACS environment
  - Be able to write a basic ACS routine
  - Understand how to Translate and Validate an ACS routine
  - Understand how to determine what Translate and/or Validate errors occurred and why
  - Be familiar with much of the ACS syntax
  - Be able to test your ACS routines

Advanced Lab 12 (Optional)

- Create new DCA, SCA, MCA, SGA members in SHARAxx.S16115.ACS
- Implement the following rules:
  - PDS and LIBRARY data sets are assigned to:
    - DC: PDSE, SC: Default, MC: none, SG: Default
  - Data sets over 100MB are assigned to:
    - DC: Extended, SC: Extended, MGMTCLAS: RLSIMM, SG: Extended
  - Data sets with HLQ of SYSTEM are assigned null (‘’) for all routines
  - Data sets with second qualifier as “GS” are assigned to
    - SC: GSPACE
- Translate/validate/test your routines!

For Additional Experimentation

- Create your own routines!
  - Available structures are on page 7 of the lab
- Create your own structures!
  - Just be sure to use SCDS of SHARAxx.S16115.SMS.SCDS
- Try using the ACS Routine Test case Generator
  - ISMF Option 11.1

Notices & Disclaimers

Trademarks

DFSMS Basics: How to Write ACS Routines Workbook/Lab (Section 2)

Neal Bohling and Tom Reed, IBM

August 8, 2014
Session Number 16115

#SHAREorg

Lab 6 - Solution

SHARAxx.S16115.ACS(SGLAB):

```
PROC &STORGRP
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &DSN(2)=&SPF THEN SET &STORGRP='DEFAULT'

END
```

Back

Lab 7 - Solution

SHARAxx.S16115.ACS(DCLAB):

```
PROC &DATACLAS
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &DSN(2)=&SPF THEN SET &DATACLAS='DEFAULT'
END
```

SHARAxx.S16115.ACS(MCLAB):

```
PROC &MGMTCLAS
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &DSN(2)=&SPF THEN SET &MGMTCLAS='RLSEIMM'
END
```

Back

Lab 8 – Solution DCLAB

SHARAxx.S16115.ACS(DCLAB):

```
PROC &DATACLAS
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &DSN(2)=&SPF THEN SET &DATACLAS='DEFAULT'
SELECT(&DSNTYPE)
  WHEN ('LIBRARY') SET &DATACLAS='PDSE'
  WHEN ('PDS') SET &DATACLAS='DEFAULT'
END
END
```

Lab 8 – Solution SCLAB

SHARAxx.S16115.ACS(SCLAB):

```
PROC &STORCLAS
/* FILTLISTS */
FILTLIST SYSTEM INCLUDE('SYS1','SYS2')
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &HLQ EQ &SYSTEM THEN SET &STORCLAS=''
IF &DSN(2)=&SPF THEN SET &STORCLAS='DEFAULT'
SELECT (&DATACLAS)
  WHEN ('VEAEXTENT') SET &STORCLAS='EXTENDED'
  END
SELECT (&DSNTYPE)
  WHEN ('LIBRARY') SET &STORCLAS='DEFAULT'
  WHEN ('PDS') SET &STORCLAS='DEFAULT'
END
WRITE 'STORCLAS SET TO ' &STORCLAS
END
```

Lab 8 – Solution SGLAB

SHARAxx.S16115.ACS(SGLAB):

```
PROC &STORGRP
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &DSN(2)=&SPF THEN SET &STORGRP='DEFAULT'
SELECT (&DSNTYPE)
  WHEN ('LIBRARY') SET &STORGRP='DEFAULT'
  WHEN ('PDS') SET &STORGRP='DEFAULT'
  END
END
```

Back

Lab 9 – Solution DCLAB

SHARAxx.S16115.ACS(DCLAB):

```
PROC &DATACLAS
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &DSN(2)=&SPF THEN SET &DATACLAS='DEFAULT'
SELECT(&DSNTYPE)
  WHEN ('LIBRARY') SET &DATACLAS='PDSE'
  WHEN ('PDS') SET &DATACLAS='DEFAULT'
END
IF &MAXSIZE > 100MB AND &DSORG='VS' THEN
  SET &DATACLAS='VEAEXTND'
END
```

Lab 9 – Solution SCLAB

SHARAxx.S16115.ACS(SCLAB):

```
PROC &STORCLAS
/* FILTLISTS */
FILTLIST SYSTEM INCLUDE('SYS1','SYS2')
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &HLQ EQ &SYSTEM THEN SET &STORCLAS=''
IF &DSN(2)=&SPF THEN SET &STORCLAS='DEFAULT'
SELECT (&DATACLAS)
  /* NOTE THIS RULE WILL ALSO CATCH FOR LAB9 */
  WHEN ('VEAEXTENT') SET &STORCLAS='EXTENDED'
  END
SELECT (&DSNTYPE)
  WHEN ('LIBRARY') SET &STORCLAS='DEFAULT'
  WHEN ('PDS') SET &STORCLAS='DEFAULT'
END
WRITE 'STORCLAS SET TO ' &STORCLAS
END
```

Lab 9 – Solution SGLAB

SHARAxx.S16115.ACS(SGLAB):

```
PROC &STORGRP
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &DSN(2)=&SPF THEN SET &STORGRP='DEFAULT'
SELECT (&DSNTYPE)
  WHEN ('LIBRARY') SET &STORGRP='DEFAULT'
  WHEN ('PDS') SET &STORGRP='DEFAULT'
  END
IF &MAXSIZE > 100MB AND &DSORG='VS' THEN
  SET &STORGRP='EXTENDED'
END
```

Back

Lab 10 – Solution DCLAB

SHARAxx.S16115.ACS(DCLAB):

```
PROC &DATACLAS
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &DSN(2)=&SPF OR &DSNTYPE='PDS' THEN DO
  SET &DATACLAS='DEFAULT'
  WRITE 'DATACLAS = ' &DATACLAS
EXIT
END
IF &DSNTYPE='LIBRARY' THEN DO
  SET &DATACLAS='PDSE'
  WRITE 'DATACLAS = ' &DATACLAS
  EXIT
END
IF &MAXSIZE > 100MB AND &DSORG='VS' THEN DO
  SET &DATACLAS='VEAEXTND'
  WRITE 'DATACLAS = ' &DATACLAS
  EXIT
END
WRITE 'NOTHING ASSIGNED'
END /* END PROC */
```

Grouped the two rules that result in &DATACLAS='DEFAULT'

Added WRITE and EXIT statements after every SET

Watch for fall-through LOGIC

Lab 10 – Solution SCLAB

SHARAxx.S16115.ACS(SCLAB):

```
PROC &STORCLAS
/* FILTLISTS */
FILTLIST SYSTEM INCLUDE('SYS1','SYS2')
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &HLQ EQ &SYSTEM THEN DO
  SET &STORCLAS=''
  WRITE 'SYSTEM DS SC SET NULL'
  EXIT
END
IF &DSN(2)=&SPF OR
   &DSNTYPE='LIBRARY' OR
   &DSNTYPE='PDS' THEN DO
  SET &STORCLAS='DEFAULT'
  WRITE 'STORCLAS=' &STORCLAS
  EXIT
END
IF &DATACLAS='VEAEXTENT' THEN DO
  SET &STORCLAS='EXTENDED'
  WRITE 'STORCLAS=' &STORCLAS
  EXIT
END
WRITE 'STORCLAS NOT SET'
END /* END PROC */
```

Added WRITE and EXIT statements after every SET

Grouped the two rules that result in &DATACLAS='DEFAULT'

Watch for fall-through LOGIC

Lab 10 – Solution SGLAB

SHARAxx.S16115.ACS(SGLAB):

```
PROC &STORGRP
/* FILTLISTS */
FILTLIST SPF INCLUDE(SPF*)
/* LOGIC */
IF &MAXSIZE > 100MB AND &DSORG='VS' THEN DO
  SET &STORGRP='EXTENDED'
  WRITE 'STORGRP=' &STORGRP
  EXIT
END
IF &DSN(2)=&SPF OR
   &DSNTYPE='LIBRARY' OR
   &DSNTYPE='PDS' THEN DO
  SET &STORGRP='DEFAULT'
  WRITE 'STORGRP='&STORGRP
  EXIT
END
WRITE 'STORGP NOT SET AT END'
END /* END IF */
```

Grouped the rules that result in &DATACLAS='DEFAULT'

Back

Watch for fall-through LOGIC