VisualAge Pacbase 2.5

VA PAC 2.5 – IBM DOS/VSE
OPERATIONS MANUAL VOLUME I : ENVIRONMENT & INSTALLATION

DEPDO001251A

Note

Before using this document, read the general information under "Notices" on the next page.

According to your license agreement, you may consult or download the complete up-to-date collection of the VisualAge Pacbase documentation from the VisualAge Pacbase Support Center at:
http://www.software.ibm.com/ad/vapacbase/support.htm

Consult the Catalog section in the Documentation home page to make sure you have the most recent edition of this document.

First Edition (June 1998)

This edition applies to the following licensed program:

- VisualAge Pacbase Version 2.5

Comments on publications (including document reference number) should be sent electronically through the Support Center Web site at:
http://www.software.ibm.com/ad/vapacbase/support.htm
or to the following postal address:
IBM Paris Laboratory
VisualAge Pacbase Support
30, rue du Château des Rentiers
75640 PARIS Cedex 13
FRANCE

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

NOTICES

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Intellectual Property and Licensing
International Business Machines Corporation
North Castle Drive, Armonk, New-York 10504-1785
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of information which has been exchanged, should contact:

IBM Paris Laboratory
SMC Department
30, rue du Château des Rentiers
75640 PARIS Cedex 13
FRANCE

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

IBM may change this publication, the product described herein, or both.

TRADEMARKS

TABLE OF CONTENTS

1. FOREWORD 7

2. VISUALAGE PACBASE COMPONENTS 9

2.1. INTRODUCTION 10
2.2. CODES OF FUNCTIONS, EXTENSIONS AND UTILITIES 11
2.3. ON-LINE PROGRAM LIBRARY 12
2.4. BATCH PROGRAM LIBRARY 16
2.5. SYSTEM PARAMETER LIBRARY 21
2.6. PROCEDURE LIBRARY 22
2.7. BATCH PROCEDURES 23
2.8. SYSTEM FILES 25
2.9. EVOLVING FILES 29

3. ENVIRONMENT 34

3.1. INTRODUCTION 35
3.2. ON-LINE ENVIRONMENT 36
3.3. ACCESS METHODS 40
3.4. BATCH ENVIRONMENT 41
3.5. TAPE USAGE 42
3.6. FILE SIZE 43
3.7. EFFICIENCY ENHANCEMENT 47
3.7.1. INSTALLATION OF THE INDEX AND DATA FILES 48
3.7.2. VISUALAGE PACBASE STRUCTURE UNDER CICS 49
3.7.3. RESOURCES REQUIRED UNDER CICS 51
3.7.4. EVALUATION OF THE NUMBER OF REQUESTS UNDER CICS 54
3.8. ABNORMAL EXECUTION 56

4. INSTALLATION 59

4.1. PRELIMINARY NOTES 60
4.2. INTRODUCTION 62
4.3. INSTALLATION TAPE 63
4.4. UNLOADING OF INITIAL JCL 65
4.5. INITIAL JCL 68
4.6. COMPLETE JCL INSTALLATION 73
4.7. INSTALLATION VARIANTS 75
4.8. OPTION: SQL INTERFACE (DB2) 76
4.9. JCL MODULES 78
4.10. JCL PARAMETERS 83
4.11. SEPARATION OF JCL MODULES 88
4.12. INSTALLATION PROCESS 90
4.13. 1. CHANGING THE POWER JCL 92
4.14. 2.1 UPDATING THE CICS TABLES (CSD) 94
4.15. 2.2 UPDATING THE CICS TABLES (ASSEMBLER) 100
4.16. 3.1 CATALOGING THE SLI PROCEDURES 108
4.17. 3.2 CATALOGING THE VSE PROCEDURES 116
4.18. 4.1 DEFINITION OF PERMANENT FILES 121
4.19. 4.2 CATALOGING OF GPRT PROCEDURE 124
4.20. 4.3 CATALOGING OF PACX PROCEDURE 128
4.21. 4.4 DEFINITION OF GPRT WORK FILES 131
4.22. 4.5 DEFINITION OF PACX WORK FILES 137
4.23. 4.6 INSTALLATION OF EXECUTABLE MODULES 140
4.24. 4.7 LOADING OF SYSTEM FILES 147
4.25. 4.8 LOADING OF USER FILES 150
4.26. 4.9 LOADING OF ERROR MESSAGES 152
4.27. 5.1 LOADING OF TEST DATABASE 154
4.28. 5.2 INITIALIZATION OF GENERATION-PRINT REQUESTS 156
4.29. 5.3 INITIALIZATION OF PEI MODULE 157

4.30. 6.1 COMPLEMENT: VISUAL DICTIONARY INSTALLATION....158
4.31. 6.2.1 COMPLEMENT: PAF ENVIRONMENT....159
4.32. 6.2.2 COMPLEMENT: PAF+ EXTENSION....160
4.33. 6.3 COMPLEMENT: WORKSTATION....163
4.34. 6.4 COMPLEMENT: PACBENCH QUALITY CONTROL....164
4.35. 6.5 COMPLEMENT: CONTROL CARD INSTALLATION....165
4.36. 6.6 COMPLEMENT: DCOB CATALOGING PROCEDURE....169
4.37. 7. MODIFICATION OF THE CICS JCL....172
4.38. 8.1 COMPLEMENT: PAC/IMPACT MODULE....174
4.39. 8.2 COMPLEMENT: PAC/TRANSFER MODULE....178
4.40. 8.3 COMPLEMENT: BRIDGE VA PAC - TEAMCONNECTION....180
4.41. 9. LIST OF INSTALLED PROGRAMS AND FILES....182
4.42. INSTALLATION TESTS....184
4.43. UTILIZATION TESTS....185
4.44. DATABASE MANAGEMENT TESTS....187
4.45. EXTRACTION UTILITIES TESTS....189
4.46. UTILIZATION TEST: PAC/IMPACT....191
4.47. UTILIZATION TESTS: PAC/TRANSFER....192
4.48. UTILIZATION TESTS: VA PAC - VA SMALLTALK....193

5. REINSTALLATION OF SUB-RELEASE....194

5.1. STANDARD REINSTALLATION....195

6. RETRIEVAL PACBASE 802.02, ..., 1.6....197

6.1. WARNING....198
6.2. RETRIEVAL PACBASE 8.0, 8.0.1....199
6.2.1. RETRIEVAL OPERATIONS....199
6.2.2. RPPG: GENERATION-PRINT FILE (AG) RETRIEVAL....202
6.2.3. PC80 : DATABASE RETRIEVAL....203
6.2.4. RPPG : GENERATION-PRINT FILE (AG) RETRIEVAL....204
6.2.5. PJ80 : ARCHIVED JOURNAL RETRIEVAL....205
6.2.6. PP16 : PRODUCTION ENVIRONMENT RETRIEVAL....208
6.3. RETRIEVAL VA PAC 802.02, ..., 1.6....209
6.3.1. OPERATIONS TO BE DONE....209
6.3.2. RPPG : GENERATION-PRINT FILE (AG) RETRIEVAL....212
6.3.3. PJ16: ARCHIVED JOURNAL RETRIEVAL....213
6.3.4. PP16: PRODUCTION ENVIRONMENT RETRIEVAL....214

1. FOREWORD

FOREWORD

HOW TO USE THIS MANUAL

This manual is intended for the person in charge of installing the VisualAge Pacbase system.

It describes the system's COMPONENTS and its ENVIRONMENT, lays out recommendations for the INSTALLATION of the new release, and explains the operations that must be performed for a standard RE-INSTALLATION of correction versions.

USERS OF PREVIOUS SYSTEM RELEASES

It is generally recommended to install the new release in an environment distinct from that of any earlier release, particularly as far as the installation parameters are concerned. To complete the new installation, the set of tests provided on the installation media must be run.

. Retrieval from VisualAge Pacbase releases older than 8.02v02.

Contact VisualAge Pacbase Support.

. Retrieval from VisualAge Pacbase 8.02v02, 1.2, 1.5, 1.6, 2.0

In this case, the new release may be installed in the same environment as the older release. In this case, refer to chapter '8.02v02, 1.2, 1.5, 1.6 RETRIEVAL', or chapter '2.0 RETRIEVAL'

2. VISUALAGE PACBASE COMPONENTS

2.1. INTRODUCTION

INTRODUCTION

One of the goals of the VisualAge Pacbase system is to manage permanent data in either batch or on-line mode, by using two types of resources:

LIBRARIES which store the system programs, and the parameters needed to run them:

- One On-Line Program library,
- One Batch Program library,
- One System Parameter library,
- One Parameter library for each VisualAge Pacbase Database
- One library for the batch procedure's JCLs.

PERMANENT FILES, containing the data handled by the system programs. These files can be classified into two categories:

. 'System' files, which are not linked to a particuliar VisualAge Pacbase database and remain relatively unchanged,

. 'Evolving' files, which are associated to a VisualAge Pacbase Database, and whose volumes vary according to the updates performed.

NOTES:

The WorkStation, DSMS, revamped DSMS, Pacbase Web Connection, and Pactables Functions are installed independently of the other VisualAge Pacbase functions.

The VisualAge Pacbase-ENDEVOR Interface must also be installed independently from all other functions.

The installation and operation of these Functions and Facilities are described in the operations manuals specific to each one.

2.2. CODES OF FUNCTIONS, EXTENSIONS AND UTILITIES

CODES OF FUNCTIONS, EXTENSIONS AND UTILITIES

The following lists provide abbreviated codes for system functions, extensions, and optional utilities:

```
  .Specifications Dictionary                = DIC

  .Extensions:

     -Personalized Documentation Manager     = PDM
     -Security Systems Interface             = SEC

  .Optional Utilities:

     -Sub-Network Comparison Utility         = LCU
     -Rename/Move Entity Utility             = RME
     -Journal Statistics Utility             = ACT

  .Functions/Facilities:

     -Structured Code                        = SC
     -Batch Systems Development              = BSD
     -COBOL Generator                        = COB
     -On-Line Systems Development            = OSD
     -Client/Server Facility                 = OCS
     -DBD                                    = DBD
     -DBD/ Relational SQL                    = SQL
     -Pactables                              = TAB
     -Development & Support Management System
      (DSMS)                                 = DSM
     -Production Environment Interface       = PEI
     -Dictionary Extensibility               = DEX
     -Pactransfer                            = TRF
     -VA Smalltalk / VA Pac Bridge           = VIS
     -VA Pac / TeamConnection Bridge         = PTC
     -Impact Analysis                        = S2K
     -Pacbench Quality Control               = PQC
     -The WorkStation                        = WST
     -Pacbase Access Facility                = PAF
     -PacReverse                             = REV
     -Pacbase Web Connection                 = PAW
```

The Security Systems Interface and the DBD DB2 functions are not operational with CICS/DOS/VS.

2.3. ON-LINE PROGRAM LIBRARY

THE ON-LINE PROGRAM LIBRARY (MTR8)

Its size is approximately 1,500 blocks of 6,144 bytes.

PROGRAM CODE	FUNCTION OPTION	CORRESPONDING CHOICE COMMENTS
		xxEE transaction
xxPA00	DIC	.Connection screen
xxPA01	-	H
xxPA10	-	HP .
xxPA11	-	LCPC..
xxPA12	-	PC..
xxPA13	-	PT.
xxPA14	-	PE.
xxPA15	-	PU........
xxPA16	-	PK
xxPA17	-	PD
xxPA18	-	LCPU........
xxPA19	-	PW
xxPA20	-	PB
xxPA21	-	PM
xxPA22	-	LCPM
xxPA30	PEI	Prod. Env. Interface menu
xxPA31	-	EE.....
xxPA32	-	EG............
xxPA33	-	ES....
xxPA34	-	LSEP.............
xxPA35	-	ED........
xxPBND	DIC	Abend map
xxPPHP	-	Help
		VA Pacbase transaction
xxQA00	DIC	D..
xxQB00	BSD	R...
xxQC00	DIC	E......
xxQC01	-	LUE
xxQC50	WST	++5 Up/Dw mapping
xxQD00	SC	P......B et O......B
xxQE00	DIC	E......D
xxQF00	COB	P......SC
xxQF10	-	P......STR
xxQG00	DIC	K......
xxQH00	OSD	O......
xxQH01	DIC	LA LC LE LF LM LN LP LS LT LX
xxQH20	OSD	O......CS
xxQH30	-	O......O
xxQI00	-	O......L

```
----------------------------------------------------------
! PROGRAM   !FUNCTION! CORRESPONDING CHOICE              !
! CODE      ! OPTION ! COMMENTS                          !
!--------------------------------------------------------!
! xxQI01    !  OSD   ! O......CE (C1)                    !
! xxQI02    !   -    ! O......CE (C2)                    !
! xxQI03    !   -    ! O......SIM                        !
! xxQI04    !   -    ! O......ADR                        !
! xxQI05    !   -    ! O......CE (C3)                    !
! xxQI20    !   -    ! O......M                          !
! xxQI21    !   -    !                                   !
! xxQI50    !  WST   ! ++4 Up/Dw screen data element     !
! xxQK10    !  DIC   ! M......                           !
! xxQK20    !   -    ! M......CM                         !
! xxQK30    !   -    ! M......CE                         !
! xxQL10    !   -    ! B......                           !
! xxQL20    !   -    ! B......DH                         !
! xxQL21    !   -    ! B......DT                         !
! xxQL30    !   -    ! B......DC                         !
! xxQL40    !  SQL   ! B......DR...                      !
! xxQL41    !   -    ! B......DR...K                     !
! xxQL45    !   -    ! B......GEN                        !
! xxQL46    !   -    !                                   !
! xxQM00    !  SC    ! P......CP et O......CP            !
! xxQP00    !   -    ! P......P  et O......P             !
! xxQP01    !   -    ! Display   P......TC               !
! xxQP02    !   -    ! Display   O......TC               !
! xxQP03    !   -    ! P......TC et O......TC            !
! xxQP04    !   -    ! P......TO                         !
! xxQP05    !   -    ! O......TO                         !
! xxQP06    !   -    ! P......PG et O......PG            !
! xxQP07    !   -    ! O......PG                         !
! xxQP08    !   -    ! P......PG                         !
! xxQP50    !  WST   ! ++6 Up/Dw specific codes          !
! xxQR00    !  DIC   ! LL........L_......                !
! xxQS02    !   -    ! -XP                               !
! xxQS03    !   -    ! -ACT                              !
! xxQS04    !   -    ! WS                                !
! xxQS05    !   -    ! ?                                 !
! xxQS06    !   -    ! Menus                             !
! xxQS08    !  DEX   ! -XQ                               !
! xxQT00    !  DIC   ! T......D                          !
! xxQT10    !   -    ! T......                           !
! xxQT20    !  PDM   ! T......SIM                        !
! xxQT50    !  WST   ! ++2 Up/down textes                !
!-----------!--------! xxEE transaction                  !
! xxQUPA    !  DIC   ! English monitor                   !
! xxQUPF    !   -    ! French monitor                    !
----------------------------------------------------------
```

PROGRAM	FUNCTION	CORRESPONDING CHOICE
CODE	OPTION	COMMENTS
		VA Pacbase transaction
xxQU00	DIC	U..
xxQU01	-	U......D
xxQU10	PDM	V......
xxQU20	-	V......D
xxQV10	DIC	I......
xxQV20	-	I......D
xxQV30	-	-G
xxQX00	-	*
xxQX01	-	LH
xxQY01	DEX	F......
xxQY02	-	F......CE
xxQY03	-	$ ....
xxQY04	-	$ ....D
xxQY05	-	Q......
xxQY10	WST	++1 Design entities Upload
xxQY11	-	++3 Design entities Download
xxQY20	DIC	GP
xxQY30	-	JO
xxQZ00	-	VA Pac initial screen
xxQ000	SC	P......
xxQ100	-	P......CD
xxQ101	-	P......HCD
xxQ102	-	
xxQ103	-	
xxQ104	-	
xxQ200	DIC	S....
xxQ210	TAB	S....SS
xxQ300	DIC	S....CE
xxQ400	BSD	R...L
xxQ500	-	R...D
xxQ600	-	R...CE
xxQ700	SC	P......W et O......W
xxQ800	-	P......8
xxQ900	-	P......9
xxR000	DIC	VA Pacbase monitor
xxR005	-	Return to CICS
xxR100	-	Choice processing
xxR200	-	Format conversion
xxR400	-	N*... ou NH....
xxR500	-	Abend map

PROGRAM CODE	FUNCTION OPTION	CORRESPONDING CHOICE COMMENTS
		VA Pacbase transaction
xxR600	DIC	Text editing
xxR980	-	Screen formatting
xxR990	-	WEB formatting
xxSECT	SEC	Security system interface
		PUF-TP transaction
xxFANM	DIC	
xxFBIB	-	
xxFCTL	-	
xxFDBD	-	
xxFECR	-	
xxFENU	-	
xxFFOG	-	
xxFMCL	-	
xxFPGM	-	
xxFRUB	-	
xxFSDO	-	
xxFTXT	-	
xxFVER	-	
xxF000	-	
Sub-programs called in the generated programs :		
CALL	OSD	For I.M.S. OLSD
PBLTDLI	-	IMS or CICS DL/1 OLSD
xxTPST	PAF	Online PAF
xxTPWS	-	Keywords on-line PAF
(must be inserted into the application program library)		

2.4. BATCH PROGRAM LIBRARY

THE BATCH PROGRAM LIBRARY (MBR8)

Its size is approximately 2,000 blocks of 6,144 bytes.

CODE	PROCEDURES	OPTION	COMMENTS
PACA05	UPDT	DIC	
PACA15	UPDT REST	-	
PACA10	GPRT	-	
PACA20	-	-	
PACB	-	-	GPRT monitor
PACB30	-	-	
PACB31	-	SQL	
PACB40	-	DBD	
PACB80	-	-	
PACCTL	PACX	DIC	
PACC30	GPRT	COB	
PACC40	-	-	
PACC80	-	-	
PACDTP	INSL	DIC	
PACD30	-	-	
PACD40	-	-	
PACD80	-	-	
PACD90	-	-	
PACE30	-	OSD	
PACE40	-	-	
PACE80	-	-	
PACFGY	PACX	DIC	
PACFMB	-	-	
PACFTD	-	-	
PACG3C	GPRT	OCS	
PACG3S	-	-	
PACG4S	-	-	
PACG8C	-	-	
PACG8S	-	-	
PACHOI	PACX	DIC	
PACINS	VINS	-	
PACK30	GPRT	OCS	
PACK80	-	-	
PACK90	-	-	
PACL30	-	SC	
PACL80	-	-	
PACL90	-	-	
PACL92	EMUP	-	
PACL93	EMLD	-	

CODE	PROCEDURES	OPTION	COMMENTS
PACL95	GPRT	PAW	
PACM30	-	DIC	
PACM80	-	-	
PACN30	-	PDM	
PACN35	-	-	
PACN40	-	-	
PACN50	-	-	
PACN80	-	-	
PACP30	-	SC	
PACP40	-	-	
PACP80	-	-	
PACP82	-	-	
PACQ	PQCA	PQC	PQCA Monitor
PACR01	INPE	PEI	
PACR10	PRPE	-	
PACR20	GPRT	-	
PACR22	SIPE	PEI	
PACR30	HIPE	-	
PACR40	GRPE	-	
PACR60	SVPE	-	
PACR61	RSPE	-	
PACSJO	PACX	DIC	
PACSMD	-	-	
PACSPU	-	-	
PACSRM	-	-	
PACS30	-	-	
PACS40	-	-	
PACS50	-	-	
PACS60	-	-	
PACS75	-	-	
PACS80	-	-	
PACTIN	GETI	TAB	
PACTI1	GET0	-	
PACT40	GETA GETD	-	
PACT41	GET1 GET2	-	
PACT45	GETA GETD	-	
PACT50	GETD	-	
PACT51	GET2	-	
PACU15	PARM	DIC	
PACU80	-	-	
PACU99	CRYP	-	
PACX	PACX	-	
PADM10	SADM	WST	SSADM integrity check
PAFP10	PPAF GPRP	PAF	PAF pre-processor
PAF900	UPDP	-	PAF update input

CODE	PROCEDURES	OPTION	COMMENTS
PAN200	INFQ	S2K	
PAN205	INFP	-	
PAN210	ISEP	-	
PAN212	ISOS	-	
PAN215	ISEP IANA	-	
PAN220	IPFQ IANA	-	
	IPEP	-	
PAN230	IANA	-	
PAN240	IPFQ	-	
PAN250	IANA	-	
PAN255	IGRA	-	
PAN260	IANA	-	
PAN270	IPIA	-	
PAN280	IPIA	-	
PBBTST	Sub-pgm	PAF	Called by batch user pgms
PBBTWS	-	-	- - - - - - - - -
PBBT98	-	-	Called by PBBTST PBBTWS
PDS600	DEXP	DSM	DEXP before 2.0
PDS610	-	-	- -
PREI00	RVDE	REV	PACREVERSE Interface
PREI40	RVKE	-	- -
PREI50	-	-	- -
PRE986	RVDE	-	- -
PTARSD	RMTD	TAB	Migration: rest. TD
PTAR20	RPTD	-	Migration: retrieval TD
PTASVD	SMTD	-	Migration: saves TD
PTC010	TCLS	PTC	VA Pac-TeamConnection
PTC030	-	-	- -
PTC100	TCGP	-	- -
PTC200	TCME	-	- -
PTC220	TCCI TCME	-	- -
PTC400	TCCI	-	- -
PTC440	-	-	- -
PTED30	XPDM	PDM	PDM Extension
PTED60	-	-	- -
PTEP90	PRGS	-	- -
PTEXD0	XPAF	PAF	PAF Extension
PTEX30	-	-	- -
PTEX80	-	-	- -
PTUBAS	SAVE UPDT	DIC	Database integrity check
PTUCSS	CSES	-	Compression Session No.
PTUESS	ESES	-	Extraction Session No.

CODE	PROCEDURES	OPTION	COMMENTS
PTUG05	TRJC	TRF	Pac/Transfer Facility
PTUG06	-	-	- -
PTUG07	-	-	- -
PTUG10	TRUP	-	- -
PTUG11	-	-	- -
PTUG12	-	-	- -
PTUG42	TRDU	-	- -
PTUG44	-	-	- -
PTUG46	-	-	- -
PTUG50	TRPF	-	- -
PTUG60	TRRP	-	- -
PTUG61	-	-	- -
PTUG90	TRRT	-	- -
PTULOI	RTLO	DIC	Locks retrieval
PTULVB	LVBL	-	Repl. low-values with blanks
PTUQ10	PQCE	PQC	
PTUQ15	-	-	
PTUQ20	PQCA	-	
PTUQ24	-	-	
PTUQ25	-	-	
PTUQ30	-	-	
PTUR00	STOP	-	
PTU001		DIC	All procedures with input transactions
PTU004	REST REAG	-	User code check
PTU100	MLIB	DIC	
PTU120	-	-	
PTU130	SASN	LCU	
PTU140	-	-	
PTU2CL	REOR	DIC	
PTU200	-	-	
PTU208	-	-	
PTU210	-	-	
PTU220	-	-	
PTU240	-	-	
PTU300	ARCH	DIC	
PTU320	-	-	
PTU380	REST	-	
PTU400	-	-	
PTU402	RESY	-	
PTU420	REST	-	
PTU500	SAVE	-	
PTU502	SASY	-	

CODE	PROCEDURES	OPTION	COMMENTS
PTU550	SVAG	DIC	
PTU560	REAG	-	
PTU630	ACTI	ACT	
PTU640	-	-	
PTU810	EMSN	LCU	
PTU815	MESN	-	
PTU850	CPSN	-	
PTU855	-	-	
PVA100	VDWN	VIS	
PVA110	-	-	
PVA300	VUP1	-	
PVA310	-	-	
PVA320	VUP2	-	
PVA400	VPUR	-	
PYSMCC	YSMC	WST	YSM consistency check
PYSMC2	-	-	
PYSMC3	-	-	
UTIXSR	UXSR	DIC	
PACSECB	SUB-PROG	SEC	Called by VA Pac programs
PACN25	GPRT	PDM	- - -
PACN90	-	-	- - -
PACSEP	GPRT UPDT REST	DIC	Banner print sub-program (to separate reports)
PACA90	GPRT UPDT REST	-	Data Element format Analysis sub-program
Retrieval from Pacbase 802.02, 1.2, 1.5, 1.6			
PACR90	PP16	PEI	PEI backup
REP2PJ	PJ16	DIC	Archive
Retrieval from VA Pacbase 2.0			
PTU908	RPPG	DIC	Generation-printing req.

2.5. SYSTEM PARAMETER LIBRARY

THE (SLI) SYSTEM PARAMETER LIBRARY

Its size does not exceed one track of a 3380 disk unit.

It contains the JCL members or IDCAMS commands inserted into$ SLI').

- DFxxyyff contains the IDCAMS commands for defining the database files,

```
           where: ff   = the file code,
                  xxyy = a prefix made up of the system root and of
                         a code representing either a system or
        user
                         file.
```

- RGxxyyff contains the IDCAMS commands for rotating the back-up files, permitting the centralized management of the number of generations saved.

- DBJxxyy contains the JCL inserted at the beginning of each job (resource definitions, standards).

- ppppxxyy contains the JCL for certain procedures, where: pppp = the procedure code.

In order to simplify installation, it is preferable to use an existing library already declared in the POWER JCL in LIBDEF SOURCE,SEARCH.

2.6. PROCEDURE LIBRARY

THE (VSE) PROCEDURE LIBRARY

Its size does not exceed one track of a 3380 disk unit.

It contains the procedures for parameterized JCL and for defining sort space or non-VSAM work space.

It must be shared by VA PAC , PACTABLE, and DSMS in order to avoid work space conflicts between partitions.

2.7. BATCH PROCEDURES

THE BATCH PROCEDURES

Procedures associated with batch processing are described in Parts II and III of the Operations Manual (Batch Procedures: Administrator's Guide --for those procedures involving only the Database Administrator-- and User's Guide --for those available to the user).

The documentation of each procedure includes the following:

. General presentation

- Introduction
- Execution conditions
- Corrections in case of an ABEND

. Description of user input, processing and results, as well as recommendations for use.

. Description of steps
- List of permanent and temporary files in use,
- Return codes (if any) produced by each step.

. Command file (JCL lines)

PROCEDURE CLASSIFICATION

Batch procedures are documented in the following manuals:

Batch procedures - Administrator's guide:

1) Database Management utilities.

2) Versioning utilities (PEI and Pactransfer).

3) Manager's utilities.

4) Migrations.

Batch procedures - User's guide:

1) Standard procedures.

2) Personalized extraction and automated documentation.

3) Quality analysis and control.

4) Methodology integrity check.

5) Pactables.

6) Impact Analysis.

7) VisualAge Smalltalk / VisualAge Pacbase Bridge.

Environment and installation:

Previous release retrieval procedures:

. Releases 8.02v02 - 1.6:
- Archive Journal retrieval (PJ16)
- Sequential PEI backup retrieval (PP16)

. Release 2.0:

- Generation-print Requests file retrieval (RPPG)

2.8. SYSTEM FILES

SYSTEM FILES

Besides the libraries described in the preceding sub-chapters, the VA PAC system includes the following permanent files:

. A file containing the ERROR MESSAGES and the ON-LINE DOCUMENTATION of the VA PAC system: AE0.

```
. Size         : approx. 32,000 records
. Organization : SAM-ESDS
. Recsize      : 80
. Utilization  : Batch
. Definition   : SLI member DF$PRO.$PRO.AE0
```

This file is not directly used in normal system operation. It is concatenated with the USER PARAMETERS required by the on-line operation of the system (*), to form one file called AE.

```
. Size         : = AE0 + user parameters
. Organization : VSAM-KSDS
. Recsize      : 80
. Key          : 12 (position 0)
. Utilization  : batch and on-line
. Definition   : SLI member DF$PRO.$PRO.AE
```

. A file containing the USER PARAMETERS required by batch operations only (*): AP.

```
. Size         : user parameters
. Organization : VSAM-KSDS
. Recsize      : 80
. Key          : 7 (position 0)
. Utilization  : batch and on-line
. Definition   : SLI member DF$PRO.$PRO.AP
```

(*) The AE file handles the following USER PARAMETERS:

-User codes,
-VA PAC access keys,
-Security System Interface activation,
-Activation of blank password check,
-Activation of batch procedure execution check,
-Text types,
-Management of accented character conversion,
-Pacdesign methodology parameterization, and
-the DSMS-VA PAC connection.

The AP file handles the following USER PARAMETERS:
modification of fixed parts of standard error messages,
and control cards necessary for the generation of programs.

All user parameters are managed by a special transaction and a special procedure (PARM). For more information, refer to Chapter 'PARM: USER PARAMETER UPDATE', in the 'BATCH PROCEDURES' part of this manual.

PARM builds the AE and AP files, and manages:

. A USER PARAMETER backup file: PE.

```
. Organization : sequential (SAM-ESDS or SAM)
. Physical name: $PXU.SEQ.$PRO$PRO.PE
. Format type  : fixed
. Length       : 80
. Utilization  : Batch
. Rotation     : SLI member RG$PRO.$PRO.PE
```

The PE file is the actual backup of ALL THE USER PARAMETERS found in the AE and AP files.

. A skeleton generation file: SC, used by the STRUCTURED CODE and BATCH SYSTEMS DEVELOPMENT functions:

```
. Size         : approximately 40 records
. Organization : VSAM-KSDS

. Recsize      : 3,204
. Key          : 4 (position 0)
. Definition   : SLI member DF$PRO.$PRO.SC
```

. A skeleton file for generation, SG, used by the ON-LINE SYSTEMS DEVELOPMENT, DATABASE DESCRIPTION, and SPECIFICATIONS DICTIONARY functions:

```
 .Size         : approximately 360 records
 .Organization : VSAM-KSDS
 .Recsize      : 4,605
 .Key          : 5 (position 0)
 .Utilization  : Batch only
. Definition   : SLI member DF$PRO.$PRO.SG
```

. A skeleton file for generation, SR, used by the COBOL GENERATOR function:

```
 .Size         : approximately 25 records
 .Organization : VSAM-KSDS
 .Recsize      : 4,605
 .Key          : 5 (position 0)
 .Utilization  : Batch only
. Definition   : SLI member DF$PRO.$PRO.SR
```

. A skeleton generation file SP, used by the XPAF function for the generation of extractors:

```
 .Size         : approximately 5 records
 .Organization : VSAM-KSDS
 .Recsize      : 4,605
 .Key          : 5 (position 0)
 .Utilization  : Batch only
. Definition   : SLI member DF$PRO.$PRO.SP
```

. A skeleton generation file SF, used by the XPAF function for the generation of extractors:

```
 .Size         : approximately 3,000 records
 .Organization : SAM-ESDS
 .Recsize      : 119
 .Utilization  : Batch only
. Definition   : SLI member DF$PRO.$PRO.SF
```

. A skeleton file for generation, SS, used for the CLIENT/SERVER ON-LINE SYSTEM DEVELOPMENT functions:

```
 .Size         : approximately 300 records
 .Organization : VSAM-KSDS
 .Recsize      : 4,605
 .Key          : 5 (position 0)
 .Utilization  : Batch only
. Definition   : SLI member DF$PRO.$PRO.SS
```

PAF FUNCTION (VA PAC ACCESS FACILITY)

An indexed work file is required to use the PAF function, for all on-line user programs accessing databases which have the same root.

. PAF on-line work file (PA)

```
. Organization : VSAM-KSDS
. Format       : variable
. Recsize      : average 200, maxi 539
. Key          : 37 (position 2)
. Utilization  : updated by sub-programs
                 xxTPST and xxTPWS,
                 called by the on-line user
                 programs
. Definition   : SLI member DF$PRO.$PRO.PA
```

To use the PAF function, an indexed work file is also required for all batch user programs. This file is defined by the NOALLOCATION option and can be deleted at the end of the job or simply de-allocated (using IDCAMS VERIFY with DISP=NEW in DLBL).

If PAF user programs are to be run simultaneously in several partitions, the '%' character must be placed on the first position of the physical name. This automatically generates a suffix linked to the partition, concatenated to the file name.

. PAF batch work file

```
. Organization : VSAM-KSDS
. Format       : variable
. Recsize      : average 170, maxi 468
. Key          : 12 (position 0)
. Utilization  : Updated by sub-program
                 PBBT98 called by PBBTST/PBBTWS
                 (PAF sub-programs called by
                 the user programs)
. Definition   : SLI member DFSYSPAF
```

PAF FUNCTION EXTENSION

. An extraction Master Paths file (GS) containing the user extractors and macro-commands.

```
. Organization : VSAM-KSDS
. Recsize      : 203
. Key          : 25 (position 0)
. Utilization  : Batch and TP
. Definition   : SLI member DF$PRO.$PRO.GS
```

2.9. EVOLVING FILES

VA PAC DATABASE 'EVOLVING' FILES:

These files contain user information. They are handled by the system in on-line or in batch mode.

The first four make up the actual VA PAC Database. They contain all data related to application development, as follows:

THE DATA FILE (AR)

```
     .Organization : VSAM-RRDS
     .Recsize      : 140
     .Utilization  : Batch and On-line
     .Definition   : SLI member  DF$PRO.$PFI.AR
```

Each VA PAC line managed by the system is stored in the data file under a fixed internal number. The successive states of a given line from the various archived sessions form a chain; at the beginning of this chain is the most recent state of the line, and at the end, the oldest state of the line. Programs never access a VA PAC line directly in this file, but first obtain the number of the top of the chain by consulting the index file (AN).

THE INDEX FILE (AN)

```
     .Organization : VSAM-KSDS
     .Recsize      : 54
     .Key          : 43 (position 0)
     .Utilization  : Batch and On-line
     .Definition   : SLI member  DF$PRO.$PFI.AN
```

The index file, by the contents of its key, describes the various views of the VA PAC Database that are offered to the user. It identifies the VA PAC line based on its position in the database and on the consultation view. It also ensures additional technical functions.

The essential information provided here is the internal num- ber of the VA PAC line to which the index points.

THE GENERATION-PRINTING REQUEST FILE (AG)

```
     .Organization : VSAM-KSDS
     .Recsize      : 150
     .Key          : 26 (position 0)
     .Utilization  : Batch and On-line
     .Definition   : SLI member  DF$PRO.$PFI.AG
```

This is a storage area for users, which allows input and manipulation of user generation-printing requests. This file is rather small, however it is subject to heavy update activities on a daily basis. It is saved by the SVAG procedure.

It is initialized, restored and can be reorganized by the REAG procedure.

THE JOURNAL FILE (AJ)

```
     .Organization : VSAM-RRDS
     .Recsize      : 165
     .Utilization  : Batch and On-line
     .Definition   : SLI member  DF$PRO.$PFI.AG
```

All update transactions performed on the database, whether in batch or on-line, are saved in order 1) to allow database restoration should the standard security system fail, and 2) to provide an information resource to the database evolution analysis.

Transactions are usually stored in the Journal back-up file (PJ). The Journal file is only used as a transition between the time the transactions are processed and the time the ARCH procedure sends them to their final destination: the PJ file.

NOTE:

The transactions contained in the Generation-Printing Request file (AG) are not saved in the Journal file (AJ).

Three sequential files, found either on a disk or on tape, make up the back-up of the database.

When on a disk, the number of generations saved at any time is centrally controlled in an SLI member. At the time of installation, the generations saved are G(-1) and G(0), which are coded OLDff and ff.

DATABASE BACK-UP FILE (PC)

```
    .Organization  : Sequential (VSAM-ESDS or SAM)
    .Physical Name : $PXU..SEQ.$PRO$PFI.PC
    .Format        : Variable
    .Utilization   : Batch
    .Size          : 149 bytes per data record
                     55 bytes per index record
    .Definition    : SLI member  DF$PRO.$PFI.PC
    .Rotation      : SLI member  RG$PRO.$PFI.PC
```

This is a sequential back-up file of the VA PAC components: Index (AN), Data (AR).

JOURNAL BACK-UP FILE (PJ)

```
    .Organization  : Sequential (VSAM-ESDS or SAM)
    .Physical Name : $PXU..SEQ.$PRO$PFI.PJ
    .Format        : Fixed
    .Length        : 165
    .Utilization   : Batch
    .Rotation      : SLI member RG$PRO.$PFI.PJ
```

The purpose of this file is to store all update transactions that have affected the VA PAC Database since installation and that have passed through the Journal file (AJ).

When its size becomes incompatible with operations needs, the ARCH procedure divides the PJ file into several files and the most recent one is used on a daily basis.

GENERATION-PRINTING REQUEST BACK-UP FILE (PG)

```
    .Organization  : Sequential (VSAM-ESDS or SAM)
    .Physical Name : $PXU..SEQ.$PRO$PFI.PG
    .Format        : Fixed
    .Length        : 150
    .Utilization   : Batch
    .Rotation      : SLI member RG$PRO.$PFI.PG
```

The purpose of this file is to back up the generation-printing requests and to reorganize them using the REAG procedure.

'PEI' FUNCTION (PRODUCTION ENVIRONMENT INTERFACE)

For sites where the PEI Function is available, three three additional files are managed either in on-line or batch mode. (See the Chapter dedicated to this option). These files contain all the data necessary for the management of the production environment.

BATCH PRODUCTION ENVIRONMENT FILE (AB)

```
        .Organization : VSAM-KSDS
        .Recsize      : 110
        .Key          : 26 (position 0)
        .Utilization  : Batch and on-line reads
        .Definition   : SLI member  DF$PRO.$PFI.AB
```

ON-LINE PRODUCTION ENVIRONMENT FILE (AC)

```
        .Organization : VSAM-KSDS
        .Recsize      : 110
        .Key          : 26 (position 0)
        .Utilization  : Batch and on-line
        .Definition   : SLI member  DF$PRO.$PFI.AC
```

PEI BACKUP FILE (PP)

```
        .Organization : Sequential (VSAM-ESDS or SAM)
        .Physical Name: $PXU..SEQ.$PRO$PFI.PP
        .Format       : Fixed
        .Length       : 110
        .Utilization  : Batch
        .Rotation     : SLI member  RG$PRO.$PFI.PP
```

DEVELOPMENT AND SUPPORT MANAGEMENT SYSTEM (DSMS) FUNCTION

For sites where the Development and Support Management Function (DSMS) is installed, a file from this module is read in batch and on-line mode by VA PAC. (For more information, refer to the DSMS Operations Manual.)

This file contains the list of VA PAC entities which are to be updated for each CHANGE NUMBER. (The Change Number is entered on the VA PAC Sign-On screen.)

DSMS VA PAC ENTITY FILE (DC)

```
.Organization : VSAM-KSDS
.Format       : Variable
.Recsize      : Minimum: 50 - Maximum: 168
.Key          : 31 (position 2)
.Utilization  : Read by on-line or batch mode
                updates.
```

This file's allocation and initialization are performed when installing the DSMS Function.

3. ENVIRONMENT

3.1. INTRODUCTION

INTRODUCTION

This chapter details the environment and the resources required by VisualAge Pacbase, so as to help you to determine the necessary disk space.

3.2. ON-LINE ENVIRONMENT

ON-LINE ENVIRONMENT (CICS)

The monitor used is CICS VSE since the Release 1.7.

It must include BMS (the PAGING option is not used by the VA Pac system) with the option MAPALGN=NO.

It must include the option SPOOL=YES for the submission of batch jobs (GP choice).

The Data (AR), Index (AN), Journal (AJ), Generation-Printing Requests (AG) as well as the Error Messages (AE) and User Parameter (AP) files are updated on-line and must be protected by the 'DYNAMIC BACKOUT' option.

The same applies for the on-line production environment file (AC) of the PEI function.

The maximum number of strings to be used per file is two. This number can thus be adjusted without restriction according to the available space in the CICS partition. (See Subchapter 'Efficiency Enhancement.)

The use of the CICS 'EMERGENCY RESTART' option is recommended.

The size of the COMMAREA in use is 3.5 K.

The average size of an on-line program is 48K, the largest being 115K.

GENERAL INFORMATION - HOW THE SYSTEM RUNS

The general characteristics are:

VISUALAGE PACBASE TRANSACTION:

A single transaction code is used. It points to a 'monitor' program whose purpose is to chain together the different programs that make up the system. No screen is directly driven by this program. (Program code: R000).

. Each screen that can be used for updating is associated with a given program (for example: QC00 updates the definition of a data element).

. Screens that display the 'simple' lists are processed in a single program (QH01).

. The so-called 'special' lists, such as the cross-references, keywords, etc., are processed in specific programs (QS02, QS03, QS04 and QS05).

. Menus are processed in QS06.

. Some programs are called by the LINK command. This is the case for the program that processes the CHOICE field (R100), as well as for the program that validates the data element formats (R200).

. In case of a system ABEND, an 'ABEND MAP' is generated. The display program for this map (R500) is called by XCTL.

. Updates are serialized; in other words, the system protects the database against concurrent accesses by placing all update transactions in a wait queue (ENQUEUE and DEQUEUE in the update cycle).

A specific program is called when the user quits the VA Pac transaction (program code: R005). This program displays the message:

CURRENT PACBASE CONVERSATION IS SAVED

when users leave the transaction using the CHOICE .12 or the corresponding PFkey (standard : PF12), and executes an "EXEC CICS RETURN".

In order to avoid giving control to CICS, the R005 program may be replaced by a user-program which will, for instance, call a user transaction.

VA Pac sends a 3-character COMMAREA at the end of the transaction:

position 1-2: Blank or 12 (conversation back-up)
position 3 : 'A' or 'F': language code (install. parameter)

The program replacing R005 may use the COMMAREA, and display the message 'CURRENT CONVERSATION SAVED' when the first two positions are '12' and the third is 'A'.

USER PARAMETER AND PEI TRANSACTION:

A specific transaction code is used to:

. Manage user parameters (standard),

. Manage the production environment (available with the PEI function).

UPPERCASE AND LOWERCASE MANAGEMENT

The VisualAge Pacbase system has its own input character management system:

. All codes entered in lowercase are automatically transformed into uppercase,

. All entity names, as well as text, remain in lowercase. Implicit keywords drawn from entity names are changed into uppercase words.

In order to inhibit the lowercase to uppercase switch, the letter 'X' should be entered in the ACTION CODE field.

If active, the UCTRAN option of the terminal is automatically inhibited by PACBASE (UCTRAN=NO) for the time of the connection, so that this control is available.

It is nevertheless possible to work in uppercase mode, by neutralizing the supplied standard routine. To do this, rename the xxUCTR program, then rename xxUCTX as xxUCTR.

In Batch mode, the character transformation follows the rules that apply to On-Line mode.

If you wish to work in uppercase mode only, use the functionalities of the editor that contain the PACBASE batch transactions, to ensure automatic switch from lowercase to uppercase, since VA Pac itself does not perform this transformation in batch mode.

3.3. ACCESS METHODS

ACCESS METHODS (VSAM)

The VA Pac system manages its files using the indexed VSAM- KSDS access method without a secondary index and the relative VSAM-RRDS access method.

All files are protected against concurrent write accesses (SHARE OPTION 2). This share option is the only protection of the VA Pac system against simultaneous updates, both in on-line and batch modes.

Production Environment Interface:

As the AB file may be updated in batch mode by simultaneous GPRT procedures, it is not protected (SHARE OPTION 4).

All batch procedures include DELETE/DEFINE steps to take care of file reloading. Therefore, the files do not need to include the REUSE default option.

3.4. BATCH ENVIRONMENT

TEMPORARY FILES

In batch mode, the system runs using both the standard functions of the operating system and the VSAM access method.

The internal work files of procedures are mostly of the SAM-ESDS type. They are defined and deleted for each utilization, except for those files specific to the GPRT procedure, whose 'envelope' (i.e. the entry of the VSAM catalog) is permanent. Their size remains null while they are not in use.

PARTITION SIZE

The memory size necessary to the execution of the batch procedures varies mostly by the size of the bufers allocated to the files used by those procedures.

Three parameters are of particular significance:

- The size of the EXEC Card, given in the parameter SIZE, which relates to the loading area of the executable load- modules.
- The size of the GETVIS AREA, used especially by VSAM for buffers.
- The total size of the partition, defined for the system, and which is the sum of the two preceding areas.

All procedures, except those described below, can be executed in a 512K partition.

PROC	PROGRAM	SIZE	GETVIS	COMMENTS
REOR	PTU220	870K	250K	
GPRT	PACE80	425K	350K	On-line generator
	PAFP10	520K	50K	PAF user pre-compilation program
SASN	PTU140	560K	150K	
9INS	PACDTP	910K	150K	

3.5. TAPE USAGE

TAPE USAGE

Files that are likely to be large can be re-directed onto tapes.

In procedures, the SYSnnn logical unit associated to the file(s) --which must be assigned to a tape drive-- is specified by a comment.

For files re-directed onto a tape, the maximum blocking factor is used. Therefore, files with smaller blocking factors are automatically accepted.

3.6. FILE SIZE

FILE SIZE

The total amount of space needed for the files depends on the size of the applications managed by the system.

The following table indicates the disk space necessary for a standard installation:

VisualAge Pacbase installation size	Approximation in millions of bytes		
	TOTAL	VSAM	Non-VSAM
Installation total	55	24	31
System total	35	7	28
Installation test user file total	20	17	3

The size of the VA Pac system files remains fixed.

The size of the user files is explained on the next page. It is possible to get a rough estimate of the space to be allocated by allowing 400 bytes per VA Pac data, all files included.

The total amount of space needed for the EVOLVING (USER) FILES can be calculated by assuming the following:

Let NPAC be the number of VA Pac records, all libraries and sessions included. Then the following applies:

(AR) Data File: NPAC records of 140 bytes.

(AN) Index File: Approx. 3 * NPAC records of 54 bytes (on average, an element is used 3 times), plus the free space (FSPC) added when defining the file.

(AG) Generation-Printing Request File: Generally takes up little space. As an estimate, it should be able to contain approximately a hundred or so requests per user accessing the system (150 bytes).

(AJ) Journal File: It must contain enough space for all update transactions, in batch or on-line mode, that may pass between 2 re-initializations of the Journal file. A VA Pac transaction corresponds to one record of the Journal file (167 bytes).

(AP) User Parameter File: Takes up minimal space. It contains the site JCL lines and optional modifications of standard error messages (80 bytes per record).

CALCULATION OF VSAM FILE SIZE

```
KSDS file:
No. of Rec./CI = (CISIZE DATA - 10 - Free space CI) / RECSZ

RRDS file:
No. of Rec./CI = (CISIZE - 4) / (RECSZ + 3)
```

<u>Example</u>

For a database containing 1,000,000 data records, there are:

AN file: DATA CISIZE(4096) RECSZ(54) FSPC(10,6)

```
   .CI free space = 10 % of 4096 = 409
   .No. of records per CI = (4096 - 10 - 409) / 54 = 71
   .No. of CI's per 3380 disk track = 10
    (See 'VSAM Administration Guide' IBM GC26-4066-1)
   .Maximum No. of CI's/CA (1 CA = one 3380 cylinder) =
                                              10 * 15 = 150
   .CA free space = 6 % of 150 = 9
   .No. of CI's used for each CA = 150 - 9 = 141
```

Thus, for 3,000,000 data records (3 * NPAC), there are about:

```
      3,000,000 / 71 = 42,254 CI's,
        42,254 / 141 = 300 DATA CA's,
```

which represents about 300 cylinders of a 3380 disk.

AR file: CISIZE(2048) RECSZ(140)

```
   .No. of records per CI = (2048 - 4) / (140 + 3) = 14
   .No. of CI's per 3380 disk track = 18
    (See 'VSAM Administration Guide' IBM GC26-4066-1)
   .No. of CI's/CA (1 CA = one 3380 cylinder) =
                                              18 * 15 = 270
```

Thus, for 1,000,000 data records (NPAC), there are about:

```
      1,000,000 / 14 = 71,429 CI's,
        71,429 / 270 = 265 CA's,
```

which represents about 265 cylinders of a 3380 disk.

The total represents about 400 million bytes for the two most important evolving files of the database.

PRODUCTION ENVIRONMENT INTERFACE OPTION (PEI)

The AB and AC files contain the same data. Therefore they should have the same amount of space, calculated as follows:

.record length: 110
.number of load-module libraries: NLM
.number of entities in production: NEP
.number of generated entities per load-module library: NEG

The number of storage bytes needed should be:

110 * (NLM + (2 * NEP) + NEG).

3.7. EFFICIENCY ENHANCEMENT

EFFICIENCY ENHANCEMENT

The performance of the VisualAge Pacbase system may be improved in batch or on-line mode in several ways, depending on the environment and on the size of the managed VA Pac databases.

This chapter provides information necessary for efficiency enhancement.

The file, program and JCL installation default options may be modified according to the suggestions described below. It is also possible to enhance efficiency by modifying the system environment, which is not described in this chapter since it depends on the specifications of each site.

Moreover, it is recommended not to modify too many chosen default options, so that the VA Pac Database Manager can retrieve these modifications easily when reinstalling the system.

3.7.1. INSTALLATION OF THE INDEX AND DATA FILES

INSTALLATION OF THE INDEX (AN) AND DATA (AR) FILES

Where possible, the AN and AR files should be copied onto separate volumes.

For large VA Pac Databases (more than one million data items), it may also be appropriate to allocate the AN and AR files onto several volumes.

Since the AN file is used in its entirety by the system, it is possible to carry out a multi-volume installation simply by modifying the file's DEFINE in the DF$PRO.$PFI.AN SLI member.

Moreover, about 80 percent of the VA Pac activity use the data stored at the beginning of the AR file, in the case of normal operations carried out by a development team. Therefore, the allocation of the AR file on several volumes does not in itself provide access optimization.

In order to take advantage of this multi-volume distribution, a standard procedure, called Storage Optimization of Multi-Volume Data (STOP) is provided. It allows for a wide distribution of the most frequently used data within the whole file.

The STOP procedure, combined with the DEFINE of the AR file (DF$PRO.$PFI.AR SLI member) on several volumes, must therefore be used before restoration of the database.

(For a description of the STOP procedure and details concerning data distribution parameterization, see the Batch Procedures: Administrator's Guide.)

3.7.2. VISUALAGE PACBASE STRUCTURE UNDER CICS

VISUALAGE PACBASE STRUCTURE UNDER CICS

As a general rule, each VA Pac line type is processed by a specific program in the PPT. There are two types of programs:

- Programs that can update the database,

- Programs that can only read the database.

There may be several programs which correspond to one VA Pac transaction:

The first loaded program, called a 'monitor', sends control through XCTL to the program able to process the type of line displayed on the screen.

This program executes the following instructions:

. Screen read (RECEIVE).

. If an update is detected:

- Enqueue execution on a character string specific to the update database,

- Read update of the first record on the AR file,

- Rewrite of this record,

- Loop on requested updates, which may contain per updated line:

. Several positionings or sequential reads on the AN file (STARTBR, RESETBR, or READNEXT),
. Several simple reads on the AR file (READ),
. One or two write commands on the AR file (WRITE or READ UPDATE, REWRITE),
. Several write commands on the AN file (WRITE or READ UPDATE, REWRITE),
. A write command on the AJ file (WRITE),

- READ UPDATE of first record in the AR file,

- REWRITE of this record,

- DEQUEUE execution corresponding to the ENQUEUE.

. If the CHOICE field is entered: LINK to the CHOICE field decoding program.

. If the CHOICE is valid: XCTL on the corresponding program.

. Display step broken down as follows:

- Several positionings or sequential reads on the AN file (STARTBR, RESETBR, or READNEXT),

- Several simple reads on the AR file (READ).

. SEND (send screen).

. RETURN TRANSID (end of transaction).

NOTES

ENQUEUE/DEQUEUE operates in such a way that updates are serialized.

The program structure is such that, at a given time, only one BROWSE or READ UPDATE is active on the AN file; only one string is used by a transaction during a time lapse greater than one input/output.

3.7.3. RESOURCES REQUIRED UNDER CICS

RESOURCES REQUIRED UNDER CICS

In order to improve the VA Pac system efficiency under CICS, two objectives should be pursued:

- Reduce program loading time,
- Make VSAM buffer management more efficient.

1. Program management

The first solution is fairly easy to implement: it requires a frequency survey on program runs. As a result, the programs that are most often used are said to be resident in the CICS partition (RESIDENT(YES) in CSD).

Example

Here are the programs most commonly used by the VA Pac system:

xxR000		Monitor
xxR100		CHOICE field decoding
xxR200		Format validation
xxQ300	xxN300	Update of segment -CE lines
xxQT00	xxNT00	Update of text -D lines
xxQP00	xxNP00	Update of -P lines
xxQI00	xxNI00	Update of dialogue -CE lines
xxQI01	xxNI01	-
xxQI02	xxNI02	-
xxQH20	xxNH20	Update of dialogue -CS lines.

2. Buffer management

Three parameters associated with the AN and AR files must be entered in CSD: STRINGS, DATABUFFERS, and INDEXBUFFERS.

- STRINGS:

It specifies the number of STRINGS or requests which can be executed in parallel by VSAM on the same field: usually, a STRING is used during an input/output on disk.

Certain commands, however, do not follow this rule:

READ UPDATE and STARTBR or RESETBR reserve a STRING up to the REWRITE or UNLOCK and ENDBR commands, respectively.

- DATABUFFERS:

It specifies the number of DATA buffers to be allocated by VSAM when the file is opened. A minimum of two buffers is required: one for the read of control interval DATA, one for the control interval 'split'.

Default: DATABUFFERS = STRINGS + 1

- INDEXBUFFERS:

It specifies the number of INDEX buffers to be allocated by VSAM when the file is opened. The minimum number of buffers per request is one. It is preferable, however, to increase this number because for a file with three INDEX levels, three EXCP's are required to point to the searched CI DATA for each on-line request.

It is recommended to allocate:

AT LEAST: one buffer per INDEX level, plus one buffer per STRING (STRING value).

NO MORE THAN: one buffer per CI of INDEX SET, plus one buffer per STRING.

Conclusion

A sufficient number of buffers should be provided for, according to the number of STRINGS.

However, too many buffers may lead to an excessive pagination, and therefore EXCP's.

Below is a chart of optimized values for the different parameters, according to the number of STRINGS:

NUMBER OF STRINGS	AN FILE			AR FILE	
	STRINGS	INDEX BUFFERS	DATA BUFFERS	STRINGS	DATA BUFFERS
1	1	4	4	1	2
5	5	8	7	5	6
10	10	13	11	10	11
20	20	23	21	20	21
30	30	33	31	30	31
40	40	43	41	40	41

The remaining question is how to evaluate the number of concurrent requests at a given time.

3.7.4. EVALUATION OF THE NUMBER OF REQUESTS UNDER CICS

EVALUATION OF THE NUMBER OF REQUESTS UNDER CICS

If the number of requests is estimated by multiplying the number of strings necessary for a transaction by the maximum number of users, the resulting number of buffers would be too large, which would thus imply excessive memory requirements.

A preferable method is the probability approach, taking into account simple hypotheses which are relevant in analyzing how resources are managed under CICS.

Request number evaluation method

Transaction execution requests are entered according to the "POISSON distribution" law and the transaction's duration is a constant. Taking into account this duration and the frequency of requests, the number 'n' is determined so that the probability of having more than n transactions at a given time is less than 1/100,000.

The average duration of a transaction at a specific site has to be calculated, as well as the average frequency of requests. The following chart aids in finding this number 'n' based on these parameters.

This number equals the number of STRINGS indicated in the preceding chart, which allows the STRINGS, DATABUFFERS and/or INDEXBUFFERS parameters for the AN and AR files.

Time is expressed in seconds, average frequencies vary from 0.1 to 2, the duration of requests therefore varies from 10 to 0.5 seconds.

Refer to the table on the following page.

DURAT. FREQU.	0.1	0.5	0.9	1.3	1.7	2.1	2.5	2.9	3.3	3.7	4.1	4.5	4.9
0.1	2	3	4	4	5	5	5	5	6	6	6	6	6
0.2	3	4	5	5	6	6	7	7	7	8	8	8	8
0.3	3	4	5	6	7	7	8	8	9	9	9	10	10
0.4	3	5	6	7	7	8	9	9	10	10	11	11	12
0.5	3	5	6	7	8	9	9	10	11	11	12	12	13
0.6	3	5	7	8	9	9	10	11	12	12	13	14	14
0.7	3	6	7	8	9	10	11	12	13	13	14	15	15
0.8	4	6	7	9	10	11	12	13	13	14	15	16	16
0.9	4	6	8	9	10	11	12	13	14	15	16	17	17
1.0	4	7	8	10	11	12	13	14	15	16	17	18	19
1.1	4	7	9	10	11	13	14	15	16	17	18	19	20
1.2	4	7	9	10	12	13	14	15	16	18	19	20	21
1.3	4	7	9	11	12	14	15	16	17	18	19	20	22
1.4	4	7	9	11	13	14	15	17	18	19	20	21	22
1.5	4	8	10	12	13	15	16	17	19	20	21	22	23
1.6	4	8	10	12	14	15	17	18	19	21	22	23	24
1.7	5	8	10	12	14	16	17	19	20	21	23	24	25
1.8	5	8	11	13	14	16	18	19	21	22	23	25	26
1.9	5	8	11	13	15	17	18	20	21	23	24	26	27
2.0	5	9	11	13	15	17	19	20	22	24	25	26	28

3.8. ABNORMAL EXECUTION

ABNORMAL EXECUTIONS

A batch program execution may terminate abnormally. For example, input-output errors on the system files or on the database cause a forced abnormal end with a USER ABEND, accompanied by a message on the SYSLST file.

When an ABEND occurs, the user must first find this message. It is displayed in the following manner:

```
PROGR : pppppp   INPUT-OUTPUT ERROR : FILE ff   OP: oo
STATUS : nn
END OF RUN DUE TO PROVOKED ABEND
```

In most cases, examining the status and type of operation allows the user to find the cause of the abnormal end.

The following table contains the most common values for the status (STATUS) and type of operation (OP).

nn	STATUS
21	SEQUENCE ERROR
22	DUPLICATE KEY
23	NO RECORD FOUND
24	BOUNDARY VIOLATION (KSDS-RRDS)
30	SYSTEM ERROR
30	SYSTEM ERROR
34	BOUNDARY VIOLATION(SEQUENTIAL)
92	LOGIC ERROR (FOR EXAMPLE, OPENING AN ALREADY OPEN FILE)
93	FILE STILL OPEN UNDER CICS
95	INVALID OR INCOMPLETE FILE INFORMATION

oo	OPERATION
W	WRITE
RW	REWRITE
RU	READ UPD
OP	OPEN
CL	CLOSE
D	DELETE
R	READ
P	START
RN	READ NEXT

If this message is absent and the type of ABEND generated directly signals a problem in the PACBASE system programs, it will be necessary to contact the PACBASE technical team at IBM. Be sure to KEEP ALL LISTINGS that may be necessary to analyze the problem, for example:

-The report of the execution with // OPTION LOG,PARTDUMP
-The list of installed programs and files (Job 9INS)
-etc.

PROCEDURE RESUMING AFTER AN ABEND

If a procedure terminates for one of the following reasons:

- output file too small,
- partition too small to open a file,
- file open in another partition, or
- any error not specifically associated with processing programs or processed data,

after correcting the error, it is possible to restart the procedure from the point where it terminated.

This option saves processing time, but must be executed with care.

The JCL line '/* GOTO JCLSTxx' must be activated, without change of location, with the label of the step which produced the abend.

The following cautions must be observed:

- If the physical names of the temporary files are prefixed by the '%' character, the retrieval job must be submitted in the same partition as the initial job; or, the names of the temporary files must be substituted in the retrieval job, with the deletion of the '%' prefix and the addition of a suffix '.BG', '.F2', '.F3', etc. depending on whether the initial job was submitted by BG, F2, F3, etc.

- If the abend is due to an output file that is too small, make sure that a redefinition of the file won't affect the temporary files in the queue.

These files can be determined by referring to the order of execution in the description of the procedure and locating:

- the input files of the step where the abend occurred, and

- the input files of one of the subsequent steps, if they are not ouput from any of the steps to be resubmitted.

4. INSTALLATION

4.1. PRELIMINARY NOTES

PRELIMINARY NOTES

NEW FEATURES AND MODIFICATIONS

To facilitate the installation of this Release of VA PAC from Release 7.3, 8.0 or 8.0.1, the list below highlights the major new features and modifications. These fall into two major categories:

```
1. Preparation:  JCL organization
2. Environment:  Files modified, added, etc.
```

1. PREPARATION

- The VSAM temporary files use the 'NOALLOCATION' option, which permits streamlining the JCL and putting the DELETE/DEFINEs at the beginning of the procedure.

- Sending a return code to the supervisor via the COBOL programs makes it possible to manage the return code through the JCL and to merge procedures that executed separately before (PARM and PAR2, MLIB and MLIS, etc.)

- Using the POWER procedures (SLI members) provides for: 1) no redundant IDCAMS request cards for DEFINE CLUSTERs on database files that are now centralized in SLI members; and 2) cataloging cumbersome procedures, such as GPRT. A procedure called 'beginning of standard job' has been put into all JCL in order to allow for storage of site-specific standards in one place.

- The VSE procedures are used to define work space (primarily when sorting and, in rare cases, when VSAM work files are not adapted). Implementing VSAM SPACE MANAGEMENT for sortwork files no longer poses any problems.

- To facilitate back-up file management and to select the number of generations reserved for each back-up, the IDCAMS request cards that execute the generation rotation are centralized in SLI members.

2. ENVIRONMENT

- Two user files are new: PA (on-line PAF work file) and GS (PAF-PDM Extraction Master Paths file).

- There is a new 'envelop' for the work file, through a batch partition (batch PAF work file).

- Three system files are new: SS, SF, and SP (generation skeleton files).

- The JCL for submitting POWER must be ready in order to be taken into account with the SLI procedures.

- The VSE variant (system releases prior to VSE/SP 2) is no longer supported.

- The initial parameterization (MM1JCL) makes it possible to distribute the contents of the index and data components, the user and system files, the back-up and operations files, and the permanent and temporary files onto several different disk volumes.

4.2. INTRODUCTION

INTRODUCTION

The installation procedure is broken down into three main stages:

. Preparation for installation,
. Installation,
. On-line and batch tests.

The installation process requires a special tape (or cartridge) and is described in this chapter.
You should be familiar with the VA Pac System's characteristics, which are fully described in this manual, in order to properly prepare an environment for the installation procedure (disk space, CICS options, codification, etc.).

Once the environment is ready, you can proceed with the installation.

PREPARATION

Unload the initial JCL from the tape, and execute it:

. Perform a backup of the installation tape,
. Copy the module for processing the complete JCL,
. Restore the complete VA Pac installation and operation JCL.

INSTALLATION

See Subchapter 'Installation Process'.

TESTS

. On-line procedure tests,
. Batch procedure tests.

4.3. INSTALLATION TAPE

INSTALLATION TAPE

The installation tape (6,250 BPI, no label) includes the following files:

RANK	FORMAT	LRECL	BLKSZ	CONTENTS
1	Fixed	80	3440	Initial preparation JCL
2	Fixed	80	80	Load-module of the MM1JCL installation utility
3	Fixed	80	3440	Complete installation and operation JCL
4	Fixed	3204	32040	Batch skeleton file
5	Fixed	4605	32235	OLSD and DBD skeleton file
6	Fixed	4605	32235	COBOL generator skeleton file
7	Fixed	80	32720	PACBASE error messages
8	Var.	155	32760	Test database backup
9	Fixed	4605	23025	PAF-PDM Extension generation skeleton file
10	Fixed	119	11900	PAF-PDM Extension generation skeleton file
11	Fixed	4605	32235	C/S skeleton file
12	Fixed	3204	32040	Batch skeleton file (French)
13	Fixed	4605	32235	OLSD and DBD skeleton file (F)
14	Fixed	4605	32235	COBOL skeleton file (French)
15	Var.	155	32760	Test database (French version)

RANK	FORMAT	LRECL	BLKSIZE	CONTENTS
16	Fixed	80	32720	Initial AEO, bilingual version
17	Fixed	4605	32235	C/S skeleton file (French)
18	Fixed.	80	32720	SQL program source
19	Fixed.	117	7020	Visualage Dictionary
20	Fixed.	117	7020	TeamConnection Dictionary
21	Undef.		-----	Empty file (Backup header)
22	Undef.		17000	Backup file-id
23	Undef.		17000	Backup file
24	Undef.		-----	Empty file (Backup trailer)
25	Fixed	80	80	End-of-backup record

4.4. UNLOADING OF INITIAL JCL

UNLOADING OF INITIAL JCL

The first file on the installation tape, which contains the initial JCL, must be unloaded onto the on-site editor (LRECL 80 BLKSIZE 80).

```
* $$ JOB JNM=$JOB,CLASS=$CLASS,DISP=$DISP,USER='$USER'
* $$ LST CLASS=$PCLO,DISP=$PDIO
* $$ PUN CLASS=$PCLP,DISP=$PDIP,DEST=$DEST
// JOB $JOB    UNLOAD PACBASE INITIAL JCL
// ASSGN SYS004,TAPE
// PAUSE MOUNT PACBASE INSTALLATION TAPE ON SYS004
// UPSI 1
// EXEC DITTO
$$DITTO REW OUTPUT=SYS004
$$DITTO TC INPUT=SYS004,NFILES=1
$$DITTO RUN OUTPUT=SYS004
$$DITTO EOJ
/*
/&
* $$ EOJ
```

4.5. INITIAL JCL

INITIAL JCL

The function of the initial JCL is to generate the installation and operation JCL of the System.

This initial JCL is a parameterized skeleton which allows customization; it is processed by the MM1JCL utility program, which generates JCL that will be adapted according to the parameters specified by the user.

The unloading and execution of the MM1JCL utility is contained in the initial JCL, which includes three jobs:

.JOB0 Copies the installation tape provided by IBM onto a user tape; the latter is considered as the System back-up and must be used for the actual installation.

.JOB1 Unloads the MM1JCL and PACDEB utilities by LNKEDT. (through the RESTORE and LIBR functions).

.JOB2 Executes the MM1JCL utility, which creates the actual installation and operation JCL.

The MM1JCL output must be kept, since it may be used for reinstallations.

```
/* --------------------------------------------------------
/* I N S T A L L A T I O N  O F  I N I T I A L      J C L
/* --------------------------------------------------------
/*
/* THIS JCL CONTAINS THREE JOBS :
/*    - COPY OF THE REFERENCE TAPE ON AN INSTALLATION
/*      TAPE,
/*    - RECEPTION OF THE INSTALLATION UTILITY IN A USER
/*      LIBRARY,
/*    - RECEPTION OF INSTALLATION AND OPERATION
/*      JCL UNDER TEXT EDITOR.
/*
/* THE VARIABLE ELEMENTS USED IN THESE THREE JOBS ARE :
/*    - A VSAM CATALOG FOR WORK FILES,
/*    - A LIBRARY FOR THE UTILITY,
/*    - THE POWER SPOOL FOR THE OUTPUT PUNCHED JCL,
/*    - A USER CODE FOR POWER,
/*    - TWO TAPE NAMES,
/*    - TWO TAPE UNITS ADRESSES.
/*
/* -------- REQUIRED SUBSTITUTIONS --------------------
/*
/* %USER = JOB IDENTIFIER (4 CHAR)
/* %ACCT = POWER ACCOUNTING INFORMATION (USER='%ACCT')
/*
/* %CATN = IDENTIFICATION OF VSAM CATALOG
/* %CATV = VSAM SPACE VOLUME SERIAL NUMBER
/*
/* %LIBN = IDENTIFICATION OF A LIBRARY FOR THE UTILITY
/* %LIB  = LIB FOR UTILITY
/* %SLIB = SUBLIB FOR UTILITY
/*
/* %PLST = PARAMETERS  * $$ LST
/* %PPCH = PARAMETERS  * $$ PUN  FOR RECEPTION UNDER TEXT EDITOR
/*
/* %INST = VOLUME SERIAL NUMBER OF REFERENCE TAPE
/* %BCKP = VOLUME SERIAL NUMBER OF BACK-UP TAPE
/*
/* %TAP1 = FIRST TAPE DRIVE ADDRESS
/* %TAP2 = SECOND TAPE DRIVE ADDRESS




/* *************************************************************
*         COPY OF THE INSTALLATION TAPE
/* *************************************************************
* $$ JOB JNM=%USERJOB0,CLASS=0,DISP=D,USER='%ACCT'
* $$ LST %PLST
// JOB JOB0         COPY OF THE INSTALLATION TAPE
// PAUSE MOUNT  : %INST,%TAP1,READ    %BCKP,%TAP2,WRITE
// UPSI 1
// EXEC DITTO
$$DITTO REW OUTPUT=%TAP1
$$DITTO REW OUTPUT=%TAP2
$$DITTO TT INPUT=%TAP1,OUTPUT=%TAP2,NFILES=24
$$DITTO RUN OUTPUT=%TAP1
$$DITTO EOJ
/*
/&
* $$ EOJ




* $$ JOB JNM=%USERJOB1,CLASS=0,DISP=D,USER='%ACCT'
* $$ LST %PLST
// JOB JOB1        COPY OF THE UTILITY
/* *************************************************************
*  *     LOADING OF "MM1JCL" UTILITY
/* *     IN THE PREDEFINED USER LIBRARY
/* *************************************************************
// PAUSE MOUNT THE INSTALLATION TAPE %BCKP ON %TAP2
```

```
 MTC REW,%TAP2
 MTC FSF,%TAP2,1
// ASSGN SYSIPT,%TAP2
// DLBL %LIB,'%LIBN',,VSAM
// LIBDEF PHASE,CATALOG=%LIB.%SLIB
// OPTION CATAL
 INCLUDE
// EXEC LNKEDT
/*
/&
* $$ EOJ




* $$ JOB JNM=%USERJOB2,CLASS=0,DISP=D,USER='%ACCT'
* $$ LST %PLST
* $$ PUN %PPCH
// JOB JOB2        CREATION OF INSTALLATION JCL
/* ************************************************************
*   --- CREATION OF THE INSTALLATION JCL BY THE MM1JCL UTILITY
/* ************************************************************
/*
/*  MODIFY THE PROVIDED LIST OF COMMANDS SPECIFYING THE INSTAL-
/*  LATION VARIANTS (IF ANY) BY REQUESTING, IF NECESSARY, THE
/*  SELECTION OF PORTIONS OF THE INSTALLATION JCL (JCL MODULES):
/*  ASSIGN THE APPROPRIATE VALUES TO THE INSTALLATION PARAMETERS
/*  AND, IF NEEDED, SPECIFY THE LINES TO BE ADDED AT THE BEGIN-
/*  NING AND THE END OF THE MODULES.
/*
/* ************************************************************
* --- STEP 1 --- IDCAMS DEFINE
// DLBL IJSYSUC,'%CATN',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  DEL            (PTR.IMPL.PAC7GC) CL
  DEF CL   (NAME (PTR.IMPL.PAC7GC) -
           NAL RUS NIXD CISZ(4096) -
           RECFM(FB(104)) -
           VOL (%CATV) RECSZ (3432 3432) TRK (10 5) )
  DEL            (PTR.IMPL.PAC7GI) CL
  DEF CL   (NAME (PTR.IMPL.PAC7GI) -
           NAL RUS NIXD CISZ(4096) -
           RECFM(FB(104)) -
           VOL (%CATV) RECSZ (3432 3432) TRK (50 10) )
  DEL            (PTR.IMPL.PAC7LA) CL
  DEF CL   (NAME (PTR.IMPL.PAC7LA) -
           NAL RUS NIXD CISZ(4096) -
           RECFM(FB(137)) -
           VOL (%CATV) RECSZ (1370 1370) TRK (5 1) )
  DEL            (PTR.IMPL.PAC7LI) CL
  DEF CL   (NAME (PTR.IMPL.PAC7LI) -
           NAL RUS NIXD CISZ(4096) -
           RECFM(FB(137)) -
           VOL (%CATV) RECSZ (1370 1370) TRK (5 1) )
/*
* --- STEP 2 --- MM1JCL
// ASSGN SYS004,%TAP2
// PAUSE MOUNT THE INSTALLATION TAPE %BCKP ON %TAP2
// MTC REW,SYS004
// MTC FSF,SYS004,2
// ASSGN SYS005,SYSLST
// ASSGN SYS010,SYSIPT
// ASSGN SYS011,SYSPCH
// DLBL IJSYSUC,'%CATN',,VSAM
// DLBL %LIB,'%LIBN',,VSAM
// LIBDEF PHASE,SEARCH=%LIB.%SLIB
// DLBL WK1,'PTR.IMPL.PAC7GC',,VSAM
// DLBL WK2,'PTR.IMPL.PAC7GI',,VSAM
// DLBL LBI,'PTR.IMPL.PAC7LA',,VSAM
// DLBL LBO,'PTR.IMPL.PAC7LI',,VSAM
/*   THE GENERATED JCL IS PRODUCED AS PUNCHED OUTPUT
/*   THIS JCL IS TO BE RECEIVED UNDER A TEXT EDITOR
// EXEC MM1JCL,SIZE=AUTO
```

```
 *                                                              A
===IDENC CLIENT 'ANY CLIENT',SITE 'PROTOTYPE JCL'
===SELV MER              .WORKSTATION:  MERISE METHODOLOGY
===SELV YSM              .WORKSTATION:  YSM    METHODOLOGY
===SELV SSA              .WORKSTATION:  SSADM  METHODOLOGY
===SELV IFW              .WORKSTATION:  IFW    METHODOLOGY
===SELV OMT              .WORKSTATION:  OMT    METHODOLOGY
===SELV PEI              .PACBASE PEI OPTION
===SELV PACTR            .PACBASE PAC/TRANSFER
===SELV PUF              .PACBASE PUFTP
===SELV SQL              .PACBASE SQL
===SELV S2K              .PACBASE Pac/Impact
===SELV TAB              .PACBASE TABLES
===SELV DSM              .PACBASE DSMS
===SELV TEAM             .PACBASE TEAMCONNECTION BRIDGE
===SELV VISP             .PACBASE VISUALAGE SMALLTALK BRIDGE
===SELV ROLD             .RETRIEVAL:  OLDER PACBASE
===PRM PCJ=0             .EXECUTION CLASS      (JOB CARD)
===PRM PCL=X             .OUTPUT CLASS         (LST CARD)
===PRM PCP=X             .OUTPUT CLASS         (PUN CARD)
===PRM PDJ=D             .JOBS DISPOSITION     (JOB CARD)
===PRM PDL=H             .OUTPUT DISPOSITION   (LST CARD)
===PRM PDP=H             .OUTPUT DISPOSITION   (PUN CARD)
===PRM LST2=60C          .PRINTER 2 (POWER LST=--- PARAMETER)
===PRM LST3=60D          .PRINTER 3 (POWER LST=--- PARAMETER)
===PRM PRFJ=%USER        .JOB NAME PREFIXES    (4 CHARACTERS)
===PRM PUJ='%ACCT'       .USER PARAMETER       (JOB CARD)
===PRM PWJ=''            .POWER JOB CARD ADDITIONAL PARAMETERS
===PRM PWL=',RBS=250'    .POWER LST CARD ADDITIONAL PARAMETERS
===PRM PWP=''            .POWER PUN CARD ADDITIONAL PARAMETERS
===PRM GROUP='PACB'      .CICS/RDO: PACBASE RESOURCE GROUP NAME
===PRM LIST='LCGI'       .CICS/RDO: PACBASE RESOURCE LIST
===PRM NLSR=1            .LSR POOL NUMBER
===PRM PFI=25            .NUMBER OF THE PACBASE USER DATABASE
===PRM PRO=PB            .ROOT OF PACBASE SYSTEM (2 CHARACTERS)
===PRM PXU='PAC250.USR'  .INDEX OF PACBASE USER FILES
===PRM PXY='PAC250.SYS'  .INDEX OF PACBASE SYSTEM FILES
===PRM TAPEI=%BCKP       .INSTALLATION TAPE NAME
===PRM TAP1=%TAP1        .TAPE DRIVE ADDRESS
===PRM TAP2=%TAP2        .ALTERNATIVE TAPE DRIVE ADDRESS
===PRM VOLVWK=%CATV      .TEMPORARY/DYNMA. FILES VOLUME
===PRM VWORK=%CATV       .TEMPORARY FILES VOLUME
===PRM XW=%WPAC          .WORK FILES INDEX
===PRM WKBG1='12875,400' .WORK AREA 1/BG (400 TRACKS 3380)
===PRM WKBG2='12475,400' .WORK AREA 2/BG
===PRM WKBG3='12075,400' .WORK AREA 3/BG
===PRM WKF21='0,0'       .WORK AREA 1/F2  - OPTIONAL
===PRM WKF22='0,0'       .WORK AREA 2/F2       "
===PRM WKF23='0,0'       .WORK AREA 3/F2       "
===PRM WKF31='0,0'       .WORK AREA 1/F3       "
===PRM WKF32='0,0'       .WORK AREA 2/F3       "
===PRM WKF33='0,0'       .WORK AREA 3/F3       "
===PRM WKF41='11675,400' .WORK AREA 1/F4       "
===PRM WKF42='11275,400' .WORK AREA 2/F4       "
===PRM WKF43='10875,400' .WORK AREA 3/F4       "
===PRM WKF51='10475,400' .WORK AREA 1/F5       "
===PRM WKF52='10075,400' .WORK AREA 2/F5       "
===PRM WKF53='9675,400'  .WORK AREA 3/F5       "
===PRM WKF61='0,0'       .WORK AREA 1/F6       "
===PRM WKF62='0,0'       .WORK AREA 2/F6       "
===PRM WKF63='0,0'       .WORK AREA 3/F6       "
===PRM WKF71='0,0'       .WORK AREA 1/F7       "
===PRM WKF72='0,0'       .WORK AREA 2/F7       "
===PRM WKF73='0,0'       .WORK AREA 3/F7       "
===PRM WKF81='0,0'       .WORK AREA 1/F8       "
===PRM WKF82='0,0'       .WORK AREA 2/F8       "
===PRM WKF83='0,0'       .WORK AREA 3/F8       "
===PRM WKF91='0,0'       .WORK AREA 1/F9       "
===PRM WKF92='0,0'       .WORK AREA 2/F9       "
===PRM WKF93='0,0'       .WORK AREA 3/F9       "
===PRM MODB='%LIBN'      .BATCH MODULES LIB PHYSICAL NAME
===PRM MODO='%LIBN'      .ONLINE MODULES LIB PHYSICAL NAME
===PRM SLIL='%LIBN'      .SLI MEMBERS LIB PHYSICAL NAME
```

```
===PRM PRLB='%LIBN'      .VSE PROCEDURES LIB PHYSICAL NAME
===PRM LMODB='%LIB'      .LIB / MODULES BATCH
===PRM LMODO='%LIB'      .LIB / MODULES TP
===PRM LSLI='%LIB'       .LIB / SLI MEMBER
===PRM LPRLB='%LIB'      .LIB / VSE PROCEDURES
===PRM SMODB='.%SLIB'    .SUBLIB / BATCH MODULES
===PRM SMODO='.%SLIB'    .SUBLIB / ON-LINE MODULES
===PRM SLIS='.%SLIB'     .SUBLIB / SLI MEMBER
===PRM SLIT='PAC'        .TYPE   / SLI MEMBER
===PRM SPRLB='.%SLIB'    .SUBLIB / VSE PROCEDURES
===PRM SORTWK='$$SORTWK' .DEFN SORTWORK SPACE PROCEDURE NAME
===PRM VCAP='%CATN'      .VSAM CATALOG FOR PACBASE FILES
===PRM OPT='LOG,NODUMP'  .'// OPTION' CARD PARAMETERS
===PRM RAJP=3000         .AJ PRIMARY ALLOCATION   (RECORDS)
===PRM RAJS=3000         .AJ SECONDARY ALLOCATION (RECORDS)
===PRM RPCP=20000        .PC PRIMARY ALLOCATION   (RECORDS)
===PRM RPCS=20000        .PC SECONDARY ALLOCATION (RECORDS)
===PRM VOLPSV=%CATV      .BACKUP/LOG FILES VOLUME
===PRM VOLPUD=%CATV      .USER FILES DATA COMPONENTS VOLUME
===PRM VOLPUX=%CATV      .USER FILES INDEX COMPONENTS VOLUME
===PRM VOLPYD=%CATV      .SYSTEM FILES DATA COMPONENTS VOLUME
===PRM VOLPYX=%CATV      .SYSTEM FILES INDEX COMPONENTS VOLUME
===PRM TRO=TA            .ROOT OF PACTABLE SYSTEM (2 CHARACTERS)
===PRM TXU='TAB250'      .INDEX OF PACTABLE USER FILES
===PRM VOLTUX=%CATV      .VOLUME SERIAL OF TABLES INDEX FILES
===PRM VOLTUD=%CATV      .VOLUME SERIAL OF TABLES DATA FILES
===PRM VCAT='%CATN'      .VSAM CATALOG FOR PACTABLE
===PRM VCAD='%CATN'      .VSAM CATALOG FOR DSMS
===PRM DSMS=<>           .FILE NAME OF DSMS FILE (DC)
===PRM MODELF='PAC250.MODELF' .SAM-ESDS FILE, FIXED MODEL
===PRM MODELV='PAC250.MODELV' .SAM-ESDS FILE, VARIABLE MODEL
===BEGMOD
./ ADD NAME=$PRO$PFI$ZMODUL
//%USER$ZMODUL JOB %ACCT,'VSE',CLASS=X,MSGCLASS=X,NOTIFY=%USER
/*XMIT A07POWER DLM=%%
===ENDMOD
%%
/*
* --- STEP 3 --- IDCAMS DELETE
// DLBL IJSYSUC,'%CATN',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  DEL    (PTR.IMPL.PAC7GC) CL
  DEL    (PTR.IMPL.PAC7GI) CL
  DEL    (PTR.IMPL.PAC7LA) CL
  DEL    (PTR.IMPL.PAC7LI) CL
/*
/&
* $$ EOJ
```

4.6. COMPLETE JCL INSTALLATION

COMPLETE JCL INSTALLATION

The MM1JCL module reads the JCL skeleton file and outputs an adapted JCL. It uses commands which allow the user to:

. Select portions of the JCL skeleton, which are called 'JCL modules',

. Parameterize the skeleton JCL in order to obtain a JCL requiring a minimum of modifications to make it operational.

. Select the installation variants in order to generate the JCL needed for specific processing based on site and installation conditions.

. Add lines before and after the JCL modules in order to separate them.

This step can be executed as many times as necessary to generate a completely adequate JCL jobstream.

The MM1JCL provides a list of the newly created JCL by module taking into account all parameters and requested variants.

Once the JCL is created, it may be retrieved under a text editor. At this point, the actual installation procedure can begin.

It is advisable to create in the text editor library as many members as there are JCL modules, each module being a logical step in the installation procedure.

Adding the proper cards before and after the modules will automatically split up the JCL. (Refer to the examples in Subchapter "JCL MODULE SEPARATORS").

FILES USED BY MM1JCL

```
   Input      : JCL skeleton          : SYS004 (tape)
                Utility requests      : SYS010 (reader)
   Output     : Generated JCL         : SYS011 (punch)
                List of generated JCL : SYS005 (printer)
   Work files : WK1 (300,000 bytes)
                LBI (Dummy)
```

MM1JCL INSTALLATION DEFAULT OPTIONS

```
. Variants   :  All.

. Parameters :  No default values, indicated values are
                examples.

. JCL modules:  No selection, all modules are copied.
```

CODING OF MM1JCL COMMANDS

```
===SELL lll               .Selection of installation
                           language:
                           lll = ENG (English)
                                 FRA (French)

===SELV vvvv              .Selection of variant
                           vvvv = variant code

===SELM mmmm1 mmmm2 ...   .Selection of JCL modules
                           mmmm1 = name of JCL module
                           mmmm2 = name of JCL module
                           etc.
                           The absence of a ===SELM line
                           involves the selection of all
                           JCL modules.

===PRM  PPPP=pppp         .Parameter
                           PPPP = name of parameter
                           pppp = value of parameter

NOTE: On ===PRM or ===SELV lines, comments may be entered.
      They should be preceded by a period, and not exceed
      column 72.

===BEGMOD         Insertion of lines at beginning of module.
....1         )
.....         ) Lines to be inserted before each module
....n         )

===ENDMOD         Insertion of lines at end of module.
....1         )
.....         ) Lines to be inserted after each module
....n         )
```

4.7. INSTALLATION VARIANTS

INSTALLATION VARIANTS

```
===SELV VVV                     .Comments
```

VVV	MEANING	COMMENTS
SQL	SQL Relational DBD Installation of the SQL catalog update COBOL source	VA Pac function
PEI	Production Environment Interface PEI file installation and recognition in VA Pac	VA Pac function
TAB	Pactables Installation of the VA Pac - Pactables interface components	VA Pac function
DSM	DSMS Installation of the VA Pac - DSMS interface components	VA Pac function
PACTR	Pac/Transfer	VA Pac function
PUF	PUFTP	VA Pac function
S2K	Pac/Impact	VA Pac function
VISP	VisualAge Smalltalk-VA Pac	VA Pac function
TEAM	Teamconnection-VA Pac	VA Pac function
ROLD	Retrieval of 8.0.x, 1.2 or 1.6 databases	Users of release 8.0.x to 1.6

4.8. OPTION: SQL INTERFACE (DB2)

SQL INTERFACE INSTALLATION

Environment

DB2 is not supported in VSE. Therefore, only the SQL interface is available.

Description of the program

A program calls, via a LINK, a COBOL module which contains the SQL accesses:

- xxQS10 generates the update SQL and calls the update module: xxRS12.

Installation

The xxQS10 program is shipped as an executable module, and installed with PACBASE standard programs (job '$prfj.2PHA' of the installation procedure).

The xxRS12 module is shipped as a COBOL source program, to be prepared and installed, and supported in DB2. To do this:

1. The source program must be installed using an editor, and the statements should be checked for consistency with the user's requirements (they should be customized if necessary).

   This source program should be handled as any standard SQL program, i.e.:

   - Preparation via a preprocessor, yielding:

     .a COBOL source program,
     .a DBRM program (same name as the COBOL source program)

   - Compilation and LINK-EDIT on the COBOL source program, yielding an executable program (xxRS12).

2. The plan is built via the DBRM program (BIND operation).

3. The RCT CICS table is updated with the VA Pac transaction code and the code of the plan built before.

NOTE:

Updates are performed via dynamic SQL statements.

COMPLEMENT: INSTALLATION OF THE SQL INTERACTIVE APPLICATION

Description of the programs

A program calls, via a LINK, a COBOL module containing the SQL statements. The module is called xxQL46. It generates the update SQL and calls the SQL update module: xxRS12.

Installation

The xxQL46 program is shipped as an executable module, and installed with the VA PAC standard programs (job '$prfj.2PHA' of the installation procedure).

The xxRS12 module is shipped as a COBOL source program for preparation and installation.

4.9. JCL MODULES

JCL MODULES

TABLE OF INSTALLATION JCL MODULES

The JCL modules are identified by a four-character code. The installation modules are identified by a numeric prefix.

The MM1JCL utility command '===SELM' is used to select the modules to unload. All the JCL modules are selected by default.

```
===SELM aaaa bbbb cccc
```

CODE	CONTENTS	NATURE
	*** INSTALLATION ***	
1CSD	CSD source: CICS tables PCT, PPT	CICS SYS
1PCT	Assembler source: transactions (PCT)	CICS SYS
1PPT	Assembler source: programs (PPT)	CICS SYS
1FCT	Assembler source: files (FCT)	CICS SYS
1VSE	Catalogue of VSE procedures	VSE JCL
1SLI	Catalogue of SLI members	VSE JCL
1PWR	POWER JCL submission	VSE JCL
2FIC	Definition of VSAM permanent files	VSE JCL
2GPR	Catalogue of GPRT JCL (SLI member)	VSE JCL
2PAC	Catalogue of PACX JCL (SLI member)	VSE JCL
2WGP	Definition of GPRT work files	VSE JCL
2WPA	Definition of PACX work files	VSE JCL
2PHA	Installation of executable modules	VSE JCL
3SYS	Initialization of system files	VSE JCL
3USR	Initialization of user files	VSE JCL
PJ16	Retrieval of archive journal 8.0.2 ->	VSE JCL
PP16	Retrieval of PEI environment 8.0.2 ->	VSE JCL
RPPG	Retrieval of generation-print commands	VSE JCL
PC80	Retrieval of database backup 8.0, 8.01	VSE JCL
PE80	Retrieval of user parameters 8.0, 8.01	VSE JCL
PJ80	Retrieval of archive journal 8.0, 8.01	VSE JCL

TABLE OF INSTALLATION JCL MODULES (CONT.)

CODE	CONTENTS	NATURE
5TPC	Installation of PACBASE test database	VSE JCL
6JCI	CICS JCL submission	VSE JCL
9INS	List of installed programs and files	VSE JCL
9TSG	Database management tests	SYSIN
9TSE	Tests for extraction utilities	SYSIN
9TSU	On-line system tests	text

9DSG	User entity transactions for the WorkStation, with variant:
	MER = Merise methodology
	SSA = SSADM methodology
	YSM = YSM methodology
	IFW = IFW methodology
	OMT = OMT methodology
		SYSIN

9DSP	Parameter setting for PACDESIGN, with variant:
	MER = Merise methodology
	SSA = SSADM methodology
	YSM = YSM methodology
	IFW = IFW methodology
	OMT = OMT methodology
		SYSIN

9INC	Example of optional control cards	VSE JCL
9PAF	PAF dictionary source	SYSIN
9PQU	Quality rules source	SYSIN
9PQR	Compiled quality rules	file
9XPD	User entity source (PAF-PDM extension)	SYSIN

TABLE OF OPERATION JCL MODULES

CODE	CONTENTS
ACTI	Journal statistics utility
ARCH	Journal archiving
CPSN	Sub-network comparison
CRYP	Encryption-decryption of passwords
CSES	Compression of session numbers
DCOB	Documented COBOL printing
EMLD	Load of user-defined error messages
EMSN	Extraction for sub-network merge
EMUP	Updating of user-defined error messages
ESES	Session number extraction
GETA	Generation of table descriptions
GETD	Generation of table descriptions
GETI	Table initialization
GET0	Table initialization
GET1	Generation of table description
GET2	Generation of table description
GPRT	Generation-print
GRPE	Transaction generation for database reorganization
HIPE	Automatic freeze
IANA	Impact analysis
IGRA	Group zone breakdown
INFO	Initialization FO
INFP	Initialization FP
INFQ	Initialization FQ
INPE	Back-up initialization
IPEP	Edition of search criteria
IPFQ	Edition of FQ file
IPIA	Edition of the results
ISEP	Selection of input points
ISOS	Selection of chains, operators
LOAE	User parameter restoration
LVBL	Replacement of low-values with blanks
MESN	Sub-network merge
MLIB	Library management
PACX	Extractor
PAFX	All-purpose user extractor
PARM	User parameters updating
PQCA	PACBENCH QUALITY CONTROL - Analysis
PQCE	PACBENCH QUALITY CONTROL - Extraction

TABLE OF OPERATION JCL MODULES (CONT.)

CODE	CONTENTS
PRGS	Master file edition
PRPE	PEI environment printouts
REAG	Reorganization of generation-print requests
REOR	Reorganization of the database
REST	Restoration of the database
RESY	Database system restoration via system util.
RMEN	Entity rename/move
RMTD	Post-migration restoration of TD
RSPE	PEI file restoration
RTLO	Retrieval of locked entities
SADM	Consistency check on SSADM entities
SASN	Sub-network backup
SASY	Logical database backup
SAVE	Database backup
SIPE	PEI simulation
SMTD	TD backup for migration
STOP	Storage optimization of data
SVAG	Generation-print requests backup
SVPE	PEI file backup
TCCI	TeamConnection Consistency check
TCGP	TeamConnection Parameters extension
TCLS	TeamConnection database update
TRDU	Production of DSMS transactions
TRJC	Compression of the journal file
TRPF	Generation of transfert transactions
TRRP	Extraction of the journal
TRUP	Updating parameters
UPDP	PACBASE database updating from PAF tables
UPDT	PACBASE database updating
UXSR	Extraction for sub-network updating
VDWN	Downloading VA Pac-VA Smalltalk
VINS	Updating C.G.I. user entities
VPUR	Dumping VisualAge Pacbase entities
VUP1	Uploading VisualAge Pacbase codes
VUP2	Uploading VisualAge Pacbase transactions
XPAF	Extraction master path validation
XPDM	Master outline validation
YSMC	Consistency check on YSM entities

TABLE OF SOURCE MODULES FOR CROSS GENERATORS:

CODE	CONTENTS
Z980	Contains the following modules:
ZARCVS	VS COBOL MVS/CICS, VS COBOL VSE/CICS
ZARCII	COBOL II MVS/CICS
ZARG7	GCOS7/TDS
ZARG8	GCOS8/DMIV and TP8
ZARICL	ICL
ZARBUR	Unisys A
ZARDEC	DEC (characters)
ZARDE2	DEC (fields)
ZARTRM	DEC (assembler)
SCRDEC	DEC sub-program
HPFORM	HP3000 screen message processing
ZARMF1	Microfocus
SCRCODIF	Microfocus sub-program
SCRIOPAR	Microfocus sub-program
SCRPEINT	Microfocus sub-program
SCRSAISI	Microfocus sub-program
ZARMFO	VisualAge Pacbase-reserved
SCRMFO	VisualAge Pacbase-reserved

4.10. JCL PARAMETERS

JCL PARAMETERS

NOTE: If a parameter's value includes special character(s), this value must be entered between quotes: (Example: '1000,600').

POWER CARD PARAMETERS

```
    +-------+--------------------------------------------+
    ! CODE  ! MEANING                                    !
    +-------+--------------------------------------------+
    !       !                                            !
    ! PRFJ  ! 4-character POWER job name prefix          !
$ JOB card         !
$ JOB card         !
$ JOB card         !
$ JOB card  !
    !       ! (Ex: PWJ = ',NTFY=USER1')                  !
$ LST cards        !
$ LST cards        !
$ LST card  !
    !       ! (Ex: PWL = ',RBS=100,DEST=RMTNODE')        !
$ PUN cards        !
    !       ! (for retrieval under the editor)           !
$ PUN card          !
    !       ! (for retrieval under the editor)           !
$ PUN card  !
    !       ! (for retrieval under the editor)           !
$ LST card       !
$ LST card       !
    +-------+--------------------------------------------+
```

PARAMETERIZATION OF VSAM FILE NAMES

All PACBASE files (except work files and internal sort files) are VSAM files whose physical names are parameterized in the following format:

```
      .PXU.PROPFIff         : VA Pacbase User Files
      .PXY.PROPROff         : VA Pacbase System Files
      .PXU.SEQ.PROPFI.OLDff : Back-up Files: (-1) Generation
      .PXU.SEQ.PROPFI.ff    :                ( 0) Generation
      .PXU.SEQ.PROPFI.NEWff :                (+1) Generation
      .XW.WPROC.ff          : Work Files
      .PXU.RPROC.ff         : Procedure Results Files
      .TXU.TRO00ff          : VA Pactables Files
```

CODE	MEANING
PXU	Index of VA Pacbase user files
PXY	Index of VA Pacbase system files
PRO	Root of VA Pacbase system (2 characters other than 99)
PFI	Code of VA Pacbase Database (2 characters other than 99)
XW	Index of work file names (must be prefixed by '%' to allow simultaneous execution of procedures in two or more partitions: VSAM concatenates the partition's ID with the physical names of files having this character in position 1.)
VCAP	VSAM catalogue of VA Pacbase files (At installation, it is assumed that all the files are in the same catalog.)
	===SELV TAB (PACTABLE option)
TXU	Index of VA Pactable file names
TRO	Root of VA Pactable option (2 characters other than PW)
VCAT	VSAM catalogue of VA Pactable files
	===SELV DSM (DSMS option)
DSMS	Name of VA Pacbase-Dsms interface file
VCAD	DSMS option VSAM catalogue name

RESOURCE PARAMETERS (VOLUMES, DRIVES):

CODE	MEANING
TAPEI	VA Pac installation back-up tape (used in the installation)
TAP1	Tape drive address
TAP2	Address of a second tape drive
VOLVWK	Volume with VSAM work files
VOLPSV	Volume with back-up and journal files (To ensure data security, it is advisable to keep them separate.)
VOLPUD	Volume with DATA components from VA Pac user files
VOLPUX	Volume with INDEX components from VA Pac user files
VOLPYD	Volume with DATA components from VA Pac system files
VOLPYX	Volume with INDEX components from VA Pac system files
VOLTUD	Volume with DATA components from VA Pactable user files (if ===SELV TAB)
VOLTUX	Volume with INDEX components from VA Pactable system files (if ===SELV TAB)
VWORK	Volume of non-VSAM work files
WKBG1 WKBG2 WKBG3	3 work areas on the work unit, coded WORK.='p,n' where: p = position of the 1st track n = number of allocated tracks
WKFx1 WKFx2 WKFx3	Optional: Equivalent to the 3 preceding areas to authorize the concurrent execution of select procedures in BG, F2, F3, etc.
RPCP	Primary allocation of the PC file in records
RPCS	Secondary allocation of the PC file in records (The PC file is the sequential back-up of a VA Pacbase database on the VOLPSV volume.)
RAJP	Primary allocation of the AJ file in records
RAJS	Secondary allocation of the AJ file in records (The AJ file is the VA Pacbase journal file on the VOLPSV volume.)

PARAMETERS OF LOAD-MODULE LIBRARIES

CODE	MEANING
MODB	Physical name of the library (or CIL) of the product's batch programs
LMODB	Name of the product's program library
SMODB	Name of the product's batch program sub-library, preceded by a period: SMODB='.sublib'
MODO	Physical name of the library (or CIL) of the product's on-line programs.
SMODO	Name of the on-line program sub-library, preceded by a period: SMODO='.sublib'
PRLB	Physical name of the library (or PL) of VSE procedures
LPRLB	Name of the VSE procedure library
SPRLB	Name of the VSE procedure sub-library, preceded by a period: SPRLB='.sublib'
SLIL	Physical name of the library (or SL) of SLI members
LSLI	Name of the SLI-member library
SLIS	Name of the SLI-member sub-library, preceded by a period: SLIS='.sublib'
SLIT	Type of SLI members. Default: SLIT='PAC'

MISCELLANEOUS PARAMETERS

CODE	MEANING
OPT	Parameters of the card : //OPTION Default : OPT=LOG, NODUMP
SORTWK	Name of the SORTWK procedure $SORTWK : Non-VSAM, default value The procedures ($0SORTWK,$xSORTWK,...) are provided in the '1VSE' module. SORTWK=SORTWK : SAM under VSAM Space Management The SORTWK procedure is provided in the '1VSE' module. SORTWK=...... : Your own procedure
LIST	CICS/RDO (Resource Definition Online): List name for the VA Pacbase package resources
GROUP	CICS/RDO (Resource Definition Online): Group name for the installed product's resources

4.11. SEPARATION OF JCL MODULES

JCL BEFORE/AFTER LINES

```
===BEGMOD
....1    )
.....    ) Lines to be inserted before each JCL module
....n    )

===ENDMOD
....1    )
.....    ) Lines to be inserted after each JCL module
....n    )
```

Lines may be inserted as input in the MM1JCL if the default option is not appropriate (see Subchapter "INSTALLATION DEFAULT OPTIONS" above).

The purpose of these lines is to execute the separation of the JCL file created by the MM1JCL utility into as many members as there are JCL modules.

This utility adds ....1 to ....n lines in front of each JCL module and ....1 to ....n lines to the end of each JCL module.
The parameter W$MODUL represents the module name; a period must be added after this parameter in order to concatenate it with a character string:

```
W$MODUL.B   gives  AXXXXB
W$MODUL..B  gives  AXXXX.B
W$MODULB is not transformed.
```

In order to automatically split the JCL into CMS/XEDIT members:

1. Insert the following cards:

   ===BEGMOD
   :READ xxxx$MODUL   (xxxx=up to 4-character prefix)

2. Execute the JCL processing job

3. Set the Punch received from POWER as the first card in the CMS reader (ORDER RDR nnnn where nnnn= spool number)

4. Adjust the Reader and Spool classes (SPOOL RDR CLASS *)

5. If the volume onto which the modules are loaded is to be shared among several machines, it must be accessed with filemode A (ACCESS cuu where cuu = disk address).

6. Activate automatic dispatching by providing the desired filetype (READ * filetype)

NOTE: It is possible to give the filetype at the :READ card level which is inserted by the BEGMOD command as follows:

```
READ xxxxW$MODUL filetype
```

7. Restore CMS initial configuration. (ACC 191 A then ACC cuu filemode)

To automatically split a SP2 or VSE/ESA library into members:

1. Insert the following cards:

```
===BEGMOD
CA xxxx$ZMODUL..P(xxxx=up to 4-character prefix)
===ENDMOD
/+
```

2. Execute the JCL processing job and store the MM1JCL PUNCH on a disk.

3. Set the PUNCH as input to LIBR with:

```
  ASSGN SYSIPT,DISK,.. (if the PUNCH is on a disk)
 // EXEC LIBR,PARM=' ACCESS SUBLIB=yourlib.sublib'
```

To automatically split a PDS into members in MVS:

1. Insert the following cards:

```
===BEGMOD
./ ADD NAME=$PRO$PFI$ZMODUL
//%USER$ZMODUL JOB
%ACCT,VSE,CLASS=X,NOTIFY=%USER
/*XMIT name_of_VSE_machine DLM=%%
===ENDMOD
%%
```

NOTE: This is also the JCL sample provided by default in step 2.

2. Run the JCL processing job.

3. Set the resulting PUNCH.

JCL modules are separated by the following command:
./ ADD NAME=<JCL_module>
This flow makes up the IEBUPDTE input used to catalog the various modules in the PDS provided for that purpose.

4.12. INSTALLATION PROCESS

INSTALLATION PROCESS

Once all JCLs are obtained, the installation of the VA PAC system is divided into the following phases:

1. Update and submission of POWER JCL

2. Update of CICS tables

2.1. CSD

2.2. ASSEMBLER

3. Cataloging of VSE procedures

3.1. SLI members

3.2. VSE procedures

4. Installation of the VA PAC system

4.1. Definition of permanent files

4.2. Cataloging of GPRT JCL (SLI member)

4.3. Cataloging of PACX JCL (SLI member)

4.4. Definition of GPRT work files

4.5. Definition of PACX work files

4.6. Installation of executable modules

4.7. Installation of system files,

4.8. Installation of user files,

4.9. Initialization of documentation error messages and user parameters.

5. Initialization of a database

5.1. Installation of VisualAge Pacbase test database,

5.2. Initialization of the generation-print request file,

5.3. Initialization of the PEI Function files.

6. Optional extensions to the VA PAC system installation

6.1. VisualAge Smalltalk Function

6.2. PAF Function

6.2.1. Environment

6.2.2. PAF+ Extension

6.3. WORKSTATION Function

6.4. PACBENCH QUALITY CONTROL Function

7. Modification of the CICS run-start JCL

8. Others Optional extensions system installation

8.1. Pac/Impact VisualAge Pacbase Function

8.2. PAC/TRANSFER Function

8.3. Bridge VA Pac - TeamConnection Function

9. List of installed programs and files

4.13. 1. CHANGING THE POWER JCL

1. MODIFICATION OF THE POWER JCL

The JCL module '1PWR' contains the lines to be added to the POWER submission JCL in order to use the control card$ SLI cards).

```
===MOD 1PWR
/* ************************************************************
/*      --- MODIFICATION OF THE POWER EXECUTION JCL           *
* ==============================================================
*  ADD         LSL$PRO$SLIS  IN THE LIBDEF SOURCE,SEARCH CARD  *
* ==============================================================
/* ************************************************************
// DLBL $LSLI,'$SLIL',,VSAM
```

4.14. 2.1 UPDATING THE CICS TABLES (CSD)

2.1 UPDATING THE CICS TABLES (CSD)

CICS/RDO:

The DFHCSDUP utility control cards, found in the 1CSD member, are used to update the DFHCSD.

The file table is updated by the 1FCT member. (See next sub-chapter.)

The elements contained in the 1CSD members are described in the next sub-chapter.

```
===MOD 1CSD
**********************************************
********** $PROD $PREL PROGRAMS **************
**********************************************
===SEQ FOR PUF
DEFINE PROGRAM($PRO.FANM) GROUP($GROUP)
DEFINE PROGRAM($PRO.FBIB) GROUP($GROUP)
DEFINE PROGRAM($PRO.FCTL) GROUP($GROUP)
DEFINE PROGRAM($PRO.FDBD) GROUP($GROUP)
DEFINE PROGRAM($PRO.FECR) GROUP($GROUP)
DEFINE PROGRAM($PRO.FENU) GROUP($GROUP)
DEFINE PROGRAM($PRO.FFOG) GROUP($GROUP)
DEFINE PROGRAM($PRO.FMCL) GROUP($GROUP)
DEFINE PROGRAM($PRO.FPGM) GROUP($GROUP)
DEFINE PROGRAM($PRO.FRUB) GROUP($GROUP)
DEFINE PROGRAM($PRO.FSDO) GROUP($GROUP)
DEFINE PROGRAM($PRO.FTXT) GROUP($GROUP)
DEFINE PROGRAM($PRO.FVER) GROUP($GROUP)
DEFINE PROGRAM($PRO.F000) GROUP($GROUP)
===SEQ
DEFINE PROGRAM($PRO.PA00) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA01) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA10) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA11) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA12) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA13) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA14) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA15) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA16) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA17) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA18) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA19) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA20) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA21) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA22) GROUP($GROUP)
===SEQ FOR PEI
DEFINE PROGRAM($PRO.PA30) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA31) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA32) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA33) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA34) GROUP($GROUP)
DEFINE PROGRAM($PRO.PA35) GROUP($GROUP)
===SEQ
DEFINE PROGRAM($PRO.PBND) GROUP($GROUP)
DEFINE PROGRAM($PRO.PPHP) GROUP($GROUP)
DEFINE PROGRAM($PRO.QA00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QB00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QC00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QC01) GROUP($GROUP)
DEFINE PROGRAM($PRO.QC50) GROUP($GROUP)
DEFINE PROGRAM($PRO.QD00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QE00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QF00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QF10) GROUP($GROUP)
DEFINE PROGRAM($PRO.QG00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QH00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QH01) GROUP($GROUP)
DEFINE PROGRAM($PRO.QH20) GROUP($GROUP)
DEFINE PROGRAM($PRO.QH30) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI00) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI01) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI02) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI03) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI04) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI05) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI20) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI21) GROUP($GROUP)
DEFINE PROGRAM($PRO.QI50) GROUP($GROUP)
DEFINE PROGRAM($PRO.QK10) GROUP($GROUP)
DEFINE PROGRAM($PRO.QK20) GROUP($GROUP)
DEFINE PROGRAM($PRO.QK30) GROUP($GROUP)
DEFINE PROGRAM($PRO.QL10) GROUP($GROUP)
DEFINE PROGRAM($PRO.QL20) GROUP($GROUP)
```

```
    DEFINE PROGRAM($PRO.QL21) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QL30) GROUP($GROUP)
    ===SEQ FOR SQL
    DEFINE PROGRAM($PRO.QL40) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QL41) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QL45) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QL46) GROUP($GROUP)
    ===SEQ
    DEFINE PROGRAM($PRO.QM00) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP00) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP01) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP02) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP03) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP04) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP05) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP06) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP07) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP08) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QP50) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QR00) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QS02) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QS03) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QS04) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QS05) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QS06) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QS08) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QS09) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QS10) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QT00) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QT10) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QT20) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QT50) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QUPA) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QUPF) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QU00) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QU01) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QU10) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QU20) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QV10) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QV20) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QV30) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QX00) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QX01) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY01) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY02) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY03) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY04) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY05) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY10) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY11) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY20) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QY30) GROUP($GROUP)
    DEFINE PROGRAM($PRO.QZ00) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q000) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q100) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q101) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q102) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q103) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q104) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q200) GROUP($GROUP)
    ===SEQ FOR TAB
    DEFINE PROGRAM($PRO.Q210) GROUP($GROUP)
    ===SEQ
    DEFINE PROGRAM($PRO.Q300) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q400) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q500) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q600) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q700) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q800) GROUP($GROUP)
    DEFINE PROGRAM($PRO.Q900) GROUP($GROUP)
    DEFINE PROGRAM($PRO.R000) GROUP($GROUP)
    DEFINE PROGRAM($PRO.R005) GROUP($GROUP)
    DEFINE PROGRAM($PRO.R100) GROUP($GROUP)
```

```
DEFINE PROGRAM($PRO.R200) GROUP($GROUP)
DEFINE PROGRAM($PRO.R400) GROUP($GROUP)
DEFINE PROGRAM($PRO.R500) GROUP($GROUP)
DEFINE PROGRAM($PRO.R600) GROUP($GROUP)
DEFINE PROGRAM($PRO.R980) GROUP($GROUP) RESIDENT(YES)
DEFINE PROGRAM($PRO.TPST) GROUP($GROUP)
DEFINE PROGRAM($PRO.TPWS) GROUP($GROUP)
DEFINE PROGRAM($PRO.UCTR) GROUP($GROUP) RESIDENT(YES)
**********************************************
********** $PROD $PREL TRANSACTIONS **********
**********************************************
DEFINE TRANSACTION($PRO$PFI) GROUP($GROUP) PROGRAM($PRO.R000)
DEFINE TRANSACTION($PRO.EE) GROUP($GROUP) PROGRAM(PARM PEI)
ADD GROUP($GROUP) LIST($LIST)
DEFINE TRANSACTION($PRO.EE) GROUP($GROUP) PROGRAM(PARM PEI)
```

```
===MOD 1FCT
* ***************************************************************
*
*   FCT CICS $PROD $PREL : $PRO$PFI.XX & $PRO$PRO.XX FILES
*
* ***************************************************************
===SEQ FOR PEI
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AB,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(BROWSE),                           $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=2,BUFNI=1,STRNO=1,                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AC,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,BUFNI=2,STRNO=2,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
===SEQ
         DFHFCT TYPE=DATASET,DATASET=$PRO$PRO.AE,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,BUFNI=2,STRNO=2,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AG,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=2,BUFNI=1,STRNO=1,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AJ,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=2,STRNO=1,LOG=YES,                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AN,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,BUFNI=2,STRNO=2,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PRO.AP,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,BUFNI=2,STRNO=2,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AR,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,STRNO=2,                            $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PRO.PA,          $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
```

```
                  LSRPOOL=$NLSR,                            $Y.*
                  BUFND=2,BUFNI=1,STRNO=1,                  $Y.*
                  RECFORM=(VARIABLE,BLOCKED),               $Y.*
                  FILSTAT=(ENABLED,CLOSED)
```

4.15. 2.2 UPDATING THE CICS TABLES (ASSEMBLER)

2.2 UPDATING THE CICS TABLES (ASSEMBLER)

```
Assembler sources:

  . PCT: JCL Module '1PCT' - two transaction codes:

    $PRO$PFI (DTB=YES option required) to access VA PAC .
             One transaction per database.

    $PRO.EF  (DTB=YES option required) for user parameters
             (and also PEI function) management.
             One transaction for the VA PAC  system.

  . PPT: JCL Module '1PPT' - Codified Maps and Programs:

    $PRO.Qxxx, $PRO.Pxxx or $PRO.Rxxx for programs,

    $PRO.Nxxx, $PRO.Mxxx or $PRO.Oxxx for maps,
    where xxxx = an alphanumeric code.

    One PPT for the VA PAC  system.

  . FCT: JCL Module '1FCT':

    -  2 VA PAC  System files:
       $PRO$PRO.AE : Error messages
       $PRO$PRO.AP : User parameters

    -  4 user files per database:

       $PRO$PFI.AN : VA PAC  Database, index file
       $PRO$PFI.AR : VA PAC  Database, data file
       $PRO$PFI.AJ : Journal file
       $PRO$PFI.AG : Generation-print request file

    -  2 PEI-user files per database:
       $PRO$PFI.AB : Consulted PEI file
       $PRO$PFI.AC : Updated PEI file

    -  1 PACTABLE System file, common to the system:
       $TRO.00TD   : Table description file

    -  1 work file for PACSQL, common to the system:
       $PRO$PRO.PA
```

The option LOG=YES for the FCT is required for files AN, AR, AJ, AG, and AP, as well as for the PEI AC file.

The estimated number of STRINGS, INDEX BUFFERS and DATA BUFFERS indicated are the minimum necessary for the system.

The two characters '$PRO' of transaction codes, names of maps and programs and file codes are user-defined.

The $PRO$PFI transaction allows the execution of current operations on the database (consultation and updating).

The $PRO.EF transaction allows the execution of current operations on user parameters stored in the AE and AP files (and in the PEI AB and AC files).

```
===MOD 1PCT
* ***********************************************************
*
*   PCT CICS $PROD $PREL : $PRO.XX TRANSACTIONS
*
* ***********************************************************
         DFHPCT TYPE=ENTRY,TRANSID=$PRO.$PFI.,             $Y.*
               PROGRAM=$PRO.R000,DTB=YES
         DFHPCT TYPE=ENTRY,TRANSID=$PRO.EE,                $Y.*
               PROGRAM=$PRO.QUPA,DTB=YES
* THE ENTRY BELOW IS THE FRENCH VERSION OF $PRO.EE
*        DFHPCT TYPE=ENTRY,TRANSID=$PRO.EF,                $Y.*
*              PROGRAM=$PRO.QUPF,DTB=YES
```

```
===MOD 1PPT
* ****************************************************************
*
*   PPT CICS $PROD $PREL : PROGRAMS $PRO.XXXX
*
* ****************************************************************
===SEQ FOR PUF
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FANM,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FBIB,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FCTL,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FDBD,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FECR,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FENU,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FFOG,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FMCL,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FPGM,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FRUB,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FSDO,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FTXT,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.FVER,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.F000,PGMLANG=COBOL
===SEQ
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA01,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA11,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA12,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA13,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA14,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA15,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA16,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA17,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA18,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA19,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA21,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA22,PGMLANG=COBOL
===SEQ FOR PEI
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA30,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA31,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA32,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA33,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA34,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PA35,PGMLANG=COBOL
===SEQ
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PBND,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.PPHP,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QA00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QB00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QC00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QC01,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QC50,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QD00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QE00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QF00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QF10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QG00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QH00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QH01,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QH20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QH30,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI01,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI02,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI03,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI04,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI05,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI21,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QI50,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QK10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QK20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QK30,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QL10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QL20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QL21,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QL30,PGMLANG=COBOL
===SEQ FOR SQL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QL40,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QL41,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QL45,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QL46,PGMLANG=COBOL
===SEQ
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QM00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP01,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP02,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP03,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP04,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP05,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP06,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP07,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP08,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QP50,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QR00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QS02,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QS03,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QS04,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QS05,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QS06,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QS08,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QS09,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QS10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QT00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QT10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QT20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QT50,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QUPA,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QUPF,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QU00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QU01,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QU10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QU20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QV10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QV20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QV30,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QX00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QX01,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY01,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY02,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY03,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY04,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY05,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY10,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY11,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY20,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QY30,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.QZ00,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q000,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q100,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q101,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q102,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q103,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q104,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q200,PGMLANG=COBOL
===SEQ FOR TAB
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q210,PGMLANG=COBOL
===SEQ
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q300,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q400,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q500,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q600,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q700,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q800,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.Q900,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.R000,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.R005,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.R100,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.R200,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.R400,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.R500,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.R600,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.R980,PGMLANG=COBOL,RES=YES
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.TPST,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.TPWS,PGMLANG=COBOL
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.UCTA,PGMLANG=COBOL,RES=YES
 DFHPPT TYPE=ENTRY,PROGRAM=$PRO.UCTR,PGMLANG=COBOL,RES=YES
```

```
===MOD 1FCT
* **************************************************************
*
*   FCT CICS $PROD $PREL : $PRO$PFI.XX & $PRO$PRO.XX FILES
*
* **************************************************************
===SEQ FOR PEI
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AB,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(BROWSE),                           $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=2,BUFNI=1,STRNO=1,                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AC,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,BUFNI=2,STRNO=2,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
===SEQ
         DFHFCT TYPE=DATASET,DATASET=$PRO$PRO.AE,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,BUFNI=2,STRNO=2,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AG,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=2,BUFNI=1,STRNO=1,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AJ,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=2,STRNO=1,LOG=YES,                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AN,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,BUFNI=2,STRNO=2,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PRO.AP,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,BUFNI=2,STRNO=2,                    $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PFI.AR,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
               LSRPOOL=$NLSR,                              $Y.*
               BUFND=3,STRNO=2,                            $Y.*
               LOG=YES,                                    $Y.*
               RECFORM=(FIXED,BLOCKED),                    $Y.*
               FILSTAT=(ENABLED,OPENED)
         DFHFCT TYPE=DATASET,DATASET=$PRO$PRO.PA,         $Y.*
               ACCMETH=VSAM,                               $Y.*
               SERVREQ=(ADD,UPDATE,DELETE,BROWSE),         $Y.*
```

```
                    LSRPOOL=$NLSR,                          $Y.*
                    BUFND=2,BUFNI=1,STRNO=1,                $Y.*
                    RECFORM=(VARIABLE,BLOCKED),             $Y.*
                    FILSTAT=(ENABLED,CLOSED)
```

4.16. 3.1 CATALOGING THE SLI PROCEDURES

3.1 CATALOGING OF SLI PROCEDURES

$ SLI in POWER)

The JCL Module '1SLI' contains the source codes of procedures to be cataloged in a library which may be accessed in read mode through POWER. (Card: // LIBDEF SOURCE,SEARCH in the POWER submission JCL).

The procedures are:

-	DBJB$PRO.$PFI	Start of standard job
-	JPAF$PRO.$PFI	Standard JCL for PAF user program
-	DF$PRO.$PFI.ff	DELETE-DEFINE of ff user file
-	DF$PRO.$PRO.ff	DELETE-DEFINE of ff system file
-	D$PRO.$PRO.fff	DELETE-DEFINE of fff file
-	DFSYSPAF	DELETE-DEFINE of SYSPAF file (PACSQL users)
-	RG$PRO.$PFI.ff	Rotation of ff back-up file

The 'DFxxxxff' and 'Dxxxxfff' members are used to centrally manage the VSAM file allocation parameters of the VA PAC environment.

The 'RGxxxxff' members are used to centrally manage the number of back-up file generations to be saved.

The 'DBJBxxxx' members are used to centrally manage the JCL common to all VA PAC operation procedures that is found at the beginning of jobs (for ex., allocating resources such as libraries).

```
===MOD 1SLI
* $X JOB JNM=$PRFJ.1SLI,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.1SLI        ****** $PROD $PREL ******
/*      PROCEDURE CATALOGING
/* ************************************************************
// OPTION $OPT
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL $LSLI,'$SLIL',,VSAM
/* GOTO JCLSTXX
* ----- STEP 1 ----- LIBR
// EXEC LIBR,PARM=' AC S=$LSLI$SLIS'
CA DBJB$PRO.$PFI..$SLIT R=Y
// ASSGN SYS005,SYSLST
// ASSGN SYS006,SYSRDR
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL $LMODB,'$MODB',,VSAM
// DLBL $LPRLB,'$PRLB',,VSAM
// LIBDEF PHASE,SEARCH=$LMODB$SMODB
// LIBDEF PROC,SEARCH=$LPRLB$SPRLB
/+
CA PAFB$PRO.$PFI..$SLIT R=Y
* --- STEP 1 --- PAFP10
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7AE,'$PXY..$PRO$PRO.AE',,VSAM
// DLBL PAC7AN,'$PXU..$PRO$PFI.AN',,VSAM
// DLBL PAC7AR,'$PXU..$PRO$PFI.AR',,VSAM
// EXEC PROC=$$PACWK1,DLBL=COB80,ASSGN=SYS015
// EXEC PAFP10,SIZE=AUTO
 CBL APOST,LIB,NOADV,TRUNC(BIN),NOXREF,RES,RENT
* $X DATA INPUT
* $X /*
* --- STEP 2 --- IGYCRCTL
// EXEC PROC=$$PACWK1,ASSGN=SYSIPT
// EXEC IGYCRCTL,SIZE=IGYCRCTL
* --- STEP 3 --- LNKEDT
// EXEC LNKEDT
* $X /&
 CLOSE SYSIPT,SYSRDR
/+
CA PAFT$PRO.$PFI..$SLIT R=Y
* --- STEP 1 --- PAFP10
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7AE,'$PXY..$PRO$PRO.AE',,VSAM
// DLBL PAC7AN,'$PXU..$PRO$PFI.AN',,VSAM
// DLBL PAC7AR,'$PXU..$PRO$PFI.AR',,VSAM
// EXEC PROC=$$PACWK1,DLBL=COB80,ASSGN=SYS015
// EXEC PAFP10,SIZE=AUTO
 CBL APOST,LIB,NOADV,TRUNC(BIN),NOXREF,RES,RENT
 CBL XOPTS(CICS,OPT,NOLIST)
* $X DATA INPUT
* $X /*
* --- STEP 2 --- DFHECP1$
// ASSGN SYS091,SYSPCH
// EXEC PROC=$$PACWK1,ASSGN=SYSIPT
// EXEC PROC=$$PACWK2,ASSGN=SYSPCH
// EXEC DFHECP1$
* --- STEP 3 --- IGYCRCTL
 CLOSE SYSIPT,SYSRDR
 CLOSE SYSPCH,SYS091
// EXEC PROC=$$PACWK2,ASSGN=SYSIPT
 INCLUDE DFHECI
// EXEC IGYCRCTL,SIZE=IGYCRCTL
* --- STEP 4 --- LNKEDT
// EXEC LNKEDT
* $X /&
 CLOSE SYSIPT,SYSRDR
/+
CA JPAF$PRO.$PFI..$SLIT R=Y
/* == TOP OF PAF JCL SKELETON ==
// DLBL $PRO$PRO.CT,'$VCAP',,VSAM
// DLBL SY8PAF,'$XW..SYSPAF',,VSAM,CAT=$PRO$PRO.CT
// DLBL PAC7AE,'$PXY..$PRO$PRO.AE',,VSAM,CAT=$PRO$PRO.CT
// DLBL PAC7AN,'$PXU..$PRO$PFI.AN',,VSAM,CAT=$PRO$PRO.CT
// DLBL PAC7AR,'$PXU..$PRO$PFI.AR',,VSAM,CAT=$PRO$PRO.CT
/* == END OF PAF JCL SKELETON ==
/+
CA DFMODFBK.$SLIT R=Y
    /* --------- DELETE-DEFINE FIXBLK MODEL ------- */
 DEL                   ($MODELF) CL
 DEF CL          (NAME ($MODELF) -
                  VOL($VOLVWK) -
                  REC  (100 100) -
                  RECSZ(80 80) -
                  CISZ (8192) -
                  RECFM(FB(80)) -
                  RUS -
                  NAL -
                  NIXD -
      ) DATA     (NAME ($MODELF..D) )
/+
CA DFMODVBK.$SLIT R=Y
    /* --------- DELETE-DEFINE VARBLK MODEL ------- */
 DEL                   ($MODELV) CL
 DEF CL          (NAME ($MODELV) -
                  VOL($VOLVWK) -
                  REC  (100 100) -
                  RECSZ(80 200) -
                  CISZ (8192) -
                  RECFM(VB) -
                  RUS -
                  NAL -
                  NIXD -
      ) DATA     (NAME ($MODELV..D) )
/+
CA DF$PRO.$PFI.AB.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AB ------------ */
 DEL                   ($PXU..$PRO.$PFI.AB) CL
 DEF CL          (NAME ($PXU..$PRO.$PFI.AB) -
                  SHR(4) RUS KEYS(26 0) -
                  RECSZ(110 110) REC(200 200) -
      ) INDEX    (NAME ($PXU..$PRO.$PFI.AB.I) -
                  VOL($VOLPUX) -
      ) DATA     (NAME ($PXU..$PRO.$PFI.AB.D) -
                  VOL($VOLPUD) FSPC(10 5) CISZ(4096) )
/+
CA DF$PRO.$PFI.AC.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AC ------------ */
 DEL                   ($PXU..$PRO.$PFI.AC) CL
 DEF CL          (NAME ($PXU..$PRO.$PFI.AC) -
                  SHR(2 3) RUS KEYS(26 0) -
                  RECSZ(110 110) REC(200 200) -
      ) INDEX    (NAME ($PXU..$PRO.$PFI.AC.I) -
                  VOL($VOLPUX) -
      ) DATA     (NAME ($PXU..$PRO.$PFI.AC.D) -
                  VOL($VOLPUD) FSPC (10 5) CISZ(4096) )
/+
CA DF$PRO.$PRO.AE.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AE ------------ */
 DEL                   ($PXY..$PRO.$PRO.AE) CL
 DEF CL          (NAME ($PXY..$PRO.$PRO.AE) -
                  SHR(2 3) RUS KEYS(12 0) -
                  RECSZ(80 80) REC(30000 1000) -
      ) INDEX    (NAME ($PXY..$PRO.$PRO.AE.I) -
                  VOL($VOLPYX) -
      ) DATA     (NAME ($PXY..$PRO.$PRO.AE.D) -
                  VOL($VOLPYD) FSPC(2 1) CISZ(4096) )
/+
CA DF$PRO.$PFI.AG.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AG ------------ */
 DEL                   ($PXU..$PRO.$PFI.AG) CL
 DEF CL          (NAME ($PXU..$PRO.$PFI.AG) -
                  SHR(2) RUS KEYS(27 0) -
                  RECSZ(150 150) REC(500 500) -
      ) INDEX    (NAME ($PXU..$PRO.$PFI.AG.I) -
```

```
                 VOL($VOLPUX) -
      ) DATA    (NAME ($PXU..$PRO.$PFI.AG.D) -
                 VOL($VOLPUD) FSPC(10 5) CISZ(4096) )
/+
CA DF$PRO.$PFI.AJ.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AJ ------------ */
 DEL                  ($PXU..$PRO.$PFI.AJ) CL
 DEF CL         (NAME ($PXU..$PRO.$PFI.AJ) -
                 SHR(2) RUS NUMD VOL($VOLPUD) -
                 RECSZ(167 167) REC($RAJP $RAJS) -
      ) DATA    (NAME ($PXU..$PRO.$PFI.AJ.D) -
                 CISZ(8192) )
/+
CA DF$PRO.$PFI.AN.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AN ------------ */
 DEL                  ($PXU..$PRO.$PFI.AN) CL
 DEF CL         (NAME ($PXU..$PRO.$PFI.AN) -
                 SHR(2) RUS KEYS(43 0) -
                 RECSZ(54 54) REC(90000 20000) -
      ) INDEX   (NAME ($PXU..$PRO.$PFI.AN.I) -
                 VOL($VOLPUX) -
      ) DATA    (NAME ($PXU..$PRO.$PFI.AN.D) -
                 VOL($VOLPUD) FSPC(5 5) -
                 CISZ(8192)  BUFSP(35840) )
/+
CA DF$PRO.$PRO.AP.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AP ------------ */
 DEL                  ($PXY..$PRO.$PRO.AP) CL
 DEF CL         (NAME ($PXY..$PRO.$PRO.AP) -
                 SHR(2 3) RUS KEYS(7 0) -
                 RECSZ(80 80) REC(500 500) -
      ) INDEX   (NAME ($PXY..$PRO.$PRO.AP.I) -
                 VOL($VOLPYX) -
      ) DATA    (NAME ($PXY..$PRO.$PRO.AP.D) -
                 VOL($VOLPYD) FSPC(10 5) CISZ(4096) )
/+
CA DF$PRO.$PFI.AR.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AR ------------ */
 DEL                  ($PXU..$PRO.$PFI.AR) CL
 DEF CL         (NAME ($PXU..$PRO.$PFI.AR) -
                 SHR(2) RUS NUMD VOL($VOLPUD) -
                 RECSZ(140 140) REC(40000 15000) -
      ) DATA    (NAME ($PXU..$PRO.$PFI.AR.D) -
                 CISZ(8192) )
/+
CA D$PRO.$PRO.AE0.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE AE0 ----------- */
 DEL                  ($PXY..SEQ.$PRO$PRO..AE0) CL
 DEF CL         (NAME ($PXY..SEQ.$PRO$PRO..AE0) -
                 MODEL($MODELF) -
                 VOL($VOLPSV) REC(30000 1000) -
      ) DATA    (NAME ($PXY..SEQ.$PRO$PRO..AE0.D) )
/+
CA DF$PRO.$PRO.GS.$SLIT R=Y
 /* ----- DELETE-DEFINE FILE GS ---------- */
 DEL                  ($PXU..$PRO.$PRO.GS) CL
 DEF CL         (NAME ($PXU..$PRO.$PRO.GS) -
                 SHR(2 3) RUS KEYS(25 0) -
                 RECSZ(203 203) REC(1000 1000) -
      ) INDEX   (NAME ($PXU..$PRO.$PRO.GS.I) -
                 VOL($VOLPUX) -
      ) DATA    (NAME ($PXU..$PRO.$PRO.GS.D) -
                 VOL($VOLPUD) FSPC (5 5) CISZ(4096) )
/+
CA DF$PRO.$PRO.PA.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE PA ------------ */
 DEL                  ($PXY..$PRO.$PRO.PA) CL
 DEF CL         (NAME ($PXY..$PRO.$PRO.PA) -
                 SHR (2 3) RUS KEYS (37 2) -
                 RECSZ(200 539) REC(10000 10000) -
      ) INDEX   (NAME ($PXY..$PRO.$PRO.PA.I) -
                 VOL($VOLPYX)        -
      ) DATA    (NAME ($PXY..$PRO.$PRO.PA.D) -
```

```
                 VOL($VOLPYD) FSPC(5 5) CISZ(8192) )
/+
CA DF$PRO.$PFI.PC.$SLIT R=Y
  /* ------ DELETE-DEFINE FILE PC ----------- */
 DEL                   ($PXU..SEQ.$PRO$PFI..NEWPC) CL
 DEF CL          (NAME ($PXU..SEQ.$PRO$PFI..NEWPC) -
                 MODEL($MODELV) -
                 RECSZ(63 159) REC($RPCP $RPCS) -
                 VOL($VOLPSV) -
      ) DATA     (NAME ($PXU..SEQ.$PRO$PFI..NEWPC.D) )
/+
CA DF$PRO.$PFI.PD.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE PD ------------ */
 DEL                   ($PXU..SEQ.$PRO$PFI..NEWPD) CL
 DEF CL          (NAME ($PXU..SEQ.$PRO$PFI..NEWPD) -
                 MODEL($MODELV) -
                 RECSZ(63 159) REC($RPCP $RPCS) -
                 VOL($VOLPSV) -
      ) DATA     (NAME ($PXU..SEQ.$PRO$PFI..NEWPD.D) )
/+
CA DF$PRO.$PRO.SC.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE SC ------------ */
 DEL                   ($PXY..$PRO.$PRO.SC) CL
 DEF CL          (NAME ($PXY..$PRO.$PRO.SC) -
                 SHR(2 3) RUS KEYS(4 0) -
      ) INDEX    (NAME ($PXY..$PRO.$PRO.SC.I) -
                 VOL($VOLPYX) -
      ) DATA     (NAME ($PXY..$PRO.$PRO.SC.D) -
                 RECSZ(3204 3204) CISZ(22528) -
                 VOL($VOLPYD) REC (40 5) )
/+
CA DF$PRO.$PRO.SF.$SLIT R=Y
   /* -------- DELETE-DEFINE FILE SF ----------- */
 DEL                   ($PXY..$PRO.$PRO.SF) CL
 DEF CL          (NAME ($PXY..$PRO.$PRO.SF) -
                 MODEL($MODELF) -
                 SHR(2 3) -
                 RECFM(FB(119)) -
                 RECSZ(119 119) REC(3000 100) -
                 VOL($VOLPYD) -
      ) DATA     (NAME ($PXY..$PRO.$PRO.SF.D) )
/+
CA DF$PRO.$PRO.SG.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE SG ------------ */
 DEL                   ($PXY..$PRO.$PRO.SG) CL
 DEF CL          (NAME ($PXY..$PRO.$PRO.SG) -
                 SHR(2 3) RUS KEYS(5 0) -
      ) INDEX    (NAME ($PXY..$PRO.$PRO.SG.I) -
                 VOL($VOLPYX) -
      ) DATA     (NAME ($PXY..$PRO.$PRO.SG.D) -
                 RECSZ(4605 4605) CISZ(22528) -
                 VOL($VOLPYD) REC(400 10) )
/+
CA DF$PRO.$PRO.SP.$SLIT R=Y
   /* -------- DELETE-DEFINE FILE SP ----------- */
 DEL                   ($PXY..$PRO.$PRO.SP) CL
 DEF CL          (NAME ($PXY..$PRO.$PRO.SP) -
                 SHR(2 3) RUS KEYS(5 0) -
      ) INDEX    (NAME ($PXY..$PRO.$PRO.SP.I) -
                 VOL($VOLPYX) -
      ) DATA     (NAME ($PXY..$PRO.$PRO.SP.D) -
                 RECSZ(4605 4605) CISZ(22528) -
                 VOL($VOLPYD) REC(5 1) )
/+
CA DF$PRO.$PRO.SR.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE SR ------------ */
 DEL                   ($PXY..$PRO.$PRO.SR) CL
 DEF CL          (NAME ($PXY..$PRO.$PRO.SR) -
                 SHR (2 3) RUS KEYS(5 0) -
      ) INDEX    (NAME ($PXY..$PRO.$PRO.SR.I) -
                 VOL($VOLPYX) -
      ) DATA     (NAME ($PXY..$PRO.$PRO.SR.D) -
                 RECSZ(4605 4605) CISZ(22528) -
                 VOL($VOLPYD) REC(30 5) )
/+
CA DF$PRO.$PRO.SS.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE SS ------------ */
 DEL                  ($PXY..$PRO.$PRO.SS) CL
 DEF CL         (NAME ($PXY..$PRO.$PRO.SS) -
                 SHR(2 3) RUS KEYS(5 0) -
      ) INDEX   (NAME ($PXY..$PRO.$PRO.SS.I) -
                 VOL($VOLPYX) -
      ) DATA    (NAME ($PXY..$PRO.$PRO.SS.D) -
                 RECSZ(4605 4605) CISZ(22528) -
                 VOL($VOLPYD) REC(300 10) )
/+
CA DFSYSPAF.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE SYSPAF -------- */
 DEL                  ($XW..SYSPAF) CL
 DEF CL         (NAME ($XW..SYSPAF) -
                 SHR(2 3) RUS NAL KEYS(12 0) -
                 RECSZ(170 468) REC(1000 1000) -
                 VOL($VOLVWK) -
      ) INDEX   (NAME ($XW..SYSPAF.I) -
      ) DATA    (NAME ($XW..SYSPAF.D) CISZ(8192) )
/+
CA D$PRO.$PRO.TCO.$SLIT R=Y
    /* --------- DELETE-DEFINE FICHIER TCO -------- */
 DEL                  ($PXY..$PRO.$PRO.TCO) CL
 DEF CL         (NAME ($PXY..$PRO.$PRO.TCO) -
                 MODEL($MODELF) -
                 VOL($VOLPSV) REC(50 10) -
                 RECSZ(117 117) RECFM(FB(117)) -
      ) DATA    (NAME ($PXY..$PRO.$PRO.TCO.D) )
/+
CA DF$TRO.00TD.$SLIT R=Y
    /* --- DELETE-DEFINE FICHIER 00TD  --- */
 DEL                  ($TXU..$TRO.00TD) CL
 DEF CL         (NAME ($TXU..$TRO.00TD) -
                 SHR(2 3) RUS KEYS(21 0) -
                 RECSZ(240 240) REC(1000 1000) -
      ) INDEX   (NAME ($TXU..$TRO.00TD.I) -
                 VOL($VOLTUX) -
      ) DATA    (NAME ($TXU..$TRO.00TD.D) -
                 VOL($VOLTUD) FSPC(10 5) CISZ(4096) )
/+
CA D$PRO.$PRO.VGE.$SLIT R=Y
    /* --------- DELETE-DEFINE FILE VGE ----------- */
 DEL                  ($PXY..$PRO.$PRO.VGE) CL
 DEF CL         (NAME ($PXY..$PRO.$PRO.VGE) -
                 MODEL($MODELF) -
                 VOL($VOLPSV) REC(100 10) -
                 RECSZ(117 117) RECFM(FB(117)) -
      ) DATA    (NAME ($PXY..$PRO.$PRO.VGE.D) )
/+
CA RG$PRO.$PFI.FH.$SLIT R=Y
    /* --- ROTATING BACKUP FILE FH --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDFH) CL
 ALTER           $PXU..SEQ.$PRO$PFI..FH -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDFH)
 ALTER           $PXU..SEQ.$PRO$PFI..FH.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDFH.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWFH -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..FH)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWFH.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..FH.D)
/+
CA RG$PRO.$PFI.FO.$SLIT R=Y
    /* --- ROTATING BACKUP FILE FO --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDFO) CL
 ALTER           $PXU..SEQ.$PRO$PFI..FO -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDFO)
 ALTER           $PXU..SEQ.$PRO$PFI..FO.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDFO.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWFO -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..FO)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWFO.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..FO.D)
/+
CA RG$PRO.$PFI.FQ.$SLIT R=Y
    /* --- ROTATING BACKUP FILE FQ --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDFQ) CL
 ALTER           $PXU..SEQ.$PRO$PFI..FQ -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDFQ)
 ALTER           $PXU..SEQ.$PRO$PFI..FQ.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDFQ.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWFQ -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..FQ)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWFQ.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..FQ.D)
/+
CA RG$PRO.$PFI.FR.$SLIT R=Y
    /* --- ROTATING BACKUP FILE FR --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDFR) CL
 ALTER           $PXU..SEQ.$PRO$PFI..FR -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDFR)
 ALTER           $PXU..SEQ.$PRO$PFI..FR.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDFR.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWFR -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..FR)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWFR.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..FR.D)
/+
CA RG$PRO.$PFI.PC.$SLIT R=Y
    /* --- ROTATING BACKUP FILE PC --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDPC) CL
 ALTER           $PXU..SEQ.$PRO$PFI..PC -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPC)
 ALTER           $PXU..SEQ.$PRO$PFI..PC.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPC.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPC -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PC)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPC.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PC.D)
/+
CA RG$PRO.$PFI.PD.$SLIT R=Y
    /* --- ROTATING BACKUP FILE PD --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDPD) CL
 ALTER           $PXU..SEQ.$PRO$PFI..PD -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPD)
 ALTER           $PXU..SEQ.$PRO$PFI..PD.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPD.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPD -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PD)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPD.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PD.D)
/+
CA RG$PRO.$PRO.PE.$SLIT R=Y
    /* --- ROTATING BACKUP FILE PE --- */
 DEL            ($PXY..SEQ.$PRO$PRO..OLDPE) CL
 ALTER           $PXY..SEQ.$PRO$PRO..PE -
 NEWNAME        ($PXY..SEQ.$PRO$PRO..OLDPE)
 ALTER           $PXY..SEQ.$PRO$PRO..PE.D -
 NEWNAME        ($PXY..SEQ.$PRO$PRO..OLDPE.D)
 ALTER           $PXY..SEQ.$PRO$PRO..NEWPE -
 NEWNAME        ($PXY..SEQ.$PRO$PRO..PE)
 ALTER           $PXY..SEQ.$PRO$PRO..NEWPE.D -
 NEWNAME        ($PXY..SEQ.$PRO$PRO..PE.D)
/+
CA RG$PRO.$PFI.PG.$SLIT R=Y
    /* --- ROTATING BACKUP FILE PG --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDPG) CL
 ALTER           $PXU..SEQ.$PRO$PFI..PG -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPG)
 ALTER           $PXU..SEQ.$PRO$PFI..PG.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPG.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPG -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PG)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPG.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PG.D)
/+
CA RG$PRO.$PFI.PJ.$SLIT R=Y
    /* --- ROTATING BACKUP FILE PJ --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDPJ) CL
 ALTER           $PXU..SEQ.$PRO$PFI..PJ -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPJ)
 ALTER           $PXU..SEQ.$PRO$PFI..PJ.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPJ.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPJ -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PJ)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPJ.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PJ.D)
/+
CA RG$PRO.$PFI.PP.$SLIT R=Y
    /* --- ROTATING BACKUP FILE PP --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDPP) CL
 ALTER           $PXU..SEQ.$PRO$PFI..PP -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPP)
 ALTER           $PXU..SEQ.$PRO$PFI..PP.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPP.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPP -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PP)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPP.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PP.D)
/+
CA RG$PRO.$PFI.PQ.$SLIT R=Y
    /* --- ROTATING BACKUP FILE PQ --- */
 DEL            ($PXU..SEQ.$PRO$PFI..OLDPQ) CL
 ALTER           $PXU..SEQ.$PRO$PFI..PQ -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPQ)
 ALTER           $PXU..SEQ.$PRO$PFI..PQ.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..OLDPQ.D)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPQ -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PQ)
 ALTER           $PXU..SEQ.$PRO$PFI..NEWPQ.D -
 NEWNAME        ($PXU..SEQ.$PRO$PFI..PQ.D)
/+
/*
// EXEC LISTLOG
/&
* $X EOJ
```

4.17. 3.2 CATALOGING THE VSE PROCEDURES

3.2 CATALOGING OF VSE PROCEDURES

The JCL Module '1VSE' contains the source code of the JCL procedures to be cataloged. These procedures are offered as examples only and can be changed (for example, by altering the 'SORTWK' parameter of installation job 'JOB2').

The following VSE procedures are in the stream of procedures to catalog:

```
SORTWK    Procedure to allocate a SORTWK in SAM
          under VSAM

$0SORTWK  Procedure to allocate a SORTWK in SAM
          for the BG partition

$xSORTWK  Procedure to allocate a SORTWK in SAM
          for the Fx partition (F2 to F9)

$0PACWK1  Procedures to allocate SAM work files
$0PACWK2  for the BG partition

$xPACWK1  Procedures to allocate SAM work files
$xPACWK2  for the Fx partition (F2 to F9)
```

```
===MOD 1VSE
* $X JOB JNM=$PRFJ.1VSE,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.1VSE        ****** $PROD $PREL ******
/* CATALOGING OF VSE PROCEDURES
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
* ----- STEP 1 ----- LIBR
// EXEC LIBR,PARM=' AC S=$LPRLB$SPRLB'
 CATALP SORTWK
// DLBL SORTWK1,'$XW..SORTWK',0,VSAM,DISP=(NEW,DELETE,DELETE),   $Y.*
               RECSIZE=9999,RECORDS=(999,999)
/+
 CATALP $0SORTWK
// DLBL SORTWK1,'PACWK1.BG',0,DA
// EXTENT SYS001,$VWORK,,1,$WKBG1
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $2SORTWK
// DLBL SORTWK1,'PACWK1.F2',0,DA
// EXTENT SYS001,$VWORK,,1,$WKF21
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $3SORTWK
// DLBL SORTWK1,'PACWK1.F3',0,DA
// EXTENT SYS001,$VWORK,,1,$WKF31
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $4SORTWK
// DLBL SORTWK1,'PACWK1.F4',0,DA
// EXTENT SYS001,$VWORK,,1,$WKF41
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $5SORTWK
// DLBL SORTWK1,'PACWK1.F5',0,DA
// EXTENT SYS001,$VWORK,,1,$WKF51
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $6SORTWK
// DLBL SORTWK1,'PACWK1.F6',0,DA
// EXTENT SYS001,$VWORK,,1,$WKF61
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $7SORTWK
// DLBL SORTWK1,'PACWK1.F7',0,DA
// EXTENT SYS001,$VWORK,,1,$WKF71
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $8SORTWK
// DLBL SORTWK1,'PACWK1.F8',0,DA
// EXTENT SYS001,$VWORK,,1,$WKF81
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $9SORTWK
// DLBL SORTWK1,'PACWK1.F9',0,DA
// EXTENT SYS001,$VWORK,,1,$WKF91
// ASSGN SYS001,DISK,VOL=$VWORK,SHR
/+
 CATALP $0PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.BG',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKBG2
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $0PACWK2
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.BG',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKBG3
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $2PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.F2',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF22
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $2PACWK2
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.F2',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF23
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $3PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.F3',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF32
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $3PACWK2
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.F3',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF33
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $4PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.F4',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF42
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $4PACWK2
```

```
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.F4',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF43
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $5PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.F5',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF52
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $5PACWK2
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.F5',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF53
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $6PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.F6',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF62
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $6PACWK2
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.F6',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF63
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $7PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.F7',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF72
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $7PACWK2
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.F7',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF73
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $8PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.F8',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF82
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $8PACWK2
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.F8',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF83
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $9PACWK1
// PROC DLBL=SORTWK2,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK2.F9',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF92
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
 CATALP $9PACWK2
// PROC DLBL=SORTWK3,ASSGN=SYS001,OPT=TEMP
// IF ASSGN=SYSIPT THEN
// SETPARM OPT=PERM,DLBL=IJSYSIN
// IF ASSGN=SYSPCH THEN
// SETPARM OPT=PERM,DLBL=IJSYSPH
// IF ASSGN=SYSLST THEN
// SETPARM OPT=PERM,DLBL=IJSYSLS
// DLBL &DLBL,'PACWK3.F9',0,SD
// EXTENT &ASSGN,$VWORK,,,$WKF93
// ASSGN &ASSGN,DISK,&OPT,VOL=$VWORK,SHR
/+
/*
// EXEC LISTLOG
/&
* $X EOJ
```

4.18. 4.1 DEFINITION OF PERMANENT FILES

4.1 DEFINITION OF PERMANENT FILES

The JCL Module '2FIC' performs a DELETE-DEFINE on all VSAM permanent files after checking for each file that it does not currently exist on the designated catalog.

```
===MOD 2FIC
* $X JOB JNM=$PRFJ.2FIC,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.2FIC        ****** $PROD $PREL ******
*            PERMANENT FILE DEFINITION
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
===SEQ FOR PEI
  LISTC ENT($PXU..$PRO.$PFI.AB)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PFI.AB.$SLIT
  END
  LISTC ENT($PXU..$PRO.$PFI.AC)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PFI.AC.$SLIT
  END
===SEQ
  LISTC ENT($PXY..$PRO.$PRO.AE)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PRO.AE.$SLIT
  END
  LISTC ENT($PXU..$PRO.$PFI.AG)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PFI.AG.$SLIT
  END
  LISTC ENT($PXU..$PRO.$PFI.AJ)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PFI.AJ.$SLIT
  END
  LISTC ENT($PXU..$PRO.$PFI.AN)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PFI.AN.$SLIT
  END
  LISTC ENT($PXY..$PRO.$PRO.AP)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PRO.AP.$SLIT
  END
  LISTC ENT($PXU..$PRO.$PFI.AR)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PFI.AR.$SLIT
  END
  LISTC ENT($MODELF)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DFMODFBK.$SLIT
  END
  LISTC ENT($MODELV)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DFMODVBK.$SLIT
  END
  LISTC ENTRY($PXU..SEQ.$PRO$PFI..OLDGL)
  IF LASTCC EQ 4 THEN DO
    /* --------- DELETE-DEFINE FILE GL (-1) ---- */
 DEL                  ($PXU..SEQ.$PRO$PFI..OLDGL) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..OLDGL) -
                 MODEL($MODELF) -
                 RECFM(FB(90)) -
                 RECSZ(90 90) REC(1000 1000) -
                 VOL($VOLPUD) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..OLDGL.D) )
  END
  LISTC ENTRY($PXU..SEQ.$PRO$PFI..GL)
  IF LASTCC EQ 4 THEN DO
    /* --------- DELETE-DEFINE FILE GL (0) ---- */
 DEL                  ($PXU..SEQ.$PRO$PFI..GL) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..GL) -
                 MODEL($MODELF) -
                 RECFM(FB(90)) -
                 RECSZ(90 90) REC(1000 1000) -
                 VOL($VOLPUD) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..GL.D) )
  END
  LISTC ENTRY($PXU..SEQ.$PRO$PFI..OLDGK)
  IF LASTCC EQ 4 THEN DO
    /* --------- DELETE-DEFINE FILE GK (-1) ---- */
 DEL                  ($PXU..SEQ.$PRO$PFI..OLDGK) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..OLDGK) -
                 MODEL($MODELF) -
                 RECFM(FB(100)) -
                 RECSZ(100 100) REC(1000 1000) -
                 VOL($VOLPUD) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..OLDGK.D) )
  END
  LISTC ENTRY($PXU..SEQ.$PRO$PFI..GK)
  IF LASTCC EQ 4 THEN DO
    /* --------- DELETE-DEFINE FILE GK (0) ---- */
 DEL                  ($PXU..SEQ.$PRO$PFI..GK) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..GK) -
                 MODEL($MODELF) -
                 RECFM(FB(100)) -
                 RECSZ(100 100) REC(1000 1000) -
                 VOL($VOLPUD) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..GK.D) )
  END
  LISTC ENTRY($PXU..$PRO.$PRO.GS)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PRO.GS.$SLIT
  END
  LISTC ENTRY($PXY..$PRO.$PRO.PA)
  IF LASTCC EQ 4 THEN DO
* $X SLI MEM=DF$PRO.$PRO.PA.$SLIT
  END
/*
// EXEC LISTLOG
/&
* $X EOJ
```

4.19. 4.2 CATALOGING OF GPRT PROCEDURE

4.2 CATALOGING OF THE GPRT PROCEDURE JCL

The JCL Module '2GPR' executes, as a SLI member, the cataloging of the JCL for the generation-print (GPRT) procedure. This method is provided only with GPRT because of its complex and frequent use; however, it can be extended to all VA PAC procedures.

```
===MOD 2GPR
* $X JOB JNM=$PRFJ.2GPR,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.2GPR     ****** $PROD $PREL ******
/* ************************************************************
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
// DLBL $LSLI,'$SLIL',,VSAM
// EXEC LIBR,PARM='AC S=$LSLI$SLIS;CA GPRT$PRO$PFI..$SLIT R=Y'
/* ************************************************************
*           GENERATION AND PRINTING
/* ************************************************************
* $P SLI MEM=DBJB$PRO.$PFI..$SLIT
/. JCLST02
* --- STEP 02 --- PTU001
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..WGPRT.INPUT',,VSAM
// EXEC PTU001,SIZE=AUTO
* $X DATA GPRTDATA
/*
/. JCLST03
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL OUT,'$XW..WGPRT.SYSPAF',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE (SYSIPT) RUS
999999999999
/*
/. JCLST04
* --- STEP 04 --- PACB
// DLBL IJSYSUC,'$VCAP',,VSAM
===SEQ FOR PEI
// DLBL PAC7AB,'$PXU..$PRO.$PFI.AB',,VSAM
// DLBL PAC7AC,'$PXU..$PRO.$PFI.AC',,VSAM
===SEQ
// DLBL PAC7AE,'$PXY..$PRO.$PRO.AE',,VSAM
// DLBL PAC7AG,'$PXU..$PRO.$PFI.AG',,VSAM
// DLBL PAC7AN,'$PXU..$PRO.$PFI.AN',,VSAM
// DLBL PAC7AP,'$PXY..$PRO.$PRO.AP',,VSAM
// DLBL PAC7AR,'$PXU..$PRO.$PFI.AR',,VSAM
// DLBL PAC7EB,'$XW..WGPRT.EB',,VSAM
// DLBL PAC7EE,'$XW..WGPRT.EE',,VSAM
// DLBL PAC7EG,'$XW..WGPRT.EG',,VSAM
// DLBL PAC7EN,'$XW..WGPRT.EN',,VSAM
// DLBL PAC7EP,'$XW..WGPRT.EP',,VSAM
// DLBL PAC7EQ,'$XW..WGPRT.EQ',,VSAM
// DLBL PAC7ER,'$XW..WGPRT.ER',,VSAM
// DLBL PAC7EV,'$XW..WGPRT.EV',,VSAM
// DLBL PAC7GI,'$XW..WGPRT.GI',,VSAM
// DLBL PAC7GK,'$PXU..SEQ.$PRO$PFI..GK',,VSAM
// DLBL PAC7GL,'$PXU..SEQ.$PRO$PFI..GL',,VSAM
// DLBL PAC7GM,'$XW..WGPRT.GM',,VSAM
// DLBL PAC7GN,'$XW..WGPRT.GN',,VSAM
// DLBL PAC7GO,'$XW..WGPRT.GO',,VSAM
// DLBL PAC7GS,'$PXU..$PRO.$PRO.GS',,VSAM
// DLBL PAC7GT,'$XW..WGPRT.GT',,VSAM
// DLBL PAC7G6,'$XW..WGPRT.G6',,VSAM
// DLBL PAC7JG,'$XW..WGPRT.JG',,VSAM
// DLBL PAC7KB,'$XW..WGPRT.KB',,VSAM
// DLBL PAC7KD,'$XW..WGPRT.KD',,VSAM
// DLBL PAC7KE,'$XW..WGPRT.KE',,VSAM
// DLBL PAC7KF,'$XW..WGPRT.KF',,VSAM
// DLBL PAC7KG,'$XW..WGPRT.KG',,VSAM
// DLBL PAC7KM,'$XW..WGPRT.KM',,VSAM
// DLBL PAC7KN,'$XW..WGPRT.KN',,VSAM
// DLBL PAC7KP,'$XW..WGPRT.KP',,VSAM
// DLBL PAC7KQ,'$XW..WGPRT.KQ',,VSAM
// DLBL PAC7KR,'$XW..WGPRT.KR',,VSAM
// DLBL PAC7KS,'$XW..WGPRT.KS',,VSAM
// DLBL PAC7KU,'$XW..WGPRT.KU',,VSAM
// DLBL PAC7KV,'$XW..WGPRT.KV',,VSAM
// DLBL PAC7LG,'$PXU..SEQ.$PRO$PFI..OLDGL',,VSAM
// DLBL PAC7LI,'$XW..WGPRT.LI',,VSAM
// DLBL PAC7LK,'$PXU..SEQ.$PRO$PFI..OLDGK',,VSAM
// DLBL PAC7ME,'$XW..WGPRT.INPUT',,VSAM
// DLBL PAC7MG,'$XW..WGPRT.MG',,VSAM
// DLBL PAC7SC,'$PXY..$PRO.$PRO.SC',,VSAM
// DLBL PAC7SG,'$PXY..$PRO.$PRO.SG',,VSAM
// DLBL PAC7SO,'$XW..WGPRT.SO',,VSAM
// DLBL PAC7SR,'$PXY..$PRO.$PRO.SR',,VSAM
// DLBL PAC7SS,'$PXY..$PRO.$PRO.SS',,VSAM
// DLBL PAC7W1,'$XW..WGPRT.W1',,VSAM
// DLBL PAC7W2,'$XW..WGPRT.W2',,VSAM
// DLBL PAC7W3,'$XW..WGPRT.W3',,VSAM
// DLBL PAC7W4,'$XW..WGPRT.W4',,VSAM
// DLBL PAC7W5,'$XW..WGPRT.W5',,VSAM
// DLBL PAC7W6,'$XW..WGPRT.W6',,VSAM
// DLBL PAC7W7,'$XW..WGPRT.W7',,VSAM
// DLBL PAC7W8,'$XW..WGPRT.W8',,VSAM
// DLBL PAC7W9,'$XW..WGPRT.W9',,VSAM
// DLBL SY8PAF,'$XW..WGPRT.SYSPAF',,VSAM
// EXEC PROC=$SORTWK CALL SORTWORK DEFINITION
// EXEC PACB,SIZE=(PACE80,250K)
/. JCLST05
* --- STEP 05 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7EB,'$XW..WGPRT.EB',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7EE,'$XW..WGPRT.EE',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7EG,'$XW..WGPRT.EG',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7EN,'$XW..WGPRT.EN',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7EP,'$XW..WGPRT.EP',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7EQ,'$XW..WGPRT.EQ',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7ER,'$XW..WGPRT.ER',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7EV,'$XW..WGPRT.EV',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7GI,'$XW..WGPRT.GI',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7GM,'$XW..WGPRT.GM',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7GN,'$XW..WGPRT.GN',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7JG,'$XW..WGPRT.JG',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KB,'$XW..WGPRT.KB',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KD,'$XW..WGPRT.KD',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KE,'$XW..WGPRT.KE',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KF,'$XW..WGPRT.KF',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KG,'$XW..WGPRT.KG',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KM,'$XW..WGPRT.KM',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KN,'$XW..WGPRT.KN',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KP,'$XW..WGPRT.KP',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KQ,'$XW..WGPRT.KQ',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KR,'$XW..WGPRT.KR',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KS,'$XW..WGPRT.KS',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KU,'$XW..WGPRT.KU',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7KV,'$XW..WGPRT.KV',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7LI,'$XW..WGPRT.LI',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7ME,'$XW..WGPRT.INPUT',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7MG,'$XW..WGPRT.MG',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7W1,'$XW..WGPRT.W1',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7W2,'$XW..WGPRT.W2',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7W3,'$XW..WGPRT.W3',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7W4,'$XW..WGPRT.W4',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7W6,'$XW..WGPRT.W6',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7W7,'$XW..WGPRT.W7',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7W8,'$XW..WGPRT.W8',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7W9,'$XW..WGPRT.W9',,VSAM,DISP=(,DELETE,DELETE)
// DLBL SY8PAF,'$XW..WGPRT.SYSPAF',,VSAM,DISP=(,DELETE,DELETE)
// EXEC IDCAMS,SIZE=AUTO
 VERIFY FILE (PAC7EB)
 VERIFY FILE (PAC7EE)
 VERIFY FILE (PAC7EG)
 VERIFY FILE (PAC7EN)
 VERIFY FILE (PAC7EP)
 VERIFY FILE (PAC7EQ)
 VERIFY FILE (PAC7ER)
 VERIFY FILE (PAC7EV)
 VERIFY FILE (PAC7GI)
 VERIFY FILE (PAC7GM)
 VERIFY FILE (PAC7GN)
 VERIFY FILE (PAC7JG)
 VERIFY FILE (PAC7KB)
 VERIFY FILE (PAC7KD)
 VERIFY FILE (PAC7KE)
 VERIFY FILE (PAC7KF)
 VERIFY FILE (PAC7KG)
 VERIFY FILE (PAC7KM)
 VERIFY FILE (PAC7KN)
 VERIFY FILE (PAC7KP)
 VERIFY FILE (PAC7KQ)
 VERIFY FILE (PAC7KR)
 VERIFY FILE (PAC7KS)
 VERIFY FILE (PAC7KU)
 VERIFY FILE (PAC7KV)
 VERIFY FILE (PAC7ME)
 VERIFY FILE (PAC7MG)
 VERIFY FILE (PAC7LI)
 VERIFY FILE (PAC7W1)
 VERIFY FILE (PAC7W2)
 VERIFY FILE (PAC7W3)
 VERIFY FILE (PAC7W4)
 VERIFY FILE (PAC7W6)
 VERIFY FILE (PAC7W7)
 VERIFY FILE (PAC7W8)
 VERIFY FILE (PAC7W9)
 VERIFY FILE (SY8PAF)
/*
/. JCLST06
* --- STEP 06 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL IN,'$PXU..SEQ.$PRO$PFI..GL',,VSAM
// DLBL OUT,'$PXU..SEQ.$PRO$PFI..OLDGL',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE (IN) RUS
/*
/. JCLST07
* --- STEP 07 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL IN,'$PXU..SEQ.$PRO$PFI..GK',,VSAM
// DLBL OUT,'$PXU..SEQ.$PRO$PFI..OLDGK',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE (IN) RUS
/*
/+
/&
* $X EOJ
```

4.20. 4.3 CATALOGING OF PACX PROCEDURE

4.3 CATALOGING OF THE PACX PROCEDURE JCL

The JCL module '2PAC' executes, as a SLI member, the cataloging of the JCL for the generation-print procedure (PACX). This method is provided only in standard to complex and frequently used procedures; however, it can be extended to all VA Pac procedures.

```
===MOD 2PAC
* $X JOB JNM=$PRFJ.2PAC,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.2PAC     ****** $PROD $PREL ******
*            EXTRACTION OF DATABASE VA PAC
/* ************************************************************
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
// DLBL $LSLI,'$SLIL',,VSAM
// EXEC LIBR,PARM='AC S=$LSLI$SLIS;CA PACX$PRO$PFI..$SLIT R=Y'
/* ************************************************************
*           EXTRACTIONS
/* ************************************************************
* $P SLI MEM=DBJB$PRO.$PFI..$SLIT
/. JCLST02
* --- STEP 02 --- PTU001
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..WPACX.INPUT',,VSAM
// EXEC PTU001,SIZE=AUTO
* $X DATA PACXDATA
* <------------------  INPUT STREAM
/*
/. JCLST03
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL OUT,'$XW..WPACX.SYSEXT',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE (SYSIPT) RUS
999999999999
/*
/. JCLST04
* --- STEP 04 --- PACX
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7AE,'$PXY..$PRO.$PRO.AE',,VSAM
// DLBL PAC7AN,'$PXU..$PRO.$PFI.AN',,VSAM
// DLBL PAC7AR,'$PXU..$PRO.$PFI.AR',,VSAM
// DLBL PAC7BM,'$XW..WPACX.BM',,VSAM
// DLBL PAC7GY,'$PXU..RPACX.GY',,VSAM
// DLBL PAC7MB,'$XW..WPACX.INPUT',,VSAM
// DLBL PAC7MJ,'$XW..WPACX.MJ',,VSAM
// DLBL PAC7MM,'$XW..WPACX.MM',,VSAM
// DLBL PAC7MR,'$PXU..RPACX.MR',,VSAM
// DLBL PAC7MV,'$PXU..RPACX.MV',,VSAM
// DLBL PAC7PJ,'$PXU..SEQ.$PRO$PFI..PJ',,VSAM
// DLBL PAC7RE,'$XW..WPACX.RE',,VSAM
// DLBL PAC7RM,'$XW..WPACX.RM',,VSAM
// DLBL PAC7TD,'$PXU..RPACX.TD',,VSAM
// DLBL PAC7UE,'$PXU..RPACX.UE',,VSAM
// DLBL PAC7WD,'$XW..WPACX.WD',,VSAM
// DLBL SY8EXT,'$XW..WPACX.SYSEXT',,VSAM
// EXEC PROC=$SORTWK CALL SORTWORK DEFINITION
// EXEC PACX,SIZE=(PACS30,250K)
/. JCLST05
* --- STEP 05 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7BM,'$XW..WPACX.BM',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7MJ,'$XW..WPACX.MJ',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7MM,'$XW..WPACX.MM',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7RE,'$XW..WPACX.RE',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7RM,'$XW..WPACX.RM',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7TE,'$XW..WPACX.TE',,VSAM,DISP=(,DELETE,DELETE)
// DLBL PAC7WD,'$XW..WPACX.WD',,VSAM,DISP=(,DELETE,DELETE)
// DLBL SY8EXT,'$XW..WPACX.SYSEXT',,VSAM,DISP=(,DELETE,DELETE)
// EXEC IDCAMS,SIZE=AUTO
 VERIFY FILE (PAC7BM)
 VERIFY FILE (PAC7MJ)
 VERIFY FILE (PAC7MM)
 VERIFY FILE (PAC7RE)
 VERIFY FILE (PAC7RM)
 VERIFY FILE (PAC7TE)
 VERIFY FILE (PAC7WD)
 VERIFY FILE (PAC7MB)
 VERIFY FILE (SY8EXT)
```

```
/*
/+
/&
* $X EOJ
```

4.21. 4.4 DEFINITION OF GPRT WORK FILES

4.4 DEFINITION OF THE GPRT WORK FILES

The JCL Module '2WGP' defines the GPRT work files which, being very numerous, would seriously impact system performance if they had to be defined each time. It also defines the PACSQL work file.

If '%' is used as the first character of the temporary file index (Job 'JOB2', parameter ===PRM XW), which allows simultaneous submission of GPRT in several partitions, the procedure '2WGP' should be performed in each partition where the procedures are to be executed. Because these files are defined with the NOALLOCATION option, they only take up disk space when they are being used.

```
===MOD 2WGP
* $X JOB JNM=$PRFJ.2WGP,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.2WGP        ****** $PROD $PREL ******
*            GPRT WORK FILES DEFINITION                             *
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- DEFINE GPRT WORK FILES
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  /*  --- DEFINE GPRT WORK FILE EB ---  */
  DEL          ($XW..WGPRT.EB) CL
  DEF CL (NAME ($XW..WGPRT.EB) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.EB.D) )
  /*  --- DEFINE GPRT WORK FILE EE ---  */
  DEL          ($XW..WGPRT.EE) CL
  DEF CL (NAME ($XW..WGPRT.EE) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.EE.D) )
  /*  --- DEFINE GPRT WORK FILE EG ---  */
  DEL          ($XW..WGPRT.EG) CL
  DEF CL (NAME ($XW..WGPRT.EG) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.EG.D) )
  /*  --- DEFINE GPRT WORK FILE EN ---  */
  DEL          ($XW..WGPRT.EN) CL
  DEF CL (NAME ($XW..WGPRT.EN) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.EN.D) )
  /*  --- DEFINE GPRT WORK FILE EP ---  */
  DEL          ($XW..WGPRT.EP) CL
  DEF CL (NAME ($XW..WGPRT.EP) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.EP.D) )
  /*  --- DEFINE GPRT WORK FILE EQ ---  */
  DEL          ($XW..WGPRT.EQ) CL
  DEF CL (NAME ($XW..WGPRT.EQ) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.EQ.D) )
  /*  --- DEFINE GPRT WORK FILE ER ---  */
  DEL          ($XW..WGPRT.ER) CL
  DEF CL (NAME ($XW..WGPRT.ER) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.ER.D) )
  /*  --- DEFINE GPRT WORK FILE EV ---  */
  DEL          ($XW..WGPRT.EV) CL
  DEF CL (NAME ($XW..WGPRT.EV) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.EV.D) )
  /*  --- DEFINE GPRT WORK FILE GI ---  */
  DEL          ($XW..WGPRT.GI) CL
  DEF CL (NAME ($XW..WGPRT.GI) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(080 080) -
          RECFM(FB(080)) -
      ) DATA(NAME ($XW..WGPRT.GI.D) )
  /*  --- DEFINE GPRT WORK FILE GM ---  */
  DEL          ($XW..WGPRT.GM) CL
  DEF CL (NAME ($XW..WGPRT.GM) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(080 080) -
          RECFM(FB(080)) -
      ) DATA(NAME ($XW..WGPRT.GM.D) )
  /*  --- DEFINE GPRT WORK FILE GN ---  */
  DEL          ($XW..WGPRT.GN) CL
  DEF CL (NAME ($XW..WGPRT.GN) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(265 265) -
          RECFM(FB(265)) -
      ) DATA(NAME ($XW..WGPRT.GN.D) )
  /*  --- DEFINE GPRT WORK FILE GO ---  */
  DEL          ($XW..WGPRT.GO) CL
  DEF CL (NAME ($XW..WGPRT.GO) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(204 204) -
          RECFM(FB(204)) -
      ) DATA(NAME ($XW..WGPRT.GO.D) )
  /*  --- DEFINE GPRT WORK FILE GT ---  */
  DEL          ($XW..WGPRT.GT) CL
  DEF CL (NAME ($XW..WGPRT.GT) -
          MODEL($MODELF) -
          REC  (2000 2000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.GT.D) )
  /*  --- DEFINE GPRT WORK FILE INPUT ---  */
  DEL          ($XW..WGPRT.INPUT) CL
  DEF CL (NAME ($XW..WGPRT.INPUT) -
          MODEL($MODELF) -
          REC  (0010 0010) -
          RECSZ(080 080) -
          RECFM(FB(080)) -
      ) DATA(NAME ($XW..WGPRT.INPUT.D) )
  /*  --- DEFINE GPRT WORK FILE JG ---  */
  DEL          ($XW..WGPRT.JG) CL
  DEF CL (NAME ($XW..WGPRT.JG) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(080 080) -
          RECFM(FB(080)) -
      ) DATA(NAME ($XW..WGPRT.JG.D) )
  /*  --- DEFINE GPRT WORK FILE KB ---  */
  DEL          ($XW..WGPRT.KB) CL
  DEF CL (NAME ($XW..WGPRT.KB) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KB.D) )
  /*  --- DEFINE GPRT WORK FILE KD ---  */
  DEL          ($XW..WGPRT.KD) CL
  DEF CL (NAME ($XW..WGPRT.KD) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KD.D) )
  /*  --- DEFINE GPRT WORK FILE KE ---  */
  DEL          ($XW..WGPRT.KE) CL
  DEF CL (NAME ($XW..WGPRT.KE) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KE.D) )
  /*  --- DEFINE GPRT WORK FILE KF ---  */
  DEL          ($XW..WGPRT.KF) CL
  DEF CL (NAME ($XW..WGPRT.KF) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KF.D) )
  /*  --- DEFINE GPRT WORK FILE KG ---  */
  DEL          ($XW..WGPRT.KG) CL
  DEF CL (NAME ($XW..WGPRT.KG) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KG.D) )
  /*  --- DEFINE GPRT WORK FILE KM ---  */
  DEL          ($XW..WGPRT.KM) CL
  DEF CL (NAME ($XW..WGPRT.KM) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KM.D) )
  /*  --- DEFINE GPRT WORK FILE KN ---  */
  DEL          ($XW..WGPRT.KN) CL
  DEF CL (NAME ($XW..WGPRT.KN) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KN.D) )
  /*  --- DEFINE GPRT WORK FILE KP ---  */
  DEL          ($XW..WGPRT.KP) CL
  DEF CL (NAME ($XW..WGPRT.KP) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KP.D) )
  /*  --- DEFINE GPRT WORK FILE KQ ---  */
  DEL          ($XW..WGPRT.KQ) CL
  DEF CL (NAME ($XW..WGPRT.KQ) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KQ.D) )
  /*  --- DEFINE GPRT WORK FILE KR ---  */
  DEL          ($XW..WGPRT.KR) CL
  DEF CL (NAME ($XW..WGPRT.KR) -
          MODEL($MODELF) -
          REC  (6000 6000) -
          RECSZ(180 180) -
          RECFM(FB(180)) -
      ) DATA(NAME ($XW..WGPRT.KR.D) )
  /*  --- DEFINE GPRT WORK FILE KS ---  */
  DEL          ($XW..WGPRT.KS) CL
  DEF CL (NAME ($XW..WGPRT.KS) -
          MODEL($MODELF) -
          REC  (6000 6000) -
```

```
            RECSZ(180 180) -
            RECFM(FB(180)) -
        ) DATA(NAME ($XW..WGPRT.KS.D) )
    /*  --- DEFINE GPRT WORK FILE KU ---  */
    DEL          ($XW..WGPRT.KU) CL
    DEF CL (NAME ($XW..WGPRT.KU) -
            MODEL($MODELF) -
            REC  (6000 6000) -
            RECSZ(180 180) -
            RECFM(FB(180)) -
        ) DATA(NAME ($XW..WGPRT.KU.D) )
    /*  --- DEFINE GPRT WORK FILE KV ---  */
    DEL          ($XW..WGPRT.KV) CL
    DEF CL (NAME ($XW..WGPRT.KV) -
            MODEL($MODELF) -
            REC  (6000 6000) -
            RECSZ(180 180) -
            RECFM(FB(180)) -
        ) DATA(NAME ($XW..WGPRT.KV.D) )
    /*  --- DEFINE GPRT WORK FILE LI ---  */
    DEL          ($XW..WGPRT.LI) CL
    DEF CL (NAME ($XW..WGPRT.LI) -
            MODEL($MODELF) -
            REC  (6000 6000) -
            RECSZ(080 080) -
            RECFM(FB(080)) -
        ) DATA(NAME ($XW..WGPRT.LI.D) )
    /*  --- DEFINE GPRT WORK FILE MG ---  */
    DEL          ($XW..WGPRT.MG) CL
    DEF CL (NAME ($XW..WGPRT.MG) -
            MODEL($MODELF) -
            REC  (3000 3000) -
            RECSZ(150 150) -
            RECFM(FB(150)) -
        ) DATA(NAME ($XW..WGPRT.MG.D) )
    /*  --- DEFINE GPRT WORK FILE SO ---  */
    DEL          ($XW..WGPRT.SO) CL
    DEF CL (NAME ($XW..WGPRT.SO) -
            MODEL($MODELF) -
            REC  (6000 6000) -
            RECSZ(312 312) -
            RECFM(FB(312)) -
        ) DATA(NAME ($XW..WGPRT.SO.D) )
    /*  --- DEFINE GPRT WORK FILE SYSPAF ---  */
    DEL          ($XW..WGPRT.SYSPAF) CL
    DEF CL (NAME ($XW..WGPRT.SYSPAF) -
            SHR(2 3) RUS NAL KEYS(12 0) -
            REC  (1000 1000) -
            RECSZ(170 468) -
            VOL($VOLVWK) -
        ) INDEX(NAME ($XW..WGPRT.SYSPAF.I) -
        ) DATA (NAME ($XW..WGPRT.SYSPAF.D) CISZ(8192) )
    /*  --- DEFINE GPRT WORK FILE W1 ---  */
    DEL          ($XW..WGPRT.W1) CL
    DEF CL (NAME ($XW..WGPRT.W1) -
            MODEL($MODELF) -
            REC  (2000 2000) -
            RECSZ(180 180) -
            RECFM(FB(180)) -
        ) DATA(NAME ($XW..WGPRT.W1.D) )
    /*  --- DEFINE GPRT WORK FILE W2 ---  */
    DEL          ($XW..WGPRT.W2) CL
    DEF CL (NAME ($XW..WGPRT.W2) -
            MODEL($MODELF) -
            REC  (2000 2000) -
            RECSZ(180 180) -
            RECFM(FB(180)) -
        ) DATA(NAME ($XW..WGPRT.W2.D) )
    /*  --- DEFINE GPRT WORK FILE W3 ---  */
    DEL          ($XW..WGPRT.W3) CL
    DEF CL (NAME ($XW..WGPRT.W3) -
            MODEL($MODELF) -
            REC  (2000 2000) -
```

```
              RECSZ(180 180) -
              RECFM(FB(180)) -
          ) DATA(NAME ($XW..WGPRT.W3.D) )
      /*  --- DEFINE GPRT WORK FILE W4 ---  */
      DEL          ($XW..WGPRT.W4) CL
      DEF CL (NAME ($XW..WGPRT.W4) -
              MODEL($MODELF) -
              REC  (5000 5000) -
              RECSZ(090 090) -
              RECFM(FB(090)) -
          ) DATA(NAME ($XW..WGPRT.W4.D) )
      /*  --- DEFINE GPRT WORK FILE W6 ---  */
      DEL          ($XW..WGPRT.W6) CL
      DEF CL (NAME ($XW..WGPRT.W6) -
              MODEL($MODELF) -
              REC  (2000 2000) -
              RECSZ(308 308) -
              RECFM(FB(308)) -
          ) DATA(NAME ($XW..WGPRT.W6.D) )
      /*  --- DEFINE GPRT WORK FILE W7 ---  */
      DEL          ($XW..WGPRT.W7) CL
      DEF CL (NAME ($XW..WGPRT.W7) -
              MODEL($MODELF) -
              REC  (2000 2000) -
              RECSZ(308 308) -
              RECFM(FB(308)) -
          ) DATA(NAME ($XW..WGPRT.W7.D) )
      /*  --- DEFINE GPRT WORK FILE W8 ---  */
      DEL          ($XW..WGPRT.W8) CL
      DEF CL (NAME ($XW..WGPRT.W8) -
              MODEL($MODELF) -
              REC  (2000 2000) -
              RECSZ(308 308) -
              RECFM(FB(308)) -
          ) DATA(NAME ($XW..WGPRT.W8.D) )
      /*  --- DEFINE GPRT WORK FILE W9 ---  */
      DEL          ($XW..WGPRT.W9) CL
      DEF CL (NAME ($XW..WGPRT.W9) -
              MODEL($MODELF) -
              REC  (9000 9000) -
              RECSZ(055 055) -
              RECFM(FB(055)) -
          ) DATA(NAME ($XW..WGPRT.W9.D) )
    /*
    // EXEC LISTLOG
    /&
    * $X EOJ
```

4.22. 4.5 DEFINITION OF PACX WORK FILES

4.5 DEFINITION OF THE PACX WORK FILES

The JCL module '2WPA' defines the PACX work files which, being very numerous, would seriously impact system performance if they had to be defined each time.

If '%' is used as the first character of the temporary file index (job JOB2, parameter ===PRM XW), which allows simultaneous submission of GPRT in several partitions, the '2WPA' procedure should be performed in each partition where the procedures are to be executed. Because these files are to be defined with the 'NOALLOCATION' option, they only take up disk space when they are being used.

```
===MOD 2WPA
* $X JOB JNM=$PRFJ.2WPA,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.2WPA        ****** $PROD $PREL ******
*            PACX WORK FILES DEFINITION
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- DEFINE DES FICHIERS DE TRAVAIL PACX
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  /*  --- DEFINE GPRT WORK FILE BM ---  */
  DEL          ($XW..WPACX.BM) CL
  DEF CL (NAME ($XW..WPACX.BM) -
          MODEL($MODELF) -
          REC  (0010 0010) -
          RECSZ(080 080) -
          RECFM(FB(080)) -
      ) DATA(NAME ($XW..WPACX.BM.D) )
  /*  --- DEFINE GPRT WORK FILE INPUT ---  */
  DEL          ($XW..WPACX.INPUT) CL
  DEF CL (NAME ($XW..WPACX.INPUT) -
          MODEL($MODELF) -
          REC  (0010 0010) -
          RECSZ(080 080) -
          RECFM(FB(080)) -
      ) DATA(NAME ($XW..WPACX.INPUT.D) )
  /*  --- DEFINE GPRT WORK FILE MJ ---  */
  DEL          ($XW..WPACX.MJ) CL
  DEF CL (NAME ($XW..WPACX.MJ) -
          MODEL($MODELF) -
          REC  (1000 1000) -
          RECSZ(152 152) -
          RECFM(FB(152)) -
      ) DATA(NAME ($XW..WPACX.MJ.D) )
  /*  --- DEFINE GPRT WORK FILE MM ---  */
  DEL          ($XW..WPACX.MM) CL
  DEF CL (NAME ($XW..WPACX.MM) -
          MODEL($MODELF) -
          REC  (1000 1000) -
          RECSZ(055 055) -
          RECFM(FB(055)) -
      ) DATA(NAME ($XW..WPACX.MM.D) )
  /*  --- DEFINE GPRT WORK FILE RE ---  */
  DEL          ($XW..WPACX.RE) CL
  DEF CL (NAME ($XW..WPACX.RE) -
          MODEL($MODELF) -
          REC  (1000 1000) -
          RECSZ(012 012) -
          RECFM(FB(012)) -
      ) DATA(NAME ($XW..WPACX.RE.D) )
  /*  --- DEFINE GPRT WORK FILE RM ---  */
  DEL          ($XW..WPACX.RM) CL
  DEF CL (NAME ($XW..WPACX.RM) -
          MODEL($MODELF) -
          REC  (1000 1000) -
          RECSZ(167 167) -
          RECFM(FB(167)) -
      ) DATA(NAME ($XW..WPACX.RM.D) )
  /*  --- DEFINE GPRT WORK FILE SYSEXT ---  */
  DEL          ($XW..WPACX.SYSEXT) CL
  DEF CL (NAME ($XW..WPACX.SYSEXT) -
          SHR(2 3) RUS NAL KEYS(12 0) -
          REC  (1000 1000) -
          RECSZ(112 112) -
          VOL($VOLVWK) -
      ) INDEX(NAME ($XW..WPACX.SYSEXT.I) -
      ) DATA (NAME ($XW..WPACX.SYSEXT.D) CISZ(8192) )
  /*  --- DEFINE GPRT WORK FILE TE ---  */
  DEL          ($XW..WPACX.TE) CL
```

```
      DEF CL (NAME ($XW..WPACX.TE) -
              MODEL($MODELF) -
              REC  (1000 1000) -
              RECSZ(180 180) -
              RECFM(FB(180)) -
          ) DATA(NAME ($XW..WPACX.TE.D) )
      /*  --- DEFINE GPRT WORK FILE WD ---  */
      DEL          ($XW..WPACX.WD) CL
      DEF CL (NAME ($XW..WPACX.WD) -
              MODEL($MODELF) -
              REC  (1000 1000) -
              RECSZ(167 167) -
              RECFM(FB(167)) -
          ) DATA(NAME ($XW..WPACX.WD.D) )
    /*
    // EXEC LISTLOG
    /&
    * $X EOJ
```

4.23. 4.6 INSTALLATION OF EXECUTABLE MODULES

4.6 INSTALLATION OF EXECUTABLE MODULES

Execute job '2PHA'.

```
===MOD 2PHA
* $X JOB JNM=$PRFJ.2PHA,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.2PHA        ****** $PROD $PREL ******
*   MODULE INSTALLATION                  (RESTORE)
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
// DLBL $LMODO,'$MODO',,VSAM
/* GOTO JCLSTXX
* --- STEP 1 --- LIBR
// ASSGN SYS004,$TAP1
// PAUSE MOUNT TAPE $TAPEI ON $TAP1
// MTC REW,SYS004
// MTC FSF,SYS004,20
// EXEC LIBR
 ACCESS SUBLIB=$LMODO$SMODO
 RENAME $PRO.*.PHASE:D2*.PHASE
RESTORE -
 CGI.LB.PACA05.PHASE:$LMODB$SMODB -
 CGI.LB.PACA10.PHASE:$LMODB$SMODB -
 CGI.LB.PACA15.PHASE:$LMODB$SMODB -
 CGI.LB.PACA20.PHASE:$LMODB$SMODB -
 CGI.LB.PACB.PHASE:$LMODB$SMODB -
 CGI.LB.PACB30.PHASE:$LMODB$SMODB -
===SEQ FOR SQL
 CGI.LB.PACB31.PHASE:$LMODB$SMODB -
===SEQ
 CGI.LB.PACB40.PHASE:$LMODB$SMODB -
 CGI.LB.PACB80.PHASE:$LMODB$SMODB -
 CGI.LB.PACCTL.PHASE:$LMODB$SMODB -
 CGI.LB.PACC30.PHASE:$LMODB$SMODB -
 CGI.LB.PACC40.PHASE:$LMODB$SMODB -
 CGI.LB.PACC80.PHASE:$LMODB$SMODB -
 CGI.LB.PACDTP.PHASE:$LMODB$SMODB -
 CGI.LB.PACD30.PHASE:$LMODB$SMODB -
 CGI.LB.PACD40.PHASE:$LMODB$SMODB -
 CGI.LB.PACD80.PHASE:$LMODB$SMODB -
 CGI.LB.PACD90.PHASE:$LMODB$SMODB -
 CGI.LB.PACE30.PHASE:$LMODB$SMODB -
 CGI.LB.PACE40.PHASE:$LMODB$SMODB -
 CGI.LB.PACE80.PHASE:$LMODB$SMODB -
 CGI.LB.PACFGY.PHASE:$LMODB$SMODB -
 CGI.LB.PACFMB.PHASE:$LMODB$SMODB -
 CGI.LB.PACFTD.PHASE:$LMODB$SMODB -
 CGI.LB.PACG3C.PHASE:$LMODB$SMODB -
 CGI.LB.PACG3S.PHASE:$LMODB$SMODB -
 CGI.LB.PACG4S.PHASE:$LMODB$SMODB -
 CGI.LB.PACG8C.PHASE:$LMODB$SMODB -
 CGI.LB.PACG8S.PHASE:$LMODB$SMODB -
===SEQ FOR VISP
 CGI.LB.PACINS.PHASE:$LMODB$SMODB -
===SEQ
 CGI.LB.PACK30.PHASE:$LMODB$SMODB -
 CGI.LB.PACK80.PHASE:$LMODB$SMODB -
 CGI.LB.PACK90.PHASE:$LMODB$SMODB -
 CGI.LB.PACL30.PHASE:$LMODB$SMODB -
 CGI.LB.PACL80.PHASE:$LMODB$SMODB -
 CGI.LB.PACL90.PHASE:$LMODB$SMODB -
 CGI.LB.PACL92.PHASE:$LMODB$SMODB -
 CGI.LB.PACL93.PHASE:$LMODB$SMODB -
 CGI.LB.PACL95.PHASE:$LMODB$SMODB -
 CGI.LB.PACM30.PHASE:$LMODB$SMODB -
 CGI.LB.PACM80.PHASE:$LMODB$SMODB -
 CGI.LB.PACN25.PHASE:$LMODB$SMODB -
 CGI.LB.PACN30.PHASE:$LMODB$SMODB -
 CGI.LB.PACN35.PHASE:$LMODB$SMODB -
 CGI.LB.PACN40.PHASE:$LMODB$SMODB -
 CGI.LB.PACN80.PHASE:$LMODB$SMODB -
 CGI.LB.PACP30.PHASE:$LMODB$SMODB -
 CGI.LB.PACP40.PHASE:$LMODB$SMODB -
 CGI.LB.PACP80.PHASE:$LMODB$SMODB -
 CGI.LB.PACP82.PHASE:$LMODB$SMODB -
 CGI.LB.PACQ.PHASE:$LMODB$SMODB -
===SEQ FOR PEI
 CGI.LB.PACR01.PHASE:$LMODB$SMODB -
 CGI.LB.PACR10.PHASE:$LMODB$SMODB -
 CGI.LB.PACR20.PHASE:$LMODB$SMODB -
 CGI.LB.PACR22.PHASE:$LMODB$SMODB -
 CGI.LB.PACR30.PHASE:$LMODB$SMODB -
 CGI.LB.PACR40.PHASE:$LMODB$SMODB -
 CGI.LB.PACR60.PHASE:$LMODB$SMODB -
 CGI.LB.PACR61.PHASE:$LMODB$SMODB -
 CGI.LB.PACR90.PHASE:$LMODB$SMODB -
===SEQ
 CGI.LB.PACSJO.PHASE:$LMODB$SMODB -
 CGI.LB.PACSMD.PHASE:$LMODB$SMODB -
 CGI.LB.PACSPU.PHASE:$LMODB$SMODB -
 CGI.LB.PACSRM.PHASE:$LMODB$SMODB -
 CGI.LB.PACS30.PHASE:$LMODB$SMODB -
 CGI.LB.PACS40.PHASE:$LMODB$SMODB -
 CGI.LB.PACS50.PHASE:$LMODB$SMODB -
 CGI.LB.PACS60.PHASE:$LMODB$SMODB -
 CGI.LB.PACS75.PHASE:$LMODB$SMODB -
 CGI.LB.PACS80.PHASE:$LMODB$SMODB -
===SEQ FOR TAB
 CGI.LB.PACTIN.PHASE:$LMODB$SMODB -
 CGI.LB.PACTI1.PHASE:$LMODB$SMODB -
 CGI.LB.PACT40.PHASE:$LMODB$SMODB -
 CGI.LB.PACT41.PHASE:$LMODB$SMODB -
 CGI.LB.PACT45.PHASE:$LMODB$SMODB -
 CGI.LB.PACT50.PHASE:$LMODB$SMODB -
 CGI.LB.PACT51.PHASE:$LMODB$SMODB -
===SEQ
 CGI.LB.PACU15.PHASE:$LMODB$SMODB -
 CGI.LB.PACU80.PHASE:$LMODB$SMODB -
 CGI.LB.PACU99.PHASE:$LMODB$SMODB -
 CGI.LB.PACX.PHASE:$LMODB$SMODB -
 CGI.LB.PADM10.PHASE:$LMODB$SMODB -
 CGI.LB.PAFP10.PHASE:$LMODB$SMODB -
 CGI.LB.PAF900.PHASE:$LMODB$SMODB -
===SEQ FOR S2K
 CGI.LB.PAN200.PHASE:$LMODB$SMODB -
 CGI.LB.PAN205.PHASE:$LMODB$SMODB -
 CGI.LB.PAN210.PHASE:$LMODB$SMODB -
 CGI.LB.PAN212.PHASE:$LMODB$SMODB -
 CGI.LB.PAN215.PHASE:$LMODB$SMODB -
 CGI.LB.PAN220.PHASE:$LMODB$SMODB -
 CGI.LB.PAN230.PHASE:$LMODB$SMODB -
 CGI.LB.PAN240.PHASE:$LMODB$SMODB -
 CGI.LB.PAN250.PHASE:$LMODB$SMODB -
 CGI.LB.PAN255.PHASE:$LMODB$SMODB -
 CGI.LB.PAN260.PHASE:$LMODB$SMODB -
 CGI.LB.PAN270.PHASE:$LMODB$SMODB -
 CGI.LB.PAN280.PHASE:$LMODB$SMODB -
===SEQ
 CGI.LB.PBBTST.PHASE:$LMODB$SMODB -
 CGI.LB.PBBTWS.PHASE:$LMODB$SMODB -
 CGI.LB.PBBT98.PHASE:$LMODB$SMODB -
===SEQ FOR PACTR
 CGI.LB.PDS600.PHASE:$LMODB$SMODB -
 CGI.LB.PDS610.PHASE:$LMODB$SMODB -
===SEQ
 CGI.LB.PREI00.PHASE:$LMODB$SMODB -
 CGI.LB.PREI40.PHASE:$LMODB$SMODB -
 CGI.LB.PREI50.PHASE:$LMODB$SMODB -
 CGI.LB.PRE986.PHASE:$LMODB$SMODB -
===SEQ FOR TAB
 CGI.LB.PTARSD.PHASE:$LMODB$SMODB -
 CGI.LB.PTAR20.PHASE:$LMODB$SMODB -
 CGI.LB.PTASVD.PHASE:$LMODB$SMODB -
===SEQ
===SEQ FOR TEAM
 CGI.LB.PTC010.PHASE:$LMODB$SMODB -
 CGI.LB.PTC030.PHASE:$LMODB$SMODB -
 CGI.LB.PTC100.PHASE:$LMODB$SMODB -
 CGI.LB.PTC200.PHASE:$LMODB$SMODB -
 CGI.LB.PTC220.PHASE:$LMODB$SMODB -
 CGI.LB.PTC400.PHASE:$LMODB$SMODB -
 CGI.LB.PTC440.PHASE:$LMODB$SMODB -
===SEQ
 CGI.LB.PTED30.PHASE:$LMODB$SMODB -
 CGI.LB.PTED60.PHASE:$LMODB$SMODB -
 CGI.LB.PTEP90.PHASE:$LMODB$SMODB -
 CGI.LB.PTEXD0.PHASE:$LMODB$SMODB -
 CGI.LB.PTEX30.PHASE:$LMODB$SMODB -
 CGI.LB.PTEX80.PHASE:$LMODB$SMODB -
 CGI.LB.PTUBAS.PHASE:$LMODB$SMODB -
 CGI.LB.PTUCR1.PHASE:$LMODB$SMODB -
 CGI.LB.PTUCR2.PHASE:$LMODB$SMODB -
 CGI.LB.PTUCSS.PHASE:$LMODB$SMODB -
 CGI.LB.PTUD10.PHASE:$LMODB$SMODB -
 CGI.LB.PTUD20.PHASE:$LMODB$SMODB -
 CGI.LB.PTUD30.PHASE:$LMODB$SMODB -
 CGI.LB.PTUESS.PHASE:$LMODB$SMODB -
===SEQ FOR PACTR
 CGI.LB.PTUG05.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG06.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG07.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG10.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG11.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG12.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG42.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG44.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG46.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG50.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG60.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG61.PHASE:$LMODB$SMODB -
 CGI.LB.PTUG90.PHASE:$LMODB$SMODB -
===SEQ
 CGI.LB.PTULOI.PHASE:$LMODB$SMODB -
 CGI.LB.PTULVB.PHASE:$LMODB$SMODB -
 CGI.LB.PTUN00.PHASE:$LMODB$SMODB -
 CGI.LB.PTUN10.PHASE:$LMODB$SMODB -
 CGI.LB.PTUN40.PHASE:$LMODB$SMODB -
 CGI.LB.PTUQ10.PHASE:$LMODB$SMODB -
 CGI.LB.PTUQ15.PHASE:$LMODB$SMODB -
 CGI.LB.PTUQ20.PHASE:$LMODB$SMODB -
 CGI.LB.PTUQ24.PHASE:$LMODB$SMODB -
 CGI.LB.PTUQ25.PHASE:$LMODB$SMODB -
 CGI.LB.PTUQ30.PHASE:$LMODB$SMODB -
 CGI.LB.PTUR00.PHASE:$LMODB$SMODB -
 CGI.LB.PTU001.PHASE:$LMODB$SMODB -
 CGI.LB.PTU004.PHASE:$LMODB$SMODB -
 CGI.LB.PTU100.PHASE:$LMODB$SMODB -
 CGI.LB.PTU120.PHASE:$LMODB$SMODB -
 CGI.LB.PTU130.PHASE:$LMODB$SMODB -
 CGI.LB.PTU140.PHASE:$LMODB$SMODB -
 CGI.LB.PTU2CL.PHASE:$LMODB$SMODB -
 CGI.LB.PTU200.PHASE:$LMODB$SMODB -
 CGI.LB.PTU208.PHASE:$LMODB$SMODB -
 CGI.LB.PTU210.PHASE:$LMODB$SMODB -
 CGI.LB.PTU220.PHASE:$LMODB$SMODB -
 CGI.LB.PTU240.PHASE:$LMODB$SMODB -
 CGI.LB.PTU300.PHASE:$LMODB$SMODB -
 CGI.LB.PTU320.PHASE:$LMODB$SMODB -
 CGI.LB.PTU380.PHASE:$LMODB$SMODB -
 CGI.LB.PTU400.PHASE:$LMODB$SMODB -
 CGI.LB.PTU402.PHASE:$LMODB$SMODB -
 CGI.LB.PTU420.PHASE:$LMODB$SMODB -
 CGI.LB.PTU500.PHASE:$LMODB$SMODB -
 CGI.LB.PTU502.PHASE:$LMODB$SMODB -
 CGI.LB.PTU550.PHASE:$LMODB$SMODB -
 CGI.LB.PTU560.PHASE:$LMODB$SMODB -
 CGI.LB.PTU630.PHASE:$LMODB$SMODB -
 CGI.LB.PTU640.PHASE:$LMODB$SMODB -
 CGI.LB.PTU810.PHASE:$LMODB$SMODB -
 CGI.LB.PTU815.PHASE:$LMODB$SMODB -
```

```
     CGI.LB.PTU850.PHASE:$LMODB$SMODB -
     CGI.LB.PTU855.PHASE:$LMODB$SMODB -
    ===SEQ FOR ROLD
     CGI.LB.PTU908.PHASE:$LMODB$SMODB -
    ===SEQ
    ===SEQ FOR VISP
     CGI.LB.PVA100.PHASE:$LMODB$SMODB -
     CGI.LB.PVA110.PHASE:$LMODB$SMODB -
     CGI.LB.PVA300.PHASE:$LMODB$SMODB -
     CGI.LB.PVA305.PHASE:$LMODB$SMODB -
     CGI.LB.PVA310.PHASE:$LMODB$SMODB -
     CGI.LB.PVA320.PHASE:$LMODB$SMODB -
     CGI.LB.PVA400.PHASE:$LMODB$SMODB -
    ===SEQ
     CGI.LB.PYSMCC.PHASE:$LMODB$SMODB -
     CGI.LB.PYSMC2.PHASE:$LMODB$SMODB -
     CGI.LB.PYSMC3.PHASE:$LMODB$SMODB -
    ===SEQ FOR ROLD
     CGI.LB.REP2PJ.PHASE:$LMODB$SMODB -
    ===SEQ
     CGI.LB.UTIXSR.PHASE:$LMODB$SMODB -
    ===SEQ FOR PUF
     CGI.LB.D2FANM.PHASE:$LMODO$SMODO -
     CGI.LB.D2FBIB.PHASE:$LMODO$SMODO -
     CGI.LB.D2FCTL.PHASE:$LMODO$SMODO -
     CGI.LB.D2FDBD.PHASE:$LMODO$SMODO -
     CGI.LB.D2FECR.PHASE:$LMODO$SMODO -
     CGI.LB.D2FENU.PHASE:$LMODO$SMODO -
     CGI.LB.D2FFOG.PHASE:$LMODO$SMODO -
     CGI.LB.D2FMCL.PHASE:$LMODO$SMODO -
     CGI.LB.D2FPGM.PHASE:$LMODO$SMODO -
     CGI.LB.D2FRUB.PHASE:$LMODO$SMODO -
     CGI.LB.D2FSDO.PHASE:$LMODO$SMODO -
     CGI.LB.D2FTXT.PHASE:$LMODO$SMODO -
     CGI.LB.D2FVER.PHASE:$LMODO$SMODO -
     CGI.LB.D2F000.PHASE:$LMODO$SMODO -
    ===SEQ
     CGI.LB.D2PA00.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA01.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA10.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA11.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA12.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA13.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA14.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA15.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA16.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA17.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA18.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA19.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA20.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA21.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA22.PHASE:$LMODO$SMODO -
    ===SEQ FOR PEI
     CGI.LB.D2PA30.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA31.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA32.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA33.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA34.PHASE:$LMODO$SMODO -
     CGI.LB.D2PA35.PHASE:$LMODO$SMODO -
    ===SEQ
     CGI.LB.D2PBND.PHASE:$LMODO$SMODO -
     CGI.LB.D2PPHP.PHASE:$LMODO$SMODO -
     CGI.LB.D2QA00.PHASE:$LMODO$SMODO -
     CGI.LB.D2QB00.PHASE:$LMODO$SMODO -
     CGI.LB.D2QC00.PHASE:$LMODO$SMODO -
     CGI.LB.D2QC01.PHASE:$LMODO$SMODO -
     CGI.LB.D2QC50.PHASE:$LMODO$SMODO -
     CGI.LB.D2QD00.PHASE:$LMODO$SMODO -
     CGI.LB.D2QE00.PHASE:$LMODO$SMODO -
     CGI.LB.D2QF00.PHASE:$LMODO$SMODO -
     CGI.LB.D2QF10.PHASE:$LMODO$SMODO -
     CGI.LB.D2QG00.PHASE:$LMODO$SMODO -
     CGI.LB.D2QH00.PHASE:$LMODO$SMODO -
```

```
  CGI.LB.D2QH01.PHASE:$LMODO$SMODO -
  CGI.LB.D2QH20.PHASE:$LMODO$SMODO -
  CGI.LB.D2QH30.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI00.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI01.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI02.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI03.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI04.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI05.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI20.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI21.PHASE:$LMODO$SMODO -
  CGI.LB.D2QI50.PHASE:$LMODO$SMODO -
  CGI.LB.D2QK10.PHASE:$LMODO$SMODO -
  CGI.LB.D2QK20.PHASE:$LMODO$SMODO -
  CGI.LB.D2QK30.PHASE:$LMODO$SMODO -
  CGI.LB.D2QL10.PHASE:$LMODO$SMODO -
  CGI.LB.D2QL20.PHASE:$LMODO$SMODO -
  CGI.LB.D2QL21.PHASE:$LMODO$SMODO -
  CGI.LB.D2QL30.PHASE:$LMODO$SMODO -
 ===SEQ FOR SQL
  CGI.LB.D2QL40.PHASE:$LMODO$SMODO -
  CGI.LB.D2QL41.PHASE:$LMODO$SMODO -
  CGI.LB.D2QL45.PHASE:$LMODO$SMODO -
  CGI.LB.D2QL46.PHASE:$LMODO$SMODO -
 ===SEQ
  CGI.LB.D2QM00.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP00.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP01.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP02.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP03.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP04.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP05.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP06.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP07.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP08.PHASE:$LMODO$SMODO -
  CGI.LB.D2QP50.PHASE:$LMODO$SMODO -
  CGI.LB.D2QR00.PHASE:$LMODO$SMODO -
  CGI.LB.D2QS02.PHASE:$LMODO$SMODO -
  CGI.LB.D2QS03.PHASE:$LMODO$SMODO -
  CGI.LB.D2QS04.PHASE:$LMODO$SMODO -
  CGI.LB.D2QS05.PHASE:$LMODO$SMODO -
  CGI.LB.D2QS06.PHASE:$LMODO$SMODO -
  CGI.LB.D2QS08.PHASE:$LMODO$SMODO -
  CGI.LB.D2QS09.PHASE:$LMODO$SMODO -
  CGI.LB.D2QS10.PHASE:$LMODO$SMODO -
  CGI.LB.D2QT00.PHASE:$LMODO$SMODO -
  CGI.LB.D2QT10.PHASE:$LMODO$SMODO -
  CGI.LB.D2QT20.PHASE:$LMODO$SMODO -
  CGI.LB.D2QT50.PHASE:$LMODO$SMODO -
  CGI.LB.D2QUPA.PHASE:$LMODO$SMODO -
  CGI.LB.D2QUPF.PHASE:$LMODO$SMODO -
  CGI.LB.D2QU00.PHASE:$LMODO$SMODO -
  CGI.LB.D2QU01.PHASE:$LMODO$SMODO -
  CGI.LB.D2QU10.PHASE:$LMODO$SMODO -
  CGI.LB.D2QU20.PHASE:$LMODO$SMODO -
  CGI.LB.D2QV10.PHASE:$LMODO$SMODO -
  CGI.LB.D2QV20.PHASE:$LMODO$SMODO -
  CGI.LB.D2QV30.PHASE:$LMODO$SMODO -
  CGI.LB.D2QX00.PHASE:$LMODO$SMODO -
  CGI.LB.D2QX01.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY01.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY02.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY03.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY04.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY05.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY10.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY11.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY20.PHASE:$LMODO$SMODO -
  CGI.LB.D2QY30.PHASE:$LMODO$SMODO -
  CGI.LB.D2QZ00.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q000.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q100.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q101.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q102.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q103.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q104.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q200.PHASE:$LMODO$SMODO -
 ===SEQ FOR TAB
  CGI.LB.D2Q210.PHASE:$LMODO$SMODO -
 ===SEQ
  CGI.LB.D2Q300.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q400.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q500.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q600.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q700.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q800.PHASE:$LMODO$SMODO -
  CGI.LB.D2Q900.PHASE:$LMODO$SMODO -
  CGI.LB.D2R000.PHASE:$LMODO$SMODO -
  CGI.LB.D2R005.PHASE:$LMODO$SMODO -
  CGI.LB.D2R100.PHASE:$LMODO$SMODO -
  CGI.LB.D2R200.PHASE:$LMODO$SMODO -
  CGI.LB.D2R400.PHASE:$LMODO$SMODO -
  CGI.LB.D2R500.PHASE:$LMODO$SMODO -
  CGI.LB.D2R600.PHASE:$LMODO$SMODO -
  CGI.LB.D2R980.PHASE:$LMODO$SMODO -
  CGI.LB.D2TPST.PHASE:$LMODO$SMODO -
  CGI.LB.D2TPWS.PHASE:$LMODO$SMODO -
  CGI.LB.D2UCTA.PHASE:$LMODO$SMODO -
  CGI.LB.D2UCTR.PHASE:$LMODO$SMODO -
  CGI.LB.D2UCTX.PHASE:$LMODO$SMODO -
  TAPE=SYS004 REPLACE=Y SCAN=NO
  ACCESS SUBLIB=$LMODO$SMODO
  RENAME D2*.PHASE:$PRO.*.PHASE
 /*
 // EXEC LISTLOG
 /&
 * $X EOJ
 ===SEQ
```

4.24. 4.7 LOADING OF SYSTEM FILES

4.7 LOADING OF THE SYSTEM FILES

The JCL Module '3SYS' contains the JCL to define and load the following system files:

```
SC  :  Batch program generation skeleton file
SG  :  On-line program generation skeleton file
SR  :  COBOL generator skeleton file
AE0 :  VA PAC  error messages database
SP  :  Variable skeleton file (PAF Extension)
SF  :  Fixed skeleton file (PAF Extension)
SS  :  Skeleton file of the Client/Server generator
VGE :  Visualage Dictionary
TCO :  TeamConnection Dictionary
```

```
===MOD 3SYS
* $X JOB JNM=$PRFJ.3SYS,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.3SYS        ****** $PROD $PREL ******
*   PACBASE SYSTEM FILES INITIALIZATION AND LOADING           *
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* -- -STEP 01 --- DEFINE PACBASE SYSTEM VSAM FILES
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=D$PRO.$PRO.AE0.$SLIT
* $X SLI MEM=D$PRO.$PRO.VGE.$SLIT
* $X SLI MEM=D$PRO.$PRO.TCO.$SLIT
* $X SLI MEM=DF$PRO.$PRO.SC.$SLIT
* $X SLI MEM=DF$PRO.$PRO.SF.$SLIT
* $X SLI MEM=DF$PRO.$PRO.SG.$SLIT
* $X SLI MEM=DF$PRO.$PRO.SP.$SLIT
* $X SLI MEM=DF$PRO.$PRO.SR.$SLIT
* $X SLI MEM=DF$PRO.$PRO.SS.$SLIT
/*
/. JCLST02
* --- STEP 02 --- UNLOAD SKELETON FILE FOR BATCH GENERATION
// DLBL IJSYSUC,'$VCAP',,VSAM
// ASSGN SYS004,$TAP1
// PAUSE MOUNT TAPE $TAPEI ON $TAP1
 MTC REW,SYS004
 MTC FSF,SYS004,3
// TLBL IN
// DLBL OUT,'$PXY..$PRO.$PRO.SC',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (3204) -
       BLKSZ($BLKSC) NLBL)) -
       OFILE (OUT) RUS
/*
/. JCLST03
* --- STEP 03 --- UNLOAD SKELETON FILE FOR ONLINE GENERATION
// DLBL IJSYSUC,'$VCAP',,VSAM
 MTC REW,SYS004
 MTC FSF,SYS004,4
// TLBL IN
// DLBL OUT,'$PXY..$PRO.$PRO.SG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (4605) -
       BLKSZ($BLKSG)  NLBL)) -
       OFILE (OUT) RUS
/*
/. JCLST04
* --- STEP 04 --- UNLOAD SKELETON FILE FOR COBOL GENERATOR
// DLBL IJSYSUC,'$VCAP',,VSAM
 MTC REW,SYS004
 MTC FSF,SYS004,5
// TLBL IN
// DLBL OUT,'$PXY..$PRO.$PRO.SR',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (4605) -
       BLKSZ($BLKSR)  NLBL)) -
       OFILE (OUT) RUS
/*
/. JCLST05
* --- STEP 05 --- UNLOAD ERROR MESSAGES FILE
// DLBL IJSYSUC,'$VCAP',,VSAM
 MTC REW,SYS004
 MTC FSF,SYS004,6
// TLBL IN
// DLBL OUT,'$PXY..SEQ.$PRO$PRO..AE0',,VSAM
// DLBL OUT2,'$PXY..$PRO.$PRO.AE',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (80) -
       BLKSZ($BLKAE)  NLBL)) -
       OFILE (OUT) RUS
 REPRO IFILE(OUT) OFILE(OUT2) REP
/*
/. JCLST06
* --- STEP 06 --- UNLOAD SKELETON FOR PAF-PDM EXTENSION
// DLBL IJSYSUC,'$VCAP',,VSAM
 MTC REW,SYS004
 MTC FSF,SYS004,8
// TLBL IN
// DLBL OUT,'$PXY..$PRO.$PRO.SP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (4605) -
       BLKSZ($BLKSP)  NLBL)) -
       OFILE (OUT) RUS
/*
/. JCLST07
* --- STEP 07 --- UNLOAD FIXED SKELETON FOR PAF-PDM EXTENSION
// DLBL IJSYSUC,'$VCAP',,VSAM
 MTC REW,SYS004
 MTC FSF,SYS004,9
// TLBL IN
// DLBL OUT,'$PXY..$PRO.$PRO.SF',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (119)  -
       BLKSZ($BLKSF)  NLBL)) -
       OFILE (OUT) RUS
/*
/. JCLST08
* --- STEP 08 --- UNLOAD SERVER/REQUESTER SKELETON FILE
// DLBL IJSYSUC,'$VCAP',,VSAM
 MTC REW,SYS004
 MTC FSF,SYS004,10
// TLBL IN
// DLBL OUT,'$PXY..$PRO.$PRO.SS',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (4605)  -
       BLKSZ($BLKSS)  NLBL)) -
       OFILE (OUT) RUS
/*
/. JCLST09
* --- STEP 09 --- UNLOAD VISUALAGE DICTIONARY
// DLBL IJSYSUC,'$VCAP',,VSAM
 MTC REW,SYS004
 MTC FSF,SYS004,18
// TLBL IN
// DLBL OUT,'$PXY..$PRO.$PRO.VGE',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (117)  -
       BLKSZ($BLKVG)  NLBL)) -
       OFILE (OUT) RUS
/*
/. JCLST10
* --- STEP 10 --- UNLOAD TEAMCONNECTION DICTIONARY
// DLBL IJSYSUC,'$VCAP',,VSAM
 MTC REW,SYS004
 MTC FSF,SYS004,19
// TLBL IN
// DLBL OUT,'$PXY..$PRO.$PRO.TCO',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO IFILE (IN ENV (PDEV (2400) RECFM(FB) RECSZ (117)  -
       BLKSZ($BLKTC)  NLBL)) -
       OFILE (OUT) RUS
/*
// EXEC LISTLOG
/&
* $X EOJ
```

4.25. 4.8 LOADING OF USER FILES

4.8 LOADING OF USER FILES

The JCL module '3USR' contains the initialization JCL of some user files:

AP	:	user parameters
PA	:	work file (PAF)
GS	:	extraction paths

```
===MOD 3USR
* $X JOB JNM=$PRFJ.3USR,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.3USR        ****** $PROD $PREL ******
*   INITIALIZATION/UNLOADING OF PACBASE USER FILES
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 1 --- INITIALIZE AP FILE
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL OUT,'$PXY..$PRO.$PRO.AP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE(SYSIPT) RUS
9999999
/*
/. JCLST02
* --- STEP 2 --- INITIALIZE PA FILE
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL OUT,'$PXY..$PRO.$PRO.PA',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE(SYSIPT) RUS
999999999999
/*
/. JCLST03
* --- STEP 3 --- INITIALIZE GS FILE
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL OUT,'$PXU..$PRO.$PRO.GS',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE(SYSIPT) RUS
9999999999999999999999999
/*
// EXEC LISTLOG
/&
* $X EOJ
```

4.26. 4.9 LOADING OF ERROR MESSAGES

4.9 LOADING OF USER ERROR MESSAGES AND PARAMETER FILES

This is done via the PARM procedure.

```
***********************************************************
IMPORTANT - IMPORTANT - IMPORTANT - IMPORTANT - IMPORTANT -
***********************************************************

FOR A PROPER OPERATION OF THE SYSTEM, VA PAC  MUST CONTAIN
AT LEAST THE FOLLOWING USER PARAMETERS:
```

- USER CODE

- VA PAC ACCESS KEY

Default entries provided include:

- One NU line: common user code with all database access rights.

  Name: TEST - Password: (blank)

  This code is used to perform the installation tests and should be deleted once the user code for the manager of the PABASE database(s) is entered.

- One NK line: INSERT HERE THE VA PAC ACCESS KEY PROVIDED WITH THE INSTALLATION LEAFLET.

WORKSTATION OPERATION: CHOICE OF METHODOLOGY PARAMETERS

Specify the parameters corresponding to the methodology or methodologies used with the site's Workstation installation.

Update transactions are found in the JCL '9DSP'.

The various versions of these user entities are selected via the MER, YSM, SSA, IFW and OMT variants, in the MM1JCL utility commands.

In case of an erroneous selection of no variant, or too many variants, run the JOB2 installation job again, with the proper ===SELV cards and the ===SELM 9DSP card.

4.27. 5.1 LOADING OF TEST DATABASE

5.1 LOADING OF THE TEST DATABASE

Job '5TPC' performs the loading of the test database backup provided with the product. This database must then be restored using the REST procedure.

This test database is provided for a limited number of initial operation tests only.

```
===MOD 5TPC
* $X JOB JNM=$PRFJ.5TPC,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.5TPC        ****** $PROD $PREL ******
*            LOADING THE VA Pacbase TEST DATABASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=DF$PRO.$PFI.PC.$SLIT
/*
/. JCLST02
* --- STEP 01 --- MOUNT INSTALLATION TAPE
// DLBL IJSYSUC,'$VCAP',,VSAM
// ASSGN SYS004,$TAP1
// PAUSE MOUNT PACBASE TAPE  $TAPEI  ON  $TAP1
 MTC REW,SYS004
 MTC FSF,SYS004,7
// TLBL IN
// DLBL OUT,'$PXU..SEQ.$PRO$PFI..NEWPC,,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO INFILE (IN ENV (PDEV (2400) RECFM(VB) RECSZ (155) -
       BLKSZ($BLKPC) NLBL)) -
       OUTFILE (OUT)
/*
/. JCLST03
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.PC.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
```

4.28. 5.2 INITIALIZATION OF GENERATION-PRINT REQUESTS

5.2 INITIALIZATION OF THE GENERATION-PRINT REQUESTS

This is performed via the execution of the REAG procedure with an initialization command of the AG file (AGI).

4.29. 5.3 INITIALIZATION OF PEI MODULE

5.3 INITIALIZATION OF THE PEI FUNCTION

For the PEI Function only, initialization is performed by running the INPE procedure, followed by the RSPE procedure.

4.30. 6.1 COMPLEMENT: VISUAL DICTIONARY INSTALLATION

6.1 INITIALIZATION OF THE VISUAL DICTIONARY

Loading of the Visual objects as user entities in the VA PAC Dictionary is performed through the VINS procedure.
This is to be carried out only by users of the VISP variant.

4.31. 6.2.1 COMPLEMENT: PAF ENVIRONMENT

6.2.1 CREATION OF THE PAF DICTIONARY

The '9PAF' JCL module contains the VA PAC transactions that should be specified as input for the UPDT procedure, in order to copy the PAF dictionary standard entities into your own database. This dictionary will be used to develop PAF queries.

4.32. 6.2.2 COMPLEMENT: PAF+ EXTENSION

6.2.2 COMPLEMENT: PAF+ EXTENSION

The following elements must be installed in order to operate the PAF+ Extension:

. User Entity .PPTEX

. SP and SF skeleton files

. GS user file containing the Extraction Master Paths

User definition of an Extraction Master Path is done through an occurrence of the User entity (UE).

This UE is copied by the installation process in the '9XPD' JCL module.

This member contains BATCH transactions which will be input in the Database via the UPDT batch update procedure.

Once you have chosen a library to host the UE, modify the PAFPTEX member with a text editor by filling in the '*' line located at the beginning of the transactions, then execute the UPDT procedure.

The PAF skeletons are installed as a default. They are:

. SP, which translates the User Entity Occurrence into PAF queries.

. SF, which generates a COBOL program. This program, once translated by the PAFP10 program, will be a User Extractor or a Macro-Command called in the Master Outline.

The GS file contains the user Extraction Master Paths.

An example of an execution JCL for a User Extractor can be found in the PAFX JCL module described thereafter.

```
*  EXAMPLE OF PAF+ USER EXTRACTOR
/*  SUBSTITUTIONS TO BE MADE
/*   %EXTRAC  = EXTRACTOR CODE
/*   %RECSZ   = RECORDSIZE OF EXTRACTED FILE PAC7SQ
/*  INPUT     : COMMAND FOR THE EXTRACTION
/*  COL. 2    :   'X'
/*  COL. 3    :   PAF CURSOR CODE
/*  COL. 7    :   UEO CODE
/*  COL. 15   :   BEGIN-DELIMITER
/*  COL. 23   :   END-DELIMITER
/*  COL. 31   :   DEBUG TYPE (0 OR 1)
/*  COL. 32   :   NUMBER OF PAF BUFFERS
/* ************************************************************
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  /*  --- DEFINE "INPUT" TEMPORARY FILE --- */
  DEL          ($XW..W$MODUL..INPUT) CL PURGE
  DEF CL (NAME ($XW..W$MODUL..INPUT) -
          MODEL($MODELF) -
          REC  (200 200) ) -
        DATA (NAME ($XW..W$MODUL..INPUT.D) )
  /*  --- DEFINE SO TEMPORARY FILE ---  */
  DEL          ($PXU..R$MODUL..SO) CL PURGE
  DEF CL (NAME ($PXU..R$MODUL..SO) -
          MODEL($MODELF) -
          REC  (100 100) -
          RECSZ(436 436) -
          RECFM(FB(436))) -
        DATA (NAME ($PXU..R$MODUL..SO.D) )
  /*  --- DEFINE SO TEMPORARY FILE ---  */
  DEL          ($PXU..R$MODUL..SQ) CL PURGE
  DEF CL (NAME ($PXU..R$MODUL..SQ) -
          MODEL($MODELF) -
          REC  (100 100) -
          RECSZ(%RECSZ %RECSZ) -
          RECFM(FB(%RECSZ))) -
        DATA (NAME ($PXU..R$MODUL..SQ.D) )
* $X SLI MEM=DFSYSPAF.$SLIT
/*
/. JCLST02
* --- STEP 02 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL OUT,'$XW..SYSPAF',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE(SYSIPT) RUS
999999999999
/*
* --- STEP 03 --- PTU001
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM
// EXEC PTU001,SIZE=AUTO
 *USER    PASSWORDLIB
 XTTTTCCCCCCCCBBBBBBBBEEEEEEEE1000100
/*
/. JCLST04
* --- STEP 04 --- %EXTRAC
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7AE,'$PXY..$PRO.$PRO.AE',,VSAM
// DLBL PAC7AN,'$PXU..$PRO.$PFI.AN',,VSAM
// DLBL PAC7AR,'$PXU..$PRO.$PFI.AR',,VSAM
// DLBL SY8PAF,'$XW..SYSPAF',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM,DISP=(,DELETE,KEEP)
// DLBL PAC7SO,'$PXU..R$MODUL..SO',,VSAM
// DLBL PAC7SQ,'$PXU..R$MODUL..SQ',,VSAM
// EXEC PROC=$SORTWK CALL SORTWORK DEFINITION
// EXEC %EXTRAC,SIZE=(AUTO,50K)
/*
/. STEPEND
* --- STEP 05 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
```

```
      DEL          ($XW..W$MODUL..INPUT) CL
      DEL          ($XW..SYSPAF)      CL
    /*
```

4.33. 6.3 COMPLEMENT: WORKSTATION

6.3 CREATION OF WORKSTATION USER ENTITIES

The JCL Module '9DSG' contains the VA PAC transactions to be used as input to the UPDT procedure, before the VA PAC WorkStation is used.

There are three 'versions' of these user entities. They are selected by the 'MER', 'YSM or 'SSA' variant, in the MM1JCL commands.

If, by mistake, neither variant or more than one variant were selected, re-run the installation job 'JOB2' with the appropriate ===SELV cards and the ===SELM 9DSG card.

4.34. 6.4 COMPLEMENT: PACBENCH QUALITY CONTROL

6.4 PACBENCH QUALITY CONTROL CREATION OF THE COMPILED QUALITY RULE FILE

The JCL module '9PQR' contains a set of compiled standard quality rules. This file can be copied onto a SAM-ESDS file, with a fixed length of 80, to be used in the PQCA quality control procedure (step 4, DLBL name PACQMF).

INSTALLATION OF THE USER ENTITY

The PERSONALIZATION option of the PACBENCH QUALITY CONTROL Function requires, for its operation, that a certain User Entity be present in the Database.

This User Entity is provided with the installation package, in the '9PQU' JCL module, in the form of batch transactions which should be copied in the Database via the UPDT batch update procedure.

Once you have chosen the Database library in which the User Entity will be copied, you should edit the '9PQU' JCL in a text editor, and complete the '*'-line at the beginning of the transactions, then run the UPDT procedure.

4.35. 6.5 COMPLEMENT: CONTROL CARD INSTALLATION

6.5 INSTALLATION OF OPTIONAL CONTROL CARDS

The JCL Module '9INC' contains a stream of optional control cards to compile batch or on-line programs, with or without PAF (PACSQL) pre-processor or Documented COBOL (DCOB).

If the user parameter (AP) file does not yet contain these cards, they can be installed. If the file does contain the cards, the line numbers should be verified so that they are inserted without interfering with other control card sets.

```
===MOD 9INC
*            CONTROL CARD INITIALIZATION
 NCD0  COMPILATION OF BATCH COBOL PROGRAM (BEFORE)
 NCF0  COMPILATION OF BATCH COBOL PROGRAM (AFTER)
 NCD1  COMPILATION OF OLSD COBOL PROGRAM (BEFORE)
 NCF1  COMPILATION OF OLSD COBOL PROGRAM (AFTER)
 NCD2  COMPILATION OF BATCH PROGRAM WITH PAF (BEFORE)
 NCF2  COMPILATION OF BATCH PROGRAM WITH PAF (AFTER)
 NCD3  COMPILATION OF BATCH COBOL PROGRAM WITH DCOB (BEFORE)
 NCF3  COMPILATION OF BATCH COBOL PROGRAM WITH DCOB (AFTER)
 NCD5  BMS COMPILATION MAP (BEFORE)
 NCF5  BMS COMPILATION MAP (AFTER)
 NCD7  COMPILATION OF ON-LINE PROGRAM WITH PAF (BEFORE)
 NCF7  COMPILATION OF ON-LINE PROGRAM WITH PAF (AFTER)
 NCD000/* COMPILATION OF A COBOL BATCH PROGRAM (TOP)
 NCD010* $X JOB JNM=$PRFJ.COB0,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'
 NCD012* $X LST CLASS=$PCL,DISP=$PDL.$PWL
 NCD019// JOB COB0 COMPIL COBOL BATCH (-/-/- OF -)   $Y   PAGD -
 NCD020* - : -                                       $Y   SR   -
 NCD025// OPTION $OPT
 NCD040* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
 NCD080// OPTION CATAL
 NCD082 PHASE -,*                                    $Y   P    -
 NCD085 MODE RMODE(ANY)
 NCD086// DLBL LUSER,'-',,VSAM                       $Y   1    -
 NCD087// LIBDEF PHASE,CATALOG=LUSER.-               $Y   2    -
 NCD089* --- STEP 1 --- IGYCRCTL
 NCD090// EXEC IGYCRCTL,SIZE=IGYCRCTL
 NCD097 CBL NOADV,APOST,RES,LIB,RENT,NOXREF,TRUNC(BIN)
 NCF000/* COMPILATION OF A COBOL BATCH PROGRAM (BOTTOM)
 NCF010* --- STEP 2 --- LNKEDT
 NCF030// EXEC LNKEDT
XNCF089/&
 NCF099* $X EOJ
 NCD100/* COMPILATION OF AN ONLINE COBOL PROGRAM (TOP)
 NCD110* $X JOB JNM=$PRFJ.COB1,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'
 NCD112* $X LST CLASS=$PCL,DISP=$PDL.$PWL
 NCD119// JOB COB1 COMPIL COBOL ONLINE   (-/-/- OF -)$Y   PAGD -
 NCD120* - : -                                       $Y   SR   -
 NCD125// OPTION $OPT
 NCD140* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
 NCD142// DLBL LUSER,'-',,VSAM                       $Y   1    -
 NCD144// LIBDEF PHASE,CATALOG=LUSER.-               $Y   2    -
 NCD150* --- STEP 1 --- DFHECP1$
 NCD155// ASSGN SYS091,SYSPCH
 NCD160// EXEC PROC=$$PACWK1,ASSGN=SYSPCH
 NCD190// EXEC DFHECP1$
 NCD197 CBL APOST,RES,LIB,RENT,NOXREF,TRUNC(BIN)
 NCD199 CBL XOPTS(CICS,OPT,NOLIST)
 NCF100/* COMPILATION OF AN ONLINE COBOL PROGRAM (BOTTOM)
 NCF110* --- STEP 2 --- IGYCRCTL
 NCF115 CLOSE SYSPCH,SYS091
 NCF120// EXEC PROC=$$PACWK1,ASSGN=SYSIPT
 NCF130// OPTION CATAL
 NCF132 PHASE -,*                                    $Y   P    -
 NCF134 INCLUDE DFHECI
 NCF140// EXEC IGYCRCTL,SIZE=IGYCRCTL
 NCF141/*
 NCF150* --- STEP 3 --- LNKEDT
 NCF190// EXEC LNKEDT
XNCF196/&
 NCF197 CLOSE SYSIPT,SYSRDR
 NCF199* $X EOJ
 NCD200/* COMPILATION OF A COBOL BATCH PROGRAM WITH PAF (TOP)
 NCD210* $X JOB JNM=$PRFJ.COB2,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'
 NCD212* $X LST CLASS=$PCL,DISP=$PDL.$PWL
 NCD219// JOB COB2 COMPIL COBOL WITH PAF (-/-/- OF -)$Y   PAGD -
 NCD220* - : -                                       $Y   SR   -
 NCD225// OPTION $OPT
 NCD240* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
 NCD248// OPTION CATAL
 NCD250 PHASE -,*                                    $Y   P    -
 NCD255 MODE RMODE(ANY)
 NCD260// DLBL LUSER,'-',,VSAM                       $Y   1    -
 NCD283// LIBDEF PHASE,CATALOG=LUSER.-               $Y   2    -
 NCD290* $X SLI MEM=PAFB$PRO.$PFI..$SLIT
 NCD293* $X DATA INPUT
 NCF200/* COMPILATION OF A BATCH PROGRAM WITH PAF (BOTTOM)
 NCF299* $X EOJ
 NCD300/* COMPILATION OF A BATCH COBOL PROGRAM WITH DCOB (TOP)
 NCD219// JOB COB3 COMPIL COBOL WITH DCOB(-/-/- OF -)$Y   PAGD -
 NCD300/* COMPILATION OF A BATCH COBOL PROGRAM WITH DCOB (BOT)
 NCD500/* BMS COMPILATION MAP (BEFORE)
 NCD510* $X JOB JNM=$PRFJ.MAP5,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'
 NCD512* $X LST CLASS=$PCL,DISP=$PDL.$PWL
 NCD519// JOB MAP5 COMPIL MAP BMS     (-/-/- LE -)   $Y   PAGD -
 NCD520* - : -                                       $Y   SR   -
 NCD525// OPTION $OPT
 NCD552* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
 NCD555// DLBL LUSER,'-',,VSAM                       $Y   1    -
 NCD563// LIBDEF PHASE,CATALOG=LUSER.-               $Y   2    -
 NCD564// LIBDEF SOURCE,SEARCH=LUSER.-               $Y   2    -
 NCD565* --- STEP 1 --- LIBR
 NCD570// EXEC LIBR,PARM='AC S=LUSER.-;CA -.A R=Y'   $Y   3P   -
 NCF500/+ BMS COMPILATION MAP (AFTER)
 NCF505/*
 NCF510* --- STEP 2 --- ASSEMBLY
 NCF515// OPTION CATAL,SYSPARM='MAP',ALIGN
 NCF520 PHASE   -,*                                  $Y   P    -
 NCF530// EXEC ASSEMBLY
 NCF535 PRINT ON,NOGEN
 NCF540 COPY  -                                      $Y   P    -
XNCF545 AIF ('&SYSPARM'(1,3) NE 'MAP').G565
 NCF550 DS 0D
 NCF552* PACBASE MAP CONSTANTS
 NCF555 DC CL8'WORKING'
 NCF560 DC CL5'-',CL3'-',CL8'-',CL6'-',CL8'-'        $Y   GADSU-
 NCF565.G565 ANOP
 NCF570 END
 NCF575/*
 NCF576* --- STEP 3 --- LNKEDT
 NCF577// EXEC LNKEDT
 NCF579/*
 NCF580* --- STEP 4 --- LIBR
 NCF582// DLBL IJSYSUC,'$VCAP',,VSAM
 NCF585// DLBL LUSER,'-',,VSAM                       $Y   1    -
 NCF587// EXEC LIBR,PARM='AC S=LUSER.-;DEL -.A'      $Y   2P   -
 NCF588/*
XNCF589/&
 NCF599* $X EOJ
 NCD700/* ON-LINE PROGRAM COMPILE WITH PAF (BEFORE)
 NCD710* $X JOB JNM=$PRFJ.COB7,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'
 NCD712* $X LST CLASS=$PCL,DISP=$PDL.$PWL
 NCD719// JOB COB7 COBOL COMPIL WITH PAF (-/-/- LE -)$Y   PAGD -
 NCD720* - : -                                       $Y   SR   -
 NCD725// OPTION $OPT
 NCD740* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
 NCD748// OPTION CATAL
 NCD750 PHASE -,*                                    $Y   P    -
 NCD760// DLBL LUSER,'-'                             $Y   1    -
 NCD783// LIBDEF PHASE,CATALOG=LUSER.-               $Y   2    -
 NCD790* $X SLI MEM=PAFT$PRO.$PFI..$SLIT
 NCD793* $X DATA INPUT
 NCF700/* ON-LINE PROGRAM COMPILATION WITH PAF (AFTER)
 NCF799* $X EOJ
 NCD300/* COMPILATION OF PROGRAM COBOL BATCH WITH DCOB (BEFORE)
 NCD310* $X JOB JNM=$PRFJ.COB3,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'
 NCD312* $X LST CLASS=$PCL,DISP=$PDL.$PWL
 NCD313* $X LST CLASS=$PCL,DISP=$PDL,LST=$LST2$PWL
 NCD319// JOB COB3 COMPIL COBOL AVEC DCOB(-/-/- LE -)$Y   PAGD -
 NCD320* - : -                                       $Y   SR   -
 NCD325// OPTION $OPT
 NCD343* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
 NCD344// OPTION CATAL
 NCD345 PHASE -,*                                    $Y   P    -
 NCD346 MODE RMODE(ANY)
 NCD348// DLBL LUSER,'-',,VSAM                       $Y   1    -
 NCD349// LIBDEF PHASE,CATALOG=LUSER.-               $Y   2    -
 NCD350* --- STEP 1 --- IGYCRCTL
 NCD351// ASSGN SYS091,SYSLST
 NCD352// EXEC PROC=$$PACWK2,ASSGN=SYSLST
 NCD390// EXEC IGYCRCTL,SIZE=IGYCRCTL
 NCD397 CBL NOADV,APOST,RES,LIB,RENT,NOXREF,TRUNC(BIN)
 NCF300/* COMPILATION OF PROGRAM COBOL BATCH WITH DCOB (AFTER)
 NCF310* --- STEP 2 --- LNKEDT
 NCF330// EXEC LNKEDT
 NCF389/*
 NCF390 CLOSE SYSLST,SYS091
 NCF392// ASSGN SYS007,$LST2
 NCF397* $X SLI MEM=DCOB$PRO.$PFI..$SLIT
XNCF398/&
 NCF399* $X EOJ
/*
```

4.36. 6.6 COMPLEMENT: DCOB CATALOGING PROCEDURE

6.6 CATALOGING OF THE DCOB PROCEDURE (DOCUMENTED COBOL)

The JCL Module 'DCOB' contains the source of the SLI procedure DCOB$PRO.$PFI which, once it has been cataloged, can be run by the optional control cards supplied (call code '3').

```
===MOD DCOB
* $X JOB JNM=$PRFJ.DCOB,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.DCOB     ****** $PROD $PREL ******
/* ************************************************************
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
// DLBL $LSLI,'$SLIL',,VSAM
// EXEC LIBR,PARM='AC S=$LSLI$SLIS;CA DCOB$PRO$PFI..$SLIT R=Y'
/* ************************************************************
/* ************************************************************
*            DOCUMENTED COBOL
/* ************************************************************
/* WITH THE STANDARD JCL PROCEDURE $$PACWK2, IF USED IN THE
/* COMPILATION JOB WITH THE PARAMETERS :
/*
/*          DLBL=IJSYSLS,ASSGN=SYSLST,OPT=PERM
/*
/* YOU CAN RETRIEVE THE COMPILATION LISTING ON DISK.
/* IN THIS CASE, IF THIS JOB RUNS IN A PARTITION DIFFERENT FROM
/* THE PARTITION IN WHICH THE COMPILATION JOB WAS RUN,
/* IT IS NECESSARY TO OVERRIDE '$$PACWK2' BY 'PACWK2',
/* 'PACWK2' AND SO ON, ACCORDING TO BG, F2 AND SO ON.
/* THE DISK FILE PAC8SY IS FORMAT UNDEFINED, BLKSZ=133
/* ************************************************************
* $P SLI MEM=DBJB$PRO.$PFI..$SLIT
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE 'INPUT' TEMPORARY FILE   --- */
 DEL                  ($XW..W$MODUL..INPUT) CL
 DEF CL         (NAME ($XW..W$MODUL..INPUT) -
                 MODEL($MODELF) -
                 REC  (1 200) -
      ) DATA    (NAME ($XW..W$MODUL..INPUT.D) )
/*
/. JCLST02
* --- STEP 02 --- PTU001
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM
// EXEC PTU001,SIZE=AUTO
FIRST004007          COORDINATES 1ST COBOL LINE ON THE PAGE
WINDW004064009       WINDOW (1ST LINE, BEHIND, 1ST COL)
NUMBR065             NUMBER OF LINES/PAGE
NULGG00105000001     NUM OF THE GENERATED LINE (COL, LTH, INCR)
SAUTP1               SKIP CHARACTERS AND CORRESPONDING SKIP
SAUTL+00
SAUTL 01
SAUTL002
SAUTL-03
/*
/. JCLST03
* --- STEP 03 --- PTUCR1
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAQYLC,'$XW..W$MODUL..LC',,VSAM,DISP=NEW
/*
/. JCLST04
* --- STEP 04 --- PTUD10
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC8FA,'$XW..W$MODUL..FA',,VSAM,DISP=(NEW,,DELETE)
// DLBL PAC8FC,'$XW..W$MODUL..INPUT',,VSAM
// DLBL PAC8LC,'$XW..W$MODUL..LC',,VSAM
// DLBL PAC8PG,'$XW..W$MODUL..PG',,VSAM,DISP=(NEW,,DELETE)
// DLBL PAC8RA,'$XW..W$MODUL..RA',,VSAM,DISP=(NEW,,DELETE)
/* DLBL PAC8SY           IF PAC8SY ON DISK (TO COMPLETE)
/* EXTENT SYS017,......
/* ASSGN  SYS017,DISK,VOL=......,SHR
// EXEC PROC=$$PACWK2,DLBL=PAC8SY,ASSGN=SYS017 <<--ALSO DISK
/* PAC8SY ON TAPE             SYS016 ASSIGNED
// ON $RC GT 4 GOTO STEPEND
// EXEC PTUD10,SIZE=AUTO
/*
```

```
// ON $RC<16 CONTINUE
/. JCLST05
* --- STEP 05 --- PTUD20
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7AE,'$PXY..$PRO.$PRO.AE',,VSAM
// DLBL PAC7AN,'$PXU..$PRO.$PFI.AN',,VSAM
// DLBL PAC7AR,'$PXU..$PRO.$PFI.AR',,VSAM
// DLBL PAC8FB,'$XW..W$MODUL..PG',,VSAM,DISP=(,DELETE,KEEP)
// DLBL PAC8FE,'$XW..W$MODUL..FA',,VSAM,DISP=(,DELETE,KEEP)
// DLBL PAC8FS,'$XW..W$MODUL..FS',,VSAM,DISP=(NEW,,DELETE)
// DLBL PAC8LC,'$XW..W$MODUL..LC',,VSAM
// EXEC PROC=$SORTWK CALL SORTWORK DEFINITION
// ON $RC GT 4 GOTO STEPEND
// EXEC PTUD20,SIZE=(AUTO,50K)
/*
// ON $RC<16 CONTINUE
/*
/. JCLST06
* --- STEP 06 --- SORT
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL SORTIN1,'$XW..W$MODUL..FS',,VSAM
// DLBL SORTOUT,'$XW..W$MODUL..KS',,VSAM,DISP=NEW
// EXEC PROC=$SORTWK CALL SORTWORK DEFINITION
// EXEC SORT,SIZE=(AUTO,50K)
 SORT FIELDS=(1,15,CH,A)
 RECORD TYPE=F,LENGTH=(100)
 INPFIL VSAM
 OUTFIL KSDS
/*
/. JCLST07
* --- STEP 07 --- PTUD30
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC8FA,'$XW..W$MODUL..KS',,VSAM
// DLBL PAC8FC,'$XW..W$MODUL..INPUT',,VSAM,DISP=(,DELETE,KEEP)
// DLBL PAC8LC,'$XW..W$MODUL..LC',,VSAM
// DLBL PAC8RA,'$XW..W$MODUL..RA',,VSAM,DISP=(,DELETE,KEEP)
/* DLBL PAC8SY          IF PAC8SY ON DISK (TO COMPLETE)
/* EXTENT SYS016,......
/* ASSGN  SYS016,DISK,VOL=......,SHR
// EXEC PROC=$$PACWK2,DLBL=PAC8SY,ASSGN=SYS016 <<--ALSO DISK
/* PAC8SY ON TAPE             SYS015 ASSIGNED
// EXEC PTUD30,SIZE=AUTO
/. JCLST08
* --- STEP 08 --- PTUCR2
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAQYLC,'$XW..W$MODUL..LC',,VSAM,DISP=(,DELETE,KEEP)
// DLBL PAQYQN,'$XW..W$MODUL..QN',,VSAM
// EXEC PTUCR2,SIZE=AUTO
/. STEPEND
* --- STEP 09 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 DEL                  ($XW..W$MODUL..INPUT) CL
/*
/+
/&
* $X EOJ
```

4.37. 7. MODIFICATION OF THE CICS JCL

7. MODIFICATION OF THE CICS START-UP JCL

Once the CICS tables are updated and compiled, and the files correctly defined, the CICS start-up JCL must be completed with the JCL Module '6JCI', which contains:

- The DLBL of the VA PAC On-Line Module library

- The DLBL of the VSAM catalogue used for VA PAC

- The DLBL of each database's files: AN, AR, AJ, and AG

- The DLBL of system files: AE, AP, and PA

- The DLBL of additional option files: AB and AC for PEI

This module does NOT contain the //LIBDEF PHASE,SEARCH=.... card which is already included in the JCL. This card must be completed by the call of the .Sublib VA PAC On-Line Module library.

```
===MOD 6JCI
/* ***************************************************************
/*      --- MODIFICATIONS TO THE CICS EXECUTION JCL              *
* ===============================================================
*      ADD     LCO$PRO$SMODO  IN THE LIBDEF PHASE,SEARCH CARD
* ===============================================================
/* ***************************************************************
// DLBL $LMODO,'$MODO',,VSAM
// DLBL PACCT,'$VCAP',,VSAM
===SEQ FOR PEI
// DLBL $PRO$PFI.AB,'$PXU..$PRO$PFI.AB',,VSAM,CAT=PACCT
// DLBL $PRO$PFI.AC,'$PXU..$PRO$PFI.AC',,VSAM,CAT=PACCT
===SEQ
// DLBL $PRO$PRO.AE,'$PXY..$PRO$PRO.AE',,VSAM,CAT=PACCT
// DLBL $PRO$PFI.AG,'$PXU..$PRO$PFI.AG',,VSAM,CAT=PACCT
// DLBL $PRO$PFI.AJ,'$PXU..$PRO$PFI.AJ',,VSAM,CAT=PACCT
// DLBL $PRO$PFI.AN,'$PXU..$PRO$PFI.AN',,VSAM,CAT=PACCT
// DLBL $PRO$PRO.AP,'$PXY..$PRO$PRO.AP',,VSAM,CAT=PACCT
// DLBL $PRO$PFI.AR,'$PXU..$PRO$PFI.AR',,VSAM,CAT=PACCT
// DLBL $PRO$PRO.PA,'$PXY..$PRO$PRO.PA',,VSAM,CAT=PACCT
```

4.38. 8.1 COMPLEMENT: Pac/Impact MODULE

```
8.1. COMPLEMENT : Pac/Impact FUNCTION
     -------------------------------------------

Execute the jobs:
                  'INFO'
                  'INFP'
                  'INFQ'
```

```
===SEQ FOR S2K
===MOD INFO
* $X JOB JNM=$PRFJ.INFO,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.INFO        ****** $PROD $PREL ******
*                INITIALIZATION OF 'FO' FILE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE BACKUP FILE        FO --- */
 DEL                  ($PXU..SEQ.$PRO$PFI..NEWFO) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..NEWFO) -
                 MODEL($MODELF) -
                 REC  (1000 1000) -
                 RECSZ(260 260) -
                 RECFM(FB(260)) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..NEWFO.D) )
/*
/. STEPEND
* --- STEP 02 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.FO.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

```
===SEQ FOR S2K
===MOD INFP
* $X JOB JNM=$PRFJ.INFP,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.INFP        ****** $PROD $PREL ******
*                INITIALIZATION OF 'FP' FILE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE 'INPUT' TEMPORARY FILE   --- */
 DEL                  ($XW..W$MODUL..INPUT) CL
 DEF CL         (NAME ($XW..W$MODUL..INPUT) -
                 MODEL($MODELF) -
                 REC  (1 200) -
      ) DATA    (NAME ($XW..W$MODUL..INPUT.D) )
    /* --- DELETE-DEFINE PERMANENT FILE    FP --- */
 DEL                  ($PXU..$PRO..FP) CL
 DEF CL         (NAME ($PXU..$PRO..FP) -
                 NAL RUS VOL($VOLVWK) -
                 KEYS (9 0) -
                 REC  (1000 1000) -
                 RECSZ(9 9) -
      ) DATA    (NAME ($PXU..$PRO..FP.D) -
      ) INDEX   (NAME ($PXU..$PRO..FP.I) )
/*
/. JCLST02
* --- STEP 02 --- PTU001
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM
// EXEC PTU001,SIZE=AUTO
/*
/. JCLST03
* --- STEP 03 --- PAN205
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7AE,'$PXY..$PRO.$PRO.AE',,VSAM
// DLBL PAC7FP,'$PXU..$PRO.$PFI.FP',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM
// EXEC PROC=$SORTWK CALL SORTWORK DEFINITION
// ON $RC GE 12 GOTO STEPEND
// EXEC PAN205,SIZE=(AUTO,50K)
/*
// ON $RC<16 CONTINUE
/. STEPEND
* --- STEP 04 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 DEL                  ($XW..W$MODUL..INPUT) CL
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

```
===SEQ FOR S2K
===MOD INFQ
* $X JOB JNM=$PRFJ.INFQ,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.INFQ        ****** $PROD $PREL ******
*                REINITIALIZATION OF 'FQ' FILE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE BACKUP FILE        FQ --- */
 DEL                  ($PXU..SEQ.$PRO$PFI..NEWFQ) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..NEWFQ) -
                 MODEL($MODELF) -
                 REC  (1000 1000) -
                 RECSZ(100 100) -
                 RECFM(FB(100)) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..NEWFQ.D) )
/*
/. JCLST02
* --- STEP 02 --- PAN200
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7FQ,'$PXU..SEQ.$PRO$PFI..NEWFQ',,VSAM
// EXEC PAN200,SIZE=AUTO
/. STEPEND
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.FQ.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

4.39. 8.2 COMPLEMENT: PAC/TRANSFER MODULE

```
8.2. COMPLEMENT: PAC/TRANSFER FACILITY
     --------------------------------
```

The PAC/TRANSFER facility enables to manage data in batch mode using two types of resources:

. LIBRARIES in which the programs that make up the system are stored, as well as the parameters required for its operation:

A batch program library,
A SLI member library (POWER),
A VSE procedure library.

. A PERMANENT FILE containing the parameters used to carry out the transfers.

<u>LOADING OF USER FILES</u>

The '3IUV' JCL module contains the initialization JCL of the transfer parameter file UV.

```
===SEQ FOR PACTR
===MOD 3IUV
* $X JOB JNM=$PRFJ.3IUV,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.3IUV        ****** $PROD $PREL ******
*   CREATION AND INITIALIZATION OF PARAMETER FILE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE PERMANENT FILE    UV --- */
 DEL                  ($PXU..$PRO..UV) CL
 DEF CL         (NAME ($PXU..$PRO..UV) -
                 NAL RUS VOL($VOLVWK) -
                 KEYS (19 1) -
                 REC  (1000 1000) -
                 RECSZ(80 80) -
      ) DATA    (NAME ($PXU..$PRO..UV.D) -
      ) INDEX   (NAME ($PXU..$PRO..UV.I) )
/*
/. STEPEND
* --- STEP 02 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL OUT,'$PXU..$PRO.$PFI.UV',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE(SYSIPT)
 9999999999999999999
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

4.40. 8.3 COMPLEMENT: BRIDGE VA PAC - TEAMCONNECTION

```
8.3. COMPLEMENT : VA PAC / TeamConnection BRIDGE
     -------------------------------------------
```

8.3.1. ALLOCATION OF SESSION FILE

This loading is made of with the procedure 'INTS' which includes the following steps:

STEP1: IDCAMS: This step executes the DELETE/DEFINE of the session file (TS).

STEP2: IDCAMS: Initialization of the session file (TS).

This process allows the creation of a certain record on the session file, without which the file is ineffective.

8.3.2. INSTALLATION OF THE TEAMCONNECTION DICTIONARY

This loading is made of with the procedure 'VINS'.

In the string of the DLBL 'PAC7MV', you have to change VGE into TCO.

```
===SEQ FOR TEAM
===MOD INTS
* $X JOB JNM=$PRFJ.INTS,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.INTS        ****** $PROD $PREL ******
*   INITIALIZATION OF TS , TEAMCONNECTION  FILE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE PERMANENT FILE    TS --- */
 DEL                  ($PXU..$PRO..TS) CL
 DEF CL         (NAME ($PXU..$PRO..TS) -
                 NAL RUS VOL($VOLVWK) -
                 KEYS (19 2) -
                 REC  (1000 1000) -
                 RECSZ(80 80) -
      ) DATA    (NAME ($PXU..$PRO..TS.D) -
      ) INDEX   (NAME ($PXU..$PRO..TS.I) )
/*
/. STEPEND
* --- STEP 02 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL OUT,'$PXU..$PRO.$PRO.TS',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 REPRO OFILE (OUT) IFILE(SYSIPT)
 9999999999999999999
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

4.41. 9. LIST OF INSTALLED PROGRAMS AND FILES

9. LIST OF INSTALLED PROGRAMS AND FILES

Execute JCL module '9INS'.

This job prints:

- A list of batch and on-line programs, including their compilation dates.
- A list of permanent system files: AE, SC, SF, SG, SP, SR, and SS, with their creation dates.

It is recommended that these listings be kept so that, in case of VA PAC System operating difficulties, the installation references can be sent to VisualAge Pacbase Support .

```
===MOD 9INS
* $X JOB JNM=$PRFJ.9INS,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.9INS        ****** $PROD $PREL ******
*     LIST OF INSTALLED PROGRAMS AND FILES
/* ************************************************************
/* INPUT FOR PACDTP :
/*
/* .IF A SELECTION OF PROGRAMS IS NEEDED, ONE LINE
/*  PER SELECTED PROGRAM WITH :
/* - CHARACTERS 3-8 : SELECTED PROGRAM CODE
/*
/* .TO OBTAIN THE WHOLE LIST OF THE PROGRAMS, ONE LINE IN
/*  INPUT, WHOSE FORMAT IS :
/* - CHARACTERS 3-4 : ROOT OF THE SYSTEM ($PRO)
/*
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  /*  --- DEFINE "INPUT" TEMPORARY FILE --- */
  DEL          ($XW..W$MODUL..INPUT) CL PURGE
  DEF CL (NAME ($XW..W$MODUL..INPUT) -
          MODEL($MODELF) -
          REC  (200 200) ) -
        DATA (NAME ($XW..W$MODUL..INPUT.D) )
/*
/. JCLST02
* --- STEP 02 --- PTU001
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM
// EXEC PTU001,SIZE=AUTO
  $PRO
/*
/. JCLST03
* --- STEP 03 --- PACDTP
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM
// DLBL PAC7AE,'$PXY..$PRO.$PRO.AE',,VSAM
// DLBL PAC7SC,'$PXY..$PRO.$PRO.SC',,VSAM
// DLBL PAC7SF,'$PXY..$PRO.$PRO.SF',,VSAM
// DLBL PAC7SG,'$PXY..$PRO.$PRO.SG',,VSAM
// DLBL PAC7SP,'$PXY..$PRO.$PRO.SP',,VSAM
// DLBL PAC7SR,'$PXY..$PRO.$PRO.SR',,VSAM
// DLBL PAC7SS,'$PXY..$PRO.$PRO.SS',,VSAM
// EXEC PACDTP,SIZE=AUTO
/*
/. STEPEND
* --- STEP 04 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  DEL          ($XW..W$MODUL..INPUT) CL
/*
// EXEC LISTLOG
/&
* $X EOJ
```

4.42. INSTALLATION TESTS

INSTALLATION TESTS

Tests on the operations are divided into three groups:

. Database Utilization Tests,

. Database Management Tests,

. Extraction Utility Tests.

4.43. UTILIZATION TESTS

1. UTILISATION TESTS

These tests include the following steps:

. On-line utilization tests under CICS,

. batch update test,

. Program generation-printing test,

On-line System test:

Open the test database files under CICS.
Test the screen branching.
Execute some updates.

Batch update test:

Execute the '$prfjUPDT' job (UPDT procedure).
The Database files must be closed under CICS.

Generation-printing test:

Execute the '$prfjGPRT'job (GPRT procedure).

The Database files must be closed under CICS if the generation-printing request file (AG)is to be read (+AG transaction as input to GPRT).

```
*
*      INPUT FOR UPDT PROCEDURE :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 *TEST            XCN
 0    TEST1       TEST
 P5010   N  TEST                      10BL
 P5010 1 M  AA00 BB00
 P5010 2 A  1 IWA10R
*
*      INPUT FOR GPRT PROCEDURE :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 *TEST            XCN
 Z  LCP
 Z  LCE
 Z  DCP TEST1
```

4.44. DATABASE MANAGEMENT TESTS

2. VA PACBASE DATABASE MANAGEMENT TESTS

The purpose of these tests is to execute the VA Pacbase database management procedures.

They include the following steps, to be executed in this order:

. Archiving of the journal created during the utilization test phase: execute the '$prfj.ARCH' job yielding the PJ(1) file.

. Direct backup of the Database: execute the '$prfj.SAVE' job yielding the PC(1) file.

. Backup of generation-printing requests: execute the '$prfj.SVAG'job yielding the PG file.

. Library management: addition/deletion of a Library: execute the '$prfj.MLIB' job yielding the PC(2) file.

. Reorganization of sequential backup PC(2) of the Database: execute the '$prfj.REOR' job yielding the PC(3)file.

. Reorganization of sequential backup (PG) of the generation-printing request file and restoration of the AG file: execute the '$prfj.REAG' job.

. Database restoration using the PJ(1) archived transaction file and the PC(3) Database backup file: execute the '$prfj.REST' job.

The VA Pac Database files must be closed under CICS when all these tests are run.

Once the VA Pac Database is restored, it is advisable to briefly test on-line operations again, after having re- opened the Database files.

```
*
*      INPUT FOR ARCH PROCEDURE :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 S
*
*      INPUT FOR MLIB PROCEDURE :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 G                          EXISTING DATABASE - NO SIMULATION
C*AP5XCN                    CREATE AP5 LIBRARY UNDER XCN
*
*      INPUT FOR REOR PROCEDURE :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 V0010
 BAP5
*
*      INPUT FOR REAG PROCEDURE :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 AG
*
*      INPUT FOR REST PROCEDURE :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 Y01000     REC                   123456789ABC
```

4.45. EXTRACTION UTILITIES TESTS

3. EXTRACTION UTILITY TESTS

The purpose of these tests is to execute the VA Pac Database extraction procedures.

They are made up of the following steps, to be executed in the following order:

1. Extraction of a library as transactions: execute job '$prfj.PACX' with EXLI type input,

2. Extraction of entities from a library: execute job '$prfj.PACX' with EXTR type input,

3. Extraction of selected transactions and/or lists of transactions from the archived journal (PJ): execute job '$prfj.PACX' with EXPJ type input.

The VA Pac Database files can be opened under CICS for all of these tests.

Each of these jobs can be followed by a UPDT to verify the validity of these extracted transactions.

```
*
*      INPUT FOR PACX PROCEDURE (EXLI TYPE) :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 *TEST            XCN
*
*      INPUT FOR PACX PROCEDURE (EXTR TYPE) :
*...+....1....+....2....+....3....+....4....+....5....+....6....

 *TEST            XCN
 W1EXCEXCHOIX
 W1EXCPBBMONI
 W1EXUDXS
 W1EXCSXS00
*
*      INPUT FOR PACX PROCEDURE (EXPJ TYPE) :
*...+....1....+....2....+....3....+....4....+....5....+....6....
 *TEST            CEN       EXPJ1
 JS
```

4.46. UTILIZATION TEST: Pac/Impact

2. TESTING Pac/Impact

The purpose of these tests is to execute the Pac/Impact procedures for VisualAge Pacbase.

To execute these tests, it is recommended to create a separate database for the UXSR procedure.
The initialization of the FP and FQ files must be performed.

They include the following steps, to be executed in this order:

. definition of entry points, execute the '$prfj.ISEP' job indicating the selection lines and the context,

. definition of complementary entry points, execute the '$prfj.ISOS' job indicating the selection lines and the context. This step is optional.

. Printing of the entry points, execute the '$prfj.IPEP' job.

. impact analysis, execute the '$prfj.IANA' job. For the tests, one execution of this procedure may be sufficient.

. printing of results, execute the '$prfj.IPIA' job, specifying the context and desired printouts.

. printing of impacted criteria, execute the '$prfj.IPFQ' job.

4.47. UTILIZATION TESTS: PAC/TRANSFER

2. TESTING THE PAC/TRANSFER FUNCTION

The purpose of these tests is to execute the procedures of the PAC/TRANSFER versioning management (for more details, refer to the 'ADMINISTRATOR'S GUIDE', chapter 'VERSIONING UTILITIES').

To execute these tests, the VA Pac database journal must have been archived.

The tests include the following steps, to be executed in this order:

. Update of the transfer parameters: execute the '$prfj.TRUP' job, specifying the required transfer parameters in the input transaction stream.

. Compression of the archived journal (optional test): execute the '$prfj.TRJC' job, specifying the name of the archived Journal file to be compressed.

. Creation of the transaction file: execute the '$prfj.TRPF' job, specifying the user code and password, as well as the transaction SET to be built.

. Preparation of the DSMS environment (this test is optional, and necessary only when the VA Pac database is under DSMS control): execute the '$prfj.TRDU' job, specifying the SET to be processed, then execute a batch update of the DSMS database with the first stream of transactions generated by the DUPD procedure.

. Generation of transfer transactions: execute the '$prfj.TRRP' job, specifying the user code and password, selecting all the SETs and all the sessions to be processed.

. Update of the VA Pac database using the UPDT procedure with the generated transfer transactions.

. Reinitialization of the DSMS environment (this test is optional, and necessary only if the VA Pacbase database is under DSMS control): batch update of the DSMS database with the second stream of transactions generated by the DUPD procedure.

4.48. UTILIZATION TESTS: VA PAC - VA SMALLTALK

VISUALAGE PACBASE-VISUALAGE SMALLTALK BRIDGE

To be able to test the Bridge procedures, you need a VA Pacbase test database and a file provided by VisualAge (for the VUP1 procedure), compatible with the VA Pacbase test database.

The procedures must be executed in the following sequence:

. VUP1 (JOB $PRFJ.VUP1)

. VUP2 (JOB $PRFJ.VUP2)

. VDWN (JOB $PRFJ.VDWN)

. VPUR (JOB $PRFJ.VPUR)

5. REINSTALLATION OF SUB-RELEASE

5.1. STANDARD REINSTALLATION

STANDARD REINSTALLATION OF THE SYSTEM

The System must be reinstalled when a new VA PAC release is shipped containing corrections to the preceding release or specific developments for this release.

The new sub-release is identified by a number, and usually includes:

. A complete installation cartridge or tape,

. A list of corrections made since the latest sub-release.

. If applicable, a set of instructions describing the reinstallation procedures to be followed,

Typically, only the program libraries and the system files are affected by a new version.

Therefore, reinstallation generally involves the execution of the following jobs:

1. JOB0, which copies the installation tape (job contained in the initial JCL file).

2. JOB2, which creates the reinstallation JCL.

   It executes the MM1JCL utility with the parameters provided at installation, but to which the following JCL module selection lines will be added:

   ===SELM 2PHA ===SELM 3SYS

   Additional selections may also be included in the set of instructions provided with the tape.

   Once the JCL modules are obtained via this job, or retrieved from a previous (re-)installation, the reinstallation continues with the execution of the jobs described below.

3. LOADING OF BATCH AND ON-LINE PROGRAM PHASES

Module 2PHA

IMPORTANT: According to the instructions provided with the tape, the only JCL lines to be kept are those which correspond to reinstalled phases.

4. RELOADING OF GENERATION 'SKELETON' FILES

Module 3SYS

5. RELOADING OF ERROR MESSAGE AND DOCUMENTATION FILE, RETRIEVING USER PARAMETERS

Two cases are possible:

1. There are no additional user parameters to be added for the reinstallation (no new users, no new VA PAC access key, etc.), and the parameter backup (PAC7CE file) is valid: run the LOAE procedure.

2. There are additional user parameters to be added for the reinstallation: run the PARM procedure with the NRCHAR command, followed by the update transactions.

6. RETRIEVAL PACBASE 802.02, ..., 1.6

6.1. WARNING

FOREWORD

Users of the WorkStation, DSMS and Pactables functions should ensure the operating compatibility of these functions with VisualAge Pacbase, according to the versions/releases in use.

The current VisualAge Pacbase release is compatible with:

- WorkStation new release

- DSMS from release 8.0.2 compatible with VA Pac 8.0.2

- Pactables all releases

NOTE:

Users of Pactables 7.3 or 8.0, need a special program, PTA250, for the GETT Pactables procedure. This program is available on request with IBM.

6.2. RETRIEVAL PACBASE 8.0, 8.0.1

6.2.1. RETRIEVAL OPERATIONS

RETRIEVAL OF PACBASE 8.0, 8.0.1: OPERATIONS

OPERATIONS TO BE PERFORMED

The retrieval of existing applications allowing for the transformation of a PACBASE 8.0 or 8.0.1 database into a database compatible with the new Visualage Pacbase release is broken down into the following phases:

```
     . Retrieval of 8.0 user parameters

     . Retrieval of 8.0 database(s)

For users of the PEI function:

     . Retrieval of Production Environment Interface (PEI)

For users of the WorkStation:
```

. The WorkStation's dedicated User Entities must be uploaded into the Database via the UPDT procedure, after the Database has been restored in the new release.

1. Retrieval of user parameters

. Backup of the user parameters, producing a PE file, old release.

. Retrieval of the 8.0 or 8.0.1 user parameters backup (PE80 procedure).

CAUTION: The SYSTEM ACCESS KEY must be entered on NK input line of the retrieval procedure.

. Execution of the new LOAE procedure using as input backup file the PE file produced by the parameter retrieval, and the '********' user code and NRREST as input parameters.

. For users of the VA Pac WorkStation, execution of the new PARM procedure, including in the input the transactions associated to the methodolgy used on the site. (See Chapter INSTALLATION, Subchapter 'INSTALLATION PROCESS', Section 'Loading of Error Messages, Documentation, and User Parameters'.) Use the NRCHAR command.

RESULT: AE and AP files, containing the old release user parameters as operational under the new VA Pac release, and the methodology parameters.

2. Retrieval of a VA Pac Database

. Backup of the database, producing a PC file (old release).

. Retrieval of the resulting PC 8.0 file in the new format (PC80 procedure), producing a new-format PC backup.

. Reorganization (new REOR) of the new-format backup obtained from the previous step.

. Journal file initialization (new ARCH procedure).

. Database restoration from the backup resulting from the REOR procedure (new REST procedure).

. Backup of generation-print commands, producing a PG file (old release).

. Retrieval of the generation-print commands (RPPG),

. Restoration of generation-print commands resulting from the backup of the previous step (new REAG procedure).

. Retrieval of sequential archive file (PJ80 procedure). This procedure is optional. It extracts Journal transactions from older archives, using new programs that handle the century in dates.

RESULT: AJ, AN, AR, and AG files operational under the new VisualAge Pacbase Release.

3. Retrieval of the Production Environment Interface

. PEI backup, producing a PP file (old release).

. Retrieval of sequential backup (PP16 procedure). The retrieval adds the century digits to all dates handled by the PEI Function.

. PEI restoration (new RSPE procedure) using in input the backup produced by the previous step.

RESULT: AB and AC files, operational under the new VA Pac Release.

6.2.2. RPPG: GENERATION-PRINT FILE (AG) RETRIEVAL

```
===SEQ FOR ROLD
===MOD PE80
* $X JOB JNM=$PRFJ.PE80,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.PE80        ****** $PROD $PREL ******
*               REPRISE DES PARAMETRES UTILISATEUR
/* ************************************************************
/* SUBSTITUTION: £OLDPE BY PHYSICAL NAME 'PE' OLD RELEASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE 'INPUT' TEMPORARY FILE   --- */
 DEL                  ($XW..W$MODUL..INPUT) CL
 DEF CL         (NAME ($XW..W$MODUL..INPUT) -
                 MODEL($MODELF) -
                 REC  (1 200) -
      ) DATA    (NAME ($XW..W$MODUL..INPUT.D) )
    /* --- DELETE-DEFINE BACKUP FILE        PE --- */
 DEL                  ($PXY..SEQ.$PRO$PRO..NEWPE) CL
 DEF CL         (NAME ($PXY..SEQ.$PRO$PRO..NEWPE) -
                 MODEL($MODELF) -
                 REC  (1000 1000) -
                 RECSZ(80 80) -
                 RECFM(FB(80)) -
      ) DATA    (NAME ($PXY..SEQ.$PRO$PRO..NEWPE.D) )
/*
/. JCLST02
* --- STEP 02 --- PTU001
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM
// EXEC PTU001,SIZE=AUTO
XNK000<----------------  SYSTEM ACCESS KEYS      -----------$K
/*
/. JCLST03
* --- STEP 03 --- PTU902
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7IN,'£OLDPE',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM,DISP=(,DELETE,KEEP)
// DLBL PAC7OU,'$PXY..SEQ.$PRO$PRO..NEWPE',,VSAM
// ON $RC GT 0 GOTO STEPEND
// EXEC PTU902,SIZE=AUTO
/*
// ON $RC<16 CONTINUE
/. JCLST04
* --- STEP 04 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PRO.PE.$SLIT
/*
/. STEPEND
* --- STEP 05 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 DEL                  ($XW..W$MODUL..INPUT) CL
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

6.2.3. PC80 : DATABASE RETRIEVAL

```
===SEQ FOR ROLD
===MOD PC80
* $X JOB JNM=$PRFJ.PC80,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.PC80        ****** $PROD $PREL ******
*               REPRISE DE LA SAUVEGARDE DE LA BASE
/* ************************************************************
/* SUBSTITUTION: £OLDPC BY PHYSICAL NAME 'PC' OLD RELEASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=DF$PRO.$PFI.PC.$SLIT
/*
/. JCLST02
* --- STEP 02 --- REPGDP
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MC,'£OLDPC',,VSAM
/* PAC7MC ON TAPE  : SYS011 ASSIGNED
// DLBL PAC7PC,'$PXU..SEQ.$PRO$PFI..NEWPC',,VSAM
/* PAC7PC ON TAPE  : SYS012 ASSIGNED
// EXEC REPGDP,SIZE=AUTO
/. STEPEND
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.PC.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

6.2.4. RPPG : GENERATION-PRINT FILE (AG) RETRIEVAL

```
===SEQ FOR ROLD
===MOD RPPG
* $X JOB JNM=$PRFJ.RPPG,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.RPPG        ****** $PROD $PREL ******
*               RETRIEVAL OF AG FILE (PG)
/* ************************************************************
/* SUBSTITUTION: £OLDPG BY PHYSICAL NAME 'PG' OLD RELEASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE BACKUP FILE        PG --- */
 DEL                  ($PXU..SEQ.$PRO$PFI..NEWPG) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..NEWPG) -
                 MODEL($MODELF) -
                 REC  (100 100) -
                 RECSZ(150 150) -
                 RECFM(FB(150)) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..NEWPG.D) )
/*
/. JCLST02
* --- STEP 02 --- PTU908
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7IN,'£OLDPG',,VSAM
/* PAC7IN ON TAPE  : SYS011 ASSIGNED
// DLBL PAC7OU,'$PXU..SEQ.$PRO$PFI..NEWPG',,VSAM
/* PAC7OU ON TAPE  : SYS012 ASSIGNED
// ON $RC GT 0 GOTO $EOJ
// EXEC PTU908,SIZE=AUTO
/*
// ON $RC<16 CONTINUE
/. JCLST03
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.PG.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

6.2.5. PJ80 : ARCHIVED JOURNAL RETRIEVAL

ARCHIVED JOURNAL RETRIEVAL

With release 8.0.2 or later of the DSMS Function, it is possible to modify the Product Code when retrieving the previous database. The Code format changes from 1 to 3 characters.

This possibility is also offered when retrieving the PACBASE archived journal from a previous release.

USER INPUT (optional)

Pos.	Len.	Value	Meaning

1	4	'onnn'	Modification of DSMS Product Code
			o = former Product Code
			nnn = new Product Code
			(1, 2 or 3 characters)

On a line, the 'onnn' character string may be repeated as many times as there are Product Codes to be modified, with a limit of 20.

NOTE: The validity of this input is not checked for proper syntax. The procedure's syntax should be carefully checked before execution, since no output report is produced.

```
===SEQ FOR ROLD
===MOD PJ80
* $X JOB JNM=$PRFJ.PJ80,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.PJ80        ****** $PROD $PREL ******
*               REPRISE DU JOURNAL ARCHIVE
/* ************************************************************
/* SUBSTITUTION: £OLDPJ BY PHYSICAL NAME 'PJ' OLD RELEASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE 'INPUT' TEMPORARY FILE   --- */
 DEL                  ($XW..W$MODUL..INPUT) CL
 DEF CL         (NAME ($XW..W$MODUL..INPUT) -
                 MODEL($MODELF) -
                 REC  (1 200) -
      ) DATA    (NAME ($XW..W$MODUL..INPUT.D) )
    /* --- DELETE-DEFINE TEMPORARY FILE     PJ --- */
 DEL                  ($XW..W$MODUL..PJ) CL
 DEF CL         (NAME ($XW..W$MODUL..PJ) -
                 MODEL($MODELF) -
                 REC  ($RAJP $RAJS) -
                 RECSZ(165 165) -
                 RECFM(FB(165)) -
      ) DATA    (NAME ($XW..W$MODUL..PJ.D) )
    /* --- DELETE-DEFINE BACKUP FILE        PJ --- */
 DEL                  ($PXU..SEQ.$PRO$PFI..NEWPJ) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..NEWPJ) -
                 MODEL($MODELF) -
                 REC  ($RAJP $RAJS) -
                 RECSZ(167 167) -
                 RECFM(FB(167)) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..NEWPJ.D) )
/*
/. JCLST02
* --- STEP 02 --- PTU001
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM
// EXEC PTU001,SIZE=AUTO
ONNN
/*
/. JCLST03
* --- STEP 03 --- PTU917
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7IN,'£OLDPJ',,VSAM
// DLBL PAC7MB,'$XW..W$MODUL..INPUT',,VSAM,DISP=(,DELETE,KEEP)
// DLBL PAC7OU,'$XW..W$MODUL..PJ',,VSAM
// ON $RC GT 0 GOTO STEPEND
// EXEC PTU917,SIZE=AUTO
/*
// ON $RC<16 CONTINUE
/. JCLST04
* --- STEP 04 --- REP2PJ
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7JP,'$PXU..SEQ.$PRO$PFI..NEWPJ',,VSAM
/* PAC7JP ON TAPE  : SYS011 ASSIGNED
// DLBL PAC7PJ,'$XW..W$MODUL..PJ',,VSAM
/* PAC7PJ ON TAPE  : SYS012 ASSIGNED
// ON $RC GT 0 GOTO STEPEND
// EXEC REP2PJ,SIZE=AUTO
/*
// ON $RC<16 CONTINUE
/. JCLST05
* --- STEP 05 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.PJ.$SLIT
```

```
/*
/. STEPEND
* --- STEP 06 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 DEL                  ($XW..W$MODUL..INPUT) CL
 DEL                  ($XW..W$MODUL..PJ)    CL
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

6.2.6. PP16 : PRODUCTION ENVIRONMENT RETRIEVAL

```
===SEQ FOR ROLD
===MOD PP16
* $X JOB JNM=$PRFJ.PP16,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.PP16        ****** $PROD $PREL ******
*               RETRIEVAL OF PEI FILE (PP)
/* ************************************************************
/* SUBSTITUTION: £OLDPP BY PHYSICAL NAME 'PP' OLD RELEASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE BACKUP FILE        PP --- */
 DEL                  ($PXU..SEQ.$PRO$PFI..NEWPP) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..NEWPP) -
                 MODEL($MODELF) -
                 REC  (200 200) -
                 RECSZ(110 110) -
                 RECFM(FB(110)) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..NEWPP.D) )
/*
/. JCLST02
* --- STEP 02 --- PACR90
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7PE,'£OLDPP',,VSAM
/* PAC7PE ON TAPE  : SYS011 ASSIGNED
// DLBL PAC7PS,'$PXU..SEQ.$PRO$PFI..NEWPP',,VSAM
/* PAC7PS ON TAPE  : SYS012 ASSIGNED
// ON $RC GT 0 GOTO $EOJ
// EXEC PACR90,SIZE=AUTO
/*
// ON $RC<16 CONTINUE
/. JCLST03
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.PP.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

6.3. RETRIEVAL VA PAC 802.02, ..., 1.6

6.3.1. OPERATIONS TO BE DONE

RETRIEVAL OF PACBASE 802v02, 1.2, 1.6: OPERATIONS

OPERATIONS TO BE PERFORMED

The installation of the new VA Pac release does not require any particular retrieval of the database files and associated user files, except for the generation-print command file, the PEI files (AB, AC) and the archived journal (PJ).

Once the new VA Pac release is installed, you must back up the databases and associated user files via the standard procedures of the release to be retrieved, and retrieve and restore these databases and files via the standard procedures of the new VA Pac release. To benefit from the new choices, you should include the reorganization procedure in the retrieval process.

For users of the WorkStation:

. The WorkStation's dedicated User Entities must be uploaded into the Database via the UPDT procedure, after the Database has been restored in the new release.

1. Reinstallation of the User parameters

. Backup of the user parameters, producing a PE file, old release (PARM procedure R 8.02 to 1.6)

. Execution of the new LOAE procedure using as input backup file the PE file produced by the preceding backup, and the '********' user code and NRREST command as input parameters.

. Execution of the PARM procedure with, as input, the MBPARM file containing the new key supplied with the product.

. For users of the VA Pac WorkStation, execution of the new PARM procedure, including in the input the transactions associated to the methodolgy used on the site. (See Chapter INSTALLATION, Subchapter 'INSTALLATION PROCESS', Section 'Loading of Error Messages, Documentation, and User Parameters'.) Use the NRCHAR command.

RESULT: AE and AP files, containing the old release user parameters as operational under the new VA Pac release, and the methodology parameters.

2. Reinstallation of a VA Pac Database

. Backup of the database, producing a PC file (old release);

. Journal file initialization (new ARCH procedure).

. Database restoration from the backup resulting from the backup procedure (new REST procedure).

. Backup of generation-print commands, producing a PG file (old release).

. Retrieval of the generation-print commands (RPPG), producing a PG file (new release).

. Restoration of generation-print commands resulting from the backup of the previous step (new REAG procedure).

. Retrieval of sequential archive file (PJ16 procedure). This procedure is optional. It extracts Journal transactions from older archives, using new programs that handle the century in dates.

RESULT: AJ, AN, AR, and AG files operational under the new VisualAge Pacbase Release.

3. Reinstallation of the Production Environment Interface

. PEI backup, producing a PP file (old release).

. Retrieval of sequential backup (PP16 procedure). The retrieval adds the century digits to all dates handled by the PEI Function.

. PEI restoration (new RSPE procedure) using in input the backup produced by the previous step.

RESULT: AB and AC files, operational under the new VA Pac Release.

6.3.2. RPPG : GENERATION-PRINT FILE (AG) RETRIEVAL

```
===SEQ FOR ROLD
===MOD RPPG
* $X JOB JNM=$PRFJ.RPPG,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.RPPG        ****** $PROD $PREL ******
*               RETRIEVAL OF AG FILE (PG)
/* ************************************************************
/* SUBSTITUTION: £OLDPG BY PHYSICAL NAME 'PG' OLD RELEASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE BACKUP FILE        PG --- */
 DEL                  ($PXU..SEQ.$PRO$PFI..NEWPG) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..NEWPG) -
                 MODEL($MODELF) -
                 REC  (100 100) -
                 RECSZ(150 150) -
                 RECFM(FB(150)) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..NEWPG.D) )
/*
/. JCLST02
* --- STEP 02 --- PTU908
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7IN,'£OLDPG',,VSAM
/* PAC7IN ON TAPE  : SYS011 ASSIGNED
// DLBL PAC7OU,'$PXU..SEQ.$PRO$PFI..NEWPG',,VSAM
/* PAC7OU ON TAPE  : SYS012 ASSIGNED
// ON $RC GT 0 GOTO $EOJ
// EXEC PTU908,SIZE=AUTO
/*
// ON $RC<16 CONTINUE
/. JCLST03
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.PG.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

6.3.3. PJ16: ARCHIVED JOURNAL RETRIEVAL

```
===SEQ FOR ROLD
===MOD PJ16
* $X JOB JNM=$PRFJ.PJ16,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.PJ16        ****** $PROD $PREL ******
*               RETRIEVAL OF AJ FILE (PJ)
/* ************************************************************
/* SUBSTITUTION: £OLDPJ BY PHYSICAL NAME 'PJ' OLD RELEASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE BACKUP FILE        PJ --- */
 DEL                  ($PXU..SEQ.$PRO$PFI..NEWPJ) CL
 DEF CL         (NAME ($PXU..SEQ.$PRO$PFI..NEWPJ) -
                 MODEL($MODELF) -
                 REC  ($RAJP $RAJS) -
                 RECSZ(167 167) -
                 RECFM(FB(167)) -
      ) DATA    (NAME ($PXU..SEQ.$PRO$PFI..NEWPJ.D) )
/*
/. JCLST02
* --- STEP 02 --- REP2PJ
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7JP,'$PXU..SEQ.$PRO$PFI..NEWPJ',,VSAM
/* PAC7JP ON TAPE  : SYS012 ASSIGNED
// DLBL PAC7PJ,'£OLDPJ',,VSAM
/* PAC7PJ ON TAPE  : SYS011 ASSIGNED
// ON $RC GT 0 GOTO $EOJ
// EXEC REP2PJ,SIZE=AUTO
/*
// ON $RC<16 CONTINUE
/. JCLST03
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.PJ.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```

6.3.4. PP16: PRODUCTION ENVIRONMENT RETRIEVAL

```
===SEQ FOR ROLD
===MOD PP16
* $X JOB JNM=$PRFJ.PP16,CLASS=$PCJ,DISP=$PDJ,USER='$PUJ'$PWJ
* $X LST CLASS=$PCL,DISP=$PDL$PWL
/* ************************************************************
// JOB $PRFJ.PP16        ****** $PROD $PREL ******
*                RETRIEVAL OF PEI FILE (PP)
/* ************************************************************
/* SUBSTITUTION: £OLDPP BY PHYSICAL NAME 'PP' OLD RELEASE
/* ************************************************************
// OPTION $OPT
* $X SLI MEM=DBJB$PRO.$PFI..$SLIT
/* GOTO JCLSTXX
/. JCLST01
* --- STEP 01 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
    /* --- DELETE-DEFINE BACKUP FILE        PP --- */
 DEL                  ($PXU..SEQ.$PRO$PFI..NEWPP) CL
 DEF CL          (NAME ($PXU..SEQ.$PRO$PFI..NEWPP) -
                  MODEL($MODELF) -
                  REC  (200 200) -
                  RECSZ(110 110) -
                  RECFM(FB(110)) -
      ) DATA     (NAME ($PXU..SEQ.$PRO$PFI..NEWPP.D) )
/*
/. JCLST02
* --- STEP 02 --- PACR90
// DLBL IJSYSUC,'$VCAP',,VSAM
// DLBL PAC7PE,'£OLDPP',,VSAM
/* PAC7PE ON TAPE  : SYS011 ASSIGNED
// DLBL PAC7PS,'$PXU..SEQ.$PRO$PFI..NEWPP',,VSAM
/* PAC7PS ON TAPE  : SYS012 ASSIGNED
// ON $RC GT 0 GOTO $EOJ
// EXEC PACR90,SIZE=AUTO
/*
// ON $RC<16 CONTINUE
/. JCLST03
* --- STEP 03 --- IDCAMS
// DLBL IJSYSUC,'$VCAP',,VSAM
// EXEC IDCAMS,SIZE=AUTO
* $X SLI MEM=RG$PRO.$PFI.PP.$SLIT
/*
// EXEC LISTLOG
/&
* $X EOJ
===SEQ
```