Editor Buffer Pool

This document describes purpose, use and operation of the Editor Buffer Pool which is an intermediate main storage area used by the Software AG Editor.

The following topics are covered:

- Purpose of the Editor Buffer Pool

- Obtaining Free Blocks

- Initializing the Editor Buffer Pool

- Restarting the Editor Buffer Pool

- Editor Buffer Pool Parameters

- Buffer Pool Initialization for Multi-User Environments

Purpose of the Editor Buffer Pool

The editor buffer pool can be seen as an extension of the editor buffer (`SSIZE`). It is an intermediate main storage area used by the Software AG Editor to maintain its logical files.

A logical file consists of one or more logical records and contains the data of an object (for example, a file member) maintained by the editor. As a user can work with more than one object at the same time, several logical files can exist concurrently for each user.

The number of logical files (as well as the percentage of recovery records in the Editor Work File is defined in the buffer pool parameter macro.

The editor buffer pool can be defined as a local or a global (z/OS and BS2000/OSD only) or an auxiliary (`EDPSIZE`) buffer pool. In multi-user environments (CICS, IMS TM, UTM), the editor buffer pool is shared by all editor users of either the same region (local pool) or more than one region (global pool). Under CMS, the buffer pool is always a local one. A global buffer pool cannot be shared by Com-plete and other regions due to the separate SD editor work file under Com-plete.

The editor buffer pool contains various control tables and a number of data blocks:

Area	Size
Main control block	500 bytes
Logical file table	20 bytes per logical file
Work file table	4 bytes per record
Recovery file table	16 bytes per record
Buffer pool block table	28 bytes per block
Buffer pool blocks	see text below

As the size of a buffer pool block is equal to the size of a work file record, one buffer pool block can contain one logical file record.

The buffer pool is initialized by the first editor user. During warm start buffer pool initialization, all recovery records are checked to build the recovery file table.

Several functions are provided to access the buffer pool (for example, functions to allocate, read, write or delete a record).

Obtaining Free Blocks

If the buffer pool becomes full, buffer pool blocks have to be moved to an external dataset, the editor work file, to obtain free blocks.

In such a situation, the editor checks all logical files for their timeout value and deletes any logical file which has not been accessed within the specified time. This means that all its buffer pool blocks and work file records are freed, and the logical file is lost.

If there is still no buffer pool block available, the editor moves the oldest block to the work file, according to the specified timeout parameter values (see the *Generation Parameters* function of the *SYSEDT Utility* in the Natural *Utilities* documentation).

Initializing the Editor Buffer Pool

An uninitialized editor buffer pool is initialized when the Software AG editor is called for the first time. Then the various control blocks are created. There are two different modes of buffer pool and work file initialization: "cold start" and "warm start".

Buffer Pool Cold Start

A buffer pool cold start can be triggered by the editor buffer pool subparameter `COLD` or by the Editor Buffer Pool Services utility `SYSEDT` or automatically (if the editor work file is unformatted).

During a buffer pool cold start, the values of the editor buffer pool parameter `EDBP` or the corresponding macro `NTEDBP` are stored into the work file control record and all work file recovery records are cleared.

Buffer Pool Warm Start

During a buffer pool warm start, the buffer pool parameters are read from the work file control record and all work file recovery records are read to build the recovery file table in the buffer pool.

Restarting the Editor Buffer Pool

The `SYSEDT` utility can be used to terminate the editor buffer pool, that is, to set it to the unitialized state. This avoids the restart of the TP system or of the global buffer pool.

If `SYSEDT` is not available due to buffer-pool problems, the program `BPTERM` can be used to terminate the buffer pool.

Important:
All Natural sessions must be restored if you want them to use the editor after buffer-pool restart.

Editor Buffer Pool Parameters

The editor buffer pool parameter `EDBP` or the corresponding macro `NTEDBP` in the Natural parameter module `NATPARM` is required to define parameters for the operation of the editor buffer pool.

When the editor work file is formatted, these parameters are stored into the work file control record while all other records are cleared. Thus, reformatting a work file that has been previously used, means that all editor checkpoint and recovery information is lost.

Some of these parameters can be modified dynamically during execution of the buffer pool by using the Editor Buffer Pool Services utility `SYSEDT`.

Buffer Pool Initialization for Multi-User Environments

During the buffer pool initialization, all recovery records are read from the editor work file. Therefore, the first users have to wait for a long time or even receive a timeout message until the editor buffer pool initialization is finished.

For this reason, a special Natural program has been supplied to trigger the buffer pool initialization before the first user becomes active. This program can be activated either during the startup of the TP monitor, or by a batch job if a global buffer pool is used.

The session must then be started with the session parameter:

```
STACK=(LOGON SYSEDT,user,password;BPINIT;FIN)
```

Under CICS: If the session runs asynchronously, `SENDER=CONSOLE` must be specified to obtain any error messages issued during initialization. The source program `FRONTPLT` is supplied as a sample program to show you how to start an asynchronous Natural session during CICS startup via `PLTPI`.