

# C02


## Floating BMPs and N+n upgrades in a Sysplex Environment.



**Anders Öhrnberg**

IMS Local Technical Support

**Volvo IT**

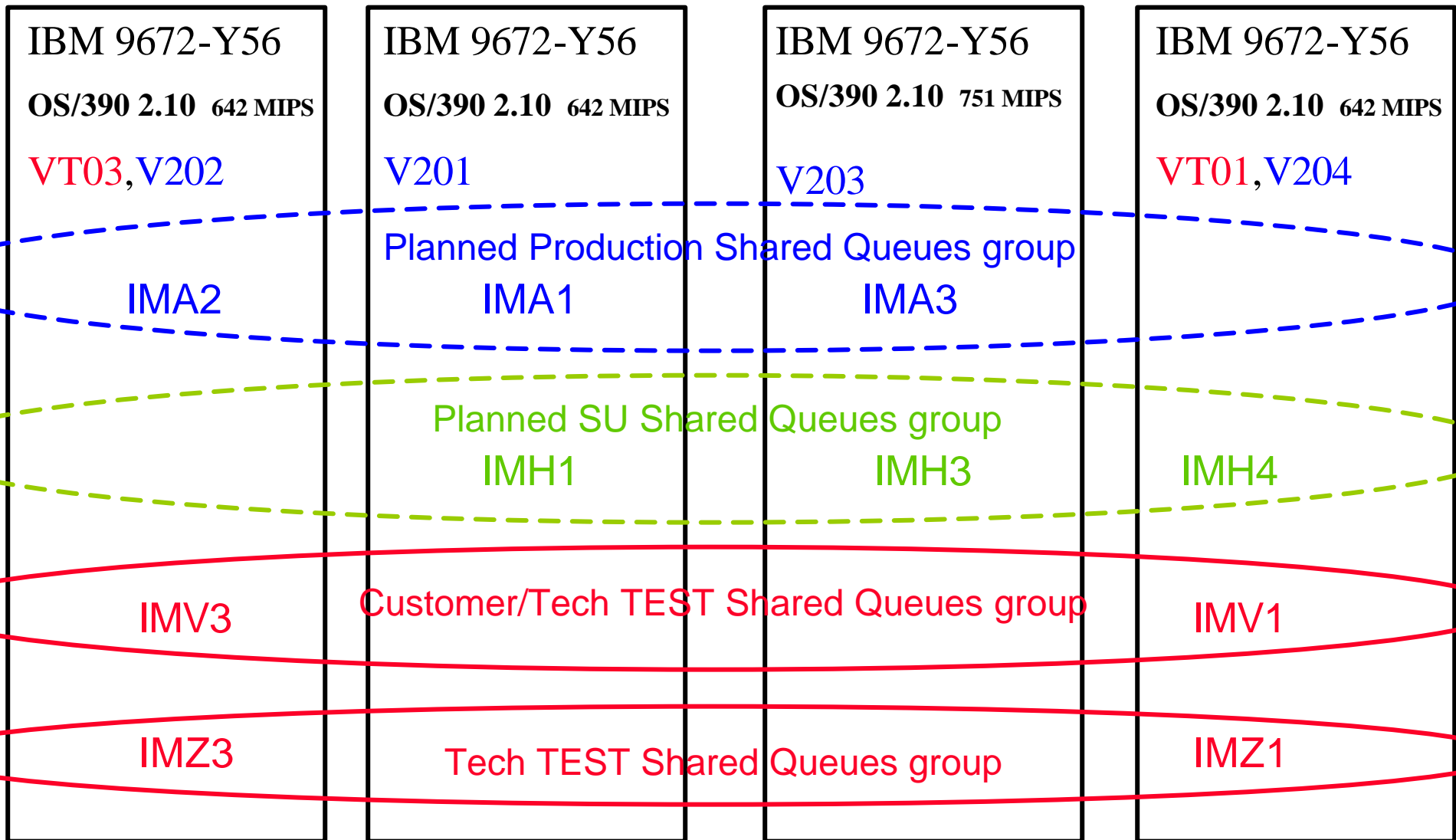
Gothenburg, Sweden 

# VOLVO



This page is intentionally left blank

# Overview Volvo IT Shared Queues Environment



As you can see the IMS namestandard follows the SMFID.

**IMZ1** in **VT01**, **IMZ3** in **VT03** etc

<u>Site</u>	<u>Total Mips</u>	<u>PROD IMS systems</u>
Gothenburg (Sweden)	2766	4

IMS V6 DB2 V6 is used.

Other Volvo IT S390 sites:

Greensboro (USA), Allentown (USA), Lyon(France), Gent (Belgium)

# **N+n upgrades in a SYSPLEX or Rolling changes across the Sysplex.**

## **WHAT IS IT ?**

- ***ROLLING IPLs.***
- ***Update ONE***
  - ***LPAR at the time.***
  - ***MEMBER of a Data Sharing Group at the time.***
- **Mixed release environment.**

To address growing availability requirements, many IT vendors, including IBM, are moving to forms of clustering as a means to attain high availability. S/390 clustering, called Parallel Sysplex technology, uses a form of clustering where all servers appear to the business application as a single server. This form of clustering, known as "single system image," enables Parallel Sysplex clusters to provide industry leading availability by allowing workloads to be balanced across multiple servers to provide near continuous availability. S/390 Parallel Sysplex clustering is the business solution for helping to ensure that applications are available through any downtime event and that revenues from sales and other business opportunities are not lost to the competition.

Another significant and unique advantage of using S/390 Parallel Sysplex technology is the ability to perform hardware and software maintenance and installations in a non-disruptive manner. Through data sharing and dynamic workload management, servers can be dynamically removed from or added to the cluster allowing installation and maintenance activities to be performed while the remaining systems continue to process work. Furthermore, by adhering to IBM's software and hardware coexistence policy, software and/or hardware upgrades can be introduced one system at a time. This capability allows customers to roll changes through systems at a pace that makes sense for their business. The ability to perform rolling hardware and software maintenance in a non-disruptive manner allows business to implement critical business function and react to rapid growth without affecting customer availability.

Since continuous availability is a prime objective of many Parallel Sysplex configurations, installing preventive maintenance to avoid known defects is a key component to meeting those objectives. To have the least impact on Parallel Sysplex availability, it is recommended that maintenance be installed and activated on one system at a time. This is known as **rolling IPLs**. The amount of time between rolling IPLs may vary depending on the amount of maintenance and the urgency to get the maintenance installed on all systems in the Parallel Sysplex configuration

## **N+n upgrades in a SYSPLEX.**

### **WHAT ABOUT IMS ?**

**Cloning IMS means sharing IMS datasets.**

**At upgrade changes are made in ACBLIB, RESLIB, MODBLKS, MACLIB, USERLIB (Exits, modifications etc), DFSMRCL0, DFSAFMD0.**

**SYS1.PARMLIB members APF list, MLPA list, LINK list**

**RECON ?**



The task of cloning your IMS subsystem is greatly simplified if you can share as many IMS datasets as possible. An upgrade can be prepared in advance and will be implemented whenever the IPL of the lpar is done. **(This can also be used in a single system to make the cutover time to a minimal.)**

When upgrading IMS to a new release of IMS, any user written exits that use IMS macros should be re-assembled with the macros from the new release. This means that separate load libraries for these exits and other upgraded modules will need to be maintained for each release or upgrade.

IBM has provided a way to upgrade one Lpar at the time (Which means one IMS in a Data Sharing group (Shared queues group) at the time. Jobs and STCs can still run in any Lpar but runs on the version of the product that runs in the Lpar it is executing.

**This presentation is a description how this can be accomplished.**

## **N+n upgrades in a SYSPLEX.**

### **Functions used:**

**SYMBOLIC Variables**

(MVS/ESA SP 5.2)

**SYMBOLIC ALIAS FACILITY** (DFSMS 1.5 / OS/390 2.7)

This page is intentionally left blank

# N+n upgrades in a SYSPLEX.

## SYMDEF VARIABLES

### LPAR VT01

```
EDIT      SYS1.PARMLIB(IEASYMT1) - 01.20
Command ==>
*****
***** Top of Data
000100     SYSDEF
000200         SYMDEF(&LVL.='00B')
000500         SYMDEF(&IMSVERS.='V601P001')
*****
***** Bottom of Data
```

### LPAR VT03

```
EDIT      SYS1.PARMLIB(IEASYMT3) - 01.16
Command ==>
*****
***** Top of Data
000100     SYSDEF
000200         SYMDEF(&LVL.='00B')
000500         SYMDEF(&IMSVERS.='V601P001')
*****
***** Bottom of Data
```

### LPAR VT01 and VT03

```
EDIT      SYS1.PARMLIB(IEASYM00) - 01.13
Command ==>
*****
***** Top of Data
000100     SYSDEF
000200         SYMDEF(&SITE.='GBG')
000300         SYMDEF(&VSYSD.= '&SYSNAME(1:2).&SYSNAME(4:1)')
000400         SYMDEF(&VSYSD.= '&SYSNAME(1:2)')
000500         SYMDEF(&SUF.= '&SYSNAME(2:1).&SYSNAME(4:1)')
000600         SYMDEF(&NODE.= '&SYSNAME(1:2)')
000700         SYMDEF(&VN.= '&SYSNAME(1:2)')
000800         SYMDEF(&SUF.= '&SYSNAME(4:1)')
000900         SYMDEF(&UU.= 'F1')
001000         SYMDEF(&RID.= '&SYSR1(6:1)')
*****
***** Bottom of Data
```

You can use one IEASYMxx member for the whole Sysplex and one member per Lpar. The Lpar members are the ones that controls the whole maintenance flow. If a variable has a value at IPL, then that will be the value used in that Lpar. So implementation plans and preparations have to be carefully done.

At the moment, there is no *official* way to change a symbol without an IPL. However, IBM has provided a program, called SYMUPDTE, that provides the capability to **change** MVS System Symbols without an IPL.

**Use this program with care.** The program is only provided to give us the ability to change symbols that are used by the Symbolic Alias Facility. IBM do not recommend using this program to change any other symbols.

You can read more about the program in:

**Parallel Sysplex - Managing Software for Availability, SG24-5451-00**

**My recommendation is to have a complete implementation plan for the whole Sysplex and not by sub product. To use this program just makes it more complicated.**

# N+n upgrades in a SYSPLEX.

## SYMDEF VARIABLES

### LPAR VT01

#### D SYMBOLS

#### RESPONSE=VT01

IEA007I STATIC SYSTEM SYMBOL  
VALUES 873

&SYSCclone. = "01"  
&SYSNAME. = "VT01"  
&SYSPLEX. = "PLEXVT"  
**&IMSVERS. = "V601P001"**  
&SITE. = "GBG"  
&SUF. = "T1"  
&SUFX. = "1"  
&VN. = "VT"  
&VSYs. = "VT"  
&VSYsID. = "VT1"

### LPAR VT03

#### D SYMBOLS

#### RESPONSE=VT03

IEA007I STATIC SYSTEM SYMBOL  
VALUES 499

&SYSCclone. = "03"  
&SYSNAME. = "VT03"  
&SYSPLEX. = "PLEXVT"  
**&IMSVERS. = "V601P001"**  
&SITE. = "GBG"  
&SUF. = "T3"  
&SUFX. = "3"  
&VN. = "VT"  
&VSYs. = "VT"  
&VSYsID. = "VT3"

The value represented by the variable string as specified in SYS1.PARMLIB member IEASYMxx can be used in /by Symbolic Alias Facility, STC JCL, picked up by REXX execs, MLPA list, APF list, Linklist, OS390 commands, TCP , OPC , Netview, System Automation for OS/390 and VTAMLST. They can **NOT** be used in batch jobs.

Example REXX:

If SYMDEF(&IMSVERS.='V601P001') is specified in IEASYMxx, then

imsvers = MVSVAR('SYMDEF','IMSVERS') can be used in a REXX exec to pick up the value of variable IMSVERS in the LPAR the REXX exec is executing.

Example VTAMLST:

IMZ&SUF~~X~~.VOL APPL AUTH=(ACQ,PASS,NOTSO,VSPACE),EAS=5,PARSESS=YES

Variable SUFX is used and it represents the last character in the SMFID. In this case it will be 1 for VT01 or 3 for VT03.

D SYMBOLS is the command to display the active variables in the LPAR where the command is issued.

# N+n upgrades in a SYSPLEX.

## SYMDEF VARIABLES

### MLPA

```
INCLUDE LIBRARY(SYS1.F1IM00.IMS.&IMSVERS..RESLIB)
      MODULES(DFSMRCL0,DFSAFMD0)
```

### APF List

```
APF ADD DSNAME(F1IMZ&SUFEX..IMS.&IMSVERS..RESLIB)           SMS
APF ADD DSNAME(SYS1.F1IM00.IMS.&IMSVERS..RESLIB)           SMS
APF ADD DSNAME(F1IMZ&SUFEX..IMS.&IMSVERS..USERLIB)         SMS
APF ADD DSNAME(F1IMZ&SUFEX..IMS.&IMSVERS..MATRIFA)         SMS
```

### LINK List

```
LNKLST ADD NAME(VT1.PUO00B) DSNAME(SYS1.F1IM00.IMS.&IMSVERS..RESLIB)
LNKLST ADD NAME(VT1.PUO00B) DSNAME(SYS1.F1IM00.IMS.&IMSVERS..LINKLIB)
```



When the IPL is done the values are picked up from the IEASYMxx member valid for that Lpar. (Again it is important that the physical datasets really exists.)

If the product requires the use of LNKLST or LPALST, you ***MUST*** place the libraries in the master catalog.

## N+n upgrades in a SYSPLEX.

# SYMBOLIC ALIAS FACILITY

## PHYSICAL DATASET ALLOCATION

NAMESTANDARD (**Important**)

F1IM00.IMS.V601P001.MACLIB

SYS1.F1IM00.IMS.V601P001.RESLIB

F1IM00.IMS.V601P002.MACLIB

SYS1.F1IM00.IMS.V601P002.RESLIB

DFSMS 1.5, introduced with OS/390 V2.7, introduced a new facility called Symbolic Alias Facility. Symbolic Alias Facility provides the ability to include a system symbol in a catalog alias. The alias gets resolved at the time the data set is allocated.

However, as well as enabling the sharing of master catalogs, there is another, even more significant, benefit of the Symbolic Alias Facility. Symbolic Alias Facility can also be used to simplify the implementation of rolling restarts

**The benefits at cutover time of this approach are as follows:**

No customer JCL changes are required to switch between the two sets of IMS libraries.

There is no need to recatalog any data set.

There is no need to rename any data set.

There is no copying involved (apart from the creating the **new** libraries in the first place).

If there is a problem, all that is required is that the subsystem is stopped, the system symbol changed to set it back to the old version and the subsystem restarted.

## N+n upgrades in a SYSPLEX. SYMBOLIC ALIAS FACILITY

```
//SYMB EXEC PGM=IDCAMS
```

```
//SYSPRINT DD SYSOUT=*
```

```
//SYSIN DD *
```

```
DEF ALIAS (NAME(F1IMZ0.IMS.USERLIB) -
```

```
SYMBOLICRELATE(F1IMZ0.IMS.&IMSVERS..USERLIB))
```

```
DEF ALIAS (NAME(F1IM00.IMS.USERLIB) -
```

```
SYMBOLICRELATE(F1IM00.IMS.&IMSVERS..USERLIB))
```

```
DEF ALIAS (NAME(SYS1.F1IM00.IMS.RESLIB) -
```

```
SYMBOLICRELATE(SYS1.F1IM00.IMS.&IMSVERS..RESLIB))
```

```
/*
```

Symbolic Alias Facility provides the ability to have an alias, but have that alias resolve to different data set names depending on which system the data set is allocated on. It does this by allowing you to place a system symbol in the alias definition. DFSMS 1.5 introduced the ability by adding a new keyword on the DEFINE ALIAS command — SYMBOLICRELATE. Using this keyword, you would define the alias as follows:

```
DEFINE ALIAS (NAME(SYS1.F1IM00.IMS.RESLIB)-  
SYMBOLICRELATE(SYS1.F1IM00.IMS.&IMSVERS..RESLIB '))
```

Initially, the symbol &IMSVERS would be set to V601P001 on all systems. When you are ready to test the new release on one of the systems, you would change the &IMSVERS symbol *on that system only* to V601P002. Thus, you can have a shared catalog, and have users on two systems use the same data set name, but have each user get a different data set. This facility is very powerful.

## N+n upgrades in a SYSPLEX. **SYMBOLIC ALIAS FACILITY**

An example of a BMP using these datasets:

```
//BMPSTRT EXEC PGM=DFSRRC00,  
// PARM=('BMP,V50220,V50220,V50220T0,V50220T0,W.00,,,,,,,,,')  
//STEPLIB DD DISP=SHR,DSN=F1IMZ0.IMS.USERLIB  
// DD DISP=SHR,DSN=F1IM00.IMS.USERLIB  
// DD DISP=SHR,DSN=SYS1.F1IM00.IMS.RESLIB  
//SYSOUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSDBOUT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//*
```

This BMP can be executed anywhere in the Sysplex.

It executes towards the right maintenance level by using the aliases created with the Symbolic Alias Facility.

## N+n TEST.

```
//SYMTEST EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DEF ALIAS (NAME(F1IMZ0.IMS.SYMTEST) -
    SYMBOLICRELATE(F1IMZ0.IMS.&SYSNAME..SYMTEST))
/*
```

### Command D SYMBOLS in VT01

RESPONSE=VT01

IEA007I STATIC SYSTEM SYMBOL VALUES 798

&SYSALVL. = "1"

&SYSCLONE. = "01"

&SYSNAME. = "VT01"

### Command D SYMBOLS in VT03

RESPONSE=VT03

IEA007I STATIC SYSTEM SYMBOL VALUES 109

&SYSALVL. = "1"

&SYSCLONE. = "03"

&SYSNAME. = "VT03"



To be able to show how this actually works we had to create a scenario that shows what we access at execution time.

Above you can see the Symbolic Relate alias job step that was used in the test. This alias is now valid across the Sysplex.

Above is the command D SYMBOLS in both LPARs. (VT01 and VT03).

You can see that the value of the `SYSNAME` variables are different in each LPAR.

# N+n TEST.

EDIT F1IMZ0.IMS.VT01.SYMTEST

Command ===>

\*\*\*\*\*  
\*\*\*\*\*

000001 IMS/ESA VERSION 6

\*\*\*\*\*  
\*\*\*\*\*

EDIT F1IMZ0.IMS.VT03.SYMTEST

Command ===>

\*\*\*\*\*  
\*\*\*\*\*

000001 IMS/ESA VERSION 7

\*\*\*\*\*  
\*\*\*\*\*

This is the Files used. As it is a Sysplex and all DASD are shared, the datasets coexists, and is reachable from anywhere in the Sysplex.

Notice the content in each file VERSION 7 in the VT01 file and VERSION 6 in the VT03 file.

## N+n TEST

```
//IMZ0F JOB (510VV401100,8295,01,25),'SYM TEST VT03',CLASS=N,  
//      MSGCLASS=H  
//*+JBS BIND VT03  
//*  
//PRINT      EXEC PGM=IEBGENER  
//SYSPRINT DD SYSOUT=*  
//SYSUT1     DD DISP=SHR,DSN=F1IMZ0.IMS.SYMTEST  
//SYSUT2     DD SYSOUT=*  
//SYSIN      DD DUMMY
```

```
//IMZ0F JOB (510VV401100,8295,01,25),'SYM TEST VT01',CLASS=N,  
//      MSGCLASS=H  
//*+JBS BIND VT01  
//*  
//PRINT      EXEC PGM=IEBGENER  
//SYSPRINT DD SYSOUT=*  
//SYSUT1     DD DISP=SHR,DSN=F1IMZ0.IMS.SYMTEST  
//SYSUT2     DD SYSOUT=*  
//SYSIN      DD DUMMY
```

In this test we have to be sure that the job is executing in the LPAR we would like it to execute.

To be able to do this we used a product called Thruput manager.

This BIND card executes the job in the LPAR were an agent with the name VT03 is set.

```
//*+JBS BIND VT03
```

This BIND card executes the job in the LPAR were an agent with the name VT01 is set.

```
//*+JBS BIND VT01
```

(This can also be achieved by using a specific jobclass and have an INITIATOR started in only one lpar with that jobclass).

## N+n TEST

HASP906-3 DATE: 010327 ROOM: 8295 NAME: SYM TEST VT01 .  
\$HASP373 IMZ0F STARTED - WLM INIT - SRVCLASS BATMED - SYS **VT01**  
IEF196I ACF9CCCD USERID V038007 IS ASSIGNED TO THIS JOB - IMZ0F  
IEF403I IMZ0F - STARTED - TIME=17.03.36  
-JOBNAME STEPNAME PROCSTEP RC EXCP CPU SRB VECTOR CLOCK  
-IMZ0F PRINT 00 17 0.00 0.00 0.00 0.00  
IEF404I IMZ0F - ENDED - TIME=17.03.36  
**IGD104I F1IMZ0.IMS.SYMTEST** **RETAINED, DDNAME=SYSUT1**

ICE052I 0 END OF DFSORT  
**IMS/ESA VERSION 6**

HASP906-3 DATE: 010327 ROOM: 8295 NAME: SYM TEST VT03 .  
\$HASP373 IMZ0F STARTED - WLM INIT - SRVCLASS BATMED - SYS **VT03**  
IEF196I ACF9CCCD USERID V038007 IS ASSIGNED TO THIS JOB - IMZ0F  
IEF403I IMZ0F - STARTED - TIME=17.08.47  
-JOBNAME STEPNAME PROCSTEP RC EXCP CPU SRB VECTOR CLOCK  
-IMZ0F PRINT 00 17 0.00 0.00 0.00 0.01  
IEF404I IMZ0F - ENDED - TIME=17.08.47  
**IGD104I F1IMZ0.IMS.SYMTEST** **RETAINED, DDNAME=SYSUT1**

ICE052I 0 END OF DFSORT  
**IMS/ESA VERSION 7**

This is the Sysouts for the two jobs.

See that the retained dataset in both jobs is the alias name **F1IMZ0.IMS.SYMTEST** and **NOT** the physical dataset.

Look for the listing at the end of each job and you will see that we reached the two physical datasets that we wanted.

The different content in the file is shown in the sysouts.

One with **IMS/ESA VERSION 6** and one with **IMS/ESA VERSION 7**.

## **N+n upgrades in a SYSPLEX.**

### **WHATS GOOD ?**

- TO ACHIEVE MAXIMUM AVAILABILITY IN THE SYSPLEX.
- LESS DEGRADATION OF THE SYSPLEX AT UPGRADE
- UPGRADE ONE LPAR AT THE TIME.
- SAVE IMPLEMENTATION TIME.
- **CAN BE USED ON A SINGLE SYSTEM**

### **WHATS BAD ?**

Complex ?

Can not see in the job what physical dataset you were pointing to at execution time.

Does not cover changes in CF structures or shared datasets such as CQS SRDS that always has to be available.



This page is intentionally left blank

## TIPS:

**ACF2 apar LO77311** Has to be applied for ALIAS name with &

**Manuals and Redbooks** (<http://www.redbooks.ibm.com/> Search on IMS) :

**Parallel Sysplex - Managing Software for Availability, SG24-5451-00**

**IMS/ESA Version 6 Shared Queues, SG24-5088-00**

**IMS/ESA Shared Queues: A Planning Guide, SG24-5257-00**

**OS/390 V2R10.0 MVS System Commands, GC28-1781-10**

**MVS/ESA SP V5 Sysplex Migration Guide, SG24-4581**

**OS/390 MVS Setting Up a Sysplex, GC28-1779.**

**Parallel Sysplex Automation Guidelines, SG24-5441**

**Parallel Sysplex Operational Scenarios, SG24-2079**

## Websites:

<http://www.s390.ibm.com/products/pso/>

<http://www.s390.ibm.com/os390/support/os390tst/>

<http://www.s390.ibm.com/cfsizer/>

If you use ACF2 remember the above mentioned APAR.

## Floating BMPs

### ***FACTS we had to consider.***

- BMP can, should be able to execute in all clones where a member of a Data Sharing Group exists.
- BMP should execute in one node at the time to reduce overhead in locking and DB buffer notify.
- BMP have to execute in other clones when maintenance occurs.
- BMP have to execute in other clones when the workload is too high where it is currently supposed to execute.
- BMP execution classes (Initiators) E,V,1 can exists in all LPARs.
- A BMP can be submitted in one clone, can be converted in a second clone and executed in third.
- BMP has to be able to execute at the right maintenance level for the specific LPAR (Member in a Data Sharing Group).

This page is intentionally left blank

## Floating BMPs

### Description of Functions used.

We set IMSGROUP=IMZ0 in the control region start-up parameters.

Module DFSVC000 is Zapped with the generic ID IMZ0 and placed in the BMPs STEPLIB.

BMPs has to specify blank as IMSID in the parameter for DFSRRC00.

The BMP picks up the generic id from DFSVC000 and translates it to the right IMSID at execution time.

This means that the dependent regions can connect to the IMS system in the LPAR where it is executed.

(Where it is executed is determined by TM, INCLUDE, INIT classes and/or WLM)

This page is intentionally left blank

## Floating BMPs

**An example of a BMP running in Data Sharing group IMZ0 ready for floating and upgrades:**

```
//BMPSTRT EXEC PGM=DFSRRC00,  
// PARM=('BMP,V50220,V50220,V50220T0,V50220T0,W.00,,,,,,,,,')  
//STEPLIB DD DISP=SHR,DSN=F1IMZ0.IMS.USERLIB  
// DD DISP=SHR,DSN=F1IM00.IMS.USERLIB  
// DD DISP=SHR,DSN=SYS1.F1IM00.IMS.RESLIB  
//SYSOUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSDBOUT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//*
```



This BMP can be executed anywhere in the Sysplex.

**The Data Sharing group ID IMZ0** is picked up from DFSVC000 in F1IMZ0.IMS.USERLIB. The BMP then picks up the IMSID from the IMSGROUP parameter in the Control Regions startup parms (DFSPBxxx) at execution time.

It executes towards the **right maintenance level** by using the aliases created with the Symbolic Alias Facility. For example F1IMZ0.IMS.USERLIB point to physical dataset for example:

F1IMZ0.IMS.V601P001.USERLIB in LPAR VT01 or  
F1IMZ0.IMS.V701P001.USERLIB in LPAR VT03.

**Use of IMSID before Data Sharing (DS):**

**Floating BMPs  
IMSGROUP=**

For single IMS system:

Control Region            IMSID=IMKP

Dependent Region        IMSID=IMKP

**Use of IMSID and IMSGROUP before DS:**

For original IMS system:

Control Region            **IMSGROUP=IMA0**

Control Region            IMSID=IMKP

Dependent Region        IMSID=IMKP

**Use of IMSID and IMSGROUP before DS but new namestandard:**

For cloned IMS system:

Control Region            **IMSGROUP=IMA0**

Control Region            IMSID=IMA1

Dependent Region        IMSID=IMA0

**Use of IMSID and IMSGROUP with DS and new namestandard:**

For cloned IMS system:

Control Region            **IMSGROUP=IMA0**

Control Region            IMSID=IMA1

Dependent Region        IMSID=IMA0

## IMS V6 ADDS ABILITY TO EASILY SCHEDULE BMPs ANYWHERE IN IMSPLEX

APAR PQ21039 for IMS V6 simplifies the specification of the IMSID for users whose BMPs might execute on multiple control regions in a Parallel Sysplex. Without this enhancement, moving a BMP from one control region to another typically requires a change in the BMP IMSID parameter. With this enhancement, no changes to BMP JCL are required, even when different control regions are used. The same IMSID parameter may be used by BMPs running with different control regions. The meaning of the IMSID parameter in dependent region JCL has been expanded. It now indicates either a control region ID or an IMS control region group name. IMSGROUP is a new control region parameter which identifies the group name. Each control region in an IMSplex may specify the same value for IMSGROUP. When cloning, it is expected that this name will be the name that was the IMSID of the control region before cloning.

The above illustrates a use of the IMSGROUP parameter:

It is important to note that dependent region JCL does not have to change. Only control region JCL is changed to exploit the function supplied by IMSGROUP.

The new function applies to MPP and IFP regions as well as BMPs.

When IMSGROUP is specified for an IMS control region, IMS builds an MVS name/token pair whose name is based on the IMSGROUP specification and the IMS version and release. It stores its IMSID in the token. Dependent regions may examine the token to discover the control region's IMSID.

Dependent regions (MPP, BMP, and IFP) use the following logic to determine the IMSID to use:

.The dependent region attempts to connect to a control region whose subsystem name matches the IMSID specification of the dependent region. If there is no such control region on this MVS, the search continues.

.If ALTID is specified for the dependent region, it attempts to connect to a control region whose subsystem name matches the ALTID specification of the dependent region. If there is no such control region on this MVS, the search continues.

**.The dependent region looks for an MVS name/token pair whose name matches its IMSID and IMS version and release. If it finds such an MVS name/token pair, it finds the IMSID stored there and connects to it. If there is no such MVS name/token, the search continues.**

.If ALTID is specified, the dependent region looks for an MVS name/token whose name matches its ALTID and IMS version and release. If it finds such an MVS name/token pair, it finds the IMSID stored there and connects to it. If there is no such name/token pair, message DFS690A is issued:

```
DFS690A job.step.proc - CTL PGM NOT ACTIVE, REPLY 'WAIT', 'CANCEL ' OR 'ALT-ID'
```

## **Floating BMPs**

**When a IMS BMP has abended, it must be restarted on the same IMS.**

**IBM IMS Program Restart Facility, Program Number 5655-E14.**

**BMC Application Restart Control.**

**INCLUDE CARDS and Thruput manager**

***When a IMS BMP has abended, it must be restarted on the same IMS.***

When a IMS bmp has abended, it must be restarted on the same IMS.

**How can this be accomplished without changing the JCL? Use of BIND cards? Products ? HOW?**

**IMS Program Restart Facility, Program Number 5655-E14 solves the problems with restarting BMPs.**

As others have mentioned, IBM's Program Restart Facility doesn't require any JCL changes and allows the BMP to be restarted on any sharing partner. It can be used in conjunction with IMSGROUP which allows the BMP to connect to whichever IMS control region is active on that system. Obviously, JES2 initiator class placement is also important.

**The BMC offering is called Application Restart Control.** This product, along with a lot of other things, will let you restart a failed job anywhere within the IMSGROUP. No JCL changes are required to invoke the restart.

BMC have a bit of a different wrinkle in that ARC can also re-drive certain ABEND's within the same job step in an effort to keep the BMP up and running. I'm thinking about U0777 or U0778 ABENDS that are the result of ROLL calls issued by an application if it can't get to a locked segment. If these kinds of ABENDS are a significant percentage of your total failures, then it might make sense to take a look at the added benefit of reducing the number of failed jobs, in addition to the ability to restart a job within an IMS Sysplex.

**INCLUDE CARDS AND THRUPUT MANAGER**

As the customer points to a logical environment and it is decided centralized were the BMPs should be executed we give the customer a restart member to include. This member has a BIND card to the lpar where the BMP executed at abend time. Remember that we only execute BMPs for a customer in one lpar at the time.

## Floating BMPs

**Do we want the BMP to run anywhere ?**

How to force a BMP to run in a specific lpar ?

BMP class INITIATORS open only in those LPARs.

TM agent active in those LPARs.

We use THRUPUT MANAGER to say that a BMP that wants to connect to Data Sharing group IMZ0 only can run where an agent for IMZ0 is started. This is done with BIND (JBS) cards in the JCL.

HOW CAN WE DO THIS FOR ONLY ONE CUSTOMER.

We get the customers who is Data Shared to INCLUDE the right BIND card for agent IMA0.

The customer actually INCLUDES their logical environment (for example customerA). We then set where the customers BMPs should execute that day.

An included bind card member can look like this:

```
*****      ***** Top of Data ***  
000001      /*+JBS BIND (CUSTA)  
*****      ***** Bottom of Data *
```

A Thruput Manager agent CUSTA is activated at IMS restart by AOC in the lpar we have decided that the customers BMPs should execute. It can look like this when it is displayed.

CUSTA - ACTIVE ON VT01



**IMS Technical Conference**

## **Anders Öhrnberg**

Volvo IT

IMS Local Technical Support

Dept 8295, DA2N

S-405 08 Gothenburg

Tel: +46 31 32 16 041

Fax: +46 31 66 57 75

E-mail: **Anders.Ohrnberg@Volvo.com**