innovative. individual. independent.

improv IT

Software Innovations GmbH

eXtended Productivity Facility

Implementation Guide

Version 6.1.0

June 2016

This page intentionally left blank

I. Content

II. Listings

III.Screens

IV.   Graphics

V. Tables

1. XPF Installation

This chapter covers all steps required to successfully install the products XCS and XPF after the software upload.

1.1. XPF Architecture

XPF is made up of three components:

− eXtended Common Services (XCS)

− XPF Base

− XPF ISPF Dialogs
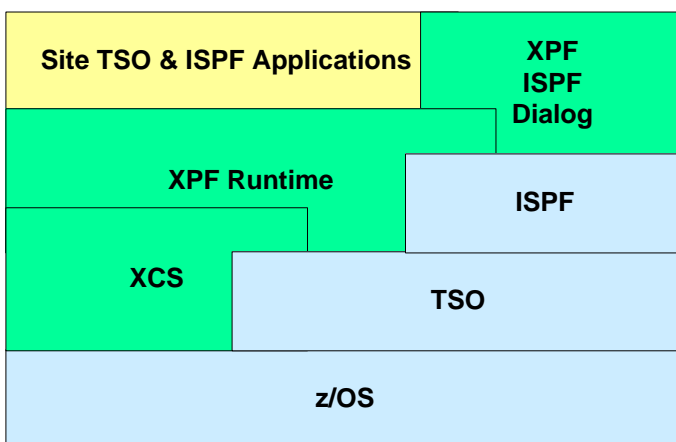
Graphic 1: XPF Architecture

1.2. Software Prerequisites

The following IBM software is required:

− z/OS V1.8 or later

− TSO/E V2.5 or later

− ISPF V4.1 or later

The following **improvIT Software Innovations** software is also required:

− eXtended Common Services (XCS) V2.8.0 or later

1.3. Runtime Datasets

It is recommended, that the XCS and XPF load libraries are defined in the Linklist concatenation.

If the runtime libraries are added to the Steplib concatenation, then all datasets must be APF authorised.

The use of a Steplib is not recommended, as it can be detrimental to performance.

1.4. Product Datasets - XCS

eXtended Common Services (XCS):

− Runtime datasets

- XCS280.SXCSAPF   (APF-Authorised modules)
- XCS280.SXCSLOAD   (normal modules)

1.5. Product Datasets - XPF

eXtended Productivity Facility (XPF):

− Runtime datasets

- XPF610.SXPFOLOD   (Runtime)

− XPF dialog datasets

- XPF610.SXPFMSG   (ISPF-Dialog Messages)
- XPF610.SXPFPANL   (ISPF-Dialog Panels)

− Others

- XPF610.SXPFSAMP   (Samples)

1.6. System Definitions - Parmlib

Add the following libraries to the Linklist (recommended) or LPA concatenation:

− XCS280.SXCSAPF

− XCS280.SXCSLOAD

− XPF610.SXPFOLOD

APF authorise the library:

− XCS280.SXCSAPF

Modify PARMLIB member IKJTSO00 :

− Modules "XCSTEDAT" and "XCSAWSMF" must be defined in the following sections: "AUTHCMD", "AUTHPGM" and "AUTHTSF"

− Module "XCSADSTP" must be defined in section "AUTHTSF" (for use of the new "TSO Dynamic STEPLIB function". See the "XPF Users Guide" for more details)

1.7. System Definitions - ISPF

Modify the ISPF control table ISPTCM:

− Always add the following module to the table with hex value '02'

- XPFEISST

− If the XPF load library is located in the LPA then also add the following modules with hex value '42' to the table

- XPFEISST
- XPFAG4IT
- XPFEGATE
- XPFEICTL
- XPFEIGTW
- XPFEILOG
- XPFEITRC
- XPFEMAIN
- XPFEXCTL
- XPFEXGTW

1.8. System Definitions - Logon

Create a new TSO logon procedure using the supplied sample "XPFLOGON". Modify the datasets to match your installation.

```
//**********************************************************************/
//*                                                                    */
//* PROC NAME    : LOGON USING XPF600                                  */
//*                                                                    */
//* DATE         : FEB 2012                                            */
//*                                                                    */
//* WRITTEN BY   : IMPROVIT SOFTWARE INNOVATIONS GMBH (C)OPYRIGHT      */
//*                                                                    */
//* NOTICE       : THIS PROGRAM MAY NOT BE COPIED, REPRODUCED OR       */
//*                DISTRIBUTED IN ANY FORM (ELECTRONIC OR OTHERWISE)    */
//*                WITHOUT WRITTEN PERMISSION FROM THE AUTHOR.          */
//*                                                                    */
//* RETURN CODES: GATEWAY TERMINATION VALUE                            */
//*                                                                    */
//* DESCRIPTION : TSO STARTUP PROCEDURE USING XPF GATEWAY              */
//*                                                                    */
//*                THE DDNAME XPFDTRCE MAY BE ALLOCATED. THIS IS ONLY   */
//*                REQUIRED BY IMPROVIT SUPPORT.                        */
//*                                                                    */
//**********************************************************************/
//XPFLOGON EXEC PGM=IKJEFT01,PARM='XPFESTRT',DYNAMNBR=300
//* STEPLIB (if XPF is not in LPA or LLA)
//*STEPLIB  DD    DISP=SHR,DSN=????.XPF600.SXPFOLOD
//* BOOT EXIT DATASET
//SYSEXEC  DD    DSN=SYS4.XPF.EXEC,DISP=SHR
//* BOOT PARMLIB DATASET
//XPFDBOOT DD    DSN=SYS4.XPF.PARMLIB(BOOTIT00),DISP=SHR
```

Screen 1: System Definitions - Logon

Note: When logged on, a log file is dynamically allocated to Sysout (DDName: XPFDMSGS)

1.9. System Definitions - Logon

The logon procedure contains the following libraries:

− DDName STEPLIB

- Is not normally required as all XCS & XPF modules should be in the Linklist or LPA concatenation

− DDName SYSEXEC

- This DDName should point to the REXX exec library containing the logon exits. This library is automatically freed when the exits have been processed

− DDName XPFDBOOT

- This DDName must point to the XPF Parmlib dataset/member. These definitions control the runtime behaviour of XPF and Gateways

− DDName XPFDTRCE

- This DDName causes the internal XPF trace to be activated. The trace information is only required by improvIT Software Innovations in case of an internal XPF error. Trace data is written to the Sysout log file "XPFDTRCM". It should not generally be activated as it is detrimental to performance. Tracing can also be dynamically

activated/deactivated by using the command XPFEITRC.

1.10. System Definitions - Logon Procedure

An optional Gateway name and start option may be specified. These values must be comma delimited. If the parameters are omitted, then the normal start Gateway will be used. These values are generally only used by batch applications

If Gateway name "**NONE**" is specified, then the XPF environment will be initialised, but a Gateway will not be directly started. This feature is generally only useful for online applications such as SAS/PC, which automatically connect to TSO using native Telnet and need to enter TSO commands directly prior to starting the application Gateway

Several other options are available for Gateway maintenance in batch. These are described in the XPF Users Guide

2. XPF Customisation

This chapter explains all XPF customisation steps. After all points have been completed, the first logon using XPF can be carried out.

2.1. Installing the Licence Key

To use XPF you will need a licence key. This key is supplied by improvIT Software Innovations.

The key needs to be installed using the samplib member "ACTXPFLI". The sample job updates the licence information in the XPF loadlib.

If an invalid XPF licence key is found, then XPF runs in "slow mode" and a message is displayed.

2.2. Parmlib

The XPF parmlib needs to be customised before the first logon with XPF can take place. The parmlib attributes are as follows:

− Can be a sequential or partitioned dataset

− The name of the parmlib dataset/member is defined in the TSO logon procedure

− It is only processed once during the TSO logon

− Can be used to control the runtime behaviour of XPF and Gateways

− System and site variables are defined in the parmlib

− Member "BOOTINI" in the samplib can be used as a template

The following syntax rules apply to the XPF parmlib:

− Literals

  • Can be defined with or without quotation marks or apostrophes

− Variables

  • Must begin with an ampersand "&"

  • Can be used on both sides of an assignment

  • Concatenated variable names must be terminated with a ".", space or another variable name

  • Can be deleted by assigning a null value (i.e. "")

  • The default XPF variables can not be deleted

  • &GATENAME is the only variable that is resolved at runtime

− Simple If/Then/Else logic can be used

  • The basic syntax is similar to REXX. However the following restrictions apply:

- One Statement may not span records
- Nested IFs are not possible
- IF statements must be terminated using ENDIF
- System symbols (defined in SYS1.PARMLIB)
  - Can be assigned to site variables by using the SYM() function
- Comments
  - Are started using "/*" and terminated with "*/"
  - Must start and complete within the same record
- Messages may be displayed using the XPF command Say
- The following pages describe the default parmlib entries

2.3. Variables

The following variables are always available during XPF parmlib and Gateway processing and cannot be deleted:

Variable Name	Description
&SYSNAME	System name
&SYSCLONE	System clone id
&SYSID	SMF system id
&USERID	Current Userid
&DRACFGRP	Default SAF group for current user
&XPFRELNO	Current XPF release
&GATENAME	Name of Gateway being currently processed
&JESTYPE	Installed JES (JES2 or JES3)
&TCPIPID	Name of TCPIP subsystem
Variable Name	Description
&JOBNAME	Current job name
&JOBID	Current job id
&JOBSTEP	Name of job step
&JOBPROC	Name of job procedure
&XCFGROUP	XCF group name
&LUNAME	Terminal LU id

Table 1: Variables

2.4. Parmlib - Default Gatelibs

The following extract shows how to define user variables. These are then used to dynamically build the site Gatelib dataset names. At least one Gatelib name must be defined. The last step demonstrates how to remove a user defined variable by assigning an empty string.

```
/*------------------------------------------------------------------*/
/* Some user defined variables                                      */
/*------------------------------------------------------------------*/
&I                 = &SYSNAME
&U                 = &USERID
&G                 = &GATENAME
&LLQ               = GATELIB


/*------------------------------------------------------------------*/
/* Gateway library template definitions (use 'none' to ignore)      */
/*------------------------------------------------------------------*/
&GLOBAL_GATE_MASK  = SYS4.XPF.&LLQ
&SYSTEM_GATE_MASK  = 'NONE'
&GROUP_GATE_MASK   = &G..XPF.&LLQ
&USER_GATE_MASK    = &U..XPF.&LLQ
```

Screen 2: Parmlib - Default Gatelibs

The number of available Gatelibs can vary from 1 to N. N depends on how variable the Gatelibs name have been defined and can theoretically be a very large value. The search sequence when loading Gateways from Gatelibs is User, Group, System, Global. The following suggestions may be useful when defining Gatelibs:

− Global: should be the same for all users on all systems

− System: could be LPAR dependant or act as a pre-production stage

− Group: can vary from user to user based on e.g. default security groups

− User: should always be a private user Gatelib

2.5. Parmlib - Logon Gateway

Use variable &START_GATE to specify which Gateway should be started after XPF initialisation. This Gateway should contain all base TSO & ISPF datasets. A Gateway start option value may also be specified. The value is passed to the application. A sample Gateway ISPFGATE is contained in the XPF samplib dataset.

```
/*------------------------------------------------------------------*/
/* Name of Gateway that contains all the base allocations and       */
/* possibly a start option to pass to the initial Gateway.          */
/* !!! Note - If &START_OPTS is assigned a null value, then it  !!! */
/* !!!        will NOT be deleted (unlike other variables)      !!! */
/*------------------------------------------------------------------*/
&START_GATE        = XPFISPF
&START_OPTS        = ''
```

Screen 3: Parmlib - Logon Gateway

2.6. Parmlib - Security

Use the following variables to define whether XPF is to carry out SAF checking. If activated, XPF checks whether the user is authorised to start the required Gateway. The corresponding profile must be defined and should always contain &GATENAME. Either class DATASET or FACILITY may be used (default is Dataset).

```
/*-------------------------------------------------------------------*/
/* Activate SAF check when starting a gate against the specified     */
/* profile. The profile type can either be "D" (Dataset) or         */
/* "F" (Facility). The default for TYPE is "D" and the default       */
/* profile name is "XPFGATE.&GATENAME".                              */
/*-------------------------------------------------------------------*/
&SAF_CHECK        = Y
&SAF_PROFILE      = XPFGATE.&GATENAME
&SAF_PROFILE_TYPE = F
```

Screen 4: Parmlib - Security

2.7. Parmlib - Application Logging

The variable &SMF_RECNUM controls whether SMF records are to be written by XPF. If the value is 0 (zero) then no records are created. Otherwise the entered numeric value will be used as the SMF record type. The default value is 0.

```
/*-------------------------------------------------------------------*/
/* Activate SMF logging for Gateways. Specify a SMF record type      */
/* number (normally between 128 & 255). If the value 0 (zero) is     */
/* specified, then no logging is performed                           */
/*-------------------------------------------------------------------*/
&SMF_RECNUM       = 0
/* &SMF_RECNUM       = 254 */
```

Screen 5: Parmlib - Application Logging

If application logging is activated, then four different data record types are created by XPF and written to SMF according the requested processing:

− When a Gateway is loaded from a Gatelib

- Format: "XPF-GW-LOAD   <gateway name> <gatelib name> <load rc>"

− When a Gateway is removed from the cache

- Format: "XPF-GW-REMOVE <gateway name> <remove rc>"

− When an application is activated by XPF

- Format: "XPF-GW-ACT   <gateway name>"

− When an application is terminated by the user

- Format: "XPF-GW-TERM   <gateway name> <application rc>"

The collected information can e.g. be used to establish which application Gateways are in use and how often.

2.8. Parmlib - Logon & Logoff

The following values can be used to prevent the user being prompted during the logon and from being returned to native TSO after exiting from the application. The default values are as shown.

```
/*------------------------------------------------------------------*/
/* Automatically clear screen after '***' is displayed during logon */
/*------------------------------------------------------------------*/
&TSO_CLEAR_SCR    = Y

/*------------------------------------------------------------------*/
/* TSO Logoff after leaving the last active gate                    */
/*------------------------------------------------------------------*/
&TSO_LOGOFF       = Y
```

Screen 6: Parmlib - Logon & Logoff

2.9. Parmlib - Initialisation Exits

Change the following variables to activate the initialisation exits. The exits must be coded in REXX or adhere to the REXX function linkage conventions when coded in another language. The exits are described in chapter "XPF Initialisation Exits".

```
/*------------------------------------------------------------------*/
/* Name of user initialisation exit command        (N = None)       */
/*------------------------------------------------------------------*/
&USER_BOOT_EXIT   = 'STRTIT00'

/*------------------------------------------------------------------*/
/* Name of user startup Gateway override exit       (N = None)       */
/*------------------------------------------------------------------*/
&USER_BOOT_GATE   = 'N'
```

Screen 7: Parmlib - Initialisation Exits

2.10.   Parmlib - Module Preloading

The following value specifies if modules are to be preloaded into the JPAQ. Changing this value can affect the general performance of the XPF environment, especially if a STEPLIB or ISPLLIB is active. Note: This is required as VLF is not always searched by TSO when functions are called. The default value is 2.

```
/*------------------------------------------------------------------*/
/* Select the modules to preload during logon which can improve the */
/* performance (0 = None, 1 = XCS & XPF, 2 = XCS, XPF, TSO & ISPF)   */
/*------------------------------------------------------------------*/
&PRELOAD_MODULES  = '2'
```
Screen 8: Parmlib - Module Preloading

2.11.　　Parmlib - Housekeeping

The following variable specifies how long unused Gateways are to remain in the XPF cache before automatically being removed. Each Gateway generally requires 4K of memory (above the 16mb line if available). This value should not normally be changed as it can be detrimental to performance. The default is 0.

```
/*----------------------------------------------------------------*/
/* Number of minutes that a Gateway will remain in the cache after  */
/* the last use before being automatically removed (0 = None - 720) */
/*----------------------------------------------------------------*/
&MAX_CACHE_TIME   = '0'
```

Screen 9: Parmlib - Housekeeping

2.12.　　Parmlib - Messages

XPF tries to be very informative and can display a lot of messages. This could be irritating. Changing the value below will cause XPF to display all information, warning and error messages. Otherwise only warning and error messages are displayed. All messages are however written to the Sysout log dataset (DDName XPFDMSGS).

```
/*----------------------------------------------------------------*/
/* Display informational messages                                  */
/*----------------------------------------------------------------*/
&DISP_INFOMSG     = 'N'
```

Screen 10: Parmlib - Messages

2.13.　　Parmlib - Extended Searches

This option activates an implicit fifth Gatelib level. This can be useful when Gateways are loaded explicitly from Gatelibs which are not in the default search sequence. If nested Gateways are then called, the explicit Gatelib is searched prior to the default base Gatelibs. This option has no effect on Gateways already in the cache.

```
/*----------------------------------------------------------------*/
/* Define whether or not the extended Gatelib search sequence is to */
/* activated. If "Y", then prior to searching the 4 defined Gateway */
/* libraries (see above), the Gatelib from which the current        */
/* Gateway was loaded from will also be searched. This will only    */
/* apply when a Gateway is being started                            */
/*----------------------------------------------------------------*/
&EXT_GL_SEARCH    = 'Y'
```

Screen 11: Parmlib - Extended Searches

2.14. Parmlib – Search for updated Gateways

This option controls whether XPF searches all Gatelibs when loading a Gateway for the first time during a session. If active, XPF compares the timestamps of the requested Gateways (if found in more than one Gatelib). A message is displayed if an updated Gateway was found in a higher Gatelib

```
/*------------------------------------------------------------------*/
/* This option specifies that XPF is to search all active Gatelibs  */
/* for the entered Gateway. If the Gateway is found in more than 1  */
/* Gatelib, then the timestamps of the Gateways in the higher level */
/* Gatelibs are compared with the lowest level. If the lowest level */
/* Gateway is older than 1 or more of the higher levels, then a     */
/* message is displayed. Valid options are "Y" or "N".              */
/*------------------------------------------------------------------*/
&CHK_4_NEWER_GW   = 'Y'
```

Screen 12: Parmlib - Search for updated Gateways

2.15. Parmlib - ISPF Command Tables

This option controls if the ISPF application start commands (as defined in the Gateways) are available. If the value is NONE then the defined command name will have no effect. Otherwise enter the prefix of your site ISPF command table. This table will not be changed on DASD.

```
/*------------------------------------------------------------------*/
/* Specify the prefix of the ISPF command table name (max 4 bytes). */
/* This table contains the ISPF command name values as specified in */
/* the Gateways definitions. If the value is "NONE" then no commands*/
/* are loaded. The ISPF command table must exist. All changes are   */
/* only made in memory. Recommended is the use of the SITE level    */
/* command table.                                                   */
/*------------------------------------------------------------------*/
&ISPF_CMDTAB = 'SITE'
```

Screen 13: Parmlib - ISPF Command Tables

2.16. Parmlib - Licence

This option specifies the users to be informed when the XPF licence is about to expire. If "*" is entered, then all users will be informed during TSO logon. Otherwise a list of TSO userids must be specified (separated by commas. This option only applies between 10 and 30 days prior to expiry. When there are less than 10 days left, all users are always informed.

```
/*------------------------------------------------------------------*/
/* Define users to be notified when XPF licence is about to expire. */
/* The default value is "*" (notify all). Otherwise a list of comma */
/* delimited users must be specified.                               */
/*------------------------------------------------------------------*/
&LIC_EXPIRY_USERS = '*'
```

Screen 14: Parmlib - Licence Expiry

Note: A TSO logon is still possible when the XPF licence has expired. Instead "nag mode" is entered.

2.17. Parmlib – Edit Macro

The following extract shows how to define the default JCL job class and message class values. These are only used by the XPF ISPF edit macro #JC (create a job card).

```
/*-------------------------------------------------------------*/
/* The following variables contain information for the XPF ISPF  */
/* edit jobcard macro #JC (XPFMJCRD).                            */
/*-------------------------------------------------------------*/
&EDMAC_JCLASS = 'N'
&EDMAC_MCLASS = 'T'
```

Screen 15: Parmlib – Edit Macro Defaults

2.18. Parmlib - System Symbols

The following example demonstrates how to access a system symbol value and make this available for later usage. Only explicitly defined system symbols can be used within the XPF environment.

```
/*-------------------------------------------------------------*/
/* Sample user defined System Symbol (IEASYM##)                 */
/*-------------------------------------------------------------*/
&SYSPLEX      = SYM(SYSPLEX)        /* SYSPLEX Name */
```

Screen 16: Parmlib - System Symbols

2.19. Parmlib - Logic

The following code demonstrates the use of the IF/THEN/ELSE and Say commands. "SAY" can be used to display text and variables during initialisation file processing.

```
/*-------------------------------------------------------------*/
/* Sample logic                                                 */
/*-------------------------------------------------------------*/
If ('&G' = 'IMPROVIT' & &SYSNAME = 'TEST') then
  Say 'Preload deactivated for group &G on system &SYSNAME'
  &PRELOAD_MODULES  = '0'
  &PRIND = 'N'
Else
  &PRIND = 'Y'
EndIF
```

Screen 17: Parmlib – Logic

2.20. Defining the first Gateway

A Gatelib must be created prior to the first XPF logon. Gateways may only be maintained using the supplied ISPF XPF dialog. A logon using XPF must have carried out in order to use the dialog. A Gateway must be available in order to perform a logon using XPF. However, Gateways may only be maintained using the supplied ISPF XPF dialog.

− Problem:

- How is the first Gateway defined?

− Solution:

- Using member DLGSTART in the samplib library!

Member DLGSTART must be changed to reflect the site installation datasets. Afterwards in can be executed and a Gateway defined. Copy the samplib member ISPFGATE using the dialog to your defined Gatelib and change it as required.

```
/* REXX ******************************************************************/
/*                                                                      */
/* Exec Name    : DLGSTART                                              */
/*                                                                      */
/* Date         : February 1999                                        */
/*                                                                      */
/* Written by   : Jon Renton, (c) improvIT Software Innovations GmbH */
/*                                                                      */
/* Parameters   : None                                                 */
/*                                                                      */
/* Return Codes: Dialog RC                                             */
/*                                                                      */
/* Description  : This exec can be used to start the XPF dialog        */
/*                after installation, in order to define the base      */
/*                Gateways needed to start XPF the first time.         */
/*                                                                      */
/*                Change installation datasets as required !!!!!!      */
/*                                                                      */
/*                NOTE: No variables may be used in the Gateway        */
/*                      defintions when using this routine !           */
/*                                                                      */
/************************************************************************/
```

```
 X = TRACE('OFF')

 /* Define the installation datasets */
 LOADLIB = '????.XPF???.SXPFOLOD'
 PANLIB  = '????.XPF???.SXPFPANL'
 MSGLIB  = '????.XPF???.SXPFMSG'

 /* Start base dialog */
 "CALL '"LOADLIB"(XPFEEMER)' '"LOADLIB PANLIB MSGLIB"'"
 Say '*** DLGSTART RC: 'RC' ***'

Return RC
```

Screen 18: Defining the first Gateway

Note: When using this environment, the Gatelibs must be explicitly specified and XPF variables can not be used.

3. XPF Initialisation Exits

This chapter will explain the function of the initialisation exits. The following rules apply to both exits:

− Both exits are optional

− The names are defined in the XPF parmlib

− They are only processed once during the TSO logon

− They must be coded in REXX or adhere to the REXX function linkage conventions when coded in another language

3.1. Exit 1 - Runtime Initialisation

This exit can be used to initialise and build your TSO runtime environment. A sample "BOOTEXIT" is available in the samplib dataset. The exit has the following attributes:

− Can be used to carry out the following actions:

  • Create the ISPF profile dataset

  • Allocate ISPF work datasets

  • Etc.

− The routine is passed the following parameters:

  • Userid

  • System Id

  • Name of start-up Gateway

  • Current terminal LU

  • Jobstep Name

  • Start-up Gateway option value

3.2. Exit 2 - Initial Gateway Override

This exit can be used to override the default start Gateway and options as defined in the XPF parmlib or as passed to XPFESTRT. A sample "BOOTGATE" is available in the samplib dataset. It has the following attributes:

− Can be used to override the start-up Gateway name and define start-up options for the overridden Gateway

− The routine is passed the following parameters:

  • Userid

  • System Id

  • Name of start-up Gateway

  • Current terminal LU

- Jobstep Name
- Start-up Gateway option value

4. Activating the XPF Edit Macros

Under normal circumstances, the IBM ISPF editor will only search for and execute uncompiled edit macros (i.e. those written in REXX or CLIST).

It is necessary to define Macros written in other languages (e.g. Assembler, C, Cobol) to the editor. Otherwise the load modules will not be found, as the allocated libraries are not searched. Use the IBM ISPF editor Define command to activate compiled macros.

An example can be found in the XPF samplib (member XPFEDMAC). Define this macro as the installation wide initial ISPF macro or copy the code to an existing routine.

The sample contains one definition for every XPF edit macro. After execution the ISPF editor can call the defined macros. An alias has also been defined. This step is optional and the alias names can be changed if required. The advantage of an alias is that the user does not need to remember the module name of the edit macro.

The following listing shows the sample XPFEDMAC.

```
"MACRO NOPROCESS"

/* XPF ISPF Edit Macro Definitions */
"Define XPFMBACK macro program"
"Define #BU      alias XPFMBACK"
"Define XPFMCNTR macro program"
"Define #CE      alias XPFMCNTR"
"Define XPFMCCOL macro program"
"Define #CC      alias XPFMCCOL"
"Define XPFMCSTR macro program"
"Define #CS      alias XPFMCSTR"
"Define XPFMCUTD macro program"
"Define #CU      alias XPFMCUTD"
"Define XPFMDCOL macro program"
"Define #DC      alias XPFMDCOL"
"Define XPFMDSTR macro program"
"Define #DS      alias XPFMDSTR"
"Define XPFMEDSN macro program"
"Define #ED      alias XPFMEDSN"
"Define XPFMEIDX macro program"
"Define #EI      alias XPFMEIDX"
"Define XPFMEXEC macro program"
"Define #EX      alias XPFMEXEC"
"Define XPFMFDUP macro program"
"Define #FD      alias XPFMFDUP"
"Define XPFMHELP macro program"
"Define #HE      alias XPFMHELP"
"Define XPFMIMED macro program"
"Define #IM      alias XPFMIMED"
"Define XPFMJCRD macro program"
"Define #JC      alias XPFMJCRD"
"Define XPFMMCPY macro program"
"Define #MC      alias XPFMMCPY"
"Define XPFMMLST macro program"
"Define #ML      alias XPFMMLST"
"Define XPFMNMBR macro program"
"Define #NM      alias XPFMNMBR"
"Define XPFMONLY macro program"
"Define #ON      alias XPFMONLY"
"Define XPFMPAST macro program"
"Define #PA      alias XPFMPAST"
"Define XPFMPREF macro program"
"Define #PF      alias XPFMPREF"
"Define XPFMSUFF macro program"
"Define #SF      alias XPFMSUFF"
"Define XPFMSUBM macro program"
"Define #SU      alias XPFMSUBM"
"Define XPFMXCOL macro program"
"Define #XC      alias XPFMXCOL"

Return 0
```

Listing 1: Samplib - Defining XPF ISPF Edit Macros

5. Migrating to XPF

This chapter will present two approaches which can be used to migrate an existing TSO/ISPF environment to XPF control. Generally the used method will vary depending on the installation and site requirements.
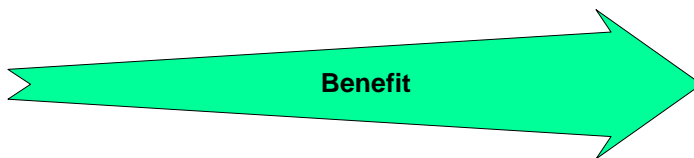
5.1. Migrating to XPF - Soft Integration

XPF is installed and activated for all logon procedures. Datasets are at first hardly controlled by XPF and applications are directly started using existing procedures. Applications are then migrated on demand to XPF control and logon procedures are only consolidated when possible.

This method is a slower approach and generally takes far longer for the integration is complete. A lot of XPF benefits are only available after the integration has been fully completed. The existing environment is also not cleaned up and legacies continue to exist.

This method should not be used if migrating from tools such as CA-ROSCOE.

Benefit

Graphic 2: Migrating to XPF - Soft Integration

5.2. Migrating to XPF - Fast Integration

XPF is installed and activated using a new logon procedure. A new parallel test environment is created and all applications are migrated as quickly as possible to XPF control. A switch to the new environment for all users can be automatically performed after all required tools and applications have been migrated.

This method is far more efficient than the soft integration. It can be completed relatively quickly as a project. The XPF benefits are available immediately after user migration and errors can easily be isolated. Legacies are automatically removed as only required tools and application are migrated into the new environment.

Benefit
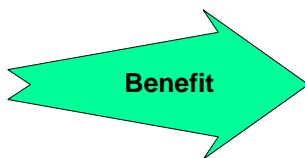
Graphic 3: Migrating to XPF - Fast Integration

6. Release Information

6.1. Compatibility

Gateways changed or created with previous XPF releases can be processed by XPF V610. It is however not possible to edit or process V610 Gateways using releases older than V600.

Modules XPFEG110 and XPFECNVT have been removed. Corresponding entries in parmlib member ISPTCM should be deleted.

6.2. New in XPF Version 6.1.0

− A non-existent INCLUDE gateway in the LOGON gateway will not result in the termination of the TSO session. Instead the INCLUDE gateway will be ignored and a warning message will be issued.

− The new REXX function "TSO Dynamic STEPLIB" is included. A sample gateway is provided in the samplib.

6.3. New in XPF Version 6.0.0.

− Errors in the XPF Logon Gateway will not automatically lead to the TSO session being terminated. Instead XPF will attempt to continue starting the application.

− A new XPF Parmlib option has been added, to check that updated Gateways do not exist in higher Gatelibs. If found, then the user is informed.

− XPF Licence Handling has been converted to ZAP format.

− XPF now verifies that the correct XCS version in available.

− The integrated ISPF Edit Macro handling has been improved.

− Sample DLGSTART compatibility has been corrected.

− Internal SMF handling has been changed.

− Modules XPFEG110 and XPFECNVT have been removed.

− Internal optimisation.

7. Contact

For further information regarding the eXtended Productivity Facility please contact:

improvIT Software Innovations GmbH

Große Elbstraße 141 a

D-22767 Hamburg

Germany

Telephone: +49 (0)40 540 90 29 - 7

Fax: +49 (0)40 540 90 29 - 9

Email: Contact@improvIT-Software-Innovations.de

Web: www.improvIT-Software-Innovations.de

8. Index