# ETP Installation

This chapter covers the following topics:

- Prerequisite Requirements and ETP Release Contents
- General Information about ETP Installation
- Additional Information about ETP Installation

---

# Prerequisite Requirements and ETP Release Contents

This section describes the operating system and companion software needed to run ETP, and describes the modules contained on the ETP release (distribution) tape.

The following topics are covered:

- Operating System and Software Requirements
- ETP Distribution Tape

## Operating System and Software Requirements

The prerequisite operating/teleprocessing systems and required versions of other Software AG products are listed in the current Natural Release Notes for Mainframes.

## ETP Distribution Tape

The ETP installation tape contains the following datasets:

| Dataset Name: | Description: |
|---|---|
| ETP*vrs*.INPL | The INPL dataset. |
| ETP*vrs*.ERRN | ETP error messages. |
| ETP*vrs*.SYSF | An empty system file in ADALOD format for use as a combined administration/confirmation/log file. |
| ETP*vrs*.SYS1 | An empty system file in ADALOD format for use as an administration file. |
| ETP*vrs*.SYS2 | An empty system file in ADALOD format for use as a log file. |
| ETP*vrs*.SYS3 | An empty system file in ADALOD format for use as a confirmation file. |
| ETP*vrs*.FDTA | System file containing FDT definitions for all ETP files, in ADACMP format. |
| ETP*vrs*.SRCE | (z/OS, z/VSE, VM/CMS system tapes) Sample program for using ETP services from 3GL programs. |
| ETP*vrs*.SRC | (BS2000 system tapes only) Sample program for using ETP services from 3GL programs. |
| ETP*vrs*.LIBR | (z/VSE system tapes only) VSE/SP LIBR backup format of the ETP product sub-library. This sub-library also contains the sample program for using ETP services from 3GL programs. |
| ETP*vrs*.MOD | (BS2000/OSD system tapes only) BS2000/OSD module library. |
| ETP*vrs*.LOAD | (z/OS system tapes only) z/OS load library. |
| ETP*vrs*.TAPE | (VM/CMS system tapes only) CMS TAPE dataset. |

- where *vrs* is the respective ETP version/release/system maintenance level number. For more information, see the *Report of Tape Creation* delivered with the distribution tape.

# General Information about ETP Installation

This section describes how to install ETP. The headings in each subsection are keyed to the corresponding job and step numbers of Software AG's System Maintenance Aid (SMA)-generated installation.

The following topics are covered:

- Loading the Installation Tape for z/OS
- Loading the Installation Tape for VM/CMS
- Loading the Installation Tape for z/VSE
- Loading the Installation Tape for BS2000/OSD
- Loading the ETP System Files
- Modifying the Natural Parameter Module

- Linking the Assembler Modules
- Specifying Master File Databases
- Installing the Maintenance Utility
- Loading Programs and Messages
- Installation for Operation with Natural Security

# Loading the Installation Tape for z/OS

If you are using SMA, refer to the *System Maintenance Aid* documentation (included in the current edition of the Natural documentation CD).

If you are *not* using SMA, follow the instructions below.

This section explains how to:

- Copy dataset `COPY.JOB` from tape to disk.
- Modify this dataset to conform to your local naming conventions.

The JCL in this dataset is then used to copy all datasets from tape to disk.

If the datasets for more than one product are delivered on the tape, the dataset `COPY.JOB` contains the JCL to unload the datasets for all delivered products from the tape to your disk.

After that, you will have to perform the individual install procedure for each component.

- Step 1 - Copy Dataset COPY.JOB from Tape to Disk
- Step 2 - Modify COPY.JOB on Your Disk
- Step 3 - Submit COPY.JOB

## Step 1 - Copy Dataset COPY.JOB from Tape to Disk

The dataset `COPY.JOB` (Label 2) contains the JCL to unload all other existing datasets from tape to disk. To unload `COPY.JOB`, use the following sample JCL:

```
//SAGTAPE JOB SAG,CLASS=1,MSGCLASS=X
//* --------------------------------
//COPY EXEC PGM=IEBGENER
//SYSUT1 DD DSN=COPY.JOB,
// DISP=(OLD,PASS),
// UNIT=(CASS,,DEFER),
// VOL=(,RETAIN,SER=tape-volume),
// LABEL=(2,SL)
//SYSUT2 DD DSN=hilev.COPY.JOB,
// DISP=(NEW,CATLG,DELETE),
// UNIT=3390,VOL=SER=volume,
// SPACE=(TRK,(1,1),RLSE),
// DCB=*.SYSUT1
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
//
```

where:

*hilev* is a valid high level qualifier
*tape-volume* is the tape volume name, for example: T12345
*volume* is the disk volume name

### Step 2 - Modify COPY.JOB on Your Disk

Modify the COPY.JOB on your disk to conform to your local naming conventions and set the disk space parameters before submitting this job:

- Set HILEV to a valid high level qualifier.
- Set LOCATION to a storage location.
- Set EXPDT to a valid expiration date.

### Step 3 - Submit COPY.JOB

Submit COPY.JOB to unload all other datasets from the tape to your disk.

## Loading the Installation Tape for VM/CMS

1. Position the tape for the TAPE LOAD command by calculating the number of tape marks, as follows:

   If the sequence number of ETP*vrs*.TAPE is *n* (shown on the *Report of Tape Creation*), you must position over 3 *n*-2 tape marks (that is, FSF 1 for the first dataset, FSF 4 for the second, etc.).

2. Access as disk A the disk that is to contain the ETP files.
3. Attach a tape drive to your virtual machine and mount the ETP installation tape.
4. Position the tape by issuing the following CMS command:

   ```
   TAPE FSF fsfs
   ```

   - where *fsfs* is calculated as described in Step 1, above.

5. Load the CMS installation material for ETP by issuing the CMS command:

   ```
   TAPE LOAD * * A
   ```

## Loading the Installation Tape for z/VSE

If you are using SMA, refer to the *System Maintenance Aid* documentation (included in the current edition of the Natural documentation CD).

If you are *not* using SMA, follow the instructions below.

This section explains how to:

- Copy dataset COPYTAPE.JOB from tape to disk.

- Modify this dataset to confom with your local naming conventions.

The JCL in this member is then used to copy all datasets from tape to disk.

If the datasets for more than one product are delivered on the tape, the member COPYTAPE.JOB contains the JCL to unload the datasets for all delivered products from the tape to your disk, except the datasets that you can directly install from tape, for example, Natural INPL objects.

After that, you will have to perform the individual install procedure for each component.

- Step 1 - Copy Dataset COPYTAPE.JOB from Tape to Disk
- Step 2 - Modify COPYTAPE.JOB
- Step 3 - Submit COPYTAPE.JOB

## Step 1 - Copy Dataset COPYTAPE.JOB from Tape to Disk

The dataset COPYTAPE.JOB contains the JCL to unload all other existing datasets from tape to disk. To unload COPYTAPE.JOB, use the following sample JCL:

```
* $$ JOB JNM=LIBRCAT,CLASS=0,                                          +
* $$ DISP=D,LDEST=(*,UID),SYSID=1
* $$ LST CLASS=A,DISP=D
// JOB LIBRCAT
* ***************************************
*     CATALOG COPYTAPE.JOB TO LIBRARY
* ***************************************
// ASSGN SYS004,nnn                              <------  tape address
// MTC REW,SYS004
// MTC FSF,SYS004,4
ASSGN SYSIPT,SYS004
// TLBL IJSYSIN,'COPYTAPE.JOB'
// EXEC LIBR,PARM='MSHP; ACC S=lib.sublib'       <------- for catalog
/*
// MTC REW,SYS004
ASSGN SYSIPT,FEC
/*
/&
* $$ EOJ
```

where:

*nnn* is the tape address
*lib.sublib* is the library and sublibrary of the catalog

## Step 2 - Modify COPYTAPE.JOB

Modify COPYTAPE.JOB to conform to your local naming conventions and set the disk space parameters before submitting this job.

## Step 3 - Submit COPYTAPE.JOB

Submit COPYTAPE.JOB to unload all other datasets from the tape to your disk.

# Loading the Installation Tape for BS2000/OSD

If you are not using System Maintenance Aid (SMA), use the procedure described below. In this procedure, the values specified below must be supplied.

To copy the datasets from tape to disk, perform the following steps:

- 1. Copy the Library SRVvrs.LIB from Tape to Disk
- 2. Copy the Procedure COPY.PROC from Tape to Disk
- 3. Copy all Product Files from Tape to Disk

## 1. Copy the Library SRV*vrs*.LIB from Tape to Disk

This step is not necessary if you have already copied the library SRV*vrs*.LIB from another Software AG installation tape. For further information, refer to the element #READ-ME in this library. The library SRV*vrs*.LIB is stored on the tape as a sequential file named SRV*vrs*.LIBS containing LMS commands. The current version *vrs* can be obtained from the *Report of Tape Creation*. To convert this sequential file into an LMS-library, execute the following commands:

```
/IMPORT-FILE  SUPPORT=*TAPE(FILE-NAME=SRVvrs.LIBS,       -
/   VOLUME=volser, DEV-TYPE=tape-device)
/ADD-FILE-LINK  LINK-NAME=EDTSAM, FILE-NAME=SRVvrs.LIBS, -
/   SUPPORT=*TAPE(FILE-SEQ=3), ACC-METH=*BY-CAT,                -
/   BUF-LEN=*BY-CAT, REC-FORM=*BY-CAT, REC-SIZE=*BY-CAT
/START-EDT
@READ  '/'
@SYSTEM 'REMOVE-FILE-LINK  EDTSAM'
@SYSTEM 'EXPORT-FILE  FILE-NAME=SRVvrs.LIBS'
@WRITE  'SRVvrs.LIBS'
@HALT
/ASS-SYSDTA  SRVvrs.LIBS
/MOD-JOB-SW  ON=1
/START-PROG  $LMS
/MOD-JOB-SW  OFF=1
/ASS-SYSDTA  *PRIMARY
```

where:

*tape-device* is the device-type of the tape, for example: TAPE-C4
*volser* is the VOLSER of the tape (see *Report of Tape Creation*)

## 2. Copy the Procedure COPY.PROC from Tape to Disk

To copy the procedure COPY.PROC to disk, call the procedure P.COPYTAPE in the library SRV*vrs*.LIB:

```
/CALL-PROCEDURE  (SRVvrs.LIB,P.COPYTAPE), -
/   (VSNT=volser, DEVT=tape-device)
```

If you use a TAPE-C4 device, you may omit the parameter DEVT.

### 3. Copy all Product Files from Tape to Disk

To copy all Software AG product files from tape to disk, enter the procedure COPY.PROC:

```
/ENTER-PROCEDURE  COPY.PROC, DEVT=tape-device
```

If you use a TAPE-C4 device, you may omit the parameter DEVT. The result of this procedure is written to the file L.REPORT.SRV.

## Loading the ETP System Files

(SMA Job I050, Step 5300)

The distribution tape contains the following empty files:

ETP*nnn*.SYSF

ETP*vrs*.SYSF has a Field Definition Table (FDT) suitable for containing the administration, confirmation and log files within one physical Adabas file. This file can be loaded using the Adabas ADALOD utility, or using SMA job I050.

The following files are suitable for installing individual ETP files:

ETP*vrs*.FDTA

ETP*nnn*.FDTA: contains the same FDTs as the ETP *nnn*.SYSF file, but in a format suitable for loading with the Adabas ADACMP utility.

ETP*vrs*.SYS*n*

ETP*vrs*.SYS*n*: are separate sample Adabas files in ADALOD format. These files have FDTs suitable for defining individual administration, log and confirmation files.

The Adabas utility parameters ISNREUSE=YES (for mainframe) and/or REUSE=ISN (for OpenVMS, UNIX or OS/2 systems) can be set to reuse freed ISNs as they become available for ETP master, replicate, confirmation, administration and/or log files.

## Modifying the Natural Parameter Module

Before linking the assembler modules as described in the next step, refer to the installation steps for Specifying Master File Databases and Defining the Administration File, later in this document.

Reassemble and relink the Natural parameter module (NATPARM) after you have modified it.

**Note:**
You must relink the modified Natural parameter module to all Natural nuclei that update a master file or start a replication task.

## Linking the Assembler Modules

You must link the ETPNUC module to all Natural nuclei that update a master file or start a replication task. Software AG recommends that you relink Natural with the ETPNUC module.

To avoid the need to relink Natural with ETPNUC, specify the Natural profile parameter RCA=ON or RCA=NATGWETP in the Natural parameter module to allow dynamic loading of ETP during Natural start-up. In this case, perform the following steps:

1. (z/OS) Rename the module ETPNUC in the ETP load library to NATGWETP.

   (VM/CMS) Rename ETPNUC TEXT to NATGWETP TEXT.

   (z/VSE) Use the following example to create a phase with the name NATGWETP:

   ```
   * $$ JOB JNM=LINK,CLASS=O,DISP=D
   * $$ LST CLASS=A,DISP=H
   // JOB LINK
   // LIBDEF OBJ,SEARCH=(SAGLIB.ETPvrs),TEMP
   // LIBDEF PHASE,CATALOG=libname.ssssss,TEMP
   // OPTION CATAL,LIST
    PHASE NATGWETP,*,NOAUTO
    INCLUDE ETPNUC
   /*
   // EXEC LNKEDT
   /*
   /&
   * $$ EOJ
   ```

2. (z/OS, z/VSE) Place the NATGWETP module in a library from which your TP monitor or batch system can perform dynamic loads. In Com-plete systems, NATGWETP can be loaded as a resident program.

## Specifying Master File Databases

Use the NTDB macro in the Natural parameter module (NATPARM) to specify the databases containing the master files:

```
NTDB ADAV7,dbid,ETP
```

- or:

```
NTDB ADAV7,(dbid,dbid,...),ETP
```

- where *dbid* specifies one or more databases separated by commas, each containing one or more master files.

**Note:**
You can define the same database as an ENTIRE database and also as an ETP database if you specify both options for the NTDB macro: NTDB ADAV7,*dbid*,ETP,ENTIREIt is also possible to specify the databases containing the master files dynamically at Natural startup using the DB parameter:
DB=(ADAV7,dbid,ETP) or
DB=(ADAV7,(*dbid*,*dbid*,...),ETP)

The first time that a database which is defined as an ETP database is accessed, ETP automatically obtains a buffer (ETPSIZE) of the required size (approx. 10 KB). Any value specified for the Natural profile parameter ETPSIZE is ignored. The size of the ETPSIZE parameter should be left at its default (ETPSIZE=0).

# Installing the Maintenance Utility

The ETP maintenance utility is a menu-based control facility for defining and managing master and replicate files. Although not required, Software AG recommends installing the maintenance utility on every node containing a master file.

## Loading Programs and Messages

(Job I061, Steps 5300/5301)

To install the maintenance utility, perform the following steps:

1. Using the `INPL` facility, load the `SYSETP` library, including all objects, from the installation tape into your Natural `FNAT` file.
2. Using the Natural utility `ERRLODUS`, load all error messages from the `SYSERR` library.

**Warning:**
**The `SYSETP` library should be protected against general access with Natural Security or an equivalent security facility to prevent uncoordinated changes to the administration file. Such changes can destroy the consistency and integrity of the master and replicate files.**

## Installation for Operation with Natural Security

ETP and the maintenance utility support the concept of functional security, meaning that selected functions can be allowed or disallowed under control of Natural Security. When a user is restricted by Natural Security from performing a specific ETP function, the function is not displayed on the user's corresponding menu.

**Note:**
Disallowing the general dialog functions - for example, `EXIT`, `CANCEL` - can cause unpredictable results.

To install ETP if Natural Security is already installed, perform the following steps:

1. Log on to library `SYSSEC`;
2. Issue the command `ADD LIBRARY SYSETP`;
3. Enter the appropriate information;
4. Select the menu choice **Additional Options**;
5. In the **Additional Options** menu, select **Functional security**;
6. Define the command processor `WADNCP1` for library `SYSETP`;
7. Enable or disable keywords (for example, `DELETE`, `MASTER` ...) or functions (for example, `REPLICATE TRANSACTIONS`), depending on your installation requirements. Note that any restrictions you define here apply to *all* users.

In a similar way, you can restrict the availability of keywords or functions for every user that has access to SYSETP.

# Additional Information about ETP Installation

## Defining the Administration File

Each database containing an ETP master file should also contain an administration file to hold all master and replicate file definitions. The logical file ID of the administration file must be 200. To define the administration file for your Natural applications, specify the NTLFILE macro in the Natural parameter module (NATPARM):

```
NTLFILE 200,dbid,filenumber
```

- where *filenumber* is the physical file number of the administration file and *dbid* is its physical database ID.

The administration file setting can also be changed dynamically at the start of the Natural session, using the Natural profile parameter LFILE:

```
LFILE=(200,dbid,filenumber)
```

The administration file must always be defined before using ETP; if the ETP maintenance utility is used, the utility prompts the user for an administration file if no valid LFILE definition is found.

ETP also works correctly, even if the Natural macro NTTF or the profile parameter TF is used. However, ETP may not work properly if you have an Adabas user exit installed which modifies the database ID or file number in the Adabas Control Block.

**Warning:**
**Do not change any of the information in the administration file while it is being used by a replication task.**

To install an administration file in your database, use either SMA job I050 or the Adabas ADALOD utility to load ETP*vrs* .SYS1. You can keep the administration file quite small since it contains only a single record for every:

- master file definition
- replicate file definition
- log file
- replicate database
- user profile

## Defining a Master File and its Log File

A master file is normally an existing Adabas file. If the master file is new, you must first create the file as a normal Adabas file. This description assumes that the file to be defined as a master already exists.

Before a master file can be defined, an administration file must first be defined. See the previous section, *Defining the Administration File*, for more information.

After defining the administration file, perform the following steps to define a master and log file:

1. Stop all updating on the file that is to be defined as a master file;
2. Copy the file to be defined as a master file. Software AG recommends using the Adabas ADAULD utility for this purpose; ADASAV may also be used, but only where the file will be reloaded on the same device type as before. Note the exact date and time of the copy;
3. Using the ETP maintenance utility's **Master File Definition** menu, specify the master file and log file. If desired, all master files on a database can share the same log file;
4. Restart database operation to make the master file available again.

From this time on, all changes applied to the master file will be recorded in the log file. Any replicate files of the master file can now be defined without interfering with the master file operation, providing the log file contains all changes applied to the master file since the copy in Step 2 was created. For more information, see *Updating the Administration File*.

Transaction logging will start as soon as a master file is defined and a new Natural session with the appropriate administration file is started or the master file is updated from within the Natural session which was used to define the master file. Therefore, the procedure described above might not be applicable, especially when a new master file is to be defined in a running environment. If it cannot be guaranteed that no updates are applied to the to-be-defined master file that is to be defined while the above steps are executed, perform the following steps (note that it is required that the log file specified for a master file is empty when the master file is defined):

1. Stop the transaction replication processes;
2. Define the new master file;
3. Shut down the database containing the new master and log files;
4. Copy the master file as described above;
5. Use the Adabas utility ADADBM with the REFRESH function to refresh the log file;
6. Restart database to make the master file available again; transaction logging will start as soon as a user updates the master file;
7. Copy the master file's contents into the replicate file;
8. Restart transaction replication.

To install a log file in your database, use either SMA job I050 or the Adabas ADALOD utility to load ETP*vrs*.SYS2.

The number of records in the log file depends on the number of transactions that update master files between two successive invocations of the **Clean Up Log File** ETP maintenance function (provided that all transactions are replicated). The approximate number of log file records equals:

*(transaction count)**(updates per transaction, +1)

When loading the log file, the ADALOD parameter PGMREFRESH=YES is required if you want the **Clean Up Log File** function to refresh the log file for improved performance.

**Note:**
It is impossible to refresh a log file that is also used as an administration or confirmation file.

## Defining and Initializing the Replicate and Confirmation Files

After the master file is defined, the replicate files can be defined. A confirmation file must also be defined on each database where a replicate file is defined. If desired, multiple replicate files on a database can share the same confirmation file. To define a replicate file, perform the following steps:

1. Using the Add Replicate File Definition menu of the on-line maintenance utility, specify the replicate, master, and confirmation files' DBIDs and file numbers;

2. Load the unloaded copy of the master file into the replicate file, using the Adabas utility ADALOD (if the file was unloaded with ADAULD) or ADASAV (if the file was unloaded with ADASAV). Specify the parameter USERISN=YES for the related Adabas mainframe utilities. For the replicate files on OpenVMS, Windows or UNIX systems, the option USERISN must be specified when loading a non-empty file unless the distribution key is used as the replication criterion. If an empty replicate file is created, the option USERISN or parameter setting USERISN=YES is not required.

   When specifying the MAXISN parameter while defining the replicate file you should remember that, when using the records' ISNs as the replication criterion, the Address Converter (AC) is not automatically extended. This can occur if the ETP N2 calls that add new replicate file records specify an ISN value that exceeds the file's MAXISN value; in such a case, a response code 113 is returned. Specify a MAXISN value large enough for future extensions of the replicate file.

3. Using the **Display Transactions** function of the maintenance utility, check for any log file entries that have been made since the master file was copied;

4. If a replicate file contains a subset of the master file records, you should either delete all unneeded records as defined by the specified distribution key ranges or copy only the selected subset of records from the master file. This can be done using one of the following methods:

   - Use a Natural program to delete the unneeded records from the replicate file;
   - Use a Natural program to copy only the selected records from the master file to an intermediate file, which is then subsequently copied to the replicate file;
   - Use the SELCRIT and SELVAL parameters of the Adabas ADAULD UNLOAD utility function to select only the subset for unloading. This is the recommended method.

**Warning:**
**After the replicate file has been initialized, it should not be manually changed. Otherwise, the consistency with the master file could be destroyed.**

5. To install a confirmation file in your mainframe database, use either SMA job I050 or the Adabas ADALOD utility to load ETP*vrs*.SYS3 from the distribution tape. For confirmation files on OpenVMS, Windows or UNIX systems, use the confirmation file field definitions in file ETP*vrs*.FDTA to supply field definitions for the Adabas utility ADAFDU. For every replicate file that uses the confirmation file, the latter file contains a single record.

6. If the log file contains any entries for the master file, start the replication task for the replicate file using the **Replicate Transactions** function of the ETP maintenance utility. The task checks the appropriate administration file for master or replicate files in the file range to be processed. The task then synchronizes the master and replicate file by applying all updates to the replicate file that are not already applied. The replicate file is now available for use.

To add other replicate files of an existing master file, perform the steps above after creating an up-to-date copy of the master file. Note that when creating a replicate file, no logged changes to the master file should be removed from the log file. If this rule is followed, a replicate file can be added without affecting the normal mode of operation.

## Related Parameter Settings

### Natural WH Parameter

When running multiple ETP replication tasks in parallel, specify WH=ON to avoid NAT3145 errors (`record already in hold status for another user`) when two tasks attempt to access the same record simultaneously.

### Reusing ISNs in ETP Files

The Adabas utility parameters ISNREUSE=YES (for mainframe) and/or REUSE=ISN (for OpenVMS, Windows or UNIX systems) can be set to reuse freed ISNs as they become available for ETP master, replicate, confirmation, administration and/or log files.

### Transactions with Many Updates

If transactions that include a lot of updates are to be logged, increase the value of the ADARUN LDEUQP parameter. The required size for transaction logging can be computed as:

```
LDEUQP = (updates per transaction * 29)
```

## Modifying the WADUSER2 and/or WADUSER3 User Exits

The two user exits WADUSER2 and WADUSER3 are delivered in souce form in the library SYSETP.

### Optional User Exit WADUSER2

The optional user exit WADUSER2 is a Natural subprogram for controlling file replication. WADUSER2 is called after ETP decides whether the record in question is to be replicated or not.

The WADUSER2 user exit, defined as User Exit 2, is called only if the user exit option is specified when a master file is defined (see *Master File Task Screens*). An example of WADUSER2 is included in the SYSETP library.

### Message Handler WADUSER3

The subprogram WADUSER3 is used to display all messages issued by the replication task. WADUSER3 can be modified to filter the task messages and, if desired, send them directly to the operator console. The WADUSER3 subprogram receives the error number, severity level and the message text from the replication task. This allows the user to select the messages to be displayed. By means of the Natural CMWTO entry (for an example, see the program WTO in the Natural library SYSEXTP), the messages can be sent to the operator console.

**Warning:**
**The user exits WADUSER2 and WADUSER3 should neither issue Adabas calls that update a database file nor should they issue any End Transaction (ET) or Back Out Transaction (BT) commands; otherwise, the results are unpredictable.**

## Running ETP in Batch Mode

(SMA I200, Step 5300)

In batch mode, command execution is possible only by means of direct commands. For a list of direct commands and their minimum abbreviations, see *Entering Direct Commands*.

The following example is for a batch file for starting a replication task:

```
LOGON SYSETP                                          (1)*
MENU                                                  (2)*
REPLICATE TRANSACTIONS                                (3)
1,1,65535,65535,1,65535,00:30:00,200,1000,N,EXIT      (4)
EXIT                                                  (5)
FIN                                                   (6)
```

* If Natural Security is installed, lines (1) and (2) may have to be changed (see *Starting the ETP Maintenance Utility*).

Line (4) contains the parameters for the corresponding "replicate transactions" ETP maintenance utility screen. Parameters are entered from top to bottom, left to right. Line (5) exits from the ETP maintenance utility.

**Warning:**
**Software AG recommends that replication tasks be started in batch mode.**

The following example is for a batch file for deleting successfully replicated transactions:

```
LOGON SYSETP                                          (1)*
MENU                                                  (2)*
CLEANUP LOGFILE                                       (3)
1,1,65535,65535,,,N,00:30:00,200,1000,EXIT            (4)
EXIT                                                  (5)
FIN
```

* If Natural Security is installed, lines (1) and (2) may have to be changed (see *Starting the ETP Maintenance Utility*). Line (4) contains the parameters for the corresponding "clean up log file" ETP maintenance utility screen. Parameters are entered from top to bottom, left to right. Line (5) exits from the ETP maintenance utility.

**Warning:**
**Software AG recommends that tasks that delete successfully replicated transactions are started in batch mode.**

If a window is displayed in batch mode, all fields are protected; the reason for this is that in most cases it is not possible to determine the number of selectable items. Therefore, the only meaningful command is PROCESS. The following is an example to reset the in-use flag for all replicate files:

```
MENU
RESET IN-USE * *
PROCESS
EXIT
EXIT
FIN
```

To run the above examples without problems, the following parameters in the Natural parameter module must be specified:

```
ID=',' (default setting)
IM=D
PC=OFF (default setting)
```

**Note:**
Either the Natural statement SET CONTROL '+' or the terminal command %= cancel the effect of PC=OFF.

## Using ETP with 3GL Programs

To handle Adabas direct calls issued by 3GL (COBOL, PL/I, etc.) programs, ETP provides an interface program, ETPDB3GL, which is part of ETPNUC. Because Adabas call handling varies with the operating systems and TP monitors, it may be necessary for you to write a program that passes Adabas calls to the ETPDB3GL interface program; this depends on your installation requirements.

For 3GL programs running under CICS, the ETP Interface for CICS (ETC) is available as a separate Software AG Selectable Unit. For more information, see the documentation of the Entire Transaction Propagator CICS Interface (ETC).

The ETPDB3GL interface program provides two ways for your 3GL programs to use ETP:

- Invoking ETPDB3GL directly from your 3GL program, using changed Adabas calls;
- Invoking ETPDB3GL from a user-written "pseudo-Adabas" program, using existing Adabas calls.

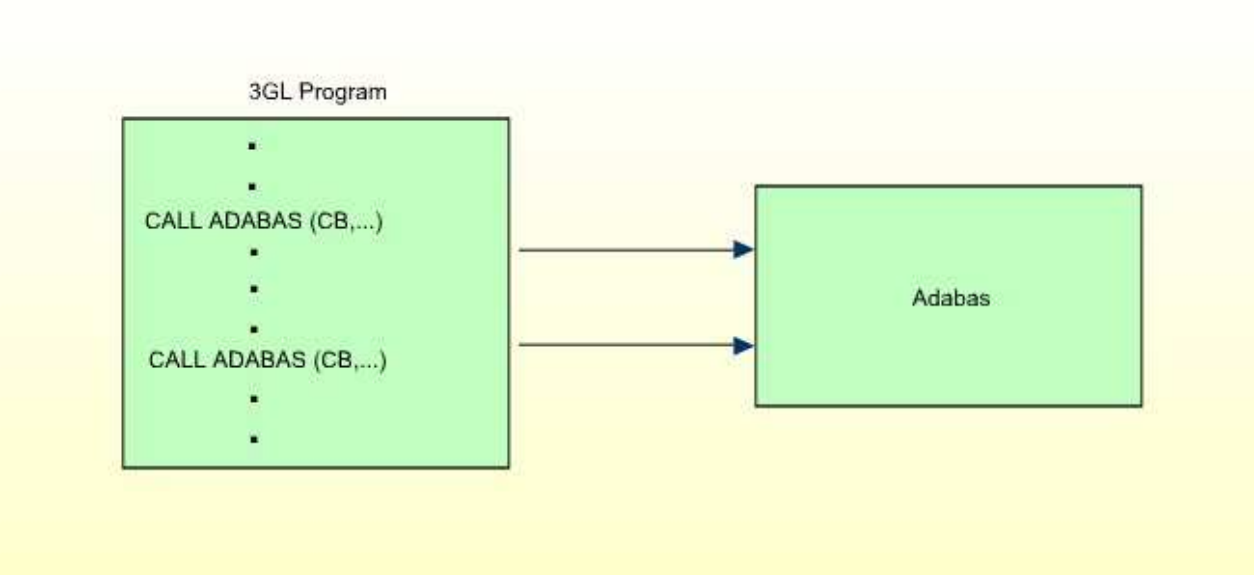
Using the interface requires initialization calls to provide ETPDB3GL with information such as the database ID and file number of the administration file. In addition to the normal Adabas parameters, you must also provide a storage area as work storage.

Let’s assume that your 3GL program now contains calls to Adabas similar to the following:

```
                .
                .
CALL ADABAS (CB,FB,RB,SB,VB,IB)
                .
                .
```

Here, all program calls are issued directly to Adabas, as shown in the following figure:

### 3GL Call Schema for Non-ETP Programs



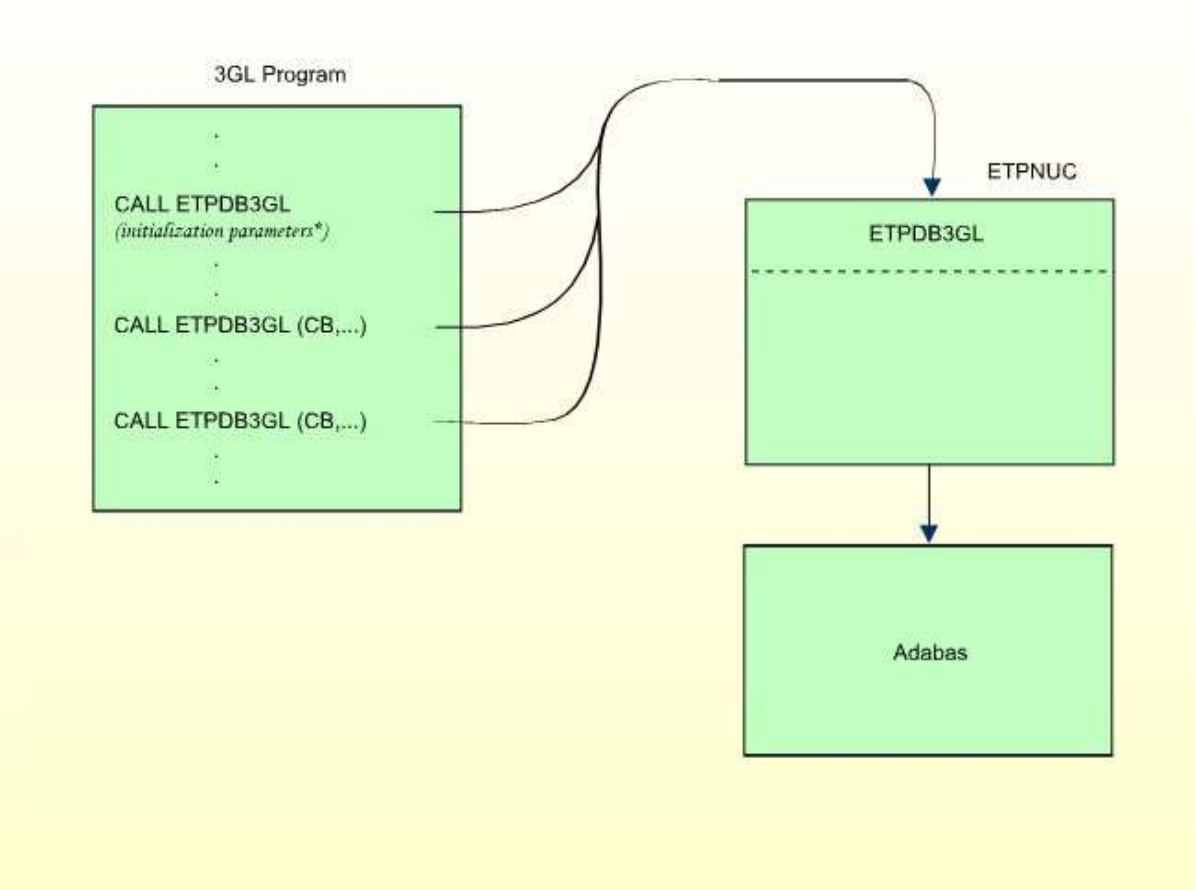

With ETP and the ETPDB3GL interface program, you now have two possibilities to pass calls:

1. Change your 3GL programs to call the interface module ETPDB3GL instead of Adabas; that is, instead of CALL ADABAS, code CALL ETPDB3GL. Include the initialization calls and modify all other Adabas calls so that they look like the following:

   ```
                   .
                   .
   CALL ETPDB3GL (CB,FB,RB,SB,VB,IB,WORKAREA)
                   .
                   .
   ```

   - where WORKAREA is the required work storage for ETP. Now, your programs have the following call logic:

**3GL ETP Calling with Changed Adabas Calls**



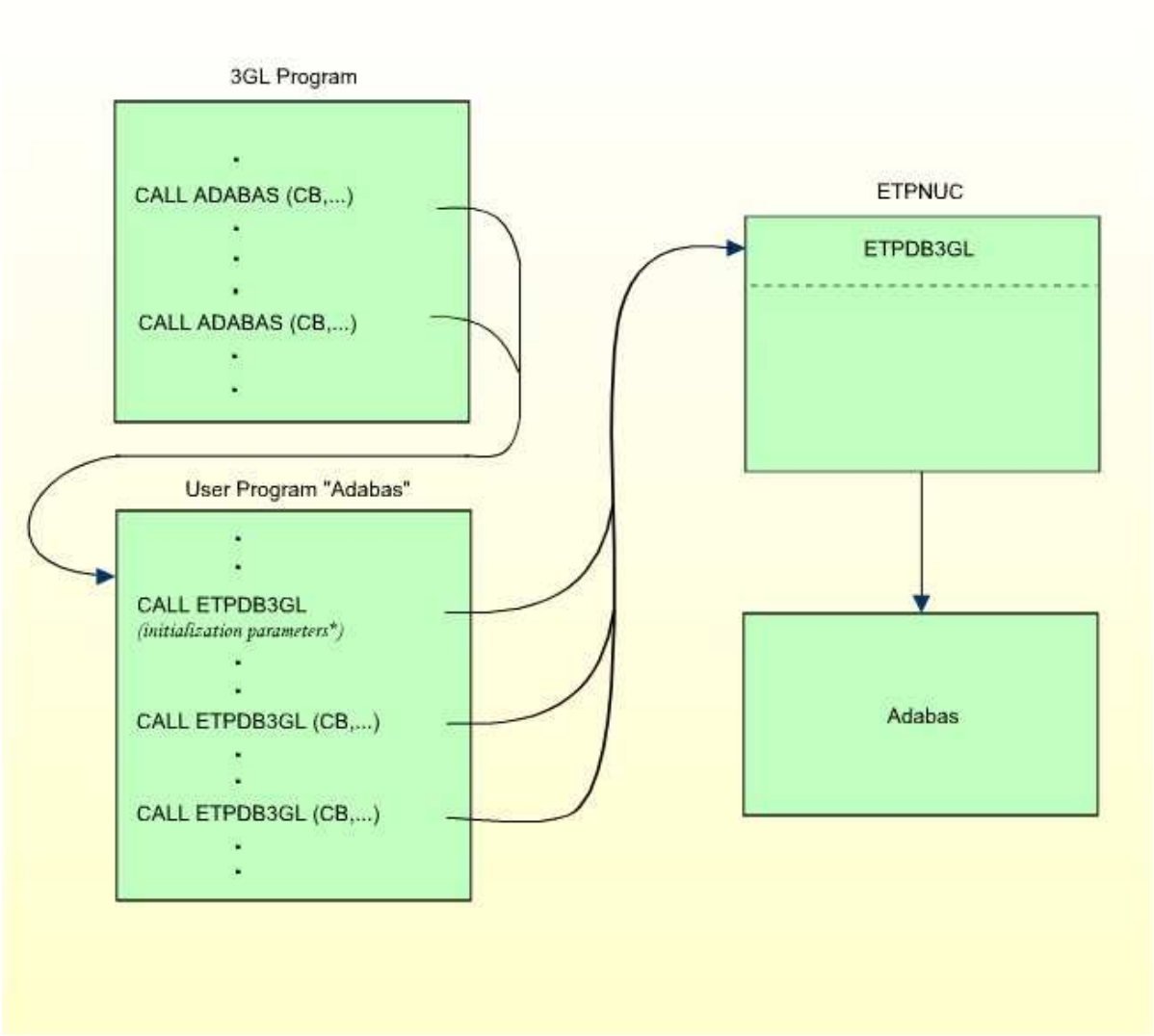

* The initialization parameters are described in the sample program which is provided in dataset ETP*vrs*.SRCE.

2. Write a program that accepts Adabas calls, performs initialization calls to ETPDB3GL, and passes all Adabas calls to ETPDB3GL (see figure above). Name the program ADABAS, and link it to your application programs. You may need to change your "compile and link" JCL to ensure that your application will always pass Adabas calls to the interface.

   This user-written program ADABAS provides all necessary information to ETPDB3GL. It also must provide the address of the "real" Adabas, which is usually an entry point in the normal Adabas interface (for example, ADAUSER). To avoid linkage editor errors caused by multiple entry points for the name ADABAS, you may also have to use the linkage editor to change the normal Adabas interface.

   With this alternative for passing calls, you do not need to change your 3GL programs. Note, however, that the program ADABAS must also specify work storage as in choice [1], above.

**3GL ETP Calling with Unchanged Adabas Calls**




* The initialization parameters are described in the sample program which is provided in dataset ETP*vrs*.SRCE.

A sample program using this second alternative for z/OS batch mode is provided in dataset ETP*vrs*.SRCE. This sample program also contains the description of the interface to program ETPDB3GL. This alternative has the call logic shown in the figure above.

Both alternatives require you to link the module ETPNUC to all your applications for which ETP services will be used; ETPNUC contains the interface program ETPDB3GL.