



# NATURAL

---

**Natural**  
Operations  
Version 5.1.1 for Windows



This document applies to Natural Version 5.1.1 for Windows and to all subsequent releases. Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

© June 2002, Software AG  
All rights reserved

Software AG and/or all Software AG products are either trademarks or registered trademarks of Software AG. Other products and company names mentioned herein may be the trademarks of their respective owners.

# Table of Contents

<b>Natural for Windows - Operations Overview</b>	1
Natural for Windows - Operations Overview	1
<b>Operations Environment</b>	3
Operations Environment	3
<b>Host Communication and System File Simulation</b>	4
Host Communication and System File Simulation	4
Host Communication	4
System File Simulation	4
System Files FNAT and FUSER	6
<b>Configuration Files</b>	8
Configuration Files	8
Local Configuration File - Natural.INI	8
Buffer Pool Assignments	8
Installation Assignments	9
Global Configuration File - NATCONF.CFG	10
DBMS Assignments	10
Dictionary Server Assignments	12
Printer Profiles	13
System File Assignments	16
<b>Work Files</b>	17
Work Files	17
Defining Work Files	17
Work Files with Environment Variables	18
Work Files with User Exit	18
Work File Formats	18
Binary Format	19
ASCII Format	19
ENTIRE CONNECTION Format	19
PORTABLE Format	19
UNFORMATTED Format	20
Special Considerations on Work Files with Extension NCD	21
<b>Natural Buffer Pool</b>	22
Natural Buffer Pool	22
Buffer Pool under Windows	22
Setting up the Buffer Pool under Windows NT	23
Windows NT Service	23
Natural Buffer Pool Service Commands	23
NATBPMON Utility	29
How to Invoke NATBPMON	29
NATBPMON Commands	30
CLEAR	30
CORPSES	30
DELETE	30
DIR	31
DUMP	32
EXIT	32
HELP	33
PARAM	33
QUIT	33
SHUTDOWN	33
STATUS	34
WHO	36
ZERO	36

Buffer Pool Trouble Shooting . . . . .	37
Problem . . . . .	37
Examples . . . . .	37
Solution . . . . .	37
<b>Invoking Natural Subprograms from 3GL Programs . . . . .</b>	<b>38</b>
Invoking Natural Subprograms from 3GL Programs . . . . .	38
Passing Parameters from the 3GL Program to the Subprogram . . . . .	38
Example of Invoking a Natural Subprogram from a 3GL Program . . . . .	38
<b>Natural in Batch Mode . . . . .</b>	<b>39</b>
Natural in Batch Mode . . . . .	39
What is Batch Mode? . . . . .	39
Batch Mode Detection . . . . .	39
Batch Mode Restrictions . . . . .	40
Real Batch Mode . . . . .	40
Input and Output Channels . . . . .	40
CMPRTnn Specifications in Batch Mode . . . . .	40
Sample Session for Batch Mode . . . . .	41
Natural Commands - CMSYNIN=batch.cmd: . . . . .	41
Natural Input Data - CMOBJIN=batch.inp: . . . . .	41
Natural Program RECCONT in Library SYSEXBAT: . . . . .	41
Natural Command Line: . . . . .	41
Contents of Batch Output File - CMPRINT=batch.out - after Execution: . . . . .	41
Hints for Using Natural Maps and Dialogs in Batch Mode . . . . .	43
Example for Event AFTER-OPEN: . . . . .	43
Example for Event CLOSE: . . . . .	43
<b>Natural Output Window Information . . . . .</b>	<b>44</b>
Natural Output Window Information . . . . .	44
Output Window Features . . . . .	44
Output Window Profiling . . . . .	44
Additional Information on Fonts . . . . .	47
<b>Natural Startup Errors . . . . .</b>	<b>48</b>
Natural Startup Errors . . . . .	48
<b>Issuing System Commands from a Natural Program . . . . .</b>	<b>50</b>
Issuing System Commands from a Natural Program . . . . .	50
Parameter Options . . . . .	51
<b>Natural Profile Parameters . . . . .</b>	<b>53</b>
Natural Profile Parameters . . . . .	53
Parameter Hierarchy . . . . .	53
Runtime Assignment of Parameter Values . . . . .	54
Dynamic Assignment of Parameter Values . . . . .	55
Static Assignment of Parameter Values . . . . .	55
<b>Profile Parameters Grouped by Function . . . . .</b>	<b>56</b>
Profile Parameters Grouped by Function . . . . .	56
Database Management . . . . .	56
General Parameters . . . . .	56
Adabas Specific . . . . .	57
Administrator DBMS Assignment . . . . .	57
User DBMS Assignment . . . . .	57
Natural Execution Configuration . . . . .	57
Batch Mode . . . . .	58
Buffer Sizes . . . . .	58
Character Assignments . . . . .	59
Command Execution . . . . .	59
Date Representation . . . . .	59
Device/Report Assignments . . . . .	59
Error Handling . . . . .	61

Field Appearance . . . . .	61
Limits . . . . .	62
Program Loading and Deletion . . . . .	62
Regional Settings . . . . .	62
Report Parameters . . . . .	62
Steplibs . . . . .	63
System Files . . . . .	63
System Variables . . . . .	63
Workfiles . . . . .	64
Natural Development Environment . . . . .	64
Compiler Options . . . . .	64
Remote Debugging . . . . .	65
Product Configuration . . . . .	65
Entire Transaction Propagator . . . . .	65
Entire System Server Interface . . . . .	66
Client/Server . . . . .	66
DCOM Support . . . . .	66
Remote Dictionary Access . . . . .	66
Remote Procedure Call . . . . .	66
<b>Natural Configuration Utility . . . . .</b>	<b>68</b>
Natural Configuration Utility . . . . .	68
Invoking the Natural Configuration Utility . . . . .	68
Menu Bar Commands . . . . .	68
File Commands . . . . .	68
Edit Commands . . . . .	69
Help . . . . .	69
Search Function . . . . .	70
Changing Parameter Settings . . . . .	70
Performing Configuration Utility Functions from the OS Command Prompt . . . . .	70
<b>Providing Natural Applications . . . . .</b>	<b>72</b>
Providing Natural Applications . . . . .	72
SYSPAUL Utility . . . . .	72
Invoking SYSPAUL . . . . .	72
Define Application Description . . . . .	73
Description . . . . .	73
Application Files . . . . .	75
Instructions . . . . .	76
Options . . . . .	76
Further Components of the Define Application Description Window . . . . .	76
Adding and Modifying Application Files . . . . .	77
Adding and Modifying Unload Instructions for Natural Objects . . . . .	77
Adding and Modifying Unload Instructions for Non-Natural Objects . . . . .	83
Unloading Natural Applications . . . . .	84
Packaging Natural Applications . . . . .	84
Unload Application . . . . .	85
The Unload Application Dialog Box . . . . .	85
Package Application . . . . .	87
Package Application Dialog Box . . . . .	87
Load Application File . . . . .	89
General Load Options . . . . .	89
Report . . . . .	90
Specify a Selection . . . . .	90
Scan Application File . . . . .	93
General Scan Options . . . . .	93
Report . . . . .	93
Specify a Selection . . . . .	94

<b>Natural Runtime Version</b>	97
Natural Runtime Version	97
General Information on Natural Runtime	97
Runtime Version Differences	97
System Commands	97
Editors	97
Natural Utilities	97
Porting Procedure Overview	98
Step 1: Packaging the Application	98
Creating a Collecting Directory	98
Customizing the Global Configuration File	98
Customizing the Natural Parameter File	99
Copying Natural Code to the Collecting Directory	100
Completing the Package	101
Step 2: Installing the Runtime Version	102
Step 3: Installing the Application to the Runtime Version	103
Installing the Configuration Files	103
Installing the Natural code	103
Step 4: Starting the Application	105
Runtime Startup Services	106
Runtime Startup Service Commands	106
<b>Support of Language-Specific Characters</b>	109
Support of Language-Specific Characters	109
Why is the Support of Language-Specific Characters Important?	109
Upper-/Lower-Case Conversion	109
Identifier Checking	109
Character Classification	109
Configuration File NATCONV.INI	109
Sample NATCONV.INI File	111
<b>Graphics Interface</b>	119
Graphics Interface	119
Generating Excel Graphics from Natural	119
General Information	119
Prerequisites for Generating Excel Graphics	119
Further Examples	119
CALL 'GRAPHICS'	120
CALL 'DRAW'	121
Operands of the CALL 'DRAW' Statement	121
'plot-name'	121
Mode of Charts - 'SINGLE', 'OVERLAY', 'MAIN'	122
Types of Charts	123
CALL 'PLOT'	131
'plot-name'	131
'y-text'	131
'MAX', 'MIN', 'NMIN', 'COUNT', 'NCOUNT', 'AVER', 'NAVER', 'SUM', 'TOTAL'	131
y-value	131
'x-text'	131
x-value	131
'ciomp-text'	131
comp-value	132
<b>Natural and Entire Access</b>	134
Natural and Entire Access	134
Purpose of Entire Access	134
Entire Access under Windows and Windows NT	134
Modifying the Global Configuration File	135
Connecting Databases	135

Making Entire Access Operational . . . . . 136  
Accessing Database Tables - the Natural DDM . . . . . 136



# Natural for Windows - Operations Overview

for Windows is a software product that enables you to fully integrate your personal computer (PC) into the information processing environment of your organization.

Natural for Windows provides a complete application development environment. It can be used to develop efficient user-oriented Natural applications which are fully source-code compatible with Natural on the mainframe computer and other platforms, for example, UNIX or OpenVMS. In addition, it can also be used in a stand-alone PC environment.

Natural for Windows allows the use of existing Natural mainframe applications on your PC, enabling the PC to be used as an application workstation.

- Operations Environment
- Profile Parameter Usage
- Profile Parameters Grouped by Function
- Natural Configuration Utility
- Providing Natural Applications
- Natural Runtime Version
- Support of Language-Specific Characters
- Graphics Interface
- Natural and Entire Access

**Note:**

We would like to remind our customers who have purchased the Natural Runtime version that the Natural development tools are not included in the Natural Runtime version. In addition, not all Natural system commands are supported in the Natural Runtime version.

**Platform-Specific Information**

Wherever necessary, platform-specific information in the documentation is identified by the following terms:

**Mainframe** Refers to the operating systems OS/390, VSE/ESA, VM/CMS and BS2000/OSD, as well as all TP monitors supported by Natural under these operating systems.

**OpenVMS** Refers to the OpenVMS operating system.

**UNIX** Refers to all UNIX systems supported by Natural.

**Windows** Refers to the following operating systems:

**In a Natural development environment:**

- Microsoft Windows NT 4.0 SP6
- Microsoft Windows 2000 Professional SP1
- Microsoft Windows 2000 Server, Advanced Server

**In a Natural run-time environment:**

- all platforms above plus:
- Microsoft Windows 98
- Microsoft Windows ME

**OS/400** Refers to the OS/400 operating system running on AS/400 and iSeries 400 machines. See the documentation provided on the Natural for OS/400 product CD-ROM.

# Operations Environment

This section describes the Natural operations environment for Windows and Windows NT. Natural takes full advantage of PC hardware flexibility.

- Host Communication and System File Simulation
- Configuration Files
- Work Files
- Natural Buffer Pool
- Invoking Natural Subprograms from 3GL Programs
- Natural in Batch Mode
- Natural Output Window Information
- Natural Exit Codes
- Issuing System Commands from a Natural Program

With Natural, you can access and update data stored in various types of databases. The Natural language statements used to access this database are the same as those used on mainframe and UNIX computers. Each Natural statement performs the same function regardless of the database management system or access method used.

Natural also provides the ability to include programs written in other programming languages (for example, C, PASCAL or Assembler) within Natural applications. In such cases, the respective compiler and linker are required. For additional information, see the CALL statement in the Natural Statements documentation.

The way in which Natural report pages are scrolled or sized and online terminal commands are used is quite unlike those on the mainframe. Input/output fields can be checked instantaneously, regarding valid range, correct type, or edit mask. Graphical user interface (GUI) elements such as buttons, drop-down menus or bitmaps can be used to enhance the user interface of a Natural application.

# Host Communication and System File Simulation

- Host Communication
  - System File Simulation
- 

## Host Communication

Communication with a mainframe computer (host) where Natural and Adabas are installed is possible using Natural.

This enables applications to be exchanged without modification, which is achieved by using the SYSOBJH Utility.

## System File Simulation

- System Files FNAT and FUSER

To ensure database independence, Natural does not store objects in an Adabas system file. The system file is simulated by a structure of directories on the disk where Natural is installed.

Since the operating system is used instead of a database system file, you do not need Adabas to run Natural.

When installing Natural on your PC or workstation, the path of your Natural root directory is specified in a configuration file.

The Natural libraries are then created as subdirectories below the Natural root directory with the same name as the libraries. The command LOGON is therefore similar to the Change Directory command.

Natural objects are stored as operating-system files in the appropriate directory. The name used for the file takes the form:

**XXXXXXXX.NKt** where:

XXXXXXXX	is the name of the object
K	is "S" (for source files), "G" (for generated programs) or "R" (for private resources)
t	is the type of the object; for valid values, see the list below.

### Note:

The file name is not always identical to the object name. Both the actual object name and the corresponding internal object name are documented in the FILEDIR.SAG file.

For example, the source program TESTPROG would be stored as file TESTPROG.NSP, while the generated code for the map TESTMAP would be stored as file TESTMAP.NGM.

The following object types and the respective letters and numbers are to be used for the extensions available:

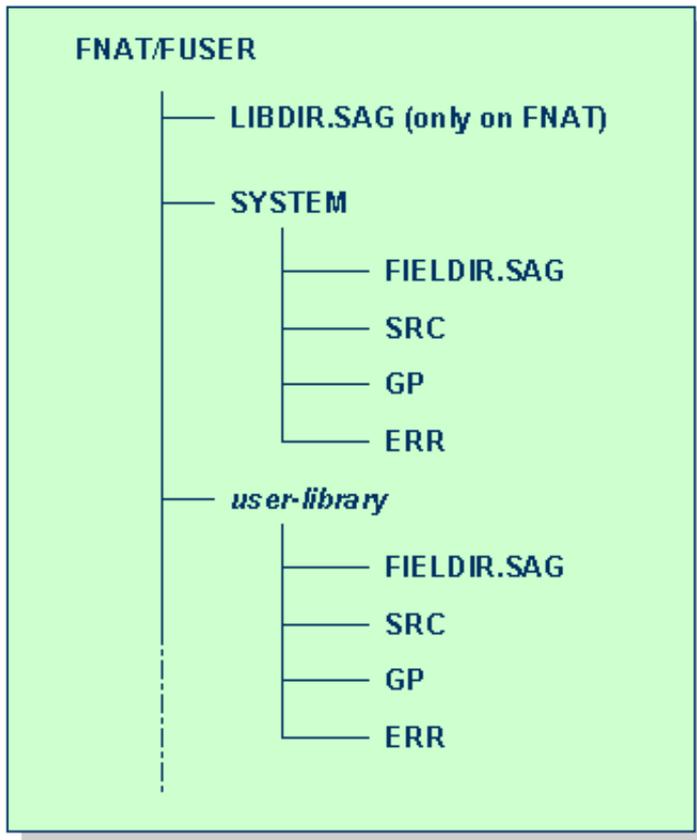
<b>P</b>	Program
<b>3</b>	Dialog
<b>6</b>	Object view
<b>N</b>	Subprogram
<b>S</b>	Subroutine
<b>C</b>	Copycode
<b>H</b>	Helproutine
<b>T</b>	Text
<b>M</b>	Map
<b>L</b>	Local data area
<b>G</b>	Global data area
<b>A</b>	Parameter data area
<b>4</b>	Class
<b>5</b>	Command processors

The name of a Natural object can consist of the following characters:

<b>Character</b>	<b>Explanation</b>
<b>A- Z</b>	alphabetical characters
<b>0 - 9</b>	numeric characters
<b>-</b>	hyphen
<b>@</b>	"at" sign
<b>_</b>	underline
<b>/</b>	slash
<b>\$</b>	dollar sign
<b>&amp;</b>	ampersand (not allowed as first character)
<b>#</b>	hash/number sign
<b>+</b>	plus sign (only allowed as first character)

## System Files FNAT and FUSER

The Natural system files FNAT (for system programs) and FUSER (for user-written programs) can be located in different subdirectories. They assume the following directory structure:



This directory structure is generated during the installation of Natural.

The file LIBDIR.SAG contains information on all further installed Software AG products using Natural. This information can be displayed by using the SYSPROD system command; see the Natural User's Guide for Windows.

The directories representing the system library SYSTEM and each "user-library" contain the following:

- A file FILEDIR.SAG containing library information used by Natural including, in particular, the programming mode of an object (structured or reporting) and internally converted object names. These internal object names are automatically created when storing Natural objects to disk with:
  - names longer than 8 characters (which can be the case with DDMs);
  - names containing any special character supported by Natural but not by the operating system.
- Internal object names are unique and consist of an abbreviation of the actual object name and an arbitrary number. Both the actual object name and the corresponding internal object name are documented in FILEDIR.SAG.

Even if an object is located in the correct directory, it can only be used by Natural after this library information is included in FILEDIR.SAG. For objects created within Natural, the library information is included automatically. Information on how to import other objects can be found in the section Importing Objects in the Natural User's Guide for Windows.

The utility FTOUCH can be used to update FILEDIR.SAG without entering Natural.

- A subdirectory SRC containing the source objects stored in the library.
- A subdirectory GP containing the generated programs stored in the library.
- A subdirectory ERR containing the error messages stored in the library.
- A subdirectory RES containing the private and shared resources stored in the library.

DDMs can be stored in local libraries (for example, *USERnn*). If DDMs are used by a program, Natural first searches the current library, then the steplib, and then SYSTEM. If the DDMs are not found, the program does not compile and displays an error message.

**Warning:**

**Do not access Natural files with operating system utilities. These utilities might modify and destroy the Natural directory information.**

**Do not store private data files in the FNAT/FUSER directories, since Natural may delete or modify them in an unexpected way.**

**Do not use one of the FNAT/FUSER subdirectories as working directories for your Windows applications, since this can cause problems when issuing Natural system commands.**

**Do not set the directories FNAT and FUSER or files (filedir.sag, for example) to read-only to prevent libraries and their files from being updated. This is not supported by Natural and can cause problems when using system commands and lead to a loss of information.**

# Configuration Files

The base directory for FNAT/FUSER is contained in the following configuration files:

- Local Configuration File - Natural.INI
- Global Configuration File - NATCONF.CFG

The local configuration file is located in the appropriate "etc" directory for each installed Natural version.

If you run a client/server environment and multiple clients access the same database and/or system file, only one global configuration file with one DBID and FNR can be available to these clients, which means that you can only have one global configuration file per client/server environment.



These configuration files should only be modified by an administrator. They contain plain ASCII text and should only be changed with the Natural Configuration Utility.

## Local Configuration File - Natural.INI

The local configuration file contains Buffer Pool Assignments and Installation Assignments.

- Buffer Pool Assignments
- Installation Assignments

### Buffer Pool Assignments

Parameter	Function
Buffer Pools	Buffer Pool Settings

If you select "Buffer Pool Assignments", a dialog with a list of existing buffer pool assignments is displayed. You can specify the name (BPID), the size (BPSIZE) and the number of directory entries (BPNLE) of the buffer pool. You can also specify the maximum number of users that can have simultaneous access to the Buffer Pool (MAXUSER).

For further information on the profile parameters BPID, BPSIZE, BPNLE, MAXUSER (to be set in the Natural Configuration Utility), see also the buffer pool.

## Installation Assignments

If you select "Installation Assignments", a dialog is displayed in which you can specify the following items:

Parameter	Function
<b>Profile Path</b>	<b>Path to Profile Parameters</b> - The location of the Natural parameter files.
<b>Global Config File</b>	<b>Global Configuration File</b> - The name and path of the global configuration file (default name is NATCONF.CFG).
<b>Error Path</b>	<b>Natural Error File Directory</b> - The location of the Natural error messages.
<b>Conversion Table</b>	<b>Natural I/O Conversion Table</b> - The name of the file which contains the character translation tables used with the internal character set "ISO-8859-1". By default, this file is called NATCONV.INI.
<b>TMP Path</b>	<b>Natural TMP Directory</b> - The location of Natural temporary output. Instead of defining a single path name in the NATPARM utility (for example: c:\Natural\temp), you can define an environment variable (for example: %usertemp%=mytempdir) and embed this variable in the path name (for example: c:/Natural/%usertemp%). The complete path name will be expanded at run time, using the currently valid environment variable (for example: c:/Natural/mytempdir\).
<b>Documentation Base</b>	<b>Natural Online Help File Directory</b> - The location of the Natural online help file. This is the default path. Alternatively you can define a path to the Natural documentation set (for example in a network environment). URLs are possible as paths also.

## Global Configuration File - NATCONF.CFG

The global configuration file contains DBMS Assignments, Dictionary Server Assignments, Assignments of Printer Profiles, and System File Assignments.

- DBMS Assignments
- Dictionary Server Assignments
- Printer Profiles
- System File Assignments

### DBMS Assignments

Parameter	Function
<b>DBMS Assignments</b>	DBMS Assignments
<b>SQL Date/Time Conversion</b>	SQL DBMS Date/Time Conversion
<b>Multifetch Disabling</b>	Multifetch Disabling

If you select "DBMS Assignments", a dialog is displayed in which you can specify the DBID, DBMS Type and DBMS Parameter.

### DBID and DBMS Type

Since the types of all databases which are to be accessed by Natural must be defined in the global configuration file, specify one of the following values for each database ID:

Value	Database Type
<b>ADA</b>	Adabas database server (this is the default).
<b>ADX</b>	Adabas single-user database (for PCs only).
<b>SQL</b>	Any SQL database that can be accessed using Entire Access, Software AG's common interface to various SQL database systems. Note that in the user interface of the Natural Configuration utility the term <b>OSQ</b> is replaced with the term <b>SQL</b> .

**Note:**

You cannot define a database type for a DBID which has already been assigned to a system file; if you do so, an error message will be issued at Natural startup, indicating an inconsistency in the system file setting and an error when reading the database assignments.

A list of existing DBMS assignments is displayed in a list box.

## DBMS Parameter

This field applies to SQL-type databases only.

In this field, you establish the connection to the database system that you want to work with. In addition, you can specify logon parameters specific to the database type.

See Natural and Entire Access for further information on how to access SQL-type database systems.

## SQL Date/Time Conversion

As Natural has only one specific time format, you must decide how this format should be interpreted in the context of SQL database access. There are several possibilities, however, there is only one possibility per SQL-type DBID which can be specified here.

If you select "SQL Date/Time Conversion", a dialog is displayed in which you can specify the conversion masks.

### Mask

The value specifies the configuration for Entire Access. It also specifies the format used to retrieve SQL DATE/TIME/DATETIME information into Natural format A fields. The mask should match the RDBMS-specific configuration for the DATE, TIME, or DATETIME character string representation.

### Date

This mask (usually a sub-string of the Mask value) specifies the character string representation into which the Natural format D fields are converted during update or retrieval of SQL DATE columns.

### Time

This mask (usually a sub-string of the Mask value) specifies the character string representation into which the Natural format T fields are converted during update or retrieval of SQL TIME or DATETIME columns.

### Remark

You can enter your remarks here, for example, to document how the SQL DATE/TIME character string representation is configured on the database site.

See also the Entire Access documentation, Using Natural with Entire Access, Date/Time Conversion.

## Multi Fetch Disabling

Natural uses Adabas command level multi fetch for Adabas L1, L2, L3 and L9 calls to minimize the number of database calls. For performance reasons it would be useful to disable multi fetch on an Adabas file and command level basis. This can be done with the Multi Fetch option. Before a multi fetch call is issued, Natural checks if multi fetch is disabled for the actual dbid/file/command code combination.

## Dictionary Server Assignments

Parameter	Function
Dictionary Servers	Dictionary Servers

With the "Dictionary Server Assignments" function you can assign three so-called dictionary servers to one common logical server name.

With dictionary servers, you can access remote DDMs, free rules and automatic rules maintained in Predict once you have access to Predict on a mainframe or UNIX host. The three dictionary servers are:

<b>The DDM Server</b>	Server for remote DDM access.
<b>The Free Rule Server</b>	Server for remote access to Predict free rules.
<b>The Automatic Rule Server</b>	Server for remote access to Predict automatic rules.

Since the servers to be assigned can be located on different nodes, both server and node name must be specified.

All dictionary servers must previously be defined by using the Natural RPC facility. The RPC parameter MAXBUFF has a maximum value of 16 KB. The actual maximum buffer length that can be processed by the transport layer may be less than 32 KB. A value of 30 KB is recommended due to broker limitations.

**Note:**

Currently, a server's node name must start with "FBKR" if you want it to be addressed using a central mainframe broker.

Multiple logical server names can be defined; server assignments can be modified or deleted. A list of existing assignments can be displayed in the "Logical Dictionary Server Name" combo box.

Specify the dictionary servers you want to use by setting the "Remote Access" parameter USEDIC to the corresponding logical server name; if USEDIC is set to blank, remote DDM access will not be possible.

**Note:**

If you want to use this feature, Predict Version 3.3 or above must be installed; with PredictT versions prior to 3.3, the Predict update tape PD2302 is required.

## Printer Profiles

Parameter	Function
<b>Printer Profiles</b>	Printer Profiles
<b>TTY Printer Profiles</b>	TTY Printer Profiles
<b>GUI Printer Profiles</b>	GUI Printer Profiles

This function is used to define, modify or delete printer profiles.

Printer profiles are used for printing additional reports, for hardcopies and for batch output generation. They recognize particular Natural field attributes and insert the appropriate control sequences (see below) as defined in the profile.

With the ability to translate Natural field attributes into escape sequences, you can control your printer in various ways by using the right profile name, and you can use the print features of a given device by using simple attributes in Natural programs.

Each profile can be assigned to a Natural report number either statically by using the "Report Assignments" function of the Natural Configuration Utility, or dynamically by using the DEFINE PRINTER statement within a Natural program.

There are two different kinds of printer profiles: TTY-Type Profiles and GUI-Type Profiles.

Use TTY-type profiles if you wish to have full control over the command sequences sent to the printer; use GUI-type profiles if you wish them to be controlled by an installed printer driver.

### TTY-Type Profiles

If you select the TTY option, a dialog is displayed in which you can specify a profile name, leading and/or trailing commands (printer control sequences) to be triggered at job, page or field level and an external character set name.

Trigger:

A triggering event controls the level on which specified printer control sequences are to be applied. Available triggering events are JOB, PAGE, AD and CD:

- Specify JOB if you want your control sequences to apply to an entire print job; the specified control sequences will represent the job header and/or job trailer respectively.
- Specify PAGE if you want the control sequences to apply to each physical output page; the specified control sequences will then represent the page headers and/or page trailers respectively.
- Specify one of the Natural session parameters AD or CD along with appropriate field attributes if you want the control sequences to be applied at field level only; any field in a Natural program with corresponding attributes will then cause these control sequences to take effect.  
See the Reference documentation for details on these session parameters.

## Printer Control Sequences:

For each control sequence, a window appears, in which you can specify the control characters in either hexadecimal or alphanumeric format.

- The leading control sequence is inserted immediately before the triggering event (for example, to define a job header or to set attributes for field representation).
- The trailing control sequence is inserted immediately after the triggering event (for example, to define a job trailer or to reset attributes previously set).

**Example:**

```
Alphanumeric specification:  
Hexadecimal specification: ^1b(s1P ^1b^28^73^31^50
```

**Note:**

The escape and the blank character must always be specified in hexadecimal format.

## External Character Set:

A character set maps characters to underlying hexadecimal byte representations. An external character set must be defined if, for example, a printer's character set is different from the (internal) character set used by Natural. To be able to use the printer with the (external) character set, you have to define a translation table. To do that, edit the NATCONF.INI local configuration file and add the following section to the already existing translation tables:

```
[ISO8859_1 -> mycharset]
```

where *mycharset* is the name you enter in the External Character Set text field.

## GUI-Type Profiles

If you select the GUI option, a dialog is displayed in which you can specify:

<b>Profile Name</b>	The name of the printer profile. A list of existing GUI-type printer profiles is displayed in this combo box.
<b>Logical Device</b>	The logical device name (LPT1 to LPT31). The specified logical device name must have been assigned to an existing physical device (by using the Natural Configuration Utility).
<b>Fonts</b>	The fonts to be used. Fonts are always associated with a particular representation attribute (AD=) which can be selected in the "Attribute" combo box. Up to six attributes are available. The fonts to be associated with these attributes are displayed for selection by choosing the "Set" button. Depending on whether you have checked the "Fixed Fonts Only" box, either only fixed fonts or all fonts (fixed and proportional ones) are displayed. You can now either set (that is, assign) a particular font to the selected attribute or modify an already existing font assignment. With the "Remove" button you can delete an existing assignment. By choosing the "Extra Leading" button you can specify (in points) an extra vertical line spacing in addition to the default line spacing.
<b>Text Color</b>	The color to be used. You can either leave the color setting unspecified (whereupon a default setting is used), ignore any color settings (that is, print everything in black), or retain any color settings in your Natural program (as specified by using the CD session parameter).
<b>Margins</b>	The page margins. You can specify (in points) an extra top, bottom, left, and/or right page margin relative to the top left-hand corner of the printable region.

## System File Assignments

Parameter	Function
System File Assignments	System File Assignments

For each Natural system file, a database ID (DBID) and a file number (FNR) is specified in the Natural parameter file NATPARAM. To ascertain the location of each system file in the structure of directories, you use the "System File Assignment" function to assign a path name to the DBID/FNR combination. A list of existing assignments is displayed in a list box.

The path name must be a valid directory path, indicating the physical path to the location of the system files on the disk.

### Example:

System file specifications in NATPARAM:

```
FNAT: DBID=098,FNR=099 FUSER: DBID=042,FNR=043
```

The path assignments in NATCONF.CFG	represent the following directory structure:
DBID: 098 FNR: 099 Path: c:\sag\natapps\db9899	c:\sag\natapps\db9899 (for FNAT)      - SYSLIB     .....
DBID: 042 FNR: 043 Path: c:\sag\natapps\db4243	c:\sag\natapps\db4243 (for FUSER)      - SYSTEM     .....

# Work Files

Work files are files where data can be written to and read from by Natural programs. They are used for intermediate storage of data and for data exchange between programs. Data can be transferred from or to a work file by using the Natural statements READ WORK FILE and WRITE WORK FILE, or UPLOAD and DOWNLOAD.

- Defining Work Files
  - Work File Formats
  - Special Considerations on Work Files with Extension NCD
- 

## Defining Work Files

To define a work file, select the "Work Files" parameter group in the Natural parameter file. A list of existing work file specifications is displayed in the "Work File Name" list box.

Scroll down the list box until the number of the work file you wish to define is visible.

If the specified work file was previously defined, the current work file definition is displayed, which can be overwritten. Enter the complete definition; that is, name and extension of your work file, and choose the desired work file format.

If the work file was never defined, the "File Name" text box is blank. Enter the work file name including its path name.

For any numbered work file, a complete work file specification can be entered. This specification must contain the path and the work file name.

If no work file definition is specified, Natural automatically creates the file name and writes the work file into the "tmp" directory specified in the local configuration file. The work file name generation is based on an algorithm which tries to generate a unique name. Depending on the Natural parameter TMPSORTUNIQ, the naming convention may vary. If work file names are referenced from outside Natural, it is recommended that one specify the names explicitly to avoid problems identifying the files.

## Work Files with Environment Variables

Work files can also be defined by using environment variables; that is, work file names can be set without changing the Natural parameter file.

### Example:

Specify the following name for a work file:

```
%Natural%\%myfile%
```

and assume the following settings:

```
set Natural=D:\nat
set myfile=sub\test
```

which will expand into the following file name during Natural startup:

```
D:\nat\sub\test
```

## Work Files with User Exit

Work files can also be defined by using the user exit USR1050N in library SYSEXT.

## Work File Formats

Three different work file formats are available. Natural recognizes the format by checking the work file specification. The work file specification consists of a file name and an extension separated by a period:

*file-name.extension*

The *file-name* can have a maximum of 8 characters, the *extension* can have a maximum of 3 characters.

The three work file formats are:

- a binary format specific to Software AG (to be preferred),
- ASCII format,
- a format which corresponds to the one used by Entire Connection.

For the work file number to be used, a work file specification is required for the appropriate format as described in the following sections.

## Binary Format

The binary format specific to Software AG is defined by using any file name and either a period and the extension "SAG" or no period and extension at all.

**Note:**

A two (2) byte length precedes each record written.

**Examples:**

**xxxxxxx.SAG (any file name with a period and an "SAG" extension)**

**xxxxxxx (any file name without a period or extension)**

The binary format can be used with all data types.

## ASCII Format

To define an ASCII work file format, enter a file name, a period and either any extension except "SAG" and "NCD" or no extension at all.

**Examples:**

**xxxxxxx.xxx (any file name with a period and any extension except "SAG" or "NCD")**

**xxxxxxx. (any file name and a period)**

Since each record written is terminated with the CR/LF character sequence, this format is recommended for alphanumeric data only.

## ENTIRE CONNECTION Format

The product Entire Connection uses two files: a data file, which contains the actual data, and a format file, which contains format information about the data in the data file.

The data file is defined by using any file name, a period and the extension "NCD"; the corresponding format file is automatically generated; it has the same name as the data file, but the extension ".NCF". Both extensions must be in upper case characters.

**Examples:**

**xxxxxxx.NCD (any file name with a period and an "NCD" extension for the data file)**

**xxxxxxx.NCF (any file name with a period and an "NCF" extension for the format file)**

You can read/write work files in Entire Connection format directly from/to your local disk.

**Note:**

With Entire Connection, the RECORD option of the READ WORK FILE statement makes no sense and can therefore not be used.

## PORTABLE Format

PORTABLE performs an automatic endian conversion of a work file when it is transferred to a different machine. For example, a work file written on a PC (little endian) can be read correctly on an RS6000 or HP machine (big endian). The endian conversion applies only to field formats I2, I4, F4 and F8. The floating point format is assumed to be IEEE. There are, however, slight differences in IEEE floating point representation by different hardware systems. As far as we know, these differences apply only to infinity and NaN representations,

which are normally not written into work files. Check the hardware descriptions if you are uncertain.

The files are always written in the machine-specific representation, so that a conversion is performed only if the file is read by a machine with different representation.

This keeps performance as fast as possible. There are no other conversions for PORTABLE, but the file format makes it possible to add them in future releases.

When a READ WORK is done for a dynamic variable, it is resized to the length of the current record.

## **UNFORMATTED Format**

UNFORMATTED reads or writes a complete file with just one dynamic variable and just one record, e.g., to store a video which was read from a database, and vice versa. No formatting information is inserted, everything is written/read just as it is.

## Special Considerations on Work Files with Extension NCD

**Note:**

When you create an ".NCD" file using Entire Connection and load this file using the Natural SYSTRANS utility, you may receive a "Source Control Record missing" error. To avoid this, create the ".NCD" file without the suppression of blanks.

If files with the extension .NCD are created by Entire Connection and are then read into Natural via READ WORK, it is required that Entire Connection is setup in the following way:

1. In the menu 'Host' click 'Session setup'.
2. Click the session and then the 'Modify' button.
3. Select the tab 'File Transfer 1'. Make sure that the box for 'Keep trailing blanks' is checked.

The following considerations apply for work files in Entire Connection format:

1. If an ".NCD" file is read with a READ WORK FILE statement and the corresponding ".NCF" format file is not available or contains invalid information, the ".NCD" file is assumed to be an ASCII work file.
2. The maximum work file record size and format file size that can be handled by Entire Connection is 1900 bytes.
3. When an array is written to or read from an ".NCD" work file, a maximum of 255 array elements can be written/read at a time.
4. The VARIABLE option of the WRITE WORK FILE statement cannot be used for ".NCD" work files.
5. If you have "old" work files whose names have ".NCD" extensions, the extensions must be changed.
6. Each of the following profile parameters must be set to the same value for both read and write operations:  
DC (decimal character),  
IA (input assign character),  
ID (input delimiter character).
7. Remember that the range of possible values for floating point variables on a mainframe computer is different from that on other platforms.  
The possible value range for F4 and F8 variables on a mainframe is:  
 $\pm 5.4 * 10^{-79}$  to  $\pm 7.2 * 10^{75}$   
The possible value range on most other platforms is,  
for F4 variables:  
 $\pm 1.17 * 10^{-38}$  to  $\pm 3.40 * 10^{38}$   
for F8 variables:  
 $\pm 2.22 * 10^{-308}$  to  $\pm 1.79 * 10^{308}$

# Natural Buffer Pool

- Buffer Pool under Windows NT
  - Setting up the Buffer Pool under Windows NT
  - Windows NT Service
  - NATBPMON Utility
  - Buffer Pool Trouble Shooting
- 

## Buffer Pool under Windows

Since Natural generates reentrant Natural object code, it is possible that a single copy of a Natural object can be executed by more than one user at the same time. For this purpose, each object is loaded only once from the system file into the Natural buffer pool, instead of being loaded by every caller of the object.

Thus, the purpose of the Natural buffer pool is to share Natural objects between several Natural users working on the same computer. It is a pool of storage into which programs compiled with Natural are loaded in preparation for execution. They are moved into and out of the buffer pool as Natural users request Natural objects.

Objects in the buffer pool can be programs, subprograms, maps, global data areas, local data areas, parameter data areas and copycodes. Local data areas, parameter data areas and copycodes are only placed in the buffer pool for compilation purposes.

When a Natural object is loaded into the buffer pool, a control block is allocated for that object. This control block contains information such as the name of the object, what library or application it belongs to, what database ID and Natural system file number the object was retrieved from, and certain statistical information (for example, the number of users who are concurrently executing a program).

Resource sharing requires that access to the buffer pool is coordinated among all users. Several system resources are necessary to accomplish this. For example, shared memory on a UNIX operating system is used to store the objects and their administrative information. To synchronize access to these objects, a set of semaphores is used. The amount of available shared memory and the number of semaphores is configured statically in the operating system, and as a result, it may be necessary to change system parameters and to recreate the operating system kernel for your installation. Further information about these topics is system-dependent and is described in the Natural installation manual for your UNIX computer. For OpenVMS, the buffer pool uses a standard locking mechanism which does not need to be configured by the administrator.

Several instances of the Natural buffer pool can be started on one computer, depending on the individual requirements. It is also possible to run different versions of the buffer pool on one computer without any problems. These buffer pools have nothing in common, except that they run on the same computer.

When a user executes a program, a call is made to the buffer pool manager. The directory entries are searched to see if the program exists. If it does not exist in the buffer pool, a copy is retrieved from the appropriate library and loaded into the buffer pool.

When a Natural object is being loaded into the buffer pool, a new directory entry is defined to identify this program, and one or more other Natural objects which are currently not being executed may be deleted from the buffer pool in order to make room for the newly loaded object.

For this purpose, the buffer pool maintains a record of which user is currently using which object, and it detects situations in which a user exits Natural without releasing all its objects. It dynamically deletes unused or out-of-date objects to accommodate new objects belonging to other applications.

When you are using the Natural buffer pool, only minimum restrictions must be considered:

- When a Natural session hangs up, do not terminate it by using WindowsNT Task Manager
- If this process is performing changes to the buffer pool internal data structures, an interruption may occur at a stage where the update is not fully completed. If the buffer pool internal data structures are inconsistent, this could have negative effects.

**Note:**

This can only happen when the Natural nucleus is executing buffer pool routines.

## Setting up the Buffer Pool under Windows NT

The buffer pool is set up by making various specifications with the Natural Configuration Utility.

### To start the Natural Configuration Utility

- From the "Start" menu, choose "Programs", then "Natural *version*", then "Natural *version* Configuration Utility".
- Select the "Local Configuration File" and then "Buffer Pool Assignments".
- A dialog is displayed in which you can make the following buffer pool specifications:
  - Buffer Pool ID (BP or BPID)** = the name of the buffer pool.
  - BPSIZE** = size (in MB) of the buffer pool.
  - MAXUSER** = maximum number of concurrent processes.
  - BPNLE** = number of directory entries.

## Windows NT Service

Natural for Windows NT uses a service to start the Buffer Pool Server when the PC is booted.

Natural for Windows NT is installed with the default buffer pool NATBP, which is also used as the default buffer pool name at Natural start up.

You can modify the service configuration to meet your requirements. The following commands are available in the Command Line Interface.

### Natural Buffer Pool Service Commands

- Install Service
- Remove Service
- Start Service
- Start the Specified Buffer Pool
- Stop Service
- Stop the Specified Buffer Pool
- Create New Buffer Pool To Be Started by the Service
- Delete Specified Buffer Pool from the Service
- Define Whether Specified Buffer Pool Is To Be Started When the Service Is Started
- Display the Buffer Pools Defined for the Service and Whether These Buffer Pools Are To Be Started When the Service Is Started
- Display Whether the Specified Buffer Pool Is To Be Started When the Service Is Started
- Display Status of All Buffer Pools - Active or Not Active
- Display Status of a Specified Buffer Pool - Active or Not Active

## Install Service

### NATBPSVC INSTALL *manual (default) / automatic*

Two parameters are available: manual (default) and automatic.

Manual: Service is installed and must be started manually (either from the Start Service command or from the "Services" dialog in the "Control Panel").

Automatic: Service is installed and automatically started when the PC is booted.

Example for manual:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Install
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Natural version Bufferpool Service successfully installed
%NATBPSVC-I: Path of binary is F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE
%NATBPSVC-I: Startup mode of Natural version Bufferpool Service is 'Manual'
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

Example for automatic:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Install Automatic
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Natural version Bufferpool Service successfully installed
%NATBPSVC-I: Path of binary is F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE
%NATBPSVC-I: Startup mode of Natural version Bufferpool Service is 'Automatic'
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

## Remove Service

### NATBPSVC REMOVE

Remove the service from the system.

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Remove
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Natural version Bufferpool Service successfully removed
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

## Start Service

### NATBPSVC START

Start service if not yet active, and also start all created buffer pools (with parameter Start=Yes). See commands Create and Set.

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Start
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Starting Natural version Bufferpool Service .
%NATBPSVC-I: Natural version Bufferpool Service started
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

## Start the Specified Buffer Pool

### NATBPSVC START *bufferpool-name*

Starts the specified buffer pool. If the service has not been started (automatically at boot time or manually by the user) an error message is displayed.

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Start NATBP
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Send request to Natural version Bufferpool Service
%NATBPSVC-I: Bufferpool 'NATBP' successfully started
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

## Stop Service

### NATBPSVC STOP

Stops the service and stops all previously started buffer pools.

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Stop
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Stopping Natural version Bufferpool Service
%NATBPSVC-I: Natural version Bufferpool Service stopped
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

## Stop the Specified Buffer Pool

**NATBPSVC STOP *bufferpool-name***

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Stop NATBP
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Send request to Natural version Bufferpool Service
%NATBPSVC-I: Bufferpool 'NATBP' successfully stopped
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

## Create New Buffer Pool To Be Started by the Service

**NATBPSVC CREATE *bufferpool-name***

The Service checks whether the buffer pool with the specified name is defined in Natural parameter file.

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Create NATBP
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: New Bufferpool 'NATBP' created
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

## Delete Specified Buffer Pool from the Service

**NATBPSVC DELETE *bufferpool-name***

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Delete NATBP
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Bufferpool 'NATBP' deleted
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

**Define Whether Specified Buffer Pool Is To Be Started When the Service Is Started****NATBPSVC SET *bufferpool-name* start= *yes/no***

The default is "no".

Examples:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Set NATBP Start=Yes
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Configuration successfully set
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

or

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Set NATXY Start=No
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Configuration successfully set
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

**Display the Buffer Pools Defined for the Service and Whether These Buffer Pools Are To Be Started When the Service Is Started****NATBPSVC SHOW**

Display configuration parameters for all buffer pools.

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Show
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Settings for Natural version Bufferpool Service:
%NATBPSVC-I: Number of Bufferpools is 1
%NATBPSVC-I: Settings for Bufferpool 'NATBP'      : START=Yes
%NATBPSVC-I: Settings for Bufferpool 'NATXY'     : START=No
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

**Display Whether the Specified Buffer Pool Is To Be Started When the Service Is Started****NATBPSVC SHOW *bufferpool-name***

Display configuration parameters for specified buffer pool.

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Show NATBP
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Settings for Bufferpool 'NATBP' : START=Yes
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

**Display Status of All Buffer Pools - Active or Not Active****NATBPSVC STATUS**

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Status
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Send request to Natural version Bufferpool Service
%NATBPSVC-I: NATBP is active
%NATBPSVC-I: NATXY is not active
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

**Display Status of a Specified Buffer Pool - Active or Not Active****NATBPSVC STATUS *bufferpool-name***

Example:

```
F:\SAG\NAT\Vnnn\BIN\NATBPSVC.EXE Status NATBP
%NATBPSVC-I: Natural version Bufferpool Service configuration startup
%NATBPSVC-I: Send request to Natural version Bufferpool Service
%NATBPSVC-I: NATBP is active
%NATBPSVC-I: Natural version Bufferpool Service configuration done
```

## NATBPMON Utility

This utility should not be generally accessible to all users of Natural, because its use can cause damage to the work of other users of the buffer pool.

The purpose of the Natural utility NATBPMON is to monitor the buffer pool's activity during its operation. NATBPMON can also be used to shut down the buffer pool, when Natural must be stopped on a computer.

NATBPMON collects information on the current state of your Natural buffer pool.

The buffer pool contains Natural objects (such as maps, programs and subprograms). When an object is invoked, Natural tries to find the object in the buffer pool. If the object is found in the buffer pool, it will be executed without accessing the system file where it is stored. If the object is not found in the buffer pool, the system file containing the object will be accessed, the object copied, and the copy placed in the buffer pool.

All objects in the buffer pool are node-specific and if an object is updated or changed in one node, the object will automatically be deleted from the buffer pool running on this node. But if the same object exists in the buffer pool of another node, the object in the buffer pool of the other node remains unchanged.

### How to Invoke NATBPMON

You invoke the NATBPMON utility by selecting the Natural Buffer Pool Monitor in the Natural Program group or by choosing Run from the Windows NT Start Menu:

**...\*Vnnn*\Bin\natbpmon.exe**

By default, the buffer pool "NATBP" is used. If another buffer pool is to be used, you invoke NATBPMON as follows:

**...\*Vnnn*\Bin\natbpmon.exe BP=*buffer-pool-name***

Once you have invoked NATBPMON, the following prompt is displayed in a DOS box:

**NATBPMON>**

## NATBPMON Commands

NATBPMON provides several commands, which you enter at the NATBPMON. The prompt individual commands are described below.

### CLEAR

```
CLEAR
```

This is a synonym of the ZERO command.

### CORPSES

```
CORPSES
```

The CORPSES command is used to display the list of "corpses".

A "corpse" is an object that has been deleted, but was still being used in the buffer pool when its deletion took place.

Once this object is no longer being used, it will automatically disappear from the list of "corpses".

**Note:**

The column "cusr" (described with the DIR command) indicates if an object is being used.

### DELETE

```
DELETE { PATTERN | * }
```

The DELETE command is used to delete an object from the buffer pool.

All objects can be deleted from a buffer pool by using an asterisk (\*).

A *pattern* is used to specify a collection of objects, similar to current operating systems which allow the specification of a class of files with wildcards. For details on how to specify a *pattern*, see the DIR command below.

## DIR

**DIR { PATTERN | \* }**

The DIR command is used to display a directory containing all objects in a buffer pool.

This list of objects contains the following information:

Column	Explanation
<b>indx</b>	A number that the NATBPMON utility automatically assigns to an object when it is loaded into the buffer pool.
<b>cusr</b>	The current number of users that are using an object in the buffer pool.
<b>pusr</b>	The peak number of concurrent activations of an object in the buffer pool.
<b>nusg</b>	The number of times an object has been activated in the buffer pool.
<b>g</b>	Specifies if an object is being loaded into the buffer pool from the system file and will have either of the following values: <b>0</b> - The object is not being loaded. <b>1</b> -The object is being loaded.
<b>size</b>	Specifies the size (in bytes) of an object in the buffer pool.
<b>KeyN</b>	Specifies the following information about an object: The name of the object. <b>L</b> -The library in which the object is located. <b>K</b> -The kind of object (G=generated object module; S=source; D=part of FILEDIR.SAG). <b>T</b> -The object type (which is "blank" in the case of "D" in the K field).

When the DIR command is issued, all objects in the pool will be displayed in a notation similar to the following:

```
NATBPMON>dir
indx:   index of the element
cusr:   current number of concurrent users
pusr:   peak number of concurrent users
nusg:   number of usages
g   :   set if object is generating
```

```
indx  cusr  pusr  nusg  g   size  key
```

1	0	1	4	0	920	(N="SEL-MAP" L="DEMO" K='G' T='M')
2	0	1	2	0	3096	(N="EMWND" L="DEMO" K='G' T='P')
3	0	1	4	0	604	(N="HDR" L="DEMO" K='G' T='P')
4	0	1	7	0	412	(N="MMUPROG1" L="RPA" K='G' T='P')
5	0	1	5	0	372	(N="MMUPROG2" L="RPA" K='G' T='P')
6	0	1	4	0	372	(N="MMUPROG3" L="RPA" K='G' T='P')

To select some objects, it is possible to restrict the values of certain key fields by specifying a matching pattern expression.

To restrict the allowed field values of a given field, the following pattern notation must be used:

*name=expression*

You can specify multiple patterns by separating them with a comma.

The specified patterns must all match their corresponding fields in order to accept the entire key.

The expression accepts the specification of the wildcard characters "\*" and "?".

The character "\*" matches any sequence of characters (also none), and the wildcard character "?" matches exactly one character.

To select all objects of type "P" in the sample above, the following command would be used:

**DIR T=P**

To select all programs in the demo library, the following command would be used:

**DIR T=P, L=DEMO**

To select all objects containing an "M" in their name, the following command would be used:

**DIR N=\*M\***

The special pattern "\*" exists, which matches all objects stored in the buffer pool. To select all objects, the following command can be used:

**DIR \***

## DUMP

**DUMP**

*Do not use this command unless you are requested to do so by Software AG Support.*

The DUMP command is used for error analysis.

## EXIT

**EXIT**

The EXIT command is used to exit the NATBPMON utility.

## HELP

**HELP**

The HELP command is used to display a list of all available NATBPMON commands.

## PARAM

Is used to display the Natural Buffer Pool settings.

```
NATBPMON> param
Active since .....: 11-03-2002 09:49:09.87, Version 1.3(435)
Last time cleared .....: 11-03-2002 09:49:09.87

Bpid .....: NATBP
Shmkey.....: NAT411BPSEM_0X14111111
Semkey.....: NAT411BPSEM_0X14111111
Memsize .....:10485760
Maxusers .....: 10
```

## QUIT

**QUIT**

The QUIT command is used to exit the NATBPMON utility; it is a synonym for EXIT.

## SHUTDOWN

**SHUTDOWN**

The SHUTDOWN command is used to shut down the buffer pool.

No new users will be able to use the buffer pool once this command has been issued.

The NATBPMON utility is able to run with the pool, which has the status shutdown "pending" (All commands are available with shutdown "pending").

## STATUS

### STATUS

The STATUS command is used to display statistical information about the buffer pool.

The following statistics are displayed:

```
NATBPMON> status
Active since .....: 11-03-2002 09:52:09.87, Version 1.3(435)
Last time cleared .....: 11-03-2002 09:52:09.87
Bpid .....: NATBP
Allocated memory (bytes) .....: 1256436 Current users .....: 2
Smallest allocation .....: 20 Peak users .....: 2
Largest allocation .....: 145404 Dead users purged .....: 4
Free memory (bytes) .....: 9229336
Smallest free .....: 15996
Largest free .....: 6355900

Dormant objects .....: 204 Smallest object (bytes) .....: 316
Active objects .....: 0 Largest object (bytes) .....: 43048
Generating objects .....: 0 Total object sizes .....: 1072570
Obsolete objects .....: 0

Attempted locates .....: 126396 Stored objects .....: 0
Attempted fast locates .....: 60116 Loaded objects .....: 3126
Successful fast locates .....: 55528 Activated objects .....: 114323
Percent .....: 92.37 Aborted loads .....: 0

Dormant objects purged .....: 0 Peak parallel activations ...: 2
Object reuse factor .....: 36.57
NATBPMON>
```

## Explanation of the individual statistics:

<b>General Information</b>	
<b>Active since</b>	Date and time when the buffer pool was started and the version number of the buffer pool.
<b>Last time cleared</b>	Date and time when the buffer pool was most recently cleared.
<b>BPID</b>	Buffer pool ID.
<b>Memory Allocation</b>	
<b>Allocated memory (bytes)</b>	Sum total of all allocated memory.
<b>Smallest allocation</b>	Smallest amount of allocated memory.
<b>Largest allocation</b>	Largest amount of allocated memory.
<b>Free memory (bytes)</b>	Sum total of all free memory.
<b>Smallest free</b>	Smallest amount of contiguous free memory.
<b>Largest free</b>	Largest amount of contiguous free memory.
<b>User Statistics</b>	
<b>Current users</b>	Number of users currently using the buffer pool.
<b>Peak users</b>	Peak number of users that have been using the buffer pool.
<b>Dead users purged</b>	Number of inactive users that have been deleted from the buffer pool.
<b>Object Use Statistics</b>	
<b>Dormant objects</b>	Number of available, but inactive objects. These objects are in the buffer pool, but are not being used. They are available for later use.
<b>Active objects</b>	Number of active objects. These objects are currently in use by one of the buffer pool users.
<b>Generating objects</b>	Number of objects that are currently being loaded into the buffer pool. These objects will become available as soon as the load operation completes.
<b>Obsolete objects</b>	Number of objects that are to be deleted from the buffer pool, but are still being used. These object can be displayed by using the CORPSES command.
<b>Object Size Statistics</b>	
<b>Smallest object (bytes)</b>	Size of smallest object in the buffer pool.
<b>Largest object (bytes)</b>	Size of largest object in the buffer pool.
<b>Total object sizes</b>	Total size of all objects in the buffer pool.
<b>Locate Statistics</b>	
<b>Attempted locates</b>	Number of object locates. This is the number of object activations when the former location of an object was known.
<b>Attempted fast locates</b>	Number of attempted activations with known slot. This is the number of object activations when the former location of an object was known. It is highly probable that an object can be found in the same place in the buffer pool when it is reactivated.
<b>Successful fast locates</b>	Number of successful fast locates.

<b>Percent</b>	Percentage of successful fast locates.
<b>Object Loading Statistics</b>	
<b>Stored objects</b>	The number of objects stored in the buffer pool. This is the number of objects that were stored into the buffer pool and which were not loaded from the system file.
<b>Loaded objects</b>	The number of objects loaded from the system file.
<b>Activated objects</b>	The number of objects activated from the buffer pool.
<b>Aborted loads</b>	The number of load operations that were aborted due to memory shortages within the buffer pool, or due to other events.
<b>General Loading Statistics</b>	
<b>Objects purged</b>	The number of unused objects deleted from the buffer pool to make room for newly loaded ones.
<b>Peak parallel activations</b>	The maximum number of parallel activations of one of the objects in the buffer pool.
<b>Object reuse factor</b>	Average number of times an object was reactivated. This number is the ratio of the number of object activations to the number of objects loaded into the buffer pool.

## WHO

### WHO

The WHO command is used to display a list of all users who are using the buffer pool.

The following statistics are displayed:

<b>Statistic</b>	<b>Explanation</b>
<b>index</b>	A number that the NATBPMON utility automatically assigns to each buffer pool user.
<b>tid</b>	The user ID, terminal ID and process ID of a process using the buffer pool.

## ZERO

### ZERO

The ZERO command is used to reset to "0" all counters that are displayed by the STATUS command.

## Buffer Pool Trouble Shooting

The following is a typical command output example with the explanation what went wrong during its execution.

### Problem

Either Natural or the NATBPMON utility cannot be started.

### Examples

You chose from the Start Menu Run ... \Vmmn\bin\Natural

```
Natural Startup Error 16: Unable to open buffer pool.  
Buffer Pool error: "unexpected system call error occurred" (20)  
Buffer pool could not attach to the global shared memory.
```

You chose from the Start Menu Run ... \Vmmn\bin\NATBPMON

```
Cannot get shared memory  
Buffer Pool error: "unexpected system call error occurred" (20)  
Buffer pool could not attach to the global shared memory.
```

These two examples describe one of the most typical problems you are likely to encounter as a Natural administrator or user. These problems occur when you start Natural or the NATBPMON utility, and the buffer pool is not active.

### Solution

Start the buffer pool service.

# Invoking Natural Subprograms from 3GL Programs

Natural subprograms can be invoked from a programming object written in a 3rd generation programming language (3GL). The invoking program can be written in any programming language that supports a standard CALL interface.

For this purpose, Natural provides the interface "ncxr\_callnat". The 3GL program invokes this interface with a specification of the name of the desired subprogram.

**Note:**

Natural must have been activated beforehand; that is, the invoking 3GL program must in turn have been invoked by a Natural object with a CALL statement.

The subprogram is executed as if it had been invoked from another Natural object with a CALLNAT statement.

When the processing of the subprogram stops (either with the END statement or with an ESCAPE ROUTINE statement), control is returned to the 3GL program.

This section covers the following topics:

- Passing Parameters from the 3GL Program to the Subprogram
  - Example of Invoking a Natural Subprogram from a 3GL Program
- 

## Passing Parameters from the 3GL Program to the Subprogram

Parameters can be passed from the invoking 3GL program to the Natural subprogram. For passing parameters, the same rules apply as for passing parameters with a CALL statement.

The 3GL program invokes the Natural interface "ncxr\_callnat" with four parameters:

- The 1st parameter is the name of the Natural subprogram to be invoked.
- The 2nd parameter contains the number of parameter fields to be passed to the subprogram.
- The 3rd parameter contains the address of the table that contains the addresses of the parameter fields to be passed to the subprogram.
- The 4th parameter contains the address of the table that contains the format/length specifications of the parameter fields to be passed to the subprogram.

The sequence, format and length of the parameter fields in the invoking program must match exactly the sequence, format and length of the fields in the DEFINE DATA PARAMETER statement of the subprogram. The names of the fields in the invoking program and the invoked subprogram can be different.

## Example of Invoking a Natural Subprogram from a 3GL Program

For an example of how to invoke a Natural subprogram from a 3GL program, refer to the samples "MY3GL.NSP" (for the main program), "MY3GLSUB.NSN" (for the subprogram) and "MYC3GL.C" (for the "C" function) in the Natural root directory's subdirectory "samples\syssexuex".

# Natural in Batch Mode

This section contains special considerations that apply when running Natural in batch mode and covers the following topics:

- What is Batch Mode?
- Batch Mode Detection
- Batch Mode Restrictions
- Real Batch Mode
- Sample Session for Batch Mode
- Hints for Using Natural Maps and Dialogs in Batch Mode

For information on starting/terminating a Natural batch session, refer to Starting/Terminating Natural.

---

## What is Batch Mode?

Natural distinguishes two processing modes:

- Interactive Mode
- Batch Mode

The main difference between these two modes is that in interactive mode, the input of commands and data comes from a keyboard and the output is displayed on a screen. In batch mode, input is read from a file and output is written to a file.

This enables Natural to run as a background batch job, where no interaction between the computer and the person who submitted the batch job is necessary. The batch job consists of a set of programs such that each is completed before the next program is started. The programs are executed serially and receive sequential input data.

Batch mode is of interest for mass data processing and re-usable execution.

## Batch Mode Detection

The system variable \*DEVICE shows if Natural is running in batch or interactive mode.

in batch mode	*DEVICE is equal "BATCH"
in interactive mode	*DEVICE is not equal "BATCH" (in most of the cases *DEVICE is equal "VIDEO")

Example:

```

IF *DEVICE = "BATCH" THEN
  /* Batch Mode Process */
  INPUT USING MAP ...
ELSE
  /* Interactive Process */
  PROCESS GUI ...
END-IF

```

## Batch Mode Restrictions

When Natural is running in batch mode, some features are not available or are disabled:

- There is no mouse support;
- No different character fonts are available;
- Only data for an INPUT statement can be processed, dialog input is only conditionally supported (see section Hints for Using Natural Maps and Dialogs in Batch Mode);
- Output appearance is not GUI-like (it is character-oriented output);
- No colors and video attributes (such as blinking, underlined, reverse video) are written to the batch output file CMPRINT;
- Filler characters are not displayed within an INPUT statement;
- No interactive input or output is possible.

## Real Batch Mode

To run Natural in real batch mode you have to specify the dynamic parameter BATCHMODE. In addition, input and output channels have to be defined (as described below).

### Input and Output Channels

The following input and output channels are necessary in batch mode:

- CMSYNIN (Batch Input File for Natural Commands and INPUT Data)  
CMSYNIN is used for the input file that contains Natural commands, and (optionally) data to be read by INPUT statements during execution of Natural programs.
- CMOBJIN (Batch Input File for Natural INPUT Data)  
CMOBJIN is used for data intended to be read by Natural INPUT statements. These types of data can alternatively be placed in the CMSYNIN file immediately following the relevant RUN or EXECUTE command.
- CMPRINT (Batch Output File)  
CMPRINT is used for the output resulting from DISPLAY, PRINT and WRITE statements in a Natural program.
- CMPRTnn (Output File for Additional Reports)  
CMPRTnn is used for additional reports referenced by any Natural program executed during the session. "nn" is two digit decimal number in the range 01 to 31 corresponding to the report number used in a DISPLAY, PRINT or WRITE statement.
- CMWRKnn (Batch Output File for Natural Work Files)  
CMWRKnn is used for Natural work files referenced by any Natural program executed during the session. "nn" is a two digit decimal number in the range 01 to 32 corresponding to the number used in a READ WORK FILE or WRITE WORK FILE statement.
- NATLOG (Natural Log File)  
NATLOG is used to log messages that could not be written to the batch output file CMPRINT. It is recommended to enable NATLOG in batch mode.

### CMPRtnn Specifications in Batch Mode

In order to allow the user to specify variable print-file names, alpha-format system variables and numeric counter markers may be embedded in the filename specification for CMPRTnn. The supported alpha-format system variables are:

- \*APPLIC-ID
- \*APPLIC-NAME
- \*DEVICE

- \*ETID
- \*INIT-USER
- \*LIBRARY-ID
- \*NET-USER
- \*PID
- \*PROGRAM
- \*USER
- \*USER-NAME

If any of these strings (in upper-case only) are encountered within the printfile specification, they will be replaced at run-time with the contents of the appropriate system variable. Additionally, a counter marker (#) may be used. This will be replaced by a 2-digit counter which will automatically be incremented for each print file.

## Sample Session for Batch Mode

The following example demonstrate how to start Natural in batch mode. A simple Natural program is executed and two data items are taken from the batch input file. After the items are processed from the INPUT statement, a subsequent DISPLAY statement writes the data to the batch output file. Then, Natural terminates.

### Natural Commands - CMSYNIN=batch.cmd:

```
LOGON SYSEXBAT
EXECUTE RECCONT
FIN
```

### Natural Input Data - CMOBJIN=batch.inp:

```
Ben,%
Smith
```

### Natural Program RECCONT in Library SYSEXBAT:

```
DEFINE DATA LOCAL
  1 #firstname (A10)
  1 #lastname (A10)
END-DEFINE
INPUT (IP=OFF AD=M) #firstname #lastname
DISPLAY #firstname #lastname
END
```

### Natural Command Line:

```
NATURAL BATCHMODE CMSYNIN=batch.cmd CMOBJIN=batch.inp CMPRINT=batch.out BMSIM=MF NATLOG=ALL
```

When you invoke Natural using the command line described above, Natural produces the following output.

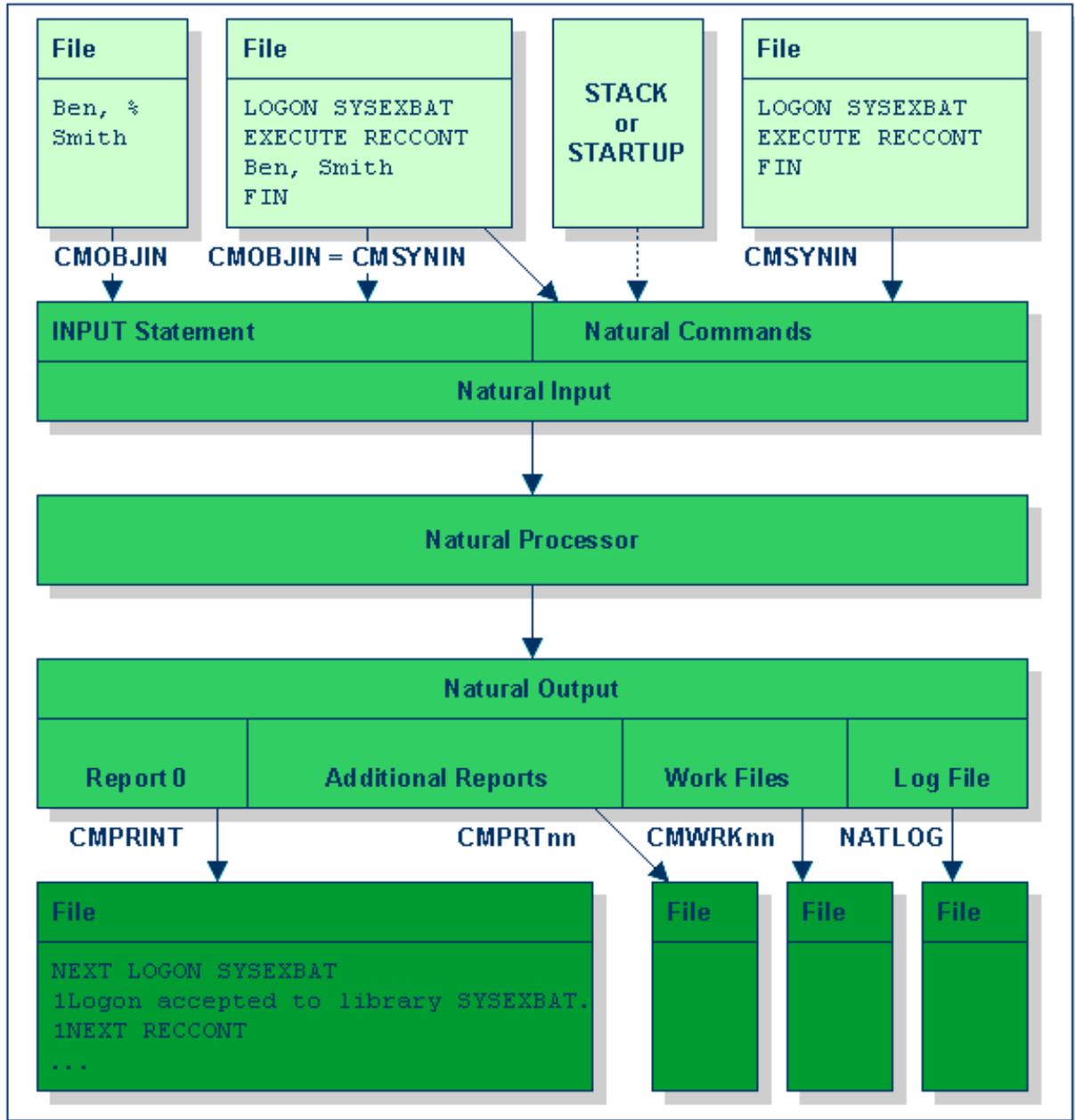
### Contents of Batch Output File - CMPRINT=batch.out - after Execution:

```
_NEXT LOGON SYSEXBAT
1Logon accepted to library SYSEXBAT.
1NEXT EXECUTE RECCONT
1#firstname          #lastname
_DATA Ben,%
_DATA Smith
1Page      1          99-06-18  10:17:11
0#FIRSTNAME #LASTNAME
```

```

-----
0Ben      Smith
1NEXT FIN
1NAT9995 NATURAL session terminated normally.
    
```

The following image illustrates how Natural reads input and writes output in batch mode.



## Hints for Using Natural Maps and Dialogs in Batch Mode

If an application is designed to run in batch mode as well as in interactive mode, the following considerations should be taken into account.

Within Natural, there are two ways to read input data:

- Using a Map (by using an INPUT statement or the Natural object Map)
- Using a Dialog (by using a Natural Dialog object)

In batch mode, data have to be processed using an INPUT statement, because a dialog does not allow data processing in batch mode. Terminal commands to navigate and control the data are also not supported by a dialog. Nevertheless, a dialog may be executed in batch mode. In this case, however, the dialog must be altered in the following way:

- The dialog attribute "visible state" must be turned off (state equals not visible).
- Within the event AFTER-OPEN, code should be inserted to read data during batch mode processing. If Natural runs in batch mode, an INPUT statement should be coded to get the input data. For interactive mode, the "visible state" has to be turned on to make the dialog visible.
- If there is a CLOSE event, ensure that the appropriate code does not contain any GUI actions in batch mode.

### Example for Event AFTER-OPEN:

```
IF *DEVICE EQ "BATCH" THEN
  /* Batch Mode Processing: Call a Map */
  INPUT USING MAP "BATCHINP" #p1 #p2 #p3

  /* ... further data processing ... */

  /* Close dialog immediately */
  CLOSE DIALOG *DIALOG-ID
ELSE
  /* Interactive Mode Processing: Make dialog visible */
  #DLG$WINDOW.VISIBLE = TRUE
END-IF
```

### Example for Event CLOSE:

```
IF *DEVICE NE "BATCH" THEN
  /* ... GUI actions ... */
END-IF
```

# Natural Output Window Information

- Output Window Features
- Output Window Profiling
- Additional Information on Fonts

Whenever a Natural program writes output to the screen, it is displayed in the output window.

---

## Output Window Features

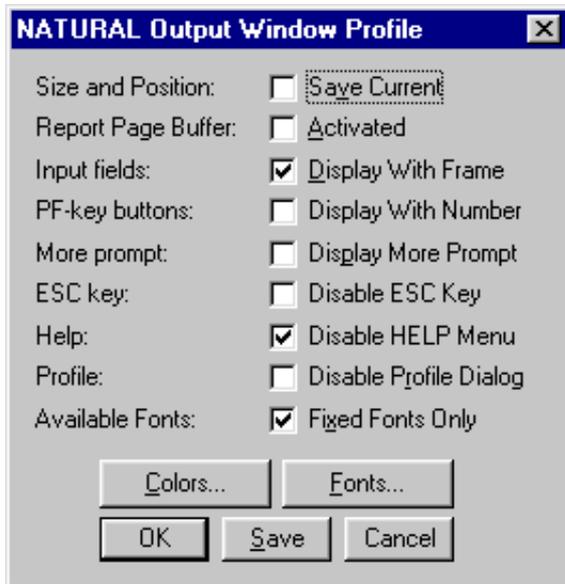
- The output window is sizeable and moveable.
- If the window size is less than the Natural output page, scroll bars appear.
- PF keys defined in a Natural program are converted into pushbuttons. These pushbuttons can be used or you can continue to use the keyboard PF keys.
- Windows created using the Natural terminal command "W" are placed into the output window. They are moveable, sizeable, and scrollable child windows of the output window.
- Double clicking the left mouse button simulates the ENTER key. The system variables \*CURSOR, \*CURS-COL, and \*CURS-LINE will be set to the current mouse position.

## Output Window Profiling

### To profile the output window

1. Select the system menu by clicking the upper left button of an output window.
2. Choose the "Profile" option on the system menu.

The following window appears:

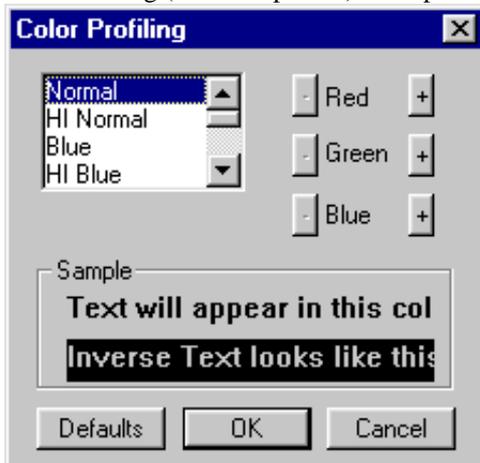


<b>Size and Position:</b>	If you select this option and select the "Save" button, the current size and position of the output window will be saved for future Natural sessions.
<b>Report Page Buffer:</b>	If you select this option, you activate buffering that will accommodate approximately 250 lines of Natural output. Input empties the report page buffer.
<b>Input fields:</b>	If you select this option, you display all input fields with a border (frame).
<b>PF-key buttons:</b>	If you select this option, the PF key button will contain the number of the associated PF key. The name of the PF key is displayed below the PF key button. If you do not select this option, the PF key button will contain the name of the PF key, and the number is not visible.
<b>More prompt:</b>	If you select this option, you activate the MORE prompt for output generated by the Natural statements DISPLAY, WRITE or PRINT.
<b>ESC key:</b>	If you select this option, you disable the ESC key. The end user will then not be able to use the ESC key to quit the current Natural program.
<b>Help:</b>	If you select this option, the Natural output window will no longer display a Help menu entry. If you do not define any other menu entries, the Natural output window will no longer display a menu bar.
<b>Profile:</b>	If you select this option, you will no longer be able to invoke (and change option settings) in the current dialog box ( <i>Profile</i> option on the system menu).
<b>Available Fonts:</b>	If you select this option, you restrict the font selection dialog to the use of fixed character width fonts only.

## Color Profile

### ▶ To change the color profile

1. Choose the "Colors..." button in the "Natural Output Window Profile" dialog box.  
The following (Natural-specific) color profiling window appears:

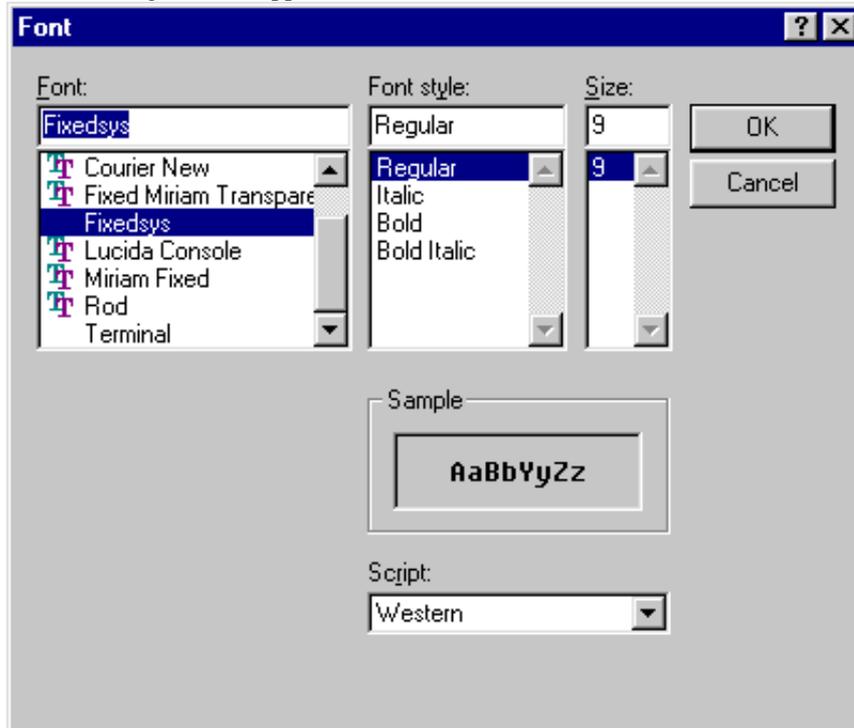


- Colors available to Natural can be altered, using this color profiling window.
2. Select the required color.  
Make the required alterations using the + or - buttons for the Red, Green or Blue portion of this color.
  3. Choose OK.

## Font Profile

### ▶ To change the font profile

1. Choose the "Fonts..." button in the "Natural Output Window Profile" dialog box.  
The following window appears:



The font profile enables the selection of the font being used for all Natural output text in the output window.

2. Select the required font *Name*, *Style*, and *Size* in the respective list box.

**Note:**

If proportional fonts are required, ensure that the option "Available Fonts", as displayed in the output window profile, has not been marked.

3. Choose OK.

## Saving Profile Changes

### ▶ To accept changes made to the profile settings

1. Choose OK in the Natural Output Window Profile dialog box.  
The changes are valid for the duration of the Natural session.
2. Choose "Save".  
The changes are valid also for future Natural sessions.

## Additional Information on Fonts

If a special font has been defined for a map and you define the same font for the Natural output window, the map will be displayed with this font.

# Natural Startup Errors

1	No error occurred.
2	Terminal Control String (TCS) capability specified in SAGtermcap or Environment Variable NATTCHARSET.
3	Failed to initialize character conversion table.
4	Failed to load character conversion table.
5	Error reading the DATABASE assignments in the global configuration file.
6	Unable to find FNAT or FUSER. Check your configuration files.
7	Cannot initialize LFILE table.
12	Unable to read specified parameter module. Please verify the parameter file.
13	Unable to read parameter module.
14	Storage manager initialization failed.
15	End of file (EOF) encountered while reading from STDIN.
16	Unable to open buffer pool; contact the NATURAL system administrator.
17	Error reading the buffer pool assignments in the local configuration file.
18	Invalid FDIC assignment.
19	Invalid FNAT assignment.
20	Invalid FSEC assignment.
21	Invalid FUSER assignment.
22	Unable to load NATURAL login module.
23	Unable to allocate memory for local data. Reduce USIZE and/or SSIZE parameter.
24	Unable to load NATURAL display module.
25,26	Error loading shareable image or DLL.
27	Login cancelled. NATURAL terminates.
28	Security violation during start of NATURAL. NATURAL terminates.
29	Security violation during start of NATURAL. Login aborted due to too many login failures.
30	Natural system error message raised.
31	This is not a security nucleus.
32	Password check failed.
33	Lock manager cannot create/initialize semaphores.
34	Unable to build NATURAL Library structure. Check your System File assignment. User/group settings might not be correct. S-bit setting might be required. Path specifications in global configuration file might be corrupted. LIBDIR.SAG might be missing.
35	Internal wfc i/o terminal driver error.
36	Internal XVT error.
37	DCOM Startup error.

38	Creation of runtime context failed.
39	NATDIR and/or NATVERS not set.
40	NATURAL zmodem error.
41	Creation of TF table failed because there are entries with different database types from older parameter module. Check parameter module with Natural Configuration Utility.
42	Batch mode driver error.
43	Screen window size is too small.
44	Exit from SQL signal handler.
45	Unable to load add-on product.
46	Unable to access FNAT library SYSLIB. Insufficient privilege or file protection violation.
47	Unable to read PARM_PATH.
48	Unable to read NATURAL.INI for CONFIG_NAME.
49	Unable to read NATURAL.INI for NATTCAP.
50	Unable to read NATURAL.INI for NATCONV.
51	Unable to read TMP_PATH.
52	Unable to read PROFILE_PATH.
53	Unable to open NATURAL.INI.
54	Unable to read NATCONF.CFG for NATOSDEP.
55	Unable to read NATURAL.INI for NATEXTLIB.
56	Unable to find NATDIR in SAG.INI.
57	Unable to find DEFAULT-VERSION in SAG.INI.
58	Unable to find NATINI in SAG.INI.
59	Invalid option specified.
60	Not enough memory to initialize internal tables.
61	Batch error occurred, but processing continued due to CC=ON parameter.
62	More than one Natural session with active repository not allowed
63	Natural session with active repository already running

**Note:**

Under Windows, you must run nderun.exe (as opposed to naturalr.exe) in order to receive the return code.

# Issuing System Commands from a Natural Program

The Natural user exit "SHCMD" can be used to issue a system command within a Natural program.

Format:

**CALL 'SHCMD' *command* [*option*]**

Parameters:

*command*

**Command** is to be executed by the operating system. To execute commands (like DIR for directory or DEL for delete) you have to specify the system command interpreter as well. For more information, see examples below.

*option*

**Note:**

This parameter is optional.

**Option** describes how the command should be executed.

The following options are available:

- ASYNCH
- NOSCREENIO
- SYNCH (This is the default option.)

Return Codes:

The following return code values are available:

Return Code	Description
0	Command successfully executed
4	Illegal SHCMD parameter specified
All others	Operating-system-dependent error code

## Parameter Options

### ASYNCH

If option 'ASYNCH' is set, Natural does not wait until the command is completely executed. This kind of processing is named asynchronous processing.

### NOSCREENIO

Option 'NOSCREENIO' is used to hide the output generated by the command. The hidden output is redirected to the null device.

### SYNCH

If option 'SYNCH' is set, Natural waits until the command is completed. This kind of processing is named synchronous processing and is set by default.

#### Note:

Option "ASYNCH" and "SYNCH" may be not set at the same time.

#### Examples:

Executing system command DIR to view a directory:

```
CALL "SHCMD" "CMD.EXE /C DIR"
```

Retrieve the return code by using the Natural function RET:

```
RESET rc (I4)

CALL "SHCMD" "CMD.EXE /C DIR"

rc = RET( "SHCMD" ) /* retrieve return code

IF rc <> 0 THEN /* in case of an error

DISPLAY "Error occurred during SHCMD" /* display an error message
```

Execute a command which includes blanks within the command by enclosing the command with quotation marks. The following example executes Microsoft Excel.

```
RESET #cmd (A253)

MOVE ' "C:\Program Files\Microsoft Office\Office\EXCEL.EXE" ' to #cmd

CALL "SHCMD" #cmd "ASYNCH"
```

In this case, parameter TQ (translate quotation marks) has to be OFF, otherwise the quotation marks have been removed. To be independent from the TQ parameter, use the hexadecimal ASCII code of quotation marks (H'22') and append it at the beginning and the end of the command. The following example demonstrates this:

```
RESET #cmd (A253)
```

```
MOVE H'22' - "C:\Program Files\Microsoft
```

```
Office\Office\EXCEL.EXE" - H'22' to #cmd
```

```
CALL "SHCMD" #cmd "ASYNCH"
```

# Natural Profile Parameters

This document describes how and where you can use Natural profile parameters to define various characteristics of the Natural environment. The following general topics are covered:

- Parameter Hierarchy
- Runtime Assignment of Parameter Values
- Dynamic Assignment of Parameter Values
- Static Assignment of Parameter Values

See also:

- Profile Parameters Grouped by Functions
- Profile Parameters (Detailed Descriptions)
- Natural Configuration Utility (Usage)

**Note:**

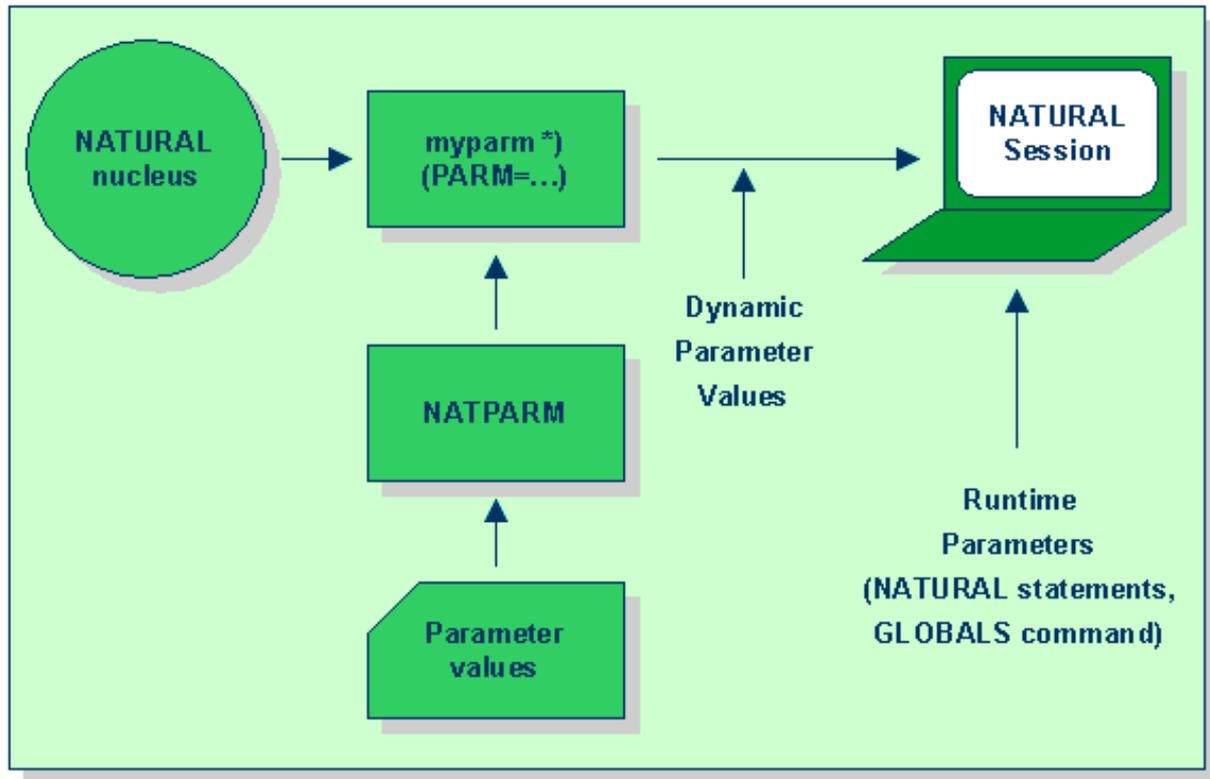
This document covers the use of Natural profile parameters in a Windows environment. If you are using Natural for Windows for remote development in conjunction with Natural's Single Point of Development (SPoD), see also the documents referring to profile parameter usage in the corresponding platform-specific Natural Operations documentation.

---

## Parameter Hierarchy

The values for the Natural profile parameters are taken from the three sources. Their priority is explained below:

1. Runtime assignment of session parameters using the Natural SET GLOBALS statement or the GLOBALS system command (highest priority).
2. Dynamic assignments which are valid for the current Natural session. These are made by specifying individual parameters and/or an alternative parameter file when starting Natural.
3. Static assignments, which are made by parameters specified in the Natural parameter file NATPARAM (lowest priority).



\* The file "myparm" represents an alternative user-defined parameter file, which overrides the settings in NATPARAM.

## Runtime Assignment of Parameter Values

A value can be assigned to a profile parameter at runtime if "Specification within Session" is specified as "YES" in the description table of the parameter:

<b>Possible values</b>	
<b>Default value</b>	
<b>Dynamic specification</b>	
<b>Specification within session</b>	YES

The parameter keyword and the required value are entered as session parameter following the Natural SET GLOBALS statement or GLOBALS system command (for detailed information on Session Parameters, see the Natural Reference documentation).

### Examples:

```
SET GLOBALS SA=ON, IM=D
```

```
GLOBALS SA=ON, IM=D
```

### Note:

The SET GLOBALS statement can be used in reporting mode only.

## Dynamic Assignment of Parameter Values

Dynamically assigned parameter values are used to override Natural profile parameter settings, as specified in the default parameter file NATPARM, for the duration of a single Natural session, and/or to select a specific system profile that is to be in effect for the Natural session.

A value can be assigned dynamically to a profile parameter if "Dynamic Specification" is specified as "YES" in the description table of the parameter:

<b>Possible values</b>	
<b>Default value</b>	
<b>Dynamic specification</b>	YES
<b>Specification within session</b>	

The setting of dynamic parameters enables a certain environment to be set up when starting Natural. The values for the dynamic parameters are passed by the operating system to Natural when the session is started.

### To specify dynamic parameter values

1. Select the Natural program-item icon.
2. Click the right mouse button and, from the resulting context menu, choose **Properties**.  
The "Natural Properties" dialog box appears.
3. Choose the "Shortcut" tab.
4. In the "Target" field, enter the desired dynamic parameters and their values after the Natural path.

#### Example:

```
C:\ProgramFiles\Software AG\Natural\v.r.s.\bin\natural.exe PARM=MYPARM SM=ON DTFORM=I
```

- where *v.r.s.* is the Natural *version, release, system maintenance level* number.

The parameter file MYPARM is to be used for the Natural session. The values for the profile parameters SM and DTFORM are dynamically specified.

## Static Assignment of Parameter Values

By default, the parameter specifications in the parameter file NATPARM are used to determine the characteristics of your Natural environment. Initially, the NATPARM parameter file contains the default values as supplied by Software AG.

If you wish to use Natural with parameter values other than the system defaults, use the Natural Configuration Utility. With this utility, you can

- modify the default parameter file NATPARM and/or
- create your own parameter files.

See Changing Parameter Settings in the Natural Configuration Utility description.

# Profile Parameters Grouped by Function

The individual parameters contained in the NATPARM parameter file (or an alternate parameter file) can be changed in the Natural Configuration Utility. They are divided into groups according to their functions.

If you expand the tree item of the NATPARM parameter file, a list of the following parameter groups is displayed:

- Database Management
- Natural Execution Configuration
- Natural Development Environment
- Product Configuration
- Client/Server

See also:

- Profile Parameter Usage
- Profile Parameters (Detailed Descriptions)
- Natural Configuration Utility (Usage)

## Note:

This document covers the use of Natural profile parameters in a Windows environment. If you are using Natural for Windows for remote development in conjunction with Natural's Single Point of Development (SPoD), see also the documents referring to profile parameter usage in the corresponding platform-specific Natural Operations documentation.

---

## Database Management

The following sections provide an overview of the parameters contained in the individual groups.

- General Parameters
- Adabas Specific
- Administrator DBMS Assignment
- User DBMS Assignment

See also:

DBMS Assignments (in Global Configuration File - NATCONF.CFG)

## General Parameters

Parameter	Function
<b>DBUPD</b>	Database updating.
<b>ET</b>	Execution of END/BACKOUT TRANSACTION statements.
<b>OPRB</b>	Adabas open/close processing.

## Adabas Specific

If Natural is used with Adabas, the following parameters should be reviewed and, if necessary, the default values should be adjusted to meet your specific requirements:

Parameter	Function
<b>ETID</b>	Adabas user identification.
<b>LDB</b>	DB Time Limit.
<b>WH</b>	Record hold processing.

## Administrator DBMS Assignment

The following parameters are used to assign database management system settings:

Parameter	Function
<b>LFILE</b>	Administrator logical files.
<b>TF</b>	Translation of file number.
<b>XADB</b>	XA databases.

## User DBMS Assignment

The following parameters are used to assign database management system settings:

Parameter	Function
<b>ETDB</b>	Database for transaction data.
<b>LFILE</b>	Dynamic specification of a user logical file.
<b>LFILMAX</b>	Maximum number of logical files.
<b>UDB</b>	User database ID.

## Natural Execution Configuration

- Batch Mode
- Buffer Sizes
- Character Assignments
- Command Execution
- Date Representation
- Device/Report Assignments
- Error Handling
- Field Appearance
- Limits
- Program Loading/Deletion
- Regional Settings
- Report Parameters
- Steplibs
- System Files
- System Variables

- Workfiles

## Batch Mode

For space considerations, the parameters affecting the batch mode behavior of Natural are arranged on three tabs: Channels, Appearance and Frame Characteristics. In the table below, these parameters are summarized in alphabetical order.

Parameter	Function
<b>BATCH</b>	Enable Batch Mode Simulation
<b>BATCHMODE</b>	Enable Real Batch Mode
<b>BMBLANK</b>	Display Trailing Blanks
<b>BMCONTROL</b>	Display Control Characters
<b>BMFRAME</b>	Frame Characters
<b>BMSIM</b>	Similar Output
<b>BMTIME</b>	Display Process Time
<b>BMTITLE</b>	Display Window Title
<b>BMVERSION</b>	Display Natural Version
<b>CC</b>	Enable Error Processing
<b>CMOBJIN</b>	Input Data Channel
<b>CMPRINT</b>	Output Channel
<b>CMSYNIN</b>	Input Commands Channel
<b>ECHO</b>	Display Input Data
<b>ENDMSG</b>	Session End-Message
<b>NATLOG</b>	Natural Log File

See also:

- Natural in Batch Mode

## Buffer Sizes

Natural uses several buffer areas for the storage of programs and data. You may need to adjust their sizes in order to achieve maximum buffer efficiency.

Parameter	Function
<b>EDTBPSIZE</b>	SAG Editor Bufferpool Size.
<b>EDTLFILES</b>	SAG Editor Logical Files.
<b>SORTSIZE</b>	Size of sort buffer area.
<b>SSIZE</b>	Source area size.
<b>USIZE</b>	Size of user buffer area.

See also:

- Buffer Pool Assignments (in Local Configuration File - Natural.INI)

## Character Assignments

The following parameters are used to change default character assignments:

Parameter	Function
<b>CF</b>	Control character for terminal commands.
<b>CLEAR</b>	Processing of CLEAR key at runtime.
<b>DC</b>	Character to be used as decimal point.
<b>FC</b>	Filler character for maps generated with an INPUT statement.
<b>HI</b>	Character to invoke field- or map-related help.
<b>IA</b>	Input assign character.
<b>ID</b>	Input delimiter character.

Once a character has been defined to replace a default character, this character cannot be used as data.

## Command Execution

The following parameters are used to control the execution of commands:

Parameter	Function
<b>CM</b>	Command mode allowed.
<b>ESCAPE</b>	Enable % %.
<b>NC</b>	Control use of Natural system commands.
<b>RECAT</b>	Dynamic recataloging.

## Date Representation

The following parameters are used to control the date representation:

Parameter	Function
<b>DFOUT</b>	Date Format on Output
<b>DFSTACK</b>	Date Format in STACK
<b>DFTITLE</b>	Date Format in Report Titles
<b>DTFORM</b>	Date Format
<b>YSLW</b>	Year Sliding Window

## Device/Report Assignments

The "Devices" parameter group allows you to modify your screen and printer configurations as well as your report assignments.

Parameter	Function
MAINPR	Override default report number.
Devices	Output devices (video, printers)
Reports	Report assignments

See also profile parameter CMPRTnn which is used for additional reports in batch mode.

### Devices

The Devices group shows a scrollable list of configurable logical devices (VIDEO and the logical printers, LPT1 to LPT31), as used in the DEFINE PRINTER statement. The following information may be changed:

- **Method**

The buttons in this column display the print method used for the corresponding print device. Repeated clicking of the button rotates through the available print methods, which are:

<b>TTY</b>	This is the raw printing mode where the text output by the Natural program is essentially forwarded to the spooler in unaltered form. Any printer commands must either be output as data by the Natural program or statically defined in TTY printer profiles. The Windows printer driver is not used.
<b>GUI</b>	This is the preferred print method under Windows, whereby the output from the Natural program is passed through the Windows printer driver associated with the specified printer. Unlike the TTY method, no knowledge of printer commands is required, since these are automatically inserted by the printer driver.

- **Line Size**

See LS Session Parameter value.

- **Page Size**

See PS Session Parameter value.

- **Max Pages**

See MP Session Parameter value.

- **Setup**

The buttons in this column invoke a print method-specific dialog allowing the specification of the printer and the setting of any method-specific options, as follows:

**Print Method TTY:**

In the TTY Setup dialog, you can also select a "Physical Specification", which corresponds to the printer name. Instead of selecting an existing physical printer specification from the drop-down list, you can also enter a file name if you want your output to be written to a file. As with work files, such a file name can be defined by using environment variables. Any existing file of the same name at the specified location is normally overwritten, unless the entered file name is immediately prefixed by two right-caret (>) characters. Note that a server printer can be specified via the UNC naming convention (\\server-name\printer-name) In addition, a check box is provided, allowing the user to indicate whether file I/O should be used to access the specified device (e.g., for printers which are set up as File System shares. If this option is unchecked, the data is output via spooler API function calls.

**Print Method GUI:**

The dialog shown in this case is the standard Windows Print Setup dialog, which allows specification of both the Windows printer and the associated print options such as page size and orientation. Note, however, that if you wish to make use of the default Windows printer on each client computer at run-time, this dialog should not be used.

Alternatively, if a specific printer has already been chosen (even in a previous session) for this logical device, the default printer can be implicitly re-selected by switching the print method to TTY, closing the application, re-opening the application and switching the print method back to GUI.

Note that any print options for the default printer must be set by using a printer profile.

## Reports

In this section, you can assign a Natural report number (Report 1 to Report 31) to a logical device name. Possible values for the output medium are: VIDEO, LPT1 to LPT31, SOURCE (source area), DUMMY and INFOLINE.

Report Number 0 must be set to VIDEO and is not reassignable; no report number other than 0 can be assigned to VIDEO.

In addition to the logical device name, you can assign a printer profile as defined in the global configuration file. All defined printer profiles are listed for selection in the "Profile" combo box. Select "blank" if you do not want to use any of these profiles.

Note that any specified printer profile must match the print method used at run-time. For example, if an attempt is made to use a GUI printer profile for a TTY logical device (or vice versa), the printer profile specification is ignored.

## Error Handling

The following parameters are used to control error handling within Natural.

Parameter	Function
<b>IKEY</b>	Error processing for PA/PF keys.
<b>MSGSF</b>	Display system error messages in full.
<b>REINP</b>	Issue internal REINPUT statement for invalid data.
<b>SA</b>	Sound terminal alarm.
<b>SNAT</b>	Sound bell in the case of a syntax error.
<b>ZD</b>	Zero division.

### Note:

You can use the profile parameter NOAPPLERR to suppress the message number prefix "NAT" in user application errors. This parameter can only be specified dynamically, therefore, it cannot be modified or viewed within the Natural Configuration utility.

## Field Appearance

Parameter	Function
<b>CVMIM</b>	Control variable modified at input.
<b>FCDP</b>	Filler character for dynamically protected input fields.
<b>LC</b>	Enable lower case.
<b>NENTRY</b>	Entry of numeric fields.
<b>OPF</b>	Overwriting of protected fields by help routines.
<b>PM</b>	Print mode.
<b>ZP</b>	Zero printing.

## Limits

The following parameters are used to prevent a single program from consuming an excessive amount of internal resources:

Parameter	Function
<b>LE</b>	Limit for error processing.
<b>LT</b>	Processing loop limit.
<b>MADIO</b>	Maximum number of DBMS calls.
<b>MAXCL</b>	Maximum number of program calls.
<b>SD</b>	System time delay.

## Program Loading and Deletion

The following parameters are used to control the dynamic loading and deletion of programs:

Parameter	Function
<b>BPSFI</b>	Search first in buffer pool.
<b>CDYNAM</b>	Dynamic loading of non-Natural programs.
<b>DYNPARAM</b>	Dynamic parameters.
<b>ETA</b>	Program to receive control after error in transaction.
<b>PRGPAR</b>	Data to be passed to the program defined by the parameter PROGRAM.
<b>PROGRAM</b>	Program to receive control after Natural termination.
<b>ROSY</b>	Disable storage of Natural programs.
<b>STACK</b>	Place data on Natural stack.

## Regional Settings

The following parameters are used to control the country or region specific settings of Natural:

Parameter	Function
<b>DD</b>	Day differential.
<b>TD</b>	Time differential.
<b>ULANG</b>	User language.

## Report Parameters

The following parameters control various attributes of Natural reports:

Parameter	Function
<b>EJ</b>	Page eject control.
<b>IM</b>	Default input mode.
<b>LS</b>	Line size.
<b>PS</b>	Page size.
<b>SF</b>	Spacing factor between fields.

## Steplibs

Parameter	Function
<b>USER</b>	User ID.
<b>LSTEP</b>	Natural steplibs.
<b>STEPLIB</b>	STEPLIB Table

## System Files

The following parameters are used to specify the Natural system files:

Parameter	Function
<b>FDIC</b>	Predict system file.
<b>FNAT</b>	Natural system file for system programs.
<b>FSEC</b>	Natural Security system file.
<b>FUSER</b>	Natural system file for user programs.

See also:

- System File Assignments (in the section Configuration Files)

## System Variables

The following parameters are used to adjust Natural system variables for the start of a Natural session:

Parameter	Function
<b>AUTO</b>	Automatic logon.
<b>INIT-LIB</b>	Startup library.
<b>STARTUP</b>	Startup program.

See also:

- System Variables (Alphabetical List of Variables, Date and Time System Variables included in the Natural Reference documentation)

### Note:

You can use the command line parameters NATVERS to specify the Natural version and PARM to specify a specific Natural parameter file at session startup. These parameters can only be specified dynamically, therefore, they cannot be modified or viewed within the Natural Configuration utility.

## Workfiles

The following parameters can be used to specify work file settings:

Parameter	Function
<b>WFOPFA</b>	Work file to be opened on first access.
<b>WORK</b>	Maximum number of work files.
<b>Work files</b>	Natural work files.

See also:

- Work Files (Purpose)
- Defining Work Files
- Work File Formats
- Special Considerations on Work Files with Extension NCD
- CMWRKnn (Profile parameter for Natural work files in batch mode)

### Note:

You can use the Natural profile parameter `TMPSORTUNIQ` to specify an alternate naming convention for sort work files. This parameter can only be specified dynamically, therefore it can not be modified or viewed within the Natural Configuration utility.

## Natural Development Environment

- Compiler Options
- Remote Debugging

### Compiler Options

The following parameters are used to set options for the Natural compiler:

Parameter	Function
<b>CO</b>	Compiler Output
<b>DBSHORT</b>	Interpretation of Database Short Names
<b>DSLIM</b>	Data Size Limitation
<b>DU</b>	Memory Dump Generation
<b>ENDIAN</b>	Endian mode
<b>FS</b>	Length/Format Specification
<b>GFID</b>	Generation of Global Format IDs
<b>KC</b>	Keyword Checking
<b>SM</b>	Structured Mode
<b>SYMGEN</b>	Generate Symbol Tables
<b>SYNERR</b>	Syntax Error Control
<b>TQ</b>	Translate Double Quotes
<b>V22COMP</b>	Natural Version 2.2 Compatibility Option
<b>XREF</b>	Active Cross References

## Remote Debugging

The following parameters are used to allow for remote debugging:

Parameter	Function
<b>RDACTIVE</b>	Activate remote debugger.
<b>RDNODE</b>	Remote debugger node name.
<b>RDPORT</b>	Remote debugger port.

See also:

- Local and Remote Debugging
- Using the Natural Debugger

## Product Configuration

- Entire Transaction Propagator
- Entire System Server Interface

### Entire Transaction Propagator

The following parameters are used in conjunction with Software AG's Entire Transaction Propagator (ETP).

Parameter	Function
<b>ETPDB</b>	Database list for Entire Transaction Propagator.
<b>ETPSIZE</b>	Size of buffer for Entire Transaction Propagator.

## Entire System Server Interface

The following parameters are used in conjunction with Software AG's Entire System Server Interface (ESX).

Parameter	Function
ESXDB	Database ID used for Entire System Server DDM's.

## Client/Server

- DCOM Support
- Remote Dictionary Access
- Remote Procedure Call

### DCOM Support

The following parameters are used to provide DCOM support:

Parameter	Function
ACTPOLICY	Activation policy.
AUTOREGISTER	Automatic update of registry.
COMSERVERID	Server name.

### Remote Dictionary Access

The following parameter is used for remote dictionary access:

Parameter	Function
USEDIC	Remote dictionary access.
USEREP	Enable Usage of Repository.

### Remote Procedure Call

For space considerations, the parameters to select options in the Natural Remote Procedure Call (RPC) are arranged on several tabs RPC General, Client, Server and RDS. In the table below, these parameters are summarized in alphabetical order.

The following parameters are used for :

<b>Parameter</b>	<b>Function</b>
<b>ACIPATT</b>	Node pattern for ACI.
<b>ACIVERS</b>	ACI Version.
<b>AUTORPC</b>	Automated remote execution.
<b>COMPR</b>	Send buffer compression.
<b>CP</b>	Code Page.
<b>CSCPATT</b>	Node pattern for CSCI.
<b>DFS</b>	Default server.
<b>LOGONRQ</b>	Logon required for server request.
<b>MAXBUFF</b>	Request buffer size.
<b>RDS</b>	Remote directory servers.
<b>RPCSIZE</b>	RPC buffer size.
<b>SERVER</b>	Start session as RPC server session.
<b>SRVNAME</b>	Server name.
<b>SRVNODE</b>	Server node.
<b>SRVUSER</b>	Server user ID.
<b>TIMEOUT</b>	Request timeout.
<b>TRACE</b>	RPC trace level.
<b>TRANSP</b>	Transport protocol.
<b>TRYALT</b>	Retry Service on alternate server.

# Natural Configuration Utility

This document describes the Natural Configuration Utility which is used to create and modify global and local configuration files (for administrator use only) and parameter files.

The following topics are covered:

- Invoking the Natural Configuration Utility
- Menu Bar Commands
- Changing Parameter Settings
- Performing Configuration Utility Functions from the OS Command Prompt

See also:

- Configuration Files (Global and Local)
- Profile Parameter Usage
- Profile Parameters Grouped by Functions
- Profile Parameters (Detailed Descriptions)

## Note:

This utility covers the use of Natural profile parameters in a Windows environment. If you are using Natural for Windows for remote development in conjunction with Natural's Single Point of Development (SPoD), see also the documents referring to profile parameter usage in the corresponding platform-specific Natural Operations documentation.

---

## Invoking the Natural Configuration Utility

### To invoke the Natural Configuration Utility

- Choose the "Configuration Utility" shortcut from the Natural program group.  
The "Natural Configuration Utility" window appears with the name of the parameter file currently active or being edited and which is displayed in the title bar.

## Menu Bar Commands

The following commands and functions are available on the menu bar:

- File Commands
- Edit Commands
- Help
- Search Function

### File Commands

The "File" commands enable you to carry out the following functions:

Function	Explanation
<b>New</b>	Establishes a new (alternative) file.
<b>Delete</b>	Deletes the selected parameter file.
<b>Save</b>	Saves a parameter file after you have created/modified it.
<b>Save as</b>	Saves a parameter file under a different name.
<b>Rename</b>	Renames the selected parameter file.
<b>Restore Saved</b>	Restores the file to its state upon the last save; any changes made since are reversed.
<b>Save All</b>	This function saves all existing files.
<b>Import</b>	Generates a binary parameter file from a parameter file generated as text file by the "Export" function (see below).
<b>Export</b>	Generates a text file showing the current values of the active parameter file. In text file form, you can transfer a parameter file to another hardware platform, where you have to generate a binary file from the text file to make the parameter file usable by Natural. This generation is done with the "Import" function (see above). The text file may be stored as <i>parameter-file.LST</i> in any directory selected by the user; otherwise, the "Import" function cannot be applied to it. By default, this directory is the "tmp" subdirectory in the Natural root directory.
<b>Exit</b>	This function is used to leave the Natural Configuration Utility.

## Edit Commands

The "Edit" option is used for the following functions:

Function	Explanation
<b>Copy</b>	Copy a selected parameter file to the clipboard.
<b>Paste</b>	Add the copied elements to the treeview.

## Help

The menu bar command **Help** function provides a general help:

Function	Explanation
Help	Invokes the general help text for the Natural Configuration Utility.
About ...	Displays the version of the Natural Configuration Utility.

In addition, a context-sensitive help is available which provides detailed information on the currently highlighted tree-view item, for example, "Global Config File".

### To invoke context-sensitive help

- Press the **F1** button.  
The associated help text is displayed, for example, a description of the global configuration file NATCONF.CFG.

## Search Function

The combo box on the tool bar contains a list of the various Natural profile parameters. You can use it to locate a parameter in a utility dialog. The search is always performed in the currently selected parameter file.

### ▶ To find a parameter in a dialog

1. Click on the down arrow on the right of the combo box.  
A box appears with a selection list of all existing Natural profile parameters.
2. Select the desired parameter from this list.  
The corresponding dialog is displayed.

## Changing Parameter Settings

### ▶ To change parameter settings, proceed as follows:

1. Select the parameter file in the treeview.
2. Expand the file item and select the desired parameter.  
The parameter appears in a dialog on the right hand side of the screen.
3. Modify the desired parameters.

### ▶ To save the modifications to the current parameter file

- From the **File** menu, choose **Save** to save the changes to the current parameter file.

### ▶ To save the modifications to a new (alternative) parameter file

- From the **File** menu, choose **Save As** to save the modified parameter file settings to a new (alternative) parameter file named "NEW1".  
You then are prompted to rename the file.

### ▶ To invoke Natural with the new (alternative) parameter file

- At the operating system command prompt, enter the command:  
`Natural PARM=parameter-file`

### ▶ To invoke Natural Runtime with the new (alternative) parameter file

- At the operating system command prompt, enter the command:  
`Naturalr PARM=parameter-file`

## Performing Configuration Utility Functions from the OS Command Prompt

The following functions can also be performed from the "Target" text box of the "Properties" of the Natural Configuration Utility program-item icon:

- exit
- import=
- export=
- natvers=
- parm=
- save

- save=

 **To perform a function from the "Target" text box**

1. Select the Natural Configuration Utility program-item icon
2. Click the right mouse button and choose **Properties** from the context menu.  
The Natural Configuration Utility Properties dialog box appears.
3. Enter the desired function in the "Target" field.  
Multiple functions can be specified one after the other (see the examples below).

**Examples:**

```
natparm parm=parameter-file
```

With this command, you invoke the Natural Configuration Utility and at the same time carry out the "File Selection" function: the specified *parameter-file* is selected and made available for editing.

```
natparm parm=parameter-file save=new-parameter-file-name
```

With this command, you invoke the Natural Configuration Utility and at the same time carry out the "File Selection" and "Save" functions of the Natural Configuration Utility. The specified *parameter-file* is selected and saved under a *new-parameter-file-name*.

```
natparm import=parameter-file save
```

With this command, you invoke the Natural Configuration Utility and at the same time perform the "Import" and "Save" functions of the Natural Configuration Utility. The specified *parameter-file* is imported and saved.

# Providing Natural Applications

This document describes how to provide a Natural application for loading into another environment. The term "Natural Application" used in this document concerns the conventional type of application and does not refer to the new application concept introduced with Natural Single Point of Development (SPoD).



Please note that the utility SYSPAUL is going to be withdrawn soon. Its functionality will be included in the Natural utility SYSOBJH.

- SYSPAUL Utility
  - Invoking SYSPAUL
  - Define Application Description
  - Unload Application
  - Package Application
  - Load Application File
  - Scan Application File
- 

## SYSPAUL Utility

The Natural utility SYSPAUL allows you to define application descriptions, to unload and package (for example, on diskette) entire Natural applications, and to load and scan individual application files. Applications are unloaded/packaged in exactly the same directory structure they are to represent when loaded into a target environment. In addition to Natural objects, non-Natural objects can also be part of such an application.

The SYSPAUL utility is used to:

- unload Natural stowed objects, cataloged objects, saved objects, DDMs and error messages from your current Natural environment into a Natural application file;
- unload non-Natural objects as, for example, bitmaps or help files from a specified library or directory into a target Natural environment;
- create delete instructions for the deletion of Natural stowed objects, cataloged objects, saved objects, DDMs and error messages from a target Natural environment;
- unload Natural application files (including delete instructions) into a specified directory;
- package unloaded Natural application files, for example, on diskette;
- load unloaded and packaged Natural application files into a target environment;
- scan application files to be loaded without actually loading them.

### Note:

Any functionality of the SYSPAUL utility can be restricted by using the user exit PAUL-E1S provided in the library SYSPAUL; for further details, refer to the information provided in the PAUL-E1S source.

## Invoking SYSPAUL

You invoke the SYSPAUL utility by entering Natural direct commands in either of the following ways:

- enter the command SYSPAUL,
- enter the command LOGON SYSPAUL first and then the command MENU.

When you invoke SYSPAUL, the "Providing Applications" dialog box appears offering you the following functions:

- Define Application Description
- Unload Application
- Package Application
- Load Application File
- Scan Application File

All functions can be chosen either from the "Function" menu located in the menu bar or from the "Function" group frame via option buttons.

In addition to the OK button, the "Providing Applications" dialog box contains the following buttons:

- "Exit" button to exit the SYSPAUL utility;
- "Help" button, which can be used as an alternative to the "Help" option of the menu bar to invoke a help text.

## Define Application Description

- Description
- Application Files
- Instructions
- Options
- Further Components of the Define Application Description Window
- Adding and Modifying Application Files
- Adding and Modifying Unload Instructions for Natural Objects
- Adding and Modifying Unload Instructions for Non-Natural Objects
- Unloading Natural Applications
- Packaging Natural Applications

When you choose the "Define Application Description" function, the "Define Application Description [*description-name*]" dialog box appears.

The menu bar of this dialog box provides the following menus:

Menu	Explanation
<b>Description</b>	Is used to maintain application descriptions for the unloading and packaging of Natural applications.
<b>Application Files</b>	Is used to define application files for the Natural objects to be unloaded and/or packaged.
<b>Instructions</b>	Is used to specify the Natural and/or non-Natural objects to be unloaded and/or packaged.
<b>Options</b>	Is used to specify where a newly created application file or unload instruction line is to be added to the existing list of application files or unload instruction lines.
<b>Help</b>	Invokes a help text that corresponds to the one invoked by choosing the "Help" button.

## Description

When you select "Description", a menu containing the following functions is displayed:

Function	Explanation
<b>New</b>	Indicates that a new application description is to be created; the "Define Application Description" dialog box becomes empty, except the: <ul style="list-style-type: none"> <li>- "Application Files Directory" text box, which by default contains the path name of the Natural TMP directory;</li> <li>- "Application Name" text box, which is "untitled";</li> <li>- "Application Files" list box, which contains a default application file.</li> </ul>
<b>Open</b>	Displays an existing application description, which has to be selected in a dialog box that appears when you choose this function.
<b>Save</b>	Saves an application description on disk after you have created and/or modified it.
<b>Save as</b>	Saves an application description under a different name, which has to be specified in a dialog box that appears when you choose this function.
<b>Delete</b>	Deletes an existing application description, which has to be selected in a dialog box that appears when you choose this function.
<b>List</b>	Invokes the List Application Description [ <i>description-name</i> ] dialog box that lists an application description.
<b>Print</b>	Prints an application description, see below.
<b>Import</b>	Imports an application description, the name of which has to be specified in a dialog box that appears when you choose this function, see below.
<b>Set</b>	Invokes the Application Load Settings [ <i>description-name</i> ] dialog box in which you can specify several settings that are to apply when loading the corresponding application.
<b>Unload</b>	Generates an application according to the specified description. The application is unloaded to the directory specified in the "Application Files Directory" text box. It consists of all Natural application files (that is, unload instructions for Natural objects) and non-Natural objects specified for unloading during the current application packaging session. When you choose this function, the "Unload Application" dialog box is displayed, in which you can specify several unload options; see below.
<b>Pack</b>	Packs the contents of the unload directory on diskette, automatically using the directory structure that the application is to represent in the target environment and a format that can be used by the Natural setup facility. When you choose this function, the "Package Application" dialog box is displayed, in which you can specify several options for packaging; see below.
<b>Exit</b>	Exits the SYSPAUL utility.

### List Application Description

In the "List Application Description [*description-name*]" dialog box, you can specify whether you want to save the listed application description in a format suitable for the "Import" function. If so, choose the "Save" button; choose the "Cancel" button to leave the application description listing without saving it.

### Print Application Description

In the "Print Application Description [*description-name*]" dialog box, you can specify how many copies (default is 1) of the selected application description you want to be printed. In addition, you can modify your printer specifications by choosing the "Setup" button, which takes you to a printer setup dialog specific to the operating system.

## Import Application Description

In the "Import Application Description" dialog box, you can specify or select the application description you want to import. The specified or selected application description must have been saved in a suitable format by using the "List" function.

## Application Load Settings

In the "Application Load Settings [*description-name*]" dialog box, you can specify:

- Several items for the setup screen of the load procedure.
- The name of the parameter module list provided with the application.  
The list of parameter modules to be specified must have been created with the Natural\_Configuration Utility.
- The full path name of the SETUP.EXE file and the full path name of the SETUP.EXE help file. The SETUP.EXE is used to load Natural applications to a Natural environment. You will find the SETUP.EXE file at the following path:  
%NATDIR%/NATVERS%/Natural/applinst.
- The required disk space for your system files in the target environment (recommended).
- The DBID and type of the database(s) to be used by the application.  
The specified values will be valid for the global configuration file of the target environment.

## Application Files

When you select "Application Files", a menu containing the following functions is displayed:

Function	Explanation
<b>Add</b>	Use this function to add a new application file, the name of which has to be specified in a dialog box that appears when you choose this function. Depending on the setting of the "Options" menu, the new application file is inserted after (default) or before the application file previously selected in the "Application Files" list box.
<b>Modify</b>	Use this function to modify the name of the application file selected in the "Application Files" list box ; an appropriate dialog box appears.
<b>Delete</b>	Use this function to delete the application file selected in the "Application Files" list box with all contained unload instructions.
<b>Import</b>	Use this function to import an application file; an appropriate dialog box appears.

For further information, see Adding and Modifying Application Files.

## Import Application Files

When you choose the "Import" function from the "Application Files" menu, the "Import Application File" dialog box is displayed, in which you specify the name of the application file to be imported and the name of the description in which this application file is contained.

You can select the description name in a further dialog box, which is invoked by choosing the "Select" button.

You can select the application file name from a selection list of all application files contained in the specified description.

Depending on the setting of the "Options" menu, the imported application file is inserted after (default) or before the application file previously selected in the "Application Files" list box. The corresponding unload instructions are placed in the list box of the "Unload Instructions" group frame.

## Instructions

When you select "Instructions", a menu containing the following functions is displayed:

Function	Explanation
<b>Add</b>	Use this function (or choose the "Add" button) to add a new line of unload instructions. An appropriate dialog box appears when you choose this function. Depending on the setting of the "Options" menu, the new line of instructions is added after (default) or before the instruction line previously selected in the list box of the "Unload" Instructions group frame.
<b>Modify</b>	Use this function (or choose the "Modify" button) to modify the line of unload instructions selected in the list box of the "Unload Instructions" group frame; an appropriate dialog box appears.
<b>Delete</b>	Use this function (or choose the "Delete" button) to delete the line of unload instructions selected in the list box of the "Unload Instructions" group frame.

For further information, see Adding and Modifying Unload Instructions for Natural Objects.

## Options

When you select "Options", a menu containing the following options is displayed:

Option	Explanation
<b>Insert Before</b>	Indicates that a newly created application file or line of unload instructions is to be added before the application file or instruction line previously selected in the "Application Files" or "Unload Instructions" list box.
<b>Insert After</b>	Indicates that a newly created application file or line of unload instructions is to be added after the application file or instruction line previously selected in the "Application Files" or "Unload Instructions" list box; this is the default setting.

## Further Components of the Define Application Description Window

In addition to the menu bar, the "Define Application Description" dialog box contains several boxes and command buttons.

### Boxes

The boxes contained in the "Define Application Description" dialog box are:

- The "Application Name" text box, which is used to specify the name of the application to be unloaded.
- The "Application Files Directory" text box, which is used to specify the directory into which the specified Natural application files are to be unloaded.
- The "Application Files" list box that lists all application files currently defined for unloading. If no application file has been defined so far, a default application file is provided.
- The "Non-Natural Objects" check box, which is used to specify that existing unload instructions for non-Natural objects are to be displayed in the "Unload Instructions" group frame (see below).
- The "Unload Instructions [*application-file-name*]" group frame that consists of a list box, which lists all unload instructions currently contained in the application file selected in the "Application Files" list box, and three command buttons.

If the "Non-Natural Objects" check box is selected (see above), the "Unload Instructions [Non-Natural Objects]" group frame is displayed instead, which consists of a list box that lists all unload instructions currently available for non-Natural objects.

## Buttons

In addition to the OK, "Cancel" and "Help" buttons, the "Define Application Description" dialog box contains the following command buttons:

- The "Add" button, which can be used as an alternative to the "Add" function of the Instructions menu to add a new line of instructions to the list box of the "Unload Instructions" group frame.
- The "Modify" button, which can be used as an alternative to the "Modify" function of the Instructions menu to modify the instruction line selected in the list box of the "Unload Instructions" group frame.
- The "Delete" button, which can be used as an alternative to the "Delete" function of the Instructions menu to delete the instruction line selected in the list box of the "Unload Instructions" group frame.

## Adding and Modifying Application Files

If you select an existing application file in the "Application Files" list box and choose the "Add" function of the "Application Files" menu, the "Add Application File" dialog box is displayed, in which you can specify a name for the application file to be added. Depending on the setting of the "Options" menu, the application file is then added either before or after the previously selected one.

If you select an existing application file in the "Application Files" list box and choose the "Modify" function of the "Application Files" menu, the "Modify Application File" dialog box is displayed, in which you can modify the application file name.

In both dialog boxes, you can also specify a user comment.

To modify the contents (that is, the unload instructions) of the application file selected in the "Application Files" list box, use the functions of the "Instructions" menu and/or the corresponding command buttons.

## Adding and Modifying Unload Instructions for Natural Objects

This section only applies if the "Non-Natural Objects" check box is not selected.

If you select an existing instruction line in the list box of the "Unload Instructions" group frame and then choose either the "Add" function of the "Instructions" menu or the "Add" button, the "Add Instructions" dialog box is displayed.

The "Add Instructions" dialog box provides a menu, from which you can choose the following Natural object types for unloading or deletion:

### Menu - Objects

- Unload Programming Objects
- Unload DDMs
- Unload Error Message Texts
- Delete Programming Objects
- Delete DDMs
- Delete Error Message Texts

With each object type, an appropriate "Add Instruction" dialog box appears when chosen. This dialog box is empty, so that you can specify the necessary items for adding a new instruction line. Depending on the setting of the "Options" menu, the instruction line is then added either after or before the previously selected one.

If you select an existing instruction line in the list box of the "Unload Instructions" group frame and then choose either the "Modify" function of the "Instructions" menu or the "Modify" button, the corresponding "Modify Instruction" dialog box is displayed. This dialog box displays the individual items of the selected instruction line for modification.

## Unload Programming Objects

If you choose "Unload Programming Objects" as the type of objects to be unloaded, a dialog box is displayed, in which you can specify the following items:

<b>Object Form</b>	Depending on your specification, you can unload the following programming objects:
	<b>Any</b> Any source-only, cataloged-only and stowed objects.
	<b>Cataloged</b> Any object which exists as a <i>cataloged</i> object.
	<b>Source</b> Any object which exists as a <i>source</i> object.
	<b>Both</b> Only those objects which exist in <i>both</i> source and cataloged form.
<b>From Library</b>	The name of the library from which the specified programming object(s) are to be unloaded. A list of existing libraries can be displayed for selection.
<b>Object Name</b>	The name of the object to be unloaded. The name can be either a specific name or a range. You can specify a range by using asterisk notation (*), wildcard notation (?), and/or the special characters "<" and ">". If you specify an asterisk only (default), all objects contained in the specified "From Library" are selected. Choose the "Select Objects" button to display the selected object range and select objects from that range.
<b>Object Type</b>	The type(s) of object(s) to be unloaded. All object types with the specified "Object Name" are selected by default (*). Choose the "Select Types" button if you want to select specific object types only.
<b>To Library</b>	The name of the library into which the unloaded objects are to be loaded. If no library is specified, the name of the library specified as "From Library" is used.
<b>Xref</b>	Depending on your specification, you can unload the Xref data or not:
	<b>Yes</b> If a cataloged object has cross-reference data, these are unloaded with the object.
	<b>No</b> The cross-reference data will not be unloaded.
<b>Comment</b>	An area, in which you can enter text, for example, a comment.

Natural programming objects are unloaded from the FNAT or FUSER system files. Objects in libraries whose names begin with "SYS" (except the library SYSTEM) are, by default, unloaded from the FNAT file; any other objects are unloaded from the FUSER file, unless a DBID and FNR is specified for a library in Natural Security.

## Unload DDMs

If you choose "Unload DDMs" as the type of objects to be unloaded, a dialog box is displayed, in which you can specify the following items:

<b>Object Form</b>	Depending on your specification, you can unload the following DDMs:
	<b>Any</b> Any source-only, cataloged-only and stowed DDM.
	<b>Cataloged</b> Any DDM which exists as a <i>cataloged</i> DDM.
	<b>Source</b> Any DDM which exists as a <i>source</i> DDM.
	<b>Both</b> Only those DDMs which exist in <i>both</i> source and cataloged form.
<b>From Library</b>	The name of the library from which the specified DDM(s) are to be unloaded. A list of existing libraries can be displayed for selection.
<b>DDM Name</b>	The name of the DDM to be unloaded. The name can be either a specific name or a range. You can specify a range by using asterisk notation (*), wildcard notation (?), and/or the special characters "<" and ">". If you specify an asterisk only (default), all DDMs contained in the specified "From Library" are selected. Choose the "Select Objects" button to display the selected DDM range and select DDMs from that range,.
<b>DDM DBID</b>	The database ID (DBID) of the DDM to be unloaded. If you specify a DBID, only DDMs with this DBID are unloaded. The DBID must be a number from 0 to 65535 (except 255).
<b>DDM FNR</b>	The file number (FNR) of the DDM to be unloaded. If you specify a FNR (file number), only DDMs with this FNR are unloaded. The FNR must be a number from 0 to 5000.
<b>To Library</b>	The name of the library into which the unloaded DDMs are to be loaded. If no library is specified, the name of the library specified as "From Library" is used.
<b>Comment</b>	An area in which you can enter text, for example, a comment.

## Unload Error Message Texts

If you choose "Unload Error Message Texts" as the type of objects to be unloaded, a dialog box is displayed, in which you can specify the following items:

<b>Message Type</b>	The type of error messages to be unloaded: <b>User</b> User-defined error messages <b>System</b> Natural error messages
<b>From Library</b>	The name of the library from which the specified messages are to be unloaded (only to be used with user messages). A list of existing libraries can be displayed for selection.
<b>Message Number</b>	The range of error message numbers to be unloaded.
<b>Language Code</b>	The language code of the error messages to be unloaded; for valid codes, see the description of the "*LANGUAGE" system variable in the Natural Reference documentation.
<b>To Library</b>	The name of the library into which the unloaded messages are to be loaded (only to be used with user messages). If no library is specified, the name of the library specified as "From Library" is used.
<b>Comment</b>	An area in which you can enter text, for example, a comment.

**Note:**

For Natural error messages, you need not specify a library, because they are always unloaded from either the FNAT system file or the error messages subdirectory.

### Delete Programming Objects

If you choose "Delete Programming Objects" as the type of object to be deleted, a dialog box is displayed, in which you can specify the following items:

<b>Object Form</b>	Depending on your specification, you can delete the following programming objects:
	<b>Any</b> - Any source-only, cataloged-only and stowed objects.
	<b>Cataloged</b> - Any object which exists as a <i>cataloged</i> object.
	<b>Source</b> - Any object which exists as a <i>source</i> object.
	<b>Both</b> - Only those objects which exist in <i>both</i> source and cataloged form.
<b>From Library</b>	The name of the library from which the specified programming object(s) are to be deleted. A list of existing libraries can be displayed for selection.
<b>Object Name</b>	The name of the object to be deleted. The name can be either a specific name or a range. You can specify a range by using asterisk notation (*), wildcard notation (?), and/or the special characters "<" and ">". If you specify an asterisk only (default), all objects contained in the specified "From Library" are selected. Choose the "Select Objects" button to display the selected object range and select objects from that range.
<b>Object Type</b>	The type(s) of object(s) to be deleted. All object types with the specified "Object Name" are selected by default (*). Choose the "Select Types" button if you want to select specific object types only.
<b>Comment</b>	An area in which you can enter text, for example, a comment.

Natural programming objects are deleted from the FNAT or FUSER system files. Objects in libraries whose names begin with "SYS" (except the library SYSTEM) are, by default, deleted from the FNAT file; any other objects are deleted from the FUSER file, unless a DBID and FNR is specified for a library in Natural Security.

## Delete DDMs

If you choose "Delete DDMs" as the type of object to be deleted, a dialog box is displayed, in which you can specify the following items:

<b>Object Form</b>	Depending on your specification, you can delete the following DDMs:
	<b>Any</b> - Any source-only, cataloged-only and stowed DDM.
	<b>Cataloged</b> - Any DDM which exists as a <i>cataloged</i> DDM.
	<b>Source</b> - Any DDM which exists as a <i>source</i> DDM.
	<b>Both</b> - Only those DDMs which exist in <i>both</i> source and cataloged form.
<b>From Library</b>	The name of the library from which the specified DDM(s) are to be deleted. A list of existing libraries can be displayed for selection.
<b>DDM Name</b>	The name of the DDM to be deleted. The name can be either a specific name or a range. You can specify a range by using asterisk notation (*), wildcard notation (?), and/or the special characters "<" and ">". If you specify an asterisk only (default), all DDMs contained in the specified "From Library" are selected. Choose the "Select Objects" button to display the selected DDM range and select DDMs from that range.
<b>Comment</b>	An area, in which you can enter text, for example, a comment.

## Delete Error Message Texts

If you choose "Delete Error Message Texts" as the type of object to be deleted, a dialog box is displayed, in which you can specify the following items:

<b>Message Type</b>	The type of error messages to be deleted: <b>User</b> - User-defined error messages <b>System</b> - Natural error messages
<b>From Library</b>	The name of the library from which the messages are to be deleted (only to be used with user messages). A list of existing libraries can be displayed for selection.
<b>Message Number</b>	The range of error message numbers to be deleted.
<b>Language Code</b>	The language code of the error messages to be deleted; for valid codes, see the description of the "*LANGUAGE" system variable in the Natural Reference documentation.
<b>Comment</b>	An area in which you can enter text.

### Note:

For Natural error messages, you need not specify a library, because they are always deleted from the FNAT system file or the error messages subdirectory.

### Selecting Objects

If you choose the "Select Objects" button when unloading or deleting Natural programming objects or DDMs, a corresponding "Select Objects:" dialog box is displayed, in which you can:

- Select objects from the displayed selection list by marking them and choosing the "Select" button.
- Remove objects from the list of objects selected from the displayed selection list either by marking them and choosing the "Remove" button, or by choosing the "Remove all" button.

When you choose OK, you return to the corresponding "Unload" or "Delete" dialog box, in which all other unload or delete items (except "To Library") are now disabled.

### Adding and Modifying Unload Instructions for Non-Natural Objects

This section only applies if you select the "Non-Natural Objects" check box; for "Natural Objects" see the corresponding section.

If you select an existing instruction line in the list box of the "Unload Instructions" group frame and then choose either the "Add" function of the "Instructions" menu or the "Add" button, the "Add Instruction: Unload Objects from Directory" dialog box is displayed.

If you select an existing instruction line in the list box of the "Unload Instructions" group frame and then choose either the "Modify" function of the "Instructions" menu or the "Modify" button, the corresponding "Modify Instruction: Unload objects from Directory" dialog box is displayed. This dialog box displays the individual items of the selected instruction line for modification.

### Unload Objects from Directory

You select "Unload Objects from Directory" if you want to unload non-Natural objects contained in an operating system directory; a dialog box is displayed, in which you can specify the following items:

<b>From Directory</b>	The name of the directory or Natural subdirectory from which the specified object is to be unloaded.
<b>Object Name</b>	The name of the object to be unloaded.
<b>To Directory</b>	The name of the Natural subdirectory in the target environment to which the specified objects are to be loaded. The path can contain environment variables. Example: %TEMP%/xy.
<b>Comment</b>	An area in which you can enter text, for example, a comment.

### Selecting Objects

If you choose the "Select Objects" button when unloading non-Natural objects, a corresponding "Select Object from Directory" dialog box is displayed, in which you can:

- Select objects from the displayed selection list by marking them and choosing the OK button.

When you choose OK, you return to the corresponding "Unload Objects" or "Delete Objects" dialog box.

## **Unloading Natural Applications**

If you want to generate and unload an application, choose the "Unload" function of the "Description" menu is displayed.

Since this function corresponds to the "Unload Application" function that can be chosen from within the "Providing Applications" dialog box, refer to the description of this function for further information.

## **Packaging Natural Applications**

If you want to package an application, choose the "Pack" function of the "Description" menu is displayed.

Since this function corresponds to the "Package Application" function that can be chosen from within the "Providing Applications" dialog box, refer to the description of this function for further information.

## Unload Application

When you choose the "Unload Application" function, the "Open Application Description" dialog box appears, in which you specify the name of the description for your application to be unloaded.

When you specify a name and choose OK, the "Unload Application [*description-name*]" dialog box is displayed.

**Note:**

The "Unload Application" dialog box also appears when you choose the "Unload" function from the "Description" menu of the "Define Application Description" dialog box.

### The Unload Application Dialog Box

Before you start to unload the application by choosing the "Unload" button, you can specify:

- several unload options in the General Unload Options group frame;
- whether and where an unload report is to be created in the Report group frame.

The application is unloaded to the directory specified in the "Application Files Directory" text box. It consists of all Natural application files (that is, unload instructions for Natural objects) and non-Natural objects as specified in the corresponding application description.

### General Unload Options

In the "Unload Application" dialog box you can specify the following general unload options:

Option	Specifies
<b>User Action</b>	Whether a user action to be performed if a report is generated. If also the option "Screen" is selected, the display of the report stops each time a user action is required; if "Screen" is not selected, no user response is required.
<b>Ignore Warning</b>	Whether processing is to be continued if an object contained in the application description could not be found.
<b>Restart</b>	Whether processing is to be resumed if it has been terminated abnormally.
<b>Work File Name (Restart)</b>	The name of the work file that is to contain the restart information required in the case of a restart.

## Report

In addition to the general unload options, you can specify the following "Report" options:

<b>Option</b>	<b>Specifies</b>
<b>Screen</b>	Whether the report is to be output on the screen. If also the option "User Action" is selected, the display of the report stops each time a user action is required; if "User Action" is not selected, the next status message is displayed without waiting for a user response.
<b>Work File</b>	Whether the report is to be written to a work file.
<b>Work File Name (Report)</b>	The name of the work file that is to contain the report if the "Work File" option has been selected.

## Package Application

When you choose the "Package Application" function, the "Open Application Description" dialog box appears, in which you specify the name of the description for your application to be packaged.

When you specify a name and choose OK, the "Package Application [*description-name*]" dialog box is displayed.

**Note:**

The "Package Application" dialog box also appears when you choose the "Pack" function from the "Description" menu of the "Define Application Description" dialog box.

## Package Application Dialog Box

Before you start to package the application by choosing the "Pack" button, you can specify:

- Several packaging options in the General Packaging Options group frame.
- Whether and where a packaging report is to be created in the Report group frame.

The application is packaged in the directory specified in the "Target Directory" text box. It consists of all unloaded Natural application files and non-Natural objects as specified in the corresponding application description.

In addition to the path name, the size of the target directory (for example, a diskette) can be specified in the "Directory Size" text box.

Choosing the "Proposal" button invokes the Packaging Proposal [*description-name*] dialog box.

## Packaging Proposal

The "Packaging Proposal [*description-name*]" dialog box provides you with a proposal of how to distribute the various application files to be packaged on, for example, several diskettes.

Only application files with message "File is ok" should be used for packaging. Multiple application files can be grouped.

Entries in the "Group" and "Disk" columns of the provided proposal can be modified, and modified proposals can be sorted and checked for feasibility. An automatic check is performed when you choose OK.

### Packaging of SETUP.EXE Files

SYSPAUL puts the following files on the first disk:

- The APPLINFO.TXT file, which is a control file for SETUP.EXE. This file contains all information necessary to load a Natural application with SETUP.EXE. This includes the appearance of the startup screen, options, and information about the files to be loaded, and custom information.
- The SETUP.EXE
- Online help for the SETUP.EXE
- The startup bitmap to be displayed in the background.
- The list of the Natural parameter modules which are to take on the same values as the settings in the customer environment. When SETUP.EXE is run, the system file settings of this parameter module are modified to the settings which exist in the user environment; other parameters are left unchanged.

### General Packaging Options

In the "Pack Application" dialog box you can specify the following general packaging options:

Option	Specifies
<b>User Action</b>	Whether a user action to be performed if a report is generated. If also the option "Screen" is selected, the display of the report stops each time a user action is required; if "Screen" is not selected, no user response is required.
<b>Ignore Warning</b>	Whether processing is to be continued if an object contained in the application description could not be found.
<b>Check Files</b>	Whether the files are to be checked beforehand.
<b>Keep Files</b>	Whether the files are to be kept in the application files directory after processing or if they are to be deleted.

### Report Creation

In addition to the general packaging options, you can specify the following "Report" options:

Option	Specifies
<b>Screen</b>	Whether the report is to be output on the screen. If also the option "User Action" is selected, the display of the report stops each time a user action is required; if "User Action" is not selected, the next status message is displayed without waiting for a user response.
<b>Work File</b>	Whether the report is to be written to a work file.
<b>Work File Name (Report)</b>	The name of the work file that is to contain the report if the "Work File" option has been selected.

## Load Application File

If you want to load an application file, choose the "Load Application File" function in the "Providing Applications" dialog box. The "Load Application File" dialog box is displayed.

The application file to be specified in the "Application File" text box can be selected in a further dialog box, which is invoked by choosing the "Select" button.

Before you start to load the specified application file by choosing the "Load" button, you can specify:

- Several load options in the General Load Options group frame.
- Whether and where a load report is to be created in the Report group frame.

### General Load Options

In the "Load Application File" dialog box you can specify the following general load options:

Option	Specifies
<b>User Action</b>	Whether a user action to be performed if a report is generated. If also the option "Screen" is selected, the display of the report stops each time a user action is required; if "Screen" is not selected, no user response is required.
<b>Ignore Warning</b>	Whether processing is to be continued if an object to be loaded could not be found.
<b>Replace</b>	Whether a source in the target environment with the same name as the one you are loading is to be replaced.
<b>Select Objects</b>	Whether you want to load selected parts of the application file only; if so, you have to specify a selection via the "Select Objects" button.
<b>Restart</b>	Whether processing is to be resumed if it has been terminated abnormally.
<b>Work File Name (Restart)</b>	The name of the work file that is to contain the restart information required in the case of a restart.

## Report

In addition to the general load options, you can specify the following report options:

Option	Specifies
<b>Screen</b>	Whether the report is to be output on the screen. If also the option "User Action" is selected (see General Load Options), the display of the report stops each time a user action is required; if "User Action" is not selected, the next status message is displayed without waiting for a user response.
<b>Work File</b>	Whether the report is to be written to a work file.
<b>Work File Name (Report)</b>	The name of the work file that is to contain the report if the "Work File" option has been selected.

## Specify a Selection

If you select the load option "Select Objects" and choose the "Select Objects" button, the "Specify Selection" dialog box is displayed.

The "Specify Selection" dialog box provides a menu, from which you can choose the following Natural object types for loading into or deletion from your target environment:

- Load Programming Objects
- Load DDMs
- Load Error Message Texts

With each object type, an appropriate "Load Selection" dialog box appears when chosen. This dialog box is empty, so that you can specify the various items necessary for your selection.

If you check the "Delete allowed" entry in the "Options" box, the delete instructions in the load file are performed. If you uncheck "Delete allowed", the delete instructions are ignored.

### Load Programming Objects

If you choose "Load Programming Objects" as the type of objects to be loaded, a dialog box is displayed, in which you can specify the following items:

<b>Object Form</b>	Depending on your specification, you can load the following programming objects:
	<b>Any</b> - Any source-only, cataloged-only and stowed objects.
	<b>Cataloged</b> - Any object which exists as a <i>cataloged</i> object.
	<b>Source</b> - Any object which exists as a <i>source</i> object.
<b>Library</b>	The name of the library to be loaded. A list of available libraries can be displayed for selection. You can only specify a library that has been specified as target library with the "Unload" function.
<b>Object Name</b>	The name of the object to be loaded. The name can be either a specific name or a range. You can specify a range by using asterisk notation (*). If you specify a value followed by an asterisk, all objects whose name begins with this value are loaded. If you specify an asterisk only (default), all objects contained in the specified library are loaded.
<b>Object Type</b>	The type(s) of object(s) to be loaded. All object types with the specified object name are selected by default (*). Choose the "Select Types" button if you want to select specific object types only.
<b>Xref</b>	Depending on your specification, the following XREF restriction applies:
	<b>No</b> - Only the cataloged objects will be loaded, not the cross-reference data.
	<b>Yes</b> - Cataloged objects are only loaded if cross-reference data for the object are present on the load file.
	<b>Force</b> - Cataloged objects are only loaded if cross-reference data for the object are present on the load file. Additionally, the object must be defined in PREDICT.
	<b>Any</b> - Cross-reference data for the object are loaded if they are present on the load file.

Natural programming objects are loaded into the FNAT or FUSER system files. Objects in libraries whose names begin with "SYS" (except the library SYSTEM) are, by default, loaded into the FNAT file; any other objects are loaded into the FUSER file, unless a DBID and FNR is specified for a library in Natural Security.

## Load DDMs

If you choose "Load DDMs" as the type of objects to be loaded, a dialog box is displayed, in which you can specify the following items:

<b>Object Form</b>	Depending on your specification, you can load the following DDMs:  <b>Any</b> - Any source-only, cataloged-only and stowed DDM.  <b>Cataloged</b> - Any DDM which exists as a <i>cataloged</i> DDM.  <b>Source</b> - Any DDM which exists as a <i>source</i> DDM.
<b>Library</b>	The name of the library to be loaded. A list of available libraries can be displayed for selection. You can only specify a library that has been specified as target library with the "Unload" function.
<b>DDM Name</b>	The name of the DDM to be loaded. You can specify either a specific name or a range. You can specify a range by using asterisk notation (*). If you specify a value followed by an asterisk, all DDMs whose name begins with this value are loaded. If you specify an asterisk only (default), all DDMs contained in the specified library are loaded.
<b>DDM DBID</b>	The database ID (DBID) of the DDM to be loaded. If you specify a DBID, only DDMs with this DBID are loaded. The DBID must be a number from 0 to 65535 (except 255).
<b>DDM FNR</b>	The file number (FNR) of the DDM to be loaded. If you specify a FNR (file number), only DDMs with this FNR are loaded. The FNR must be a number from 0 to 5000.

## Load Error Message Texts

If you choose "Load Error Message Texts" as the type of objects to be loaded, a dialog box is displayed, in which you can specify the following items:

<b>Message Type</b>	The type of error messages to be loaded: <b>User</b> - User-defined error messages <b>System</b> - Natural error messages
<b>Library</b>	The name of the library to be loaded (only to be used with user messages). A list of available libraries can be displayed for selection. You can only specify a library that has been specified as target library with the "Unload" function.
<b>Message Number</b>	The range of error message numbers to be loaded.
<b>Language Code</b>	The language code of the error messages to be loaded; for valid codes, see the description of the "*LANGUAGE" system variable in the Natural Reference documentation.

**Note:**

For Natural error messages, you need not specify a library, because they are always loaded into either the FNAT system file or the error messages subdirectory.

## Scan Application File

If you want to scan an application file without actually loading it, choose the "Scan Application File" function in the "Providing Applications" dialog box. The "Scan Application File" dialog box is displayed.

The application file to be specified in the "Application File" text box can be selected in a further dialog box, which is invoked by choosing the "Select" button.

Before you start to scan the specified application file by choosing the "Scan" button, you can specify:

- Several scan options in the General Scan Options group frame.
- Whether and where a scan report is to be created in the Report group frame.

### General Scan Options

In the "Scan Application File" dialog box you can specify the following general scan options:

Option	Explanation
<b>User Action</b>	Specifies whether a user action to be performed if a report is generated. If also the option "Screen" is selected (see General Load Options), the display of the report stops each time a user action is required; if "Screen" is not selected, no user response is required.
<b>Continue Processing</b>	Specifies whether processing is to be continued if an object to be scanned could not be found.
<b>Select Objects</b>	Specifies whether you want to scan selected parts of the application file only; if so, you have to specify a selection via the "Select Objects" button.
<b>Restart</b>	Specifies whether processing is to be resumed if it has been terminated abnormally.
<b>Work File Name (Restart)</b>	Specifies the name of the work file that is to contain the restart information required in the case of a restart.

### Report

In addition to the general scan options, you can specify the following report options:

Option	Explanation
<b>Screen</b>	Specifies whether the report is to be output on the screen. If also the option "User Action" is selected (see General Load Options), the display of the report stops each time a user action is required; if "User Action" is not selected, the next status message is displayed without waiting for a user response.
<b>Work File</b>	Specifies whether the report is to be written to a work file.
<b>Work File Name (Report)</b>	Specifies the name of the work file that is to contain the report if the "Work File" option has been selected.

## Specify a Selection

If you select the scan option "Select Objects" and choose the "Select Objects" button, the "Specify Selection" dialog box is displayed.

The "Specify Selection" dialog box provides a menu, from which you can choose the following Natural object types for scanning:

### Scan Selection - Objects

- Load Programming Objects
- Load DDMs
- Load Error Message Texts

With each object type, an appropriate "Scan Selection" dialog box appears when chosen. This dialog box is empty, so that you can specify the various items necessary for your selection.

If you uncheck "Delete allowed", the delete instructions are ignored.

### Load Programming Objects

If you choose "Load Programming Objects" as the type of objects to be scanned, a dialog box is displayed, in which you can specify the following items:

<b>Object Form</b>	Depending on your specification, you can scan the following programming objects:
	<b>Any</b> - Any source-only, cataloged-only and stowed objects.
	<b>Cataloged</b> - Any object which exists as a <i>cataloged</i> object.
	<b>Source</b> - Any object which exists as a <i>source</i> object.
<b>Library</b>	The name of the library to be scanned. A list of available libraries can be displayed for selection. You can only specify a library that has been specified as target library with the "Unload" function.
<b>Object Name</b>	The name of the object to be scanned. The name can be either a specific name or a range. You can specify a range by using asterisk notation (*). If you specify a value followed by an asterisk, all objects whose name begins with this value are scanned. If you specify an asterisk only (default), all objects contained in the specified library are scanned.
<b>Object Type</b>	The type(s) of object(s) to be scanned. All object types with the specified object name are selected by default (*). Choose the "Select Types" button if you want to select specific object types only.
<b>Xref</b>	Depending on your specification, the following XREF restriction applies:
	<b>No</b> - Only the cataloged objects will be scanned, not the cross-reference data.
	<b>Yes</b> - Cataloged objects are only scanned if cross-reference data for the object are present on the load file.
	<b>Force</b> - Cataloged objects are only scanned if cross-reference data for the object are present on the load file. Additionally, the object must be defined in PREDICT.
	<b>Any</b> - Cross-reference data for the object are scanned if they are present on the load file.

Natural programming objects are scanned on the FNAT or FUSER system files. Objects in libraries whose names begin with "SYS" (except the library SYSTEM) are, by default, scanned on the FNAT file; any other objects are scanned on the FUSER file, unless a DBID and FNR is specified for a library in Natural Security.

## Load DDMs

If you choose "Load DDMs" as the type of objects to be scanned, a dialog box is displayed, in which you can specify the following items:

<b>Object Form</b>	Depending on your specification, you can scan the following DDMs:  <b>Any</b> - Any source-only, cataloged-only and stowed DDM.  <b>Cataloged</b> - Any DDM which exists as a <i>cataloged</i> DDM.  <b>Source</b> - Any DDM which exists as a <i>source</i> DDM.
<b>Library</b>	The name of the library to be scanned. A list of available libraries can be displayed for selection. You can only specify a library that has been specified as target library with the "Unload" function.
<b>DDM Name</b>	The name of the DDM to be scanned. You can specify either a specific name or a range. You can specify a range by using asterisk notation (*). If you specify a value followed by an asterisk, all DDMs whose name begins with this value are scanned. If you specify an asterisk only (default), all DDMs contained in the specified library are scanned.
<b>DDM DBID</b>	The database ID (DBID) of the DDM to be scanned. If you specify a DBID, only DDMs with this DBID are scanned. The DBID must be a number from 0 to 65535 (except 255).
<b>DDM FNR</b>	The file number (FNR) of the DDM to be scanned. If you specify a FNR (file number), only DDMs with this FNR are scanned. The FNR must be a number from 0 to 5000.

## Load Error Message Texts

If you choose "Load Error Message Texts" as the type of objects to be scanned, a dialog box is displayed, in which you can specify the following items:

<b>Message Type</b>	The type of error messages to be scanned: <b>User</b> - User-defined error messages <b>System</b> - Natural error messages
<b>Library</b>	The name of the library to be scanned (only to be used with user messages). A list of available libraries can be displayed for selection. You can only specify a library that has been specified as target library with the "Unload" function.
<b>Message Number</b>	The range of error message numbers to be scanned.
<b>Language Code</b>	The language code of the error messages to be scanned; for valid codes, see the description of the "*LANGUAGE" system variable in the Natural Reference documentation.

### Note:

For Natural error messages, you need not specify a library, because they are always scanned on either the FNAT system file or the error messages subdirectory.

# Natural Runtime Version

- General Information on Natural Runtime
  - Runtime Version Differences
  - Porting Procedure Overview
  - Step 1: Packaging the Application
  - Step 2: Installing the Runtime Version
  - Step 3: Installing the Application to the Runtime Version
  - Step 4: Starting the Application
  - Runtime Startup Services
- 

## General Information on Natural Runtime

Natural Runtime is used to execute applications that have been written using the development version of Natural for Windows.

This section tells you how to port an application from a Natural Development version to a Natural Runtime version workstation.

This porting process can be used for a first installation, and for Runtime workstation updates.

Before porting an application to a Runtime workstation, ensure that all objects have been compiled using identical Natural Development and Runtime versions.

## Runtime Version Differences

### System Commands

The following System Commands are **not** supported by the Runtime version:

CATALL	EDIT	RENUMBER	SCRATCH
CATALOG	GLOBALS	RUN	STOW
CHECK	PURGE	SAVE	STRUCT
CLEAR	READ	SCAN	UNCAT

### Editors

Natural editors are **not** supported.

### Natural Utilities

Natural utilities providing developer functionality are **not** supported.

## Porting Procedure Overview

### ▶ To port an application from a Natural Development workstation to a Natural Runtime workstation

1. Package the application on the Natural Development workstation:
  - Create a collecting directory in your file system on PC;
  - Customize the configuration files for the application, and copy the customized configuration files to the collecting directory;
  - Unload or copy the Natural objects to the same directory;
  - Copy all files from the collecting directory to the transfer medium (e.g., diskette, tape, network) used for installation of your Natural application on the Natural Runtime workstation.
2. Install the Natural Runtime software on the workstation.
3. Install the application objects on the Natural Runtime workstation:  
Copy the Natural configuration files from the transfer medium to the Natural Runtime workstation;  
Load or copy the Natural objects from the transfer medium.
4. Start the Natural application.

These steps are described in detail below. See also Transfer Natural Generated Programs in the Natural Programming Guide for further information.

## Step 1: Packaging the Application

After customizing the configuration settings, create the application installation package on the Natural Development workstation.

### ▶ To package the Natural application on the Natural Development workstation

1. Create a collecting directory.
2. Customize the global configuration file, and copy the customized configuration file to the collecting directory;
3. Customize the Natural parameter file, and copy the customized Natural parameter file to the collecting directory;
4. Copy the Natural code to the collection directory.
5. Copy the contents of the collection directory to the transfer medium (e.g., diskette, tape, network).

## Creating a Collecting Directory

Create a new directory on the local file of the Natural Development workstation. Use this temporary directory to collect all files belong to the application.

## Customizing the Global Configuration File

### ▶ To customize the global configuration file

1. Backup the existing global configuration file.

**Note:**

You can use the Natural Configuration Utility to locate the global configuration file. Run the Natural Configuration Utility. Display the Installation Assignments of the Local Config File. The full path name of the global configuration file is listed as "Global Configuration File".

2. Run the Natural Configuration Utility.
3. Adjust the Global Config File settings as required for your application.
4. Save the Global Config File settings. Note that you cannot start the Natural Development environment until

the global configuration file is adapted to the Runtime environment.

5. Copy the customized global configuration file to the collecting directory.
6. Restore the backed up global configuration file.

**Notes:**

1. The default paths to the FNAT and to the FUSER system files differ in Natural Development version and Natural Runtime version.

To simplify the installation of the Natural application in the Runtime environment, adjust the paths just in this step.

Select Global Config File, System Files, System Files. Modify the FNAT and the FUSER paths to the settings of the Natural Runtime version.

**Example:**

For a default Natural Development version installation and Natural Runtime installation, change:

99	100	C:\Program Files\Software AG\Natural\Version\Fnat
99	101	C:\Program Files\Software AG\Natural\Natapps\Fuser

to the following paths:

99	100	C:\Program Files\Software AG\Natural Runtime\Version\Fnat
99	101	C:\Program Files\Software AG\Natural Runtime\Natapps\Fuser

*Version* is the corresponding Natural version.

2. Make sure that with every new application you are porting, the new settings are compatible with the old ones.

**Example:**

Your first application accesses an SQL database and the DBID entry applies to this SQL database. Your second application, which you are porting at a later date, accesses an Adabas C database. In this case, you must add a second DBID entry for Adabas C, because if you do not, the new global configuration file will overwrite the SQL database's DBID and your first application will no longer be able to access its database.

## Customizing the Natural Parameter File

 **To customize the Natural Parameter File**

1. Run the Natural Configuration Utility on the Natural Development version workstation.
2. Select the corresponding Natural parameter file.
3. Adjust the Natural parameters as required for your application to run in the Runtime environment.
4. Ensure that your Natural parameter file contains at least the settings listed in the table below.

AUTO	Set to Y.
INIT-LIB	Indicate the library into which the application will be moved.
STARTUP	Indicate the application program name that is started.
USER	Indicate the default User ID to be set when Natural is started.

5. Save the modified parameter file with a the name that you want to use in the Natural Runtime environment, e.g. RUNPARAM.
6. Locate the parameter file.

**Note:**

You can use the Natural Configuration Utility to locate the parameter file. Display the Installation Assignment of the Local Config File. The location of the Natural parameter files is listed as "Path to Profile Parameters".

7. Copy the customized parameter file, e.g. RUNPARAM.SAG, from the location of the parameter files to the collecting directory.

## Copying Natural Code to the Collecting Directory

To make compiled code available to the Natural Runtime version, cataloged objects from a Development version of Natural must be copied to the Runtime version. If the Natural application consist of complete Natural libraries, it is possible to copy the corresponding Natural libraries with Windows File Manager copy routines. Another way for porting Natural code is to unload the objects on the Natural Development workstation, and to load them on the Runtime environment.

### To copy Natural code

1. Create a new library in which the cataloged objects are to be contained.
2. Copy all cataloged objects, the resources, and the error messages from the development library to the new library (Runtime). Do not copy the sources.
3. Locate the directory containing this Runtime library.

**Note:**

You can use the Natural SYSPROF command to locate the directory containing this Runtime library. Start the Natural SYSPROF command. Display the current definitions of the Natural system files. Select FUSER to see the path of the FUSER system file.

The Runtime library is the subdirectory on the FUSER directory with the same name as the source Runtime library . This directory contains a file named FILEDIR.SAG, a subdirectory GP containing all cataloged objects, a subdirectory RES containing the resources, and a subdirectory ERR containing the error messages.

4. Copy the entire directory, including the GP, RES and ERR subdirectories, from the FUSER path to the collecting directory.
5. Rename the directory on the collecting directory if necessary. Enter the name of the library that is used in the Runtime environment.

**Notes:**

1. If the application consists of more than one library, treat all libraries the same way.
2. Another way to locate the FUSER directory is to use the Natural Configuration utility. Run the Natural Configuration Utility. Select the Natural parameter file copied in the step above, and used to start the application in the Runtime environment. Display the System Files of the Natural Execution Configuration. The path of the FUSER system file is listed as "User System File" of the System Files.

### To unload Natural code

1. Start the Natural Object Handler.
2. Start the Unload Wizard.
3. Unload the objects into a Natural work file.
4. In the options settings, use a Natural work file located on the collecting directory.
5. If the application uses other library names in the Runtime environment, than in the Development

environment, set the names with the parameter setting.

If the application uses the same library names in both environments, do not use parameters with the parameter setting.

6. Unload all catalogued objects, shared resources and error messages contained in the application. Do not unload the sources.
7. After the successful unload, you can check the work file created on the collecting directory. Use the Load Wizard to scan the work file for all objects.

## **Completing the Package**

When all files on the collecting directory are ready for porting, copy the contents of the collecting directory, including all subdirectories to the transfer medium.

## **Step 2: Installing the Runtime Version**

See the Natural Installation documentation for information on how to install Natural Runtime for Windows.

## Step 3: Installing the Application to the Runtime Version

### ▶ To install the application to the Runtime version

1. Install the configuration files.
2. Install the Natural code.

## Installing the Configuration Files

### ▶ To install the configuration files

1. Locate the Natural parameter file and the global configuration file in the Runtime version.

**Note:**

You can use the Natural Configuration Utility. Run the Natural Configuration Utility. Display the Installation Assignments of the Local Config File. The full path name of the global configuration file is listed as "Global Configuration File". The location of the Natural parameter file is listed as "Path to Profile Parameters".

2. Copy the global configuration file and the Natural parameter from the transfer medium to the corresponding paths.

## Installing the Natural code

Depending on the package process, complete libraries are transferred, or the Natural code is unloaded into a Natural work file, see Copying Natural Code.

If complete libraries are transferred, the transfer medium contains directories with Natural library names, and the Natural library structure (FILEDIR.SAG file, subdirectories named GP, RES, and ERR). Install the libraries with file copy.

If the Natural code is transferred via unload/load, the transfer medium contains a Natural work file. Install the Natural code with load.

### ▶ To install the Natural code with file copy

1. Locate the directory containing the Runtime libraries (FUSER).

**Note:**

You can use the Natural Configuration Utility to locate the FUSER path. Run the Natural Configuration Utility. Select the Natural parameter file copied in the step above. Display the System Files of the Natural Execution Configuration. The path of the FUSER system file is listed as "User System File" of the System Files.

2. Copy the libraries including their subdirectories to the FUSER path. Use the Windows File Manager.

### ▶ To install the Natural code with load

1. Compare the FUSER settings of the NATPARM parameter module, and the Natural parameter file copied in the step above.

**Note:**

You can use the Natural Configuration Utility. Run the Natural Configuration Utility. Select the corresponding parameter file. Display the System Files of the Natural Execution Configuration. The DBID and FNR of the FUSER are listed as "User System File" of the System Files.

2. Start the Natural Runtime Version using the standard NATPARM parameter module.
3. Execute the MENU program on the SYSOBJH library.
4. Start the Load Wizard.

5. Load the objects from Natural work file.
6. In the options settings, use the Natural work file located on the transfer medium as load file.
7. If the DBID and FNR of the new parameter module differs from the standard NATPARM setting, enter these values in the parameter settings. Choose "Use Global Parameters", and enter the corresponding values for "Load FUSER DBID" and "Load FUSER FNR".
8. Load all objects from the work file.
9. Exit the Object Handler, and exit Natural.

## Step 4: Starting the Application

### To start the application in the Runtime version

- Choose the "Natural Runtime" file in the Explorer.

If your customized Natural parameter file has a name other than 'NATPARAM', create a shortcut to the naturalr.exe in the binary path of the Natural Runtime environment, and change the Target in the Explorer using the following:

```
"..\Bin\naturalr.exe" PARM=filename
```

where *filename* is the name you have assigned to your customized Natural parameter file (without any file extension).

## Runtime Startup Services

With Natural (Runtime) for Windows, you can use a service to start Natural Runtime processes.

With the help of the startup service, you can also define "parameter templates" which are used to hold Natural command line parameters. The templates are named and thus allow you to start a Natural Runtime process with all the parameters contained within the template.

Enter the corresponding commands in the Command Prompt of your Windows system. Ensure, that the NATRTSVC is available.

### Note:

To ensure, that the NATRTSVC is available, change your current directory to the binary path of the Natural Runtime environment. In a default installation it is

```
C:\Program Files\Software AG\Natural Runtime\version\Bin,
```

where `version` is the current version of the Natural Runtime environment.

### Example:

#### To install the service with the automatic startup function

- Create a template and have the corresponding Natural Runtime process started up at boot time--you will see output from each invocation which is left out here:

```
NATRTSVC INSTALL automatic
NATRTSVC CREATE exa_temp
NATRTSVC SET exa_temp start=yes
NATRTSVC SET exa_temp parm=myparm
```

## Runtime Startup Service Commands

The following Runtime Startup Service commands are available in the Command Line Interface:

### Install Service

```
NATRTSVC
  INSTALL {manual | automatic}
```

Two parameters are available: manual (default) and automatic.

- **Manual:**  
This service is installed and must be started manually from the "Services" dialog in the "Control Panel".
- **Automatic:**  
This service is installed and started automatically when the PC is booted.

### Remove Service

```
NATRTSVC
  REMOVE
```

This command removes the service from the system.

**Start Service**

```
NATRVSVC
  START
```

This command starts the service if it had not been started yet. The service searches for previously created parameter templates where the start parameter is set to 'yes'. In addition, it starts a Natural Runtime process with the Natural parameters which are also stored in the template.

**Start Specified Parameter Template**

```
NATRVSVC
  START template-name
```

This command starts a Natural Runtime process with the Natural parameters stored in the specified template. If the service has not been started (automatically at boot time or manually by the user) an error message is displayed.

**Stop Service**

```
NATRVSVC
  STOP
```

This command stops the service and stops all Natural Runtime processes that have been started by the Natural Runtime service.

**Stop the Specified Template**

```
NATRVSVC
  STOP template-name
```

This command stops the Natural runtime processes that has been started by the startup service with the Natural parameters stored in the specified template.

**Create New Parameter Template to be Started by the Service**

```
NATRVSVC
  CREATE template-name
```

**Delete Specified Template from the Service**

```
NATRVSVC
  DELETE template-name
```

**Define Startup**

```
NATRVSVC
  SET template-name start=(yes | no)
```

If the value of

```
start
```

is set to 'yes', a Natural Runtime process with the Natural parameters also stored in the specified template will be started if the service is started. The default value of

```
start
```

is 'no'.

**Define Natural Parameters**

```
NATRISVC
  SET template-name Natural-parameter
```

This command stores the Natural parameter in the specified template. For valid Natural parameters, refer to the Natural Operations documentation, section Profile Parameters.

**Display all Templates**

```
NATRISVC
  SHOW
```

This command displays the startup setting and the stored Natural parameters for all Templates.

**Display specific Template**

```
NATRISVC
  SHOW template-name
```

This command displays the startup setting and the stored Natural parameters for the specified Template.

**Display Status of all Templates (active or inactive)**

```
NATRISVC
  STATUS
```

This command is used to determine if the Natural Runtime processes corresponding to all templates are active.

**Display Status of Specified Template**

```
NATRISVC
  STATUS template-name
```

This command is used to determine if the Natural Runtime process corresponding to the specified template is active.

# Support of Language-Specific Characters

This section describes how Natural supports language-specific characters and covers the following topics:

- Why is the Support of Language-Specific Characters Important?
  - The Configuration File NATCONV.INI
  - Sample NATCONV.INI File
- 

## Why is the Support of Language-Specific Characters Important?

The support of language-specific characters can help you when using:

- Upper-/Lower-Case Conversion
- Identifier Checking
- Character Classification
- Upper-/Lower-Case Translation of language-specific characters;
- language-specific characters in Natural identifiers, object names and library names;
- language-specific characters in an operand compared with a mask definition (see **MASK Option** in the Natural Reference documentation).

## Upper-/Lower-Case Conversion

Natural performs upper-/lower-case conversion if you specify one of the following items:

- the "%U" terminal command,
- the "AD=T" field attribute,
- the EXAMINE TRANSLATE statement.

By modifying the translation tables, you can support language-specific characters.

## Identifier Checking

Natural checks identifiers (that is, user-defined variables in source programs), names of Natural objects and names of Natural libraries by using check tables. By modifying these tables, you can, for example, allow language-specific characters. In addition, you can redefine the characters "+", "#" and "&", which have a special meaning when used as the first character in variable names.

## Character Classification

Natural performs an internal character classification when comparing an operand with a MASK option, and for the default delimiter used in the EXAMINE and SEPARATE statements. This classification is done via default tables. By modifying these tables, you can support language-specific characters.

## Configuration File NATCONV.INI

All check, translation and classification tables used by Natural to support language-specific characters reside in the configuration file NATCONV.INI.

You can modify NATCONV.INI to support local or application-specific characters.

In a standard application, NATCONV.INI need not and should not be modified, because this could lead to serious inconsistencies, in particular if Natural objects and database data are already present.

Any modifications of NATCONV.INI should be well considered and carefully performed, otherwise problems might occur that are difficult to locate.

NATCONV.INI is subdivided in sections and subsections. It contains four defined sections called:

- CHARACTERSET-DEFINITION
- CASE-TRANSLATION
- IDENTIFIER-VALIDATION
- CHARACTER-CLASSIFICATION

<b>CHARACTERSET-DEFINITION</b>	This section defines the name of the internal character set. The default is ISO8859_1. If you choose a different character set, subsections for this character set must be contained in the following sections.
<b>CASE-TRANSLATION</b>	This section contains the tables required for the conversion from upper to lower case. This conversion is done within the internal character set. If, for example, the internal character set is "ISO8859_5", the following two subsections must be contained in this section: - [ISO8859_5->UPPER] - [ISO8859_5->LOWER]
<b>IDENTIFIER-VALIDATION</b>	This section contains the tables required for the validation of identifiers, object names and library names. It contains a subsection for each defined internal character set. The special characters "#" (for non-database variables), "+" (for application-independent variables), "@" (for SQL and Adabas null or length indicators) and "&" (for dynamic source generation) can be redefined in this section. In addition, the set of valid first and subsequent characters for identifiers, object names and library names can be modified.
<b>CHARACTER-CLASSIFICATION</b>	This section contains the tables required for the classification of characters, which, for example, are used when evaluating the MASK option. It contains a subsection for each defined internal character set.

The section CHARACTERSET-DEFINITION as well as each subsection contain lines which describe how characters are to be converted and which characters are related with which attributes.

These lines are represented as follows:

```

line ::= key
= value key ::= name_key | range_key name_key
::= keyword{ CHARS } keyword ::= INTERNAL-CHARACTERSET | NON-DB-VARI
| DYNAMIC-SOURCE | GLOBAL-VARI | FIRST-CHAR | SUBSEQUENT-CHAR | LIB-FIRST-CHAR
| LIB-SUBSEQUENT-CHAR | ALTERNATE-CARET ISASCII | ISALPHA | ISALNUM | ISDIGIT
| ISXDIGIT | ISLOWER | ISUPPER | ISCTRL | ISPRINT | ISPUNCT | ISGRAPH | ISSPACE
range_key ::= hexnum | hexnum-hexnum value
::= val {, val } val ::= hexnum | hexnum-hexnum
hexnum ::= xhexdighexdigit | Xhexdighexdigit
    
```

**Notes:**

If "range\_key" variable is specified on the left-hand side, the number of values specified on the right-hand side must correspond to the number of values specified in the key range, except if only one value is specified on the right-hand side, which is then assigned to each element of the key range.

When the "name\_key" variable is specified on the left-hand side and the corresponding list of character codes does not fit in one line, it can be continued on the next line by specifying "name\_key =" again.

**Examples for Valid Lines:**

x00-x1f = x00	All characters between "x00" and "x1f" are converted to "x00".
x00-x7f = x00-x7f	All characters between "x00" and "x7f" are not converted.
x00-x08 = x00,x01-x07,x00	The characters "x00" and "x08" are converted to "x00" and characters between "x01" and "x07" are not converted.
ISALPHA = x41-x5a,x61-x7a,xc0-xd6,xd8 ISALPHA = xd9-xf6,xf8-xff	The attribute ISALPHA is assigned to all characters specified in these two lines.

**Examples for Invalid Lines:**

x41 = 'A'	All characters must be specified in hexadecimal format.
0x00-0x1f = 0x00	Hexadecimal values have to be specified in either of the following ways: <i>xdigitdigit</i> <i>Xdigitdigit</i>
x00-x0f = x00,x01	The number of specified values does not correspond to the number of elements in the key range.

## Sample NATCONV.INI File

```

#*****
# Natural character set configuration file #*****
# # NATCONV.INI consists of five sections: # # CHARACTERSET-DEFINITION - Defines
the name of the character set that is to # be used by Natural. This name has to
appear in # the subsection names of all following sections. # IBMCP_850 (PC) or
ISO8859_1 (UNIX) are defined # as default character sets. # # CHARACTERSET-TRANSLATION
- In addition to the character set used by Natural # (the so-called internal character
set) there may # exist several external character sets used for # different terminals
connected to the same # Natural. If you want to use a terminal with a # character
set differing from the internal # character set, you have to define a TCS entry
for # the terminal in SAGtermcap (e.g. ":TCS=USASCII:") # and corresponding
subsections (e.g. # "[ISO8859_1->USASCII]" and # "[USASCII->ISO8859_1]")
in this section. # The conversion between external and internal # character set
is performed during terminal I/O. # Note that this conversion is only done if
you are # running Natural for UNIX. # Beside conversion tables for the terminal
# character set there exist additional tables # for special purposes (e.g. tables
for conversion # between ASCII and EBCDIC code). # # CASE-TRANSLATION - This section
defines the translation tables # for lower- and upper-case translation. The #
tables are valid for the English language. If you # want to support language-specific
characters, see # explanation below. #

# IDENTIFIER-VALIDATION
- This section defines tables for the validation of # Natural identifiers, object
names and library # names. The tables allow identifiers, object # and library
names as described in the Natural # manuals. If you want to support language-specific
# characters, see explanation below. # # CHARACTER-CLASSIFICATION - This section
defines tables for the # classification of characters according to the # corresponding
C library functions (e.g. # isalpha). The tables are valid for the English # language.
If you want to support # language-specific characters, see explanation # below.
# # # How to support language-specific characters with NATCONV.INI: # # As mentioned
above NATCONV.INI supports the English language per # default. If your language
uses additional characters you have to make # some modifications. As an example
the required modifications for the # German language are given as comments in
the corresponding sections. # Note that those modifications might affect the portability
of Natural # source files and/or Natural applications. # # Required modifications:
# # Section CASE-TRANSLATION # Modify the tables for lower- and upper-case translation
so that all your # alphabetic characters are translated correctly. These translations
must # be consistent with the ISLOWER and ISUPPER tables in the section # CHARACTER-CLASSIFICATION.
    
```

```

# # Section IDENTIFIER-VALIDATION # If you want to allow/forbid special characters
in Natural identifiers, # object or library names, you have to modify the corresponding
tables # (see section IDENTIFIER-VALIDATION). Note that it is not possible to
# define characters as valid identifier characters which have a special # meaning
in the Natural language; e.g. "(" #

# Section
CHARACTER-CLASSIFICATION # Add your alphabetic characters to the tables ISALPHA,
ISALNUM, ISLOWER # (if lower-case), ISUPPER (if upper-case) and ISPRINT. # Beware
that ISLOWER and ISUPPER must be consistent with the translation # tables in the
section CASE-TRANSLATION. # Make sure that no alphabetic character is included
in the tables # ISCNTRL and ISPUNCT. # *****

[CHARACTERSET-DEFINITION]

# defining
the internal character set for Natural

INTERNAL-CHARACTERSET
= ISO8859_1

[CHARACTERSET-DEFINITION-END] *****

*****
[CHARACTERSET-TRANSLATION]

# translation tables between
internal and external character set

#-----
[ISO8859_1->USASCII] #translate ISO8859-1 to USASCII code

x00-x7F = x00-x7F x80-xBF = x3F xC0-xCF = x41,x41,x41,x41,x41,x41,x41,x41,x43,x45,x45,x45,x45,x49,x49,x49,x49
xD0-xDF = x44,x4E,x4F,x4F,x4F,x4F,x4F,x3F,x4F,x55,x55,x55,x55,x59,x50,x73 xE0-xEF
= x61,x61,x61,x61,x61,x61,x61,x63,x65,x65,x65,x65,x69,x69,x69,x69 xF0-xFF = x64,x6E,x6F,x6F,x6F,x6F,x6F,x6F,x3F,x6F,x75,x75,x75,x75,x79,x70,x59

[ISO8859_1->USASCII-END] #-----

#-----
[USASCII->ISO8859_1] #translate USASCII to ISO8859-1 code

x00-xFF = x00-xFF

[USASCII->ISO8859_1-END] #-----

#-----
[ISO8859_1->ASCII_GERMAN] #translate ISO8859-1 to ASCII_GERMAN code

x00-x7F = x00-x7F x80-xA6 = x3F xA7 = x40 xA8-xBF = x3F xC0-xCF = x41,x41,x41,x41,x5B,x41,x41,x43,x45,x45,x45,x45,x49,x49,x49,x49
xD0-xDF = x44,x4E,x4F,x4F,x4F,x4F,x5C,x3F,x4F,x55,x55,x55,x5D,x59,x50,x7E xE0-xEF
= x61,x61,x61,x61,x7B,x61,x61,x63,x65,x65,x65,x65,x69,x69,x69,x69 xF0-xFF = x64,x6E,x6F,x6F,x6F,x6F,x7C,x3F,x6F,x75,x75,x75,x7D,x79,x70,x59

[ISO8859_1->ASCII_GERMAN-END] #-----

#-----
[ASCII_GERMAN->ISO8859_1] #translate ASCII_GERMAN to ISO8859-1 code

x00-xFF = x00-xFF x40 = xA7 x5B-x5D = xC4,xD6,xDC x7B-x7E = xE4,xF6,xFC,xDF

[ASCII_GERMAN->ISO8859_1-END] #-----

# translation tables between internal character set and EBCDIC
code

```

```

#-----
[ISO8859_1->EBCDIC] #translate ISO8859-1 to EBCDIC code #characters which cannot
be translated properly are replaced by a '?'

x00-x0F
= x00-x03,x37,x2D,x2E,x2F,x16,x05,x25,x0B-x0F x10-x1F = x10-x13,x3C,x3D,x32,x26,x18,x19,x3F,x27,x1C-x1F
x20-x2F = x40,x5A,x7F,x7B,x5B,x6C,x50,x7D,x4D,x5D,x5C,x4E,x6B,x60,x4B,x61 x30-x3F
= xF0-xF9,x7A,x5E,x4C,x7E,x6E,x6F x40-x4F = x7C,xC1-xC9,xD1-xD6 x50-x5F = xD7-xD9,xE2-xE9,xAD,xE0,xBD,x6A,x6D
x60-x6F = x79,x81-x89,x91-x96 x70-x7F = x97-x99,xA2-xA9,x8B,x4F,x9B,xA1,x07 x80-xFF
= x6F

[ISO8859_1->EBCDIC-END] #-----

#-----
[EBCDIC->ISO8859_1] #translate EBCDIC to ISO8859-1 code #characters which cannot
be translated properly are replaced by a '?'

x00-x0F
= x00-x03,x3F,x09,x3F,x7F,x3F,x3F,x3F,x0B-x0F x10-x1F = x10-x13,x3F,x3F,x08,x3F,x18,x19,x1A,x3F,x1C-x1F
x20-x2F = x3F,x3F,x1C,x3F,x3F,x0A,x17,x1B,x3F,x3F,x3F,x05,x06,x07,x3F x30-x3F
= x3F,x3F,x3F,x3F,x3F,x3F,x04,x3F,x3F,x3F,x14,x15,x3F,x3F x40-x4F = x20,x3F,x3F,x3F,x3F,x3F,x3F,x3F,x2E,x3C,x28,x2B,x7C
x50-x5F = x26,x3F,x3F,x3F,x3F,x3F,x3F,x3F,x3F,x21,x24,x2A,x29,x3B,x5E x60-x6F
= x2D,x2F,x3F,x3F,x3F,x3F,x3F,x3F,x2C,x25,x5F,x3E,x3F x70-x7F = x3F,x3F,x3F,x3F,x3F,x3F,x3F,x3A,x23,x40,x27,x3D,x22
x80-x8F = x3F,x61-x69,x3F,x7B,x3F,x3F,x3F,x2B x90-x9F = x3F,x6A-x72,x3F,x7D,x3F,x3F,x3F
xA0-xAF = x3F,x3F,x73-x7A,x3F,x3F,x3F,x3F,x3F,x3F xB0-xBF = x3F xC0-xCF = x3F,x41-x49,x3F,x3F,x3F,x3F,x3F
xD0-xDF = x3F,x4A-x52,x3F,x3F,x3F,x3F,x3F,x3F xE0-xEF = x3F,x3F,x53-x5A,x3F,x3F,x3F,x3F,x3F,x3F
xF0-xFF = x30-x39,x3F,x3F,x3F,x3F,x3F,x3F

[EBCDIC->ISO8859_1-END]
#-----

#-----
[IBMCP_850->EBCDIC] #translate IBMCP_850 to EBCDIC code #characters which cannot
be translated properly are replaced by a '?'

x00-x0F
= x00-x03,x37,x2D,x2E,x2F,x16,x05,x25,x0B-x0F x10-x1F = x10-x13,x3C,x3D,x32,x26,x18,x19,x3F,x27,x1C-x1F
x20-x2F = x40,x5A,x7F,x7B,x5B,x6C,x50,x7D,x4D,x5D,x5C,x4E,x6B,x60,x4B,x61 x30-x3F
= xF0-xF9,x7A,x5E,x4C,x7E,x6E,x6F x40-x4F = x7C,xC1-xC9,xD1-xD6 x50-x5F = xD7-xD9,xE2-xE9,xAD,xE0,xBD,x6A,x6D
x60-x6F = x79,x81-x89,x91-x96 x70-x7F = x97-x99,xA2-xA9,x8B,x4F,x9B,xA1,x07 x80-xFF
= x6F

[IBMCP_850->EBCDIC-END] #-----

#-----
[EBCDIC->IBMCP_850] #translate EBCDIC to IBMCP_850 code #characters which cannot
be translated properly are replaced by a '?'

x00-x0F
= x00-x03,x3F,x09,x3F,x7F,x3F,x3F,x3F,x0B-x0F x10-x1F = x10-x13,x3F,x3F,x08,x3F,x18,x19,x1A,x3F,x1C-x1F
x20-x2F = x3F,x3F,x1C,x3F,x3F,x0A,x17,x1B,x3F,x3F,x3F,x05,x06,x07,x3F x30-x3F
= x3F,x3F,x3F,x3F,x3F,x3F,x04,x3F,x3F,x3F,x14,x15,x3F,x3F x40-x4F = x20,x3F,x3F,x3F,x3F,x3F,x3F,x3F,x2E,x3C,x28,x2B,x7C
x50-x5F = x26,x3F,x3F,x3F,x3F,x3F,x3F,x3F,x3F,x21,x24,x2A,x29,x3B,x5E x60-x6F
= x2D,x2F,x3F,x3F,x3F,x3F,x3F,x3F,x2C,x25,x5F,x3E,x3F x70-x7F = x3F,x3F,x3F,x3F,x3F,x3F,x3F,x3F,x3A,x23,x40,x27,x3D,x22
x80-x8F = x3F,x61-x69,x3F,x7B,x3F,x3F,x3F,x2B x90-x9F = x3F,x6A-x72,x3F,x7D,x3F,x3F,x3F
xA0-xAF = x3F,x3F,x73-x7A,x3F,x3F,x3F,x3F,x3F,x3F xB0-xBF = x3F xC0-xCF = x3F,x41-x49,x3F,x3F,x3F,x3F,x3F,x3F
xD0-xDF = x3F,x4A-x52,x3F,x3F,x3F,x3F,x3F,x3F xE0-xEF = x3F,x3F,x53-x5A,x3F,x3F,x3F,x3F,x3F,x3F
xF0-xFF = x30-x39,x3F,x3F,x3F,x3F,x3F,x3F

[EBCDIC->IBMCP_850-END]
#-----

[CHARACTERSET-TRANSLATION-END] #*****

#*****
[CASE-TRANSLATION]

# translation tables for lower/uppercase
conversion of the internal character set # The translation tables are valid for
the English language. # If you want to handle language specific characters you
have to modify the # following tables accordingly. # The required modifications
for the German language are given as comment.

```

```

#-----
[ISO8859_1->LOWER] #translate ISO8859-1 to lowercase

#---- english version x00-x3F = x00-x3F x40-x4F = x40,x61,x62,x63,x64,x65,x66,x67,x68,x69,x6A,x6B,x6C,x6D,x6E,x6F
x50-x5F = x70,x71,x72,x73,x74,x75,x76,x77,x78,x79,x7A,x5B,x5C,x5D,x5E,x5F x60-x7F
= x60-x7F x80-xFF = x80-xFF

#---- german version #x00-x3F
= x00-x3F #x40-x4F = x40,x61,x62,x63,x64,x65,x66,x67,x68,x69,x6A,x6B,x6C,x6D,x6E,x6F
#x50-x5F = x70,x71,x72,x73,x74,x75,x76,x77,x78,x79,x7A,x5B,x5C,x5D,x5E,x5F #x60-xBF
= x60-xBF #xC0-xCF = xC0,xC1,xC2,xC3,xE4,xC5,xC6,xC7,xC8,xC9,xCA,xCB,xCC,xCD,xCE,xCF
#xD0-xDF = xD0,xD1,xD2,xD3,xD4,xD5,xF6,xD7,xD8,xD9,xDA,xDB,xFC,xDD,xDE,xDF #xE0-xFF
= xE0-xFF

[ISO8859_1->LOWER-END] #-----

#-----
[ISO8859_1->UPPER] #translate ISO8859-1 to uppercase

#---- english version x00-x5F = x00-x5F x60-x6F = x60,x41,x42,x43,x44,x45,x46,x47,x48,x49,x4A,x4B,x4C,x4D,x4E,x4F
x70-x7F = x50,x51,x52,x53,x54,x55,x56,x57,x58,x59,x5A,x7B,x7C,x7D,x7E,x7F x80-xFF
= x80-xFF

#---- german version #x00-x5F = x00-x5F #x60-x6F
= x60,x41,x42,x43,x44,x45,x46,x47,x48,x49,x4A,x4B,x4C,x4D,x4E,x4F #x70-x7F = x50,x51,x52,x53,x54,x55,x56,x57,x58,x59,x5A,x7B,x7C,x7D,x7E,x7F
#x80-xDF = x80-xDF #xE0-xEF = xE0,xE1,xE2,xE3,xC4,xE5,xE6,xE7,xE8,xE9,xEA,xEB,xEC,xED,xEE,xEF
#xF0-xFF = xF0,xF1,xF2,xF3,xF4,xF5,xD6,xF7,xF8,xF9,xFA,xFB,xDC,xFD,xFE,xFF

[ISO8859_1->UPPER-END] #-----

#-----
[IBMCP_850->LOWER] #translate IBMCP_850 to lowercase

#---- english version x00-x3F = x00-x3F x40-x4F = x40,x61,x62,x63,x64,x65,x66,x67,x68,x69,x6A,x6B,x6C,x6D,x6E,x6F
x50-x5F = x70,x71,x72,x73,x74,x75,x76,x77,x78,x79,x7A,x5B,x5C,x5D,x5E,x5F x60-x7F
= x60-x7F x80-xFF = x80-xFF

#---- german version #x00-x3F
= x00-x3F #x40-x4F = x40,x61,x62,x63,x64,x65,x66,x67,x68,x69,x6A,x6B,x6C,x6D,x6E,x6F
#x50-x5F = x70,x71,x72,x73,x74,x75,x76,x77,x78,x79,x7A,x5B,x5C,x5D,x5E,x5F #x60-x7F
= x60-x7F #x80-x8F = x80,x81,x82,x83,x84,x85,x86,x87,x88,x89,x8A,x8B,x8C,x8D,x84,x8F
#x90-x9F = x90,x91,x92,x93,x94,x95,x96,x97,x98,x94,x81,x9B,x9C,x9D,x9E,x9F #xA0-xFF
= xA0-xFF

[IBMCP_850->LOWER-END] #-----

#-----
[IBMCP_850->UPPER] #translate IBMCP_850 to uppercase

#---- english version x00-x5F = x00-x5F x60-x6F = x60,x41,x42,x43,x44,x45,x46,x47,x48,x49,x4A,x4B,x4C,x4D,x4E,x4F
x70-x7F = x50,x51,x52,x53,x54,x55,x56,x57,x58,x59,x5A,x7B,x7C,x7D,x7E,x7F x80-xFF
= x80-xFF

#---- german version #x00-x5F = x00-x5F #x60-x6F
= x60,x41,x42,x43,x44,x45,x46,x47,x48,x49,x4A,x4B,x4C,x4D,x4E,x4F #x70-x7F = x50,x51,x52,x53,x54,x55,x56,x57,x58,x59,x5A,x7B,x7C,x7D,x7E,x7F
#x80-x8F = x80,x9A,x82,x83,x8E,x85,x86,x87,x88,x89,x8A,x8B,x8C,x8D,x8E,x8F #x90-x9F
= x90,x91,x92,x93,x99,x95,x96,x97,x98,x99,x9A,x9B,x9C,x9D,x9E,x9F #xA0-xFF = xA0-xFF

[IBMCP_850->UPPER-END] #-----

#-----
[USASCII->LOWER] #translate USASCII to lowercase

x00-x3F = x00-x3F x40-x4F = x40,x61,x62,x63,x64,x65,x66,x67,x68,x69,x6A,x6B,x6C,x6D,x6E,x6F
x50-x5F = x70,x71,x72,x73,x74,x75,x76,x77,x78,x79,x7A,x5B,x5C,x5D,x5E,x5F x60-xFF
= x60-xFF

```

```

[USASCII->LOWER-END] #-----
#-----
[USASCII->UPPER] #translate USASCII to uppercase

x00-x5F = x00-x5F x60-x6F = x60,x41,x42,x43,x44,x45,x46,x47,x48,x49,x4A,x4B,x4C,x4D,x4E,x4F
x70-x7F = x50,x51,x52,x53,x54,x55,x56,x57,x58,x59,x5A,x7B,x7C,x7D,x7E,x7F x80-xFF
= x80-xFF

[USASCII->UPPER-END] #-----
[CASE-TRANSLATION-END] #*****
#*****
[IDENTIFIER-VALIDATION]

# tables for validation of
identifiers, object names and library names. # # type of name first character
subsequent character # checked against checked against # -----
# identifier FIRST-CHAR * SUBSEQUENT-CHAR # *) '&' only allowed for dynamic
source generation # object name FIRST-CHAR * SUBSEQUENT-CHAR # *) '&' not
allowed # library name LIB-FIRST-CHAR LIB-SUBSEQUENT-CHAR # # The tables listed
below define valid identifiers, object and library names # as described in the
Natural manuals. # If you want to handle additional/other characters e.g. language
specific # characters you have to modify the tables accordingly.

#----- [ISO8859_1]

# special characters # # non DB variable # '#' NON-DB-VARI
= x23

# SQL separator character for null/length indicator
# '@' SQL-SEP-CHAR = x40 # dynamic source generation # '&' DYNAMIC-SOURCE
= x26 # global variable # '+' GLOBAL-VARI = x2B

# valid
first characters # '#' '&' '+' 'A'-'Z' 'a'-'z' FIRST-CHAR = x23,x26,x2B,x41-x5A,x61-x7A

# valid subsequent characters # '#' '$' '&' '-'
 '/' '0'-'9' SUBSEQUENT-CHAR = x23,x24,x26,x2D,x2F,x30-x39

# '@' 'A'-'Z' '_' 'a'-'z' SUBSEQUENT-CHAR-1 = x40,x41-x5A,x5F,x61-x7A

# valid first characters for library names # 'A'-'Z' 'a'-'z' LIB-FIRST-CHAR =
x41-x5A,x61-x7A

# valid subsequent characters for library
names # '-' '0'-'9' 'A'-'Z' '_' 'a'-'z' LIB-SUBSEQUENT-CHAR = x2D,x30-x39,x41-x5A,x5F,x61-x7A

# alternate symbol for '^' # input blank for IC, LC,
TC and edit masks, # not symbol for ciompare operators like ^= , ^< and ^>
# '^' ALTERNATE-CARET = xAA

[ISO8859_1-END] #-----
#-----
[IBMCP_850]

# special characters # # non DB variable
# '#' NON-DB-VARI = x23

```

```

# SQL separator character for
null/length indicator # '@' SQL-SEP-CHAR = x40 # dynamic source generation # '&'
DYNAMIC-SOURCE = x26 # global variable # '+' GLOBAL-VARI = x2B

# valid first characters # '#' '&' '+' 'A'-'Z' 'a'-'z' FIRST-CHAR = x23,x26,x2B,x41-x5A,x61-x7A

# valid subsequent characters # '#' '$' '&' '-'
 '/' '0'-'9' SUBSEQUENT-CHAR = x23,x24,x26,x2D,x2F,x30-x39

# '@' 'A'-'Z' '_' 'a'-'z' SUBSEQUENT-CHAR-1 = x40,x41-x5A,x5F,x61-x7A

# valid first characters for library names # 'A'-'Z' 'a'-'z' LIB-FIRST-CHAR =
x41-x5A,x61-x7A

# valid subsequent characters for library
names # '-' '0'-'9' 'A'-'Z' '_' 'a'-'z' LIB-SUBSEQUENT-CHAR = x2D,x30-x39,x41-x5A,x5F,x61-x7A

# alternate symbol for '^' # input blank for IC, LC,
TC and edit masks, # not symbol for ciompare operators like ^= , ^< and ^>
# '^' ALTERNATE-CARET = xAA

[IBMCP_850-END] #-----
#-----
[USASCII]

# special characters # # non DB variable
# '#' NON-DB-VARI = x23

# SQL separator character for
null/length indicator # '@' SQL-SEP-CHAR = x40 # dynamic source generation # '&'
DYNAMIC-SOURCE = x26 # global variable # '+' GLOBAL-VARI = x2B

# valid first characters # '#' '&' '+' 'A'-'Z' 'a'-'z' FIRST-CHAR = x23,x26,x2B,x41-x5A,x61-x7A

# valid subsequent characters # '#' '$' '&' '-'
 '/' '0'-'9' SUBSEQUENT-CHAR = x23,x24,x26,x2D,x2F,x30-x39

# '@' 'A'-'Z' '_' 'a'-'z' SUBSEQUENT-CHAR-1 = x40,x41-x5A,x5F,x61-x7A

# valid first characters for library names # 'A'-'Z' 'a'-'z' LIB-FIRST-CHAR =
x41-x5A,x61-x7A

# valid subsequent characters for library
names # '-' '0'-'9' 'A'-'Z' '_' 'a'-'z' LIB-SUBSEQUENT-CHAR = x2D,x30-x39,x41-x5A,x5F,x61-x7A

# alternate symbol for '^' # input blank for IC, LC,
TC and edit masks, # not symbol for ciompare operators like ^= , ^< and ^>
# '^' ALTERNATE-CARET = x5E

[USASCII-END] #-----

[IDENTIFIER-VALIDATION-END] #*****
#*****
[CHARACTER-CLASSIFICATION]

```

```

# classification of characters
according to the C library functions # # isascii # isalpha # isalnum # isdigit
# isxdigit # islower # isupper # iscntrl # isprint # ispunct # isgraph # isspace
# # The classification tables are valid for the English language. # If you want
to handle language specific characters you have to modify the # following tables
accordingly. # The required modifications for the German language are given as
comment.

#-----
[ISO8859_1] # valid for ISO8859_1 as internal character set # ASCII characters
ISASCII = x00-xFF

# alphabetical characters # 'A'-'Z'
'a'-'z' # #---- english version ISALPHA = x41-x5A,x61-x7A #---- german version
AE OE UE ss ae oe ue #ISALPHA = x41-x5A,x61-x7A,xC4,xD6,xDC,xDF,xE4,xF6,xFC

# alphanumeric characters # '0'-'9' 'A'-'Z' 'a'-'z' # #----
english version ISALNUM = x30-x39,x41-x5A,x61-x7A #---- german version AE OE UE
ss ae oe ue #ISALNUM = x30-x39,x41-x5A,x61-x7A,xC4,xD6,xDC,xDF,xE4,xF6,xFC

# digit characters # '0'-'9' ISDIGIT = x30-x39

# hexadecimal digit characters # '0'-'9' 'A'-'F' 'a'-'f' ISXDIGIT = x30-x39,x41-x46,x61-x66

# lowercase characters # 'a'-'z' # #---- english version
ISLOWER = x61-x7A #---- german version ss ae oe ue #ISLOWER = x61-x7A,xDF,xE4,xF6,xFC

# uppercase characters # 'A'-'Z' # #---- english version
ISUPPER = x41-x5A #---- german version AE OE UE #ISUPPER = x41-x5A,xC4,xD6,xDC

# control characters # ISCNTRL = x00-x1F,x7F-x9F

# printable characters # ISPRINT = x20-x7E,xA0-xFF

# special characters # ISPUNCT = x21-x2F,x3A-x40,x5B-x60,x7B-x7E,xA1-xBF,xD7,xF7

# graphical characters # ISGRAPH = x21-x7E,xA1-xFF

# spacing characters # ISSPACE = x09-x0D,x20

[ISO8859_1-END] #-----

#-----
[IBMCP_850] # valid for IBMCP_850 as internal character set # ASCII characters
ISASCII = x00-xFF

# alphabetical characters # 'A'-'Z'
'a'-'z' # #---- english version ISALPHA = x41-x5A,x61-x7A #---- german version
ue ae AE oe OE UE ss #ISALPHA = x41-x5A,x61-x7A,x81,x84,x8E,x94,x99,x9A,xE1

# alphanumeric characters # '0'-'9' 'A'-'Z' 'a'-'z' # #----
english version ISALNUM = x30-x39,x41-x5A,x61-x7A #---- german version ue ae AE
oe OE UE ss #ISALNUM = x30-x39,x41-x5A,x61-x7A,x81,x84,x8E,x94,x99,x9A,xE1

# digit characters # '0'-'9' ISDIGIT = x30-x39

# hexadecimal digit characters # '0'-'9' 'A'-'F' 'a'-'f' ISXDIGIT = x30-x39,x41-x46,x61-x66

# lowercase characters # 'a'-'z' # #---- english version
ISLOWER = x61-x7A #---- german version ue ae oe ss #ISLOWER = x61-x7A,x81,x84,x94,xE1

```

```

# uppercase characters # 'A'-'Z' # #---- english version
ISUPPER = x41-x5A #---- german version AE OE UE #ISUPPER = x41-x5A,x8E,x99,x9A

# control characters # ISCNTRL = x00-x1F,xFF

# printable characters # ISPRINT = x20-x7E,x80-xFE

# special characters # ISPUNCT = x21-x2F,x3A-x40,x5B-x60,x7B-x7E,x9C,x9E,x9F,xA6-xB4,xB8-xC5
ISPUNCT-1 = xC8-xCF,xD5,xD9-xDD,xDF,xE6,xEE-xFE

# graphical
characters # ISGRAPH = x21-x7E,x80-xFE

# spacing characters
# ISSPACE = x09-x0D,x20

[IBMCP_850-END] #-----

#-----
[USASCII] # valid for USASCII as internal character set # ASCII characters ISASCII
= x00-x7F

# alphabetical characters # 'A'-'Z' 'a'-'z'
# ISALPHA = x41-x5A,x61-x7A

# alphanumeric characters
# '0'-'9' 'A'-'Z' 'a'-'z' # ISALNUM = x30-x39,x41-x5A,x61-x7A

# digit characters # '0'-'9' ISDIGIT = x30-x39

# hexadecimal
digit characters # '0'-'9' 'A'-'F' 'a'-'f' ISXDIGIT = x30-x39,x41-x46,x61-x66

# lowercase characters # 'a'-'z' # ISLOWER = x61-x7A

# uppercase characters # 'A'-'Z' # ISUPPER = x41-x5A

# control characters # ISCNTRL = x00-x1F,x7F

# printable characters # ISPRINT = x20-x7E

# special characters # ISPUNCT = x21-x2F,x3A-x40,x5B-x60,x7B-x7E

# graphical characters # ISGRAPH = x21-x7E

# spacing
characters # ISSPACE = x09-x0D,x20

[USASCII-END] #-----

[CHARACTER-CLASSIFICATION-END] #*****

```

# Graphics Interface

This section describes how to generate Microsoft Excel graphics from Natural programs on Windows and covers the following topics:

- Generating Excel Graphics from Natural
  - CALL 'GRAPHICS'
  - CALL 'DRAW'
  - CALL 'PLOT'
- 

## Generating Excel Graphics from Natural

### General Information

To generate Microsoft Excel graphics from Natural programs on Windows and Windows NT, you use the special programs DRAW and PLOT, which are invoked with a CALL statement. In addition the program GRAPHICS, which is also invoked with a CALL statement, establishes the communication context for DRAW and PLOT.

The parameters specified with the CALL statements are passed to Microsoft Excel. The order of the parameters is very important: when using these programs, adhere strictly to the syntax that is described in this chapter.

The individual CALL statements are described below.

### Prerequisites for Generating Excel Graphics

You can only use these CALL statements to generate Excel graphics if Excel is installed on your workstation and the Add-In natgraph.xla stored in:

**Program Files\Software AG\Natural\411\Natural\Samples\sysexngr\excel**  
is inserted in Excel.

In addition, the environment variable "natuser" must be defined to "natgraph".

To run the examples in library "sysexngr" the SAG demo database (DB 12) must be started.

### Further Examples

In addition to the example shown in this chapter, further example programs are provided in library SYSEXNGR; their names begin with "NGREX".

## CALL 'GRAPHICS'

```
CALL 'GRAPHICS' { 'ON'
                  'OFF' }
```

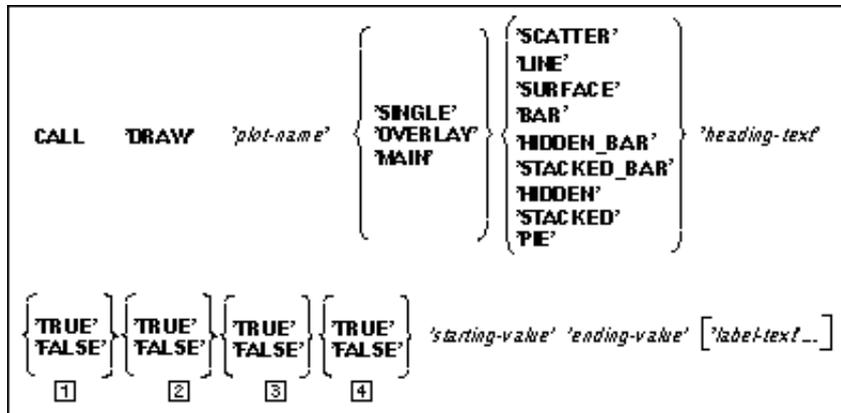
With this statement, you establish a communication context to Microsoft Excel for the statements CALL 'PLOT' and CALL 'DRAW'.

A CALL 'GRAPHICS' 'ON' statement establishes the context. This statement must be placed before the CALL 'PLOT' statement(s). The output of CALL 'PLOT' statements is available for processing by subsequent CALL 'DRAW' statements until the context is closed by a CALL 'GRAPHICS' 'OFF' statement.

### Example:

```
*** Natural GRAPHICS EXAMPLE
NGREX01 *** * Plot out two sets of data DEFINE DATA LOCAL 1 EMPL-VIEW VIEW OF
EMPLOYEES 2 BIRTH 2 SALARY (1) 1 #BIRTH-START (D) 1 #BIRTH-ALL (A8) 1 REDEFINE
#BIRTH-ALL 2 #BIRTH (N8) 1 #TODAY-ALL (A8) 1 REDEFINE #TODAY-ALL 2 #TODAY (N8)
1 #AGE (N3) 1 #SALARY (N9) END-DEFINE * CALL 'GRAPHICS' 'ON' * MOVE EDITED *DATX
(EM=YYYYMMDD) TO #TODAY-ALL MOVE EDITED '31121950' TO #BIRTH-START (EM=DDMMYYYY)
* READ EMPL-VIEW BY BIRTH STARTING From #BIRTH-START MOVE EDITED BIRTH (EM=YYYYMMDD)
TO #BIRTH-ALL #AGE:= (#TODAY-#BIRTH) / 10000 #SALARY:= SALARY (1) * CALL 'PLOT'
'AGE-SALARY' 'Fixed/Salary' 'AVER' #SALARY 'Age' #AGE CALL 'PLOT' 'AGE-NUMBER'
'Number' 'COUNT' #AGE 'Age' #AGE * END-READ * * Draw out the two sets of data
using different types of chart CALL 'DRAW' 'AGE-NUMBER' 'SINGLE' 'LINE'
*** NGR-100-010 NGREX01 PLOT 1 *** FALSE FALSE TRUE TRUE 0 0 CALL 'DRAW' 'AGE-SALARY'
'SINGLE' 'SURFACE' 'Fixed/Salary by Age' FALSE FALSE TRUE TRUE 0 0 *
CALL 'GRAPHICS' 'OFF' END
```

## CALL 'DRAW'



With the CALL 'DRAW' statement you define how data are to be output graphically.

A chart is drawn on the basis of the data accumulated in a previous CALL 'PLOT' statement. You reference a CALL 'PLOT' statement by specifying in the CALL 'DRAW' statement the same unique *plot-name* you specified with the respective CALL 'PLOT' statement.

If you wish the same data to be displayed in different charts, you use multiple CALL 'DRAW' statements which refer to the same CALL 'PLOT' statement. However, in one CALL 'DRAW' statement you can use the data from one CALL 'PLOT' statement only, that is, you cannot refer to more than one CALL 'PLOT' statement.

**Note:** If a program contains several CALL 'DRAW' statements, you can use the function "Leave Chart and Continue" from the Microsoft Excel menu "NatGraph" to "jump" from one chart to the next.

### Operands of the CALL 'DRAW' Statement

The operands must be specified in exactly the order they are presented here (see also the syntax diagram above).

#### 'plot-name'

With the *plot-name*, you refer to a specific CALL 'PLOT' statement. The *plot-name* you specify (enclosed in apostrophes) with the CALL 'DRAW' statement must be identical to the one specified in the corresponding CALL 'PLOT' statement. In this way, you identify the data which are to be used to produce the chart. You can also use a variable for *plot-name*.

**Mode of Charts - 'SINGLE', 'OVERLAY', 'MAIN'**

With this operand, you specify the mode in which a chart is to be drawn. The value you specify must be enclosed in apostrophes. The following values are available:

<b>'SINGLE'</b>	A single (separate) chart is drawn.  If the program contains multiple CALL 'DRAW' statements containing a 'SINGLE' operand, the resulting charts are not overlaid, but are produced one after the other as separate charts.
<b>'OVERLAY'</b>	Multiple charts are combined into one chart with one overlaying the other (only applies when the program contains multiple CALL 'DRAW' statements and a previous CALL 'DRAW' statement contains the 'MAIN' operand, see below).
<b>'MAIN'</b>	When you have more than one CALL 'DRAW' statement and you wish to overlay the charts, use the 'MAIN' mode for the CALL 'DRAW' statement to indicate that the chart that results from this statement is to be the "main" chart, which is then overlaid with the chart produced by the CALL 'DRAW' statement with the mode 'OVERLAY' (see above).

**Example of 'MAIN' and 'OVERLAY':**

```
... * Draw out the two sets of data using different types
of chart CALL 'DRAW' 'SUB2' 'MAIN' 'BAR' '** NGR-100-010 Example: NGREX13
**' FALSE FALSE TRUE TRUE 0 60 CALL 'DRAW' 'SUB1' 'OVERLAY' 'LINE' ' ' FALSE FALSE
TRUE TRUE 0 0 ...
```

## Types of Charts

Nine different types of charts are available with Natural.

With this operand, you specify the type of chart to be drawn. The value you specify must be enclosed in apostrophes. The following values are available:

- 'SCATTER'
- 'LINE'
- 'SURFACE'
- 'BAR'
- 'HIDDEN\_BAR'
- 'HIDDEN'
- 'STACKED\_BAR'
- 'STACKED'
- 'PIE'

### SCATTER

In a scatter plot, the data are displayed as discrete points.

### LINE

In a line graph, the data are displayed as points connected by a line.

### SURFACE

In a surface chart, the data are displayed as in a line graph, and in addition the space between the x-axis and the line of coordinates is shaded.

### BAR

In a bar chart, the data are displayed as vertical bars.

### HIDDEN\_BAR

In a hidden bar chart, the data are displayed as vertical bars; however, whereas a normal bar chart displays the bars for each x-value next to each other, in a hidden bar chart they are displayed one "hidden" behind the other. This type of chart is only applicable if you have more than one set of *y-values* for each *x-value*.

**HIDDEN - no bar**

In a hidden chart, the data are displayed as points joined by a line. The lines are displayed one above the other.

**STACKED\_BAR**

In a stacked bar chart, the data are displayed as vertical bars; however, whereas a normal bar chart displays the bars for each x-value next to each other, in a stacked bar chart they are displayed one "stacked" on top of the other.

This type of chart is only applicable if you have more than one set of *y-values* for each *x-value*.

**STACKED - no bar**

In a stacked chart, the data are displayed as points joined by a line. The lines are displayed one above the other.

**PIE**

In a pie chart, the data are displayed as segments of a circle.

**'heading-text'**

Here you can specify a heading for a chart. The *heading-text* you specify (enclosed in apostrophes) is output centered above the chart.

If you wish to have a heading which occupies multiple lines, you can use a semicolon (;) within the *heading-text* to cause a line advance: the part of the *heading-text* which follows the semicolon will then be output on the next line.

**Example of Specifying a Heading that Occupies Multiple Lines:**

```
.. CALL
'DRAW' 'AGE-NUMBER' 'SINGLE' 'LINE' '** NGR-100-01 NGREX01 PLOT 1 **;' - 'Example
of Line Chart' FALSE FALSE TRUE TRUE 0 0...
```

The heading can also be defined by a variable in the program.

**Example of Specifying a Heading via a Variable:**

```
... CALL
'DRAW' 'AGE-NUMBER' 'SINGLE' 'LINE' #HEADING FALSE FALSE TRUE TRUE 0 0
...
```

If you wish a chart to be untitled, you specify a blank enclosed in apostrophes ( ' ') as the *heading-text*.

**Example of Specifying a Blank Heading:**

```
... CALL
'DRAW' 'AGE-NUMBER' 'SINGLE' 'LINE' ' ' FALSE FALSE TRUE TRUE 0 0 ...
```

**1. PERCENT Operand**

This operand only applies to pie charts. For all other types of chart, set this operand to 'FALSE'.

The PERCENT operand is used for pie charts where the data to be displayed are percentage figures.

'TRUE'	PERCENT check is carried out.
'FALSE'	PERCENT check is not carried out.

When the PERCENT operand is set to "TRUE" the total for all sectors of the pie must not exceed 100; if it does, the PERCENT operand will be ignored and an appropriate error message be issued at runtime. If the total is less than 100%, an incomplete circle will be drawn.

If the PERCENT operand is set to "FALSE", Natural assumes that the data to be displayed are absolute figures. The sum of the figures is assumed to be 100%, that is, a full circle will be drawn and the size of each pie segment reflects the respective value in relation to the sum of values.

**2. VALUES Operand**

This operand only applies to pie charts and bar charts (normal, hidden and stacked ones). For all other types of chart, set this operand to 'FALSE'.

'TRUE'	VALUES are displayed.
'FALSE'	VALUES are not displayed.

**VALUES Operand for a Bar Chart**

When this operand is specified (TRUE) for a bar chart, the y-axis values are displayed above the individual bars.

### VALUES Operand for a Pie Chart

When this operand is specified (TRUE) for a pie chart, figures indicating the size of each sector of the pie chart are displayed. The displayed size values always reflect the *relative sizes* of pie segments in relation to the total pie.

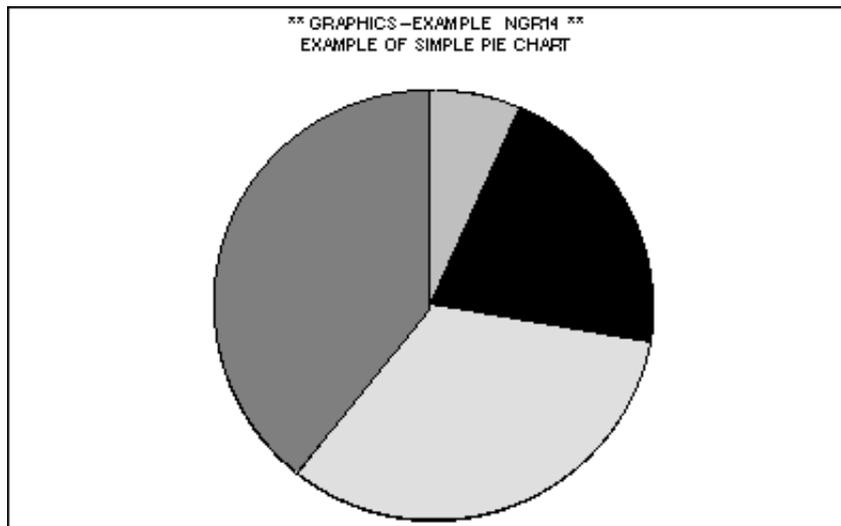
### Interaction of VALUES Operand and PERCENT Operand

If the PERCENT operand is set to "FALSE", the values displayed will always add up to 100%, even if the total of all sectors is less than 100 in absolute figures. A full circle is drawn, and the values displayed do not show the actual absolute values for each segment but are in fact percentage figures indicating the relative size of each segment in relation to the total of the pie.

If the PERCENT operand is set to "TRUE", the values displayed still indicate the relative size of each segment; the values displayed are, however, identical to the actual values of the dependent variable because the actual values are (assumed to be) percentage figures.

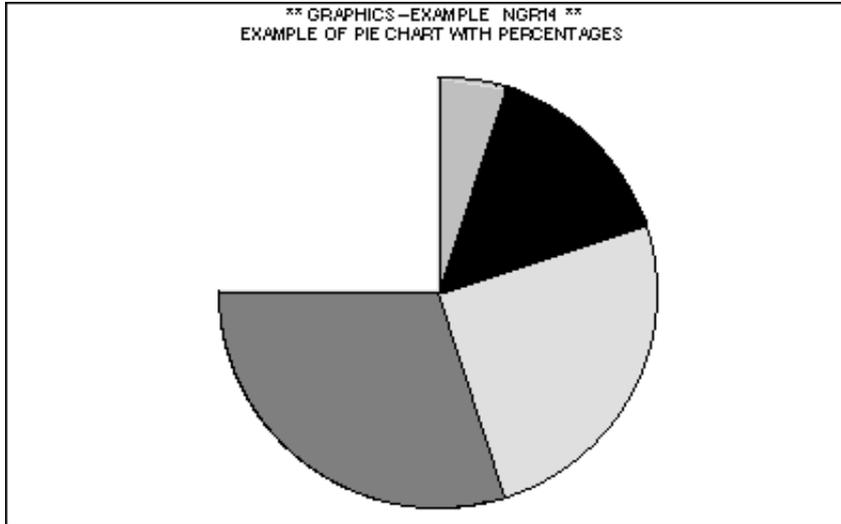
### Pie Chart with PERCENT and VALUES Operands set to 'FALSE'

The values accumulated for the dependent variable are 5, 15, 25 and 30. Since these are absolute figures and not percentage figures, the PERCENT operand is not used. Natural considers the sum of the values to be 100% and therefore draws a complete circle with the size of each segment reflecting the respective value in relation to the sum of the values.



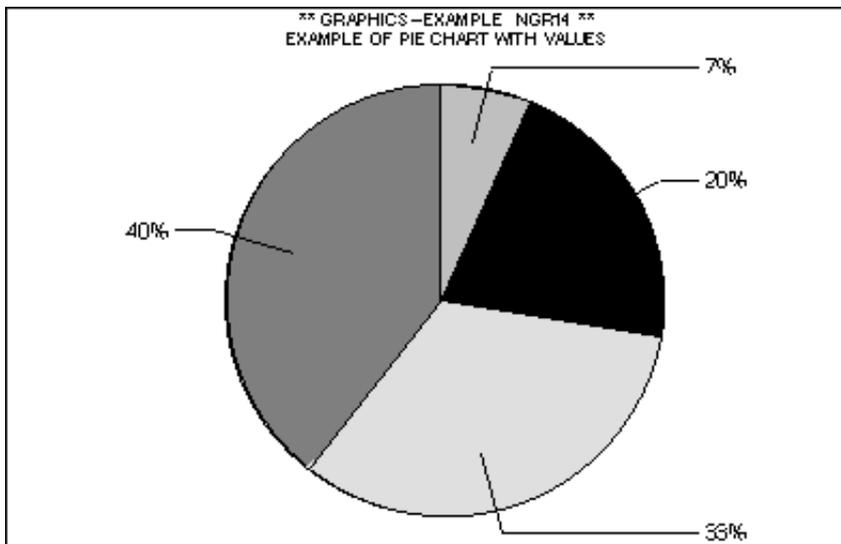
**Pie Chart with PERCENT Operand set to 'TRUE'**

This chart uses a PERCENT operand because the values accumulated for the dependent variable are percentage figures. As these figures do not add up to 100%, an incomplete circle is drawn.



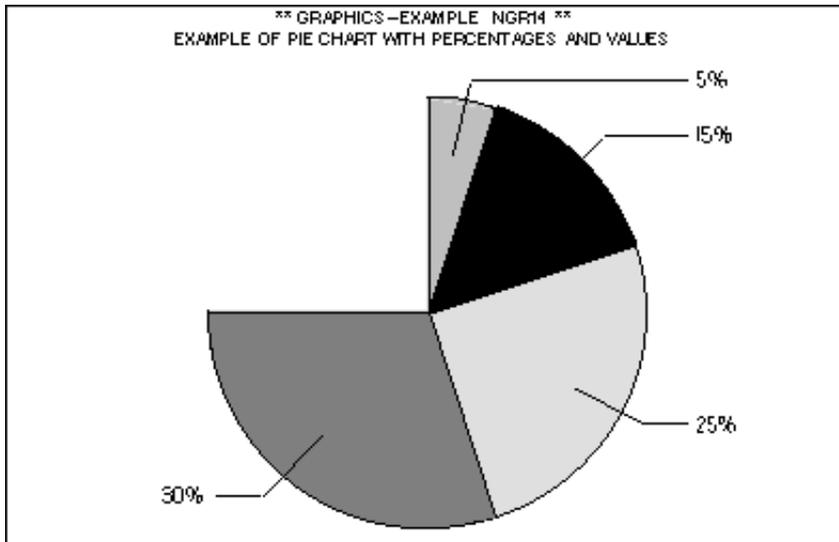
**Pie Chart with VALUES Operand set to 'TRUE'**

The values accumulated for the dependent variable are 5, 15, 25 and 30. Since these are absolute figures and not percentage figures, the PERCENT operand is not used. Natural considers the sum of the values to be 100% and therefore draws a complete circle with the size of each segment reflecting the respective value in relation to the sum of the values. The VALUES operand writes the segment sizes next to the segments.



### Pie Chart with PERCENT and VALUES Operands set to 'TRUE'

This chart uses a PERCENT operand because the values accumulated for the dependent variable are percentage figures. The VALUES operand writes the segment sizes - which in this case correspond to the actual values accumulated, as these are percentages - next to the segments.



### 3. Operand for the X-Axis

'TRUE'	X-axis is displayed.
'FALSE'	X-axis is suppressed.

### 4. Operand for the Y-Axis

'TRUE'	Y-axis is displayed.
'FALSE'	Y-axis is suppressed.

### 'starting-value' & 'ending-value' - RANGE Operand

If you specify "0" for both *starting-value* and *ending-value*, the value ranges of the axes are set dynamically according to the values accumulated by the CALL 'PLOT' statement.

If you wish the y-axis to cover a specific range of values, you can specify a *starting-value* and an *ending-value* for the desired range.

### 'label-text'

This operand is optional. The *label-text* is used to provide labels for the scale marks of a chart's x-axis.

You specify the labels as individual character strings, each enclosed in apostrophes and separated from each other by a blank:

**'label-text1' 'label-text2' 'label-text3'**

To achieve proper scale mark labeling, specify a *label-text* for each x-value. If you do not wish to have a label for a scale mark, specify an empty string (' ') as the label.

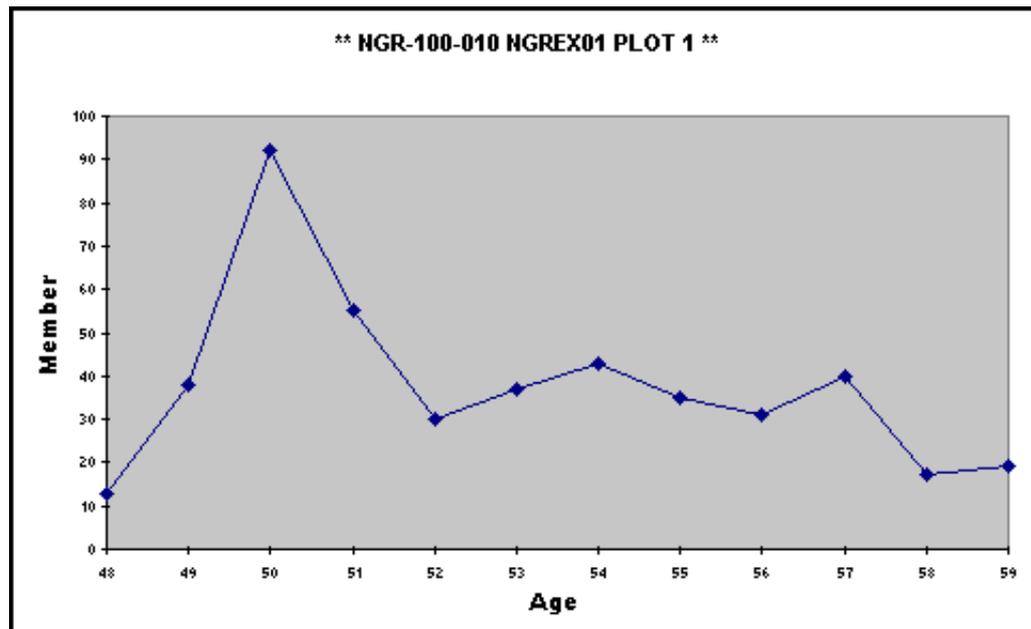
Instead of specifying strings of alphanumeric text constants as *label-text*, you can also specify a single variable, and then assign the labels as values to this variable. The value of the variable will be one string of all labels to be used with the individual labels separated from each other by commas.

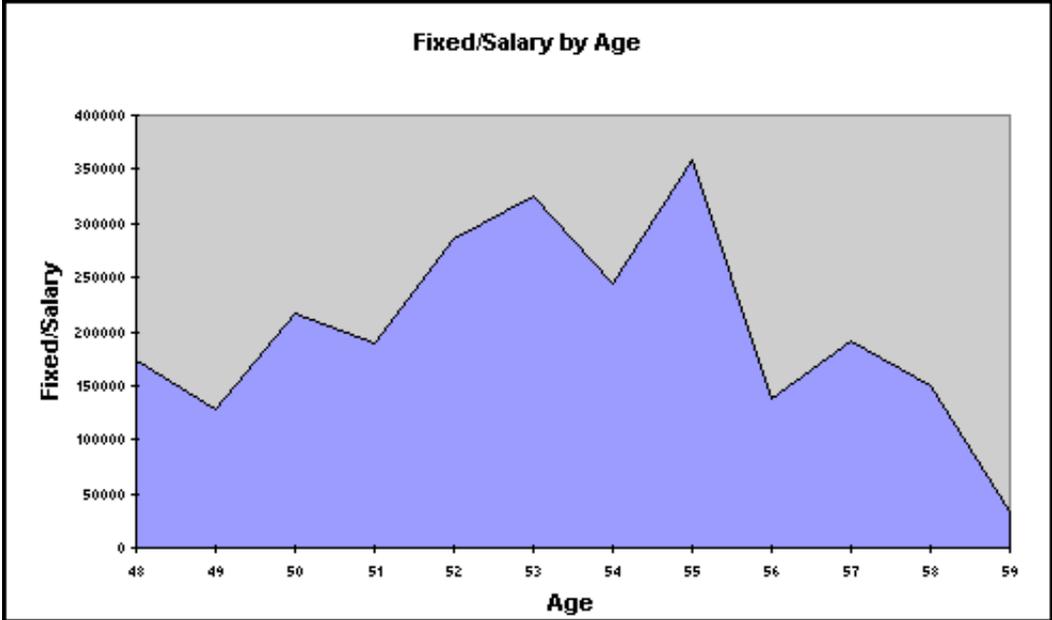
If you omit the *label-text* operand, the scale marks are labelled automatically according to the x-axis values.

**Example of CALL 'DRAW' Statements:**

```

*** Natural GRAPHICS EXAMPLE
NGREX01 *** * Plot out two sets of data DEFINE DATA LOCAL 1 EMPL-VIEW VIEW OF
EMPLOYEES 2 BIRTH 2 SALARY (1) 1 #BIRTH-START (D) 1 #BIRTH-ALL (A8) 1 REDEFINE
#BIRTH-ALL 2 #BIRTH (N8) 1 #TODAY-ALL (A8) 1 REDEFINE #TODAY-ALL 2 #TODAY (N8)
1 #AGE (N3) 1 #SALARY (N9) END-DEFINE * CALL 'GRAPHICS' 'ON' * MOVE EDITED *DATX
(EM=YYYYMMDD) TO #TODAY-ALL MOVE EDITED '31121950' TO #BIRTH-START (EM=DDMMYYYY)
* READ EMPL-VIEW BY BIRTH STARTING From #BIRTH-START MOVE EDITED BIRTH (EM=YYYYMMDD)
TO #BIRTH-ALL #AGE:= (#TODAY-#BIRTH) / 10000 #SALARY:= SALARY (1) * CALL 'PLOT'
'AGE-SALARY' 'Fixed/Salary' 'AVER' #SALARY 'Age' #AGE CALL 'PLOT' 'AGE-NUMBER'
'Number' 'COUNT' #AGE 'Age' #AGE * END-READ * * Draw out the two sets of data
using different types of chart CALL 'DRAW' 'AGE-NUMBER' 'SINGLE' 'LINE'
*** NGR-100-010 NGREX01 PLOT 1 *** FALSE FALSE TRUE TRUE 0 0 CALL 'DRAW' 'AGE-SALARY'
'SINGLE' 'SURFACE' 'Fixed/Salary by Age' FALSE FALSE TRUE TRUE 0 0 *
CALL 'GRAPHICS' 'OFF' END
    
```





## CALL 'PLOT'

With the CALL 'DRAW' statement you define how data are to be output graphically. With the CALL 'PLOT' statement you specify which data are to be accumulated to produce the graph, and you also determine the point in the program at which these data are to be captured.

The data to be collected are specified by an independent variable (the *x-value*) and a dependent variable (the *y-value*).

A PLOT statement is executed at the point at which it is encountered in the program. The data point represented by the current values of the dependent and independent variables is saved so that it can be used when an associated CALL 'DRAW' statement is executed.

### 'plot-name'

*plot-name* is a user-defined name by which you identify the PLOT data in the program, and is used to refer to the accumulated data in subsequent CALL 'DRAW' statements.

*plot-name* must be enclosed in apostrophes, it must be unique within a program, it may be 1 to 32 characters long, and it must conform to the naming conventions for user-defined variables (as described in the Natural Reference documentation).

### 'y-text'

This text (which must be enclosed in apostrophes) is used as a title for the y-axis.

If you wish the y-axis to be untitled, specify a blank as *y-text* (in apostrophes): ' '.

### 'MAX', 'MIN', 'NMIN', 'COUNT', 'NCOUNT', 'AVER', 'NAVER', 'SUM', 'TOTAL'

The data values for the y-variable are accumulated to evaluate one of the following Natural system functions: MAX, MIN, NMIN, COUNT, NCOUNT, AVER, NAVER, SUM, TOTAL. (For details on these system functions, see the Natural Reference documentation.)

### *y-value*

*y-value* operand defines the value to be used for the dependent variable.

### 'x-text'

This text (which must be enclosed in apostrophes) is used as a title for the x-axis.

If you wish the x-axis to be untitled, specify a blank as *x-text* (in apostrophes): ' '.

### *x-value*

*x-value* defines the value to be used for the independent variable.

### 'ciomp-text'

This operand is optional. If you specify *ciomp-text* (enclosed in apostrophes) before the *comp-value*, this *ciomp-text* is used to generate a default chart heading if no *heading-text* is specified in the CALL 'DRAW' statement.

If you do not wish a text to be used, specify a blank as *ciomp-text* (in apostrophes): ' '.

### ***comp-value***

This operand is optional. The *comp-value* is used to specify a component by which the y-values are subdivided. This allows you to draw multiple sets of data on the same chart: for each value of the variable *comp-value* a separate set of y-values per x-value is determined, and all sets are displayed on one chart.

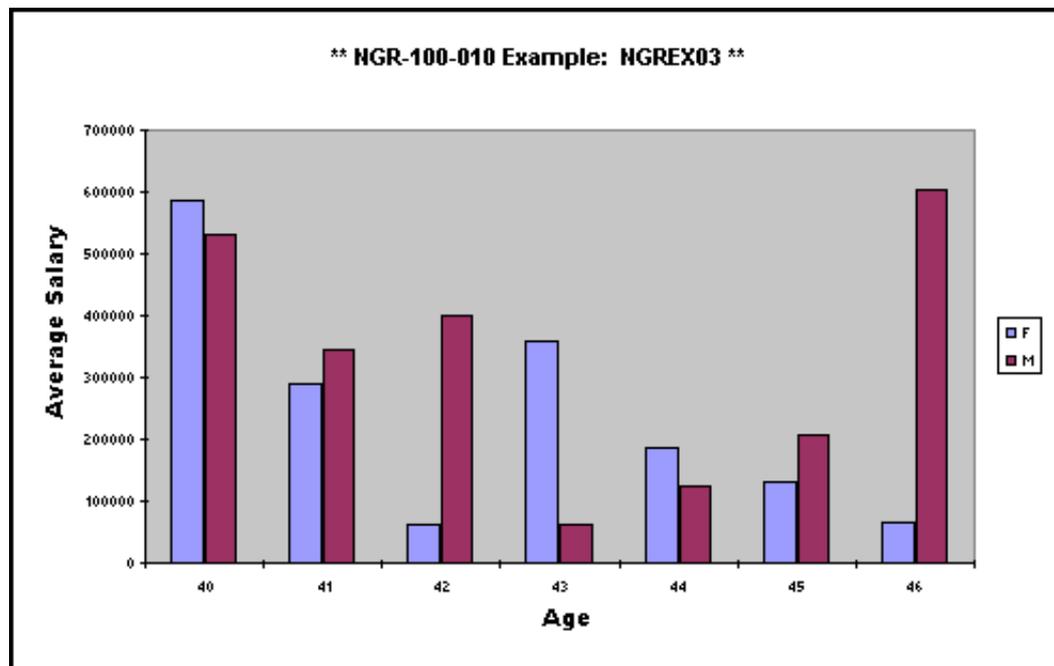
To uniquely identify each component, a legend is automatically generated using the values of the component variable and is displayed to the right of the graph.

**Example of CALL 'PLOT' Statement (with comp-value):**

```

*** Natural GRAPHICS EXAMPLE NGRGX03 *** * First gather
the plot data DEFINE DATA LOCAL 1 EMPL-VIEW VIEW OF EMPLOYEES 2 BIRTH 2 SALARY
8(1) 2 SEX 1 #TODAY-ALL (A8) 1 REDEFINE #TODAY-ALL 2 #TODAY (N8) 1 #BIRTH-ALL
(A8) 1 REDEFINE #BIRTH-ALL 2 #BIRTH 1 #BIRTH-FROM (D) 1 #BIRTH-TO (D) 1 #AGE (N3)
1 #SALARY (N9) END-DEFINE * CALL 'GRAPHICS' 'ON' * MOVE EDITED *DATX (EM=YYYYMMDD)
TO #TODAY-ALL MOVE EDITED '16021953' TO #BIRTH-FROM (EM=DDMMYYYY) MOVE EDITED
'16021959' TO 'BIRTH-TO (EM=DDMMYYYY) * FIND EMPL-VIEW WITH BIRTH = #BIRTH-FROM
THRU #BIRTH-TO SORTED BY BIRTH DESC MOVE EDITED BIRTH (EM=YYYYMMDD) TO #BIRTH-ALL
#AGE:=(#TODAY-#BIRTH) / 10000 #SALARY:=SALARY (1) CALL 'PLOT' 'SALARY-SEX-AGE'
'Average Salary' 'AVER' #SALARY 'Age' #AGE ' ' SEX END-FIND * * Now draw
it out correctly labeling each bar CALL 'DRAW' 'SALARY-SEX-AGE' 'SINGLE' 'BAR'
'*** NGR-100-010 Example: NGRGX03 ***' FALSE FALSE TRUE TRUE 0 0 * CALL 'GRAPHICS'
'OFF' * END
    
```

In this example, the *comp-value* is SEX: the chart shows the number of persons of a certain age separately for each value of the field SEX (M = male, F = female).



# Natural and Entire Access

This section describes how to prepare Natural for its use with Software AG's Entire Access under Windows NT and covers the following topics:

- Purpose of Entire Access
- Entire Access under Windows and Windows NT

**Note:**

Entire Access Version 2.1 is required for Natural for Windows Version 4.1.

---

## Purpose of Entire Access

Via Entire Access, Natural applications can transparently access multiple relational database management systems (RDBMSs), in both client-server and single-platform environments.

Entire Access has been designed to provide a common SQL-eligible application programming interface (API) for both local and remote database access. It represents a client-server solution for Software AG's database systems Adabas SQL Server and Adabas, as well as for various third-party products, for example, ODBC-compliant databases under Windows.

With Entire Access, a Natural user can access data in a relational database. In general, there is no difference between using Natural with Entire Access and using Natural with Adabas. Entire Access supports Natural DML and SQL statements which can both be used from within *one* Natural program accessing *multiple* database systems.

Natural converts Natural DML and SQL statements within Natural applications into calls to Entire Access, which performs the required data format translations and transfers these requests to the underlying database system via the appropriate database driver.

All operations requiring interaction with the database are performed by Entire Access. It supports local and/or remote databases and contains a series of database drivers, one for each supported RDBMS, so that multiple heterogeneous RDBMSs can be accessed concurrently from within the same Natural application.

For remote access, the transport media for SQL data are provided by Entire Net-Work, which establishes the communication link, or by TCP/IP.

**Note:**

For further information on Entire Access, refer to the Entire Access documentation.

## Entire Access under Windows and Windows NT

- Modifying the Global Configuration File
- Connecting Databases
- Making Entire Access Operational
- Accessing Database Tables - the Natural DDM

Under Windows NT, Entire Access can be used to access either local ODBC-compliant databases and/or remote databases on Windows NT and UNIX hosts.

**Note:**

If you want to access remote databases, you also have to install and use the Entire Access server component on the host platform; for further information and prerequisites, see the Installation chapter of the Entire Access documentation.

To be able to use Entire Access, you have to modify your global configuration file, establish a connection to the database(s) to be used, create database tables with the appropriate database tools, and generate a Natural data definition module (DDM) for each database table to be accessed by a Natural program.

## Modifying the Global Configuration File

To be able to work with Entire Access, you have to modify the global configuration file (default name: NATCONF.CFG) as follows; see also Configuration Files.

1. Invoke the Natural Configuration Utility.  
The Natural Parameter Setting window appears with the name of the parameter file currently active or edited displayed in the title bar.
2. Select the path Global Configuration File / DBMS Assignments in the tree view. A dialog appears in which you have to specify the DBID parameter.  
For all SQL databases that can be accessed via Entire Access, you always specify the same DBMS Type: "SQL".  
Via the DBMS parameter, you must then establish the connection to the actual database as described in the following section.  
**Note:**  
For each database you want to be accessed by Entire Access, you must repeat the specification of the corresponding DBID and DBMS parameter.
3. Save the updated global configuration file.

## Connecting Databases

The connection to the database(s) you want to work with must be established only once for each database. Subsequently, you merely have to log on to Natural and the database connection(s) will be established automatically.

Entire Access can support up to 7 different database drivers. Since the database drivers are reentrant, you can establish multiple separate connections to the same driver via a so-called database connect string.

### Database Connect String

The database connect string represents the DBMS parameter you or your system administrator must specify with the Natural Configuration Utility during the modification of your global configuration file (see the previous page). It consists of the following syntax:

***dbms:[db-name][@server-number:host-name!driver]***

#### ***dbms***

*dbms* is mandatory and specifies any supported driver; currently the following drivers are supported under Windows:

#### For ODBC-compliant databases:

"ODBC" for local and remote ODBC-compliant data sources.

#### For remote database access:

For remote access, you have to connect the Entire Access network component by specifying "NET"; see the Entire Access documentation for further information.

***db-name***

For ODBC-compliant databases:

*db-name* is mandatory and represents the name of your data source as it was defined to the ODBC Administrator.

**Note:**

The name of the data source is case-sensitive and must not contain any blanks.

For remote databases:

*db-name* is the name of your database if your database system requires a name; if not, it must be omitted. See the Entire Access documentation for further information.

***server-number***

*server-number* is required for remote access only; see the Entire Access documentation for further information.

***host-name***

*host-name* is required for remote access only; see the Entire Access documentation for further information.

***driver***

*driver* specifies the database driver to be used; see the Entire Access documentation for further information.

**Sample ODBC Connect String**

To connect to an ODBC-compliant database with a data source called "sample", specify:

**ODBC:sample**

**Making Entire Access Operational**

Now Entire Access is connected to the database(s) to be accessed. The user ID of the default administrator is "sag".

To make Entire Access operational:

1. Start your local ODBC-compliant database(s) in the usual way; for remote databases, refer to the Entire Access documentation.
2. Start Natural; Entire Access can now be used and you can start a Natural session as usual.

**Accessing Database Tables - the Natural DDM**

To be able to work with Entire Access, you must create database tables or views using the appropriate tools provided by your database system(s).

To be able to access a relational database table or view with a Natural program, you must then generate a Natural DDM for the defined database table or view.

The DDM name must match the name of the corresponding table or view. Thus, you can create only one DDM for each table or view. It is recommended that the DDM name include the name of the table creator (owner) separated by a hyphen. The DBID in the DDM must correspond to the DBID assigned by the Natural Configuration Utility.

Once a DDM has been defined for a relational table or view, it is possible to access the data stored in this table or view by using a Natural program.

Entire Access translates the statements of a Natural program into SQL statements. It automatically provides for the preparation and execution of each statement.

In addition to the Natural DML statements, Natural provides SQL statements, which are described in the Natural Statements documentation; for further information, see also the Entire Access documentation.

For a description of how to create DDMs from relational database tables, refer to the section **Adding DDMs** in your Natural User's Guide for Windows.