

## **Natural für Großrechner**

### **Entire Transaction Propagator**

Version 4.2.6 für Großrechner

Februar 2010

Dieses Dokument gilt für Natural für Großrechner ab Version 4.2.6 für Großrechner.

Hierin enthaltene Beschreibungen unterliegen Änderungen und Ergänzungen, die in nachfolgenden Release Notes oder Neuausgaben bekanntgegeben werden.

Copyright © 1979-2010 Software AG, Darmstadt, Deutschland und/oder Software AG USA, Inc., Reston, VA, Vereinigte Staaten von Amerika, und/oder ihre Lizenzgeber..

Der Name Software AG, webMethods und alle Software AG Produktnamen sind entweder Warenzeichen oder eingetragene Warenzeichen der Software AG und/oder der Software AG USA, Inc und/oder ihrer Lizenzgeber. Andere hier erwähnte Unternehmens- und Produktnamen können Warenzeichen ihrer jeweiligen Eigentümer sein.

Die Nutzung dieser Software unterliegt den Lizenzbedingungen der Software AG. Diese Bedingungen sind Bestandteil der Produktdokumentation und befinden sich unter <http://documentation.softwareag.com/legal/> und/oder im Wurzelverzeichnis des lizenzierten Produkts.

Diese Software kann Teile von Drittanbieterprodukten enthalten. Die Hinweise zu den Urheberrechten und Lizenzbedingungen der Drittanbieter entnehmen Sie bitte den "License Texts, Copyright Notices and Disclaimers of Third Party Products". Dieses Dokument

ist Bestandteil der Produktdokumentation und befindet sich unter <http://documentation.softwareag.com/legal/> und/oder im Wurzelverzeichnis des lizenzierten Produkts.

---

# Inhaltsverzeichnis

1 Entire Transaction Propagator .....	1
Natural Documentation .....	3
Adabas Documentation .....	3
Entire Net-work Documentation .....	3
2 What's New with ETP Version 1.5.2 .....	5
New Features, Enhancements and Corrections .....	6
Natural Version Required .....	7
ETP 1.5.2 Compatibility with Earlier Releases .....	7
3 Introducing ETP .....	9
ETP Concept .....	10
Data Integrity Choices .....	12
How ETP Operates .....	13
ETP File Structure .....	14
Replication Task Control .....	15
Tailoring and Tuning ETP .....	17
Resynchronization Modes .....	17
4 ETP Installation .....	23
Prerequisite Requirements and ETP Release Contents .....	24
General Information about ETP Installation .....	25
Additional Information about ETP Installation .....	33
5 ETP Setup .....	45
Getting Started .....	46
General Considerations when Defining Files .....	46
Performance and Maintenance Considerations .....	48
6 Programming ETP .....	51
ETP Programming Principles .....	52
ET-Time User Program .....	55
7 ETP Operations and Administration - Overview .....	57
8 Overview of General Tasks .....	59
ETP Maintenance Main Menu .....	60
9 ETP Maintenance Utility .....	61
Starting the ETP Maintenance Utility .....	62
Selecting and Performing Menu Tasks .....	63
Entering and Displaying Information .....	64
Priority of Command Input .....	66
Entering Direct Commands .....	66
Dialogue Functions .....	67
Errors and Warnings .....	69
Selecting Main Menu Tasks .....	70
10 Master File Task Screens .....	71
Master File Task Selection Menu .....	72
Add Master File Definition .....	73
Modify Master File Definition .....	74

Display Master File Definition .....	76
Delete Master File Definition .....	77
11 Replicate File Task Screens .....	79
Replicate File Task Selection Menu .....	80
Add Replicate File Definition .....	81
Modify Replicate File Definition .....	83
Display Replicate File Definition .....	85
Delete Replicate File Definition .....	86
Display Error Log for Replicate File .....	87
Reset Replicate File Error State .....	88
Reset Replicate File In-Use State .....	89
12 Replicating Logged Transactions .....	91
Function Description .....	92
Starting Replication Tasks Manually .....	92
Starting Replication Tasks with a User Program .....	92
Self-Restart Operation .....	92
Error Handling during Replication .....	93
Displaying the Replicate Database Summary .....	93
Processing of Replication Task Messages .....	94
Replicate Transactions .....	94
13 Controlling Execution of Asynchronous Tasks .....	97
Function Description .....	98
Control Execution of Asynchronous Tasks Screen .....	98
14 Deleting Successfully Replicated Transactions .....	101
Function Description .....	102
Clean Up Log File .....	102
Processing of Clean-Up Task Messages .....	104
15 Displaying X-Ref Transaction State .....	105
Function Description .....	106
X-Ref Transaction State Screen .....	106
X-Ref Transaction State User Application Programming Interface .....	107
16 Using Special Functions .....	109
Special Functions Screen .....	110
Display Transactions .....	110
Delete Transactions .....	112
Modify Confirmation File .....	114
Modify System Profile .....	115
Modify User Profile .....	116
Check Definitions for Performance Problems .....	118
Change Administration File .....	119
17 Executing a Natural Command .....	121
Function Description .....	122
Invoking and Using the Function .....	122
18 ETP Restart and Recovery .....	125
ETP Restart and Recovery Behaviour .....	126

---

General Errors that Can Occur .....	126
File Saving Requirements for Error Recovery .....	126
Recovering from the General Types of Failures .....	127
Correcting NAT3400, NAT9988 and NAT3606 Errors .....	140
19 Correcting Unlogged Transactions .....	145
Cause .....	146
Remedial Action .....	146
20 Database Errors during Logging of Master File Transactions .....	149
Isolating Call Errors with TEST DBLOG .....	150
Automatic ETP Back-Out Transaction .....	150
21 Reporting ETP Errors .....	151
22 ETP CICS Interface - Overview .....	153
23 What's New with ETC Version 1.5.2 .....	155
New Features, Enhancements and Corrections .....	156
Changes Planned for the Next Major Release of ETC .....	156
Operating/Teleprocessing System and Software Required .....	157
Restrictions .....	157
24 ETC Installation and Operation .....	159
Overview on ETC Operation .....	160
Prerequisite Requirements and ETC Release Contents .....	162
General Information on ETC Installation .....	162
25 ETC Installation Procedure .....	165
Loading ETC from the Installation Tape onto the Disk .....	166
Defining the ETC Environment with the ETCPARM Macro .....	169
Example of the ETCPARM Macro .....	174
Assembling ETCPARM and Linking ETC .....	174
26 Glossary of ETP Terms .....	179

---



# 1 Entire Transaction Propagator

---









- Natural Documentation ..... 3
- Adabas Documentation ..... 3
- Entire Net-work Documentation ..... 3

Normally, a replicated file requires an intricate control process to ensure data integrity in all file copies after each change. For distributed systems with a high ratio of read transactions compared to write transactions, however, such critical control may be unnecessary. The Entire Transaction Propagator (ETP) provides an alternative replicated file concept using a less critical control process, but with virtually all the other advantages of replicated files.

ETP allows Adabas users to have duplicate, or replicate, database files in a single database or distributed network. The copies can be distributed throughout a network to provide quick, economical access at user locations.

This documentation contains all information required to install and use Software AG's Entire Transaction Propagator (ETP) for replicate database files. The documentation describes the replicate file concept used by ETP, how it operates and how to control ETP using its online maintenance system.

The ETP documentation is intended for those who are planning for distributed database processing, are looking for a solution that combines databases into a central database resource, or are installing or already using a distributed Adabas/Net-work system.

 <b>What's New</b>	Describes the new features introduced with the current version of ETP.
 <b>Introducing ETP</b>	Describes the ETP concept for replicated files. Readers of this chapter will see the advantages of an ETP environment, and be able to recognize the benefits for their own specific requirements.
 <b>ETP Installation</b>	How to install ETP.
 <b>ETP Setup</b>	Provides start-up information as well as tips for operating ETP most effectively.
 <b>Programming ETP</b>	Explains the ETP user program considerations.
 <b>Operations and Administration</b>	Describes the ETP online maintenance and how the ETP administrator can use them to manage ETP to ensure the most effective operation.
 <b>ETP CICS Interface</b>	Describes Software AG's Entire Transaction Propagator CICS Interface (ETC) for installing and running the Entire Transaction Propagator (ETP) with CICS.
 <b>Glossary of ETP Terms</b>	Defines the terms that are referred to in the ETP documentation.

## Related Literature

---

The following Software AG documentation applies and may be useful when installing and running ETP:

## Natural Documentation

---

- Natural Parameter Reference
- Natural Utilities
- Natural Installation
- Natural Operations for Mainframes
- Natural Statements
- Natural Programming Guide
- Natural Messages and Codes(including the Entire Transaction Propagator Abend Codes)

## Adabas Documentation

---

- Adabas Implementation and Maintenance Manual
- Adabas Operations Manual
- Adabas DBA Reference Manual
- Adabas Command Reference Manual
- Adabas Messages and Codes
- Adabas Security Manual

## Entire Net-work Documentation

---

- Net-work Installation and Operation
- Net-work Release Notes

---

## 2 What's New with ETP Version 1.5.2

---

- New Features, Enhancements and Corrections ..... 6
- Natural Version Required ..... 7
- ETP 1.5.2 Compatibility with Earlier Releases ..... 7

## New Features, Enhancements and Corrections

---

This section describes the new features, enhancements and corrections that apply to the Version 1.5.2 of the Entire Transaction Propagator (ETP).

■ **Error Corrections:**

ETP Version 1.5.2 contains all Zaps and INPL updates applied to ETP Version 1.5.1 as error corrections.

■ **Database IDs:**

ETP now supports DBIDs in the range of 1 to 65535. However, the value of 255 remains excluded.

■ **File Numbers:**

ETP now supports file numbers in the range of 1 to 65535.

■ **Master Files and Log Files:**

The maximum number of master files per database has been increased to 512. Also, the maximum number of log files per database IDs has been increased to 512.

■ **Replicate Files:**

For a specific log file, transactions may be replicated to up to 65535 databases. A specific replicate database may have a maximum of 512 replicate files defined for which transactions are replicated from that log file. Other log files may also have up to 512 replicate files belonging to that database.

■ **Conversion of Existing Administration Files:**

Existing administration files will be converted automatically when the MENU program is started.

■ **New Product Documentation Medium:**

The following changes and enhancements apply to the ETP product documentation:

- The ETP product documentation is supplied online on CD-ROM as it is common practice for all Natural products.
- Hardcopies of the documentation can be printed out from the PDF files that come with the documentation.

For information about online documentation usage, see *Documentation-Related Frequently Asked Questions (FAQ)* on the Natural CD-ROM overview page.

---

## Natural Version Required

---

Entire Transaction Propagator Version 1.5.2 requires that Natural Version 4.2. is installed.

The prerequisite operating/teleprocessing systems and required versions of other Software AG products are to be found in the current *Natural Release Notes*.

---

## ETP 1.5.2 Compatibility with Earlier Releases

---

It is essential that the `ETPNUC` module and the ETP Natural programs in library `SYSETP` have the same version. Using a Version 1.5.2 `ETPNUC` module with ETP Version 1.4.1 Natural programs - or vice versa - can cause unpredictable results.

The Natural parameter `ETPSIZE` is still accepted by Natural, but it is no longer necessary to specify this parameter. The required storage (approx. 10 KB) will be automatically allocated when the first call to an ETP database is issued. Any value specified for the `ETPSIZE` parameter will be ignored. The size of the `ETPSIZE` parameter should be left at its default (`ETPSIZE=0`).

As the Natural `DATSIZE` parameter is automatically adjusted to the required length, it is no longer necessary to adjust the value of the Natural `DATSIZE` parameter for a Natural environment running a replication task. However, the `DATSIZE` in such an environment will be approximately 170 KB.

After ETP Version 1.5.2 is installed and when the `MENU` command is invoked for the first time in the ETP maintenance utility, the ETP administration file is automatically migrated to the ETP Version 1.5.2 format and an appropriate message is displayed.

Afterwards, any attempt to access the migrated administration file from an earlier version of the ETP maintenance utility is denied.

Before enabling the new ETP maintenance utility, all log files must be empty and all log and confirmation files must be accessible for the migration process. Migration of the administration file from ETP Version 1.4 format to ETP Version 1.5 format is performed automatically when the ETP Version 1.5 maintenance utility is invoked for the first time. If a previous ETP version is installed, you should use the Natural utility `SYSMAIN` to delete the contents of library `SYSETP` before installing ETP Version 1.5.2. Following installation of ETP Version 1.5.2, the maintenance utility of any earlier ETP version will be denied access to the administration file.

Otherwise, ETP Version 1.5.2 is compatible with ETP Version 1.4.1 The FDT of the administration, log and confirmation files remains unchanged.





# 3 Introducing ETP

---

- ETP Concept ..... 10
- Data Integrity Choices ..... 12
- How ETP Operates ..... 13
- ETP File Structure ..... 14
- Replication Task Control ..... 15
- Tailoring and Tuning ETP ..... 17
- Resynchronization Modes ..... 17

## ETP Concept

---

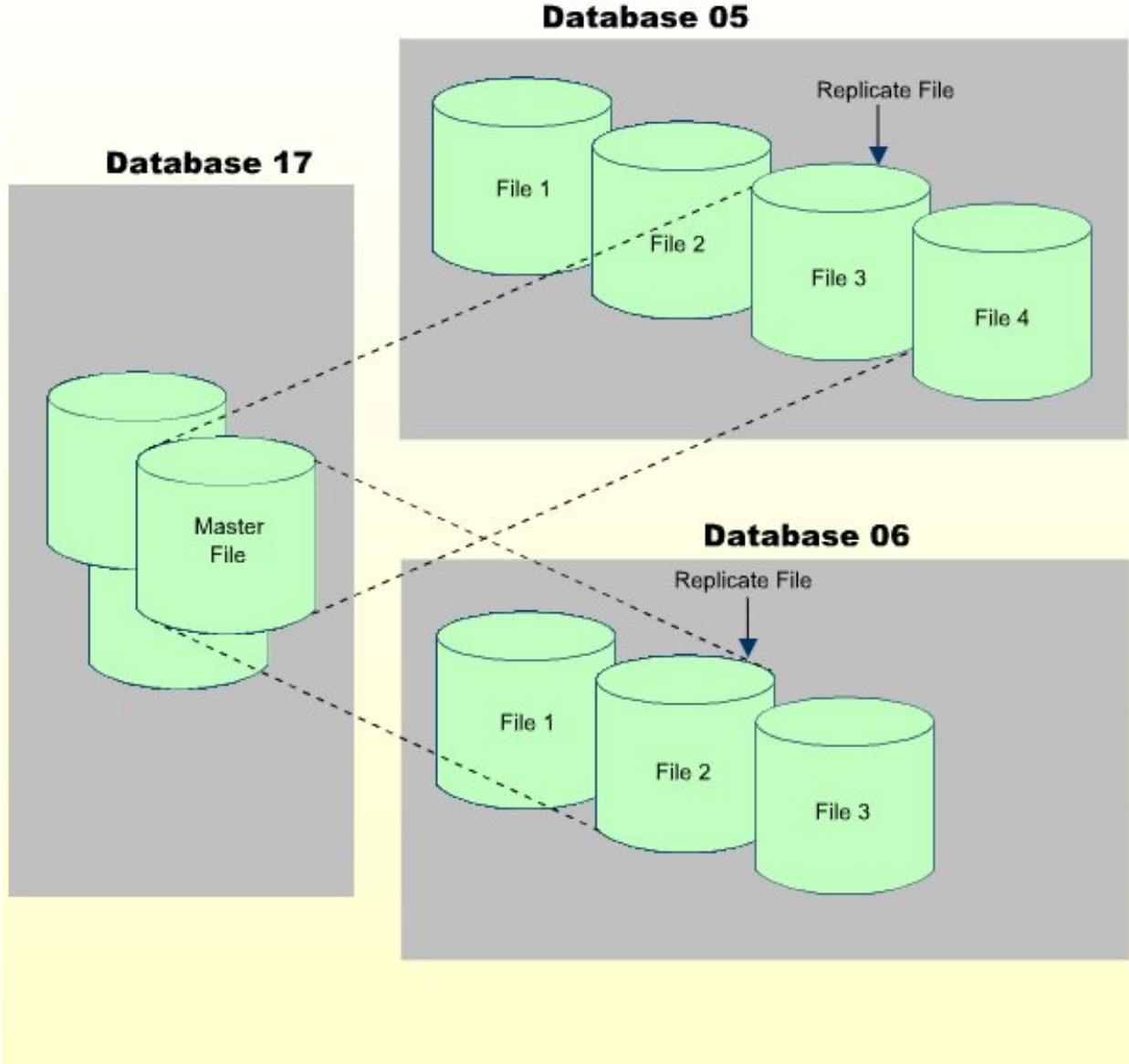
The concept of a distributed database provides much greater operating efficiency and flexibility while at the same time offering almost unlimited data capacity. Such a „networked“ database structure means that the portion of the database data needed by, for example, Corporate Operations or Personnel can be located on local systems and still be available corporate-wide as part of the common database resource.

One particularly appealing feature of distributed databases is the possibility of having duplicate copies of data at those locations where the data is needed most. This concept allows duplicate copies of a data file to be located throughout the database network, yet the copies are viewed logically by users as a single file.

However, maintaining the duplicate data in each location has generally also meant a sophisticated control mechanism to ensure simultaneous integrity among all copies of the duplicate data. Although effective and proven, such „fail-safe“ data integrity is often not necessary for some user environments. Software AG's ETP offers a cost-effective alternative to these more sophisticated systems for installations that do not require absolutely simultaneous integrity in their duplicate data resource.

ETP holds the duplicate data at each location in a *replicate file* (see the figure [A Basic ETP File Structure](#)). The replicate file is a read-only image of data held in a *master file*. The replicate files are „refreshed“ periodically if changes have been made to the master file. If and when refreshing is done and which replicate files are refreshed are the choice of the ETP administrator.

A Basic ETP File Structure



## Data Integrity Choices

---

Conventional database design standards require replicated files to have „full integrity“. This means that, whenever a logical record is available to a user program, all replicated copies of the record must contain identical and up-to-date information, regardless of where or how often the record is duplicated. Although a program performs an update transaction to only one local copy, all other copies must first be changed to reflect the transaction before another program refers to that information. It is the job of the distributed database management system (DDBMS) to ensure that all copies of the replicated file are identical and accurate at all times.

In practice, managing a replicated file to ensure data consistency in all copies requires using a concept known as „two-phase commit“ logic. Here, a program that changes a replicated file receives a confirmation of the completed change only after all active copies throughout the distributed database have been changed. If one of the copies becomes unavailable, it must first be „resynchronized“ with the other copies to restore absolute database integrity. One can imagine the need for such high integrity in an airline reservation system, which has a high rate of change and possible contention.

In many database environments, however, the need for such complete integrity is not as critical. Branch offices, for example, require frequent access to item inventories to obtain item descriptions and prices for customer billing. Copies of the item inventory dataset, which is normally revised only by the main office, can be located at each branch office. The rate of change to the item inventory dataset is relatively low; each location reads the dataset frequently but changes it infrequently, if at all.

Using the Entire Transaction Propagator (ETP), Adabas database systems with a high rate of read transactions but a low update rate can have the benefits of replicated files without the complex controls needed by systems with high transaction/contention rates. Using a „master/replicate“ system of control, ETP resynchronizes all replicate copies with a master copy at user-specified intervals.

Generally, most database requests are read requests. Even in systems where the requests are mostly STORE/DELETE/UPDATE (referred to generally in this text as update) operations rather than read requests, the update operations are often to only a small part of the overall distributed database. Frequently, the update requests come from a limited number of users and/or locations. Restructuring the data to „move it closer“, logically speaking, to its principal „owners“ and users can improve the overall system efficiency. Then, using ETP, each master file can be located where change activity is highest for that file, with replicate copies of the master file at other locations for local read-only use.

ETP also allows the replicate file copies to be „tailored“. If a particular location needs only a small portion of the overall data resource, ETP allows creation of a replicate copy containing only the needed records.

## How ETP Operates

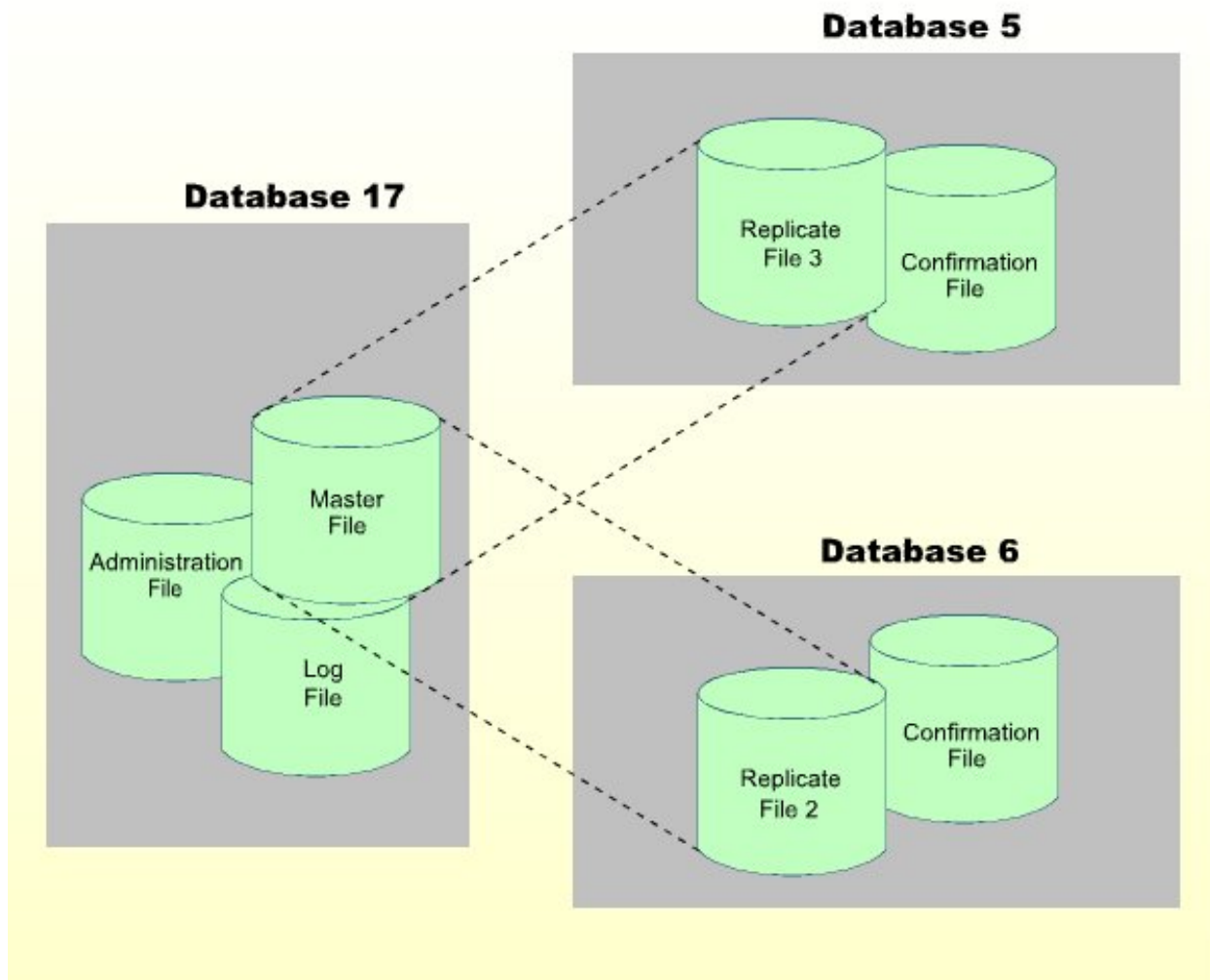
---

ETP provides the convenience of replicate files using a „master/replicate“ concept. Here, the more frequent read requests are satisfied by the nearest replicate copy, while the occasional update requests are applied only on the master file. In addition, changes to the master copy are logged. At user-defined intervals, the replicate copy is updated with the logged changes.

In this way, the requests for reading data are resolved locally, independent of the activity in other copies and with no network activity. Programs making update requests affect only the master file, and therefore need an Adabas end-of-transaction (ET) from only the master copy before continuing their operation.

Any updates made to the master file are logged in the related log file; the database operations used for logging are part of the same transaction as the original update operation; therefore, data integrity, including the log file contents, is automatically preserved by normal Adabas safeguards.

As for any Adabas file, incomplete or otherwise unsuccessful updates transactions are restored or removed („backed out“) by the normal Adabas restart/recovery facility.



A Basic ETP Replicate File with Support Files

## ETP File Structure

To manage the master file and replicate file copies, other „background“ files support the ETP environment as shown in the figure *A Basic ETP Replicate File with Support Files*, above.

Each master file has a *log file* for recording update transactions. All updates are recorded in the log file as soon as they are applied to the master file. The log file, which must be on the same database as the master file, can serve all master files on a database, is the source for update information when ETP resynchronizes replicate files with their master file.

Whenever replicate files are resynchronized, the replicate files are updated and the resynchronization is recorded in a *confirmation file*. As with the master file's log file, a confirmation file must

be located on the same database as the related replicate files, and can serve one or more replicate files on the database.

To control overall ETP operation an *administration file* contains the definition of all master/replicate file definitions, their relationships and physical locations. The administration file, which should be located on every nucleus having a master file, is the source of all information used by the ETP Replication Tasks to perform the actual master/replicate file resynchronization.

The log, confirmation and administration files can all be on one Adabas file when all master and replicate files are on the same physical database. However, the files are presented in this document as separate files to depict a more flexible ETP environment.

## Replication Task Control

---

The ETP Replication Tasks actually resynchronize the replicate files with their master files. Each time a replication task is started or restarted, it checks the log files for the master files in the administration file. If the log files contain uncompleted changes—that is, logged changes not yet applied to the replicate files, the task applies those changes to the corresponding replicate files.

The first resynchronization occurs when a task is first started. A task can be started in one of three ways:

- Manually by the ETP administrator;
- By a user program invoked after each ET on the master file;
- At regular time intervals.

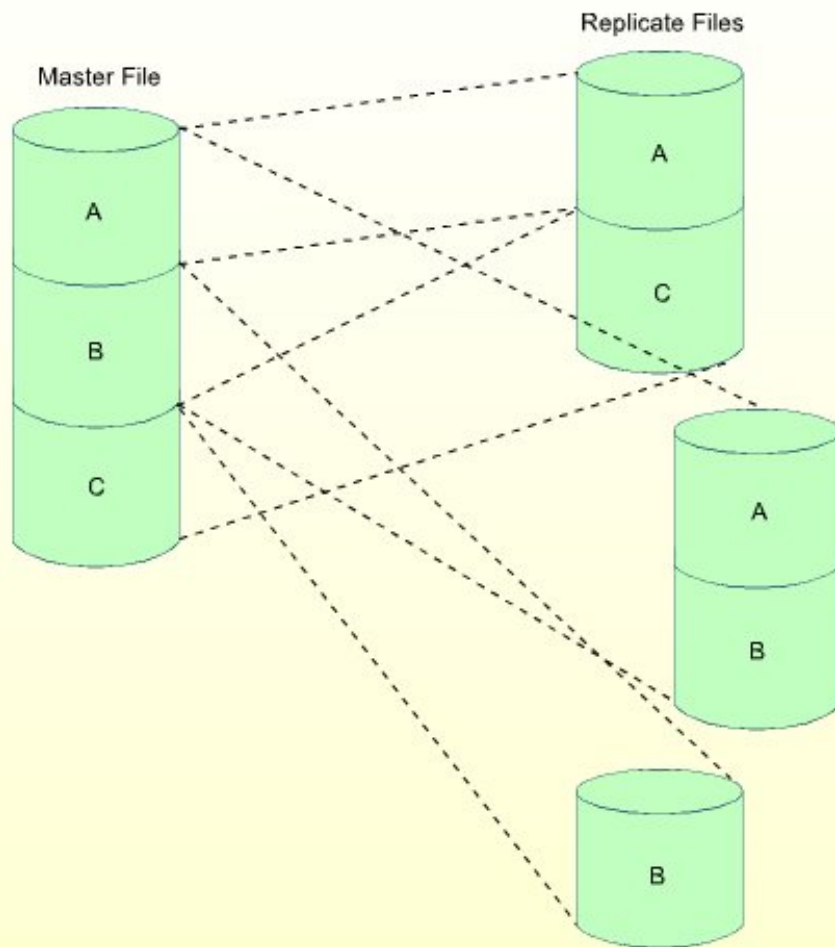
When or how often a task restarts depends on the values set at installation or by the ETP administrator. Tasks are stopped or restarted and/or placed under ET program or self-restart interval control using the ETP maintenance utility.

The maintenance utility, a menu-driven facility for setting up and managing ETP operation, allows you to:

- Define master and replicate files;
- Control Replication Task operation;
- Set up user/system profiles;

For further details, refer to the [ETP maintenance utility documentation](#).

### Replicate Files from Master File Record Subsets





## Tailoring and Tuning ETP

---

Although the possibility of inconsistency between the master and the replicate files is theoretically greater than with the two-phase commit concept, ETP offers „tailoring and tuning“ of the individual replicate copies, which can remove virtually any chance for differences to occur.

The „tailoring“ ability of ETP allows you to define each replicate file copy for the needs of a particular location. In other words, the replicate file can contain either all data in the master file, or only the data needed at a particular location, as shown in the figure [Replicate Files from Master File Record Subsets](#). This can reduce the need for „refreshing“ the replicate file, since only a portion of the complete master file is represented.

The „tuning“ ability in ETP allows you to control exactly when master/replicate synchronization occurs. You determine when resynchronization occurs by specifying when to start the ETP replication tasks. The choices for task start are:

- Manually at any time using the task control facilities of the maintenance utility;
- After each successful end-of-transaction (ET) on a master file, using your own program. You can make the ET control effective for specific master files (and their related replicate files);
- At regular self-restart intervals defined for all log files and the replicate databases to which the related master files are replicated. Multiple tasks can be run in parallel to:
  - Process different sets of log files;
  - Replicate to different replicate databases;
  - Have different self-restart intervals.

## Resynchronization Modes

---

The following examples show how ETP operates in the three resynchronization modes:

- [Manual Control](#)
- [ET Program Control](#)
- [Self-Restart Interval Control](#)

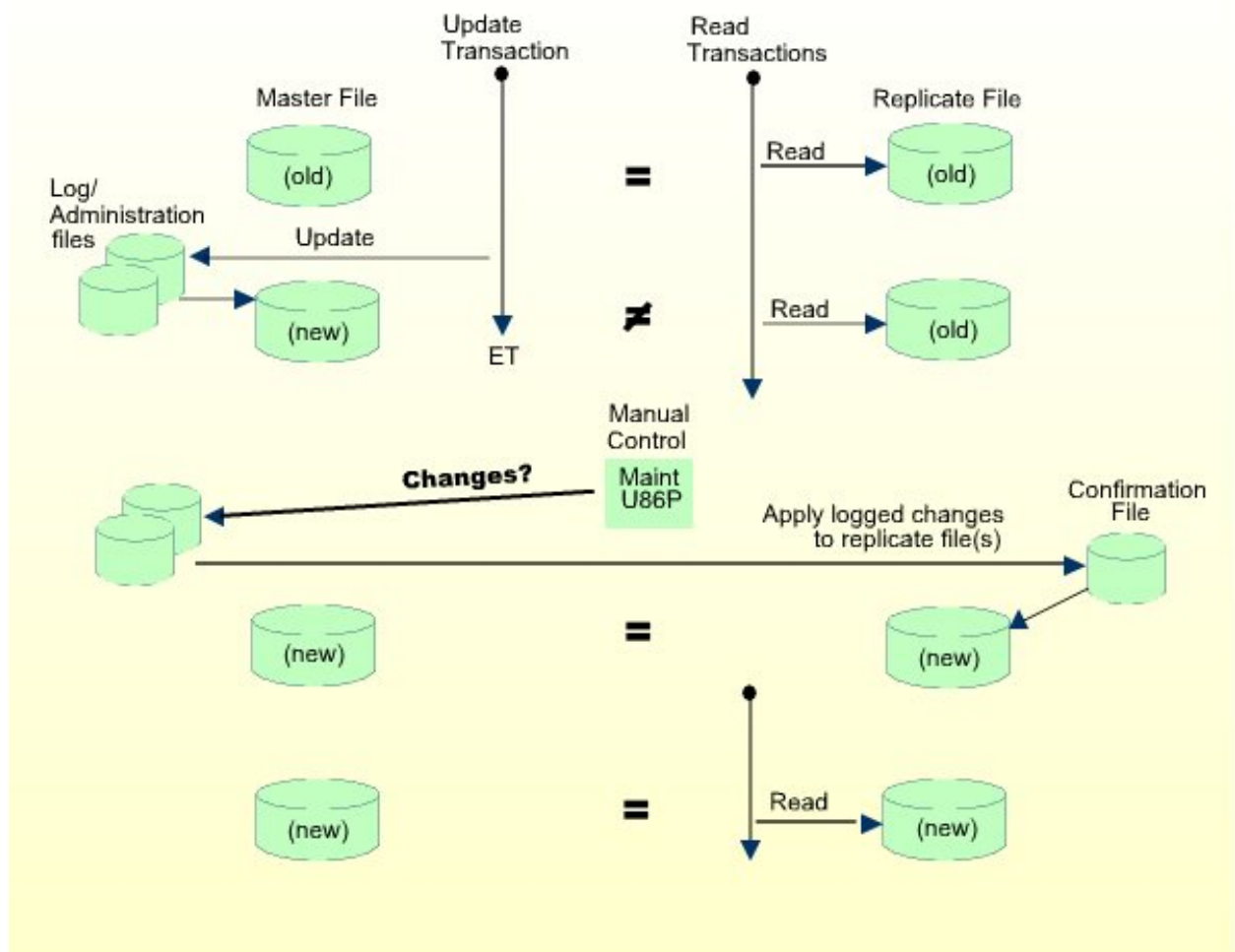
## Manual Control

The figure below shows the basic operating sequence for a simple ETP environment of one master and one replicate file. The first two read transactions access the „old“ replicate file. The single update transaction updates the „old“ master file to „new“ status.

After the change is actually applied to the master file, the program issues an end-of-transaction (ET) to denote that the update is complete.

By default, a task is executed only once to resynchronize master and replicate files. If the task is executed with a self-restart interval, all master and replicate files defined in the related administration file are resynchronized. Manual control is intended for resynchronizing of the ETP environment online.

### Master/Replicate File Synchronization (Manual Control)

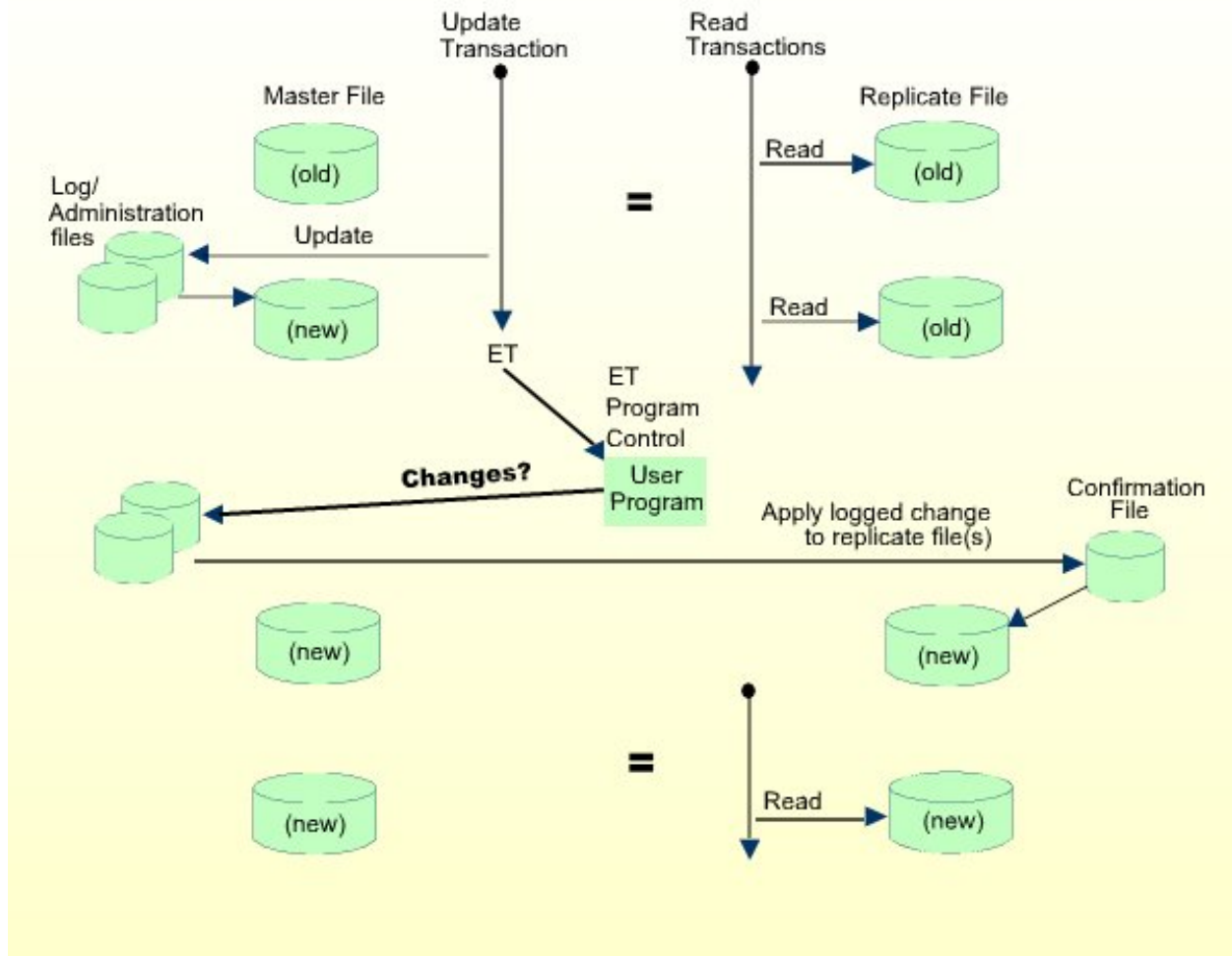


If all tasks have been stopped by means of the „Control Tasks“ direct command, task execution for the current administration file is disabled; that is, no tasks can be executed. After task execution has been re-enabled, the tasks must be started manually by the ETP administrator.

### ET Program Control

The figure below shows the ETP operation when the Adabas End-of-Transaction (ET) controls task operation.

**Master/Replicate File Synchronization (ET Program Control)**



The program must first be defined to and enabled by ETP as described under *Programming ETP*. The parameters passed to the program are described under *ET-Time User Program*.

Each master file that runs under ET program control must enable the program control. If a master file neither invokes ET program control nor operates under self-restart interval control, it and its replicate files can only be resynchronized under manual control.

If any of the master files which are updated in a transaction have enabled ET program control, the user-defined program is started after the ET command has been successfully executed.

## Self-Restart Interval Control

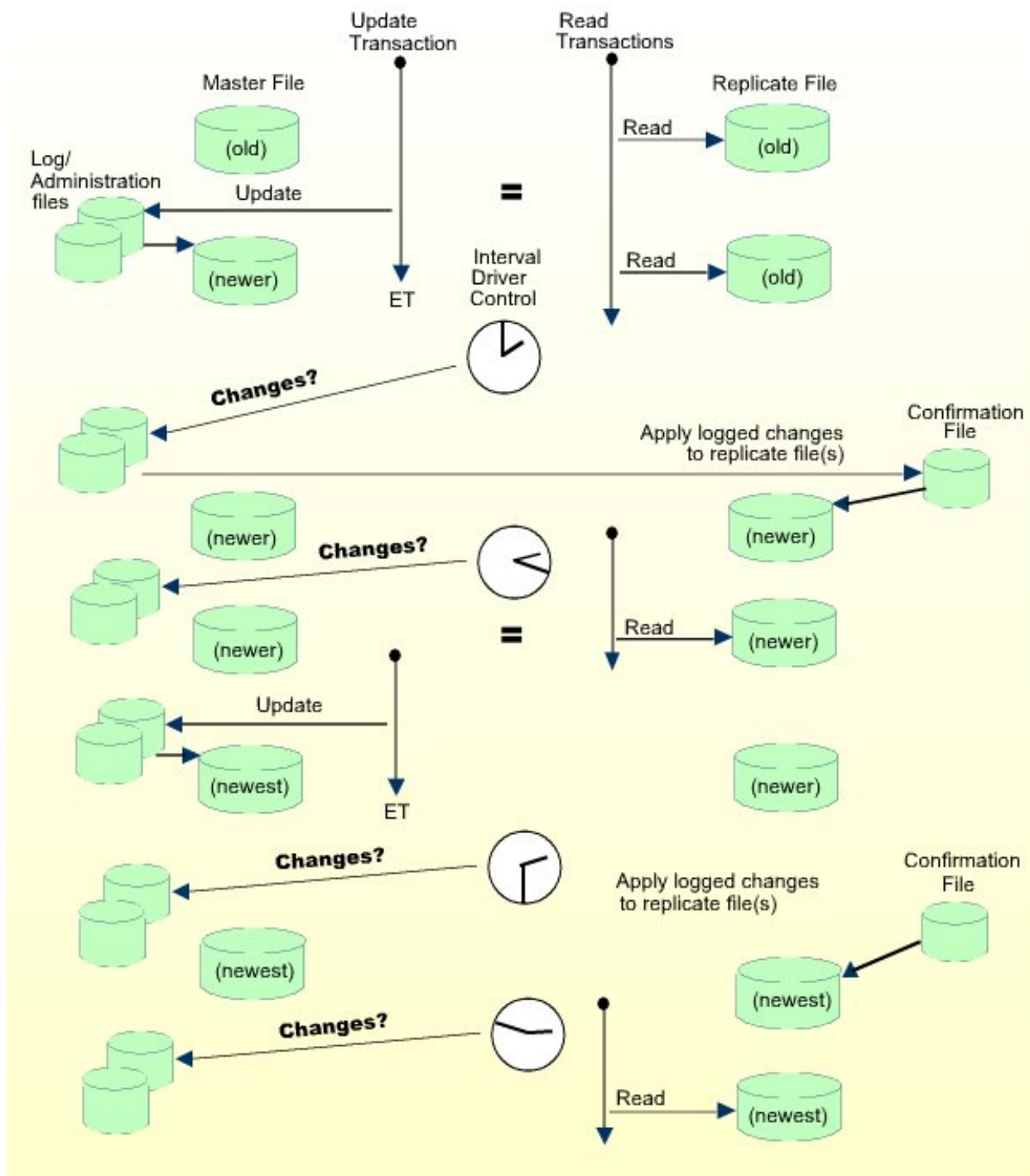
A fully automatic method of operation is the self-restart interval control. By defining and enabling an „elapsed time“ between restarts, you can force Replication Tasks to resynchronize master files with replicate files. The figure below shows an example of this type of operation.

To run ETP in self-restart mode, you must define a time interval and then enable self-restart operation. You can define a different self-restart interval for every replication task.



**Caution:** Self-restart resynchronization is intended for batch operation, and is not recommended for online operation; the terminal is locked during self-restart resynchronization. Self-restart resynchronization must not be used when running UTM, TIAM or IMS because a processor loop may result.

Master/Replicate File Synchronization (Interval Control)



# 4 ETP Installation

---

- Prerequisite Requirements and ETP Release Contents ..... 24
- General Information about ETP Installation ..... 25
- Additional Information about ETP Installation ..... 33

## 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.

- [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.

- 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  $3n-2$  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 f s f s
```

- where `f s f s` 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 conform 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 SRVvrs.LIB from Tape to Disk

This step is not necessary if you have already copied the library SRVvrs.LIB from another Software AG installation tape. For further information, refer to the element #READ-ME in this library. The library SRVvrs.LIB is stored on the tape as a sequential file named SRVvrs.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 SRVvrs . 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:

ETPnnn.SYSF

ETPvrs.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:

ETPvrs.FDTA

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

ETPvrs.SYSn

ETPvrs.SYSn: 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:

```
* $$ JOB JNM=LINK,CLASS=0,DISP=D
* $$ 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,ENTIRE`. It 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.



**Caution:** 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.



**Caution:** 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 `ETPvrs .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 `ETPvrs.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.



**Caution:** 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 `ETPvrs.SYS3` from the distribution tape. For confirmation files on OpenVMS, Windows or UNIX systems, use the confirmation file field definitions in file `ETPvrs.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 source 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.



**Caution:** 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.



**Caution:** 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.



**Caution:** 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.

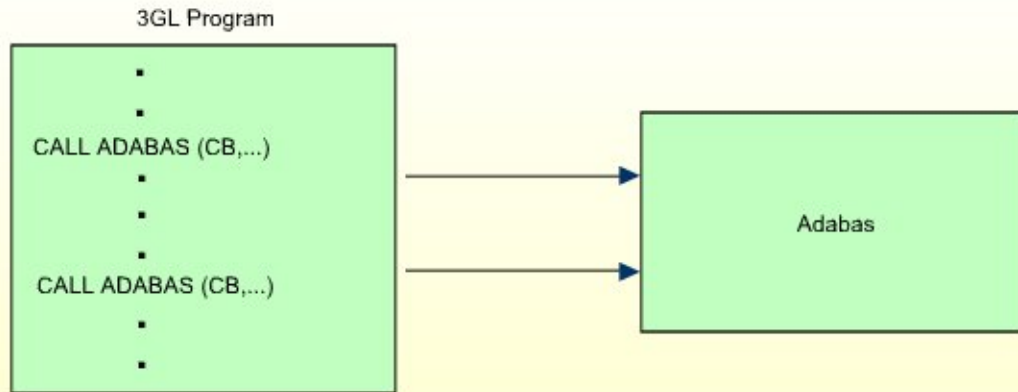
Let's assume that your 3GL program now contains calls to Adabas similar to the following:

```
      .  
      .  
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:

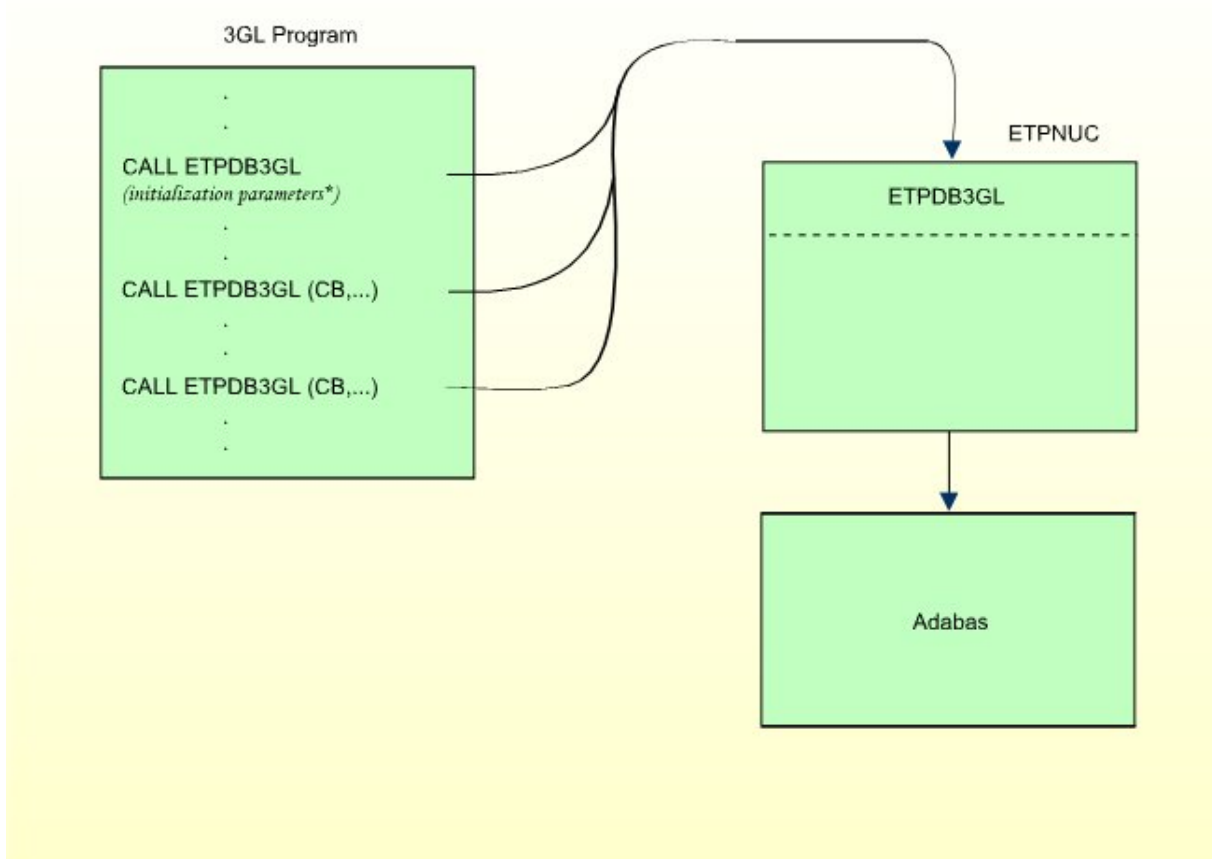
1. Change your 3GL programs to call the interface module ETPDB3GL instead of Adabas; that is, instead of `CALL ADABAS`, code `CALL ETPDB3GL`. Include the initialization calls and modify all other Adabas calls so that they look like the following:

```

      .
      .
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



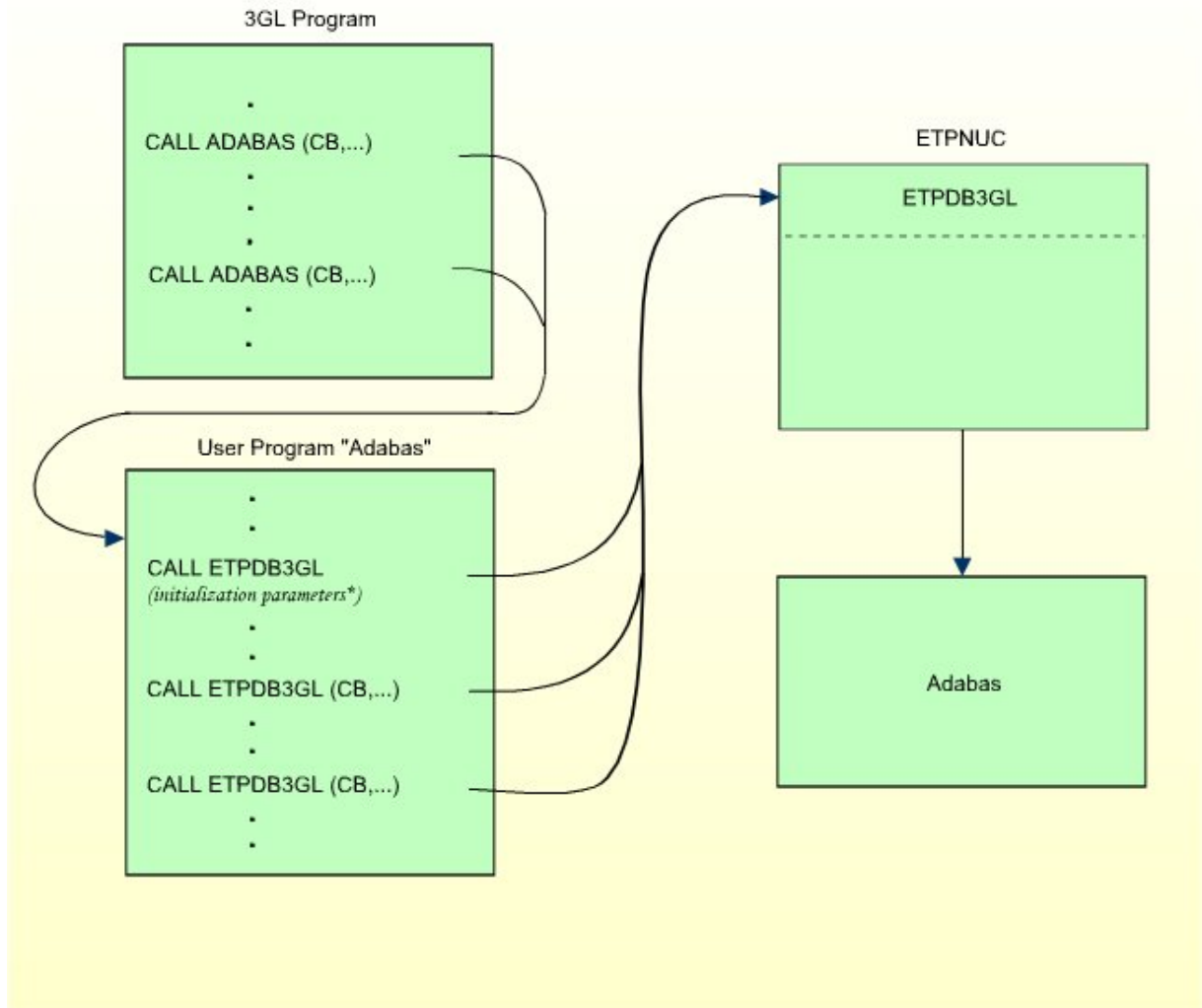
\* The initialization parameters are described in the sample program which is provided in dataset `ETPvrs.SRCE`.

- 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 `ETPvrs.SRCE`.

A sample program using this second alternative for z/OS batch mode is provided in dataset `ETPvrs.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`.



# 5 ETP Setup

---

- Getting Started ..... 46
- General Considerations when Defining Files ..... 46
- Performance and Maintenance Considerations ..... 48

This document gives hints on how to define the ETP environment for the maximum performance.

## Getting Started

---

To perform the first steps with ETP, you can use the demonstration files `EMPLOYEES` and `VEHICLES` that are supplied with every Natural installation, and the corresponding sample programs described in the *Natural Programming Guide*.

You can also use the application program that you plan on using with ETP by copying the necessary files for test purposes. For easier maintenance, Software AG recommends starting with small or empty files; this makes it easier to see which changes ETP has performed.

It is possible and recommended to initially install both master and replicate files on the same database. This eases the first steps, and avoids problems that might occur when data is transmitted over a network.

## General Considerations when Defining Files

---

You should ensure that user exits do not change either the database ID or the file number specified in the Adabas control block. If required, use the Natural macro `NTTF` or the Natural profile parameter `TF`. The following exits are a potential source of such changes:

- `NATUEX5` (Natural, called before an Adabas call)
- `NATUEX6` (Natural, called after an Adabas call)
- User exit B (`ADALNK`)
- User exit 1 (Adabas)

When accessing an Adabas database that requires the first command to be an `OPEN` command, a `NAT3009` error will occur during replication if Natural has not been set up so that an `OPEN` command is issued. When your system is set up to require an `OPEN` command, heed the hints in the following sections when defining replicate files.

## Specifying Replicate Files on OpenVMS, UNIX or Windows

When defining replicate files on diverse platforms and systems, you should be aware of the points described in the following topics.

### Generating an Automatic OPEN (OP) Command

If an Adabas database requires that an OPEN (OP) command to be issued before any other database call, the Natural system variable \*ETID must be a non-blank value; if \*ETID is blank, Natural does not issue the OP command. You can also use the dynamic Natural profile parameter ETID to specify a non-blank ETID.

If Natural Security is not installed, the default for \*ETID is the user ID. If Natural Security is installed, Software AG recommends letting Natural Security generate the \*ETID value automatically; RESTART=YES should also be specified for the library SYSETP. For more information, see the *Natural Security* documentation.

### Specifying the Record Buffer Value for the OPEN (OP) Command

You can use either the Natural profile parameter OPRB or the NTOPRB macro to specify a Record Buffer for the OPEN (OP) command issued to the replicate databases. See the *Natural Parameter Reference* documentation for parameter or macro details.

### Data Representation Inconsistencies

Alphanumeric and binary data is represented differently on mainframe and OpenVMS, UNIX and Windows systems. This can lead to unexpected results if:

- You use Natural REDEFINE statements in your application to redefine a field defined as alphanumeric in Natural and in the Adabas Field Definition Table (FDT), so that the alphanumeric data is overlaid with numeric or binary data-or vice versa;
- Binary superdescriptors are used that contain alphanumeric data;
- Alphanumeric superdescriptors are used that contain binary data (i.e., by means of a REDEFINE).

In the above cases, strange results can occur because:

- Binary data that is part of an alphanumeric field is converted from EBCDIC to ASCII, producing useless result data;
- Alphanumeric data that is part of a binary field is *not* converted from EBCDIC to ASCII;
- The sequence obtained by means of a READ LOGICAL statement may be different due to the fact that Windows and some UNIX machines have a byte-swap architecture especially affecting alphanumeric data that is contained in binary fields.

## Superdescriptor Conversion

Particularly for superdescriptors, Entire Net-Work for mainframe systems provide the superconverter feature that allows for an exact specification of the translation process for all parts of a superdescriptor. Refer to the appropriate Entire Net-work documents for a detailed description.

## Performance and Maintenance Considerations

---

ETP performance is affected by the definitions of master and replicate files. If you plan to replicate large amounts of transactions, you should specify the definitions carefully to avoid performance problems. The `CHECK DEFINITIONS` command of the **ETP maintenance utility** is useful for examining your definitions for potential performance problems. If a potential problem is reported, it will in many cases be a problem only if large numbers of transactions are to be replicated.

You should also take into account that not only the use of ETP itself, but also other resources are likely to be potential sources of data distribution problems; for example:

- Usage of the machine hosting the master file database or performing the ETP replication process may be short of resources (CPU time, etc.);
- The process performing the ETP replication task may run at low priority;
- The Adabas nucleus hosting the master and log files may run at low priority;
- The Net-work processes on either the master or the replicate site may be overloaded or run at low priority;
- The transmission rate may be low or the usage high on the network link connecting the master and replicate databases;
- The Adabas nucleus hosting the replicate file may run at low priority.

When replicating a large number of master file updates, you should examine all of the above possibilities to ensure that the updates are replicated as you intended. When performing only a small number of master file updates, the danger of performance-related problems is usually very small.

The following are hints and tips for setting up master and replicate files:

- **Avoid defining distribution keys for a master file unless you actually need them.** Distribution keys cause overhead for both logging and replication of master file updates. During replication, any defined distribution key ranges are kept in storage for performance reasons. If you have defined a number of large distribution key ranges for replicate files and those replicate files are on the same database for which the master file share a common log file (see the next point), then some key ranges have to be loaded from the database. To improve performance in this case, assign different log files to the corresponding master files.



- **For all master files with replicate files on common databases, use a common log file.**

For example, consider the following relationship of master and replicate files:

Master File DBID/File no.:	Replicate File DBID/File no.:
10/20	12/20
10/30	12/30
10/40	14/40
10/42	14/44

In this case, master files 20 and 30 could share a common log file and master files 40 and 42 could share a second log file. In fact, if the number of changes to all the master files are small, all four master files can share a single log file. Note, however, the exception when using distribution key ranges as described in the point above, and the last point on the following page.

- **Run multiple replication tasks from different log files or to different databases, in parallel.** Running parallel multiple tasks that perform replication from different log files and/or to replicate databases improves system throughput and is highly recommended if many updates are to be replicated or if the number of replicate databases is high. For a network link with a high transmission rate, running multiple tasks that replicate from different log files to a common replicate database may also improve the system throughput.
- **Short self-restart intervals are unproductive.** The checking necessary to determine if any new transactions in a log file are to be replicated requires a fixed number of database calls; therefore, using a self-restart interval that is too small results in unnecessary overhead. You can use the report written by ETP following every replication to choose an appropriate self-restart interval.
- **Use batch mode to define a large number of files.** When defining a large number of master and replicate files, Software AG recommends doing this in batch mode because changes to all definitions are more easily achieved by deleting the existing definitions and then adding the modified definitions again.
- **An error when replicating to one replicate file affects all replicate files served by that log file.** When an error occurs during replication of an update, replication is terminated immediately for all replicate files on that database that are served by the related log file. This includes transactions to replicate files from other master files that share the log file. File-specific errors (indicated by, for example, Adabas response code 113) should not occur under normal conditions; however, to ensure that replication continues for as many log files and replicate databases as possible, assign different log files to the master files.



# 6 Programming ETP

---

- ETP Programming Principles ..... 52
- ET-Time User Program ..... 55

Natural application programs written for Adabas can use the ETP master files without change. Therefore, it is assumed that you are familiar with Adabas programming techniques. If you have questions regarding Adabas programming, refer to the Adabas manuals listed on the [Overview](#) page of the Entire Transaction Propagator documentation.

This document describes the few considerations required when creating Natural programs for ETP, and describes the optional ETP user exits and driver control using a user-written ET program.



**Note:** ETP also provides an interface for 3GL (COBOL, PL/I, etc.) programs. See [Using ETP with 3GL Programs](#) (in the ETP Installation documentation) for more information. An additional product for 3GL programs running under CICS, ETP Interface for CICS (ETC), can also be ordered. For more information, see the [ETP CICS Interface](#) documentation.

## ETP Programming Principles

---

- [Limits and Restrictions](#)
- [Updating Master Files](#)
- [Updating Replicate Files](#)
- [Using Distribution Keys](#)
- [Updating the Administration File](#)
- [Processing of Asynchronous Task Messages](#)

### Limits and Restrictions

ETP is a distributed database environment that provides high reliability without the overhead needed for absolute data integrity. However, there are some limits and restrictions that you should consider when creating programs for the ETP master/replicate file environment.

ETP requires that update operations be performed to the master files only. In addition, read operations, where possible, should be addressed to the nearest replicate copy. Although the latter is not a requirement, reading the nearest replicate file is recommended to reduce network traffic.

ETP cannot handle the new control block format as available with Adabas Version 8; if such a control block is encountered, an error is returned to the application that issued the database call.

## Updating Master Files

All updates to the ETP environment must be performed *only* on master files. If an End Transaction (ET) call to a master file contains ET user data, the ET data are included in the resynchronization changes made to the replicate files.

### Natural Updates to Master Files

A Natural program can update all master files defined in a single ETP administration file. The reason is, only one administration file can be assigned using an `NTLFILE` or `LFILE` definition. Also, the master files updated by a single transaction must all be in the same database. Note, however, that master files under control of the same administration file can be in different databases.

To update files on different databases in a *single* transaction, at most one of the databases affected may be defined as an ETP database, that is, a database that contains master files.

### Multi-Processor Updates

When updating master files which use the same log file from more than one processor, the processor clocks must be synchronized to ensure that the transaction time stamps in the log file are not inconsistent. Software AG therefore recommends that you make all updates to master files using the same log file from a single processor only.

## Updating Replicate Files

Programs should read replicate files only. Replicate files may not be changed by application programs. An Adabas Security password can be defined for a replicate file to protect it from accidental updates.

## Using Distribution Keys

A distribution key is the control mechanism used by ETP to manage replicate files that contain only a subset of the master file records. You create a distribution key by defining one field in the master file's record as the key field. When you later define a replicate file having only a subset of the master file records, you can specify a value, a range of values, or a combination of values and/or value ranges for the key field that determine which records are in the replicate file.

A master/replicate file set can have only one distribution key field. The field can contain meaningful data (part number, employee number, family/last name, etc.) or it can be an „invisible“ field used only to control the ETP files. In addition, a distribution key can be any Adabas field except a periodic (PE) group or a field within a PE group, a multiple-value field, or a phonetic descriptor.

## Checking the Key with User Exit 2

The ETP optional user exit is `WADUSER2` (see *ETP Installation*). This user exit must be located in library `SYSETP`, or in a `steplib` when a replication task is executed. See the `SYSETP` library, which contains an example of the user exit.

## Storing a Record with No Key

If you store a record in a master file that has a distribution key, but the Natural view does not define the distribution key, ETP sets the key for the new field to a default value. Depending on the data format of the field, the default value is blank for a format A field, binary zeros for a format B field, or zero for all numeric types.

## Updating the Distribution Key

You cannot simply `UPDATE` a field in the master file that is defined as a distribution key. To perform an update on the field, the program must perform an explicit `DELETE/STORE` sequence. This is due to the fact that the information not included in the view used for the `UPDATE` process would otherwise be lost.

## Updating the Administration File

If a master file definition in the administration file is changed while logging is being done on the database of the master file, the changed definition is not used immediately. The new definition is first used after either a transaction has been logged for another database, or after Natural has been restarted - whichever occurs first.



**Caution:** Therefore, you should avoid master file definition changes while a Natural application using ETP logging facilities is active. For the same reasons, you should also avoid changing master or replicate file definitions while a task is executing.

## Processing of Asynchronous Task Messages

The subprogram `WADUSER3` is used to display all messages issued by the replication and clean-up tasks. `WADUSER3` can be modified to filter any resulting messages and, if desired, send them directly to the operator console. (For more information about the `WADUSER3` subprogram, which is delivered in source form with ETP, see *ETP Installation*.)

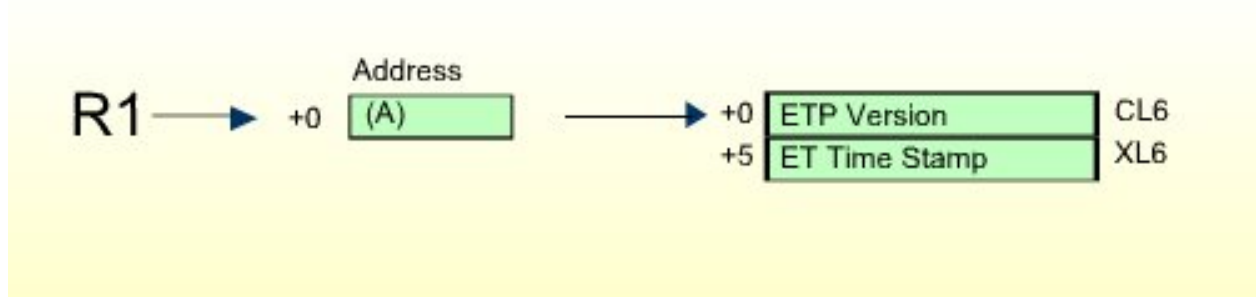
## ET-Time User Program

---

After each update to an ETP master file, you can invoke your own program to perform such functions as starting a replication task to resynchronize the replicate file(s) with their master file. When called, the program is passed a pointer to an address of a parameter list in General Register 1, as shown in the figure below.


ETP uses normal OS conventions to pass the parameter list.

**Parameters Passed to the ET User Program**



The user program can be for any purpose; however, the ET program is intended for starting the replication task(s) to resynchronize replicate files with the master file. The program that receives control after an ET is typically a CICS or Com-plete transaction which starts a Natural nucleus. That nucleus must be provided with appropriate parameters to log on to SYSETP (the ETP maintenance utility) and to start a replication task.

If any of the master files which are updated in a transaction have enabled ET program control, the user-defined program is started after the ET command has been successfully executed.

 **Note:** The ET Program Control mode for resynchronizing master and replicate files described here is presently limited to only those systems using the CICS or Com-plete TP monitors. The controlling program must be a transaction which is defined to the TP system on which it executes. For CICS, the messages issued by the replication task must be routed to the operator console by means of the subprogram WADUSER3. (For more information about the WADUSER3 subprogram, which is delivered in source form with ETP, see *ETP Installation*.)



# 7 ETP Operations and Administration - Overview

---

Controlling the Entire Transaction Propagator (ETP) environment comprises three general tasks:

1. Preparing for and installing ETP and the related data/control files;
2. Managing and „tuning“the replicated files, using the menu-based maintenance utility;
3. Determining and correcting ETP error causes, recovering from errors, and restarting operation.

This document describes the tasks of managing and tuning with the ETP menu-driven maintenance utility, and how to use the maintenance utility to set up and control ETP as well as how to analyze and recover from errors.

The following topics are covered:

- [Overview of General Tasks](#)
- [ETP Maintenance Utility](#)
- [Master File Task Screens](#)
- [Replicate File Task Screens](#)
- [Replicating Logged Transactions](#)
- [Controlling Execution of Asynchronous Tasks](#)
- [Deleting Successfully Replicated Transactions](#)
- [Displaying X-Ref Transaction State](#)
- [Using Special Functions](#)
- [Executing a Natural Command](#)

- [ETP Restart and Recovery](#)
- [Database Errors During Logging of Master File Transactions](#)
- [Correcting Unlogged Transactions](#)
- [Reporting ETP Errors](#)

### Related Documents

- For an overview of related documents, see the list on the ETP [Overview](#) page.
- For information on how to operate the Entire Transaction Propagator under the teleprocessing system CICS, refer to [ETC Installation and Operation](#).

# 8 Overview of General Tasks

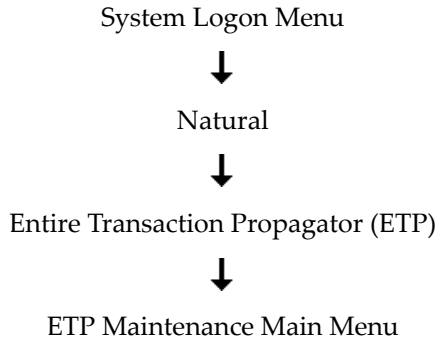
---

- ETP Maintenance Main Menu ..... 60

Controlling the Entire Transaction Propagator (ETP) environment comprises three general tasks:

- Preparing for and installing ETP and the related data/control files;
- Managing and „tuning“ the replicated files, using the menu-based maintenance utility;
- Determining and connecting ETP error causes, recovering from errors, and restarting operation.

The following tables give an overview of the call structure and the functions available:



## ETP Maintenance Main Menu

---

Cmd	Function invoked
CL	Clean up logfile
CO	Control asynchronous task execution
MM	Maintain master file definitions
MR	Maintain replicate file definitions
N	Execute Natural command/program
R	Replicate transactions
S	Special functions
X	Cross-reference transactions

# 9 ETP Maintenance Utility

---

▪ Starting the ETP Maintenance Utility .....	62
▪ Selecting and Performing Menu Tasks .....	63
▪ Entering and Displaying Information .....	64
▪ Priority of Command Input .....	66
▪ Entering Direct Commands .....	66
▪ Dialogue Functions .....	67
▪ Errors and Warnings .....	69
▪ Selecting Main Menu Tasks .....	70

The ETP Maintenance Utility provides you, the ETP Administrator, with screens and menus to:

- Add, display, change and delete master and replicate file definitions;
- Display and change a replicate or related confirmation file;
- Display and delete entries in a master file's log file;
- Stop/start replication tasks;
- Change system and user profiles.

## Starting the ETP Maintenance Utility

---

Utility functions can be performed on master or replicate files on all ETP nodes that are defined in the administration file. The link to the administration file is created when you install the maintenance utility; see [ETP Installation](#). You can also maintain multiple administration files (which can be on different databases) from a single node with the maintenance utility.

There is more than one way you can start the maintenance utility. The choice of ways depends on whether or not Natural Security is installed:

Without Natural Security, one of the following:

- Enter the command `SYSETP` on the command line;
- Log on to the library `SYSETP`, and then enter the command `MENU` on the command line;

With Natural Security and `MENU` defined as startup command:

- enter the command `SYSETP` on the command line;
- Log on to library `SYSETP`.

With Natural Security and `MENU` *not* defined as startup command:

- Enter the command `SYSETP` on the command line;
- Log on to the library `SYSETP`, and then enter the command `MENU` on the command line;

ETP then displays the ETP maintenance utility's main menu. Using its screens and menus, you can display and change ETP control information and perform maintenance tasks on the ETP file structure.

## The ETP Maintenance Utility - Main Menu

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                               Menu

MM  Maintain master   file definitions
MR  Maintain replicate file definitions
R   Replicate transactions
CD  Control execution of asynchronous tasks
CL  Clean up Log file
X   X-Ref transactions
S   Special functions
N   Execute Natural system command or program

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last          Flip          Tech          Canc

```

## Selecting and Performing Menu Tasks

The menus provide a choice of tasks that can be performed. Such tasks as those listed on the **main menu** can be performed by doing one of the following, and then pressing `Enter`:

- Placing the cursor on the line of the task in the menu;
- Overtyping the code in front of the task description - MM, for example - with any other letter;
- Entering the code in front of the task description on the command line at the bottom of the menu.

Selecting a task with the cursor or entering a task code is possible only when the related menu is displayed. Using a direct command, however, you can select a task to perform regardless of which menu is currently displayed.

The tasks or screens you can select on the **main menu** are described later in this section.

## Entering and Displaying Information

Screens that allow you to enter data designate the input fields with underscores (\_\_\_\_). The command line of each screen is not underscored, although it also is an input field. Use the Tab key to move from input field to input field. When entering input in a field, the data is aligned according to the data type; numeric data is right-aligned, and alphanumeric data is left-aligned.

### Example of a Displayed List:

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Display replicate file definition

+-----Replicate file definitions-----+
!                                     !
!  -- Replicate --  Conf  ----- Master ----- !
!    DBID      FNR   FNR    DBID      FNR  Key   Format !
!    _____*_____*_____ !
Replicat !  _   10      155   152    10      154  AC    B4    !
!  _   10      156   152    10      135  AA    A10   !
!  _   10      157   152    10      136  AB    A80   !
Printer  !  _   10      158   152    10      138  AG    N4    !
!  _   10      159   162    10      139                !
!  _                                     !
!  _                                     !
!  _                                     !
!  _                                     !
!  _                                     !
!  _                                     !
!  _                                     !
!  _                                     !
!  _                                     !
+-----+

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10---PF11---PF12---
      Help  Menu  Last          Flip          Tech          Canc

```

### Selecting using Wild Card Entries

Screens with fields allowing wild card entries display the results of wild card searches in a window, as shown in *Example of a Displayed List*, above. Fields that allow wild card entries have an „\*“ as the last description character. See the following section, *Entering Direct Commands* for more information about wild card entries.

Those fields with an asterisk (\*) as the last character of the description allow you to enter wild card characters. For example:

```
DISPLAY MASTER * *
```



- displays a window from which you can select one or more master files to in turn display their definitions, and:

```
MOD REPL * 20>
```

- displays a window that enables you to select a single replicate file for modification from those replicate files on all ETP databases with file numbers 20 and above. Wild card entries can be combined with the PROCESS dialogue function (see [Dialog Function Description](#) below) to perform a task on a range of items (files or log file entries).

If you make a wild card selection that results in a window that allows you to select one or more items, select the item or items (see [Example of a Displayed List](#)) you want in either of the following ways:

- place the cursor on the corresponding line;
- enter any non-blank character in the input field ahead of the item or items.

If you make more than one selection on a screen that allows only one single selection, an error message occurs. When only one selection is allowed, a choice made with a non-blank character has priority over a choice made by placing the cursor.

When a window appears that permits more selections than can be held in the window itself, „+“ and/or „-“ appear to show the paging possibilities for displaying the additional items. Simply place the cursor on the appropriate symbol in the bottom left corner of the window and press Enter to display the additional items. This is the same as entering „+“ or „-“ dialogue functions (see [Dialog Function Description](#) below).

## Specifying Ranges

When selecting items, you can enter an asterisk (\*) in place of a value to select all related items. To select an „equal and greater“ or „equal and less than“ range of items, enter one of the following:

```
n<    (less than and equal to nn)
n>    (equal to and greater than nn)
```

If more than one input field defines the selection for a task, either or both of these selection criteria can be used. This enables the selection of, for example, all files with numbers 100 and above in databases 10 or below.

## Priority of Command Input

The priority of menu/screen input and commands is as follows:

1. Command line input or PF key (if both entered, an error occurs);
2. Changing/entering a non-blank value in a menu selection input field;
3. Placing the cursor in a menu selection input field.

## Entering Direct Commands

As the example screen shows, the maintenance utility allows you to enter a command on the command line at the bottom of a screen. The so-called „direct commands“ can be entered either fully (ADD MASTER, for example) or abbreviated (A MA for adding a new master file).

Most direct commands also allow you to specify a database ID (DBID) and/or physical file number; however, this is not required. If you enter no DBID or file number (or if neither is allowed), the function screen appears, just as when you select the function from a higher-level menu. If you enter the DBID and file number, the function screen appears, either with the value or values entered in the appropriate fields, or overlaid with the window showing the DBID/file number selection you made as if you had made the DBID and file number selection in the function screen.

Direct commands cannot be entered in the command line if a window is currently displayed.

The following table shows the ETP direct commands. Minimum command abbreviations are shown in *bold* type:

Direct command:	Parameters allowed/required:		Wildcards allowed?
	DBID	FNR	
ADD MASTER	x	x	No
ADD REPLICATE	x	x	No
CHANGE ADMINISTRATION	x	x	No
CHECK DEFINITIONS			
CLEANUP LOGFILE			
CONTROL TASKS			
DELETE MASTER	x	x	Yes
DELETE REPLICATE	x	x	Yes
DELETE TRANSACTIONS	x	x	Yes
DISPLAY ERROR	x	x	Yes

DISPLAY MASTER	x	x	Yes
DISPLAY REPLICATE	x	x	Yes
DISPLAY TRANSACTIONS	x	x	Yes
MAINTAIN MASTER			
MAINTAIN REPLICATE			
MODIFY CONFIRMATION	x	x	Yes
MODIFY MASTER	x	x	Yes
MODIFY REPLICATE	x	x	Yes
MODIFY SYSPROF			
MODIFY USERPROF			
NATURAL	<i>(command string)</i>		
REPLICATE TRANSACTIONS			
RESET ERROR	x	x	Yes
RESET IN-USE	x	x	Yes
SPECIAL			
XREF TRANSACTIONS			Yes

## Dialogue Functions

Dialogue functions are the screen control and general operation commands for operating the maintenance utility. The dialogue functions, which can be either entered on the screen's command line as direct commands or by pressing the appropriate PF $n$  key, perform such tasks as:

- Returning to the main menu from a specific task menu (MENU);
- Saving the entries and status of the current screen (SAVE);
- Performing a maintenance utility function on a range of files (PROCESS);
- Exiting a task screen and returning to the next higher selection menu (EXIT).

To perform a dialogue function, press the appropriately labelled function key (PF $n$ ), which follow the common conventions for the general dialogue functions (PF1 for calling online help, PF3 for EXIT, and so on); otherwise, enter the dialogue function on the command line. Note, however, dialogue functions are most likely not valid for a particular screen if they are not already assigned to a function key. A Modify screen with confirmation window may look as shown below.

The dialogue functions and, where appropriate, their related function (PF $n$ ) keys are as follows:

## A Modify Screen with Confirmation Window

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Modify master file definition

Master file ..*   DBID      FNR      Verify FDT
Log   file ...           10      135      Y (Y/N)
                               152      N (Y/N)
                    +-----Exit-----+
                    !                       !
ADABAS field name of di ! _ save and exit      !
NATURAL data format of ! _ exit without save    !
                    ! _ resume current function !
Call user exit before re!                       !
Start program after ET .+-----+

Command ==> exit
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last          Flip          Tech          Canc

```

### Dialogue Function Description

BACKWARD (PF7)	Displays the previous screen of data. Entering „-“ is the same as entering BACKWARD.
BOTTOM (PF19)	Displays the last screen of data. Entering „+“ is the same as entering BOTTOM.
CANCEL (PF12)	Leaves the current screen and returns to the higher-level menu. If entered in the ETP main menu, exit from ETP. If the current screen is a MODIFY or DELETE task, a window (Figure 11) appears requesting confirmation of the MODIFY/DELETE operation unless the confirmation window is disabled by the user's profile; otherwise, the operation is <i>not</i> performed.
EXECUTE	Perform the current screen task, then leave the current screen and return to the higher-level menu. Unless disabled by the user's profile, a window appears requesting confirmation of the EXECUTE operation; otherwise, the operation is performed.
EXIT (PF3)	Leaves the current screen and returns to the higher-level menu. If entered in the ETP main menu, exit from ETP. If the current screen is a MODIFY or DELETE task, a window appears requesting confirmation of the MODIFY/DELETE operation unless the confirmation window is disabled by the user's profile; otherwise, the operation is performed. Entering „.” is the same as entering EXIT.
FLIP (PF6)	Changes the function key display at the bottom of the screen between PF1-PF12 to PF13-PF24.
FORWARD (PF8)	Displays the next screen of data. Entering „+“ is the same as entering FORWARD.

LAST (PF4)	Redisplays the direct command executed last in the command line, where it can be changed and/or re-executed. Entering LAST * on the command line shows the last 10 direct commands that were executed, in a window. Place the cursor on the corresponding line, or enter any non-blank character in the input field ahead of the command and press Enter to enter that command in the command line for changing/ re-executing.
MENU (PF2)	Leaves the current screen and returns to the main menu. If the current screen is a MODIFY or DELETE task, a window (Figure 11) appears requesting confirmation of the MODIFY/DELETE operation unless the confirmation window is disabled by the user's profile; otherwise, the operation is performed.
PROCESS	This command-line-only function performs the currently selected screen's function on all items that meet the criteria specified on the current window. PROCESS can be entered only in windows appearing in screens that require the DBID and/or FNR parameters, and is intended for use with „wild card“ parameter specifications.
SAVE	Saves the status of entries made in the current menu. A window appears requesting confirmation of the operation. If the user's profile disables the function, the confirmation window does not appear and the operation is performed. SAVE is intended for multiple updates; the current screen remains displayed.
TECH (PF9)	Performs the function of the Natural TECH system command.
TOP (PF20)	Displays the first screen of data. Entering „--“ is the same as entering TOP.
*	Displays all direct commands allowed for the current screen. The minimum abbreviations for the direct commands are displayed in capital (upper case) letters.
+ (PF8)	Performs the FORWARD screen scrolling function.
++ (PF19)	Performs the BOTTOM screen scrolling function.
-(PF7)	Performs the BACKWARD screen scrolling function.
-- (PF20)	Performs the TOP screen scrolling function.
.	Performs the EXIT function.



**Note:** The dialogue functions FLIP, LAST, LAST \*, and \* are processed before any other command input.

## Errors and Warnings

If you enter an invalid command in the command line or invalid data in an input field, the ETP utility displays an error message in the message line at the bottom of the screen.

If you try to execute a task that might cause inconsistent data or if the change is not allowed, a window appears with a warning message.

If a severe processing error occurs, the utility displays a window containing the cause and location of the error. ETP attempts to recover from the error automatically by retrying the failing task. If the same error occurs three times within a short time interval, ETP issues a STOP for the application.

If a severe and persistent error occurs that might be inherent to ETP, proceed as described under [Reporting ETP Errors](#).

## Selecting Main Menu Tasks

---

To select one of the tasks listed on the ETP **main menu**, place the cursor next to the desired entry and press Enter.

When you select either the Maintain master or replicate file definitions option, a more detailed menu of tasks appears, as shown in the example below. The tasks you can select on these screens are described under the headings [Master File Task Screens](#) and [Replicate File Task Screens](#).

### The Master File Task Selection Menu

```
23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                   Master File Definition Maintenance

                   AD  Add      master file definition
                   MO  Modify   master file definition
                   DI  Display  master file definition
                   DE  Delete   master file definition

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc
```

# 10 Master File Task Screens

---

▪ Master File Task Selection Menu .....	72
▪ Add Master File Definition .....	73
▪ Modify Master File Definition .....	74
▪ Display Master File Definition .....	76
▪ Delete Master File Definition .....	77

## Master File Task Selection Menu

---

By either entering the MM direct command or selecting the MM option on the ETP Maintenance Utility **Main Menu**, the Master File Definition Maintenance screen appears.

### The Master File Task Selection Menu

```
23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Master File Definition Maintenance

AD  Add      master file definition
MO  Modify   master file definition
DI  Display  master file definition
DE  Delete   master file definition

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc
```

The master file tasks you select from this menu cause the screens described in the following section to appear.



## Add Master File Definition

By either entering the `Add MASTER` direct command or selecting the AD option on the **Master File Definition Maintenance** menu, the following screen appears.

### The Add Master File Definition Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                        Add master file definition

Master file ..      DBID      FNR      Verify FDT
Log   file ..      _____  _____  Y (Y/N)
                        _____  _____  Y (Y/N)

ADABAS field name of distribution key ..... ___
NATURAL data format of distribution key ... ____

Call user exit before replication ..... N (Y/N)
Start program after ET ..... N (Y/N)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last Save Flip                      Tech          Canc
    
```

To define an ETP master file, enter the information in the fields, as described below.

### Screen Field Description

Master File DBID/FNR	Enter the database ID and file number of the Adabas file that becomes a master file. Specify Y (yes) under „Verify FDT“ to verify the file's Field Definition Table (FDT); otherwise, enter N (no). You must make all entries; there are no defaults.
Log File FNR	Enter the file number of the Adabas file that is or will be the ETP log file. The log file must be on the same database as the master file. A single log file can be used for more than one master file. Specify Y (yes) under „Verify FDT“ to verify the log file's Field Definition Table (FDT); otherwise, enter N (no). To ensure data consistency, the log file must be empty; otherwise, adding a master file is not allowed.

Adabas Field Name of Distribution Key	When any of the replicate files for this master file contain only a subset of master file records, one field must contain values or ranges of values that uniquely identify each replicate file's subset, and therefore the replicate file itself. Specify the two-letter Adabas field name of that identifying field here. The distribution key field cannot be a multiple-value (MU), a periodic group (PE) or field within a PE group, or a phonetic descriptor field; otherwise, an error occurs either during FDT verification, or when writing to the log file during operation.
Natural Data Format of Distribution Key	When you specify a field name for the Distribution Key field above, you must specify the field's Natural data type here (for example, A15 for a 15-byte alphanumeric field). Specifying the number of digits after the decimal point is not possible for numeric data types.
Call User Exit?	Specify Y (yes) if you want the <b>WADUSER2</b> subprogram called <i>after</i> ETP checks the current transaction record's distribution key field, if any. ETP checks any distribution key to determine if a change to the master file must also be made to a specific replicate file. The default is N (no).
Start Program after ET?	Specify Y (yes) if you want to execute the program defined in the <b>Modify System Profile</b> task screen. The default is N (no).

## Modify Master File Definition

By either entering the **M**Odify **M**AsTer direct command or selecting the **M0** option on the **Master File Definition Maintenance** menu, the Modify Master File Definition screen appears.

### The Modify Master File Definition Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Modify master file definition

Master file ..*  DBID      FNR      Verify FDT
Log   file ...  _____  _____  Y (Y/N)
                    _____  Y (Y/N)

ADABAS field name of distribution key ..... _
NATURAL data format of distribution key ... _____

Call user exit before replication? ..... _ (Y/N)
Start program after ET ..... _ (Y/N)
    
```

```

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last  Save  Flip                Tech                Canc

```

To change an existing master file's definition, enter the information in the fields as described below.

### Screen Field Description

When the „Modify Master File Definition“ screen is first displayed, you must either enter (or have already selected) a master file; otherwise, you cannot make entries in the rest of the fields. Once you have selected a master file and pressed `Enter`, you cannot change your selection unless you either exit and then re-enter the screen, or enter the `SAVE` command.

Master File DBID/FNR	Enter the database ID and file number of the existing master file. If you enter a wild card in either the database ID or file number field, a window appears from which you can select a single master file definition for modification.
Log File FNR	Enter a new log file file number here. The log file must be on the same database as the master file. Note that if the current log file is not empty, a new log file cannot be defined.
Adabas Field Name of Distribution Key	When any of the replicate files for this master file contain only a subset of master file records, one field must contain values or ranges of values that uniquely identify each replicate file's subset, and therefore the replicate file itself. If the distribution key field is being changed, specify the two-letter Adabas field name of that field here.  The distribution key field cannot be a multiple-value (MU), a periodic group (PE) or field within a PE group, or a phonetic descriptor field; otherwise, an error occurs either during FDT verification, or when writing to the log file during operation. Note that if any of the replicate files are still using the distribution key, it cannot be changed.
Natural Data Format of Distribution Key	When you specify a new field name for the Distribution Key field, you must specify the field's Natural data type here (for example, A15 for an alphanumeric field of 15 bytes). Specifying the number of digits after the decimal point is not possible for numeric data types.
Call User Exit?	Specify Y (yes) if you want the <code>WADUSER2</code> subprogram called <i>after</i> ETP checks the current transaction record's distribution key field, if any. ETP checks any distribution key to determine if a change to the master file must also be made to a specific replicate file. The default is N (no).
Start Program after ET?	Specify Y (yes) if you want to execute the program defined in the <a href="#">Modify System Profile</a> task screen. The default is N (no).

## Display Master File Definition

---

By either entering the `DI`splay `MA`ster direct command or selecting the `DI` option on the **Master File Definition Maintenance** menu, the Display Master File Definition screen appears.

### The Display Master File Definition Screen

```
23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Display master file definition

Master file ..*   DBID   FNR
                 _____

Printer name .. _____ (' ' = terminal)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc
```

To display a master file's definition, enter the information in the fields as described below. To make changes to a master file's definition, enter the `MO`dify `MA`ster direct command on the command line at the bottom of the screen.

## Screen Field Description

Master File DBID/FNR	Enter the database ID and file number of the master file you want displayed. If you enter a wild card in either the database ID or file number field, a window appears from which you can select one or more master file definitions for display.
Printer Name	To print out the selected master file definitions, enter a valid printer ID here. Leave this field empty to display the master file definition on your terminal.

## Delete Master File Definition

By either entering the `DElete MAsTer` direct command or selecting the `DE` option on the [Master File Definition Maintenance](#) menu, the Delete Master File Definition screen appears.

### The Delete Master File Definition Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Delete master file definition

Delete master file ..*  _____  _____
                        DBID      FNR

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc

```

Before you delete a master file definition, you must delete all related replicate files; the related log file must be empty.



**Caution:** Deleting a master file only removes the master file definition from the ETP administration file. The file and its contents still exist as a normal Adabas file in its original database and with its original file number.

To delete a master file's definition, enter the information in the fields as described below. To make changes to a master file's definition, enter the `MODIFY MASTER` direct command on the command line at the bottom of the current screen.

### Screen Field Description

Delete Master File DBID/FNR	Enter the database ID and file number of the master file you want to delete. If you enter a wild card in either the database ID or file number field, a window appears from which you can select one or more master file definitions for deletion.
--------------------------------	--

If you have requested the confirmation window using the [Modify User Profile](#) function, the confirmation window appears, asking you to confirm the deletion. If either of the following conditions exist:

- The log file has entries for the master file;
- A master file has one or more replicate files still defined.

- then a warning occurs, and the deletion request will be rejected.

# 11 Replicate File Task Screens

---

▪ Replicate File Task Selection Menu .....	80
▪ Add Replicate File Definition .....	81
▪ Modify Replicate File Definition .....	83
▪ Display Replicate File Definition .....	85
▪ Delete Replicate File Definition .....	86
▪ Display Error Log for Replicate File .....	87
▪ Reset Replicate File Error State .....	88
▪ Reset Replicate File In-Use State .....	89

## Replicate File Task Selection Menu

---

By either entering the MR direct command or selecting the MR option on the ETP Maintenance Utility **Main Menu**, the Replicate File Definition Maintenance screen appears.

### The Replicate File Task Selection Menu

```
23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Replicate File Definition Maintenance

AD  Add      replicate file definition
MO  Modify   replicate file definition
DI  Display  replicate file definition
DE  Delete   replicate file definition
DIE Display  error log for replicate files
RE  Reset    error state of replicate file
RI  Reset    in-use state for replicate file

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last      Flip          Tech          Canc
```

The replicate file tasks you select from this menu cause the screens described in the following section to appear.



## Add Replicate File Definition

By either entering the `Add Replicate` direct command or selecting the `AD` option on the **Replicate File Definition Maintenance** menu, the Add Replicate File Definition screen appears.

### The Add Replicate File Definition Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Add replicate file definition

Replicate   file ... DBID      FNR      Verify FDT      Password
Master     file ..* _____ _____          Y (Y/N)          _____
Confirmation file ... _____          Y (Y/N)

Replication criterion (ISN/Key)..... I (I/K)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last Save Flip                      Tech          Canc

```

To define an ETP replicate file, enter the information in the fields, as described below.

### Screen Field Description

Replicate File DBID/FNR	Enter the database ID and file number of the Adabas file that becomes a replicate file. Specify Y (yes) under „Verify FDT“ to verify the file's Field Definition Table (FDT); otherwise, enter (or allow a default of) N (no). You must enter a database ID and file number; there are no defaults for these values. To protect the replicate file from any accidental or unauthorized update by an application, you can define a password for the replicate file for ETP to use when applying transactions to the replicate file. The password is defined using Adabas Security.
----------------------------	--

Master file DBID/FNR	Enter the database ID and file number of the master file for the replicate file you specified above. If you enter a wild card for the database ID and/or file number, a list of the master files already defined is displayed. You can then select a master file from the displayed list. To ensure data consistency, the log file must be empty; otherwise, adding a replicate file is not allowed.
Confirmation File FNR	Enter the database ID and file number of the Adabas file that becomes a confirmation file for the replicate file specified above. The confirmation file must be on the same database as the replicate file; a single confirmation file can be used for multiple replicate files. You must enter a confirmation file number; there is no default, even if a confirmation file already exists on the replicate file's database. Specify Y (yes, the default) under „Verify FDT“ to verify the confirmation file's Field Definition Table (FDT); otherwise, enter N (no).
Replication criterion (ISN/Key)	<ul style="list-style-type: none"> <li>■ Enter I (the default) in this field if the duplicate ISNs are to be used to identify corresponding records in the master and partially replicated file.</li> <li>■ Enter K if a unique Adabas descriptor field has been defined as the distribution key for the master file.</li> </ul> <p>Using duplicate ISNs (option I) is more efficient, but requires a complete Associator (ASSO) for the duplicate file, irrespective of the number of replicated records in the file being modified. Option K allows use of an ASSO subset, but causes additional calls to the replicated file's database.</p>

After pressing `Enter`, the screen displays „More: +“ in the top right corner. When a distribution key has been defined for the master file and you wish to change or add a distribution key value, enter either `FORWARD` or `+` on the command line to display the screen for defining the distribution key values.

### Defining Distribution Key Values

When you define a replicate file containing only a subset of master file records, you can also define a value or value range for the distribution key. The distribution key is the control field in each file record that ETP uses to determine which records are contained in each replicate file.

### Replicate File Key Definition Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                   Add Replicate File Definition                               More: -

Distribution key format A30

Start value          End Value
-----
-----
          .          .
          .          .
          .          .
    
```

When you enter either FORWARD or + in the first Add Replicate File Definition screen, the screen shown above appears. Here you can enter up to 14 values and/or ranges of values contained in the record field with the format shown in the Distribution Key Format field.

If you make an entry in only one of the start or end value columns, the entry is copied automatically to the other column. Multiple values/value ranges are ORed to create a complete „accept“ criterion for the replicate file defined in the first screen.

When the total key length for a format is greater than 32 bytes for a format A field or 16 bytes for a format B field, another column of fields is displayed. You can use these fields to request extra space to enter a key value in full-length, wraparound fashion. If a value longer than 32 or 16 characters respectively is entered, the information entered previously is protected when the window is closed. You can change the value only by reselecting extended editing for the corresponding value.

A distribution key for a format A field is not converted to upper case.

To return to the first screen, use the BACKWARD (PF7 or -) dialogue function.

## Modify Replicate File Definition

By either entering the M`Odify` R`eplicate` direct command or selecting the M0 option on the **Replicate File Definition Maintenance** menu, the Modify Replicate File Definition screen appears.

### The Modify Replicate File Definition Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Modify replicate file definition

Replicate   file  .*  ___211  ___9  Y (Y/N)  _____
Master      file  ...  ___7   ___33  Y (Y/N)  _____
Confirmation file  ...  ___99  ___99  Y (Y/N)  _____

Replication criterion (ISN/Key)..... I (I/K)

```

```

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last  Save  Flip                Tech                Canc
    
```

To change an ETP replicate file's definition, enter the information in the fields, as described below.

### Screen Field Description

<p>Replicate File DBID/FNR</p>	<p>Enter the database ID and file number of an existing replicate. If you enter a wild card in either the database ID or file number field, a window appears from which you can select a single replicate file definition for modification.</p> <p>When the Modify Replicate File Definition screen is first displayed, you must either enter (or have already selected) a valid replicate file; otherwise, you cannot make entries in the rest of the fields. Once you have selected a replicate file and press Enter, you cannot change the file unless you either exit the screen and then re-enter it again, or perform the SAVE dialogue function.</p> <p>Specify Y (yes, the default) under Verify FDT to verify the replicate file's Field Definition Table (FDT); otherwise, enter N (no).</p> <p>To protect the replicate file from any accidental or unauthorized update by an application, you can define a password for the replicate file for ETP to use when applying transactions to the replicate file. The password is defined using Adabas Security.</p>
<p>Master file DBID/FNR</p>	<p>This field displays the current master file when a valid replicate file is first entered. To change the master file for this replicate file, specify a database ID and file number of the new master file. If you enter a wild card for the database ID and/or file number, a list of the master files already defined is displayed. You can then select a master file from the displayed list.</p>
<p>Confirmation File FNR</p>	<p>This field displays the current confirmation file when a replicate file is first entered. To change the confirmation file, enter the file number of an Adabas file that becomes the new confirmation file for the replicate file specified above. The confirmation file must be on the same database as the replicate file; a single confirmation file can be used for multiple replicate files.</p>
<p>Replication criterion (ISN/Key)</p>	<ul style="list-style-type: none"> <li>■ Enter I (the default) in this field if the duplicate ISNs are to be used to identify corresponding records in the master and partially replicated file.</li> <li>■ Enter K if a unique Adabas descriptor field has been defined as the distribution key for the master file.</li> </ul> <p>Using duplicate ISNs (option I) is more efficient, but requires a complete Associator (ASSO) for the duplicate file, irrespective of the number of replicated records in the file being modified. Option K allows use of an ASSO subset, but causes additional calls to the replicated file's database.</p>

When a distribution key has been defined for the replicate file and you wish to change or add a distribution key value, enter either FORWARD or + on the command line to display the screen for defining the distribution key values. See [Defining Distribution Key Values](#) for more information.

## Display Replicate File Definition

By either entering the `Display Replicate` direct command or selecting the `DI` option on the **Replicate File Definition Maintenance** menu, the Display Replicate File Definition screen appears. The displayed output resembles the **Modify Replicate File**.

### The Display Replicate File Definition Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Display replicate file definition

Replicate file ..*   DBID      FNR
                   _____*  _____*

Printer name ..... _____ (' ' = terminal)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip                Tech          Canc

```

### Screen Field Description

Replicate File DBID/FNR	Enter the database ID and file number of an existing replicate file. If you enter a wild card in either the database ID or file number field, a window appears from which you can select one or more replicate file definitions to display.
Printer name	To print out the replicate file definitions, enter a valid printer ID here. Leave this field empty to display the replicate file definitions on your terminal.

## Delete Replicate File Definition

By either entering the `DElete REplicate` direct command or selecting the `DE` option on the **Replicate File Definition Maintenance** menu, the Delete Replicate File Definition screen appears.

### The Delete Replicate File Definition Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Delete replicate file definition

Delete replicate file ..*  _____  _____
                                DBID      FNR

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc
    
```

To delete a replicate file's definition, enter the information in the fields as described below. To make changes to a replicate file's definition, enter the `MODify Replicate` direct command on the command line at the bottom of the current screen.

### Screen Field Description

Delete Replicate File DBID/FNR	Enter the database ID and file number of an existing replicate file. If you enter a wild card in either the database ID or file number field, a window appears from which you can select one or more replicate file definitions to delete.
-----------------------------------	--

If you have enabled the confirmation window using the „Modify User Profile“ function, the confirmation window appears, asking you to confirm the deletion. If the log file of the related master file is not empty, deletion is not allowed.

## Display Error Log for Replicate File

ETP keeps information about failed replication operations in an error log for the replicate files. You can display part or all of the logged error information with this function.

By either entering the `Display Error` direct command or selecting the `DIE` option on the **Replicate File Definition Maintenance** menu, the Display Error Log for Replicate File screen appears.

### The Display Error Log for Replicate File Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Display error log for replicate file

Replicate file .....*  _____  _____

Mark to display data of failing ADABAS transaction
Control block (CB) .. _
Format buffer (FB) .. _
Record buffer (RB) .. _

Printer name ..... _____ (' ' = terminal

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc

```

### Screen Field Description

Replicate file DBID/FNR	Enter the database ID and file number of the replicate file for which you want errors displayed. If you enter a wild card in either the database ID or file number field, a window appears from which you can select a single replicate file definition for displaying the error log.
-------------------------	---

To display more Adabas data from the failed operation, mark the appropriate field on the screen as described here below. Note that not all data are always available:

Control Block	Mark this field with any non-blank character to display the Adabas Control Block for the failed operation.
Format Buffer	Mark this field with any non-blank character to display the Adabas Format Buffer for the failed operation.
Record Buffer	Mark this field with any non-blank character to display the Adabas Record Buffer for the failed operation.
Printer name	To print out any error information selected above, enter a valid printer ID here. Leave this field empty to display the added error log information on the terminal.

## Reset Replicate File Error State

---

By either entering the `RESet Error` direct command or selecting the `RE` option on the **Replicate File Definition Maintenance** menu, the Reset Error State for Replicate File screen appears.

### The Reset Replicate File Error State Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                   Reset error state for replicate file

Replicate file ..*  DBID      FNR
                   _____  _____

Command ==>>>
```



```

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last           Flip           Tech           Canc

```

To reset a replicate file's error state, enter the information in the fields as described below. To make changes to a replicate file's definition, enter the Modify Replicate direct command on the command line at the bottom of the current screen.

### Screen Field Description

Replicate File DBID/FNR	Enter the database ID and file number of an existing replicate file. If you enter a wild card in either the database ID or file number field, a window appears from which you can select one or more replicate file definitions to reset the error state.
-------------------------	---

If you have enabled the confirmation window using the „Modify User Profile“ function, the confirmation window appears, asking you to confirm the reset operation.

## Reset Replicate File In-Use State

By either entering the `RESet In-use` direct command or selecting the `RI` option on the **Replicate File Definition Maintenance** menu, the Reset Replicate File In-Use State screen appears.

### The Reset Replicate File In-Use State Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                   Reset in-use field for replicate file

Replicate file ..*  DBID      FNR
                   _____

Command ==>>>

```

```
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc
```

To reset a replicate file's in-use state, enter the information in the fields as described below. To make changes to a replicate file's definition, enter the Modify Replicate direct command M0 on the command line at the bottom of the current screen.

**Screen Field Description**

Replicate File DBID/FNR	Enter the database ID and file number of an existing replicate file. If you enter a wild card in either the database ID or file number field, a window appears from which you can select one or more replicate file definitions to reset their in-use flags.
-------------------------	--

If you have enabled the confirmation window using the „Modify User Profile“ function, the confirmation window appears, asking you to confirm the reset operation.

# 12 Replicating Logged Transactions

---

▪ Function Description .....	92
▪ Starting Replication Tasks Manually .....	92
▪ Starting Replication Tasks with a User Program .....	92
▪ Self-Restart Operation .....	92
▪ Error Handling during Replication .....	93
▪ Displaying the Replicate Database Summary .....	93
▪ Processing of Replication Task Messages .....	94
▪ Replicate Transactions .....	94

## Function Description

---

Actual replication operation first begins when the ETP replication task or tasks are running. Each start of a replication task causes the task to check and, when needed, to resynchronize its replicate files with the master. You can choose to restart a replication task manually, to restart the replication task by running your own program after each successful ET transaction on the master file, or to allow the self-restart function to restart and thereby resynchronize its replicate files each time a specified time period elapses. Software AG recommends that you start the replication task as a batch job (see ETP Installation, [Running ETP in Batch Mode](#)).

## Starting Replication Tasks Manually

---

If you do not specify a restart interval for a replication task, you must periodically start a replication task manually using the [Replicate Transactions](#) screen as described in this function description to resynchronize the related replicate files with the master.

## Starting Replication Tasks with a User Program

---

Following every ET transaction that updates a master file, you have the option to invoke your own program. This program can be designed to restart the master file's replication task. The replication task then updates the related replicate files with the change logged by the ET transaction.

Your user program must first be defined in the [Modify System Profile](#) screen and enabled for the master file in either the [Add Master File Definition](#) or [Modify Master File Definition](#) screen. For more information about creating the ET user program, see [ET-Time User Program](#).

## Self-Restart Operation

---



**Note:** Self-restart operation is recommended for batch operation only; online self-restart operation locks the user's terminal.

In self-restart mode, the replication task restarts at intervals defined for the replication task. After each interval, which is measured between two consecutive restart times, ETP restarts the replication task to force it to check for transactions that have been logged for the master file but not yet applied to the related replicate files.

---

## Error Handling during Replication

---

If an error occurs while a previously logged transaction is being applied to a replicate file, the error is recorded and can then be displayed using the „Display Error Log for Replicate Files“ function. To prevent further errors, replicate files which are in error status are not resynchronized.



**Note:** When you use this function, the Natural DATSIZE buffer (explained in the Natural *Parameter Reference* documentation for details on specifying the DATSIZE profile parameter) will be increased to approximately 170 KB.

The Adabas response codes 9, 145, 148 and 224 are considered to be transient; that is, they are assumed to not occur again for the next replication. If one of these codes does occur, the error status of the replicate file does not need to be reset explicitly. The code will be ignored when replication restarts.

To restore the replicate file to normal status, use the „Reset Error“ utility function. Ensure that the failure cause has been corrected before resetting the error code; see the [Reset Error State for Replicate File](#) function. Error codes for database errors are in the Natural message range 3001-3255. Refer to ETP Installation, [Specifying Master File Databases](#), for more information.

If a catastrophic error occurs (the job or session has been cancelled) while the replication task is using a replicate one or more replicate files remain marked as being „in use“. Replicate files which are designated „in use“ are not resynchronized; this prevents parallel access by different tasks. To restore the state of the replicate file, use the [Reset Replicate File In-Use State](#) function.

The task reports the number of replicate files which are in error status, or in use. Use the [Display Error Log for Replicate Files](#) function to show which files are in error.

If, however, the error report does not provide you with enough information to correct the error, proceed as described under the heading [Reporting ETP Errors](#).

---

## Displaying the Replicate Database Summary

---

The replication task displays a summary of the activities the task performs on processed replicate databases. This summary information can be useful when rearranging master and log file definitions to improve replication performance. This summary is not displayed if an error occurs.

## Processing of Replication Task Messages

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. (For more information about the `WADUSER3` subprogram, which is delivered in source form with ETP, see *ETP Installation*.)

## Replicate Transactions

You must use the following screen function to manually replicate master file changes to either all or selected replicate files.

By either entering the Replicate Transactions direct command `REP TR` or selecting the `R` task on the ETP **main menu**, the screen shown below appears.

### The Replicate Transactions Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                        Replicate transactions

Log file and replicate database range

                DBID      FNR
First log file .....* ____1      ____1
Last log file .....* ____254     ____255
First replicate DB ..* ____1
Last replicate DB ..* ____254

Self-restart interval ..... _____ (hh:mm:ss)
Number of updates for which ETs are to be skipped .. ____200
Number of READs before STOP TASKS flag is checked .. ____1000
User-specific ET data processing                      N (Y/N)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last          Flip          Tech          Canc
    
```

The screen fields and their possible settings are described in the following table:

## Screen Field Description

First Log File DBID/FNR	Enter the database ID (DBID) and file number (FNR) of the first (inclusive) or only log file for which the replication task is to be started. To start a task for only one log file on a database, enter the database ID and file number in both these fields and in the „Last Log File“ DBID/FNR fields. The default (database 1/file 1) in this field and the „Last Log File“ field default select all master/replicate files. If you enter a wild card in either the database ID or file number field, a window appears that lists all selected log files. You can select only one file.
Last Log File DBID/FNR	Enter the database ID and file number of the last (inclusive) or only log file for which the replication task is to be started. The default (database 65535/file 65535) in this field and the „First Log File“ field default select all files. If you enter a wild card in either the database ID or file number field, a window appears that lists all selected log files. You can select only one file.
First/Last Replicate DBID	Enter the database ID of the first/last (inclusive) or only replicate database to which the replication task will apply transactions of the selected log files. If you enter a wild card in either the database ID, a window appears that lists all selected databases. You can select only one database.
Self-Restart Interval	Specify the elapsed time between restarts of the replication task. If this field is left blank, no restart interval applies. Valid values range „00:00:01“ through „24:00:00“.
Number of updates for which ETs are to be skipped	Enter the count of update calls to the master file for which corresponding ET calls to the replicate file's database <i>will not</i> be made. Each ET issued to the master file results in a corresponding ET call to the replicate file's database, preceded by updates to the affected confirmation files. With this count ( <i>n</i> ), you can specify that the first <i>n</i> update calls to the master file <i>will not</i> result in ET calls to the replicate file's database. If an ET is <i>not</i> issued, the confirmation files are also not updated. The default (0) means that no ET calls to the replicate file's database are skipped. When the specified count of calls has been skipped, the calls begin and continue until the next ET call to the master file. The count is then reset, and skipping begins anew.
Number of READs before STOP TASKS flag is checked	This number is the count of READ calls made before ETP checks for a user-requested stop for all <b>asynchronous tasks</b> . If asynchronous tasks are to be stopped, execution ends after the next ET command.
User-Specific ET data	Specify Y (yes) to perform an explicit Adabas OP ET processing command for the replicate files to set the ET user ID of a user who updates the master file. The default is N (no). Skipping ETs is not possible when user-specific ET data processing is desired.





# 13 Controlling Execution of Asynchronous Tasks

---

- Function Description ..... 98
- Control Execution of Asynchronous Tasks Screen ..... 98

## Function Description

---

The ETP asynchronous tasks resynchronize replicate files with their master files only when the tasks are started/restarted. If no self-restart interval is defined for the replicate transactions or if no ET time user program starts the task, you must start the task manually (see the [Replicate Transactions](#) function) to apply any unapplied master file changes to the replicate files. This function stops all active tasks at a point where no interruption occurs for the resynchronization process.



**Note:** Asynchronous tasks that clean up log files are affected by this function in the same way.

## Control Execution of Asynchronous Tasks Screen

---

By either entering the C**ON**trol T**A**sk direct command or selecting the C**0** task on the ETP main menu, the Control Execution of Asynchronous Tasks screen appears.

### The Control Execution of Asynchronous Tasks Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                   Control execution of asynchronous tasks

To stop all asynchronous tasks which use the current
administration file, enter Y below. All executing tasks
are stopped when they finish execution on their assigned
range of files. If any task is started or self-restarted
before task execution is re-enabled, the task is stopped
immediately.
Note: Asynchronous task control affects tasks that
      - replicate transactions
      - clean up log files.

To re-enable execution of asynchronous tasks, enter N
below, save the change and restart the tasks manually.

Stop all asynchronous tasks      N (Y/N)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc
    
```

**Screen Field Description**

Stop all Asynchronous Tasks	Specify Y to stop all tasks under control of the active administration file. Setting this value to N re-enables (but does not restart) the tasks.
-----------------------------	---

---

# 14

## Deleting Successfully Replicated Transactions

---

▪ Function Description .....	102
▪ Clean Up Log File .....	102
▪ Processing of Clean-Up Task Messages .....	104

## Function Description

---

A transaction that changes the master file is logged in the master file's log file to be applied later to the replicate files. After ETP applies the transactions to the replicate files, you can remove the completed transactions from the log file using this function.

## Clean Up Log File

---

By either entering the Cleanup Logfile direct command `CL` or selecting the `CL` task on the ETP **main menu**, the Clean Up Log File screen appears.

### The Clean Up Log File Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                          Clean up log file

                                DBID      FNR
First log file to be processed ..*  _____1  _____1
Last  log file to be processed ..*  _____254  _____255

Time stamp or time value for last transaction to be deleted

Time stamp .. A6A6BDBD561F6000      (hexadecimal value)
Time ..... 1992-11-26 10:07:29.3 (yyyy-mm-dd hh:mm:ss.t)

Refresh file if all transactions are replicated .... N (Y/N)
Self-restart interval ..... (hh:mm:ss)
Number of updates for which ETs are to be skipped .. 0
Number of READs before STOP TASKS flag is checked .. 1000

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc
    
```

## Screen Field Description

First log file to be processed (DBID/FNR)	Enter the beginning database ID and file number of the first (inclusive) or only log file for which replicated transactions are to be deleted. If you enter a wild card in either the database ID or file number field, a window appears from which you can select a single log file.
Last log file to be processed (DBID/FNR)	Enter the ending database ID and file number of the last (inclusive) or only log file for which replicated transactions are to be deleted. If you wish to specify only one log file on one database, enter that DBID/file number in <i>both</i> this field and in the <i>Start DBID/FNR</i> field. If you enter a wild card in either the database ID or file number field, a window appears from which you can select a single log file.

You can limit deletion to all transactions up to and including a time earlier than the current time by entering that earlier time in one of the following fields:

Time stamp	Enter a valid time stamp as stop time. Transactions that are time-stamped with a later (newer) time will not be deleted. If you enter a time stamp value here, do not enter a date/time value in the next field. The default is the current time (the time when you called this screen).
Time	Enter a valid date/time value for the confirmation file. Transactions that are time-stamped with a later (newer) time will not be deleted. If you enter a date/time value here, do not enter a time stamp value in the previous field. The default is the current time (the time when you called this screen).  Entries in the Time field are first converted to the time stamp value and then back to normal notation. This conversion can result in a value slightly different from the entered value.
Refresh file if all transactions are replicated	If you specify Y, ETP refreshes the log file when all transactions are applied to all replicate databases. Because updating the Adabas index for deletions is quite resource consuming, refreshing a file is much more efficient than deleting all transactions separately. Note that to use this option, the ADALOD parameter PGMREFRESH=YES must have been specified when the log file was created. The default is N, meaning no automatic refresh of the log file occurs.  Note that it is impossible to refresh a log file that is also used as an administration or confirmation file.
Self-Restart Interval	Specify the elapsed time between restarts of the clean-up task. If this field is left blank, no restart interval applies. Valid values range 00:00:01 through 24:00:00.
Number of updates for which ETs are to be skipped	Enter the count of update calls to the log file for which corresponding ET calls <i>will not</i> be made. With this count ( <i>n</i> ), you can specify that the first <i>n</i> update calls to the log file <i>will not</i> result in ET calls.  The default (0) means that no ET calls to the replicate file's database are skipped. When the specified count of calls has been skipped, the calls begin and continue until the next ET call to the master file. The count is then reset, and skipping begins anew.

Number of READs before STOP TASKS flag is checked	This number is the count of READ calls made before ETP checks for a user-requested stop for all <b>asynchronous tasks</b> . If asynchronous tasks are to be stopped, execution ends after the next ET command.
---	--

## Processing of Clean-Up Task Messages

---

The subprogram `WADUSER3` is used to display all messages issued by the clean-up task. `WADUSER3` can be modified to filter the task messages and, if desired, send them directly to the operator console. (For more information about the `WADUSER3` subprogram, which is delivered in source form with ETP, see *ETP Installation*.)



# 15

## Displaying X-Ref Transaction State

---

- Function Description ..... 106
- X-Ref Transaction State Screen ..... 106
- X-Ref Transaction State User Application Programming Interface ..... 107

## Function Description

---

Checking cross-references, or „X-refing“, for transactions displays the state of the transactions contained in the selected log file or files. The resulting information shows which transactions have been applied to which replicate database files and, if applicable, special replicate file states („in use“, „error“, etc.).

## X-Ref Transaction State Screen

---

By either entering the Xref TRansactions direct command or selecting the X option on the **main menu**, the „X-Ref Transaction State“ screen appears.

### The X-Ref Transaction State Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    X-Ref transaction state

Log file .....*   DBID      FNR
                  _____  _____

Information level .. 1 (1=short, 3=extended)

Printer name ..... _____ (' ' = terminal)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last      Flip      Tech      Canc
    
```

To cross-reference check a log file, enter the information in the fields as described below. To display a master file's definition without log or replicate file information, enter the Display Master direct command DI MA on the command line at the bottom of the current screen.

### Screen Field Description

Log File DBID/FNR	Enter the database ID and file number of the log file you want cross-reference checked. If you enter a wild card in either the database ID or file number field, a window appears from which you can select one or more log file definitions to check.	
Information Level	specifies the amount of information to be displayed for every selected log file. The higher the level, the more information is displayed and the more database calls and processor time is required to provide the requested information:	
	1	Display the date/time of the last transaction applied to the replicate databases.
	2	Same as 1, but additionally displays date/time of the generation of the last transaction in the log file.
	3	Same as 2, but additionally displays the number of transactions that are still unreplicated.
Printer Name	To print out the information for the selected master files, enter a valid printer ID here. Leaving this field blank causes the resulting information to be displayed on your terminal.	

## X-Ref Transaction State User Application Programming Interface

With correction ET15205 applied to ETP Version 1.5.2, the user application programming interface (API) WADUSR4N is available which provides information similar to the Xref TRansactions direct command described above. Instead of displaying the available information on the screen, WADUSR4N returns the information in output parameters.

In addition to the subprogram WADUSR4N, the following Natural objects are also delivered:

- parameter data area WADUSR4A, which describes the input and output parameters of subprogram WADUSR4N,
- program WADUSR4P, which contains a sample call to WADUSR4N.

WADUSR4A and WADUSR4P are delivered in source form in library SYSETP, whereas WADUSR4N is delivered as a cataloged object in library SYSETP.

Because WADUSR4N uses ETP service routines, it must either be called from library SYSETP or have library SYSETP as a steplib library.

---

# 16

## Using Special Functions

---

▪ Special Functions Screen .....	110
▪ Display Transactions .....	110
▪ Delete Transactions .....	112
▪ Modify Confirmation File .....	114
▪ Modify System Profile .....	115
▪ Modify User Profile .....	116
▪ Check Definitions for Performance Problems .....	118
▪ Change Administration File .....	119

This document describes certain ETP maintenance functions that are not performed often, if at all; some of these are operations that should only be performed after careful consideration.

## Special Functions Screen

---

By either entering the SPecial direct command or selecting the S option on the ETP **main menu**, the Special Functions screen appears.

### The Special Functions Screen

```
23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Special Functions

DI  Display transactions
DE  Delete  transactions
MC  Modify  confirmation file time stamp
MS  Modify  system profile
MU  Modify  user profile
CD  Check  definitions for performance problems
CA  Change  to another administration file

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip          Tech          Canc
```

## Display Transactions

---

By either entering the DIisplay TRansactions direct command or selecting the DI option on the Special Functions selection menu, the Display Transactions screen appears.

## The Display Transactions Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                        Display transactions

Log file .....*   DBID       FNR
                  _____

Time stamp or time start value for display
Time stamp ..* 0000000000000000>      (hexadecimal value)
Time .....* 1900-01-01 00:00:00.0> (yyyy-mm-dd hh:mm:ss.t)

Mark to display additional data for selected ETs
Control block (CB) ..... _
Format buffer (FB) ..... _
Record buffer (RB) ..... _
CB, FB, RB as selected for all DB operations .. _

Printer name .. _____ (' ' = terminal)

Command ===>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last          Flip          Tech          Canc

```

By default, a window appears that displays only the ET timestamps of previously logged transactions. If you want the CB (Control Block), RB (Record Buffer), or FB (Format Buffer) to be displayed for the entry, mark the appropriate fields in the screen. To display the marked data for all DB operations, mark the CB, FB, RB... screen entry.

### Screen Field Description

Log file DBID/FNR	Enter the database ID and file number of the log file for which you want transactions displayed. If you use wild card notation for the log file, a window appears which allows you to select a single log file for which the transactions are to be displayed.
Time Stamp	Enter either the hexadecimal ET time stamp to display a specific log file entry, or enter a wild card to display a list of all log file entries having time stamp values that meet the selection criterion. If you enter a value here, do not enter a time value in the next field.

Time	Enter a valid ET date/time to limit the displayed log file entry, or enter a wild card to display a list of all log file entries having time values that meet the selection criterion. If you enter a value here, do not enter a time stamp value in the field above.  <b>Anmerkung:</b> Entries in the Time field are first converted to the time stamp value and then back to normal notation. This conversion can result in a value slightly different from the entered value.
Control Block	Mark this field with any non-blank character to display the Adabas Control Block.
Format Buffer	Mark this field with any non-blank character to display the Adabas Format Buffer.
Record Buffer	Mark this field with any non-blank character to display the Adabas Record Buffer.
CB, FB, RB ... for all DB Operations	Mark this selection with any non-blank character if you also want to display the data selected in the above fields for all STORE, UPDATE and DELETE operations.
Printer Name	To print out the log file entries, enter a valid printer ID here. Leaving this field blank causes the information to be displayed on your terminal.

## Delete Transactions

By either entering the DElete TRansactions direct command or selecting the DE option on the Special Functions selection menu, the Delete Transactions screen appears.

### The Delete Transactions Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                        Delete transactions

Log file .....*          DBID          FNR

Time stamp or time start value for display
Time stamp ..* 0000000000000000>          (hexadecimal value)
Time .....* 1900-01-01 00:00:00.0> (yyyy-mm-dd hh:mm:ss.t)
    
```



```

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last           Flip           Tech           Canc

```

**Caution:** The „Delete Transactions“ function allows you to remove log file entries that *have not been completed* as well as those that have. Removing uncompleted transactions can cause loss of master/replicate file integrity. To remove only completed transactions, use the **Clean Up Log File** function.

Selection of transactions is restricted to the entries for a single log file. If you use wild card notation for the log file, a window appears which allows you to select a single log file for which the transactions are to be displayed.

### Screen Field Description

Delete Entries for Log	Enter the database ID and file number of the log file for which File DBID/FNR transactions are to be deleted. If you use wild card notation for the log file, a window appears which allows you to select a single master file for which the transactions can be deleted.
------------------------	---

The following Time Stamp and Time fields allow you to select transactions to be deleted, either individually or after a specific time:

Time Stamp	Enter either the hexadecimal ET time stamp to delete a specific transaction, or enter a wild card to display a list of all transactions having time stamp values that meet the selection criterion. If you enter a value here, do not enter a time value in the next field.
Time	Enter a valid ET date/time to limit the transactions displayed for deletion, or enter a wild card to display a list of all transactions having time values that meet the selection criterion. If you enter a value here, do not enter a time stamp value in the field above.  <b>Anmerkung:</b> Entries in the Time field are first converted to the time stamp value and then back to normal notation. This conversion can result in a value slightly different from the entered value.

This function displays a list of all transactions for a single specified log file:

### The Delete Log File Entries for Master File Window

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Delete transactions


+-----Transactions-----+
!                               !
Log file ! -----Log File (10,135) ----- !
!   Time stamp           Time           Replicated !
!   000000000000000000>  1900-01-01 00:00:00.0>  (Y/N) !

```

!	_	A691B22513246901	1992-11-09	16:23:47.8	Y	!
Time stamp!	_	A691B65A8DFDCE01	1992-11-09	16:42:37.6	Y	!
Time st!						!
Time ..!						!
!						!

## Modify Confirmation File

ETP keeps information about replicated transactions in a confirmation file for the replicate files. Using this function, you can change the confirmation file time stamp.

 **Caution:** You should use this function under special conditions and with extreme care, only to modify a replicate file's time stamp; otherwise, the integrity of the replicate file can be destroyed.

By either entering the M`O`Modify Confirmation direct command or selecting the `MC` option on the [Special Functions](#) selection menu, the Modify Confirmation File screen appears.

### The Modify Confirmation File Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Modify confirmation file time stamp

Replicate file ..*  _____  _____
                    DBID          FNR

Time stamp or time value of confirmation file

Time stamp .. _____ (hexadecimal value)
Time ..... _____ (yyyy-mm-dd hh:mm:ss.t)

Command ==>_____
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last          Flip          Tech          Canc
    
```

When the screen is first displayed, all fields except the replicate file DBID/file number are protected against updating. After you enter a valid DBID/file number, those fields are then protected against

updating, but all other fields can be changed. To change your original DBID/file number selection, you must exit and then reselect the screen.

### Screen Field Description

Replicate file DBID/FNR	Enter the database ID and file number of the replicate file whose confirmation file time stamp you want to modify. If you enter a wild card in either the database ID or file number field, a window appears from which you can select a single replicate file definition whose time stamp you want to modify.
Time stamp	Enter a valid time stamp for the confirmation file. If you enter a time stamp value here, do not enter a date/time value in the next field. The initial value of this field is the current time stamp of the confirmation file.
Time	Enter a valid date/time value for the confirmation file. If you enter a date/time value here, do not enter a date/time value in the previous field. ETP converts any time value you enter to a hexadecimal time stamp value, which ETP uses for all comparisons.  <b>Anmerkung:</b> Entries in the Time field are first converted to the time stamp value and then back to normal notation. This conversion can result in a value slightly different from the entered value.

## Modify System Profile

The ETP operating profile, defined when the administration file is first installed, permits you to set parameters that apply to all definitions contained in the administration file.

### The Modify System Profile Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Modify system profile

Parameters for program start after master file ET

Name of program to be started ..... HUGO____
Terminal ID (used for CICS only) ..... NOWHERE_

Verify transaction consistency during logging .. N (N/Y)

```

```

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Last          Flip                Tech                Canc
    
```

By either entering the Modify Sysprof direct command or selecting the MS task on the Special Functions menu, the Modify System Profile screen. The screen fields and their possible settings are:

**Screen Field Description:**

Parameters for program start after master file ET:

Name of Program	Specify the name of the user program you want ETP to run after to be Started each ET command performed on a master file. The master file's definition (see <a href="#">Add Master File Definition</a> screen or <a href="#">Modify Master File Definition</a> screen) must enable the user program call. Only one user program can be defined. For more information, see <a href="#">ET-Time User Program</a> .
Terminal ID	(CICS only) Specify the terminal ID when the user program defined above runs under CICS.
Verify transaction consistency	If, during logging, you want ETP to verify that all master files during logging updated in the same transaction have the same log file assigned, specify „Y“. If multiple master files with different log files are updated by a single transaction, the consistency of the transaction during replication cannot be preserved. If you specify „N“ (no, the default), transaction consistency is not checked.

## Modify User Profile

Certain ETP maintenance utility tasks display confirmation windows before executing. These Are you sure?... windows are always displayed for those functions unless you disable them for the individual user's profiles. This „Modify User Profile“ function lets you control the various confirmation windows.

By either entering the Modify Userprof direct command or selecting the MU task on the Special Functions menu, the Modify User Profilescreen appears. The screen fields and their possible settings are:

## The Modify User Profile Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Modify user profile

Profile for user HAL

Display a window before EXIT, CANCEL, MENU .. Y (Y/N)
Display a window before DELETE, RESET..... Y (Y/N)
Display a window before SAVE, EXECUTE ..... Y (Y/N)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last           Flip                               Tech           Canc

```



**Note:** The default values shown in the figure above are the original values effective at installation for all users, and are displayed when you first select the screen. The field descriptions below describe how to display the current settings for a specific user.

### Screen Field Description

Display a window before EXIT, CANCEL, MENU	Enables the confirmation window before the current EXIT screen, CANCEL, and MENU (return to the main menu) functions. An <b>example of a confirmation window</b> that appears, if enabled (Y), is shown under the heading <i>Dialogue Functions</i> .
Display a window before DELETE, RESET	Enables the confirmation window before one of the following objects is actually deleted; - master file definition - replicate file definition - logged transactions in the log file - the in-use or error indication is reset for a replicate file
Display a window before SAVE, EXECUTE	Enables the confirmation window before either the SAVE (current data) or EXECUTE (specified program).

The following applies:

- When the EXIT or MENU dialogue functions are executed and the confirmation window is disabled, any selected functions are performed directly before the dialogue functions are performed.

- When the CANCEL dialogue function is executed and the confirmation window is disabled, any selected functions are **not** performed before the CANCEL dialogue function is performed.
- When the confirmation window for deleting or resetting objects is disabled, the objects are deleted or the flags are reset directly before any dialogue function or direct command is performed.
- When the SAVE or EXECUTE confirmation window is disabled, the dialogue functions are performed.
- Direct commands imply execution of the EXIT dialogue function.

## Check Definitions for Performance Problems

---

To check your master and replicate file definitions for potential performance problems, use the `Check Definitions...` function. „Potential“ problems are problems that are reported by the function but that may not actually occur due to, say, a low update frequency of the log file. However, the hints given by the `Check definitions...` function should be checked carefully.

### The Change Administration File Screen

```
23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                  Check definitions for performance problems

Display definition xref .. N (Y/N)
Printer name .....

(' ' = terminal)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
          Help Menu Exit Last          Flip          Tech          Canc
```

By either entering the `CHECK DEFINITIONS` direct command or selecting the `CD` task on the Special Functions menu, the Change Administration File screen appears.

## Screen Field Description

Display definition xref	If Y (yes) is specified, all master and replicate file definitions are summarized to show their relationships and to give an overview.
Printer Name	To print out the check results, enter a valid printer ID here. Leaving this field blank causes the information to be displayed on your terminal.

## Change Administration File

Use the „Change Administration File“ function to specify a new administration file for the ETP environment. You can use this function to switch back and forth among different administration files from within the same application at a single node.

By either entering the CHAnge Administration direct command or selecting the CA task on the Special Functions menu, the „Change Administration File“ screen appears.

### The Change Administration File Screen

```

23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                   Change administration file

Current administration file ...      DBID      FNR
                                10          152
New      administration file .. _____

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last          Flip          Tech          Canc

```

### Screen Field Description

Current administration file DBID/FNR	These fields show the database ID and file number of the administration file that is currently active.
New administration file DBID/FNR	Enter the database ID and file number of the file you want to become the new administration file.



# 17 Executing a Natural Command

---

- Function Description ..... 122
- Invoking and Using the Function ..... 122

This document describes the use of the Execute a Natural Command function.

## Function Description

---

The Execute a Natural Command function allows you to execute a Natural command or run a Natural program from the ETP maintenance utility.

## Invoking and Using the Function

---

By either entering the Natural direct command or selecting the N task on the ETP main menu, the Execute a Natural Command screen appears.

### The Execute a Natural Command Screen

```
23:59:59          ***** ENTIRE TRANSACTION PROPAGATOR *****          2000-12-24
                    Execute a NATURAL command

Enter a NATURAL system command or the name of a NATURAL program
located in a steplib and press Enter for execution

Command .... _____

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Last          Flip          Tech          Canc
```

**Screen Field Description**

Command	Specify the Natural command or „steplib“ program to be run.
---------	---

When you press `Enter`, ETP executes the command or program.

When completed, the command or program returns you to this screen.



# 18 ETP Restart and Recovery

---

▪ ETP Restart and Recovery Behaviour .....	126
▪ General Errors that Can Occur .....	126
▪ File Saving Requirements for Error Recovery .....	126
▪ Recovering from the General Types of Failures .....	127
▪ Correcting NAT3400, NAT9988 and NAT3606 Errors .....	140

## ETP Restart and Recovery Behaviour

---

Following any error, ETP automatically attempts to recover from the error and continue with the operation without any operator intervention. Of course, whether or not intervention is needed depends on the type of error.

### General Errors that Can Occur

---

There are five general types of possible errors:

- Inaccessible master file (or its database), but with an intact Adabas log;
- Inaccessible master file (or its database), and a damaged Adabas log;
- Damaged replicate file (or its database) that is restorable to its current state;
- Damaged replicate file (or its database) that needs to be restored to an earlier status;
- The administration file has lost integrity, and must be restored to an earlier status.

ETP uses the basic Adabas restart/recovery process to correct the errors that can be restored from the undamaged Adabas log or master file. Such failures can be restored when the master file/database is again available. In these cases, Adabas restart/recovery recovers all participating databases and restores the integrity of the master and its log file, and of the replicate files and their confirmation files.

### File Saving Requirements for Error Recovery

---

For those errors where a file must be restored to an earlier status, the file must first be restored and then corrected from the Adabas log, using the Adabas restart/recovery facility. If, however, a catastrophic (data-damaging) error makes such restoring impossible, you must restore the files to an earlier state from file copies (obtained with the Adabas `ADAULD` or `ADASAV` utility). It is therefore specially important to obtain copies of at least the most critical ETP files regularly.

## Critical File Saving Requirements

To permit restoring of lost or damaged files, the master, log and administration files, you must obtain copies regularly with ADAULD or ADASAV. Second, only information from the time before the master file was last copied should be deleted from the log file. The date and time when the copies were made should be recorded to ensure that the most recent copy of each file is used, if it is needed. The log file should always be copied at the same time as its master file.

## Saving Replicate Files

If appropriate, replicate files and their confirmation files should also be copied using ADAULD or ADASAV after the master file has been copied. This eases automatic recovery if there has also been damage to a replicate file's Adabas log information. Copying the replicate files is not required because the replicate files can be restored from the master/log file copy. The confirmation file should always be copied at the same time as its replicate file, and the replicate file should always be copied after its master file is copied.

When „partial replicate“ files are damaged, you must use a copy of either the master file, the partial replicate file itself or of a full replicate to restore the replicate file. If the partial replicate is restored from the master or a full duplicate copy, you must perform reinitialization.

If a confirmation file is used by multiple replicate files, copy *all* replicate files and the confirmation file at the same time. If it becomes necessary to restore one of the files, *all* files must be restored.

## Recovering from the General Types of Failures

---

The rest of the chapter describes the recommended procedures for restoring ETP operation after one of the general failures summarized earlier. In any of the following cases, the general recovery procedure is to:

1. Restore the data integrity of each database to the most recent level, using the autobackout/auto-restart facility of Adabas;
2. Ensure that all links between the master files and their replicate files are active and running.

### **Case 1: Recovering from an Isolated Master/Replicate File or Database**

If the master file or its database is not available but is fully operable, local processing of related replicate files can still continue normally; that is, all read requests can be satisfied. In cases where local updates are made to an isolated master file, the updates will be applied to each remote replicate file by the replication task, once the isolation is removed and the task is restarted.

The figure below shows the general recovery sequence to follow when a failure occurs that isolates a master file from its replicate files. Such a failure is usually caused by the loss of a Net-Work (or other comparable database interconnecting system's) link.

#### **Isolated Master/Replicate File Recovery Flowchart**





**Damaged Replicate/Confirmation File Recovery Flowchart**





If the failure not only isolated the master/replicate file but also interrupted an update transaction that changed the master file, the Adabas autobackout/ autorestart must first restore each affected database to its pre-transaction state. Autobackout removes any changes made during the failed transaction; autorestart then reruns the failed transaction to reapply the update to the master file.

If no update transaction was running when the failure occurred, you only need to re-establish the link and start the replication task or tasks to restore the ETP file integrity. Tasks running with a defined restart interval need not be restarted.

### **Case 2: Recovering from a Damaged Replicate/Confirmation File**

For a failure that damages a replicate file or its confirmation file, you can usually recover the damaged file or files with the protection log, the ADALOD utility, and using the file copy obtained with the Adabas ADAULD or ADASAV utility to reconstruct the file to a recent state before the failure. In most cases, however, it might be quicker to simply reload the replicate/confirmation file from a recent copy.

Using the ETP maintenance utility, you can then restart the related replication task to apply uncompleted master file changes, if any, to the restored (and any other unsynchronized) replicate file. The figure above shows the general recovery sequence for such a failure.

### **Case 3: Recovering from/Restoring a Damaged Log File**

In this situation, the Adabas database's log for the master file has been damaged, preventing recovery. Here, one of the following must be performed:

- Restore the log file from the Adabas PLOG and the log file copy obtained using the ADAULD or ADASAV utility;
- Reinstall the master/log files from an earlier level, reload the replicate files at that same level, and then apply any pre-failure updates.

The figure below shows the sequence for recovering from a damaged log file.

### **Damaged Log File Recovery Flowchart**





#### Case 4: Restoring a Damaged Database and Master/Log File

If both the master and its log file have been irreparably damaged, the possible sources of a new master/log file are:

- Copy of the Adabas file, combined with the protection log;
- Recent master/log file copy;
- Full replicate file.

If the master and log files can be restored to their condition just before the failure by the Adabas file copy/protection log recovery, then you can restart the ETP task for that file and repeat any incomplete master file transaction related to the failure. ETP should then resynchronize the master with its replicate files at the next restart of the replication task.

If, however, the master/log files cannot be rebuilt to the pre-failure state you must determine which transactions are recoverable. Second, you must determine whether a recent master/log file copies or an available „full duplicate“ replicate file can be used to restore the master and log files to a level to which the recoverable transactions can be applied to raise the master/log files to their state just before the failure.

##### Restoring the Master File

If there is a master file copy that is newer than the most recently resynchronized full duplicate replicate file, you can use that master file copy to restore the master file. Otherwise, use that newest replicate file as source for restoring both the master file and any other „older“ replicate files. After restoring the master and affected replicate files, you must reapply all update transactions to the master file that are not already in the restored files. These reapplied transactions will be reflected in the replicate files when you restart an ETP replication task for that master file.

##### Restoring the Log File

The log file should be restored from the same source and at the same level as the master file. Even if the log is repairable at a newer level than the master file, there is a chance that it will not reflect the master file state accurately. Therefore, you should always obtain copies of the master and log files together, and the confirmation and replicate files together.



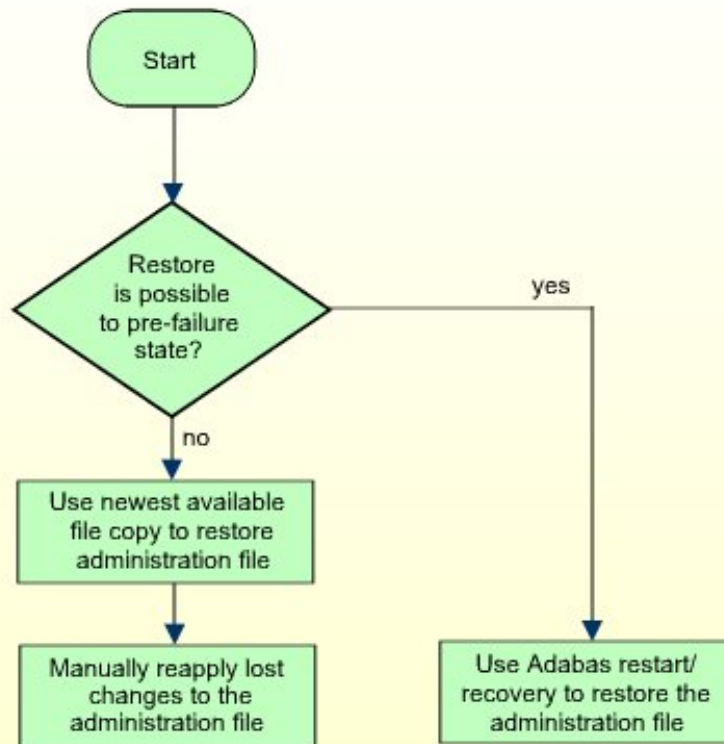
## Damaged Master and Log File Recovery Flowchart





## Case 5: Recovering from a Damaged Administration File

### Damaged Administration File Recovery Flowchart



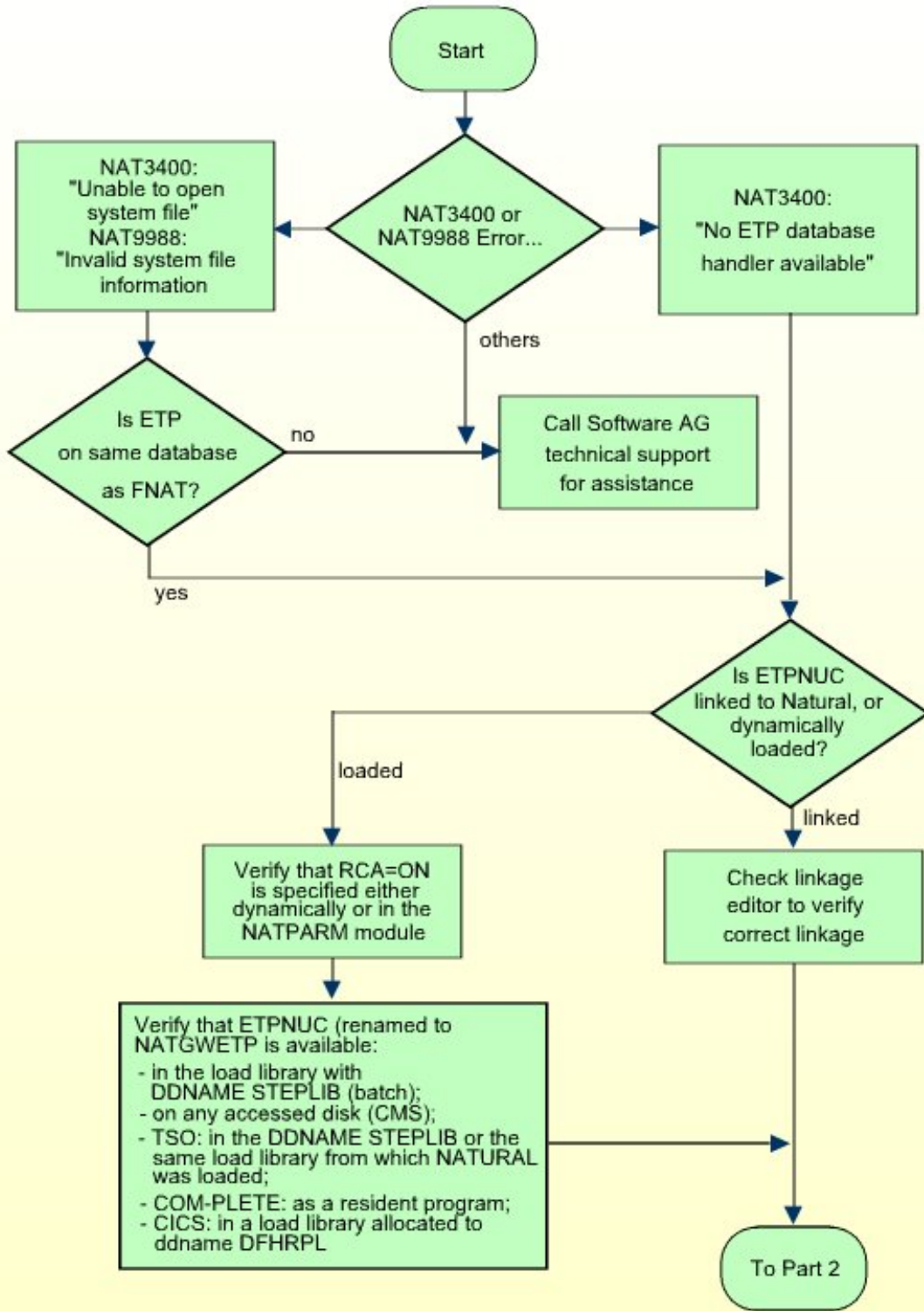
## Correcting NAT3400, NAT9988 and NAT3606 Errors

In some cases, a NAT3400 or NAT9988 error message can occur when ETP is run. The error does not occur, however, when running Natural without ETP (that is, where the ETPNUC module and Natural were not linked, or ETPNUC could not be dynamically loaded). This is usually caused by one of the following:

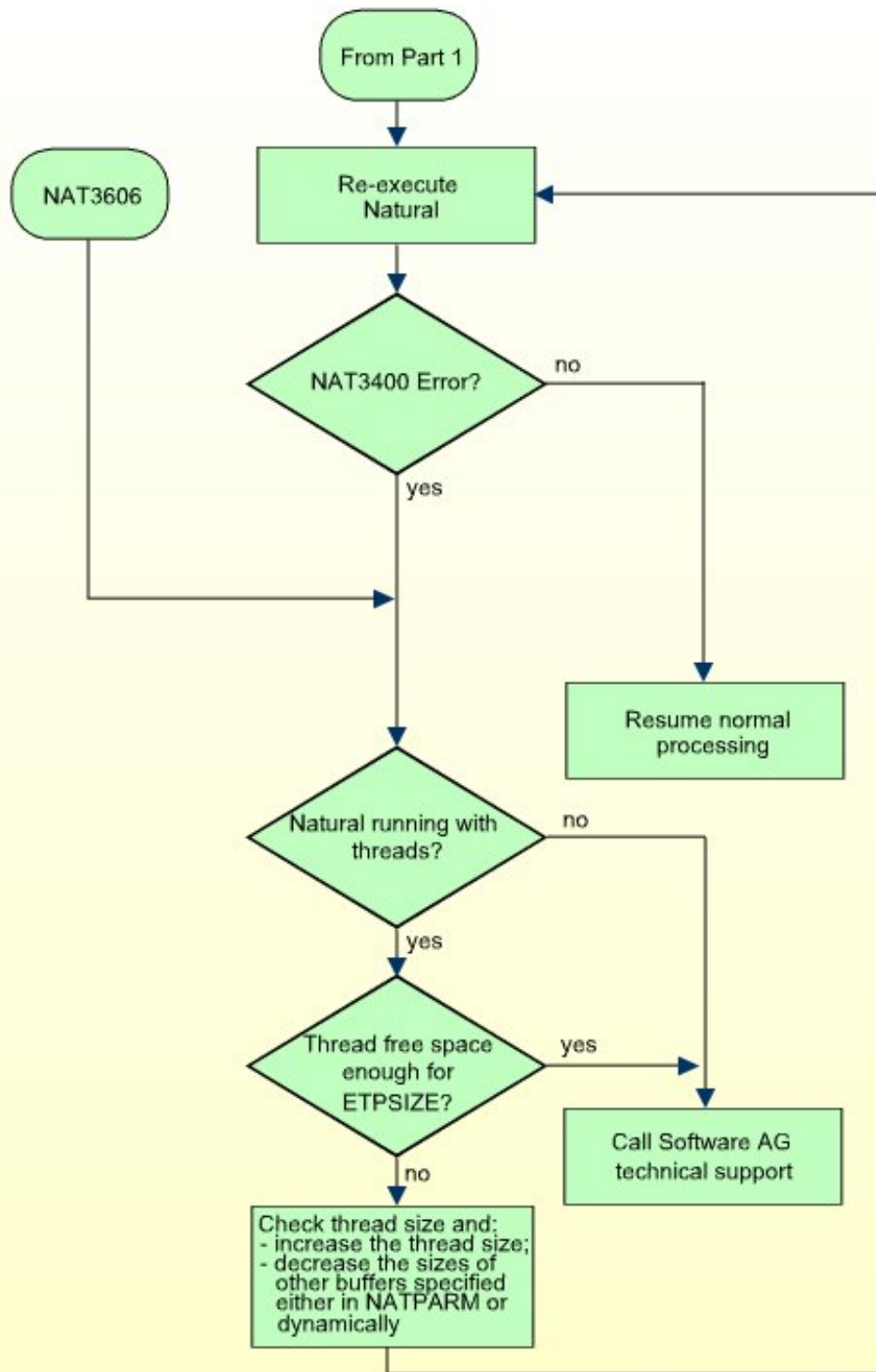
- Incorrectly linking ETP to Natural;
- The required amount of ETPSIZE (approx. 7 KB) is not available.

The figure below shows the general sequence for finding and removing the cause of a NAT3400 or NAT9988 error. See also *ETP Installation*. For NAT3606 error correction, see Part 2 of the figure below.

Correcting a NAT3400/NAT3606 Error (Part 1 of 2)



## Correcting a NAT3400/NAT3606 Error (Part 2 of 2)







# 19

## Correcting Unlogged Transactions

---

▪ Cause .....	146
▪ Remedial Action .....	146

## Cause

---

If replicate files are not being updated with master file changes, the cause could be the failure of ETP to log the master file changes in the log file.

If you look at the log file entries (use the **Display Transactions** function of the ETP maintenance utility) and find that there are no entries in the log file, the cause could be either one of the following:

- Master file not correctly defined;
- ETP NATPARM parameters or Natural start up parameters not correctly specified.

## Remedial Action

---

First, check the master file definition (with the maintenance utility's **Display Master File Definition** function) to ensure that the file has been correctly defined.

If it has, then specific installation parameters should be checked.

The figure below shows the sequence for correcting the logging problem:

### Correcting Unlogged Transactions



If no transactions have been replicated but the log file is not empty, check the replicate file definitions.

# 20 Database Errors during Logging of Master File

## Transactions

---

- Isolating Call Errors with TEST DBLOG ..... 150
- Automatic ETP Back-Out Transaction ..... 150

From the application and user viewpoint, ETP activities during master file updating are absolutely transparent; however, errors resulting from database calls issued by ETP are reported as though the error had occurred for the application or user.

## Isolating Call Errors with TEST DBLOG

---

If you suspect that an error occurred for one of the additional database calls issued by ETP, use `TEST DBLOG` to track down the error. More specifically, use code `S` (snapshot) of `TEST DBLOG` to pinpoint the error. You should set up `TEST DBLOG` before you invoke the program or application for which the error occurs. `TEST DBLOG` can also be used in batch mode.

The following sample input could be used to identify the source of a response code 17, using snapshot:

```
LOGON MYLIB      Log on to the application library.
TEST DBLOG ?    Issue the TEST DBLOG command.
S,,,,,,,,,17,9999  Input for TEST DBLOG.
MYPROG          The application program during which the error occurred.
```

Now, the Adabas call that is shown is the one that received a response code 17.

Using `TEST DBLOG` in this way may also help to identify errors that occur while running the ETP maintenance facility; however, almost all errors that occur during maintenance utility execution provide the information listed above to the ETP administrator.

## Automatic ETP Back-Out Transaction

---

In addition to supplying the information described above for an error during logging operations, ETP also issues a Back out Transaction (BT) if a non-zero response code is returned for an ETP-issued rather than the original call. A BT is also issued if the original call results in a NAT3600 - NAT3624 error.

# 21 Reporting ETP Errors

---

If you find an error that occurs:

- during execution of the replication task;
- while using the ETP maintenance utility;
- while running a Natural application that uses a database defined as an ETP database by an NTDB macro call.

- and it is not possible to determine and correct the error by, for example, following the procedures described in the ETP documentation or by using the information provided in the error messages (if any were issued), you should provide Software AG support personnel with the following information:

- The operating and TP system on which the error occurred;
- The ETP, Natural and Adabas release and version;
- If installed, the Natural Security release and version;
- Printout of the error information provided by ETP (for example, the report from the replication task, a copy of the screen display);
- A list of INPLs or zaps applied to ETP;
- If the error occurred during access to a replicate file's database, the operating system and Adabas release and version of that database.
- If requested by Software AG, a list of your master and replicate file definitions as obtained by means of the `CHECK DEFINITIONS` direct command.





# 22 ETP CICS Interface - Overview

---

This documentation describes Software AG's Entire Transaction Propagator CICS Interface (ETC) for installing and running the **Entire Transaction Propagator** (ETP) with CICS.

ETP allows Adabas users to have duplicate, or replicate, database files in a single database or distributed network. The copies can be distributed throughout a network to provide quick, economical access at user locations.

This documentation is intended for those who are planning for distributed database processing, are looking for a solution that combines databases into a central database resource, or are installing or already using a distributed Adabas/Net-work system.

It covers the following topics:

- **What's New** Describes the new features of the current release.
- **ETC Installation and Operation** Provides an overview of ETC operation, describes the prerequisite requirements and ETC release contents and gives general information on ETC installation.
- **ETC Installation Procedure** Describes the ETC installation for CICS-related operating systems.

For a list of the ABEND codes that can occur during the operation of the Entire Transaction Propagator CICS Interface, refer to the *Entire Transaction Propagator Abend Codes* in the *Natural Messages and Codes* documentation.



# 23

## What's New with ETC Version 1.5.2

---

- New Features, Enhancements and Corrections ..... 156
- Changes Planned for the Next Major Release of ETC ..... 156
- Operating/Teleprocessing System and Software Required ..... 157
- Restrictions ..... 157

## New Features, Enhancements and Corrections

---

This section describes the new features, enhancements and corrections that apply to the Version 1.5.2 of the Entire Transaction Propagator Interface for CICS (ETC):

- **Error Corrections:**  
ETC Version 1.5.2 contains all Zaps applied to ETC Version 1.5.1 as error corrections.
- **Database IDs:**  
ETC now supports DBIDs in the range of 1 to 65535. However, the value of 255 remains excluded.
- **Databases Defined in ETCPARM:**  
The number of databases that can be defined in the parameter module ETCPARM has been increased to 512.
- **PUSERTO Defined in ETCPARM:**  
The maximum value for the timeout of a PUSER slot has been increased to 570. The default value for this parameter has been changed to 30.
- **New Product Documentation Medium:**  
The following changes and enhancements apply to the ETP product documentation:
  - The ETP product documentation is supplied online on CD-ROM as it is common practice for all Natural products.
  - Hardcopies of the documentation can be printed out from the PDF files that come with the documentation.

For information about online documentation usage, see Documentation-Related Frequently Asked Questions (FAQ) on the Natural CD-ROM overview page.

## Changes Planned for the Next Major Release of ETC

---

With the next major release of ETC (following Version 1.5), the support of the following features will be discontinued:

- Format buffer optimization (enabled by the parameter `FBOPT` in the `ETCPARM` macro).
- Adabas Support for SAP R/2 Application (enabled by the parameter `SAP` in the `ETCPARM` macro).

## Operating/Teleprocessing System and Software Required

---

Entire Transaction Propagator Interface for CICS (ETC) Version 1.5.2 requires that Entire Transaction Propagator (ETP) Version 1.5.2 and Natural Version 4.2 is installed.

The prerequisite operating/teleprocessing systems and required versions of other Software AG products are to be found in the current *Natural Release Notes*.

## Restrictions

---

The Entire Transaction Propagator Interface for CICS (ETC) cannot be used to log Adabas database transactions issued by the Natural Development Server CICS Adapter.



# 24 ETC Installation and Operation

---

- Overview on ETC Operation ..... 160
- Prerequisite Requirements and ETC Release Contents ..... 162
- General Information on ETC Installation ..... 162

The Entire Transaction Propagator Interface for CICS (product code: ETC) is a separate Software AG Selectable Unit to support the operation of the Entire Transaction Propagator (product code: ETP) for 3GL programs running in a CICS/TS or CICS/VSE environment. ETC is *not* needed when running Natural application programs alone with CICS. ETC is provided in your installation library only if the ETC selectable unit has been ordered for your site.

For a description of the ETC ABEND codes that can occur when ETC is active, refer to *ETP Abend Codes* (in the *Natural Messages and Codes* documentation).

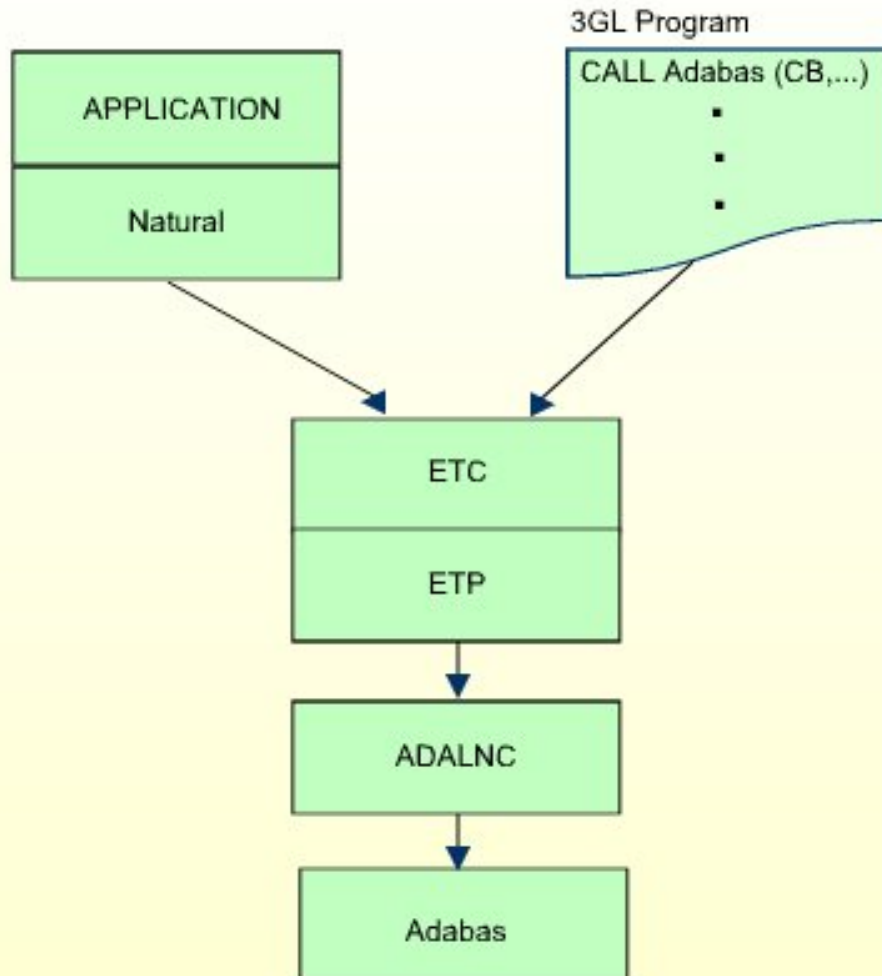
## Overview on ETC Operation

---

The figure below shows how 3GL user application programs, Natural and ETC/ETP work together to use ETP services for logging Adabas database transactions.



## ETP Interface for CICS (ETC)



## Prerequisite Requirements and ETC Release Contents

---

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

### Operating System and Software Requirements

ETP Version 1.5.2 and a CICS installation is required to install ETC. The prerequisite versions of

- z/OS and CICS/TS;
- or z/VSE and CICS/VSE;
- other Software AG products

are listed in the current Natural Release Notes for Mainframes.

### ETC Distribution Tape

The ETC distribution tape contains the following datasets:

Dataset Name:	Description:
ETCvrs.LIBR	On z/VSE system tapes only. z/VSE LIBR backup format of the ETC product sub-library. This sub-library also contains the assembler macro ETCPARM for parameter module generation and an example for calling the ETCPARM macro.
ETCvrs.LIBJ	On z/VSE system tapes only. Example installation jobs.
ETCvrs.LOAD	On z/OS system tapes only. z/OS load library.
ETCvrs.SRCE	On z/OS system tapes only. Assembler macro ETCPARM for parameter module generation and an example for calling the ETCPARM macro.

- where *vrs* is the respective ETC version/release/SM. For more information, see the *Report of Tape Creation* delivered with the distribution tape.

## General Information on ETC Installation

---

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

The Entire Transaction Propagator Interface for CICS (ETC) provides the functionality for Natural as well as 3GL applications running with ETP in a CICS environment. When running only Natural with ETP and CICS, however, only ETP is required.

When running ETC, the following restrictions apply;

- No changes can be made to the database ID or file number in the Adabas Control Block by Adabas or ADALNC user exits;
- You cannot specify a database as an ETP database in the Natural NATPARM file using the NTDB macro or dynamically at Natural startup; Natural calls to the Adabas database are also trapped by ETC, which can cause unpredictable results when processed twice by ETP.
- ETC may be linked with any combination of RMODE and AMODE values. You must not link ETC as a reentrant module.
- ETC expects that the Adabas parameter list is passed in the CICS TWA; Adabas parameter lists in the COMMAREA are not supported.
- Natural or 3GL applications running with ETC cannot run in a z/OS Parallel Sysplex environment where z/OS image switching is to be supported. Set the Natural profile parameter ADAMODE (see the *Parameter Reference* documentation) to 0 or 3 to guarantee that Adabas calls issued by the user application are correctly processed. CICS Plex support is limited to a single z/OS image.
- ETC cannot be used to log Adabas database transactions issued by the Natural Development Server CICS Adapter.
- ETC cannot run in a threadsafe CICS environment, that is, CONCURRENCY(QUASIRENT) must be set in the CICS program definition. Using CONCURRENCY(THREADSAFE) will lead to unpredictable results.
- ETC cannot handle the new control block format as available with Adabas Version 8; if such a control block is encountered, an error is returned to the application that issued the database call.

If the Adabas Bridge for VSAM is installed, AVBUSER MODE=ET should be specified.

Proceed with *ETC Installation Procedure*.

---

# 25 ETC Installation Procedure

---

- Loading ETC from the Installation Tape onto the Disk ..... 166
- Defining the ETC Environment with the ETCPARAM Macro ..... 169
- Example of the ETCPARAM Macro ..... 174
- Assembling ETCPARAM and Linking ETC ..... 174

This chapter describes the ETC installation for CICS-related operating systems.

## Loading ETC from the Installation Tape onto the Disk

---

The steps for loading ETC from the installation tape are operating-system-dependent.

- [Loading the Installation Tape for z/OS](#)
- [Loading the Installation Tape for z/VSE](#)

### 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 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 conform 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.

**Defining the ETC Environment with the ETCPARM Macro**

Create the ETC parameter module ETCPARM by coding the ETCPARM macro with appropriate parameters in an assembly file. The member SAMPLE contains an example which you should change, according to your installation requirements. The ETCPARM macro call begins with the specification of one or more database IDs as positional parameters, followed by optional keyword parameters.

The following ETCPARM parameters are available:

*dbid* | ADANAME | ADMIN | AMODE31 | ASYNC | FBOPT | PUSERS | PUSERTO | SAP | STCK | TIMEOUT | TRNAME

**dbid - Define Database as ETP Database**

```
ETCPARM dbid
```

Specify one or more database IDs that are to be defined as ETP databases.

```
ETCPARM dbid,dbid,...
```

Multiple database IDs must be separated by commas.

```
ETCPARM *
```

To define all databases as ETP databases, specify "\*" as the only positional parameter. This parameter must be specified. Valid database IDs are 1 - 65535, except 255.

```
ETCPARM (fromdbid,todbid)
```

When the format (*from dbid, to dbid*) is specified, the first database ID *from dbid* is translated into the second *to dbid* if the *from dbid* is encountered in an Adabas control block. The first database ID *from dbid* in this configuration can be zero (0).

Multiple pairs of the format (*from dbid, to dbid*) must be separated by commas. This format can be mixed with the format that specifies *dbid* only. Up to 512 of these pairs and/or *dbids* may be specified.

To limit overhead, Software AG strongly recommends specifying only those databases containing a master file defined in the administration file (see ADMIN parameter, below). This enables a function similar to that provided by the Natural NTDB macro.

### ADANAME - Specify Routine for Handling Adabas Calls

Syntax:

```
ADANAME=routinename
```

- where *routine*name is the name of the routine (default is ADALNC) to which control is passed for handling Adabas calls.

### ADMIN - Specify Database ID and File Number of Administration File

Syntax:

```
ADMIN=(dbid, file [, <password [, ciphcode]])
```

Specify the database ID and file number of the administration file.

If required, the administration file's password and/or cipher code must also be specified.

### AMODE31 - Enable/Disable 31-Bit Addressing Mode

Syntax:

```
AMODE31=value
```

Possible Values:

AMODE31=NO	Setting AMODE31=NO prevents ETC from switching to 31-bit addressing mode, even if such addressing is possible.
AMODE31=YES	This setting (default) allows the switch to 31-bit mode, if possible, and permits acquiring storage above the 16-MB storage line.

Specify NO only if any of the following are true:

- An application passes 24-bit addresses with the high-order (leftmost) byte is neither X'00' nor X'80'.
- An ADALNC module is being used which has been assembled and linked in 24-bit mode.



**Note:** Reassembling and relinking in 31-bit mode is recommended.

For operating-system-specific information, see the *Adabas Installation Manual*.

Generally, COBOL compilers generate correct 24- and 31-bit addresses.

### ASYNCH - Specify Routine Starting Async Task

Syntax:

```
ASYNCH=asynchname
```

- where *asynchname* is the name of a routine that starts an asynchronous task. The routine is addressed by a V-type address constant generated in the macro expansion, and must follow standard linking conventions.

If ASYNCH is not specified and a master file definition requires the starting of an asynchronous task following and end-of-transaction (ET), an error occurs.

### FBOPT - Provide Format Buffer Optimization

Syntax:

```
FBOPT=(ALL,minfbl,num_entry[,timeout])
```

```
FBOPT=(SAP,minfbl,num_entry[,timeout])
```

```
FBOPT=NO
```

Provide Format Buffer optimization for large buffers if GFIDs are used extensively and Adabas has been tuned so that almost no overwrites occur. Optimization is performed only if the database calls are routed through ETP - that is, the database is defined as an ETP database in the ETC Parm macro invocation (see above).

Possible Values:

Value	Explanation
SAP	If the SAP R/2 application system is installed and optimization is wanted for SAP call only, specify SAP as the first subparameter.
ALL	For optimization of calls, specify ALL.
NO	If you do not use the SAP R/2 application system, specify FBOPT=NO (the default value).
<i>minfbl</i>	If either SAP or ALL are specified, the <i>minfbl</i> value specifies the minimum Format Buffer length and can be any value in the range of 16-1024.
<i>num entry</i>	If either SAP or ALL is specified, the <i>num entry</i> value specifies the number of 16-byte entries in the table holding global format IDs, and can range from 64 to 32767. If this value is specified too low, optimization becomes ineffective.
<i>timeout</i>	If either SAP or ALL is specified, the <i>timeout</i> value specifies the time period after which the table entries defined by the <i>minfbl</i> and <i>num entry</i> that have not been accessed, are deleted.

Value	Explanation
	The value is measured in minutes, and defaults to the value of the <code>TIMEOUT</code> parameter specified below.

### PUSERS - Specify Number of Users for Parallel Adabas Call Execution

Syntax:

```
PUSERS=value
```

`PUSERS` specifies the number of users that can execute Adabas calls in parallel. For each user, a slot of 256 bytes of storage is allocated.

*value* - Valid values are 100 - 99999. The default is 100.

A slot is only used for the time required to process an Adabas call (and all calls that may possibly be issued by ETP). The slot is then marked as being free to be reused by another user. Therefore, the default value of 100 should be sufficient for most installations. If no free slot is available, the task is abended with the ABEND code `ETCB`. In such a case, the value of this parameter should be increased in steps of 100. `PUSERS` limits the number of tasks executing in `ETCNUC` in parallel; it does not limit the number of Adabas or CICS tasks that can be handled by `ETCNUC`.

### PUSERTO - Specify Timeout Before Releasing PUSER Slot

Syntax:

```
PUSERTO=value
```

`PUSERTO` specifies the time, in minutes, after which a `PUSERS` slot is marked as being free if it has not already been released (e.g. as a result of a user ABEND).

*value* - The `PUSERTO` value must be greater than the maximum of the values specified for the `ADARUN` timeout parameters `TT` and `MXTT`. Valid values are 1 - 570. The default is 30.



**Note:** The `PUSERTO` value is specified in minutes, whereas the Adabas time limits are specified in units of 1.048576 seconds.

## SAP - Specify Adabas Support for SAP R/2 Application

Syntax:

```
SAP=value
```

Possible Values:

SAP=YES	Specify YES if Adabas support is required for the SAP R/2 application system.
SAP=NO	If you do not use the SAP R/2 application system, use SAP=NO (the default).



**Note:** SAP R/2 versions that use the direct call interface without using the CICS TWA are not supported. In addition, SAP R/2 users must request a Zap from SAP support that disables SAP R/2 Format Buffer optimization.

## STCK - Optional Routine for Substituting Direct STCK Instruction

Syntax:

```
STCK=routinename
```

*routinename* - Specify the name of the optional routine that substitutes direct Store Clock (STCK) instructions. The routine is addressed by a V-type address constant generated in the macro expansion, and must follow standard linking conventions.

## TIMEOUT - Set Time Before Releasing User Work Storage

Syntax:

```
TIMEOUT=value
```

If a user does not perform any ETP actions during the number of minutes specified by `TIMEOUT`, ETC releases the user's work storage.

*value* - The `TIMEOUT` value must be greater than the maximum of the values specified for the `ADARUN` timeout parameters `TT` and `MXTT`. After any of the Adabas time limits (`TT`, `TNAA`, `TNAE`, `TNAX`) has expired, Adabas issues an implicit Back out Transaction (BT) for all open transactions (`TT` limit) and deletes the related User Queue elements. The default value is 30.



**Note:** The `TIMEOUT` value is specified in minutes, whereas the Adabas time limits are specified in units of 1.048576 seconds.

## TRNAME - Specify Names of CICS Transactions for Further Processing

Syntax:

```
TRNAME=transname
```

```
TRNAME=(transname,...)
```

*transname* - The name(s) of one or more CICS transactions for which further ETP actions should be performed. To reduce overhead, transactions that are not specified by TRNAME are not processed further. By default, all transactions are enabled for ETP processing.

## Example of the ETCPARM Macro

---

The following is an example of the ETCPARM macro:

```
ETCPARM 2,(0,30),1470,ADMIN=(4,397),TIMEOUT=60,TRNAME=MYTRANS
```

- which would set the following definitions:

- Databases 2 and 1470 are defined as ETP databases. If the DBID in an Adabas control block is zero, it is translated to 30 (the actual database 30 is *not* defined as an ETP database, but the master files defined in the administration file must be defined with a DBID of 30, not 0);
- File 397 on database 4 is defined as the administration file;
- User storage for a user that remains inactive for 60 minutes is released;
- The routine for handling Adabas calls is named ADALNC (the default);
- The only transaction allowed to process further is MYTRANS.

## Assembling ETCPARM and Linking ETC

---

This section describes how to perform the operating-system-dependent assembly and linkage to integrate ETC into the run-time library.

- [Assembling ETCPARM and Linking ETC for z/OS](#)
- [Assembling ETCPARM and Linking ETC for z/VSE](#)

## Assembling ETCPARM and Linking ETC for z/OS

- [Using SMA](#)
- [Without Using SMA](#)

### Using SMA

When using SMA to assemble ETC (SMA job I070, step 5311), the source library for the generated members must be:

```
SAGLIB.ETCnnn.SRCE(SAMPLE)
```

The output library must be:

```
SAGLIB.SMALOAD (the name of the resulting module must be ETCPARM)
```

When using SMA to link ETC (SMA job I080, step 5311), the required input and output library and module definitions are:

```
ETPLIB DD DSN=SAGLIB.ETPnnn.LOAD,DISP=SHR
ETCLIB DD DSN=SAGLIB.ETCnnn.LOAD,DISP=SHR
SMALIB DD DSN=SAGLIB.SMALOAD,DISP=SHR
TPSLIB DD DSN=CICS.LOADLIB,DISP=SHR
SYSLMOD DD DSN=cics.loadlib,DISP=SHR
      .
      .
INCLUDE TPSLIB(DFHEAI)
CHANGE ETCNUC(adaname)
INCLUDE ETCLIB(ETCNUC)
INCLUDE SMALIB(ETCPARM) (parameter module)
INCLUDE ETPLIB(ETPNUC)
INCLUDE TPSLIB(DFHEAIO)
ENTRY adaname
NAME adaname(R)
```

- where:

<i>cics.loadlib</i>	Contains the final load module. This library must be concatenated to DD name DFHRPL to permit CICS to load the module and to find it before any other module with the same name.
<i>adaname</i>	Is the name of the final load module. 3GL programs and Natural use this name to perform Adabas calls for Natural. Specify this name in the ADANAME parameter in the Natural parameter module or dynamically at Natural startup.

**Without Using SMA**

When not using SMA, use the following JCL to generate the ETC Parm module and link ETCNUC, ETC Parm and the module ETPNUC into a single module:

```
//
//jobcard

//ALLOC EXEC PGM=IEFBR14
//SYSLIN DD DSN=&&OBJ,SPACE=(TRK,(10,10,2),RLSE),UNIT=SYSDA,
// DISP=(NEW,PASS),
// DCB=(BLKSIZE=3040,LRECL=80,RECFM=FB,BUFNO=1)
//*
//HLASM EXEC PGM=ASMA90,
// PARM='TERM,OBJ,USING(NOLIMIT,MAP,WARN(15))'
//SYSLIB DD DISP=SHR,DSN=SAGLIB.ETCvrs.SRCE
//SYSUT1 DD DSN=&SYSUT1,SPACE=(1024,(120,120),,ROUND),UNIT=VIO,
// DCB=BUFNO=1
//SYSPUNCH DD DUMMY
//SYSTEM DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSLIN DD DSN=&&OBJ(ETCPARM),DISP=(MOD,PASS)
//SYSIN DD DSN=SAGLIB.ETCvrs.SRCE(SAMPLE)
//*
//LKED EXEC PGM=HEWL,
// PARM='XREF,LET,LIST,REUS,NORENT,SIZE=(768K,128K),NCAL',
// REGION=512K,COND=(0,LT)
//SYSLMOD DD DISP=SHR,DSN=cics.loadlib
//SYSLIN DD DDNAME=SYSIN
//SYSUT1 DD DSN=&&AUTO,UNIT=3380,SPACE=(TRK,(100,10)),
// DISP=(NEW,DELETE,DELETE)
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DSN=&&OBJ,DISP=(OLD,PASS)
//CICSLIB DD DSN=cics.syslib,DISP=SHR
//ETPLIB DD DSN=SAGLIB.ETPvrs.LOAD,DISP=SHR
//ETCLIB DD DSN=SAGLIB.ETCvrs.LOAD,DISP=SHR
//SYSIN DD *
INCLUDE CICSLIB(DFHEAI)
CHANGE ETCNUC(adaname)
INCLUDE ETCLIB(ETCNUC)
INCLUDE SYSLIB(ETCPARM)
INCLUDE ETPLIB(ETPNUC)
INCLUDE CICSLIB(DFHEAIO)
ENTRY adaname
MODE RMODE(ANY)
MODE AMODE(31)
NAME
adaname(R)
//
```



- where:

<i>cics.syslib</i>	Contains the modules DFHEAI and DFHEAIO. These modules are required for command-level linkage only.
<i>cics.loadlib</i>	Contains the final load module. This library must be concatenated to DD name DFHRPL to permit CICS to load the module and to find it before any other module with the same name.
<i>adaname</i>	The name of the final load module. 3GL programs and Natural use this name to perform Adabas calls for Natural. Specify this name for the ADANAME parameter in the Natural parameter module or dynamically at Natural startup.

### Assembling ETCPARM and Linking ETC for z/VSE

Using the following job control, generate the ETCPARM module and link ETCNUC, ETCPARM and the ETP module ETPNUC into a single module:

```
// JOB ASSEMBLE
// OPTION CATAL
// LIBDEF *,SEARCH=searchlibs,CATALOG=cics.loadlib
  PHASE adaname,*
  MODE RMODE(ANY),AMODE(31)
  INCLUDE DFHEAI
// EXEC PGM=ASSEMBLY
      COPY ETCPARM
      COPY SAMPLE
/*
  INCLUDE ETCNUC
  INCLUDE ETPNUC
  INCLUDE DFHEAIO
  INCLUDE
  ENTRY ETCNUC
/*
// EXEC LNKEDT
/&
```

- where:

<i>searchlibs</i>	A list of libraries containing the ETP load module and, if command-level linkage is desired, modules DFHEAI and DFHEAIO.
<i>cics.loadlib</i>	Contains the final load module. This library must be concatenated to DD name DFHRPL to permit CICS to load the module and to find it before any other module with the same name.
<i>adaname</i>	Is the name of the final load module. 3GL programs and Natural use this name to perform Adabas calls. Specify this name for the Natural profile parameter ADANAME in the Natural parameter module or dynamically at Natural startup.

## Identifying ETC to CICS

To identify ETC to CICS, specify the following entry in the CICS PPT for the load module that results from the linkage step:

```
DFHPPT TYPE=ENTRY,PROGRAM=adaname,RES=YES,PGMLANG=ASSEMBLER
```

- where *adaname* is the name of the load module after completing the linkage step (see the operating-system-specific installation information, above). It is not necessary to link the resulting module *adaname* to either your 3GL applications, to Natural or to the ADALNC module.

The parameter value

```
EXECKEY(USER)
```

must be set in the CICS program definition.

# 26

## Glossary of ETP Terms

---

The terms defined here are referred to in the ETP documentation. For a more extensive list of terms or for definitions of terms mentioned here that are not included in this glossary, refer to the glossaries of the Adabas manuals listed in the introduction to this manual.

### **Adabas**

Adabas (Adaptable **DA**tA**BA**se **S**ystem)

Software AG's database product. The main parts of an Adabas database are the Data Storage (Data), the Associator for tracking the data, the Work area for performing database operations and holding interim results, Temp for general temporary storage, Sort for required sorting operations, and the Protection Log (PLOG).

### **Adabas Control Block (CB)**

The communication area for performing Adabas calls. This area comprises the fields where user programs specify the command code, Adabas file number, etc.

### **Administration file**

The Adabas file that contains definitions of the ETP master and replicate files and the ETP operating profiles. To run ETP, an administration file must be defined for Natural which contains all information about master and replicate files. This is done as described in the section [Defining the Administration File](#).

Software AG recommends that the administration file be located on any database having an ETP master file.

### **Asynchronous**

An ETP master file and its replicate files are usually not updated simultaneously. The replicate copies are instead resynchronized with the master file sometime after the master file update; in other words, the replicate files are resynchronized *asynchronous* of the master file update.

### **Confirmation file**

The Adabas file that records the state of the replicate file as compared to the master file. At least one confirmation file must be located on every physical database containing replicate files.

### **Database Administrator (DBA)**

Controls and manages the database resources. Tasks include defining database distribution, assigning a structure and resources, creating and maintaining programming and operation standards, ensuring high performance, resolving user problems, defining and teaching user training, controlling database access and security, and planning for growth and the integration of new database resource applications and system upgrades. Also known as the Database Analyst.

### **Database**

An Adabas database comprises Adabas files administered by the Adabas nucleus. The logical components of an Adabas database are the Data, Associator, Work, TEMP and SORT.

### **DBID (database ID)**

The numeric identifier of the physical Adabas database. Each physical database must have a unique identifier. Depending on where it is applied or read, the DBID can be in either decimal, hexadecimal or binary notation. When entered in the ETP maintenance utility menus, the notation is always decimal.

### **Descriptor**

A field in an Adabas file used as a reference. For each descriptor, Adabas creates an inverted list in the Associator. Descriptors make it possible to read a file in logical sequence and to formulate search criteria to find specific records. Descriptors can be defined as unique descriptors, which means that each value of the descriptor may only occur once in a file. It is also possible to define hyperdescriptors, subdescriptors, superdescriptors and phonetic descriptors.

### **Distribution key**

When you define replicate files having only a part (subset) of the records contained in the master file, you must specify a field and value(s) or value range(s) to uniquely identify which master file records the replicate file contains. The record field you specify becomes the distribution key. The distribution key can be any arbitrary Adabas field (not necessarily a descriptor), but cannot be a periodic group, a multiple-value field, or a phonetic descriptor. See also [Key value](#).

### **ET (End Transaction command)**

All application programs that change an ETP master file must use Adabas ET logic. ET logic is denoted by specifying an OP (Open) command to begin and an ET command to end the transaction. ET logic ensures that all „housekeeping“ is done (log data is stored, records are released, etc.). If the following master file transaction fails, Adabas autobackout function can more successfully restore all ETP files to their status before the unsuccessful transaction started.

### **Field, database**

The smallest named unit of data information in the database. A field usually contains a „piece“ of information (last name, employee number, age, or similar). One or more fields, each having its own two-character name, make up a database record. A single Adabas record can contain one or more of the following types of fields:

- A *elementary field* contains one value;
- A *multiple-value field* contains several „stacked“ (from 1 to 65534, depending on the Adabas version and definition of the FDT) values of identical data type, or occurrences, per record.

The „stack“ grows or shrinks as values are added or removed. Multiple-value fields are useful for holding repetitive information within a single record (autos owned, multiple part numbers for a single stock item);

- A *periodic group*, which has the same „stack“ structure as a multiple-value field, but the occurrences in the stack hold their relative positions. Specific positions must be specified for new occurrences, and deleted occurrences leave vacant slots in the stack.

### **Field Definition Table (FDT)**

A table that defines each file's record structure and content. There is one FDT for each database file. FDTs, stored in the Associator's fixed area, have three parts: the first is a list of the file's fields in physical record order, the second part is a „quick index“ to the records in the first part, and the third part defines the files sub/superfields and sub-/super-/hyper- and phonetic descriptors.

### **FNR (file number)**

The physical number of an Adabas file. An ETP master or replicate file is always specified by its Adabas database ID (see **DBID**) and file number.

### **Format Buffer (FB)**

Holds the field definitions (formats and names) to or from which data is written or read in the Adabas database files.

### **Internal Sequence Number (ISN)**

Every Adabas record is assigned an internal sequence number (ISN) to identify the record. Each record keeps its original ISN, regardless of where it is located.

Records in a physical database file have four-byte ISNs ranging from MINISN to MAXISN. In replicated files, a record has the same ISN in all file copies. In partitioned files, the ISN ranges are non-overlapping for each physical file.

### **Inverted list**

A logically sorted list of the ISNs for each field that is defined as a descriptor. The inverted lists are located in the Adabas Associator.

### **Key value**

A specific value/value range or combination of one or more values/value ranges that are in the distribution key field of all the records in a „partial“(subset) replicate file. ETP uses the distribution key and key value to determine whether or not a change to a particular master file must be applied to the replicate file.

### **Log file**

The Adabas file that records the updates made to one or more master files on the same Adabas database. ETP replication tasks look in the log file for change information when they resynchronize replicate files with their master files.

### **Master file**

The Adabas file used as a reference in an ETP replicate file set. Either the master or a replicate file can be read by programs, but updates can only be made to the master file.

**Multiple-value field**

See also [Field](#).

**OP (Open command)**

An Open (OP) command marks the beginning of an ET logic transaction.

**Periodic group**

A group of fields which can have several (0 to 99) occurrences within a record. See also [Field](#).

**Phonetic descriptor**

A phonetic descriptor makes it possible to search for a field value on the basis of the sound of the value; all values that sound similar to the value will be found.

**Physical database**

A physical database identified by its database ID is defined with Adabas utilities. An Adabas nucleus running in an address space allows access to the physical files in the physical database.

**Physical file**

A physical file contains database records. Each physical file is identified by a file number.

**Record Buffer (RB)**

The portion of the calling program's parameter area, called the User Buffer, that contains the data transferred during Adabas read, search, and update operations. When reading data field definitions, Adabas also returns the field definition information in the Record Buffer.

**Replication task**

A transaction that updates a replicate file or files with the changes recorded in the log file of the related master file. Replication tasks are started from within the ETP maintenance utility.

**Synchronous**

A distributed database having absolute data integrity in replicate files must change all other replicate copies at the same time when one copy has been changed. This *synchronous* updating ensures that all copies are consistent at all times. ETP uses asynchronous control of replicate files to reduce the overhead of maintaining replicate files while offering site-defined control of the data integrity. See also [Asynchronous](#).

**Time stamp**

The hexadecimal date/time notation that ETP uses to mark changes and other activity. When entering a time value in the ETP maintenance utility, you can enter the hexadecimal time stamp found in the log file entry.

**User**

A batch or on-line application program that makes Adabas calls.