E52

V8 Implementing User Written Automation with the Operations Manager Interface

Pedro Vera

St. Louis, MO **Sept. 30 - Oct. 3, 2002**

RETURN TO INDEX

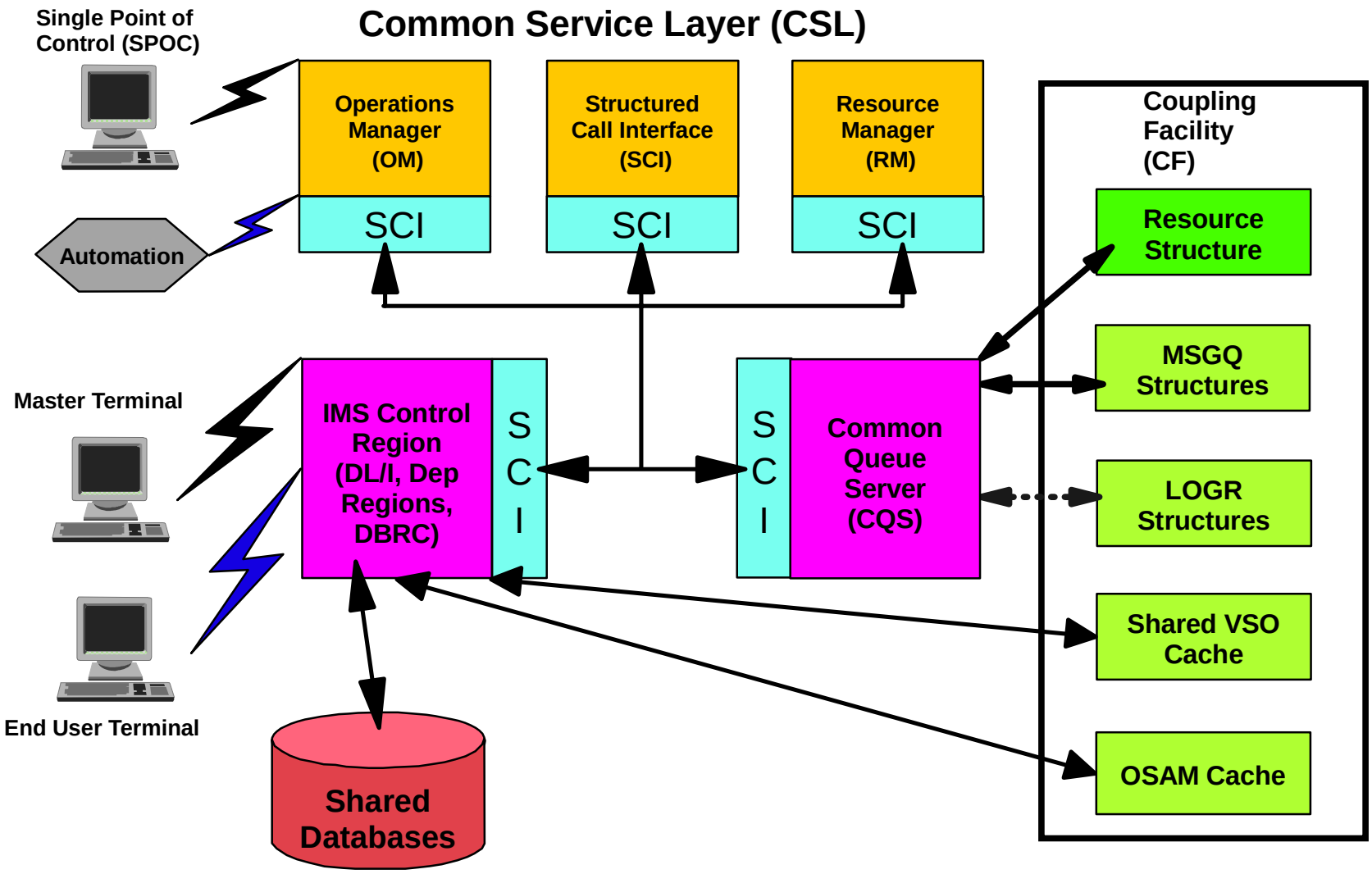
speaker notes

Biography:

I was an MVS system programmer for several years and then in MVS tools support for several years.

Now I work in IMS development, working on ISPF oriented programs.

IMS V8 IMSplex



speaker notes

IMS allows multiple IMS systems to work together as a single image, sharing databases and / or message queues. This environment is called an IMSplex. To simplify systems management, IMS V8 adds new functions such as the Operations Manager Application Programming Interface (OM API) which allows operator commands to be entered and the responses to be retrieved.

The IMS operator can issue commands to any or all of the command processing members of the IMSplex with a program that utilizes the OM API. That is, the IMSplex can be managed from a single place! This 'Single Point of Control' is referred to as a SPOC.

'Single' does not mean 'only': any number of IMS operators can start their own SPOC sessions.

Creating IMSplex

Started tasks (SCI, OM, RM, and IMS) must have the same name in their startup parms:

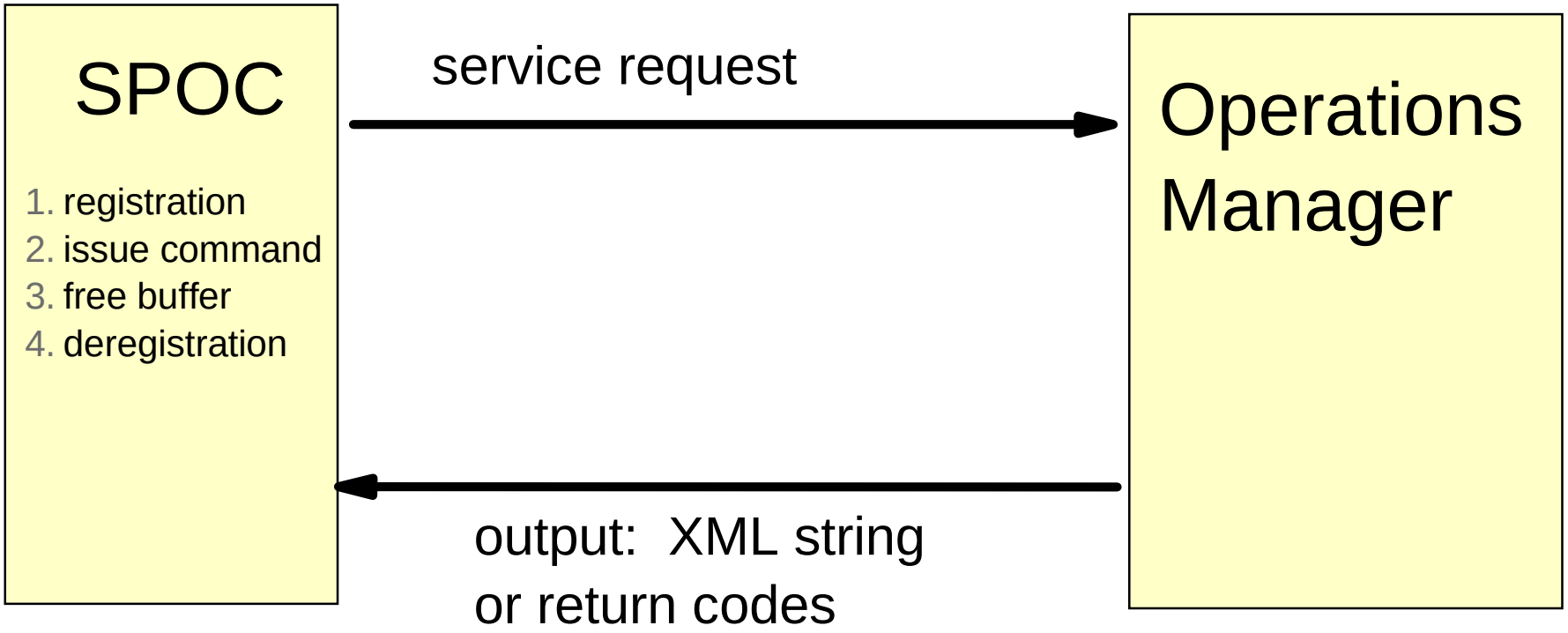
```
IMSPLEX(NAME=PLEX1)   /* Group name  = CSLPLEX1 */
```

Others may participate by using CSLSCREG macro to register as part of the IMSplex.

blank page

OM API

Macros are provided to request SCI, OM, and RM services.



SPOC overview

- Registers with SCI

- Accepts or creates command
- Send command to OM
- Processes reply (XML format)
- Release storage

- Deregisters from SCI

Regstration

In order to use any OM API service, you must register with SCI

- provide name of the IMSplex -
  - use 5 character suffix
- provide 16 byte buffer for SCI token

Registration - CSLSCREG

```
    CSLSCREG                                                    X
          FUNC=REGISTER,                                        X
          PARM=REGPARM,       WORK AREA                         X
          IMSPLEX=PLEXN,      IMSPLEX NAME                      X
          MBRNAME=MBR,        MEMBER NAME                       X
          RETCODE=RC,         RETURN CODE                       X
          RSNCODE=RSN,        REASON CODE                       X
          SCITOKEN=SCITOKN,   TOKEN RETURNED                    X
          TYPE='AOP'          MEMBER TYPE
MBR       DC CL8'PEDRO'       MEMBER NAME
PLEXN     DC CL5'PLEX1'       IMSPLEX NAME
RC        DS F                RETURN CODE
RSN       DS F                REASON CODE
SCITOKN   DS CL16             RETURNED BY REG
REGPARM   DS XL(SREG_LN)      WORK AREA
    CSLSCREG  FUNC=DSECT
```

Issue Command

Submit the comand to OM, which will route the command to the correct command processors.

You get back a long string of XML.

You may set a wait value, where OM will return a response even if not all IMSs have responded.

An ECB is optional. You can use it to get response asynchronously.

Issue command - CSLOMCMD

```
   CSLOMCMD FUNC=COMMAND,                X
         CMD=MYCMD,                      X
         CMDLEN=MYCMDLEN,                X
         OUTLEN=MYXMLLEN,                X
         OUTPUT=XMLADDR,                 X
         RQSTTKN1=RQSTTKN,               X
         PARM=CMDPARM,                   X
         RETCODE=RC,                     X
         RSNCODE=RSN,                    X
         SCITOKEN=SCITOKN
MYCMD     DC  CL12'QRY IMSPLEX'
MYCMDLEN  DC  A(12)
RQSTTKN   DC  CL16'MYTOKEN'
MYXMLLEN  DS  F
XMLADDR   DS  A
```

Free XML buffer

After you have finished with the XML buffer, you must free it. If not, your address space will eventually run out of storage.

The storage is managed by SCI.

release storage - CSLSCBFR

```
         L      R2,XMLADDR
         CSLSCBFR FUNC=RELEASE,                X
               PARM=BFRPARM,                   X
               BUFFER=(R2),                    X
               SCITOKEN=SCITOKN,               X
               RETCODE=RC,                     X
               RSNCODE=RSN

BFRPARM  DS     XL(SBFR_PARMLN)
               CSLSCBFR  FUNC=DSECT
```

Deregistration

Use the SCI deregistration request to break the connection between your program and SCI

The SCI token becomes invalid after deregistration.

deregister - CSLSCDRG

```
   CSLSCDRG                              X
         FUNC=DEREGISTER,                X
         PARM=DRGPARM,                   X
         RETCODE=RC,                     X
         RSNCODE=RSN,                    X
         SCITOKEN=SCITOKN

DRGPARM DS    XL(SDRG_LN)
         CSLSCDRG  FUNC=DSECT
```

Sample XML (abbreviated)

```
<imsout>
<ctl>
<statime>2001.234 23:14:38.113178</statime>
<stotime>2001.234 23:14:38.114084</stotime>
<rqsttkn1>MYTOKEN          </rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmdrsphdr>
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="a" key="1"
    scroll="no" len="8" dtype=" CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4"
    scroll="no" len="8" dtype="CHAR" align="left" /></cmdrsphdr>
<cmdrspdata>
<rsp>TRAN(VIDB    ) MBR(SYS3    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

(Except that it is a continuous string!)

return codes

The return codes and reason codes are set in the progam variables you specified.

X'00xxxxxx'	IMS	see DFSCMDRR
X'01xxxxxx'	SCI	see CSLSRR
X'02xxxxxx'	OM	see CSLORR

Why a REXX SPOC?

Many operator automation programs are written in REXX and run in a NetView environment.

REXX SPOC API

- Rexx interface
  - Runs under TSO or Netview
  - May or may not be on the same MVS as OM
  - Uses SCI to communicate with OM
  - Provides "host command environment" in which IMS commands may be entered to one or more members of an IMSplex
  - Saves command responses to a rexx "stem variable" by XML statements.

REXX SPOC API sample program: rexxspoc

```
 1  /* rexx */
 2  parse upper arg theIMScmd
 3  Address TSO 'CSLULXSB'
 4  if rc = 0 then do
 5     Address IMSSPOC
 6     "IMS plex1"
 7     "ROUTE imsb"
 8     "CART  test12"
 9     "WAIT  3:00"
10     theIMScmd
11     results = cslugets('resp.','test12',"3:15")
12     say 'imsrc='imsrc   'imsreason='imsreason
13     if resp.0 /= '' then do
14         say resp.0' lines of output'
15         do indx = 1 to resp.0
16           say resp.indx
17         end
18       end
19     "END"
20   End
```

blank page

Host Command Environment

TSO command CSLULXSB sets up the host command environment for rexx. It has several subcommands.

Use the 'Address' command to send command processing to particular environments.

Example:

Address TSO "CSLULXSB"

Address IMSSPOC

"subcommand"

speaker notes

The rexx host command environment is setup by CSLULXSB. After it is issued as a TSO command, the IMSSPOC environment is available to the rexx program.

Host commands are typically quoted strings and passed directly to the host command processor.

Commands IMS, ROUTE, WAIT, CART, and END are supported and perform specific local functions. Anything else is passed to SCI as a command to be performed.

Issue subcommands to set preferences

The REXX SPOC API has concepts similar to TSO preferences panel:

```
ADDRESS  IMSSPOC
 "IMS       plex1"    /*sets IMSplex name     */
 "ROUTE ims3"         /*sets name for explicit
                       route to IMS system     */
 "WAIT   3:00"        /*sets OM timeout value*/

      /* program logic here */

 "END"                /* clean up */
```

speaker notes

IMS - sets the name of the IMSplex

ROUTE - sets the name(s) of the command processors that the will process the command.

WAIT - sets the maximum timeout valued for OM to wait for a command response. If the time is reached, OM will return with a 'timed out' return code rather than command response information.

END - cleans up control blocks.

Keeping track of commands issued

The CART subcommand is used to name the response token. Function CSLULGTS is used to retrieve the command response

```
Address IMSSPOC
   .
   .
   "CART   QRYTHETRAN"
   "QRY TRAN NAME(V*)"
   abc = CSLULGTS("out.","QRYTHETRAN","2:30")
```

speaker notes

The Command and Response Token is a way to associate a name with the command that will be issued. The token is named using the 'CART' host command. The same token name is used from the CSLULGTS function.

The CART is not case sensitive.

Looking at the command response

The CSLULGTS function will create a stem variable that will have the command response.

```
abc = CSLULGTS("out.","QRYTHETRAN","2:30")
do  x = 1 to out.0
   Say out.x
End
```

speaker notes

CSLULGTS has three parameters

1. stem name
2. cartid
3. wait time

- stem name is a set of variables in rexx. It is an arrray. The convention is to set the number of entries in the zero-eth member. out.0 has the number of entries in the array.

- use the same cartid as was previously used in the CART subcommand.

- the wait time is the longest time to wait for the command response. The process is asynchronous, so your program can do something else while the command is executing.

Rexx SPOC API sample program: rexxspoc

```
 1  /* rexx */
 2  parse upper arg theIMScmd
 3  Address TSO 'CSLULXSB'
 4  if rc = 0 then do
 5     Address IMSSPOC
 6     "IMS plex1"
 7     "ROUTE imsb"
 8     "CART  test12"
 9     "WAIT  3:00"
10     theIMScmd
11     results = cslugets('resp.','test12',"3:15")
12     say 'imsrc='imsrc   'imsreason='imsreason
13     if resp.0 /= '' then do
14         say resp.0' lines of output'
15         do indx = 1 to resp.0
16           say resp.indx
17         end
18       end
19     "END"
20   End
```

speaker notes

Highlights of sample program:

- The IMS command picked up in line 2 is executed in line10
- line 5 - sets the default command process to be IMSSPOC session values are set in lines 6 through 9
- the cartid specified in lines 8 and 11 need to be the same.
- line 10 - the command is a variable. Your implemention could be a fixed valued specified as a quoted string.
- line 11 issues function CSLULGTS to retrieve the command response.
- line 12 displays the return code and reason code
- lines 13 through 18 examine the XML statements returned by CSLULGTS.
- line 19 cleans up IMSSPOC environment.

Return codes and reason codes

Each of the IMSSPOC host commands and the CSLULGTS function set return code values. The values are provided in rexx variables:

imsrc - return code

imsreason - reason code

The values of the variables are character representations of hex values. For example, the imsrc value is c'08000008X' when a parameter is not correct. The character 'x' is at the end of the string so rexx will treat it as a character.

speaker notes

A prefix of '08' is used by automation clients. The return code may also contain SCI, OM, or IMS return codes (not listed here).

Return codes
- "00000000X" Request completed successfully
- "08000004X" Warning
- "08000008X" Parameter error
- "08000010X" Environment error
- "08000014X" System error

Warning reason codes
- "00001000X" command still executing

Parm error reason codes
- "00002000X" missing or invalid wait value
- "00002008X" missing or invalid IMSplex value
- "00002012X" missing or invalid STEM name
- "00002016X" missing or invalid token name
- "00002020X" too many parameters
- "00002024X" request token not found
- "00002028X" missing or invalid CART value

System error reason codes
- "00004000X" getmain failure

Sample job to execute sample rexx spoc program

```
//RXSPOC JOB ,
//   MSGCLASS=H,NOTIFY=USRT002,USER=USRT002
//*
//SPOC     EXEC PGM=IKJEFT01,DYNAMNBR=45
//STEPLIB DD DISP=SHR,DSN=IMS.SDFSRESL
//SYSPROC  DD DISP=SHR,DSN=LOCAL.SPOC.CLIST
//SYSTSPRT DD SYSOUT=A
//SYSTSIN  DD *
 %REXXSPOC QRY TRAN NAME(V*)
/*EOF
```

speaker notes

The batch job shown is an invocation of the batch TSO command pocessor. Refer to TSO reference manuals for complete information. Here is a summary of DD name usage:

STEPLIB- the name of the IMS SDFSRESL library

SYSPROC - the name of the rexx library. sysexec can also be used.

SYSTSPRT - output dataset

SYSTSIN - commands to be executed. In this case, it is the name of the member from SYSPROC, plus the parameters.

Sample XML (abbreviated)

```
<imsout>
<ctl>
<xmlvsn>1   </xmlvsn>
<statime>2001.234 23:14:38.113178</statime>
<stotime>2001.234 23:14:38.114084</stotime>
<rqsttkn1>TEST13          </rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmdrsphdr>
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="a" key="1"
    scroll="no" len="8" dtype=" CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4"
    scroll="no" len="8" dtype="CHAR" align="left" /></cmdrsphdr>
<cmdrspdata>
<rsp>TRAN(VIDB    ) MBR(SYS3    ) CC(   0) </rsp>
<rsp>TRAN(VIDA    ) MBR(SYS3    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```