z/VSE VSAM News and Exploitation

zDO04

Wilhelm Mild
mildw@de.ibm.com

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and / or other counties.

CICS*
DB2*
DB2 Connect
DB2 Universal Database
e-business logo*
Enterprise Storage Server
HiperSockets
IBM*
IBM logo*
IMS
Intelligent Miner
Multiprise*
MQSeries*
OS/390*
S/390*
SNAP/SHOT*
Virtual Image Facility
VM/ESA*
VSE/ESA
z/VSE
VisualAge*
VTAM*
WebSphere*
xSeries
z/Architecture
z/VM
z/VSE
zSeries
System z

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

LINUX is a registered trademark of Linus Torvalds

Tivoli is a trademark of Tivoli Systems Inc.

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries

UNIX is a registered trademark of The Open Group in the United States and other countries.

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

Intel is a registered trademark of Intel Corporation.

Agenda
z/VSE VSAM new enhancements

- **VSAM Considerations and Limits**
- **VSAM new Enhancements**
- **Access VSAM files from remote platforms**
- **VSAM applications can access relational databases**
- **Tools for z/VSE VSAM files**

VSAM Considerations

"Gobi Desert" problem

- Can affect any KSDS file (including VSAM catalogs)
- Add at end, delete from beginning
- Index High Key not changed by delete
- Empty data CIs are never reused
- Impact:
  - Performance degradation
  - Cluster (catalog) growth
- Resolution:
  - Define keys (or cluster names) so that they are random
  - Frequent reorganization of file (or catalog).

VSAM Considerations

Size Limits: (rc x'1C' "No more available extents")

- **123 volumes per cluster component (data and index).**
- **16 volumes from default model.**
- **4.3 Giga-byte**
  - **4 byte RBA (Relative Byte Address):**
  - **4500 – 11000 cylinders on D/T3390 depending on CI-Size and track utilization**
  - **Compression can help**
  - **'ExtraLargeDataset' or 'XXL' max 286 GB (No-RBA access)**
- **Clusters are limited to 123 extents per component (Data/Index).**
  - This is normally only a problem if you specify a very small secondary extent.
  - VSE/VSAM will sub-allocate an extent up to 5 times.
- **Catalogs, reusable files, and unique files are limited to 16 extents.**

VSAM Considerations

Size Limits: (cont)

- **No single allocation over 16Meg records:**
- **64K control areas limitation for SHR(4)**
  - **Maximum SHR(4) file size is 51 GB**
- **16 million records per extent:** When defining files using "**RECORDS**", you can specify up to 16 million records.
  - **"RECORDSIZE":** You can request more data by specifying a larger maximum "**RECORDSIZE**" for the file.
  - **Compression can help:** Remember, if the file is compressed, VSAM uses the un-compressed maximum record length to calculate how much space to reserve for the file. This may give you more space than you actually need.

VSAM Considerations

Large DASD Support

- BIG-DASD supports dasd with up to 10017 cylinders.
- FAT-DASD supports dasd with up to 65520 cylinders
- Implementation is transparent to existing applications and JCL.
  Dasd is flagged in LISTCAT as "BIG-3390" or "FAT-3390"
- Automatic for BIG-DASD using FATDASD parm for DEFINE UCAT & SPACE
- Allocations converted to **CYLINDERS**
- Minimum data CISIZE increased to 1024 (depending on key size)
- BUFSPACE parameter may be increased
- Note: **RECOVERABLE** catalogs are not supported on "large dasd".

VSAM Considerations

Recommendations:

- Maximize size of Control Area
- Use reasonably large data Control Intervals
- Let index Control Interval size default.
- Compression will save I/Os, but will cost CPU
- Additional buffering will save I/Os
  - For sequential processing, use largest possible data CIs, and multiple data buffers.
  - For direct processing, use smallest possible index CIs, and multiple index buffers.

Dont's

- Never define a SAM-ESDS,UNIQUE or REUSable file with a big primary and a very small secondary allocation and having a long list of candidate volumes.
  - This will lead to EXTENTs wasted and those Cluster only have 16 Ext.max
- Don't try to save space with small Catalog secondary allocations
  - Catalog's can only allocate on one volume, so a big primary allocation will be used only once and never again
- Not to many AIX with UPGRADE over one BaseCluster
  - Upgrade-set is bottleneck for processing
- Don't define same Volume list in different models
  - This leads to bottle necks and performance impacts for the first Volume because all clusters have primary allocation there

Agenda
z/VSE VSAM new enhancements

- VSAM Considerations and Limits
- VSAM new Enhancements
- Access VSAM files from remote platforms
- VSAM applications can access relational databases
- Tools for z/VSE VSAM files

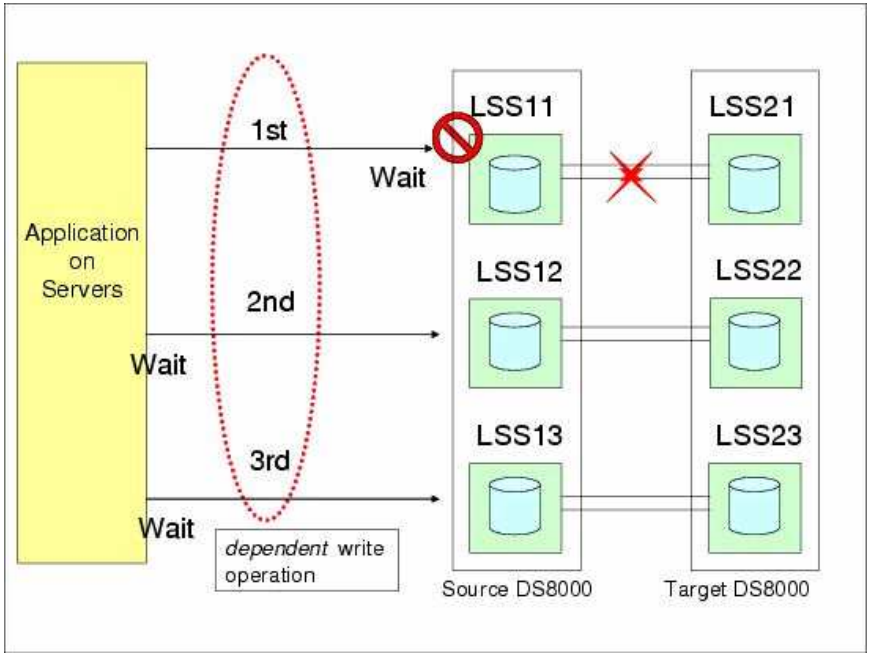
IDCAMS SNAP command Enhancements

The IDCAMS SNAP command provides an interface to the FlashCopy feature.

- New NOCOPY parameter of the IDCAMS SNAP command creates the FlashCopy. The physical copying of data to target volumes is not performed.
- New DDSR parameter of the IDCAMS SNAP command terminates the FlashCopy relation between the source and target volumes and frees the used resources.
- New parameter COPY of the IDCAMS SNAP command is now specified explicitly.
- Provided an opportunity to administrate user access rights to the IDCAMS SNAP command using the Basic Security Manager (BSM).

See New Chapter 10, “Performing an IDCAMS SNAP (FlashCopy)” “VSE/VSAM User’s Guide and Application Programming”.

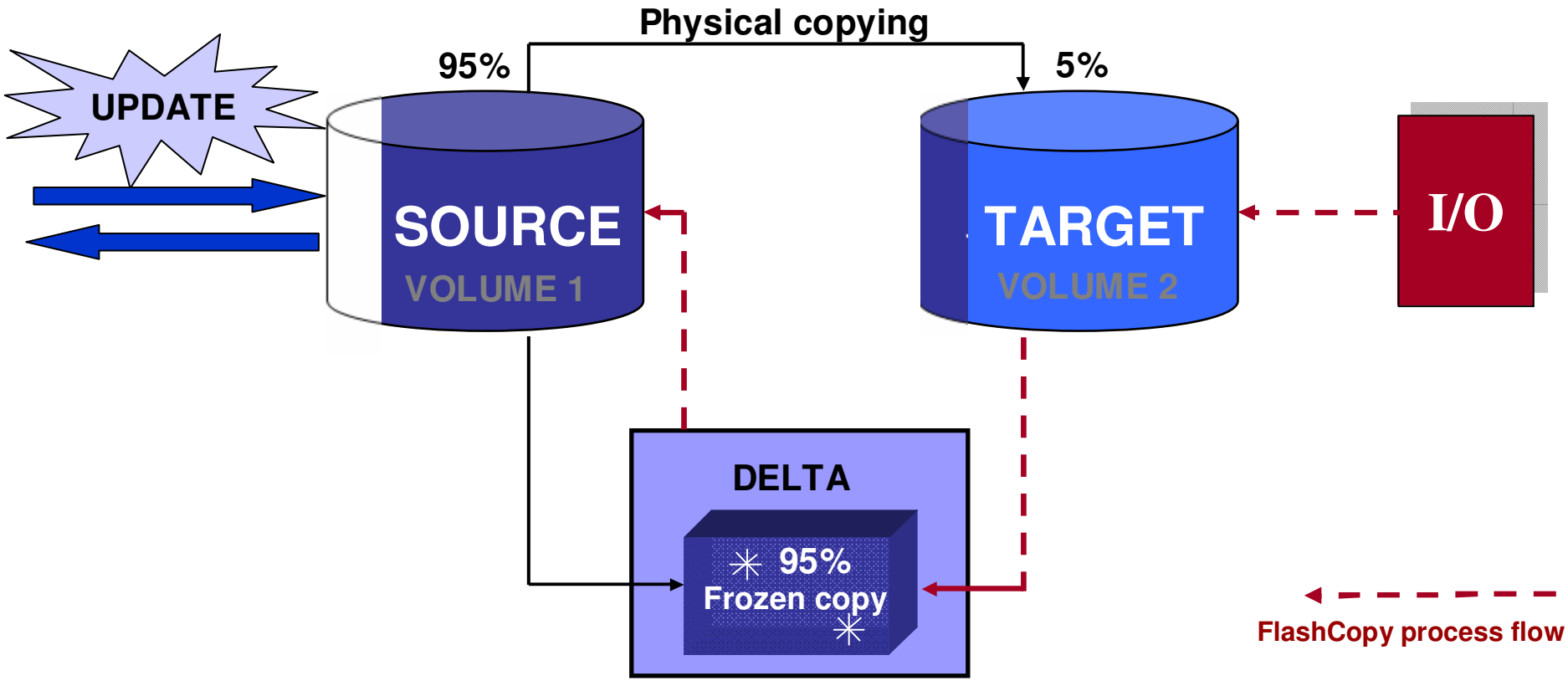
Flashcopy Consistency Group Option in z/VSE 4.3



- **Flashcopy Consistency Group** is across multiple Volumes
- **Invocation**
  - Establish: IXFP SNAP with keyword FREEZE
  - Release: IXFP DDSR with keyword THAW
- *FlashCopy Consistency Group* provides a mechanism for achieving a consistent data copy across multiple volumes without requiring that the application I/O be quiesced. In the case of production data, application impact are minimized.
- Prior to ConsistencyGroup FlashCopy, you had to first quiesce the application, establish their FlashCopy relationships, and then restart the application.

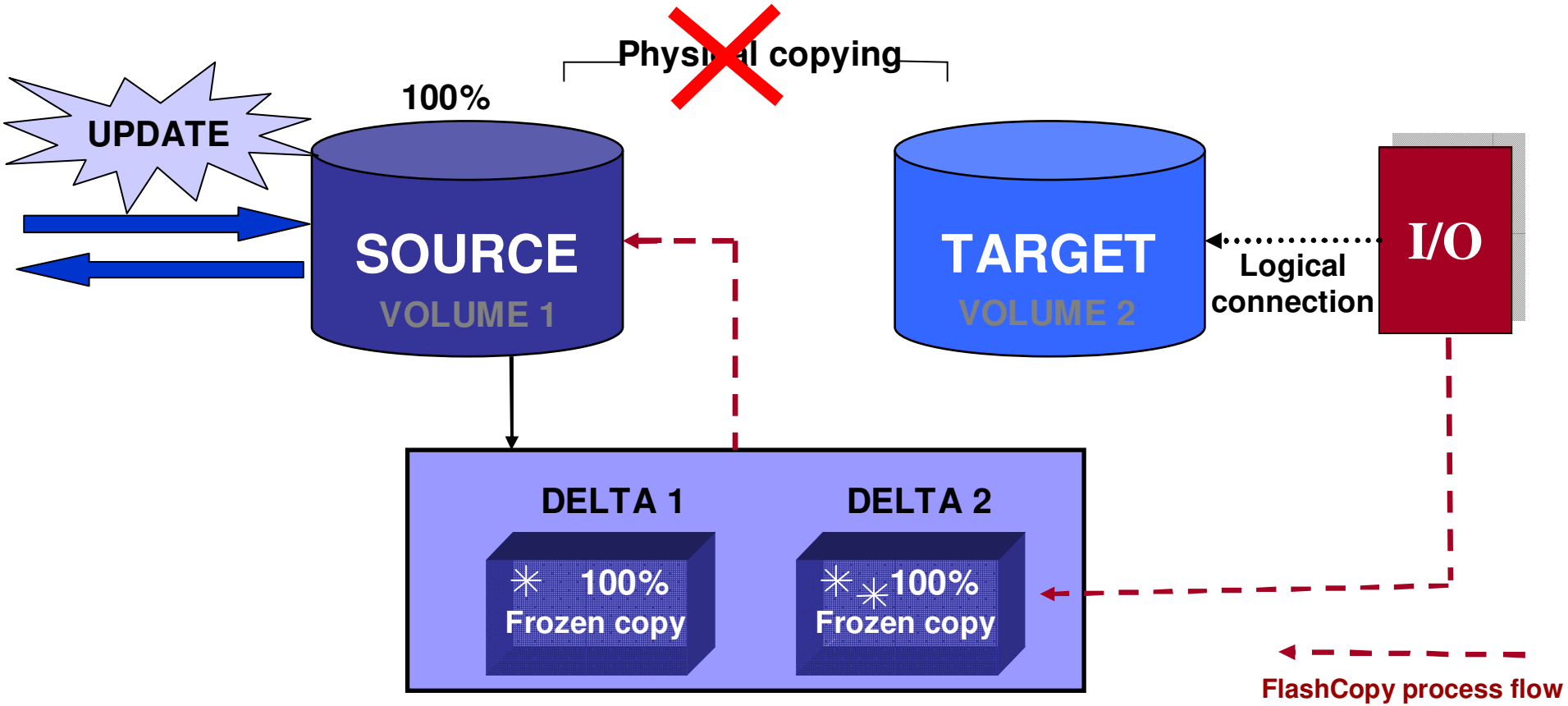
IDCAMS SNAP COPY Options

Explicit specification of the default COPY parameter of the IDCAMS SNAP command, facilitates referencing to it by other z/VSE components.



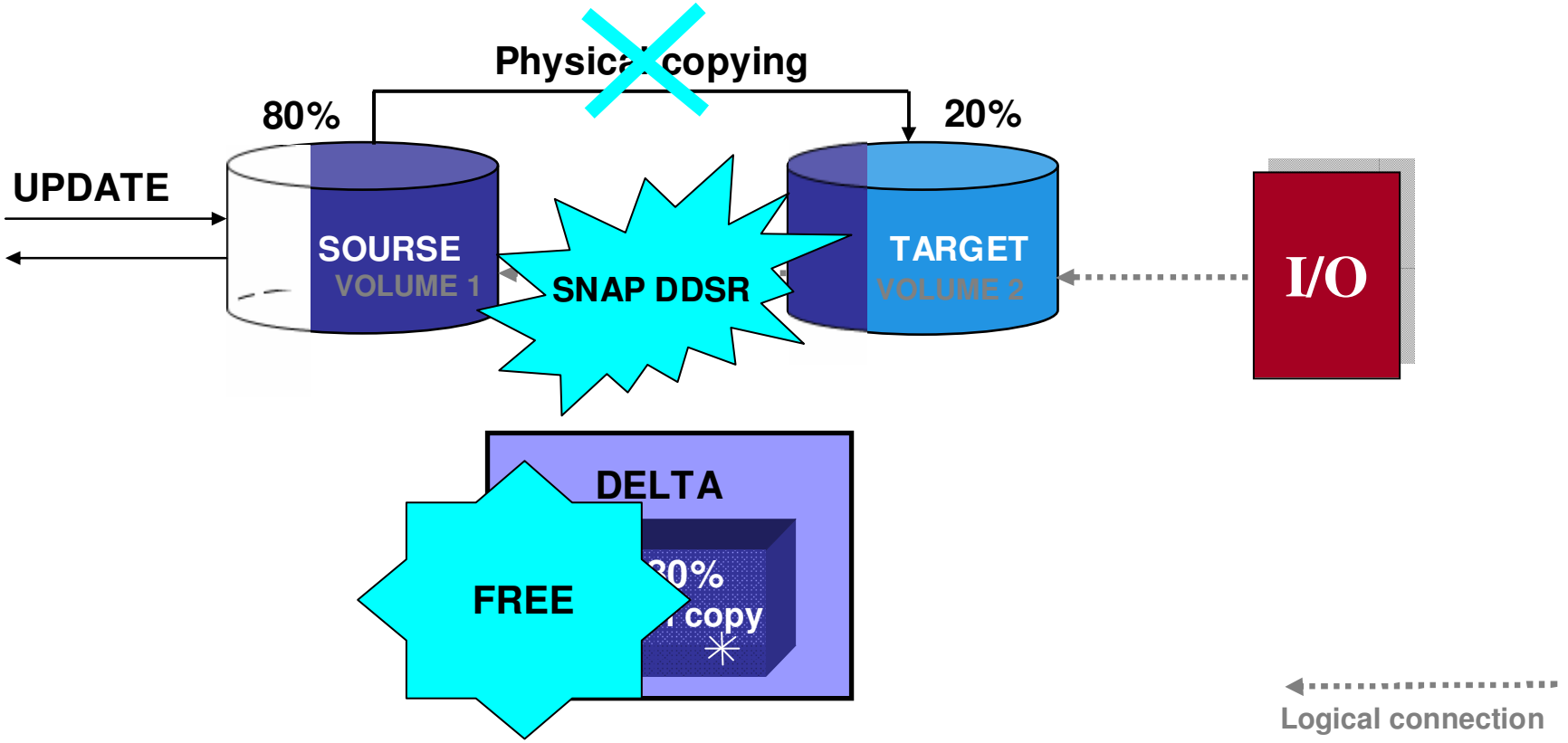
IDCAMS SNAP NOCOPY parameter

The NOCOPY parameter of the IDCAMS SNAP command allows the user to eliminate real copying of source volumes to the target volumes for temporary FlashCopy and thus eliminate the superfluous I/O activity.



IDCAMS SNAP DDSR parameter

The DDSR parameter of the IDCAMS SNAP command allows the user to delete FlashCopy relations and thus to stop unnecessary managing of a Delta File and to release internal DASD resources as soon as they are no longer needed.



Sample of SNAP COPY, NOCOPY, DDSR

COPY

```
SNAP SVOL(VSE222) TARGETVOLUMES(VSE444) COPY NOPROMPT

IMPORT CONNECT -
    OBJECTS((COPY.UCAT VOLUMES(VSE444) DEVT(3390))) -
    CAT(VSAM.MASTER.CATALOG)

BACKUP (FILE1) BPFILE(BF) SYNONYMLIST( -
      SOURCEVOLUMES(VSE222) TARGETVOLUMES(VSE444) -
      CATALOG(UCAT) SYNCATALOG(COPY.UCAT))

RESTORE OBJECTS(FILE1) BPFILE(BF) CAT(UCAT)

SNAP TARGETVOLUMES(VSE444) DDSR NOPROMPT

EXPORT COPY.UCAT   DISCONNECT
```

NOCOPY

```
SNAP SOURCEVOLUMES(VSE222) TVOL(VSE333) NOCOPY NOPROMPT

IMPORT CONNECT -
    OBJECTS((NOCOPY.UCAT VOLUMES(VSE333) DEVT(3390))) -
    CAT(VSAM.MASTER.CATALOG)

BACKUP (FILE1) BPFILE(BF) SYNONYMLIST( -
      SOURCEVOLUMES(VSE222) TARGETVOLUMES(VSE333) -
      CATALOG(UCAT) SYNCATALOG(NOCOPY.UCAT))

RESTORE OBJECTS(FILE1) BPFILE(BF) CAT(UCAT)

SNAP TARGETVOLUMES(VSE333) DDSR NOPROMPT

EXPORT NOCOPY.UCAT DISCONNECT
```

Output of SNAP COPY, NOCOPY, DDSR

COPY

```
SNAP SVOL(VSE222) TARGETVOLUMES(VSE444) COPY NOPROMPT
IDC32204I RACROUTE RESOURCE NOT PROTECTED OR BATCH SECURITY=OFF
IDC0935I IXFP/SNAPSHOT FUNCTION COMPLETED SUCCESSFULLY
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IMPORT CONNECT -
      OBJECTS((COPY.UCAT VOLUMES(VSE444) DEVT(3390))) -
      CAT(VSAM.MASTER.CATALOG)
IDC0603I  CONNECT FOR USER CATALOG  COPY.UCAT SUCCESSFUL
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

BACKUP (FILE1) BPFILE(BF) SYNONYMLIST( -
      SOURCEVOLUMES(VSE222) TARGETVOLUMES(VSE444) -
      CATALOG(UCAT) SYNCATALOG(COPY.UCAT))
IDC01300I BACKUP FILE CREATED ON XX/XX/2008 AT XX:XX:XX
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

RESTORE OBJECTS(FILE1) BPFILE(BF) CAT(UCAT)
IDC01301I RESTORE'S BACKUP FILE CREATED ON XX/XX/2008 AT XX:XX:XX
IDC01304I SUCCESSFUL DEFINITION OF FILE1
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

SNAP TARGETVOLUMES(VSE444) DDSR NOPROMPT
IDC32204I RACROUTE RESOURCE NOT PROTECTED OR BATCH SECURITY=OFF
IDC0935I IXFP/SNAPSHOT FUNCTION COMPLETED SUCCESSFULLY
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

EXPORT COPY.UCAT   DISCONNECT
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

NOCOPY

```
SNAP SOURCEVOLUMES(VSE222) TVOL(VSE333) NOCOPY NOPROMPT
IDC32204I RACROUTE RESOURCE NOT PROTECTED OR BATCH SECURITY=OFF
IDC0935I IXFP/SNAPSHOT FUNCTION COMPLETED SUCCESSFULLY
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IMPORT CONNECT -
      OBJECTS((NOCOPY.UCAT VOLUMES(VSE333) DEVT(3390))) -
      CAT(VSAM.MASTER.CATALOG)
IDC0603I  CONNECT FOR USER CATALOG  NOCOPY.UCAT SUCCESSFUL
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

BACKUP (FILE1) BPFILE(BF) SYNONYMLIST( -
      SOURCEVOLUMES(VSE222) TARGETVOLUMES(VSE333) -
      CATALOG(UCAT) SYNCATALOG(NOCOPY.UCAT))
IDC01300I BACKUP FILE CREATED ON XX/XX/2008 AT XX:XX:XX
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

RESTORE OBJECTS(FILE1) BPFILE(BF) CAT(UCAT)
IDC01301I RESTORE'S BACKUP FILE CREATED ON XX/XX/2008 AT XX:XX:XX
IDC01304I SUCCESSFUL DEFINITION OF FILE1
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

SNAP TARGETVOLUMES(VSE333) DDSR NOPROMPT
IDC32204I RACROUTE RESOURCE NOT PROTECTED OR BATCH SECURITY=OFF
IDC0935I IXFP/SNAPSHOT FUNCTION COMPLETED SUCCESSFULLY
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

EXPORT NOCOPY.UCAT DISCONNECT
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

IDCAMS SNAP using the Basic Security Manager

z/VSE administrator enabled to control the usage of the IDCAMS SNAP command.

SAMPLE (z/VSE console):

r rdr,pausebg

0 exec bstadmin

BG 0000 1S54I PHASE BSTADMIN IS TO BE FETCHED FROM IJSYSRS.SYSLIB

BG-0000 BST901A ENTER COMMAND OR END

everybody is allowed to use SNAP…COPY command

0 add facility vsam.snap.**copy** uacc(read)

BG 0000 BST904I RETURN CODE OF ADD IS 00

BG-0000 BST901A ENTER COMMAND OR END

everybody is allowed to use SNAP…DDSR command

0 add facility vsam.snap.**ddsr** uacc(read)

BG 0000 BST904I RETURN CODE OF ADD IS 00

BG-0000 BST901A ENTER COMMAND OR END

nobody is allowed to use SNAP…NOCOPY command

0 add facility vsam.snap.**nocopy** uacc(none)

BG 0000 BST904I RETURN CODE OF ADD IS 00

BG-0000 BST901A ENTER COMMAND OR END

0 end

BG-0000

0

IDCAMS SNAP using the Basic Security Manager

Administering the usage of the IDCAMS SNAP command can be done with the following

Basic Security Manager Facilities:

VSAM.SNAP.COPY	for IDCAMS SNAP COPY
VSAM.SNAP.NOCOPY	for IDCAMS SNAP NOCOPY
VSAM.SNAP.DDSR	for IDCAMS SNAP DDSR

If no BATCH security is enabled in the zVSE system (SYS SEC=NO) or it is enabled but no VSAMSNAP.[COPY|NOCOPY|DDSR] RACROUTE facility was defined using BSTADMIN, then the IDCAMS SNAP [COPY|NOCOPY|DDSR] statements are executed as requested but with a warning:

```
IDC32204I RACROUTE RESOURCE NOT PROTECTED OR BATCH SECURITY=OFF
```

If BATCH security is enabled and the BST facility is defined then IDCAMS SNAP function is executed, accompanied by the following message:

```
IDC32200I RACROUTE (AUTH) SUCCESSFUL
```

In all the other cases the requested IDCAMS SNAP function is suspended.

Backup/Restore Enhancements

- Producing cross-reference listings using the RESTORE command
- Producing cross-reference listings without objects being restored
- Identification of compressed files in cross-reference listings
- Identification of empty files in cross-reference listings

Backup/Restore Enhancements
Invocation and Description

VSE/VSAM Backup/Restore provides a new capability to produce cross-reference listings of objects backed up and their place on the tape or disk volumes as a result of the RESTORE command processing.

For a backup file on *tape*, the following two listings are produced:

- *Volume cross-reference listing*
- *Object cross-reference listing*

For a backup on *disk*, the following three listings are produced:

- *Extent cross-reference listing*
- *Object cross-reference listing*
- *Extent list*

Note: Thus, the same set of cross-reference listings are produced by both the BACKUP command and the RESTORE command.

Backup/Restore Enhancements
Invocation and Description

NOXREF|XREF|XREFONLY

Specify whether the cross-reference listings are to be produced.

- **NOXREF specifies that the cross-reference listings will not be produced but objects restoration will be performed.**

Abbreviations: NXREF

- **XREF specifies that both the cross-reference listings will be produced and objects restoration will be performed.**

Abbreviations: None

- **XREFONLY specifies that only the cross-reference listings will be only produced and thus objects restoration will not be performed.**

Abbreviations: XREFY

- **Default: NOXREF**

Backup/Restore Enhancements Sample

XREFONLY

RESTORE

```
RESTORE OBJECTS(*) XREFY
IDC01301I RESTORE'S BACKUP FILE CREATED ON XX/XX/2008 AT 13:38:25
IDCAMS SYSTEM SERVICES                                  TIME: 13:38:38         XX/XX/2008   PAGE   2


BACKUP EXTENT CROSS-REFERENCE LISTING (BECR)
        EXTSEQ    VOLSER    OBJECT NAME                                  OBJECT TYPE   SEGMENT TYPE
        001       WRK002    VSMCKD.KSDS.KEY8.A.C001 ....................  KSDS    CMP   ONLY
                            VSMCKD.KSDS.KEY8.A.C002 ....................  KSDS          ONLY
                            VSMCKD.KSDS.KEY8.A.C003 ....................  KSDS          ONLY
                            VSMCKD.KSDS.KEY8.A.C004 ....................  KSDS          ONLY
                            VSMCKD.KSDS.KEY8.A.C005 ....................  KSDS    CMP   EMPTY
                            VSMCKD.KSDS.KEY8.A.C006 ....................  KSDS    CMP   ONLY
                            VSMCKD.KSDS.KEY8.A.C007 ....................  KSDS    CMP   ONLY
                            VSMCKD.KSDS.KEY8.A.C008 ....................  KSDS    CMP   ONLY

IDCAMS  SYSTEM SERVICES                                 TIME: 13:23:45         XX/XX/2008   PAGE   3
```

VSAM Meaningful Cluster names

If not specified explicitly by the user, meaningful cluster names are now generated by VSAM for AIX/Cluster data and index component.

The generated data component and index component names will use:

- **the specified clustername + .DATA or .D for the data component**
- **the specified clustername + .INDEX or .I for the index component**

Old VSAM generated names

Up to now, VSAM generated the 44-characters name of the data and index components using the following data:

- **the current value of bits 0-55 of the time-of-date (TOD) clock at the moment of the name creation,**
- **the year and the day of creation,**
- **inserting in the name some constants and the period signs.**

Example of the OLD NAME Format:

part 1	part 2	part 3	part 4	part 5
T99EFB7B.	VSAMDSET.	DFD08086.	TC05B8EF.	T99EFB7B

The TOD clock value (bits 0-63) stored by STCK instruction: C05B8EF9 9EFB7B40

New VSAM Meaningful Clusternames

- **If the last qualifier of the name is CLUSTER** : **CLUS.TESTNAME.CLUSTER**
  Generated data name = CLUS.TESTNAME**.DATA**
  Generated index name = CLUS. TESTNAME**.INDEX**

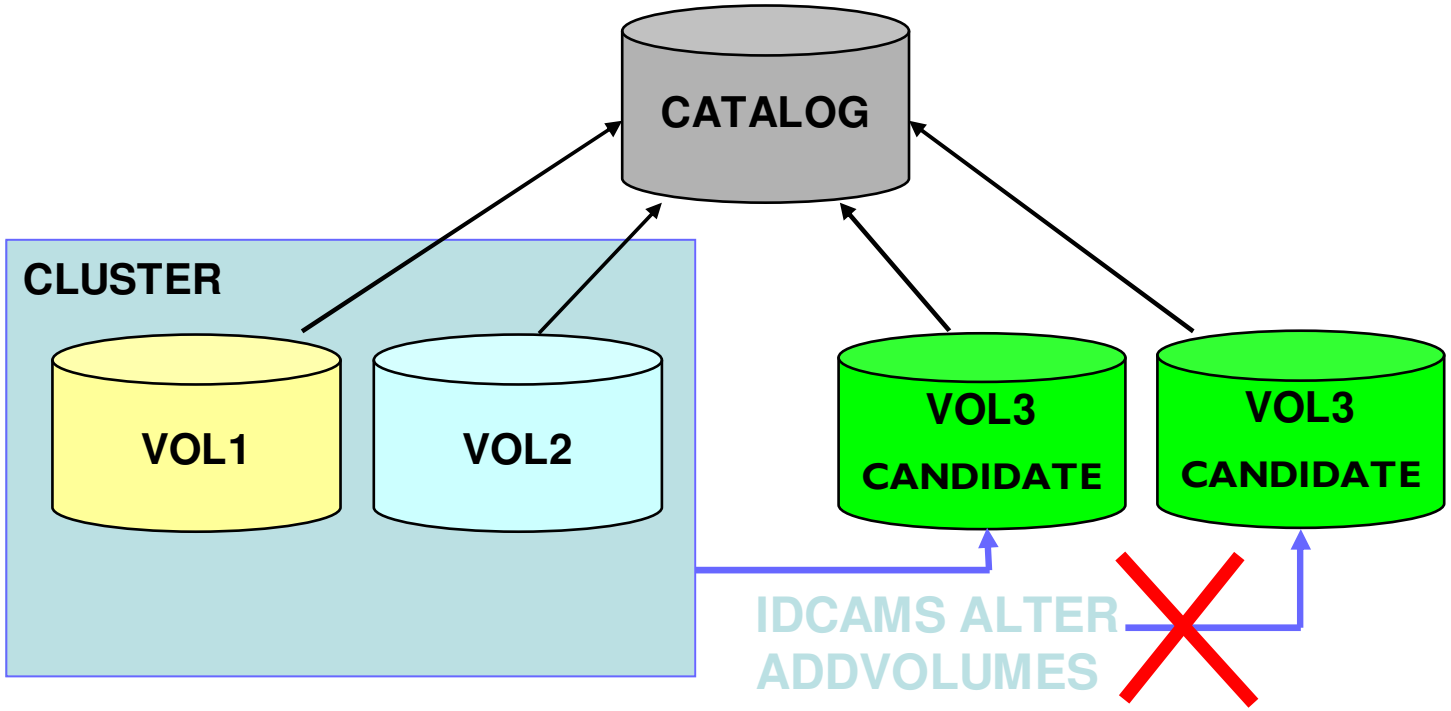
- **If the cluster name <= 38 characters** : **DEPTABC.TEST.INFO**
  Generated data name = DEPTABC.TEST.INFO**.DATA**
  Generated index name = DEPTABC.TEST.INFO**.INDEX**

- **If the cluster name is between 39 and 42 characters inclusive** :
  DEPTABCD.RESOURCE. ABCDEFGH.DATA1234.STUFF
  Generated data name =
  DEPTABCD.RESOURCE.ABCDEFGH.DATA1234.STUFF**.D**
  Generated index name = DEPTABCD.RESOURCE.
  ABCDEFGH.DATA1234.STUFF**.I**

- **If longer than 42 characters, and the last qualifier is not CLUSTER** :
  COMPANY.DEVISION.DEPT.DLREPORT.DECADE.MONTH
  Generated data name = COMPANY.DEVISION.DEPT.DLREPORT**.D99EFB7B**
  Generated index name = COMPANY.DEVISION.DEPT.DLREPORT**.I1A12FAE**

Preventing Duplicate Candidate Volumes

The IDCAMS ALTER command will allow to add any volume as candidate only once.



New error message

If the candidate volume is already present in the list of object candidate volumes, the request will be rejected with IDCAMS Return Code 60 and new Reason code 40

IDC3009I ** VSAM CATALOG RETURN CODE IS *nnn* **– REASON CODE IS IGG0CL***xx* **–** *mmm*

Return code	Reason code	Explanation
60	40	**Explanation:** An attempt was made to add a volume to the object which already has this volume as candidate. Request rejected.

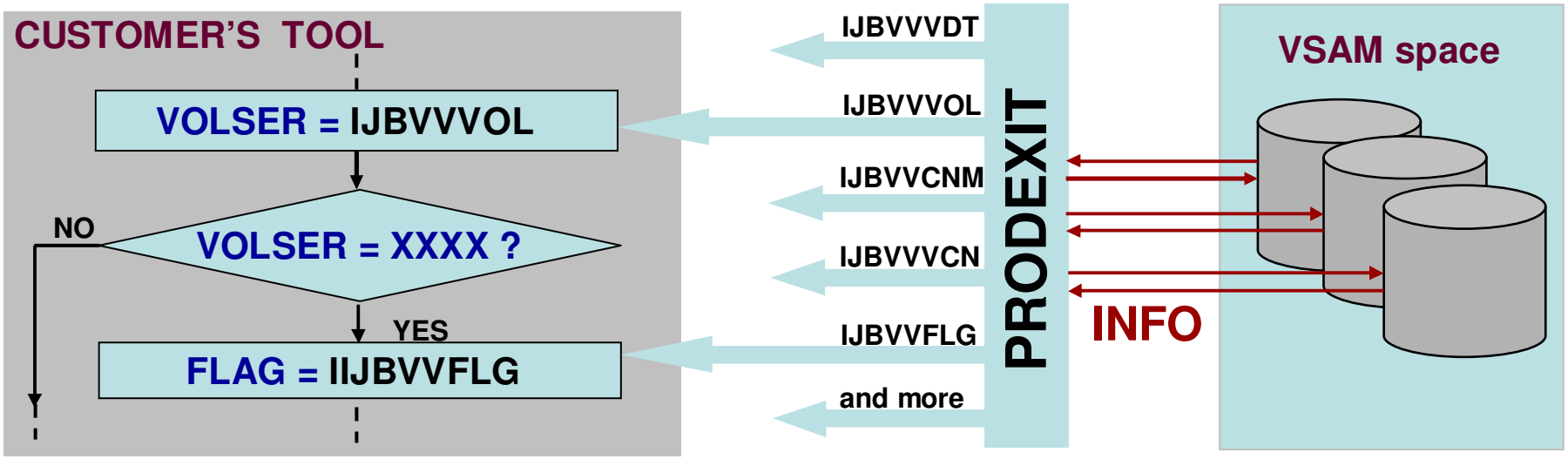
See “z/VSE Messages and Codes Volume 2”.

New VSAM Extent PRODEXIT

- New IJBVVEXU VSAM PRODEXIT allows monitoring of allocations of VSAM data space extents and the suballocation of VSAM cluster extents.

Advantages:

- The PRODEXIT provides to the customers facilities to create his own tools to monitor VSAM events and optimize DASD space usage.



New VSAM Extent PRODEXIT

Notes:

- **Any return code setting in IJBVRC is ignored**
- **At catalog creation, the catalog components which appeared in the LISTCAT as VSAM.CATALOG.BASE.INDEX and VSAM.CATALOG.BASE.DATA are reported by the exit as a single extent named VSAM.CATALOG.BASE**
- **When a catalog is deleted, the exit reports that as a single event. Since the catalog deletion can cause deletion of several data space extents on several volumes, the fields IJBVVVOL, IJBVVVCT, IJBVVVTN, IJBVVVTN, IJBVVVBN are set to binary zeroes, the fields IJBVVEXB and IJBVVEXS are set to -1.**

Task ID for VSAM Lock requests

- For VSAM X'A8 Lock requests the task id of the owner of the lock will be returned in case the lock cannot be acquired because the resource is locked already by another task.

Advantages:

- This information will help to find the reason for locked tasks without the necessity to use the LOCKTRACE on all VSAM locking activities.

Examples:

Trying to open a file, which is already in use within 1 VSE system:

```
Y2 0047 4228I FILE OPEN ERROR X'A8'(168) CAT=IJSYSCT
(OPNH1-45) FILE ALREADY OPEN IN ANOTHER PARTITION, RC X'04' TASK X'0020'
```

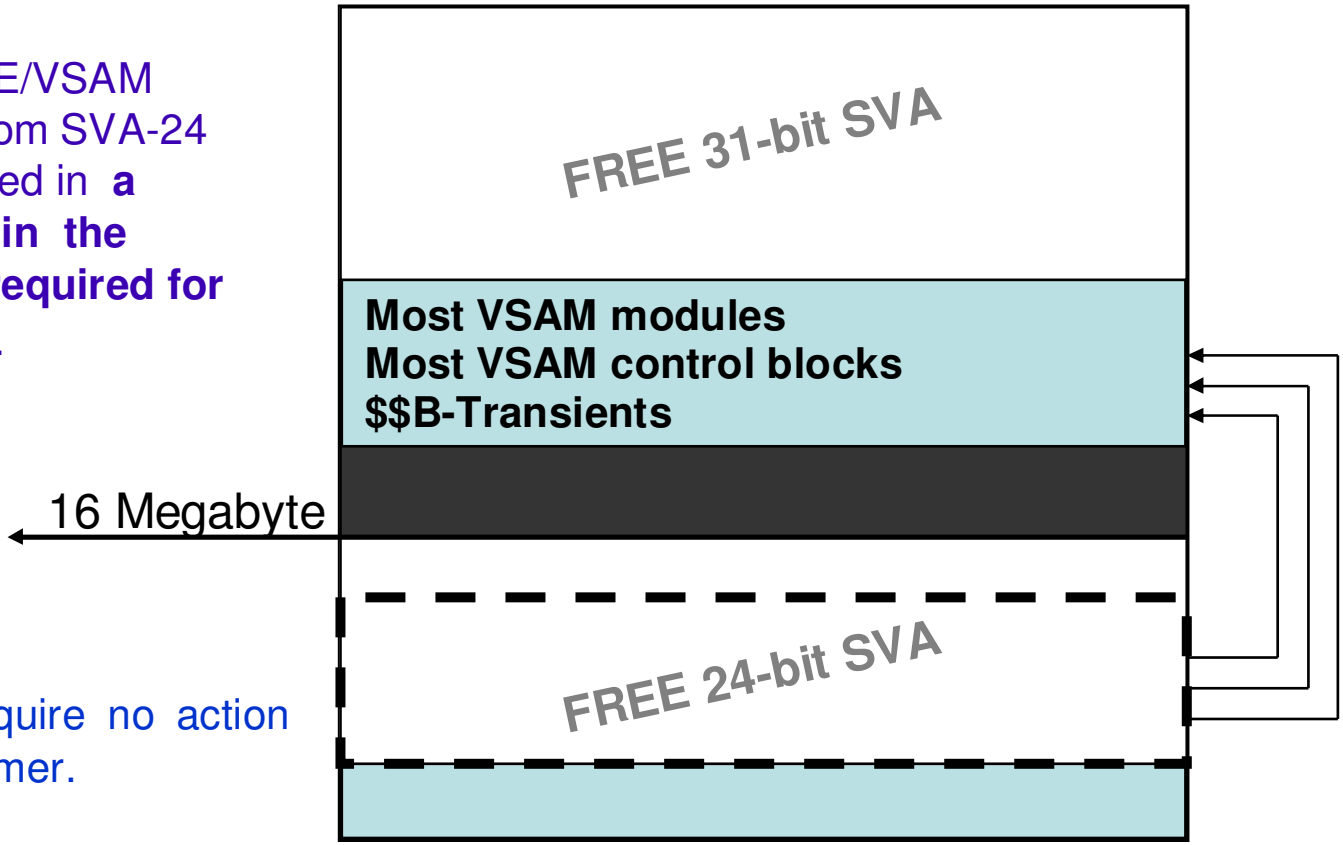
Trying to open a file, which is locked on a different VSE system (shared system):

```
Y1 0045 4228I FILE OPEN ERROR X'A8'(168) CAT=
(OPNH1-45) FILE ALREADY OPEN IN ANOTHER PARTITION, RC X'04' TASK X'FFFF'
```

VSAM 24-bit Constraint Relief

Customers with growing (CICS) workload and/or those who want to consolidate their VSE systems might have an increased need of 24-bit storage.

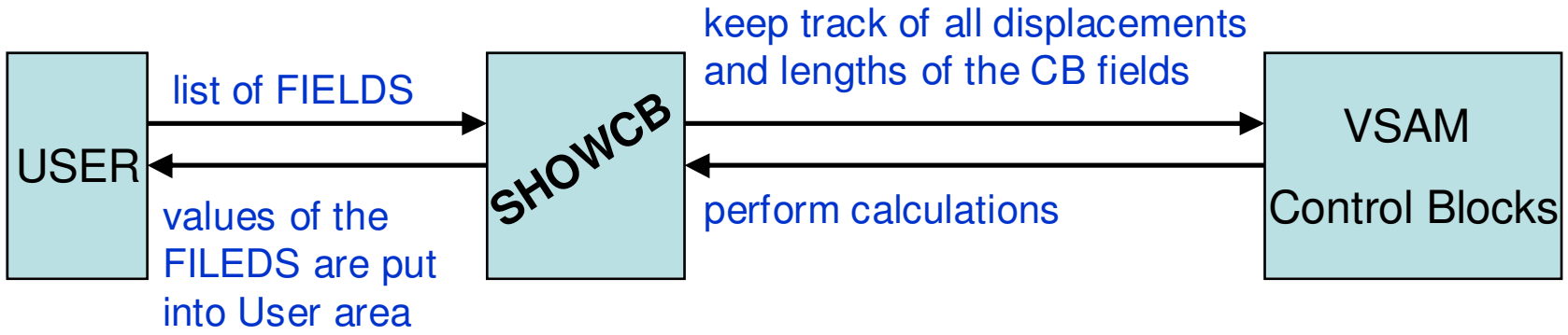
In z/VSE 4.3, most VSE/VSAM phases were moved from SVA-24 to SVA - 31. This resulted in **a significant reduction in the amount of SVA - 24 required for z/VSE system phases**.



The above benefits require no action on the part of the customer.

SHOWCB Enhancements

With SHOWCB macro you can examine the contents of fields in an ACB, EXLST, or RPL control block. VSE/VSAM displays the requested fields in a user's area.



The first part of the enhancement extends SHOWCB to enable monitoring of VSAM LSR datasets using an officially supported API.

The following two FIELDS are supported as SHOWCB ACB:

SHOWCB FIELD	Actual FIELD	FIELD Description
ASTRNUM	RPHDASTR	Number of active strings in pool
STRTOT	RPHDSTNO	Total number of strings in the pool

SHOWCB Enhancements

AMDSB and EDB API are extended in order to enable user to obtain status information for open VSAM datasets.

18 NEW AMDSB and EDB FIELDS are supported by SHOWCB ACB.

SHOWCB FIELD	Actual AMDSB	FIELD Description
ATRB	AMDATTR, AMDATTR3, AMDRCFRM	Dataset Attributes and SAM ESDS record info
SHAREOP	AMDSHOPT	SHARE OPTIONS
LNCIS	AMDLNCIS	Local number of CI SPLITS
LNUPDR	AMDLUPR	Local number of updated records
LNLOGR	AMDLNLR	Local number of logical records
LAVSPAC	AMDLASPA	Local number of bytes of free space

See the full list of a new SHOWCB FIELDS in Chapter 12 “Descriptions of VSE/VSAM Macros” in “VSE/VSAM User’s Guide and Application Programming”.

VSAM SNAP trace enhancements

Enable the following SNAP Traces:

Type:	**Enables**:
0001	Catalog management error code trace
0002	Buffer manager trace
0003	OPEN control block dump (when OPEN processing is complete) OPEN error trace (prints control blocks if an error occurs during OPEN processing) CLOSE control block dump (at the beginning of CLOSE processing)
0004	VSE/VSAM I/O trace
0005	I/O error trace
0008	Catalog management I/O trace (prints all I/O operations done by VSE/VSAM catalog management)
0009	Record management error trace (prints control blocks for any error detected by VSE/VSAM record management)
0010	Redirector Trace
0013	In-core wrap trace for trace points within VSE/VSAM Record Management
0014	Level2 SNAP013 Trace (I/O, EXCPAD and zVSE Lock Activity)
0015	Level3 SNAP013 Trace (Buffer Management)
0016	Produce a printout (PDUMP) each time the SNAP013 Trace Table wraps.

VSAM using DLBL CYL/BLK

The DLBL statement supports the following new operands:

CYL=n CYL=(n,n1)

This operand indicates the number of cylinders on a CKD device to be used for space allocation.

BLK=n BLK=(n,n1)

This operand indicates the number of blocks on an FBA device to be used for space allocation.

n specifies the number of blocks used for the primary allocation,
n1 specifies the number used for secondary allocations.
n and n1 can be a decimal number up to 2,147,483,645.

Example of DLBL statements:

```
// DLBL VSMFIL1,'VSMFILE-ID 1',0010,VSAM,CYL=10
// DLBL VSMFIL2,'VSMFILE-ID 2',0020,VSAM,BLK=(20000,10000)
```

CATLG Interface package

- **Existing CATLG interface is partially made an official external interface**
- **The CATLG interface has NOT been changed**
- **Reading from Catalog records but no update or define**
- **CTGPL (Catalog Parameter List)**
- **CTGFL (Catalog Field List Entry)**

Available for free !!

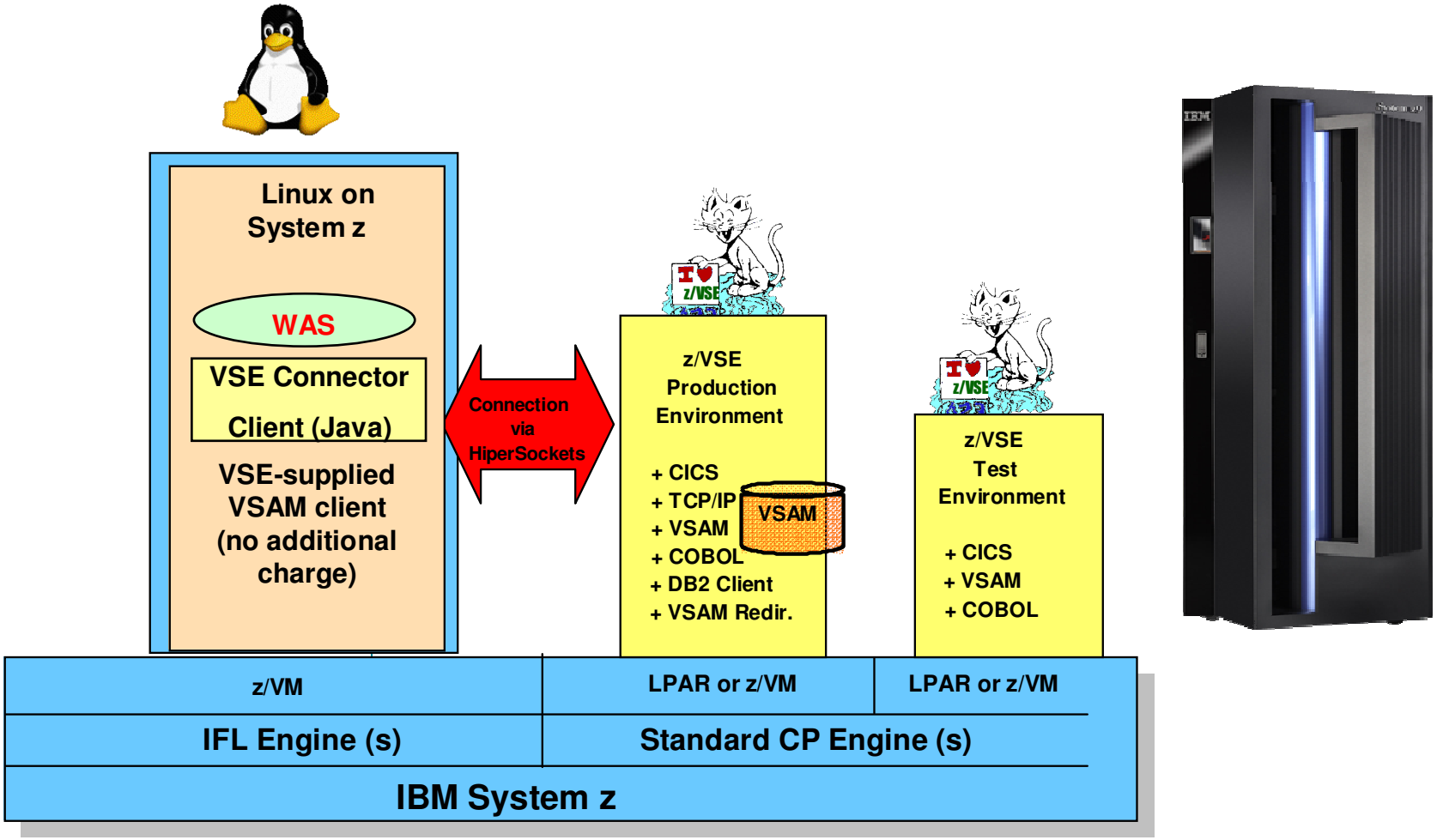
http://www-03.ibm.com/systems/z/os/zvse/downloads/samples.html#vsam

Agenda
z/VSE VSAM new enhancements

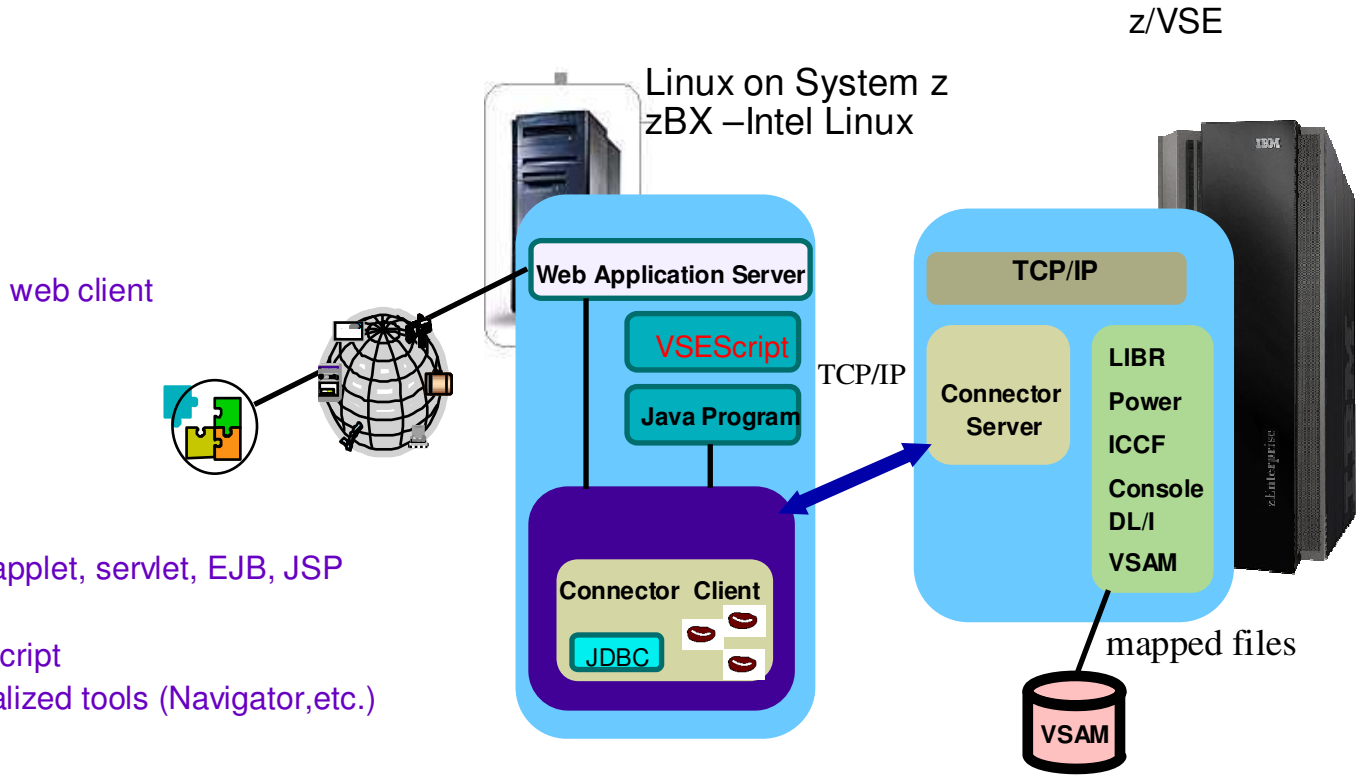
- VSAM Considerations and Limits
- VSAM new Enhancements
- Access VSAM files from remote platforms
- VSAM applications can access relational databases
- Tools for z/VSE VSAM files

Leverage z/VSE data and resources from Java

Leverage VSE/VSAM data using VSAM Connectors on Linux on System z

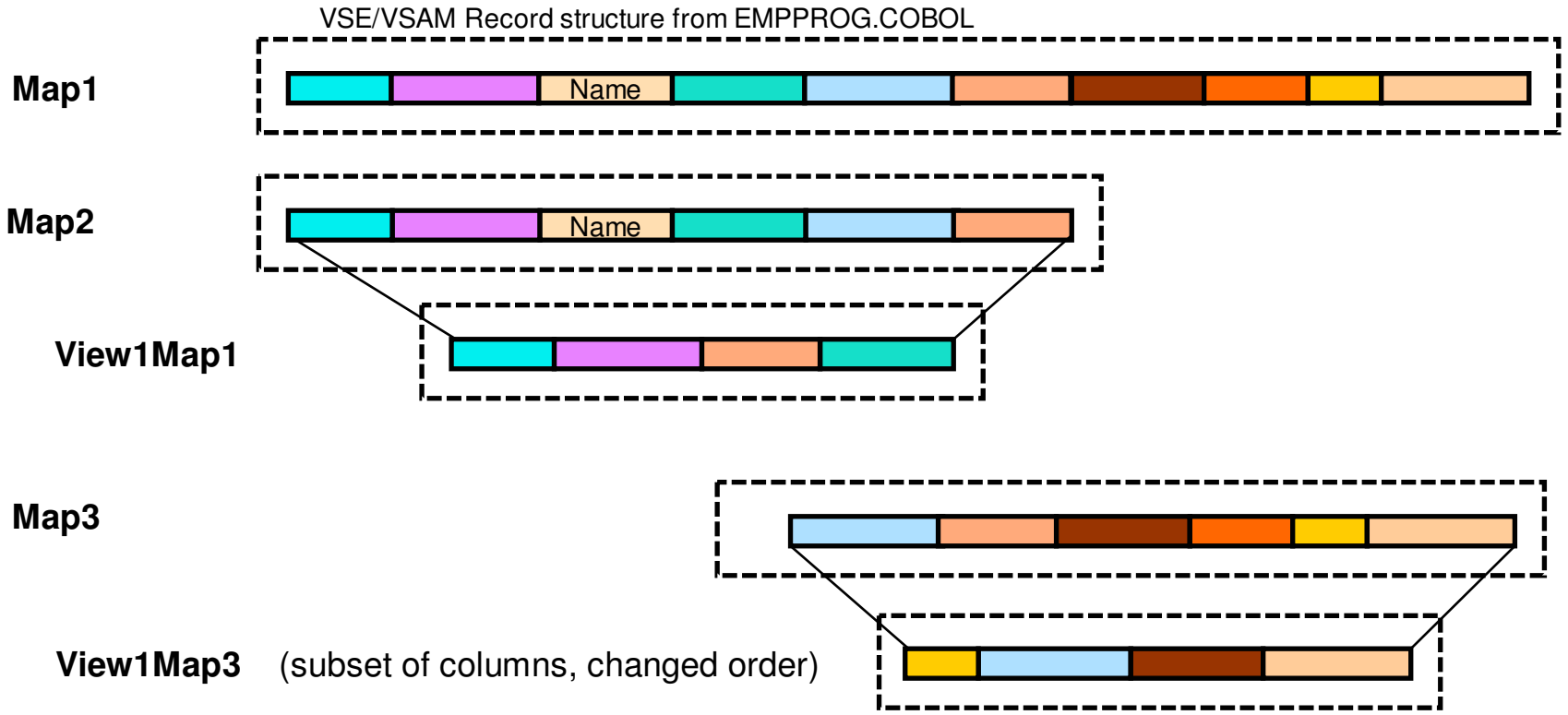


Real time access to VSE resources using the Java–Based Connector (feature included in z/VSE)



- real time access to VSE resources from remote systems
- new possibilities for leveraging the VSE investment

VSAM Record Mapping



Mapping characteristics:

- No changes to VSAM data
- Mapping information stored in a repository in VSAM (VSE.VSAM.MAPPING.DEFS)
- Multiple maps and views (subset of map fields) supported
- Possible data types: STRING, binary, signed number, unsigned number, packed data

The definition of the MAPS can be done via IDCAMS RECMAP or Java MapTool

IDCAMS RECMAP Command Enhancements

New parameter DECIMALPOS for IDCAMS RECMAP command is implemented to specify the position of decimal point for decimal numbers.

This parameter can only be applied to the field types:

PACKED, UNPACKED, ZONED, and UNZONED.

Examples:

123.45	has a decimal position of 2
12345	has a decimal position of 0 (or no decimal position)
1234500	has a decimal position of -2

Such decimal numbers are used by customer applications on COBOL or PL/1 to do calculations with decimal values. For example, for storing currency, there are usually 2 digits after the point for cents, e.g. 123.45 €

IDCAMS RECMAP Command Enhancements

Two new parameters MAPNAMES and CLUSTERNAMES, are implemented for IDCAMS RECMAP LIST command to produce a list of map names.

Example of CLUSTERNAMES:

```
UCAT1
    CLUSTER.NAME.A
                 MAP.NAME.AA
                 MAP.NAME.BB
    CLUSTER.NAME.B
                 MAP.NAME.CC
                 MAP.NAME.DD
```

These lists are to help customers find information about defined maps and clusters quicker.

Accessing VSAM data from remote systems

using VSAM JDBC Driver

- Based on VSE Connector Client
- Translates SQL into VSE/VSAM calls
- Standard JDBC API
- Requires VSAM Record Mapping

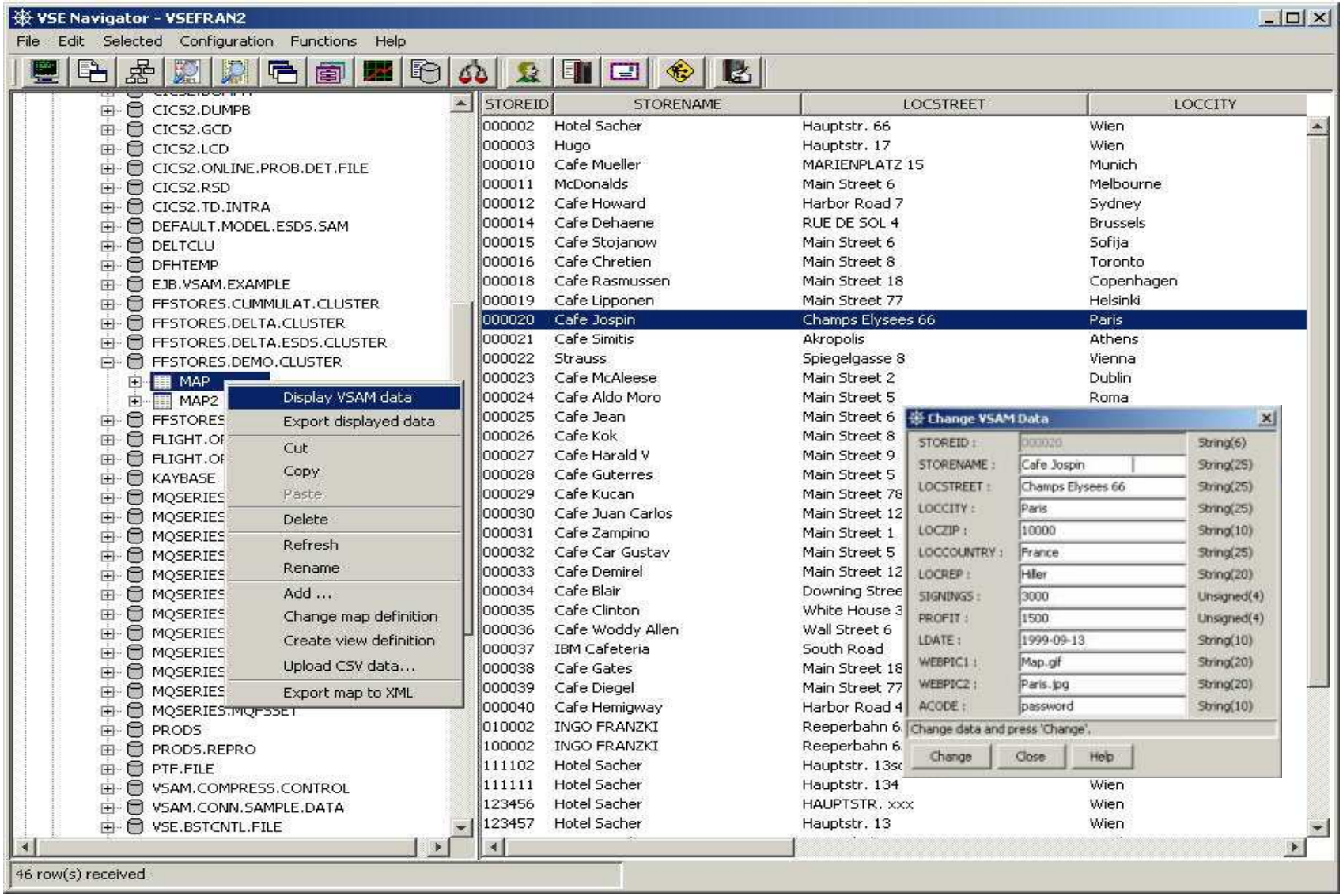
Access VSAM via batch interface - read / (or SHAREOPTION 4 for write)

```
SELECT NAME,STREET,CITY FROM
  MY.USER.CATALOG\MY.VSAM.CLISTER\MY_MAP
  WHERE PERSNR=4711
  ORDER BY NAME
```

Access VSAM via CICS (DBDCCICS) – read/write

```
SELECT NAME,STREET,CITY FROM
  #VSAM.#CICS.DBDCCICS\CLUNAME\MY_MAP
  WHERE PERSNR=4711
  ORDER BY NAME
```

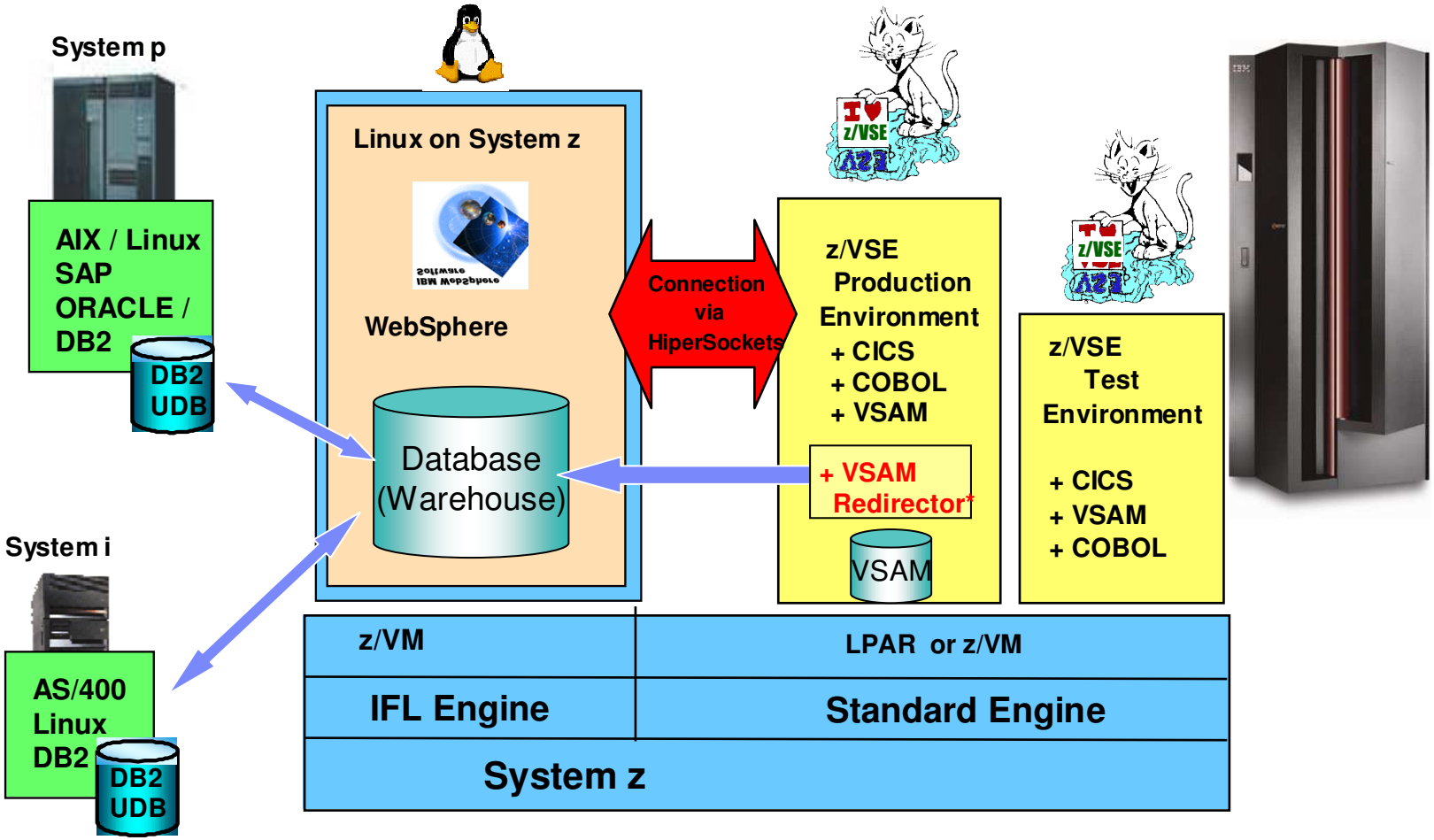
z/VSE Navigator: Graphical z/VSE Interface



Agenda
z/VSE VSAM new enhancements

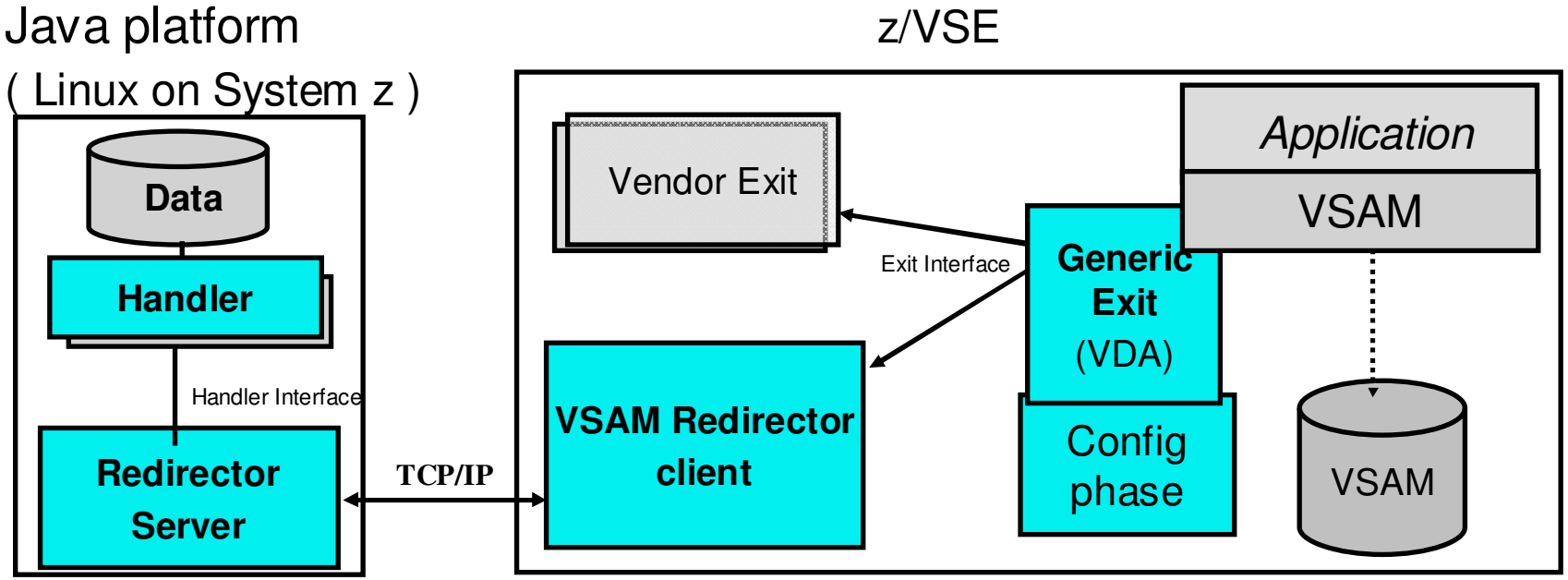
- VSAM Considerations and Limits
- VSAM new Enhancements
- Access VSAM files from remote platforms
- VSAM applications can access relational databases
- Tools for z/VSE VSAM files

VSAM Programs access DB2 on Linux on System z



(*) VSAM Redirector – Common data store solution – with DB2 on Linux on zSeries Solutions without changes to VSAM programs

VSE/VSAM Redirector - functional view



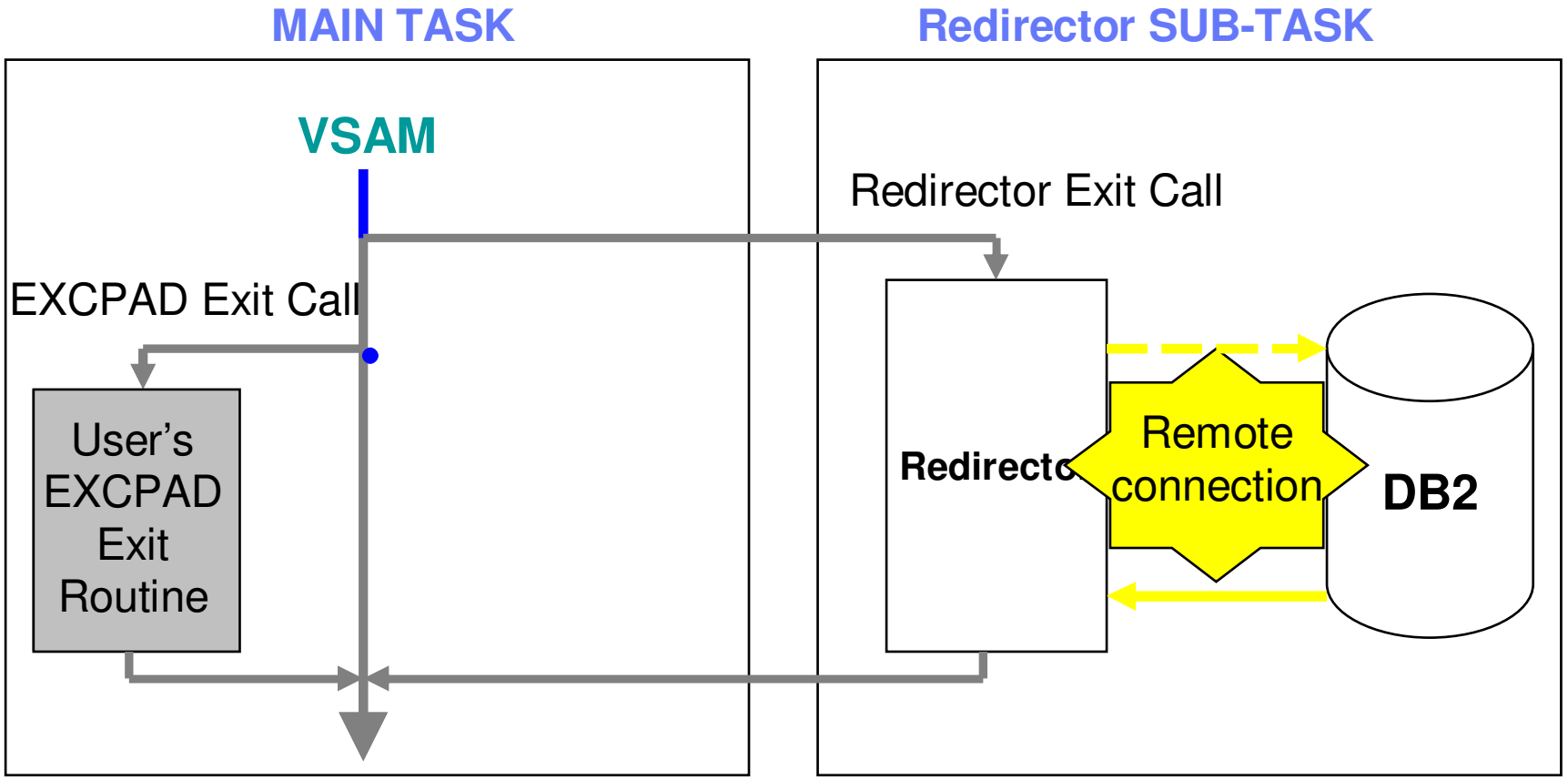
- **Redirector Components:**

- **Generic Exit is based on VSAM Data Access Exit (VDA)**
- **Config phase – contains the redirection properties**
- **Redirector client (SVA phase)**
- **Redirector server – manages the connections (Java component)**
- **Handler – takes care of data processing (Java component)**

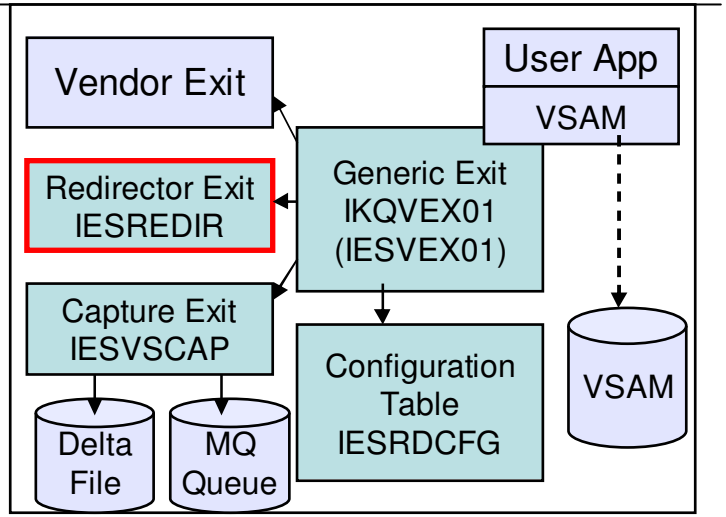
VSAM Redirector EXCPAD

New VSAM Redirector EXCPAD support.

When the EXCPAD exit routine is used, then the Redirector call is performed in a separate subtask. This allows VSAM to continue processing concurrently by returning to the EXCPAD exit routine, while the Redirector task waits for a remote connection.



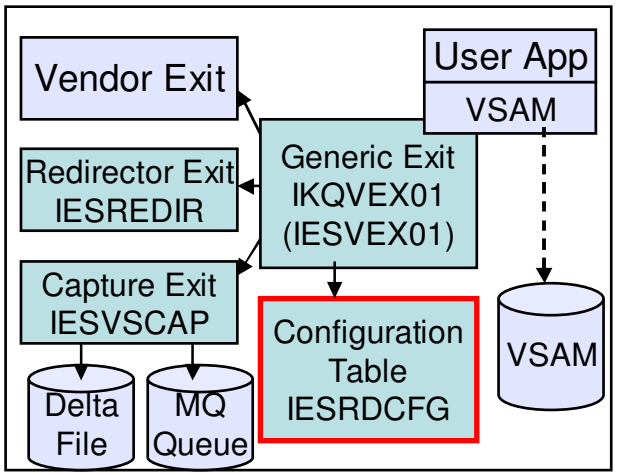
VSAM Redirector – the Client



- **The Client**, Shipped in VSE phase IESREDIR
- Forwards all requests via TCP/IP socket to VSAM Redirector Server on a remote site
- Important Parameter
  - OWNER=REDIRECTOR
    - VSAM requests are executed by remote site only
    - VSAM does not execute the request
      - controlled by return code from IKQVEX01
    - Allows migrating VSAM data to a database without changing the VSAM programs
  - OWNER=VSAM
    - Both, VSAM and remote site executes the request
    - Allows synchronization of VSAM and database without changing the VSAM programs

VSAM Redirector – Configuration Table

- Gets assembled as VSE PHASE
  - Does not contain any executable code, just a table




```
IESRDENT CATALOG='MY.VSAM.CATALOG',
      CLUSTER='MY.VSAM.CLUSTER1',
      EXIT='IESREDIR',
      OWNER=REDIRECTOR,
      IP='1.2.3.4',
      HANDLER='com.ibm.vse.db2handler.DB2Handler,
      OPTIONS='dburl=jdbc:db2:database;dbuser=user;
               maptable=maptable;map=mapname;
               dbpassword=password;dbtable=table,

IESRDENT CATALOG='VSESP.USER.CATALOG',
         CLUSTER='MY.TEST.CLUSTER2',
         EXIT='IESVSCAP',
         MODE=JOURNALING,
         DELTADD='DELTAFI',
         DELTATYPE=KSDS,
         SHARE=ENDREQ,
         ORIGIN='TEST2'
```

The flavors of VSAM redirector

(1) Remote processing - Real time VSAM access to remote

a) **Remote only processing** to DB2 (no VSAM access anymore)

b) **Synchronization** (two phase commit of VSAM and DB2)

(2) Capture - VSE local data processing (in addition to VSAM Req)

a) **Capture Exit** for VSAM processing

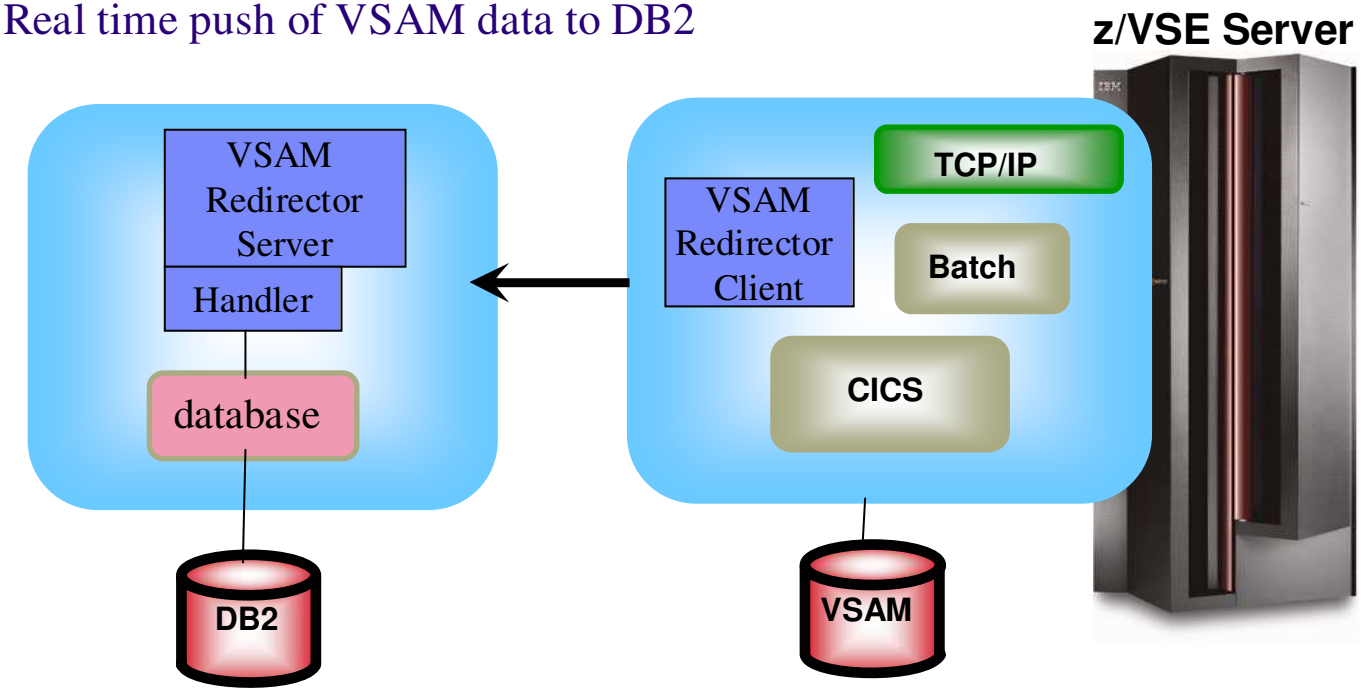
b) **MQ Exit** and MQ Series solutions

VSE/VSAM applications (without any change), access remote relational databases

(1) Real time access VSAM to DB2

a) synchronization (two phase commit of VSAM and DB2)

b) Real time push of VSAM data to DB2

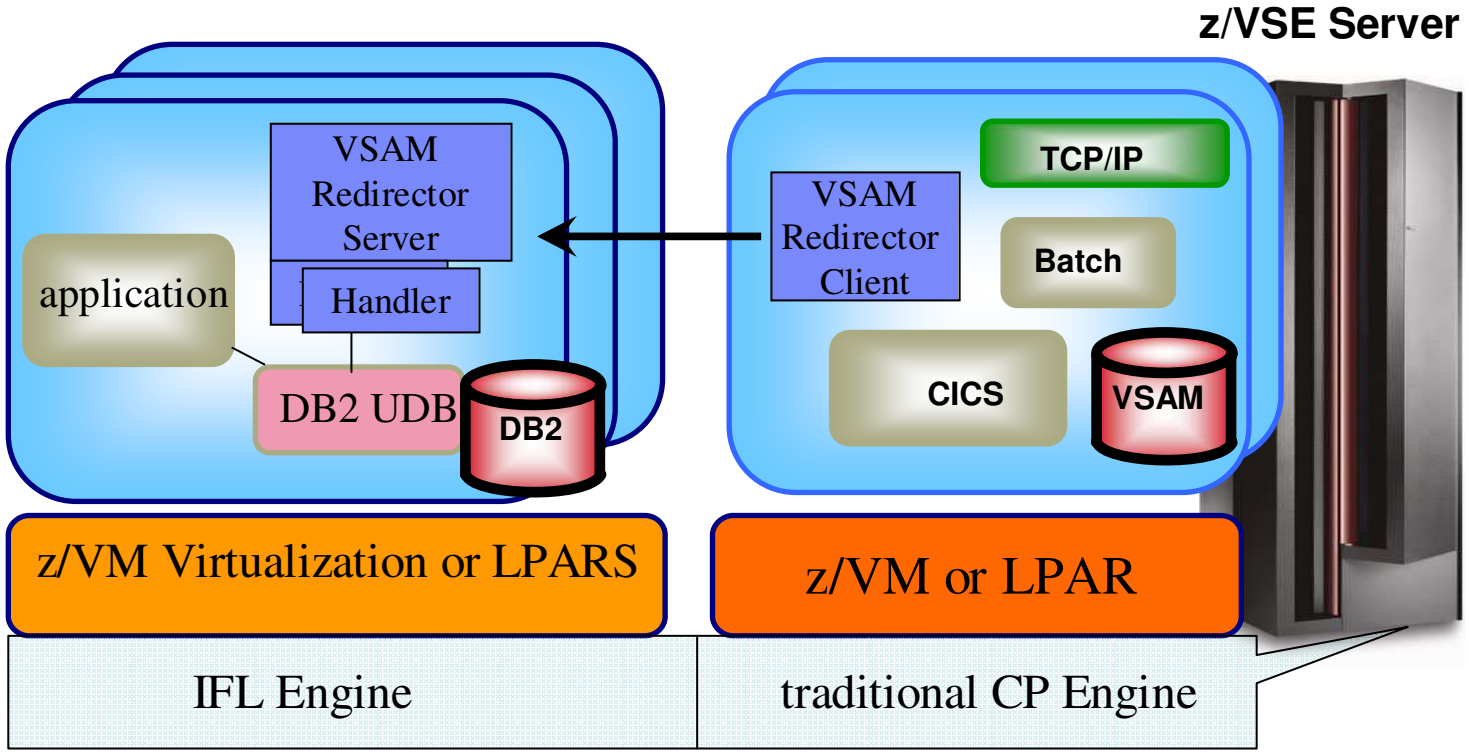


VSE/VSAM applications (without any change), access remote relational databases

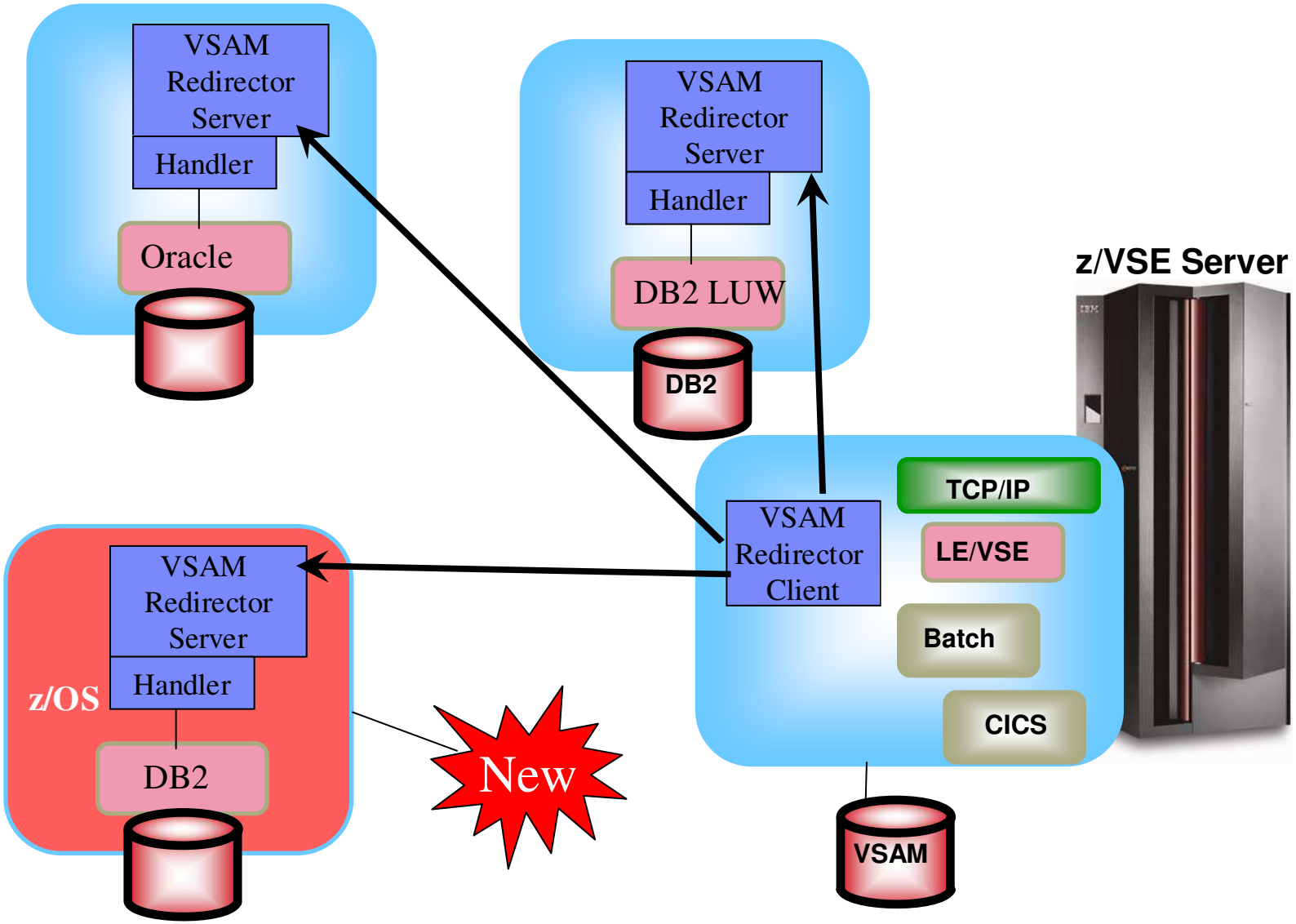
(1) Real time access VSAM to DB2

a) synchronization (two phase commit of VSAM and DB2)

b) Real time push of VSAM data to DB2



VSE/VSAM applications, access remote relational databases

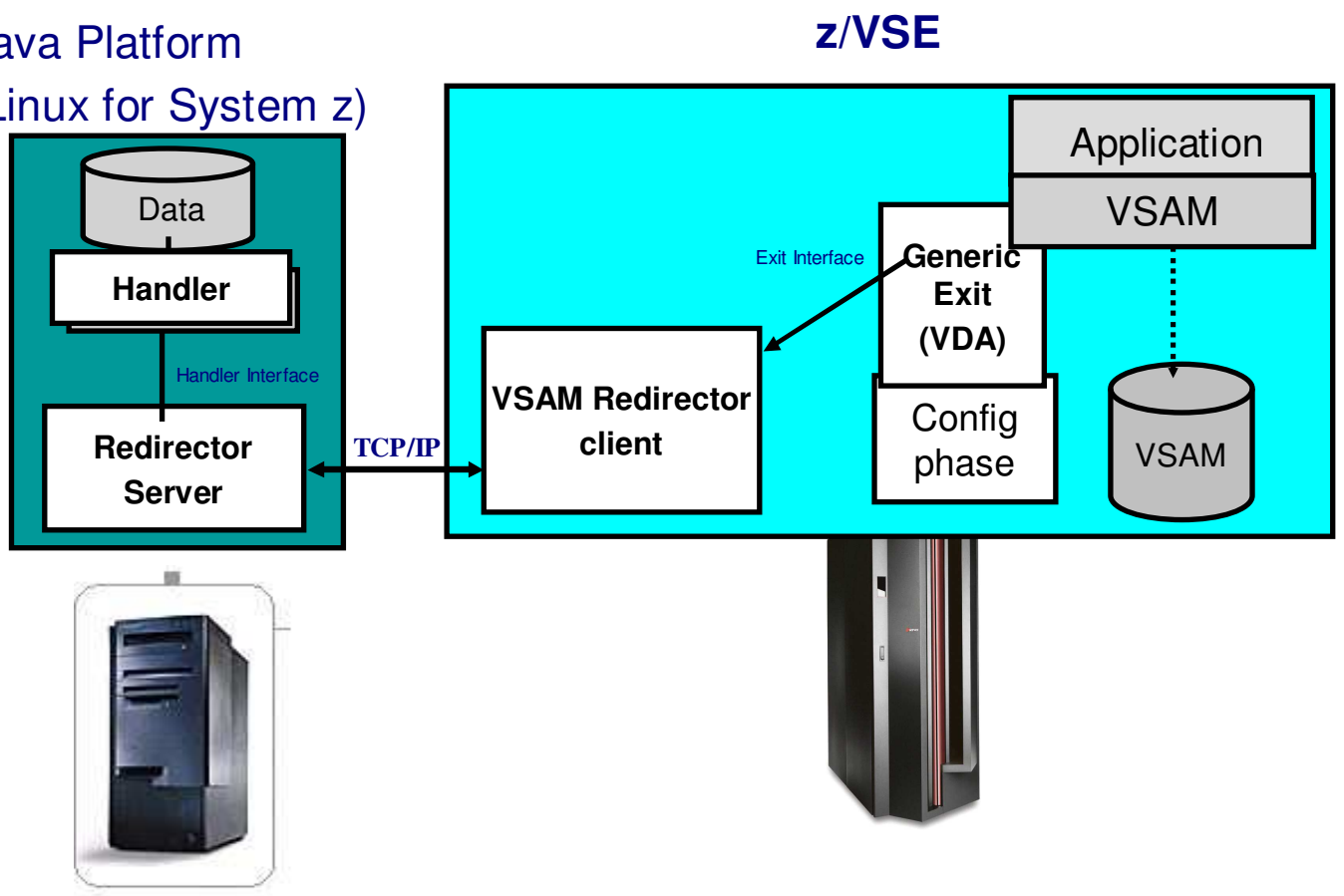


(1) Remote processing

a) Remote only processing – NO VSAM access anymore

- Requests for redirected VSAM files will be handled by VSAM Redirector client and send to the remote system.
- OWNER = REDIRECTOR

- Dataflow for a read/write request:
  - Generic exit is involved
  - VSAM Redirector Client is called
  - Redirector client sends request to Redirector Server
  - Handler processes data
  - Return Code if any is translated to VSAM error
- All reads and writes are done from/to remote

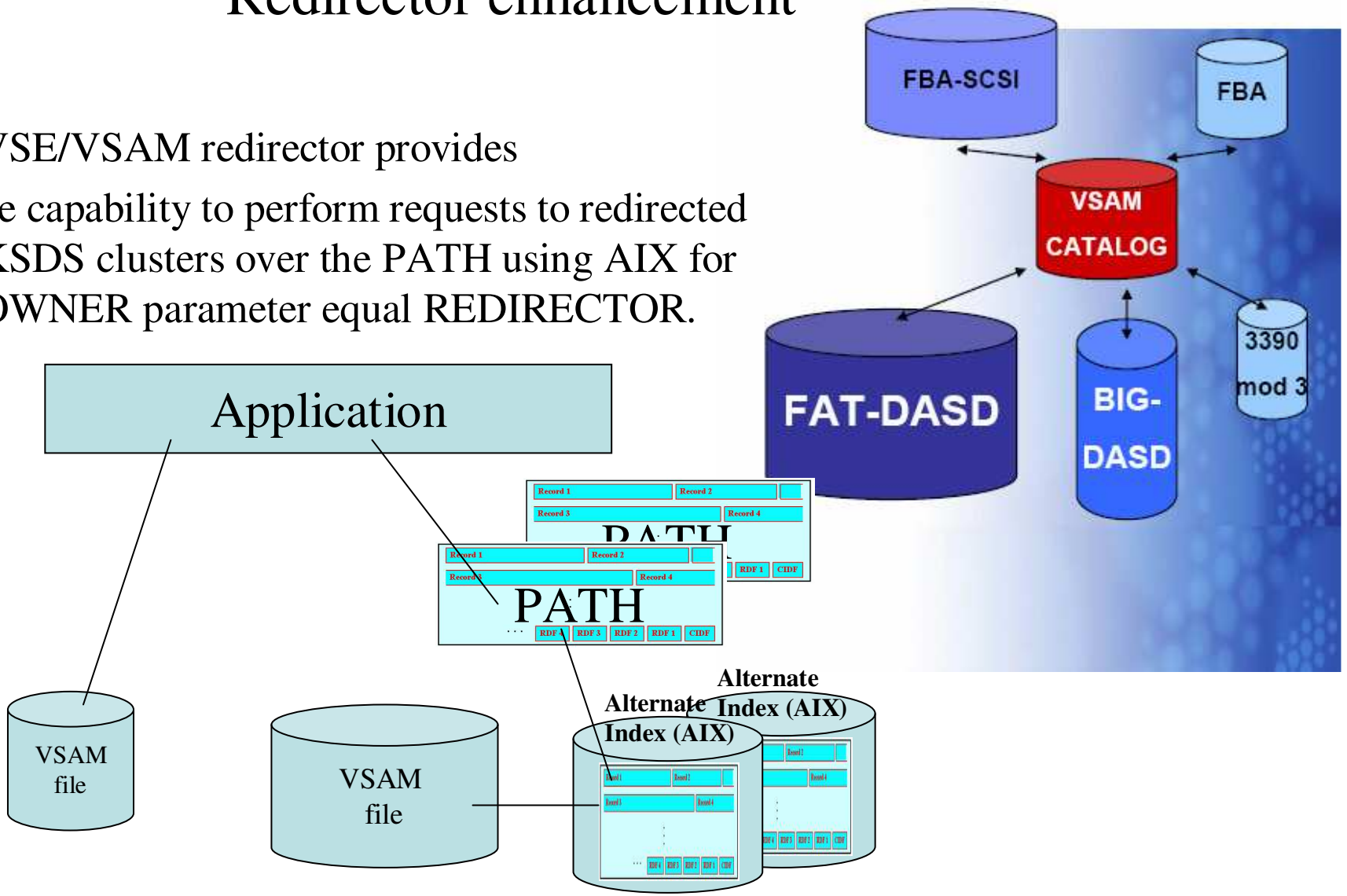


No changes required in applications (CICS, batch).

Redirector enhancement

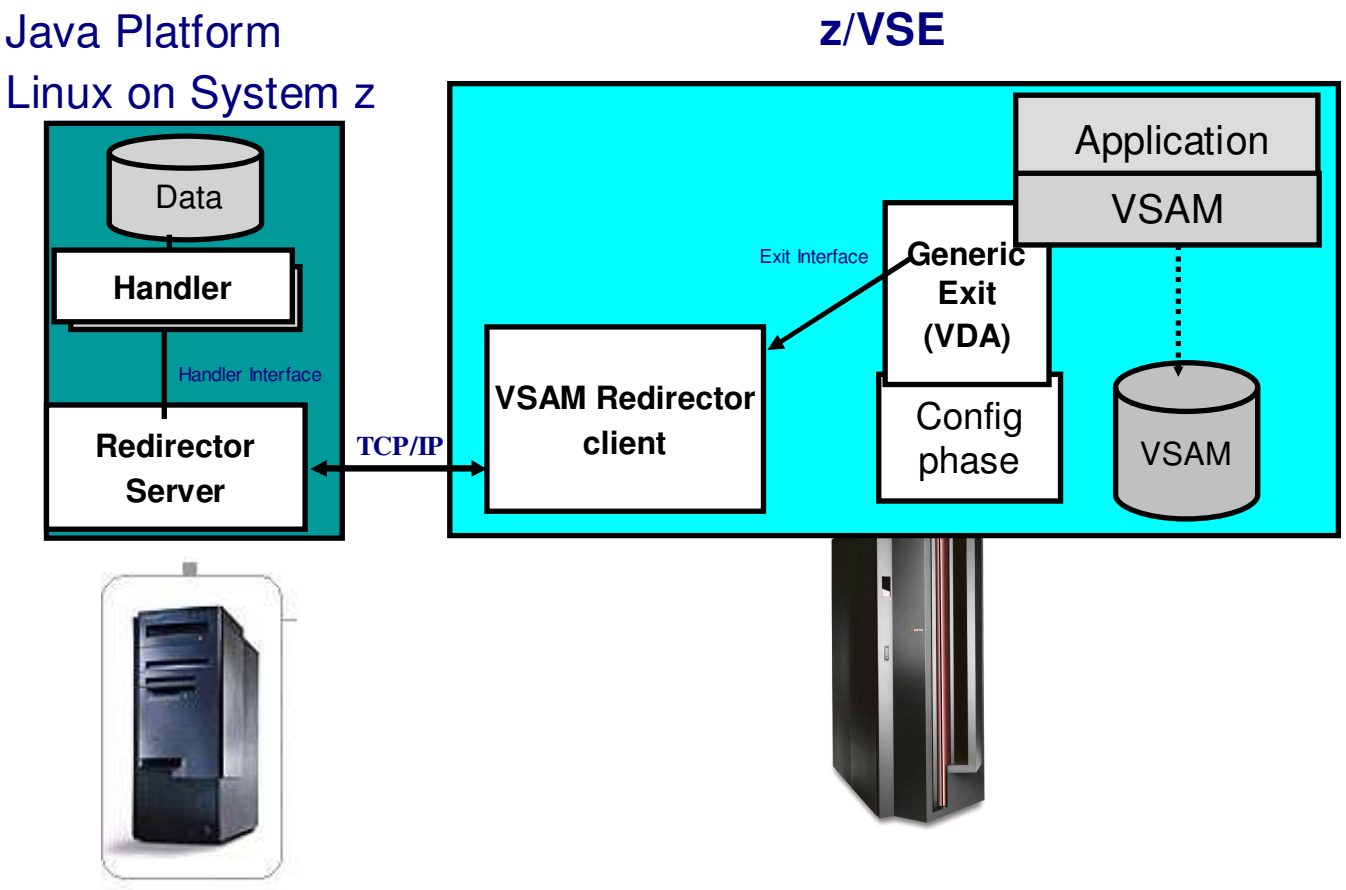
- VSE/VSAM redirector provides

  the capability to perform requests to redirected KSDS clusters over the PATH using AIX for OWNER parameter equal REDIRECTOR.



(1) Remote processing

a) Synchronization of VSAM with a database



No changes required in applications (CICS, batch).

(1) Remote processing

a) Synchronization of VSAM with a database

- Requests for redirected VSAM files will be handled by VSAM Redirector client and send to the remote system.
- OWNER = VSAM

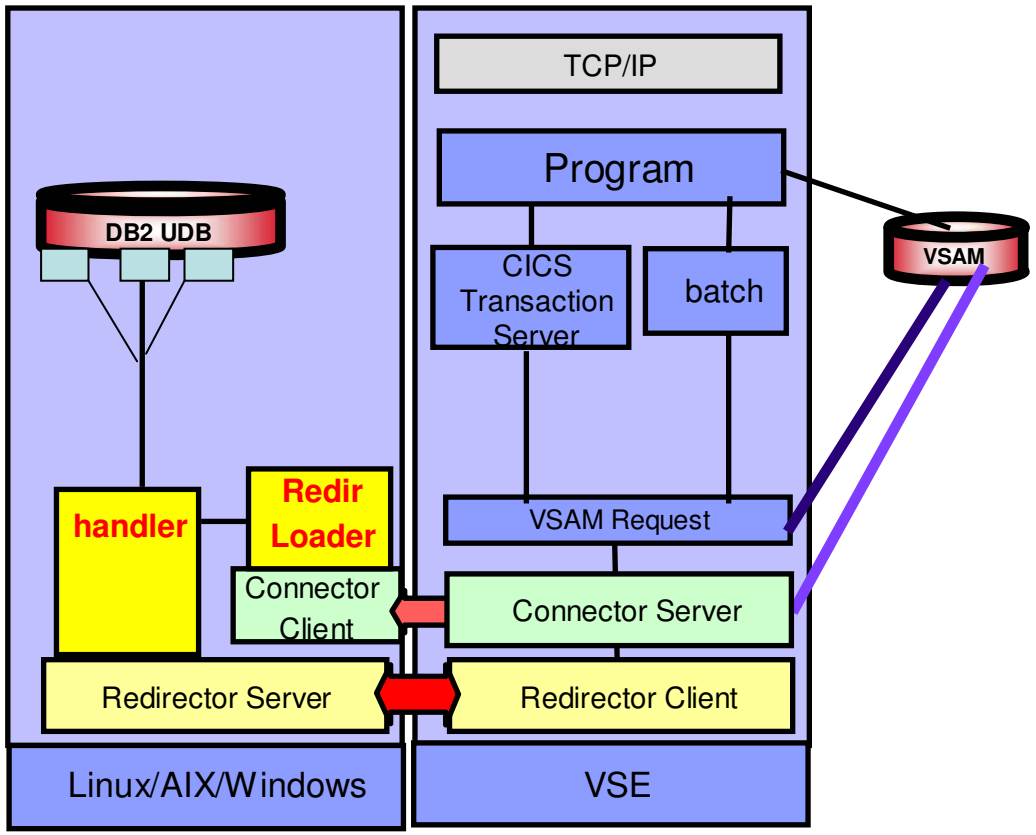
- Dataflow for a read/write request:
  - READs will be performed from VSAM only
  - Generic exit calls VSAM Redirector Client
  - Redirector client sends request to Redirector Server
  - (a) Handler processes the request in the database
  - Return Code is send back to Redirector client
    - In case of an error it will be translated to a VSAM error
  - (b) If return code is favorable the request is processed in VSAM
  - (c-1) If the VSAM request is processed correct – the database will get a COMMIT request through the Redirector Client-Server-Handler
  - (c-2) If the VSAM request ends in error – the database will get a ROLLBACK request

- This is the two phase commit mechanism to keep VSAM and DB2 in sync

Data synchronization – Normalization

VSE/VSAM Redirector can store VSAM data normalized

- No changes to the existing VSE applications
- **The new Redirector Handler in z/VSE 4.1 can store 'VSAM' data in multiple DB Tables .**
- **Redir Loader utility provided for initial data transfer**



- Applications on VSE should be able to access DB2 data on Linux
- Population of DB2 UDB on Linux with VSAM using VSAM Redirector. (VSAM Redirector is part of VSE)
- delivered by IBM

VSAM Redirector – Normalization - Handler

- Two different Handler with the New VSAM Redirector
  - Old: DB2Handler
    - Is still packaged with z/VSE
    - supports data access consolidation (OWNER=REDIRECTOR) as well as data synchronization (OWNER=VSAM)
    - supports DB2, Oracle, MS-SQL, ...
    - was enhanced with new data formats (packed, zoned, date, ...)
  - New: DBHandler
    - enables data Normalization
    - supports data synchronization only (OWNER=VSAM)
      - supports Record-Types
      - supports lists with fixed and variable length
    - supports new data formats (Packed, Zoned, Datum, ...)
      - supports DB2, Oracle, MS-SQL, ...

VSAM Redirector – Normalization - Handler

New Redirector handler in z/VSE 4.1

- Handler to Normalize VSAM data
  - store one VSAM record in multiple tables
    - based on VSAM indicator fields
    - administrator decision
    - for synchronization only (owner = VSAM) - READS are done from VSAM
  - relation between tables to be unique
  - definitions via GUI (mapping configuration)
  - SQL loader provided for database load
    - RedirLoader - fast initial LOAD of a database from VSAM
    - MQLoader - MQ trigger application
    - DeltaLoader – Processing of the Delta file – insert into the database

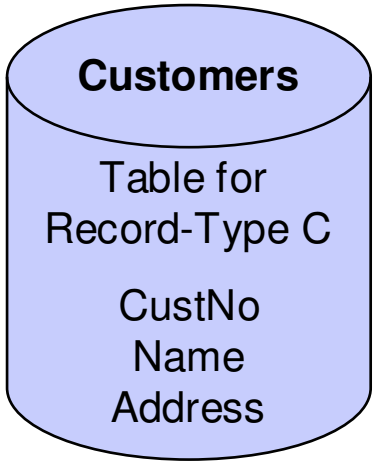
VSAM Redirector - Normalization – data types

- New extendable Concept: Converters
  - One Java-class per data type
  - Open interface
  - New data types can be extended easily
- Data types:
  - STRING
  - BINARY
  - BIT
  - Numbers (INTEGER, PACKED, ZONED, FLOAT, FIXEDTEXT, FLOATTEXT), supports Implied decimal positions
  - DATETIME, TOD
  - HEXCHAR
- Various Options
  - Settings (i.e. date format, number of decimals, ...)
  - Error handling: ONERROR= TERMINATE, TO-NULL, TO_ZERO
  - Text handling: TRIM, PAD, BLANK-TO-NULL, CODEPAGE

VSAM Redirector - Normalization – Record-Types

COBOL Copybook:

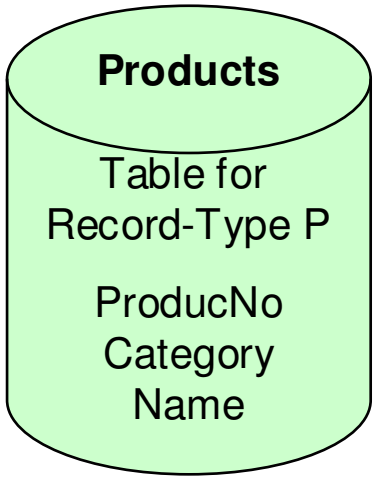
```
01 RECORD-3
  03 RECORD-TYPE           PIC X(1)
  03 RECORD-FORMAT-C
    05 CUSTOMER-NO                 PIC X(7)
    05 CUSTOMER-NAME       PIC X(25)
    05 CUSTOMER-ADDRESS    PIC X(45)
  03 RECORD-FORMAT-P REDEFINES RECORD-FORMAT-C
    05 PRODUCT-NO                  PIC X(7)
    05 PRODUCT-CATEGORY    PIC X(15)
    05 PRODUCT-NAME        PIC X(15).
```



Depending on the value of Record-Type field, the data will be store in different database tables

- Type = C → Customers-Table
- Type = P → Products-Table

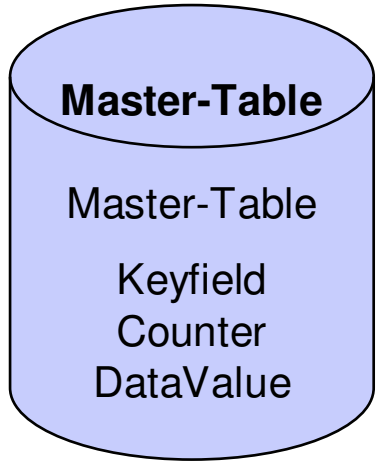
The association takes place at runtime for each individual record.

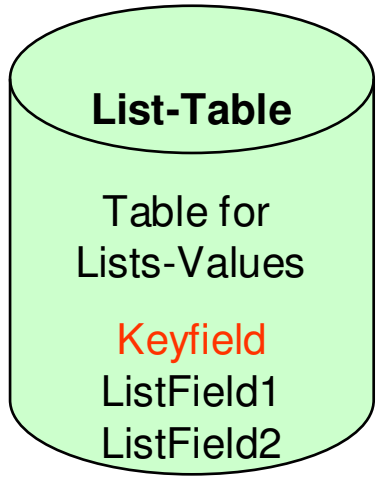


VSAM Redirector - Normalization – Lists

COBOL Copybook:

```
01 RECORD-2.
  03 KEYFIELD            PIC X(8).
  03 COUNTER             PIC 9(5) COMP-3.
  03 VARIABLE-LIST OCCURS 1 TO 5
                          DEPENDING ON COUNTER.
    05 LISTFIELD-1              PIC 9(9).
    05 LISTFIELD-2              PIC X(5).
  03 DATAVALUE                  PIC X(10).
```



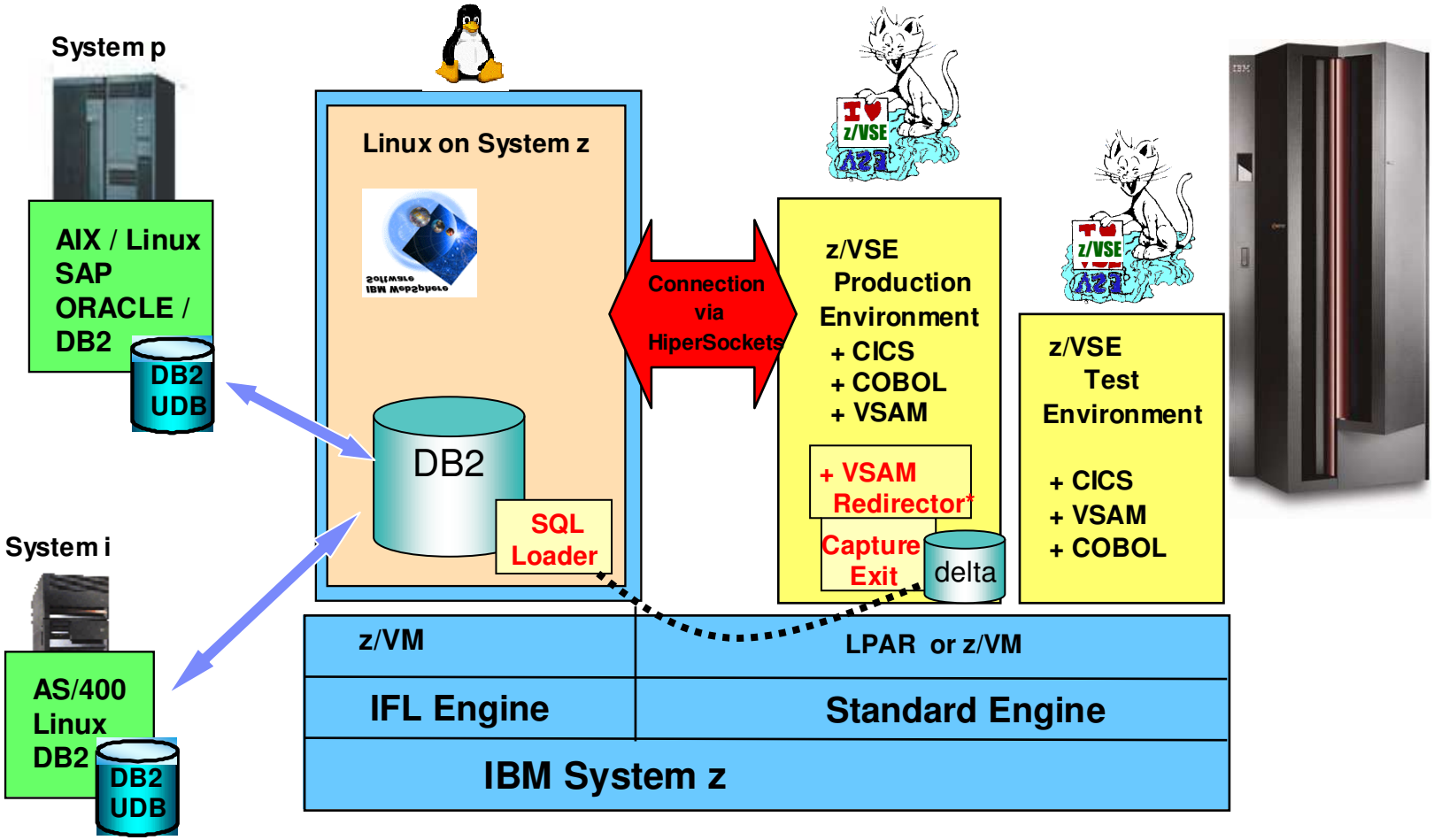


Depending on the value of the field Counter, there will be inserted 1 to 5 rows into the List-Table.

The relation to the Master-Table is defined through the foreign key Keyfield in the List-Table

Normalization takes place at runtime for each individual record.

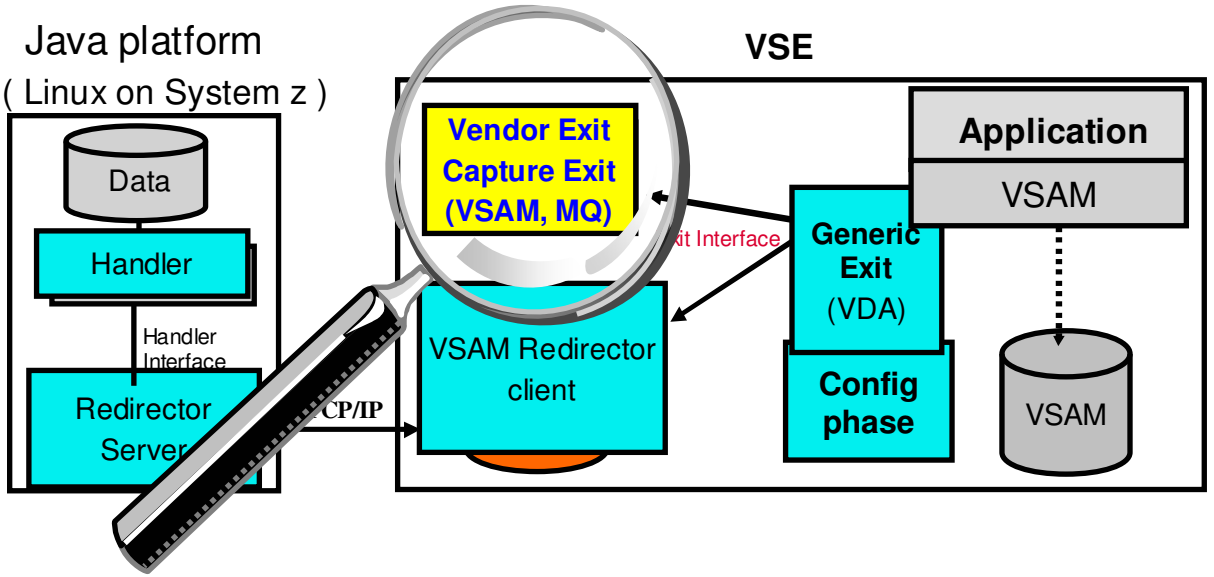
VSAM Capture for DB2 LUW on Linux on System z



(*) VSAM Redirector – Common data store solution – with DB2 on Linux on zSeries Solutions without changes to VSAM programs

VSAM Data collection / transformation / journaling on VSE

Capture Exit

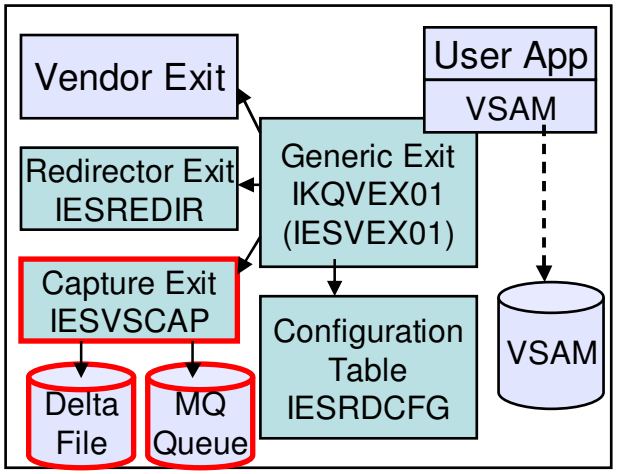


CAPTURE – wit Decision Exit as filter

- **Vendor Exit**
  - **user (vendor) written phase for data collection/transformation**
  - **has to comply with the documented Exit Interface**
- **Capture Exit**
  - **an exit delivered by IBM for capturing changed VSAM data**
  - **an exit delivered by IBM for generating MQ messages**

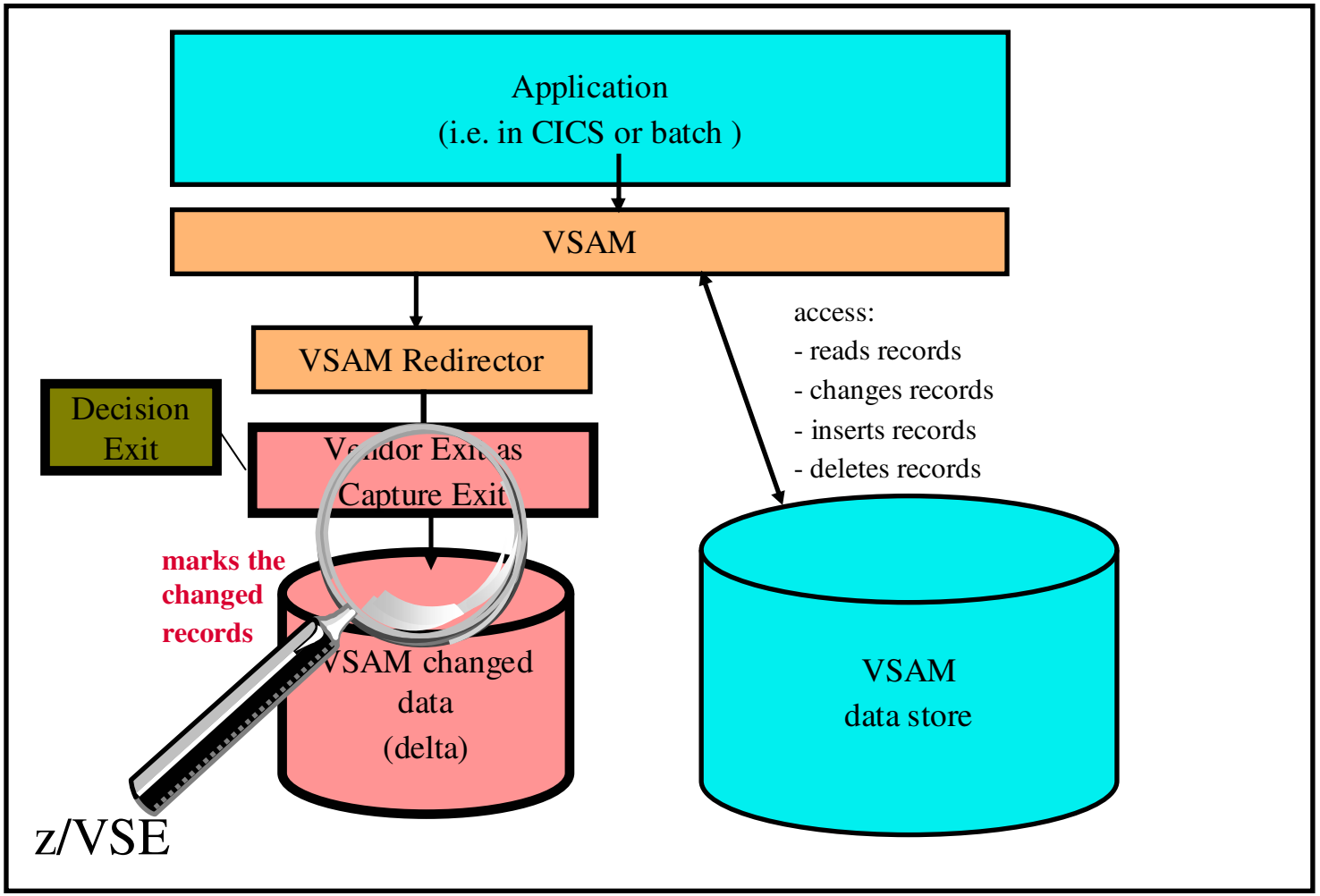
Note: No chaining of Vendor Exit with VSAM Redirector client supported

z/VSE VSAM Redirector – Capture Exit

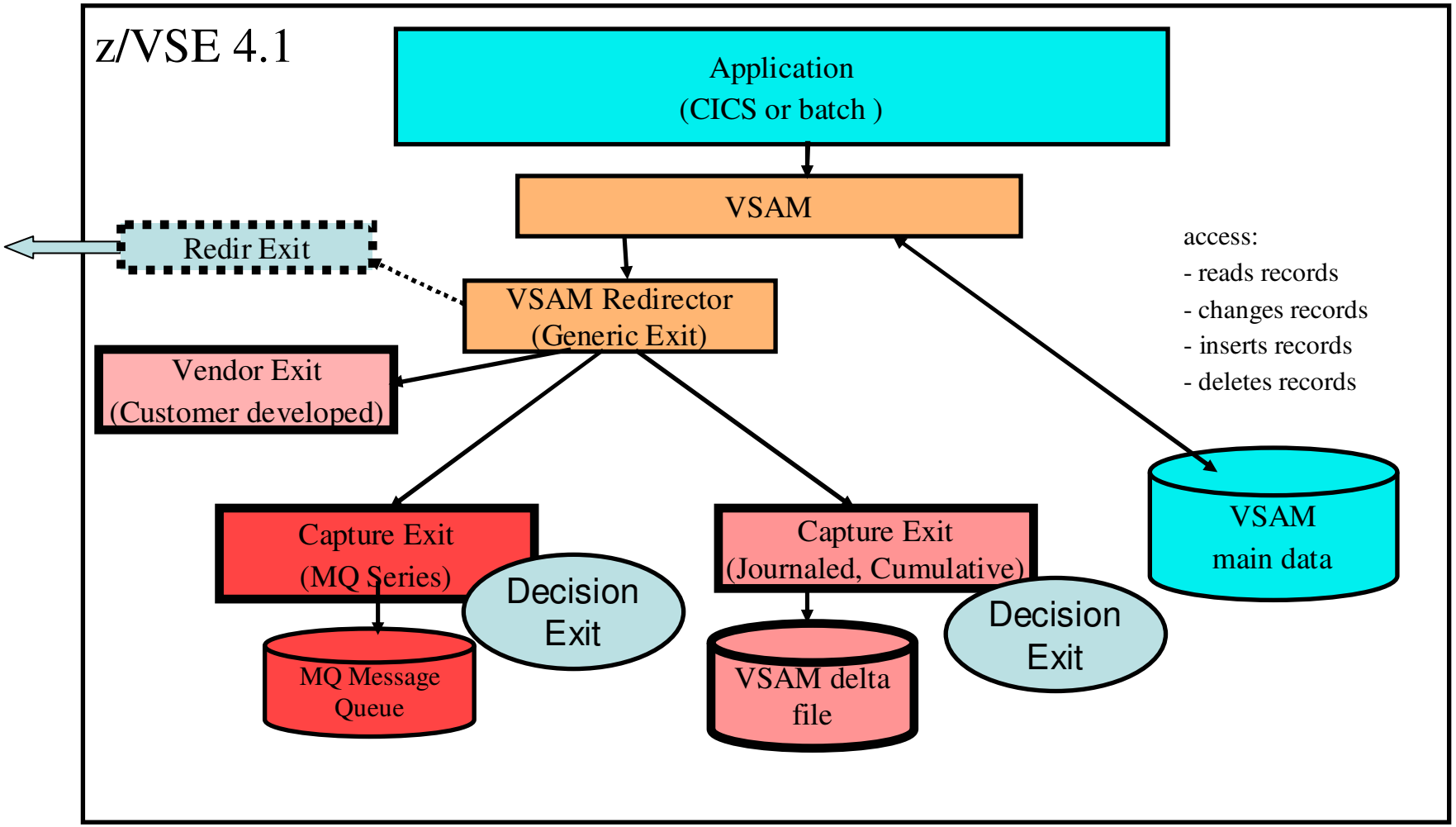


- Shipped in phase IESVSCAP
- Captures all changes to a VSAM file
  - UPDATE
  - INSERT
  - DELETE
- Creates delta records with header + original data
  - Stored in second VSAM file (delta file)
  - Or creates MQ Series message
- Delta record header contains information about when and by whom the record was changed, and which request
- Allows customers to download the delta file and apply it to a database

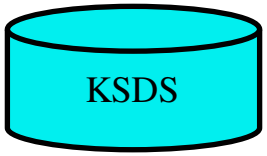
Redirector Capture.
Architectural View



VSAM Redirector Capture



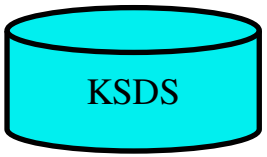
Journaling or cumulative



Delta File

Record 1	inserted
Record 2	inserted
Record 3	inserted
Record 2	updated
Record 1	deleted
Record 3	updated
Record 4	inserted
Record 1	inserted
Record 2	updated
Record 4	updated
Record 4	deleted

Delta File

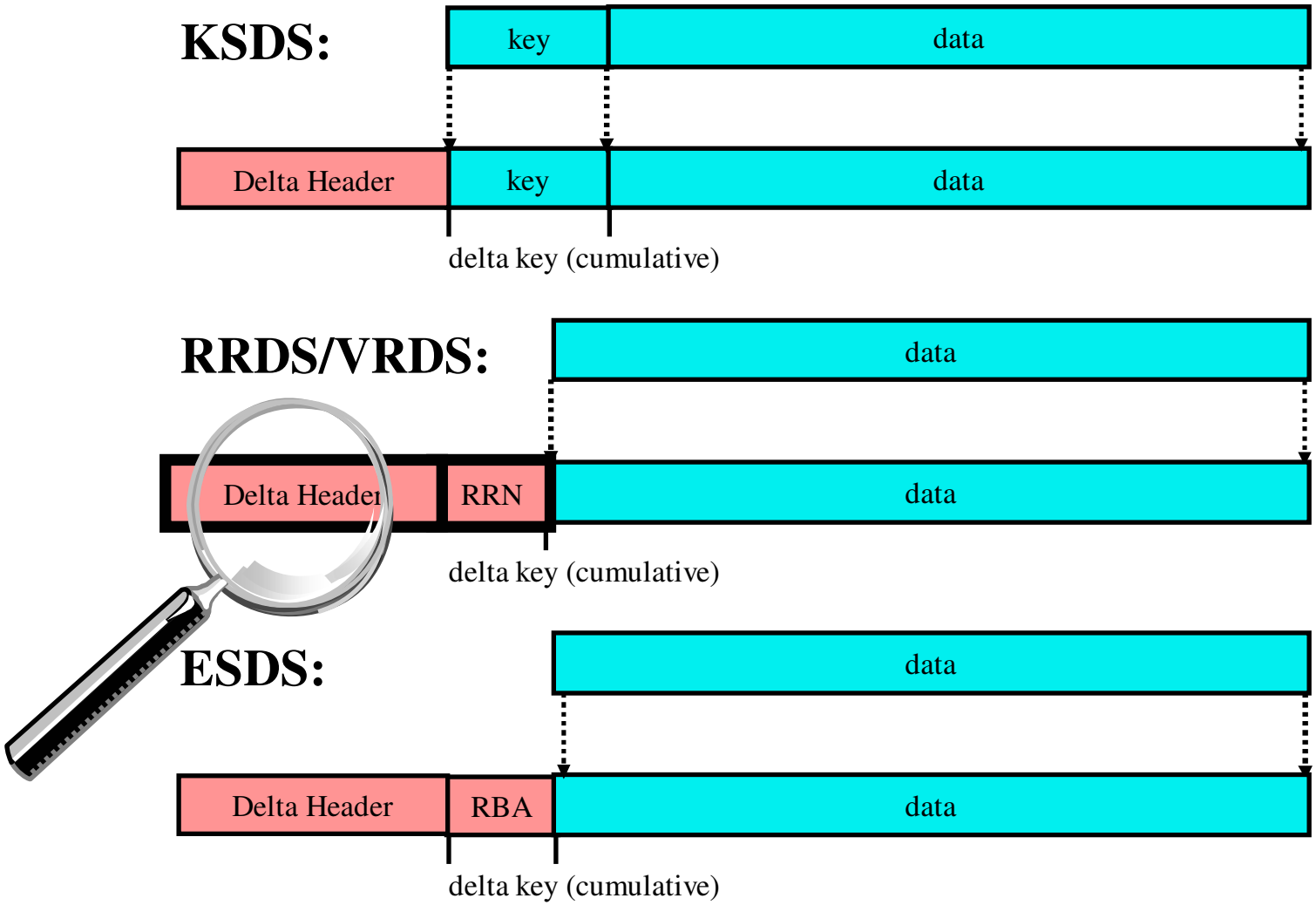


key

Record 1	inserted
Record 2	updated
Record 3	updated
Record 4	deleted

☞ The last version only of a changed VSAM record is stored into the delta file

Delta Record



Delta Header

Offset	Parameters	Length	Description
0	TODCLOCK	8	Time of change
8	JobName	8	Job name
16	PHASEName	8	Phase name
24	Origin	8	String from Config or file Label
32	PartID	2	Partition ID (i.e. F2)
34	**OpCode**	**1**	**I=Insert, D=Delete,U=Update**
35	Flags	1	X'01'=RRN/RBA follows
36	RRN/RBA	4	RRN/RBA (RRDS/VRDS/ESDS)

Contains information about:

- ➔ when change took place (TODCLOCK)
- ➔ who did the change (Job/Phase/Partition)
- ➔ request type of change (Insert/Delete/Update)
- ➔ which record was affected (key/RRN/RBA)

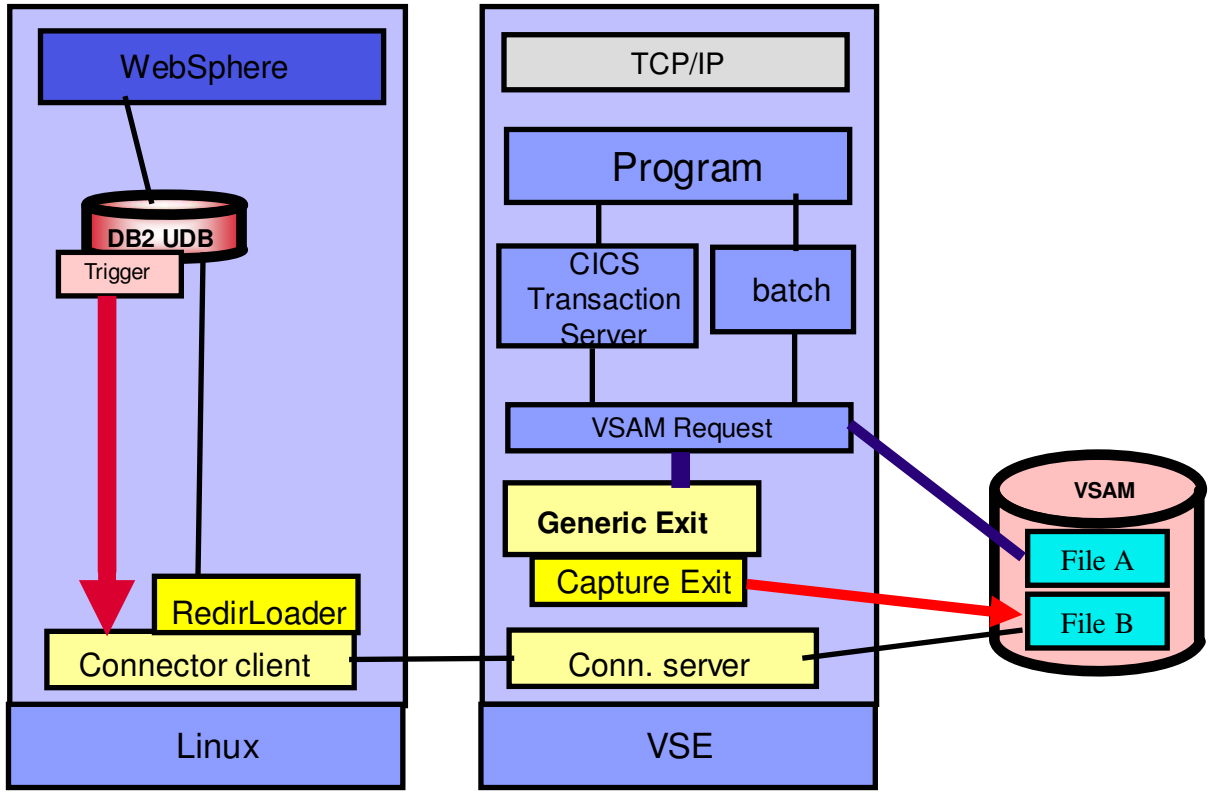
VSAM Capture Exit - Solution options

- **Delta records** can be downloaded for processing
  - Using FTP
  - Using VSE Connector Client
  - After processing, delta records are deleted
- **Delta records** can be sent to a remote system via MQ Series
  - Asynchronous transportation or messages
  - Trigger program can process incoming messages
- Multiple VSAM files can be ‘captured’ into the same Delta File or MQ Series queue
  - Distinction possible via ‘origin’ value in header

Solution for FTP Replacement:

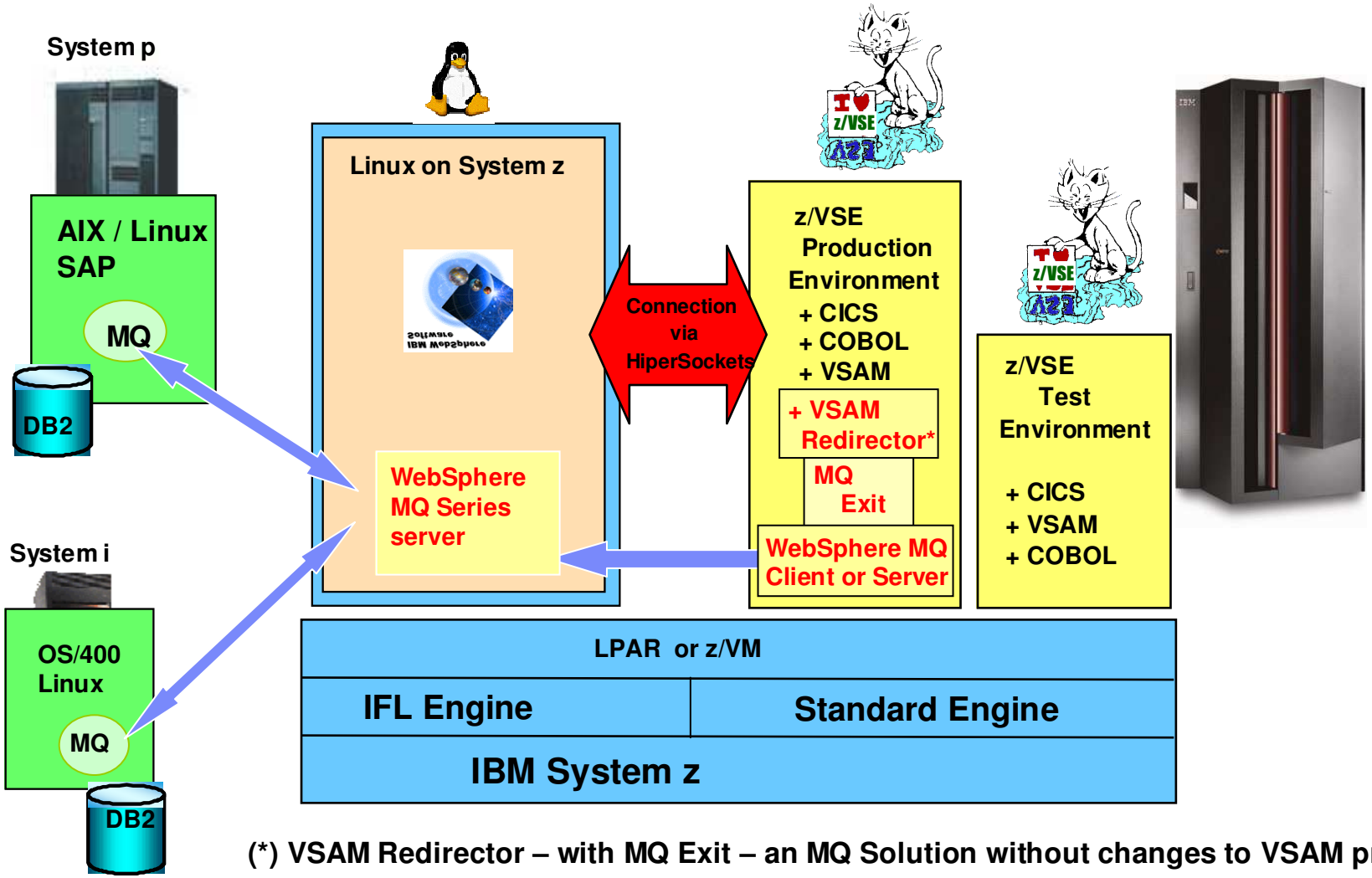
Bidirectional updates with VSE connectors

- With VSAM Capture – the performance of the VSE production system protected
- The changes are processed asynchronously and not influencing the production system



- Collect the changed records in a separate VSAM file
  - Possibility of cleansing
- Process them – with the VSE Connectors provided by IBM

VSAM Programs enablement for MQ solutions



(*) VSAM Redirector – with MQ Exit – an MQ Solution without changes to VSAM programs

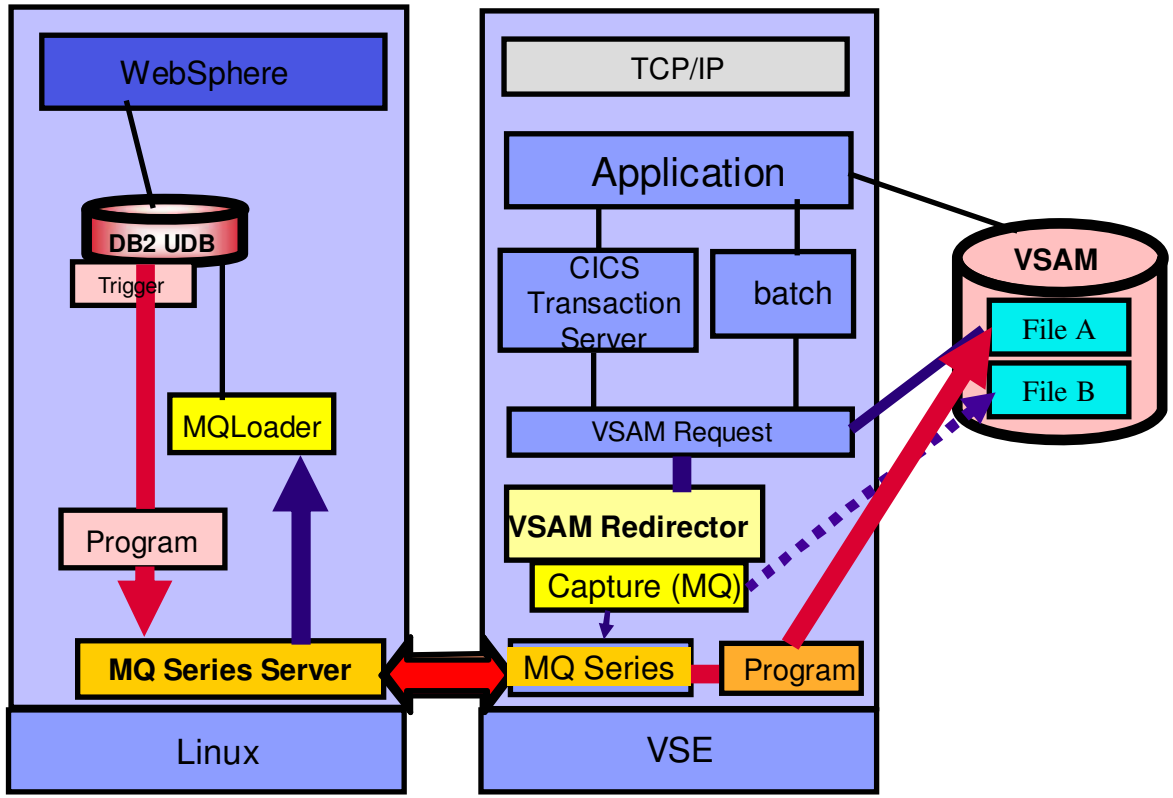
MQ Exit

Integration of VSE Application with MQ Series

► enablement for MQ Series w/o changing existing applications.

►Bidirectional processing

►Guaranteed processing using asynchronous data transfer method MQ

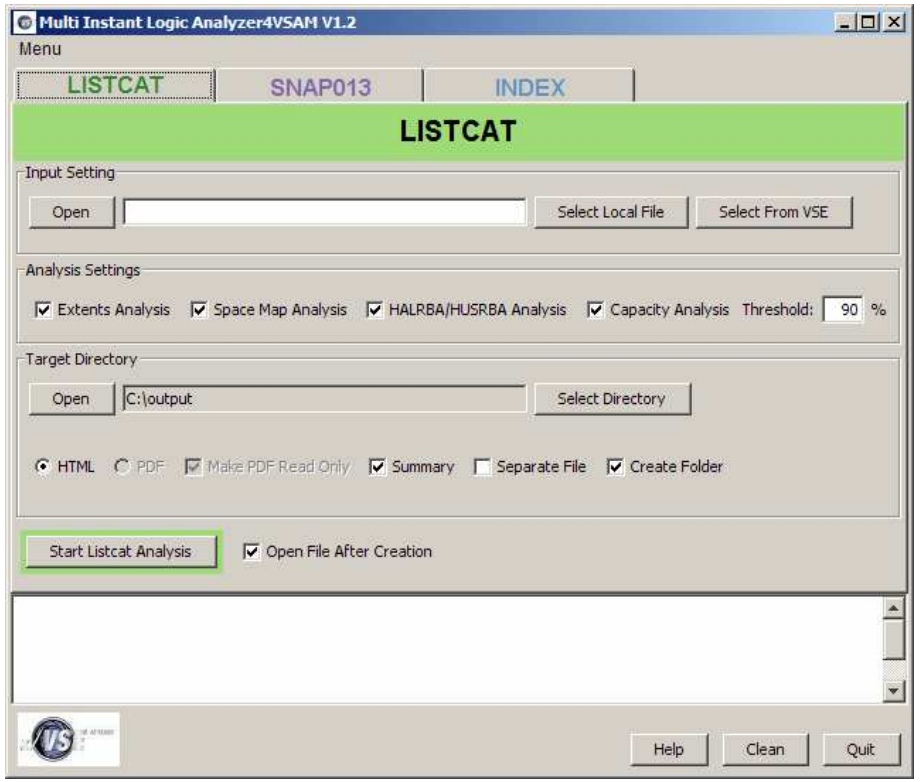


Agenda
z/VSE VSAM new enhancements

- VSAM Considerations and Limits
- VSAM new Enhancements
- Access VSAM files from remote platforms
- VSAM applications can access relational databases
- Tools for z/VSE VSAM files

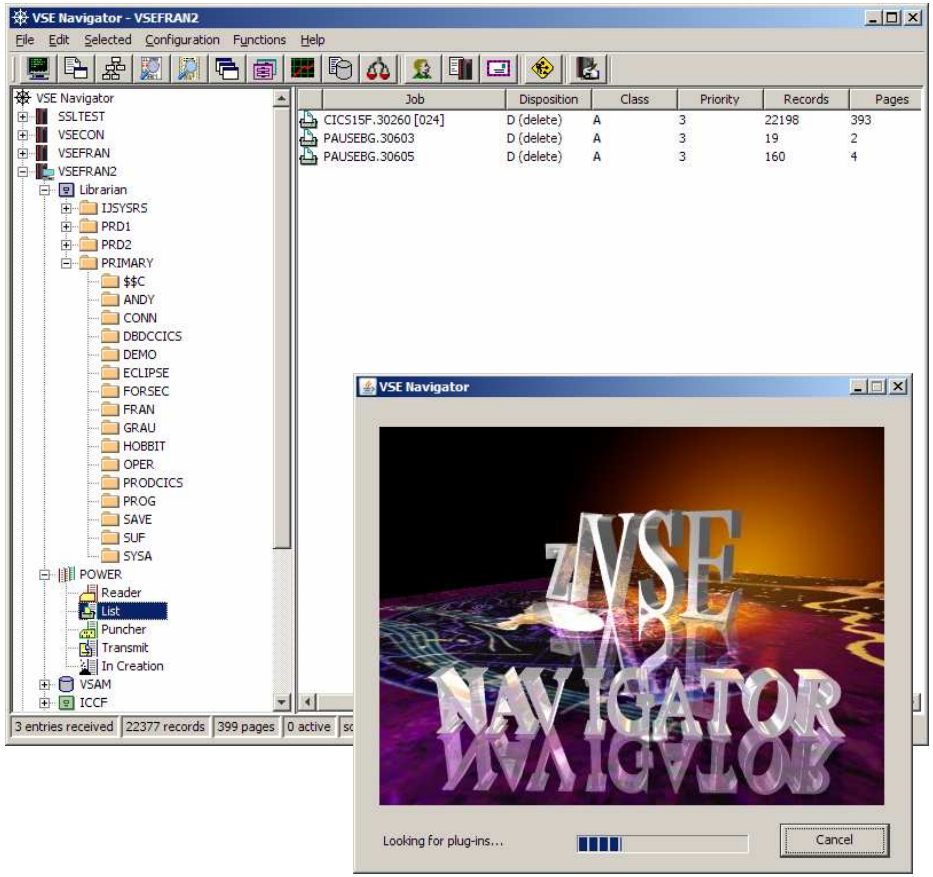
Multi Instant Logic Analyzer4VSAM

- **The Multi Instant Logic Analyzer4VSAM combines several VSAM analysis tools:**
  - **Extent analysis**
  - **Space map analysis**
  - **HALRBA/HUSRBA analysis**
  - **Capacity analysis**
  - **The SNAP013 analysis:**
    - **Extracts Snap013 trace tables from a given hex dump.**
  - **INDEX analysis tool:**
    - **Error analysis**
    - **Index component capacity analysis providing reorganization indicator**

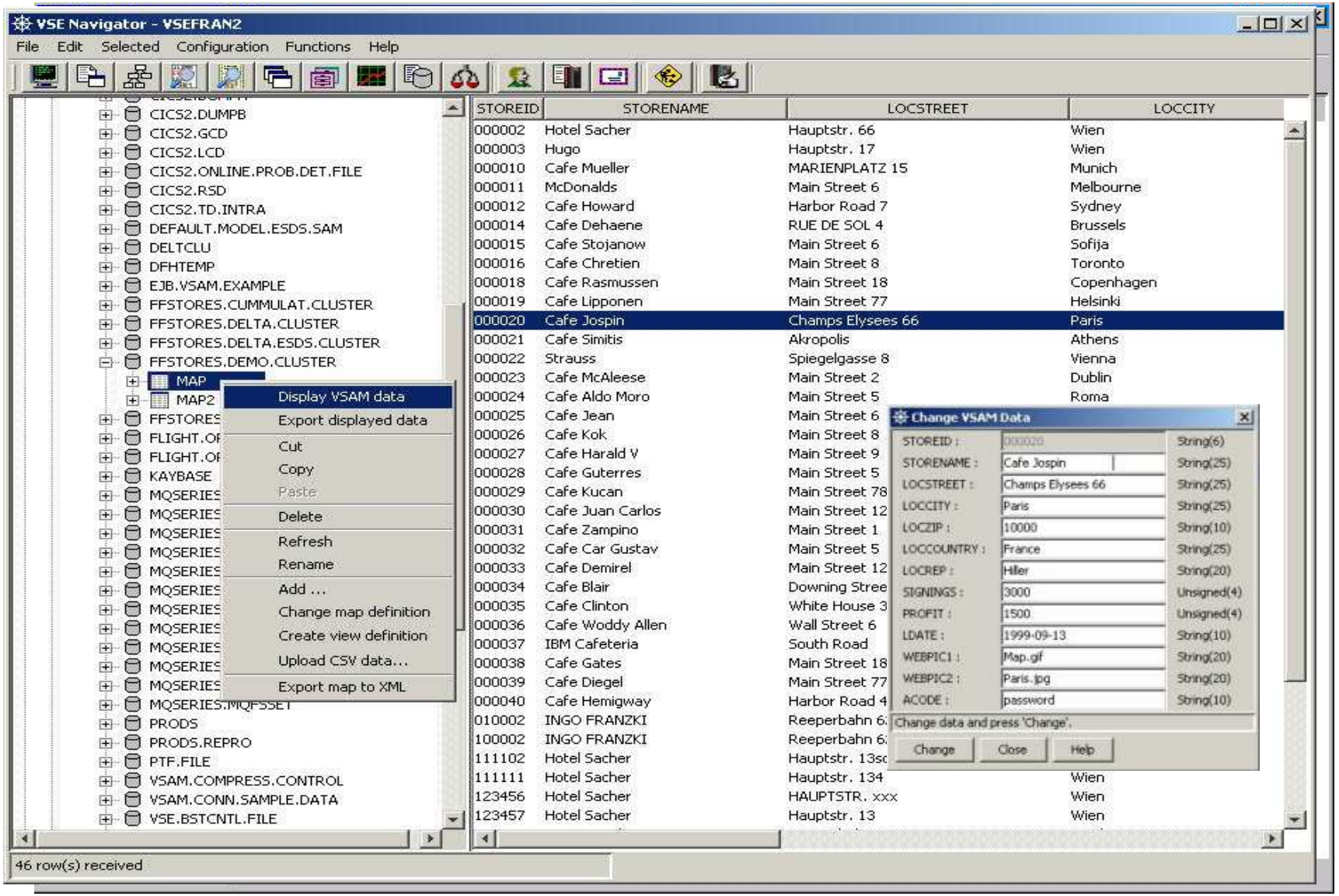


VSE Navigator

- Graphical user interface for z/VSE
  - Look and feel similar to Windows Explorer
- Based on functions provided by VSE Connector Client
- Browse VSE libraries, POWER queues, ICCF libraries, VSAM catalogs
- Copy members via Drag & Drop
- Display and edit members with your favourite editor
- Display and change VSAM data
- Provides graphical system management functions
  - System activity,
  - Retrace MSHP history file
  - ... and many more

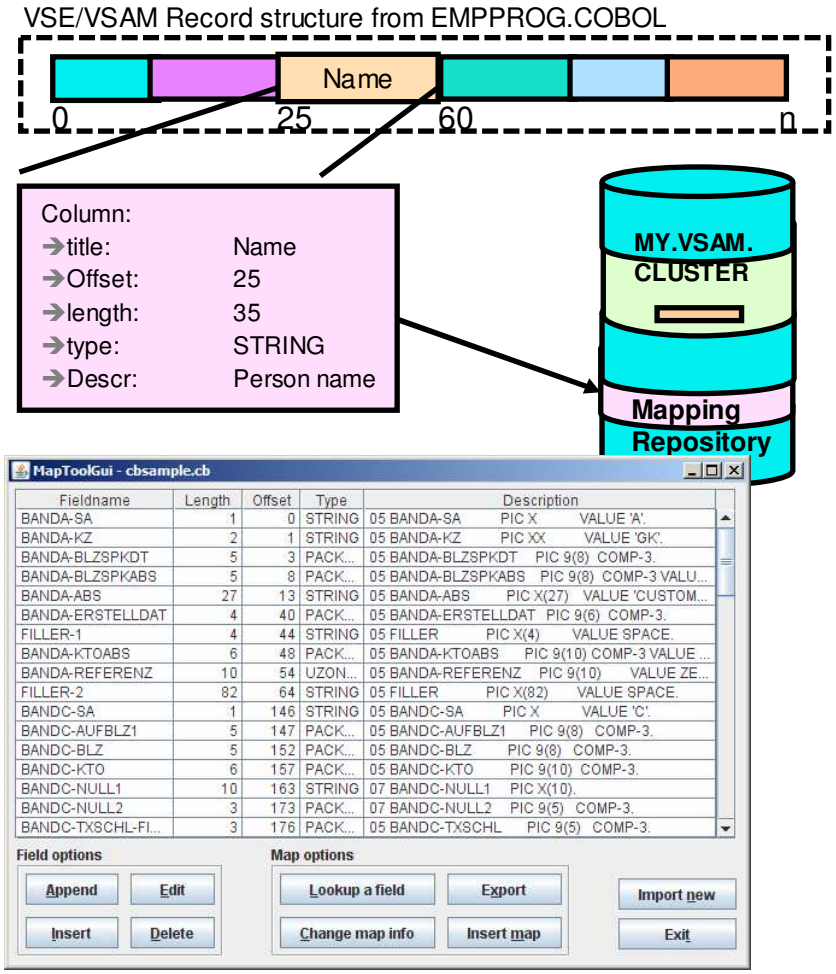


z/VSE Navigator: Windows-like VSE Interface



VSAM Maptool

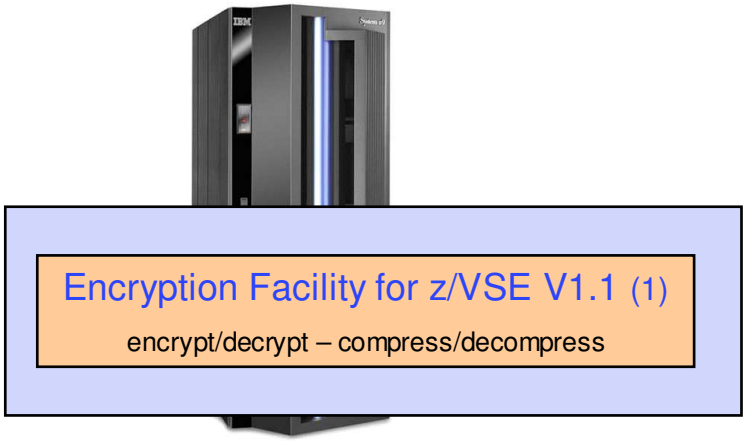
- **Assists you in creating a mapping of your VSAM files**
  - **Mapping is used by VSE Connector Client and VSAM Redirector**
  - **Mapping can also be created using the IDCAMS RECMAP command.**
- **Import Cobol ot PLI copybook to create the mapping from it**
- **Import (receive) a given map from a given z/VSE system**
- **Export a map to a VSE system (send it to z/VSE)**
- **Import a map from a XML file**
- **Export a map to a XML file**
- **Create a Java source file from a given map. The Java program can get all records from the related VSAM file via the given map.**

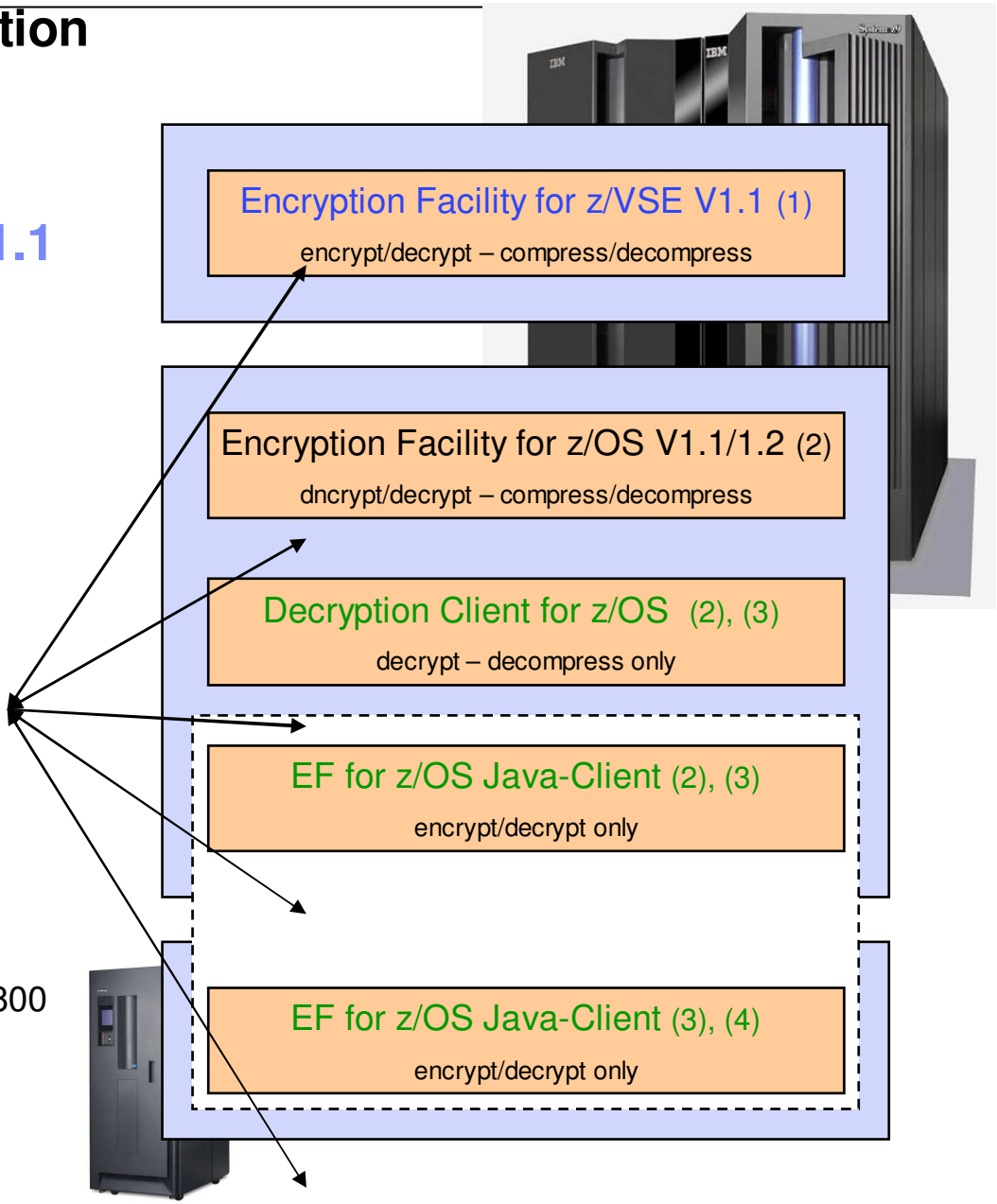


VSAM data Encryption / Protection

z/VSE V4.2 tool:

Encryption Facility for z/VSE V1.1





Note 1: z10 EC, z9 EC, z9 BC, z990/890

Note 2: z10 EC, z9 EC, z9 BC, z990/890, z900/800

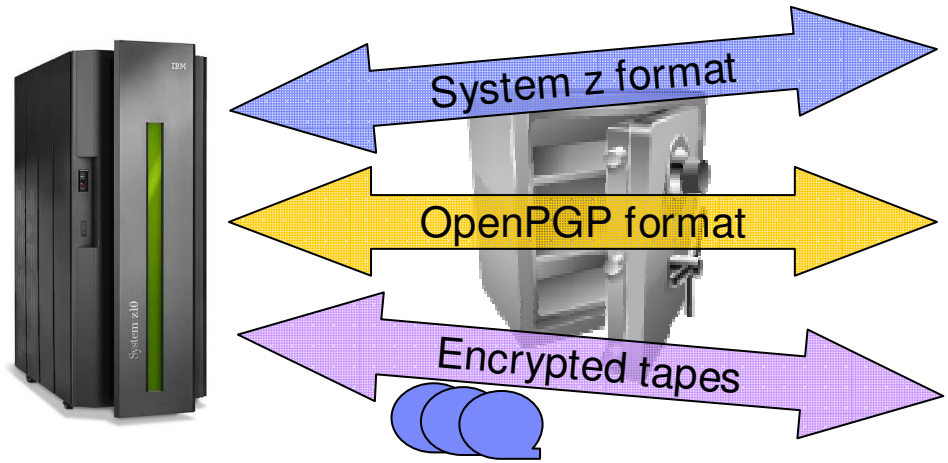
Note 3: No charge, downloadable from web

Note 4: Any Java-capable platform

VSAM data Encryption / Protection

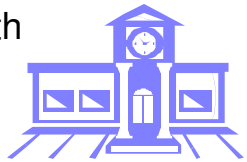
Encryption Facility for z/VSE V1.2 (EF)

- **OpenPGP**
  - Complies with selected OpenPGP standard (RFC 4880) requirements
  - **Encryption of SAM files, VSE/VSAM files,** VSE library members, tapes, or virtual tapes
- **Choice of two formats:**
  - System z format (introduced with EF for z/VSE V1.1) - compatible with EF for z/OS
  - OpenPGP - compatible with other products that are OpenPGP-compliant
- **EF is an optional priced feature for VSE Central Functions V8**
  - Requires z/VSE V4.1 or later
  - MWLC-eligible
- **Exploits hardware encryption technology: CPACF (CP Assist for Cryptographic Function) and Crypto Express2**



- Data exchange with IBM System z servers

- Data exchange with external business partners



- High volume backup/ archive

TS1120
TS1130

http://www.ibm.com/zvse/

Solutions